TOSHIBA

TOSHIBA Original CMOS 32-Bit Microcontroller

TLCS-900/H1 Series

TMP92CF29AFG

TOSHIBA CORPORATION

Semiconductor Company

Preface

Thank you very much for making use of Toshiba microcomputer LSIs.
Before use this LSI, refer the section, “Note and Restrictions”.

CMOS 32-Bit Microcontroller

TMP92CF29AFG

1. Outline and Features

The TMP92CF29A is a high-speed advanced 32-bit microcontroller developed for controlling equipment which processes mass data.

The TMP92CF29AFG is housed in a 176-pin QFP package.

This production has "-777x code": x is from 0, and it shows Boot Rom version.

(1) CPU: 32-bit CPU (High-speed 900/H1 CPU)
- Compatible with TLCS-900/L1 instruction code
- 16 Mbytes of linear address space
- General-purpose register and register banks
- Micro DMA: 8channels (62.5 ns/4 bytes at f_{SYS} = 80 MHz, best case)

(2) Minimum instruction execution time : 12.5 ns (at f_{SYS} = 80 MHz)

(3) Internal RAM: 144 Kbytes (can be used for program, data and display memory)
Internal ROM: None (However, built-in 8 Kbytes ROM of special memory for Boot)

(4) External memory expansion
- Expandable up to 2.1 Gbytes (shared program/data area)
- Can simultaneously support 8-and 16-bit width external data buses
…… Dynamic data bus sizing
- Separate bus system

(5) Memory controller
- Chip select output: 4 channels
- One channel in 4 channels is enabled detailed AC enable setting

(6) 8-bit timers: 8 channels

(7) 16-bit timer/event counter: 2 channels

(8) General-purpose serial interface: 2 channels
- UART/synchronous mode: 2 channels
- IrDA ver.1.0 (115.2 kbps) selectable

(9) Serial bus interface: 1 channel
- I^2C standard mode only

(10) USB (universal serial bus) controller: 1 channel
- Full-speed (12 Mbps) (Low-speed is not supported.)
- Endpoint 0: Control 64 bytes × 1 FIFO
Endpoint 1: BULK (output) 64 bytes × 2 FIFOs
Endpoint 2: BULK (input) 64 bytes × 2 FIFOs
Endpoint 3: Interrupt (input) 8 bytes × 1 FIFO
- Descriptor RAM: 384 bytes

(11) I²S (Inter-IC Sound) interface: 1 channel

- I²S bus mode selectable (Master, transmission only)
- Data Format is supported Left/Right Justify
- 128-byte FIFO buffer (64 bytes × 2)

(12) LCD controller

- Supports monochrome, 4, 16 and 64 gray levels and 256/4096/65536 colors for STN
- Supports monochrome, 4096/65536 colors for TFT
- Supports PIP (Picture In Picture Display)
- Supports H/W Rotation function for support to various LCDM

(13) SDRAM controller:1 channel

- Supports 16-Mbit, 64-Mbit, 128-Mbit, 256-Mbit and 512-Mbit SDR (Single-data-rate) SDRAM
- Possible to execute instruction on SDRAM

(14) Timer for real-time clock (RTC)

- Based on TC8521A

(15) Key-on wakeup (Interrupt key input)

(16) 10-bit A/D converter (Built in Sample Hold circuit): 6 channels

(17) Touch screen interface

- Built-in Switch of Low-resistor, and available to reduce external components for shift change row/column

(18) Watchdog timer

(19) Melody/alarm generator

- Melody: Output of a clock 4 to 5461-Hz clock
- Alarm: Output of 8 kinds of alarm pattern
- 5 kinds of interval interrupt

(20) MMU

- Expandable up to 2.1 Gbytes (3 local area/8 bank method)
- Independent bank for each program, read data, write data, source and destination of DMAC (Odd channel/Even channel) and LCD display data

(21) Interrupts: 57 interrupts

- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 39 internal interrupts: Seven selectable priority levels
- 9 external interrupts: Seven selectable priority levels (include key interrupt)
  (8-edge selectable)

(22) DMAC function: 6 channels

- High-speed data transfer enable by controlling which convert micro DMA function and this function

(23) Input/Output ports: 98 pins (Except Data bus (16bit), Address bus (24bit) and $\overline{RD}$ pin)

(24) NAND Flash interface: 2 channels

- Direct NAND flash connection capability
- Supports SLC type and MLC type
- Supports Data Bus 8/16 bit, Page Size 512/2048 bytes
- Built-in Reed Solomon calculation circuits which enabled correct 4-address, and detect error more than 5-address

(25) SPI controller: 1 channel

- Supports SPI mode of SD card and MMC card
- Built-in FIFO buffer of 32 bytes to each Input/Output

(26) Product/Sum calculation: 1 channel

- Supports calculation $32 \times 32 + 64 = 64$ bits, $64 - 32 \times 32 = 64$ bits and $32 \times 32 - 64 = 64$ bits
- I/O method
- Supports Signed calculations

(27) Standby function

- Three Halt modes: IDLE2 (programmable), IDLE1, STOP
- Each pin status programmable for standby mode
- Built-in power supply management circuits (PMC) for leakage current provision

(28) Clock controller

- Two blocks of clock doubler (PLL) supplies 48 MHz for USB and 80 MHz for CPU from 10 MHz
- Clock gear function: Selectable high-frequency clock fc to fc/16
- Clock for Timer (fs = 32.768 kHz)

(29) Operating voltage:

- 2 power supplies (Internal power supply (1.4 to 1.6 V), External power supply (3.0 to 3.6 V)

(30) Package

- 176-pin LQFP: LQFP176-P-2020-0.40F

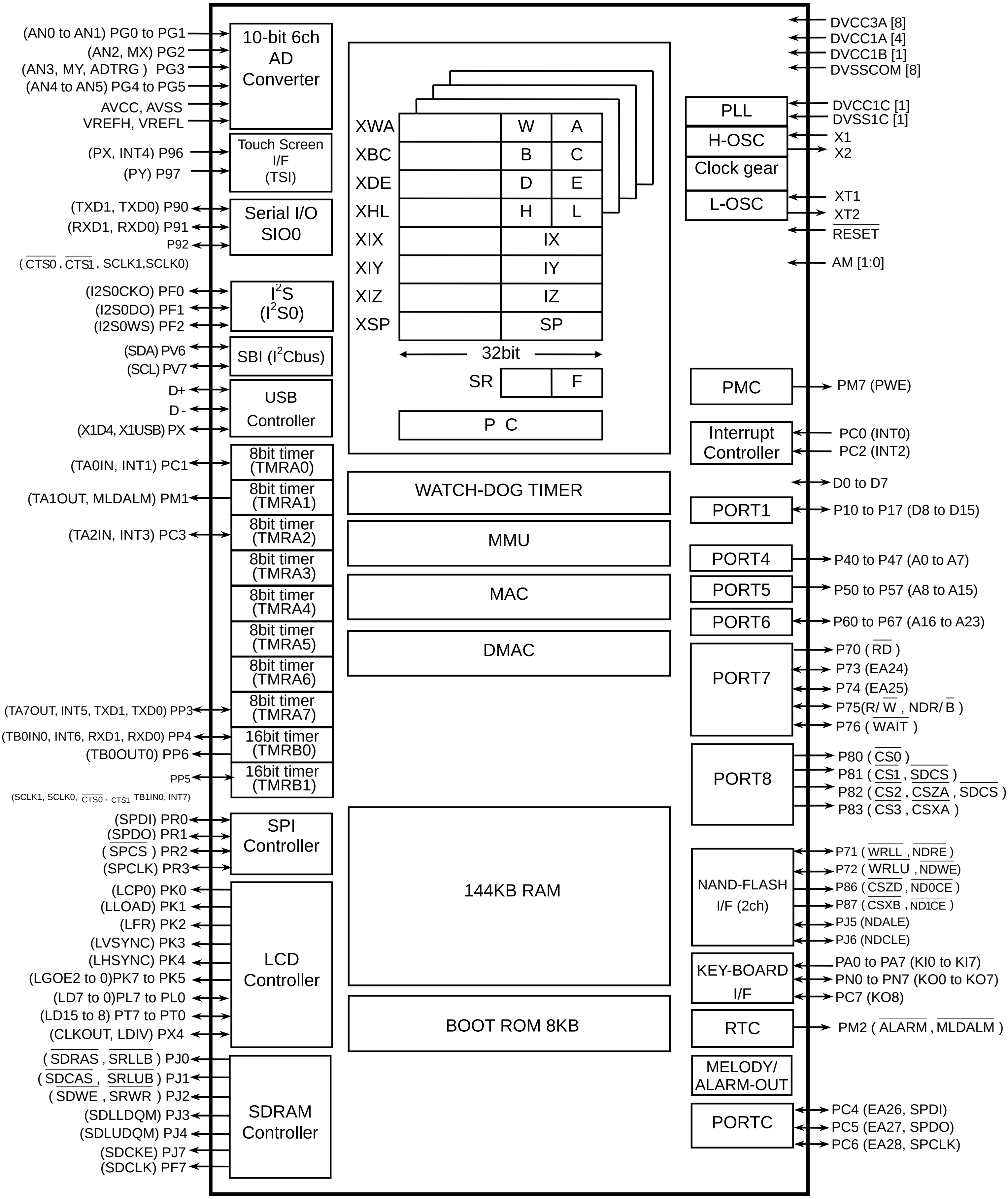


Figure 1.1 Block Diagram of TMP92CF29A

2. Pin Assignment and Pin Functions

The assignment of input/output pins for TMP92CF29A, their names and functions are as follows;

2.1 Pin Assignment Diagram (Top View)

Figure 2.1.1 shows the pin assignment of the TMP92CF29A.

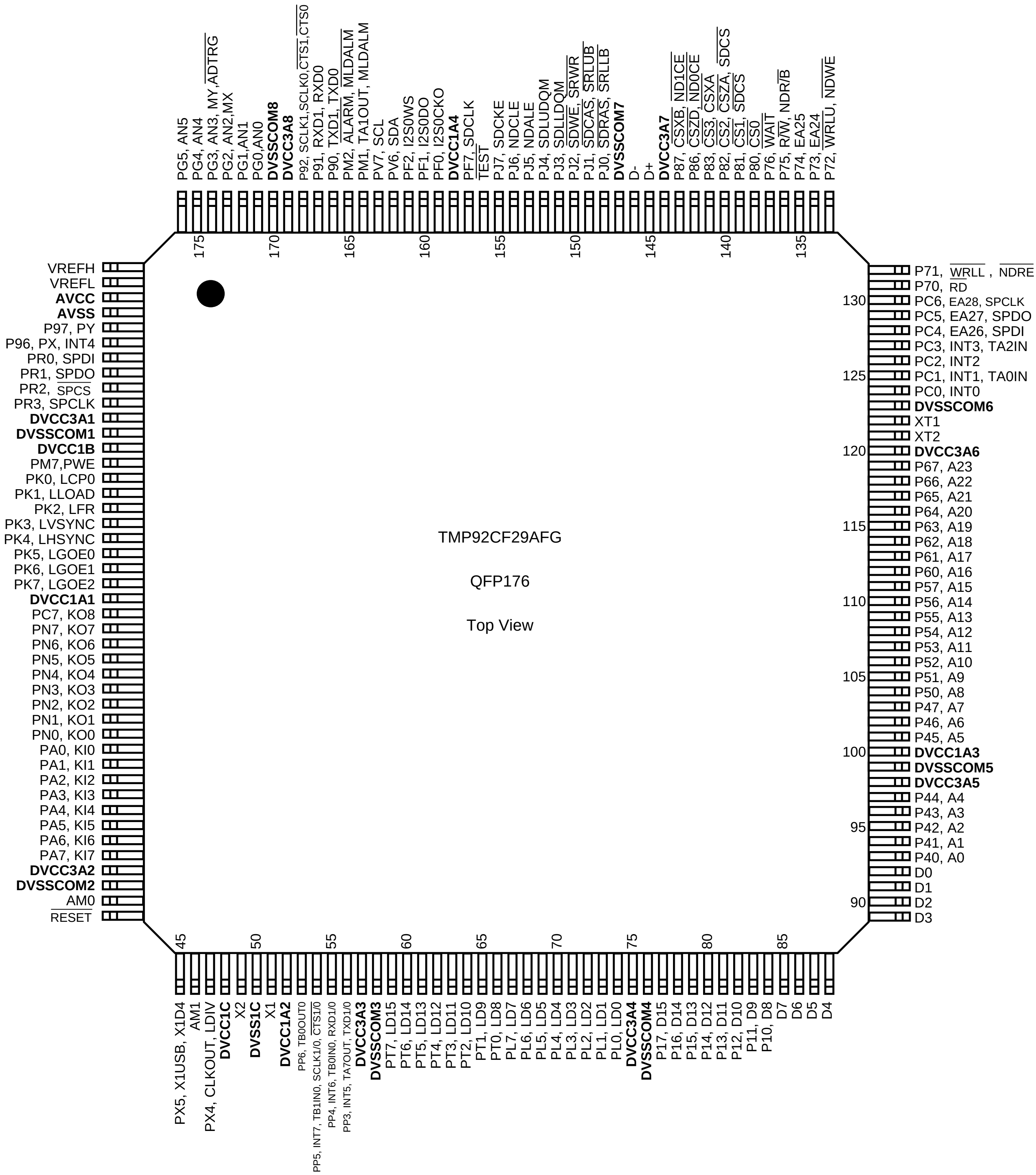


Figure 2.1.1 Pin assignment diagram (P-FBGA228)

2.2 Pin names and Functions

The names of the input/output pins and their functions are described below.

Table 2.2.1 Pin names and functions (1/6)

Pin name	Number of Pins	I/O	Functions
D0 to D7	8	I/O	Data: Data bus D0 to D7
P10 to P17 D8 to D15	8	I/O I/O	Port 1: I/O port input or output specifiable in units of bits Data: Data bus D8 to D15
P40 to P47 A0 to A7	8	Output Output	Port 4: Output port Address: Address bus A0 to A7
P50 to P57 A8 to A15	8	Output Output	Port 5: Output port Address: Address bus A8 to A15
P60 to P67 A16 to A23	8	I/O Output	Port 6: I/O port input or output specifiable in units of bits Address: Address bus A16 to A23
P70 $\overline{RD}$	1	Output Output	Port 70: Output port Read: Outputs strobe signal to read external memory
P71 $\overline{WRLL}$ $\overline{NDRE}$	1	I/O Output Output	Port 71: Output port Write: Outputs strobe signal for writing data on pins D0 to D7 NAND Flash read: Outputs strobe signal to read external NAND-Flash
P72 $\overline{WRLU}$ $\overline{NDWE}$	1	I/O Output Output	Port 72: I/O port Write: Outputs strobe signal for writing data on pins D8 to D15 NAND Flash write: Write enable for NAND Flash
P73 EA24	1	I/O Output	Port 73: I/O port Expanded address 24
P74 EA25	1	I/O Output	Port 74: I/O port Expanded address 25
P75 R/$\overline{W}$ NDR/$\overline{B}$	1	I/O Output Input	Port 75: I/O port Read/Write: “High” represents read or dummy cycle; “Low” represents write cycle NAND Flash Ready(1) / Busy(0) input
P76 $\overline{WAIT}$	1	I/O Input	Port 76: I/O port Wait: Signal used to request CPU bus wait
P80 $\overline{CS0}$	1	Output Output	Port 80: Output port Chip select 0: Outputs “Low” when address is within specified address area
P81 $\overline{CS1}$ $\overline{SDCS}$	1	Output Output Output	Port 81: Output port Chip select 1: Outputs “Low” when address is within specified address area Chip select for SDRAM: Outputs “Low” when the address is within SDRAM address area
P82 $\overline{CS2}$ $\overline{CSZA}$ $\overline{SDCS}$	1	Output Output Output Output	Port 82: Output port Chip select 2: Outputs “Low” when address is within specified address area Expanded address ZA: Outputs “Low” when address is within specified address area Chip select for SDRAM: Outputs “Low” when the address is within SDRAM address area
P83 $\overline{CS3}$ $\overline{CSXA}$	1	Output Output Output	Port 83: Output port Chip select 3: Outputs “Low” when address is within specified address area Expanded address XA: Outputs “Low” when address is within specified address area

Table 2.2.1 Pin names and functions (2/6)

Pin name	Number of Pins	I/O	Functions
P86 $\overline{\text{CSZD}}$ $\overline{\text{ND0CE}}$	1	Output Output Output	Port 86: Output port Expanded address ZD: Outputs “Low” when address is within specified address area Chip select for NAND Flash 0: Outputs “Low” when NAND Flash 0 is enable
P87 $\overline{\text{CSXB}}$ $\overline{\text{ND1CE}}$	1	Output Output Output	Port 87: Output port Expanded address XB: Outputs “Low” when address is within specified address area Chip select for NAND Flash 1: Outputs “Low” when NAND Flash 1 is enable
P90 TXD0 TXD1	1	I/O Output Output	Port 90: I/O port Transmit data for serial 0: programmable Open-drain output Transmit data for serial 1: programmable Open-drain output
P91 RXD0 RXD1	1	I/O Input Input	Port 91: I/O port (Schmitt-input) Receive data for serial 0 Receive data for serial 1
P92 SCLK0 $\overline{\text{CTS0}}$ SCLK0 $\overline{\text{CTS0}}$	1	I/O I/O Input I/O Input	Port 92: I/O port (Schmitt-input) Clock I/O for serial 0 Enable to send data for serial 0 (Clear to send) Clock I/O for serial 1 Enable to send data for serial 1 (Clear to send)
P96 INT4 PX	1	Input Input Output	Port 96: Input port (schmitt-input, with pull-up resistor) Interrupt request pin 4: Interrupt request pin with programmable rising/falling edge X-Plus: Pin connected to X+ pin for Touch Screen I/F
P97 PY	1	Input Output	Port 97: Input port (schmitt input) Y-Plus: Pin connected to Y+ pin for Touch Screen I/F
PA0 to PA7 KI0 to KI7	8	Input Input	Port A0 to A7: Input port Key input 0 to 7: Pin used for key on wake-up 0 to 7 (Schmitt-input, with pull-up resistor)
PC0 INT0	1	I/O Input	Port C0: I/O port (Schmitt-input) Interrupt request pin 0: Interrupt request pin with programmable rising/falling edge
PC1 INT1 TA0IN	1	I/O Input Input	Port C1: I/O port (Schmitt-input) Interrupt request pin 1: Interrupt request pin with programmable rising/falling edge Timer A0 input: Input pin for 8 bit timer 0
PC2 INT2	1	I/O Input	Port C2: I/O port (Schmitt-input) Interrupt request pin 2: Interrupt request pin with programmable rising/falling edge
PC3 INT3 TA2IN	1	I/O Input Input	Port C3: I/O port (Schmitt-input) Interrupt request pin 3: Interrupt request pin with programmable rising/falling edge Timer A2 input: Input pin for 8 bit timer 2
PC4 EA26 SPDI	1	I/O Output Input	Port C4: I/O port Expanded address 26 Data input pin for SD card
PC5 EA27 SPDO	1	I/O Output Output	Port C5: I/O port Expanded address 27 Data output pin for SD card
PC6 EA28 SPCLK	1	I/O Output Output	Port C6: I/O port Expanded address 28 Clock output pin for SD card
PC7 KO8	1	I/O Output	Port C7: I/O port Key output 8: Key scan strobe pin (programmable Open-drain output)

Table 2.2.1 Pin names and functions (3/6)

Pin name	Number of Pins	I/O	Functions
PF0 I2S0CKO	1	I/O Output	Port F0: I/O port Outputs clock for I^2S0
PF1 I2S0DO	1	I/O Output	Port F1: I/O port Outputs data for I^2S0
PF2 I2S0WS	1	I/O Output	Port F2: I/O port Outputs word select signal for I2S0
PF7 SDCLK	1	Output Output	Port F7: Output port Clock for SDRAM
PG0 to PG1 AN0 to AN1	2	Input Input	Port G0 to G1: Input port Analog input pin 0 to 1: Input pin for AD converter
PG2 AN2 MX	1	Input Input Output	Port G2: Input port Analog input pin 2: Input pin for AD converter X-Minus: Pin connected to X- pin for Touch Screen I/F
PG3 AN3 MY $\overline{\text{ADTRG}}$	1	Input Input Output Input	Port G3: Input port Analog input pin 3: Input pin for A/D converter Y-Minus: Pin connected to Y- pin for Touch Screen I/F A/D Trigger: Request signal for A/D start
PG4 to PG5 AN4 to AN5	2	Input Input	Port G4 to G5: Input port Analog input pin 4 to 5: Input pin for A/D converter
PJ0 $\overline{\text{SDRAS}}$ $\overline{\text{SRLLB}}$	1	Output Output Output	Port J0: Output port Outputs strobe signal for SDRAM row address Data enable signal for D0 to D7 for SRAM
PJ1 $\overline{\text{SDCAS}}$ $\overline{\text{SRLUB}}$	1	Output Output Output	Port J1: Output port Outputs strobe signal for SDRAM column address Data enable signal for D8 to D15 for SRAM
PJ2 $\overline{\text{SDWE}}$ $\overline{\text{SRWR}}$	1	Output Output Output	Port J2: Output port Outputs write enable signal for SDRAM Write enable for SRAM: Outputs strobe signal to write data
PJ3 SDLLDQM	1	Output Output	Port J3: Output port Data enable signal for D0 to D7 for SDRAM
PJ4 SDLUDQM	1	Output Output	Port J4: Output port Data enable signal for D8 to D15 for SDRAM
PJ5 NDALE	1	I/O Output	Port J5: I/O port Address latch enable signal for NAND Flash
PJ6 NDCLE	1	I/O Output	Port J6: I/O port Command latch enable signal for NAND Flash
PJ7 SDCKE	1	Output Output	Port J7: Output port Clock enable signal for SDRAM

Table 2.2.1 Pin names and functions (4/6)

Pin name	Number of Pins	I/O	Functions
PK0 LCP0	1	Output Output	Port K0: Output port Signal for LCD driver
PK1 LLOAD	1	Output Output	Port K1: Output port Signal for LCD driver: Data load signal
PK2 LFR	1	Output Output	Port K2: Output port Signal for LCD driver
PK3 LVSYNC	1	Output Output	Port K3: Output port Signal for LCD driver: Vertical sync signal
PK4 LHSYNC	1	Output Input	Port K4: Output port Signal for LCD driver: Horizontal sync signal
PK5 LGOE0	1	Output Output	Port K5: Output port Signal for LCD driver
PK6 LGOE1	1	Output Output	Port K6: Output port Signal for LCD driver
PK7 LGOE2	1	Output Output	Port K7: Output port Signal for LCD driver
PL0 to PL7 LD0 to LD7	8	I/O Output	Port L0 to L7: I/O port Data bus for LCD driver: LD0 to LD7
PM1 TA1OUT MLDALM	1	Output Output Output	Port M1: Output port Timer A1 output: Output pin for 8 bit timer 1 Melody / Alarm output pin
PM2 $\overline{\text{ALARM}}$ $\overline{\text{MLDALM}}$	1	Output Output Output	Port M2: Output port Alarm output from RTC Melody / Alarm output pin (inverted)
PM7 PWE	1	Output Output	Port M7: Output port External power supply control output: Pin to control ON/OFF for external power supply. In stand-by mode, outputs “L” level In other than stand-by mode, outputs “H” level
PN0 to PN7 KO0 to KO7	8	I/O Output	Port N: I/O port Key output 0 to 7: Key scan strobe pin (programmable Open-drain output)
PP3 INT5 TA7OUT TXD0 TXD1	1	I/O Input Output Output Output	Port P3: I/O port (Schmitt-input) Interrupt request pin 5: Interrupt request pin with programmable rising/falling edge Timer A7 output: Output pin for 8 bit timer 7 Transmit data for serial 0: programmable Open-drain output Transmit data for serial 1: programmable Open-drain output
PP4 INT6 TB0IN0 RXD0 RXD1	1	I/O Input Input Input Input	Port P4: I/O port (Schmitt-input) Interrupt request pin 6: Interrupt request pin with programmable rising/falling edge Timer B0 input: Input pin for 16 bit timer 0 Receive data for serial 0 Receive data for serial 1
PP5 INT7 TB1IN0 SCLK0 $\overline{\text{CTS0}}$ SCLK1 $\overline{\text{CTS1}}$	1	I/O Input Input I/O Input I/O Input	Port P5: I/O port (Schmitt-input) Interrupt request pin 7: Interrupt request pin with programmable rising/falling edge Timer B1 input: Input pin for 16 bit timer 1 Clock I/O for serial 0 Enable to send data for serial 0 (Clear to send) Clock I/O for serial 1 Enable to send data for serial 1 (Clear to send)
PP6 TB0OUT0	1	Output Output	Port P6: I/O port Timer B0 output: Output pin for 16 bit timer 0

Table 2.2.1 Pin names and functions (5/6)

Pin name	Number of Pins	I/O	Functions
PR0 SPDI	1	I/O Input	Port R0: I/O port Data input pin for SD card
PR1 SPDO	1	I/O Output	Port R1: I/O port Data output pin for SD card
PR2 $\overline{\text{SPCS}}$	1	I/O Output	Port R2: I/O port Chip select signal for SD card
PR3 SPCLK	1	I/O Output	Port R3: I/O port Clock output pin for SD card
PT0 to PT7 LD8 to LD15	8	I/O Output	Port T0 to T7: I/O port Data bus for LCD driver: LD8 to LD15
PV6 SDA	1	I/O I/O	Port V6: I/O port Send/receive data at I^2C mode
PV7 SCL	1	I/O I/O	Port V7: I/O port Input/output clock at I^2C mode
PX4 CLKOUT LDIV	1	Output Output Output	Port X4: Output port Internal clock output pin Output pin for LCD driver
PX5 X1USB X1D4	1	I/O Input Output	Port X5: I/O port Clock input pin for USB Direct clock output pin

Table 2.2.1 Pin names and functions (6/6)

Pin name	Number of Pins	I/O	Functions
D+, D–	2	I/O	USB-data connecting pin Connect pull-up(DVCC3A) or pull-down resistor to both pins to avoid through current when USB is not in use.
$\overline{\text{TEST}}$	1	Input	TEST pin. Usually fix to “H” level.
AM1,AM0	2	Input	Operation mode; Fix to AM1 = “0”,AM0 = “1” for 16 bit external bus starting Fix to AM1 = “1”,AM0 = “0” is prohibit to set Fix to AM1 = “1”,AM0 = “1” for BOOT (32 bit internal Mask ROM) starting Fix to AM1 = “0”,AM0 = “0” is prohibited to set
X1/X2	2	I/O	High-frequency oscillator circuit connection pin
XT1/XT2	2	I/O	Low-frequency oscillator circuit connection pin
$\overline{\text{RESET}}$	1	Input	Reset: Initialize TMP92CF29A (Schmitt-input , with pull-up resistor)
VREFH	1	Input	Pin for reference voltage input to AD converter(H)
VREFL	1	Input	Pin for reference voltage input to AD converter(L)
AVCC	1	–	Power supply pin for AD converter
AVSS	1	–	GND pin for AD converter (0V)
DVCC3A	8	–	Power supply pin for peripheral I/O-A (All DVCC3A pins should be connected to the power supply pin)
DVCC1A	4	–	Power supply pin for internal logic-A (All DVCC1A pins should be connected to the power supply pin)
DVCC1B	1	–	Power supply pin for internal logic-B (Keep the voltage DVCC1A level)
DVSSCOM	8	–	GND pin (0V) (All DVSS pins should be connected to GND(0V))
DVCC1C	1	–	Power supply pin for High speed oscillator (Keep the voltage DVCC1A level)
DVSS1C	1	–	GND pin (0V) (DVSS1C pin should be connected to GND(0V))

Table 2.2.2 shows the range of operational voltage for power supply pins.

Table 2.2.2 the range of operational voltage for power supply pins

Power supply pin	Range of operational voltage
DVCC1A	1.4V~1.6V
DVCC1B	
DVCC1C	
DVCC3A	3.0V~3.6V
AVCC	

3. Operation

This section describes the basic components, functions and operation of the TMP92CF29A.

3.1 CPU

The TMP92CF29A contains an advanced high-speed 32-bit CPU (TLCS-900/H1 CPU)

3.1.1 CPU Outline

The TLCS-900/H1 CPU is a high-speed, high-performance CPU based on the TLCS-900/L1 CPU. The TLCS-900/H1 CPU has an expanded 32-bit internal data bus to process Instructions more quickly.

The following is an outline of the CPU:

Table 3.1.1Outline of TMP92CF29A

Parameter	TMP92CF29A	
Width of CPU Address Bus	24-bit	
Width of CPU Data Bus	32-bit	
Internal Operating Frequency	Max 80 MHz	
Minimum Bus Cycle	1-clock access (12.5ns at 80 MHz)	
Internal RAM	32-bit 2-1-1-1 clock access	
Internal Boot ROM	32-bit 2-clock access	
Internal I/O	8-bit, 2-clock access	INTC,SDRAMC, MEMC,LCDC, TSI,PORT,PMC
	16-bit, 2-clock access	MMU,USB, NDFC,SPIC,DMAC
	32-bit, 2-clock access	I^{2}S
	32-bit, 1-clock access	MAC
	8-bit, 5 to 6-clock access	TMRA,TMRB, SIO,RTC, MLD/ALM, SBI CGEAR,ADC,WDT
External memory (SRAM, MASKROM etc.)	8/16-bit 2-clock access (waits can be inserted)	
External memory (SDRAM)	16-bit 1-clock access	
External memory (NAND FLASH)	8/16-bit 2-clock access (waits can be inserted)	
Minimum Instruction Execution Cycle	1-clock (12.5ns at 80 MHz)	
Conditional Jump	2-clock (25.0ns at 80 MHz)	
Instruction Queue Buffer	12-byte	
Instruction Set	Compatible with TLCS-900/L1 (LDX instruction is deleted)	
Micro DMA	8-channel	
Hardware DMA	6-channel	

3.1.2 Reset Operation

When resetting the TMP92CF29A microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input Low for at least 20 system clocks (32μs at X1=10MHz).

At reset, since the clock doublers (PLL0) is bypassed and the clock-gear is set to 1/16, the system clock operates at 625 kHz(X1=10MHz).

When the Reset has been accepted, the CPU performs the following. CPU internal registers do not change when the Reset is released.

- Sets the Stack Pointer (XSP) to 00000000H.
- Sets bits <IFF2:0> of the Status Register (SR) to “111” (thereby setting the Interrupt Level Mask Register to level 7).
- Clears bits <RFP1:0> of the Status Register to “00” (thereby selecting Register Bank 0).

When the Reset is released, the CPU starts executing instructions according to the Program Counter settings.

- Sets the Program Counter (PC) as follows in accordance with the Reset Vector stored at address FFFF00H~FFFF02H:

PC<7:0>	←	data in location FFFF00H
PC<15:8>	←	data in location FFFF01H
PC<23:16>	←	data in location FFFF02H

When the Reset is accepted, the CPU sets internal I/O, ports and other pins as follows.

- Initializes the internal I/O registers as table of “Special Function Register” in Section 5.

Note1: This LSI builds in RAM internally. However, the data in internal RAM may not be held by Reset operation. After reset, initialize the data in internal RAM.

Note2: This LSI builds in PMC function (for reducing stand-by current by blocking the power supply of DVCC1A and DVCC1C). However, if executing reset operation without supplying DVCC1A and DVCC1C, the current may flow to internal. When reset this LSI, supply the power of DVCC1A and DVCC1C first and wait until the power supply stabilizes.

Figure 3.1.1 shows reset timing chart. Figure 3.1.2 shows the example of order of supplying power and the timing of releasing reset.

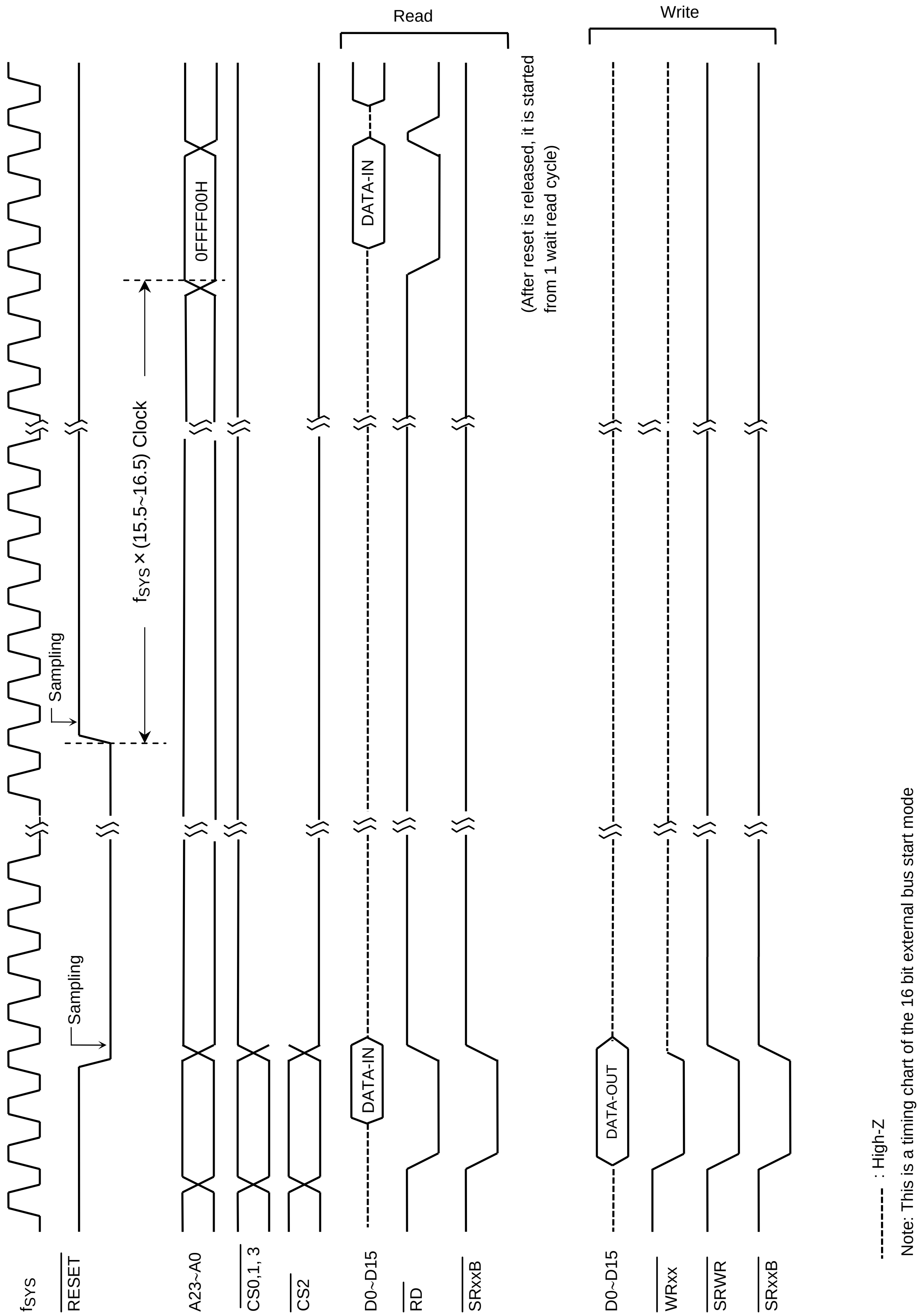
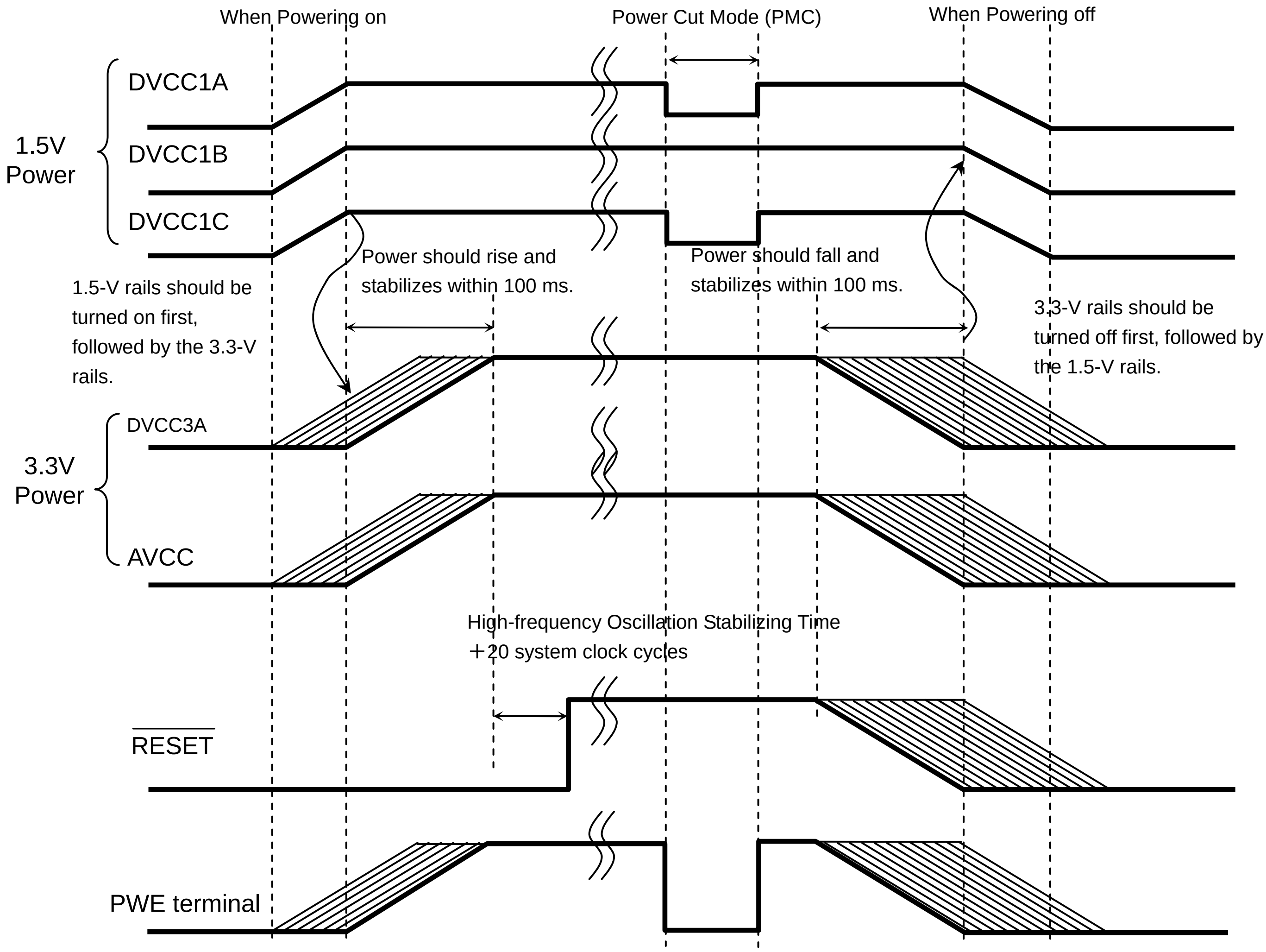


Figure 3.1.1 TMP92CF29A Reset timing chart

This LSI has the restriction for the order of supplying power. Be sure to supply external 3.3V power with 1.5V power is supplied.



Note1: Although it is possible to turn on or off the 1.5-V and 3.3-V power supply rails simultaneously, it may cause external pins to temporarily become unstable. Therefore, if there is any possibility that this would affect peripheral devices connected with the TMP92CF29A, external power supplies should be turned on or off while the internal power supplies are stable, as indicated by the heavy lines in the diagram above.

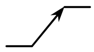
Note2: In the power-on sequence, the 3.3-V power supply rails must not be turned on before the ones of 1.5-V . In the power -off sequence, the 3.3-V power supply rails must not be turned off after the ones of 1.5-V.

Figure 3.1.2 Power on Reset Timing Example

3.1.3 Setting of AM0 and AM1

Set AM1 and AM0 pins as shown in Table 3.1.2 according to system usage.

Table 3.1.2 Operation Mode Setup Table

Mode Setup input pin			Operation Mode
$\overline{\text{RESET}}$	AM1	AM0	
	0	1	16-bit external bus starting
	1	0	Test mode (Prohibit to set)
	1	1	BOOT(32-bit internal-MROM) starting (BOOT mode)
	0	0	Test mode (Prohibit to set)

3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP92CF29A.

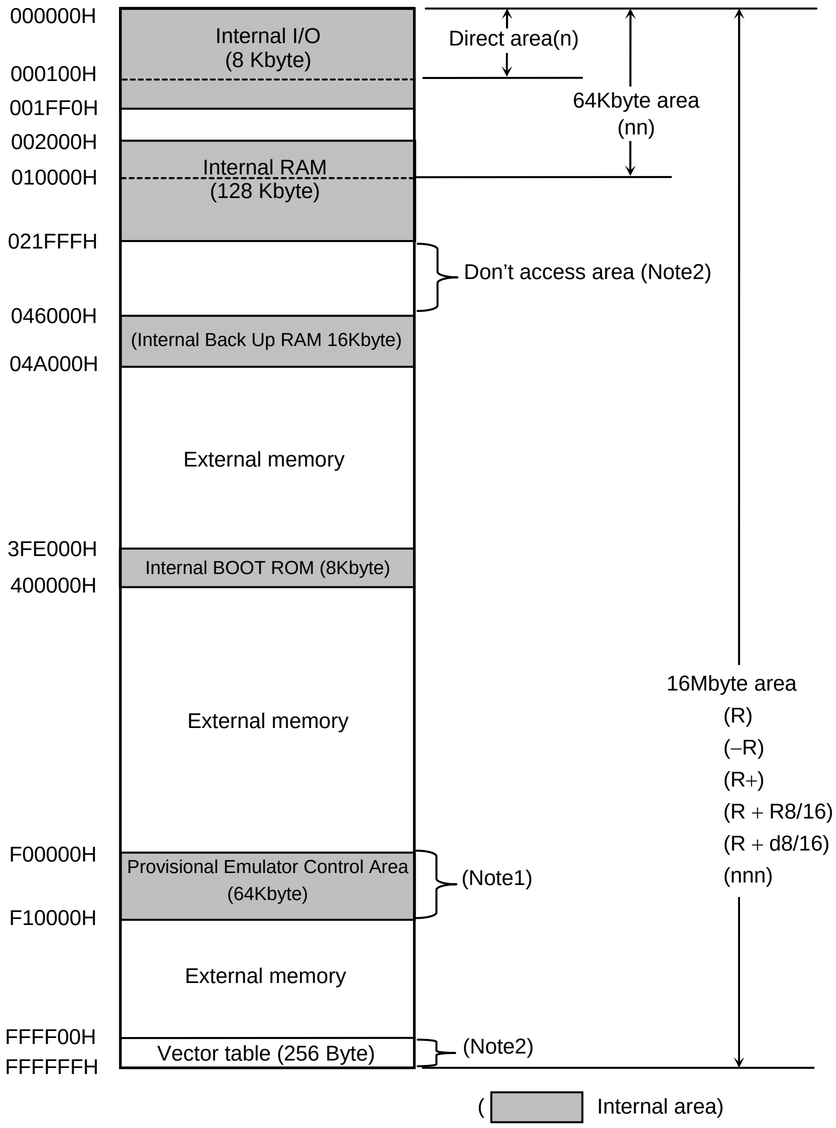


Figure 3.2.1 Memory Map

Note1: If using an emulator, an optional 64 Kbytes of the 16M bytes area is used for emulator control. Therefore, if using an emulator, this area cannot be used.

Note2: Do not use the 144K byte area (022000H to 045FFFH) and the last 16-byte area (FFFFF0H to FFFFFFH). This area is reserved as internal area.

3.3 Differences between the TMP92CZ26A/CF26A and the TMP92CF29A

The TMP92CF29A is a lower pin-count version of the TMP92CF26A with fewer functions (there are some added functions).

Sections 3.3.1 through 3.3.13 describe the functions that are deleted or newly added to the TMP29CF29A. There are no major differences in AC/DC characteristics. For details, refer to the chapter “Electrical Characteristics”.

3.3.1 DSU Circuit Deleted

The TMP92CF29A does not support the DSU function, which is available in the TMP92CZ26A/CF26A.

The development environment is offered with the TMP92CF26AXBG. (The DSU function is used and a pin conversion is required.) Therefore, functions that are modified or newly added to the TMP92CF29A cannot be debugged with development tools. (Please use the actual device or a ROM emulator to debug the TMP92CF29A.)

3.3.2 Internal I/O Functions Deleted and Modified

[Deleted function]

The TMP92CF29A has only one I^2S channel (Channel 0), whereas the TMP92CZ26A/CF26A has Channels 0 and 1. When using the TMP92CF29A, therefore, do not access the addresses where special-function registers for this deleted function have been mapped. For details, see the Table of Special Function Registers (SFRs).

[Modified function 1]

The SIO channel (SIO1) is newly added to the TMP92CF29A with its control registers. For details, see Table 3.3.1.

[Modified function 2]

There are some modifications to the port control method (multiplexed pin settings) and associated registers. If an ICE using the TMP92CF26A is used for development and debugging modified and added registers cannot be debugged. For details, see Table 3.3.1.

3.3.3 Port Pins Deleted

In the TMP92CF29A, the following port pins are deleted as opposed to the TMP92CZ26A/CF26A.

- $\overline{\text{DBGE}}$: Debug enable pin (The DSU function is not available.)
- Port 8: P84 ($\overline{\text{CSZB}}$), P85 ($\overline{\text{CSZC}}$)
- Port F: PF3 (I2S1CKO), PF4 (I2S1DO), PF5 (I2SWS)
- Port P: PP7 (TB1OUT0), PP2 (TA5OUT), PP1 (TA3OUT)
- Port U: PU0 to PU7 (LD16 to LD23)
- Port V: PV0, PV1, PV2, PV3, PV4
- Port W: PW0 to PW7
- Port X: PX7
- Port Z: PZ0 to PZ7

3.3.4 Maximum Memory Size Accessible with the MMU Function Reduced

With the deletion of the P84 ($\overline{\text{CSZB}}$) and P85 ($\overline{\text{CSZC}}$) pins, the maximum memory size that can be expanded with the MMU function is reduced, resulting in a reduced number of usable banks. In the TMP92CZ26A/CF26A the total expandable memory size is 3.1 Gbytes, which is reduced to 2.1 Gbytes in the TMP92CF29A. Accordingly, the number of banks in the Z area is reduced from 512 banks to 256 banks.

If an ICE using the TMP92CF26A is used for development and debugging, be careful about registers and banks which are available in the TMP92CF26A but do not exist in the TMP92CF29A. For details, see the chapter on the MMU function.

3.3.5 One of the I^2S Channels Deleted and I^2S Function Modified

[Deleted function]

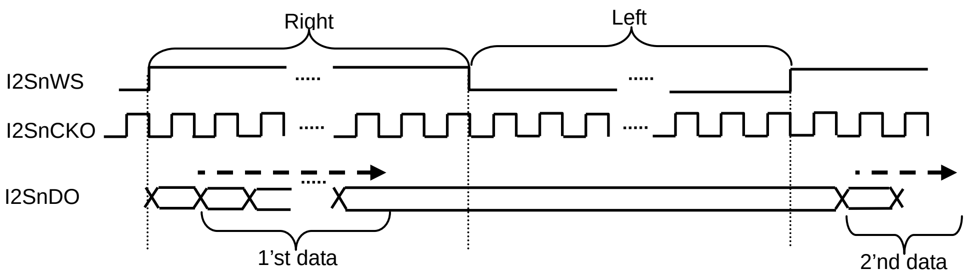
The TMP92CF29A has only one I^2S channel (Channel 0), whereas the TMP92CZ26A/CF26A has Channels 0 and 1.

[Modified function]

The monophonic data output format of the I^2S function is modified as shown below.

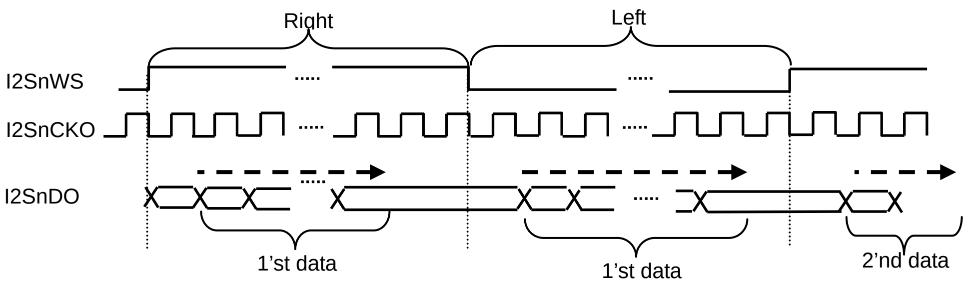
<u>TMP92CZ26A/CF26A monophonic data output (I^2S format)</u>

Data is output from either right or left channel.



<u>TMP92CF29A monophonic data output (I^2S format)</u>

Identical data is output from both right and left channels.



If an ICE using the TMP92CF26A is used for development, data is output from only one channel in monophonic mode. For details, see the chapter on the I^2S Interface.

3.3.6 $\overline{\text{TEST}}$ Pin Added

In the TMP92CF29A, the $\overline{\text{TEST}}$ pin is newly added. This pin must always be fixed to high level.

3.3.7 Port L Function Added

Port L is an output-only port in the TMP92CZ26A/CF26A, whereas the TMP92CF29A allows Port L to be used as an input or output. In the TMP92CF29A, Port L is set as an input immediately after a system reset. If an ICE using the TMP92CF26A is used for development and debugging, this new function cannot be used.

3.3.8 X1D4 Pin Added

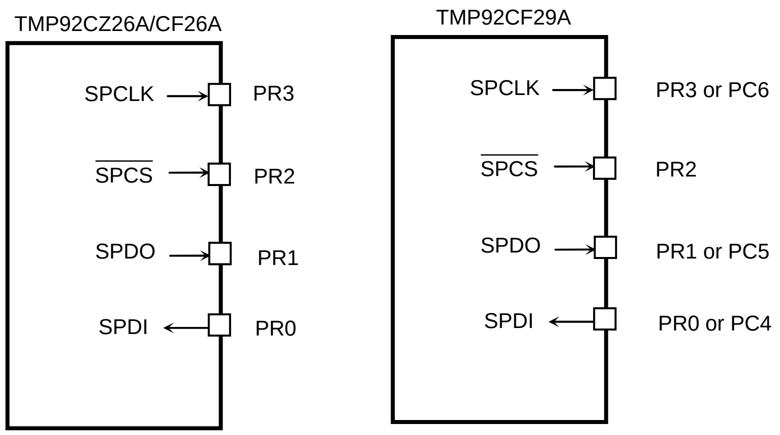
In the TMP92CF29A, a new Port PX5 function is added for outputting a clock that is 1/1, 1/2, 1/4 or 1/8 of the oscillation frequency of the X1 and X2 pins. If an ICE using the TMP92CF26A is used for development and debugging, this function cannot be used.

3.3.9 SPI Controller Function Added

In the TMP92CZ26A/CF26A, the SPI control signals are multiplexed with Port PR. In the TMP92CF29A, the SPI control signals are multiplexed with Port PR and Port PC (excluding the $\overline{\text{SPCS}}$ signal). If an ICE using the TMP92CF26A is used for development and debugging, registers asociated with the following new functions cannot be debugged.

- Output the SPCLK signal from the PC6 pin
- Output the SPDO signal from the PC5 pin
- Input the SPDI signal from the PC4 pin

For details, refer to the chapter on the SPI controller.



3.3.10 LCD Controller Functions Added and Deleted

[Deleted function]

In the TMP92CZ26A/CF26A, up to 16M colors are supported for TFT LCDs. In the TMP92CF29A, up to 64K colors are supported because the number of bus lines dedicated to the LCDD is reduced from 24 to 16.

[Modified function]

The TMP92CF26A supports TFT monochrom panels, which are not supported in the TMP92CZ26A/CF26A. However, if an ICE using the TMP92CF26A is used for development and debugging, registers associated with this new function cannot be debugged.

・TFT monochrome function

For details, refer to the chapter on the LCD controller.

- Comparison of the LCD Control Registers

TMP92CZ26A/CF26A: LCDMODE0 register

LCDMODE0 (0280H)		7	6	5	4	3	2	1	0
	Bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
	Read/Write	R/W							
	Reset State	0	0	1	1	0	0	0	0
	Function	Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transmission speed SCPW2= 0 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= 1 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		MODE selection 0000: Reserved 0001: SR monochrome 0010: SR 4Gray 0011: Reserved 0100: SR 16Gray 0101: SR 64Gray 0110: STN 256 color 0111: STN 4096 color		1000: STN 64K color 1001: Reserved 1010: TFT 256 color 1011: TFT 4096 color 1100: TFT 64K color 1101: **TFT 16M, 256k** 1110: **Reserved** 1111: Reserved	

TMP92CF29A: LCDMODE0 register

LCDMODE0 (0280H)		7	6	5	4	3	2	1	0
	Bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
	Read/Write	R/W							
	Reset State	0	0	1	1	0	0	0	0
	Function	Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transmission speed SCPW2= 0 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= 1 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		MODE selection 0000: Reserved 0001: SR monochrome 0010: SR 4Gray 0011: Reserved 0100: SR 16Gray 0101: SR 64Gray 0110: STN 256 color 0111: STN 4096 color		1000: STN 64K color 1001: Reserved 1010: TFT 256 color 1011: TFT 4096 color 1100: TFT 64K color 1101: **Reserved** 1110: **TFT monochrome** 1111: Reserved	

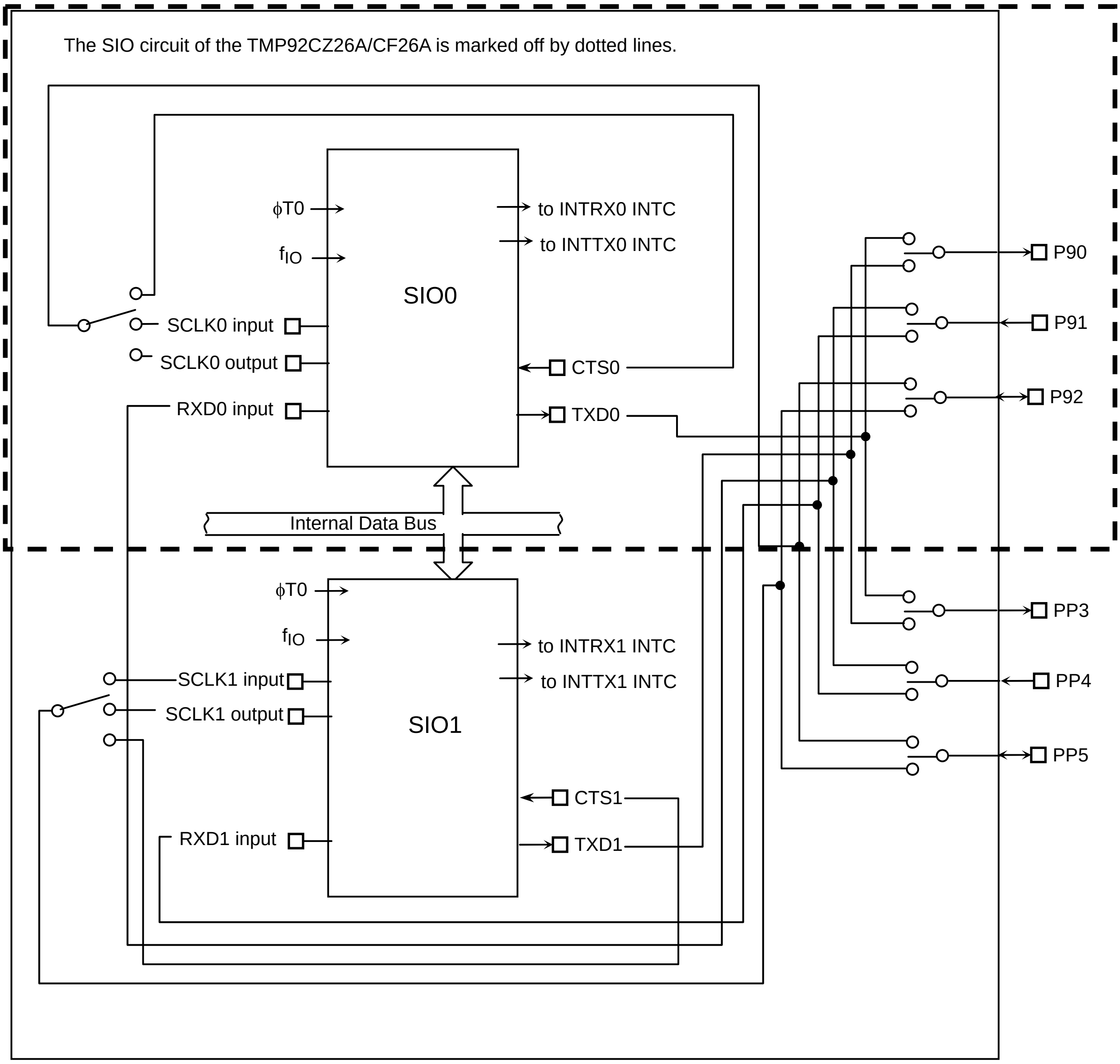
3.3.11 SIO Channel Added and SIO Function Modified

[Added function]

In the TMP92CZ26A/CF26A only one SIO channel is available, whereas the TMP92CF29A has two SIO channels. However, if an ICE using the TMP92CF26A is used for development and debugging, the newly added Channel 1 cannot be debugged.

[Modified function]

Each of the two SIO channels can be connected to the P90, P91 and P92 pins or the PP3, PP4 and PP5 pins. However, if an ICE using the TMP92CF26A is used for development and debugging, this modified port switching function cannot be debugged.



3.3.12 Interrupt Sources Deleted and Modified

[Deleted function]

As the number of I²S channels is reduced from two channels to one channel, the corresponding interrupt vector is deleted.

[Modified function]

As the number of SIO channels is increased from one channel to two channels, the interrupt vectors for SIO1 serial receive end and SIO1 serial transmission end are added in the TMP92CF29A. However, if an ICE using the TMP92CF26A is used to development and debugging, this modified interrupt function cannot be debugged.

TMP92CF29A Interrupt Vectors and Micro DMA/HDMA Start Vectors

Default Priority	Type	Interrupt Source/Micro DMA Request Source	Vector Value	Vector Reference Address	Micro DMA /HDMA Start Vector
1	Non maskable	Reset or [SWI0] instruction	0000H	FFFF00H	
2		[SWI1] instruction	0004H	FFFF04H	
(Omitted)		(Omitted)			
40		INTI2S0: I²S (Channel 0)	009CH	FFFF9CH	27H
41		**(Reserved)**	–	–	–
42		INTADM: AD monitor function	00A4H	FFFFA4H	29H
43		INTSBI: SBI	00A8H	FFFFA8H	2AH
44		INTSPIRX: SPIC receive	00ACH	FFFFACH	2BH
45		INTSPITX: SPIC transmission	00B0H	FFFFB0H	2CH
46		INTRSC: NAND Flash controller	00B4H	FFFFB4H	2DH
47		INTRDY: NAND Flash controller	00B8H	FFFFB8H	2EH
48		INTUSB: USB	00BCH	FFFFBCH	2FH
49		**INTRX1: Serial receive end**	**00C0H**	**FFFFC0H**	**30H**
50		**INTTX1: Serial transmission end**	**00C4H**	**FFFFC4H**	**31H**

3.3.13 Pull-Up Control Port for USB Boot Modified

To boot from the on-chip boot ROM via USB, the control signal for pulling up the D+ pin is required. In the TMP92CZ26A/CF26A, Port PU6 is used for this purpose, whereas the PP6 pin is used in the TMP92CF29A.

Table 3.3.1 summarizes the differences between the TMP92CZ26A and the TMP92CF29A. For details, refer to the chapter on each functional block.

Table 3.3.1 Differences between the TMP92CZ26A and the TMP92CF29A

Item	TMP92CZ26A	TMP92CF29A	Note	
RAM	288 KB	144 KB		
ROM	8 KB (BOOT)	8 KB (BOOT)		
Package	FBGA228-P-1515-0.80A	LQFP176-P-2020-0.40F		
Pin count	228	176		
DSU	Supports	Not supports	Development tools using the TMP92CZ26 are offered. 10 pins are deleted: $\overline{\text{DBGE}}$, PZ0 to PZ7, PU7	-10
I2S	2 channels	1 channel	Channel 1 are deleted 3 pins are deleted: PF3 (I2S1CKO, X1D4), PF4 (I2S1DO), PF5 (I2S1WS)	-3
8-bit timer	8 channels	8 channels	2 pins deleted: PP1(TA3OUT), PP2 (T5OUT)	-2
SIO	1 channel	2 channels	The newly added channel cannot be debugged with development tools.	
16-bit timer	2 channels	2 channels	1 pin is deleted: PP7(TB1OUT0)	-1
LCDC	TFT 16M colors	TFT 64k colors	LD bus lines are reduced from 24 lines to 16 lines 7 pins are deleted: PU0(LD16) to PU7(LD23) (Deletion of PU7 is already counted with the DSU.) The LCDC cannot be used in 32-bit bus mode.	-7
General-purpose port pins	P84/ $\overline{\text{CSZB}}$ P85/ $\overline{\text{CSZC}}$ PV0 PV1 PV2 PV3 PV4 PX7 PW0 to PW7	Deleted Deleted Deleted Deleted Deleted Deleted Deleted Deleted Deleted	15 port pins are deleted	-16
Power supply pins	DVCC3A 12 DVCC3B 1 DVCC1A 5 DVCC1B 1 DVCC1C 1 DVCC1S 1 DVSSCOM 12	DVCC3A 8 DVCC3B 0 DVCC1A 4 DVCC1B 1 DVCC1C 1 DVCC1S 1 DVSSCOM 8	10 power supply pins deleted	-10
Dummy	4 pins	None	4 dummy pins are deleted	-4
$\overline{\text{TEST}}$	Not supports	Adds	The TEST pin is added	+1
TOTAL			228-pin BGA → 176-pin QFP (A total of 52 pins are deleted)	-52

Other Specification Changes	
The number of SIO channels is increased to two channels.	In the TMPCZ26A/CF26A only one SIO channel is available, whereas the TMP92CF29A has two SIO channels. However, the added SIO function cannot be debugged with development tools.
The X1D4 pin is added.	The X1D4 pin can be used to output x1, x1/2, x1/4 or x1/8 of the external clock according to the CPU state (in NORMAL, IDLE1 and IDLE2 modes). However, this function cannot be debugged with development tools.
The LCDC supports monochrome TFT panels.	The LCDC of the TMP92CF29A supports monochrome TFT mode. However, this new function cannot be debugged with development tools.
SPI output can be made from either of two pins.	As in the case of the TMP92CZ26A/CF26A, the TMP92CF29A has only one SPI channel, but the output pin can be selected from two pins. However, this new function cannot be debugged with development tools.

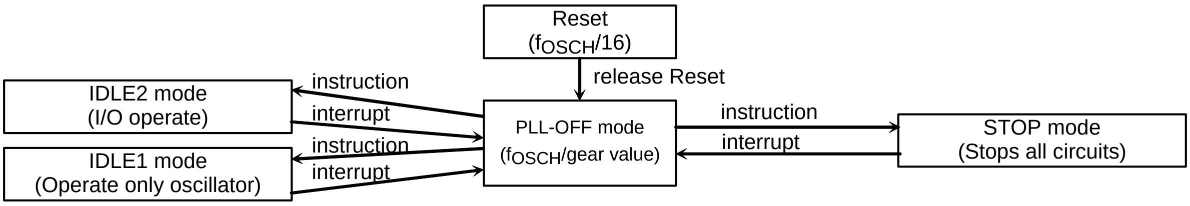
3.4 Clock Function and Standby Function

The TMP92CF29A contains (1) clock gear, (2) clock doubler (PLL), (3) standby controller and (4) noise reduction circuits. They are used for low-power, low-noise systems.
This chapter is organized as follows:

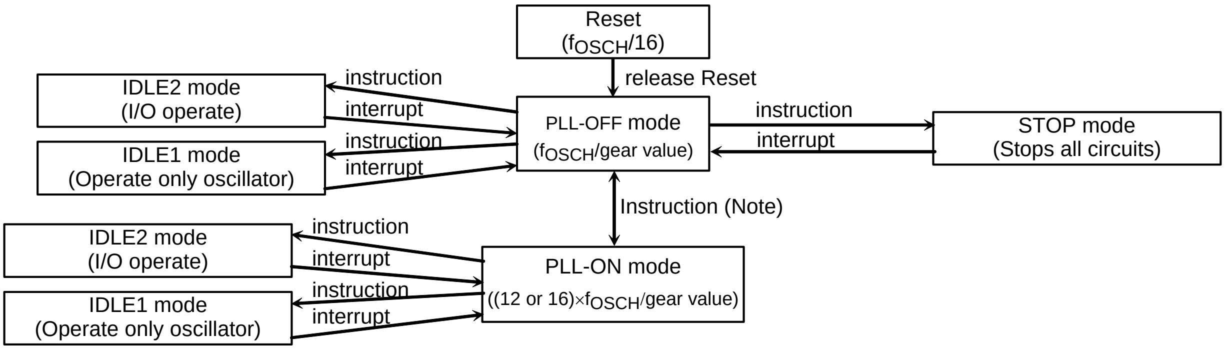
3.4.1 Block diagram of system clock
3.4.2 SFR
3.4.3 System clock controller
3.4.4 Clock doubler (PLL)
3.4.5 Noise reduction circuits
3.4.6 Standby controller

The clock operating modes are as follows: (a) PLL-OFF Mode (X1, X2 pins only), (b) PLL-ON Mode (X1, X2, and PLL).

Figure 3.4.1 shows a transition figure.



(a) PLL-OFF mode transition figure



(b) PLL-OFF , PLL-ON mode transition figure

Note 1: When shifting from PLL-ON mode to PLL-OFF mode, execute the following setting in the same order.

(1) Change CPU clock (Set “0” to PLLCR0<FCSEL>)

(2) Stop PLL circuit (Set “0” to PLLCR1<PLLON>)

Note 2: It is not possible to shift from PLL-ON mode to STOP mode directly.
PLL-OFF mode should be set once before shifting to STOP mode.

Figure 3.4.1 System clock block diagram

The clock frequency input from the X1 and X2 pins is called f_{OSCH} and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<GEAR2:0> is called the system clock f_{SYS}. And one cycle of f_{SYS} is defined to as one state.

3.4.1 Block diagram of system clock

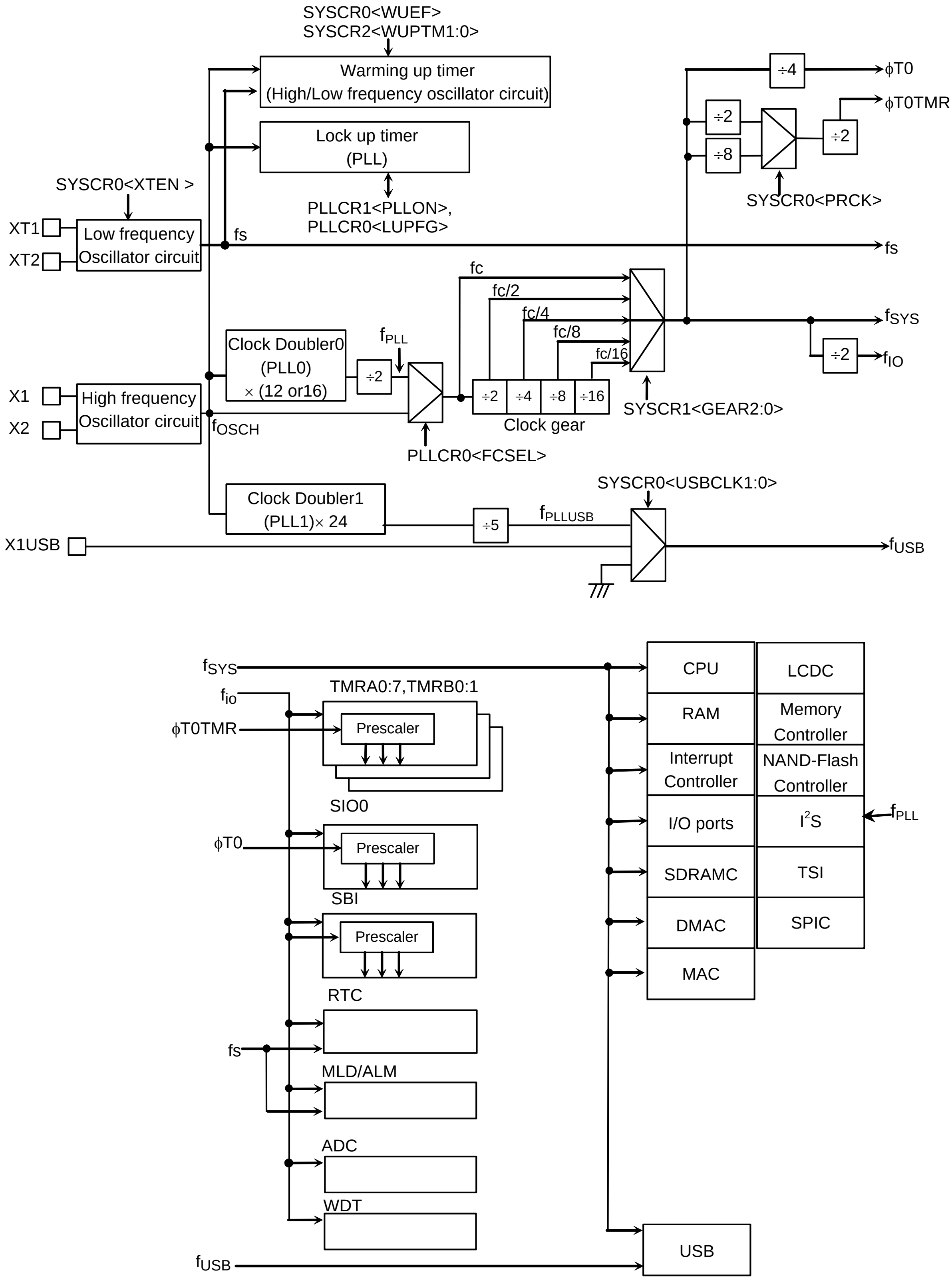


Figure 3.4.2 Block Diagram of System clock

TMP92CF29A has two PLL circuits: one is for CPU (PLL0) and the other for USB (PLL1). Each PLL can be controlled independently. Frequency of external oscillator is 6 to 10MHz.

Don't connect oscillator more than 10MHz. When clock is input by using external oscillator, range of input frequency is 6 to 10MHz.

Don't input the clock over 10MHz.

Table 3.4.1 Setting example for f_{OSCH}

	High frequency: f_{OSCH}	System clock: f_{SYS}	System clock: f_{SYS}	USB clock: f_{USB}
(a) USB in use, with PLL (PLL0 ON/PLL1 ON)	10.0 MHz	Max 80 MHz	Max 60 MHz	48 MHz
(b) USB not in use, with PLL (PLL0 ON/PLL1 OFF)	Max 10.0 MHz	Max 80 MHz	Max 60 MHz	–
(c) USB not in use, without PLL (PLL0 OFF/PLL1 OFF)	Max 10.0 MHz	Max 10 MHz	Max 10 MHz	–

Note: When using USB, the high-frequency oscillator should be 10.0 MHz.

3.4.2 SFR

SYSCR0 (10E0H)

	7	6	5	4	3	2	1	0
bit Symbol		XTEN	USBCLK1	USBCLK0		WUEF		PRCK
Read/write		R/W				R/W		R/W
Reset State		1	0	0		0		0
Function		Low -frequency oscillator circuit (fs) 0: Stop 1: Oscillation	Select the clock of USB(f_{USB}) 00:Disable 01: Reserved 10: X1USB 11: f_{PLLUSB}			Warm-up Timer 0: Write Don’t care Note3 1: Write start timer 0: Read end warm-up 1: Read do not end warm-up		Select Prescaler clock 0: $f_{SYS}/2$ 1: $f_{SYS}/8$

SYSCR1 (10E1H)

	7	6	5	4	3	2	1	0
bit Symbol						GEAR2	GEAR1	GEAR0
Read/write						R/W		
Reset State						1	0	0
Function						Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101: Reserved 110: Reserved 111: Reserved		

SYSCR2 (10E2H)

	7	6	5	4	3	2	1	0
bit Symbol	–	CKOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		
Read/write	R/W							
Reset State	0	0	1	0	1	1		
Function	Always write “0”	Select CLKOUT 0: f_{SYS} 1: f_S	Warm-Up Timer 00: Reserved 01: 2^8/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			

Note1: The unassigned registers, SYSCR0<bit7><bit3><bit1>,SYSCR1<bit7:3> and SYSCR2<bit1:0> are read as undefined value.

Note2: Low frequency oscillator circuit is enabled on reset.

Note3: Do not write SYSCR0 resiter during warming up. Because the warm-up end flag doesn’t become enable if write “0” to SYSCR0<WUEF> bit during warming up.
(A read-modify–write operation cannot be performed for SYSCR0 register during warming up.)

Figure 3.4.3 SFR for system clock

		7	6	5	4	3	2	1	0
EMCCR0 (10E3H)	Bit symbol	PROTECT				–	EXTIN	DRVOSCH	DRVOSCL
	Read/Write	R				R/W			
	Reset State	0				0	0	1	1
	Function	Protect flag 0: OFF 1: ON				Always write "0".	1: External clock	fc oscillator drive ability 1: NORMAL 0: WEAK	fs oscillator drive ability 1: NORMAL 0: WEAK
EMCCR1 (10E4H)	Bit symbol	Switch the protect ON/OFF by writing the following to 1st-KEY,2nd-KEY 1st-KEY: write in sequence EMCCR1=5AH,EMCCR2=A5H 2nd-KEY: write in sequence EMCCR1=A5H,EMCCR2=5AH							
	Read/Write								
	Reset State								
	Function								
EMCCR2 (10E5H)	Bit symbol								
	Read/Write								
	Reset State								
	Function								

Note1: When restarting the oscillator in the stop oscillation state (e.g. Restarting the oscillator in STOP mode), set EMCCR0<DRVOSCH>, <DRVOSCL>= "1".

Note2: Do not write EMCCR0<EXTIN> = "1" when using external resonator.

Figure 3.4.4 SFR for system clock

PLLCR0 (10E8H)		7	6	5	4	3	2	1	0
	bit symbol		FCSEL	LUPFG					
	Read/Write		R/W	R					
	Reset State		0	0					
	Function		Select fc-clock 0: f_{OSCH} 1: f_{PLL}	Lock-up timer Status flag 0: not end 1: end					

Note: Ensure that the logic of PLLCR0<LUPFG> is different from 900/L1's DFM.

PLLCR1 (10E9H)		7	6	5	4	3	2	1	0
	bit symbol	PLL0	PLL1	LUPSEL					PLLTIMES
	Read/Write	R/W							R/W
	Reset State	0	0	0					0
	Function	PLL0 for CPU 0: Off 1: On	PLL1 for USB 0: Off 1: On	Select stage of Lock up counter 0: 12 stage (for PLL0) 1:13 stage (for PLL1)					Select the number of PLL 0: ×12 1: ×16

Figure 3.4.5 SFR for PLL

PxDR (xxxxH)		7	6	5	4	3	2	1	0
	bit symbol	Px7D	Px6D	Px5D	Px4D	Px3D	Px2D	Px1D	Px0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Output/Input buffer drive-register for standby-mode							

(Purpose and using)

- This register is used to set each pin-status at stand-by mode.
- All ports have registers of the format shown above. ("x" indicates the port name.)
- For each register, refer to 3.7 Function of Ports.
- Before "HALT" instruction is executed, set each register pin-status. They will be effective after the CPU has executes the "HALT" instruction.
- This is the case regardless of stand-by modes (IDLE2, IDLE1 or STOP).
- This is the case regardless of using PMC function. For details, refer to PMC section.

The Output/Input buffer control table is shown below.

OE	PxnD	Output buffer	Input buffer
0	0	OFF	OFF
0	1	OFF	ON
1	0	OFF	OFF
1	1	ON	OFF

Note1: OE denotes an output enable signal before stand-by mode. Basically, PxCR is used as OE.

Note2: "n" in PxnD denotes the bit number of PORTx.

Figure 3.4.6 SFR for Drive register

3.4.3 System clock controller

The system clock controller generates the system clock signal (f_{SYS}) for the CPU core and internal I/O.

SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling of each oscillator. SYSCR1<GEAR2:0> sets the high frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8, fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

The combination of settings <XEN> = “1”, <SYSCK> = “0” and <GEAR2:0> = “100” will be PLL-OFF mode and cause the system clock (f_{SYS}) to be set to fc/16 after reset.

For example, f_{SYS} is set to 625 kHz when the 10MHz oscillator is connected to the X1 and X2 pins.

(1) Clock gear controller

f_{SYS} is set according to the contents of the Clock Gear Select Register SYSCR1<GEAR2:0> to either fc, fc/2, fc/4, fc/8 or fc/16. Using the clock gear to select a lower value of f_{SYS} reduces power consumption.

(Example)

Changing clock gear

```
SYSCR1    EQU    10E1H

          LD     (SYSCR1),XXXXX001B    ;    Changes system clock fSYS to fc/2
          LD     (DUMMY),00H                Dummy instruction
  X: don't care
```

(High-speed clock gear changing)

To change the clock gear, write the register value to the SYSCR1<GEAR2:0> register. It is necessary for the warming up time to elapse before the change occurs after writing the register value.

There is the possibility that the instruction following the clock gear changing instruction is executed by the clock gear before changing. To execute the instruction following the clock gear switching instruction by the clock gear after changing, input the dummy instruction as follows (instruction to execute the write cycle).

```
(Example)
SYSCR1    EQU    10E1H
          LD     (SYSCR1),XXXXX010B    ;    Changes fSYS to fc/4
          LD     (DUMMY),00H           ;    Dummy instruction
          Instruction to be executed after clock gear changed
```

3.4.4 Clock doubler (PLL)

PLL0 outputs the f_{PLL} clock signal, which is 12 or 16 times as fast as f_{OSCH}. A low-speed frequency oscillator can be used as external oscillator, even though the internal clock is high-frequency.

Since Reset initializes PLL0 to stop status, so setting to PLLCR0 and PLLCR1-register is needed before use.

As with an oscillator, this circuit requires time to stabilize. This is called the lock-up time and it is measured by a 12-stage binary counter. Lock-up time is about 0.41ms at f_{OSCH} = 10MHz.

PLL (PLL1) which is special for USB is built in. Lock-up time is about 0.82ms at f_{OSCH} = 10MHz measured by 13-stage binary counter.

Note1: Input frequency range for PLL
The input frequency range (High frequency oscillation) for PLL is as follows:
f_{OSCH} = X to X MHz (Vcc = 1.4 to 1.6V)

Note2: PLLCR0<LUPFG>
The logic of PLLCR0<LUPFG> is different from 900/L1's DFM.
Exercise care in determining theend of lock-up time.

Note3: PLLCR1<PLL0>, PLLCR1<PLL1>
It is not possible to turn ON both PLL0 and PLL1 simultaneously.
If turning ON simultaneously, one PLL should be turn ON after finishing the lock up of the other PLL.

Table 3.4.2 shows the frequency of f_{SYS} when using PLL and clock gear at f_{OSCH} =10MHz.

Table 3.4.2 The frequency of f_{SYS} at f_{OSCH} = 10MHz

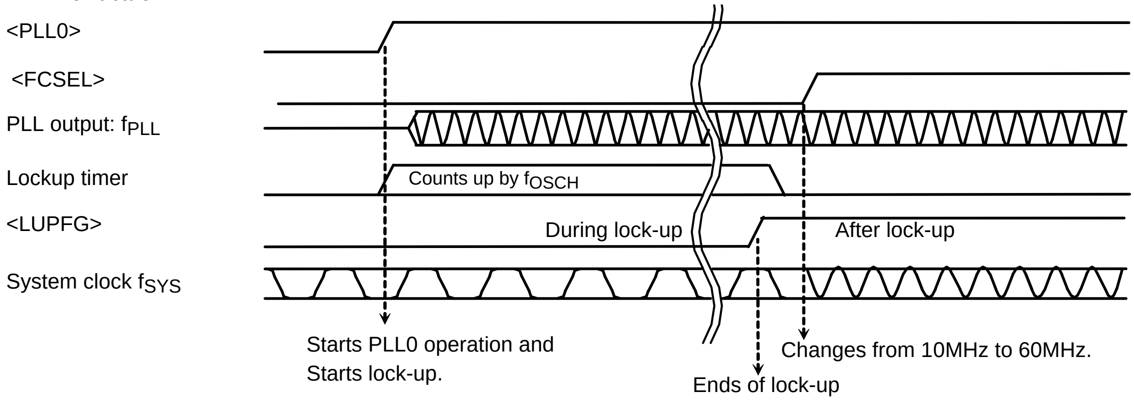
f_{OSCH}	f_{PLL}	Frequency of f_{SYS}				
		fc	fc/2	fc/4	fc/8	fc/16
10MHz	f_{OSCH} 10MHz	10MHz	5MHz	2.5MHz	1.25MHz	625kHz
	×12 120MHz	60MHz	30MHz	15MHz	7.5MHz	3.75MHz
	×16 160MHz	80MHz	40MHz	20MHz	10MHz	5MHz

The following is an example of settings for PLL0-starting and PLL0 stopping.

(Example-1) PLL0-starting

```
PLLCR0  EQU  10E8H
PLLCR1  EQU  10E9H
        LD   (PLLCR1),1XXXXXXXXB   ; Enables PLL0 operation and starts lock up.
LUP:    BIT  5,(PLLCR0)            ; } Detects end of lock-up
        JR   Z,LUP                 ; }
        LD   (PLLCR0), X1XXXXXXB   ; Changes fc from 10 MHz to 60 MHz.
```

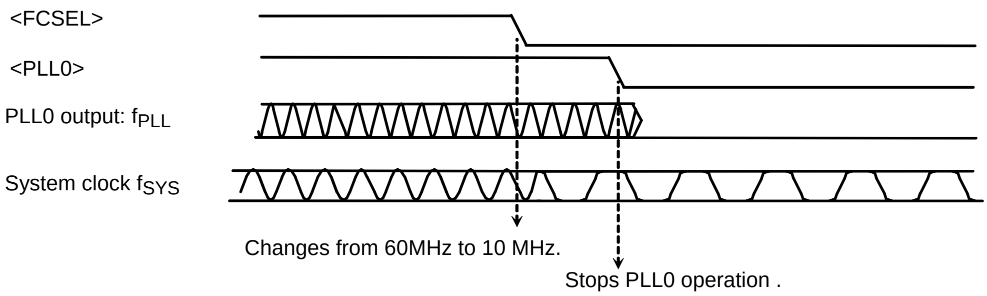
X: Don't care



(Example-2) PLL0-stopping

```
PLLCR0  EQU  10E8H
PLLCR1  EQU  10E9H
        LD   (PLLCR0),X0XXXXXXB    ; Changes fc from 60 MHz to10 MHz.
        LD   (PLLCR1),0XXXXXXXB    ; Stop PLL
```

X: Don't care



Note: PLL1 operates as well.

Limitations on the use of PLL0

1. When stopping PLL operation during PLL0 use, execute the following settings in the same order.

```
        LD      (PLLCR0),X0XXXXXXB      ;   Change the clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB      ;   Stop PLL0
```

X: Don't care

2. When shifting to STOP mode during PLL use, execute the following settings in the same order.

```
        LD      (SYSCR2),XXXX01XXB      ;   Set the STOP mode
        LD      (PLLCR0), X0XXXXXXB     ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1), 0XXXXXXXB     ;   Stop PLL0
        HALT                            ;   Shift to STOP mode
```

X: Don't care

Examples of settings are shown below:

(1) Start Up / Change Control

(OK) High frequency oscillator operation mode(f_{OSCH})→PLL0 start up
→ PLL0 use mode (f_{PLL})

```
        LD      (PLLCR1), 1XXXXXXXB     ;   PLL0 start up / lock up start
LUP:    BIT     5,(PLLCR0)              ;
        JR      Z,LUP                   ;   Check for lock up end flag
        LD      (PLLCR0), X1XXXXXXB     ;   Change the system clock fOSCH to fPLL
```

X: Don't care

(2) Change / Stop Control

(OK) PLL0 use mode (f_{PLL})→ High frequency oscillator operation mode(f_{OSCH})
→ PLL0 Stop

```
        LD      (PLLCR0),X0XXXXXXB      ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB      ;   Stop PLL0
```

X: Don't care

(OK) PLL0 use mode (f_{PLL}) → Set the STOP mode
→High frequency oscillator operation mode (f_{OSCH}) → PLL stop
→ HALT(High frequency oscillator stop)

```
        LD      (SYSCR2),XXXX01XXB      ;   Set the STOP mode
                                            (This command can be executed before use of PLL0)
        LD      (PLLCR0),X0XXXXXXB      ;   Change the system clock fPLL to fOSCH
        LD      (PLLCR1),0XXXXXXXB      ;   Stop PLL0
        HALT                            ;   Shift to STOP mode
```

X: Don't care

(NG) PLL0 use mode (f_{PLL}) → Set the STOP mode
→ HALT(High frequency oscillator stop)

```
        LD      (SYSCR2),XXXX01XXB      ;   Set the STOP mode
                                            (This command can be executed before use of PLL0)
        HALT                            ;   Shift to STOP mode
```

X: Don't care

3.4.5 Noise reduction circuits

Noise reduction circuits are built in, allowing implementation of the following features.

(1) Reduced drivability for high-frequency oscillator circuit

(2) Reduced drivability for low-frequency oscillator circuit

(3) Single drive for high-frequency oscillator circuit

(4) Runaway prevention using SFR protection register

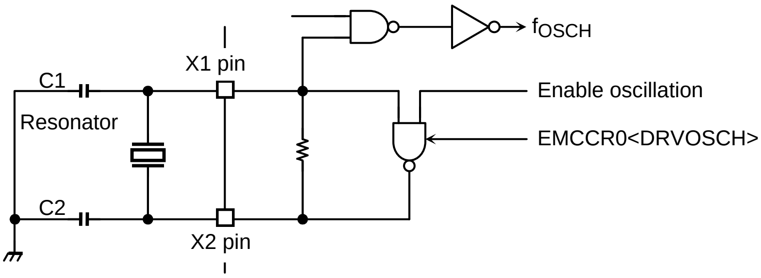
These are set in EMCCR0 to EMCCR2 registers.

(1) Reduced drivability for high-frequency oscillator circuit

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Clock diagram)



(Setting method)

The drivability of the oscillator is reduced by writing“0” to EMCCR0<DRVOSCH> register. At reset, <DRVOSCH> is initialized to “1” and the oscillator starts oscillation by normal-drivability when the power-supply is on.

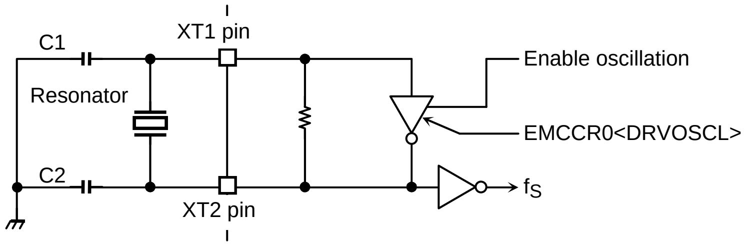
Note: This function (EMCCR0<DRVODCH>= “0”) is available when f_{OSCH} = 6 to 10MHz.

(2) Reduced drivability for low-frequency oscillator circuit

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Block diagram)



(Setting method)

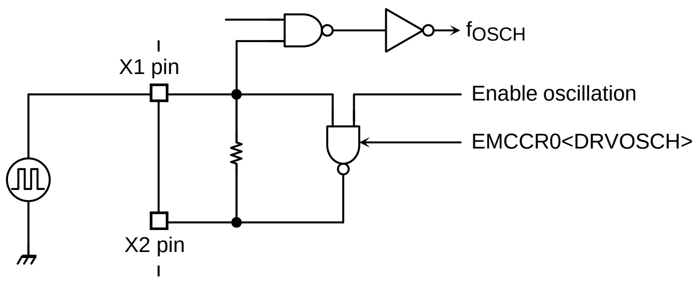
The drivability of the oscillator is reduced by writing “0” to the EMCCR0<DRVOSCL> register. At Reset, <DRVOSCL> is initialized to “1”.

(3) Single drive for high-frequency oscillator circuit

(Purpose)

Remove the need for twin-drives and protect prevent operational errors caused by noise input to X2 pin when an external-oscillator is used.

(Block diagram)



(Setting method)

The oscillator is disabled and starts operation as buffer by writing “1” to EMCCR0<EXTIN> register. X2 pin’s output is always “1”.

At reset, <EXTIN> is initialized to “0”.

Note: Do not write EMCCR0<EXTIN> = “1” when using external resonator.

(4) Runaway prevention using SFR protection register

(Purpose)

Prevention of program runaway caused by introduction of noise.

Write operations to a specified SFR are prohibited so that the program is protected from runaway caused by stopping of the clock or by changes to the memory control register (Memory controller, MMU) which prevent fetch operations.

Runaway error handling is also facilitated by INTP0 interruption.

Specified SFR list

1. Memory controller

   B0CSL/H, B1CSL/H, B2CSL/H, B3CSL/H, BECSL/H
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3, PMEMCR,
   MEMCR0, CSTMGCR, WRTMGCR, RDTMGCR0
   RDTMGCR1, BROMCR
2. MMU

   LOCALPX/PY/PZ, LOCALLX/LY/LZ,
   LOCALRX/RY/RZ, LOCALWX/WY/WZ,
   LOCALESX/ESY/ESZ, LOCALEDX/EDY/EDZ,
   LOCALOSX/OSY/OSZ, LOCALODX/ODY/ODZ
3. Clock gear

   SYSCR0, SYSCR1, SYSCR2, EMCCR0
4. PLL

   PLLCR0,PLLCR1
5. PMC

   PMCCTL

(Operation explanation)

Execute and release of protection (write operation to specified SFR) becomes possible by setting up a double key to EMCCR1 and EMCCR2 registers.

(Double key)

1st-KEY: writes in sequence, 5AH at EMCCR1 and A5H at EMCCR2

2nd-KEY: writes in sequence, A5H at EMCCR1 and 5AH at EMCCR2

Protection state can be confirmed by reading EMCCR0<PROTECT>.

At reset, protection becomes OFF.

INTP0 interruption also occurs when a write operation to the specified SFR is executed with protection in the ON state.

3.4.6 Standby controller

(1) HALT Modes and Port Drive register

When the HALT instruction is executed, the operating mode switches to IDLE2, IDLE1 or STOP Mode, depending on the contents of the SYSCR2<HALTM1:0> register and each pin-status is set according to the PxDR register, as shown below.

		7	6	5	4	3	2	1	0
PxDR (xxxxH)	bit symbol	Px7D	Px6D	Px5D	Px4D	Px3D	Px2D	Px1D	Px0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Output/Input buffer drive-register for standby-mode							

(Purpose and using)

- This register is used to set each pin-status at stand-by mode.
- All ports have this registers of the format shown above (“x” indicates the port-name.)
- For each register, refer to 3.7 Function of Ports.
- Before “HALT” instruction is executed, set each register pin-status. They will be effective after the CPU has executed the “HALT” instruction.
- This is the case regardless of stand-by mode (IDLE2, IDLE1 or STOP).
- This is the case regardless of using PMC function. For details, refer to PMC section.

The Output/Input-buffer control table is shown below.

OE	PxnD	Output buffer	Input buffer
0	0	OFF	OFF
0	1	OFF	ON
1	0	OFF	OFF
1	1	ON	OFF

Note1: OE denotes an output enable signal before stand-by mode. Basically, PxCR is used as OE.

Note2: “n” in PxnD denotes the bit number of PORTx.

The subsequent actions performed in each mode are as follows:

a. IDLE2: Only the CPU halts.
The internal I/O is available to select operation during IDLE2 mode by setting the following register.
Table 3.4.3 shows the registers setting operation during IDLE2 mode.

Table 3.4.3 SFR setting operation during IDLE2 mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRA45	TA45RUN<I2TA45>
TMRA67	TA67RUN<I2TA67>
TMRB0	TB0RUN<I2TB0>
TMRB1	TB1RUN<I2TB1>
SIO0	SC0MOD1<I2S0>
SBI	SBIBR0<I2SBI>
A/D converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

b. IDLE1: Only the oscillator, RTC (real-time clock), and MLD continue to operate.

c. STOP: All internal circuits stop operating.

The operation of each of the different Halt Modes is described in Table 3.4.4.

Table 3.4.4 I/O operation during Halt Modes

<table>
<tr><th colspan="2">HALT Mode</th><th>IDLE2</th><th>IDLE1</th><th>STOP</th></tr>
<tr><td colspan="2">SYSCR2 <HALTM1:0></td><td>11</td><td>10</td><td>01</td></tr>
<tr><td rowspan="8">Block</td><td>CPU, MAC</td><td colspan="3">Stop</td></tr>
<tr><td>I/O ports</td><td colspan="3">Depends on PxDR register setting</td></tr>
<tr><td>TMRA, TMRB</td><td rowspan="4">Available to select
Operation block</td><td colspan="2" rowspan="5">Stop</td></tr>
<tr><td>SIO,SBI</td></tr>
<tr><td>A/D converter</td></tr>
<tr><td>WDT</td></tr>
<tr><td>I²S, LCDC, SDRAMC, Interrupt controller, SPIC, DMAC, NDFC, USB</td><td rowspan="2">Operate</td></tr>
<tr><td>RTC, MLD</td><td>Operate</td><td>Stop</td></tr>
</table>

(2) How to release the Halt mode

These HALT states can be released by resetting or requesting an interrupt. The halt release sources are determined by the combination of the states of the interrupt mask register <IFF2:0> and the halt modes. The details for releasing the HALT status are shown in Table 3.4.5.

- Release by interrupt requesting

The HALT mode release method depends on the status of the enabled interrupt. When the interrupt request level set before executing the “HALT” instruction exceeds the value of the interrupt mask register, the interrupt is processed depending on its status after the HALT mode is released, and the CPU status executing the instruction that follows the HALT instruction. When the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, HALT mode release is not executed.(in non-maskable interrupts, interrupt processing is processed after releasing the halt mode regardless of the value of the mask register.) However only for INT0 to INT5, INT6, INT7 (asynchronous interrupt), INTKEY,INTRTC, INTALM interrupts, even if the interrupt request level set before executing the “HALT” instruction is less than the value of the interrupt mask register, HALT mode release is executed. In this case, the interrupt is processed, and the CPU starts executing the instruction following the HALT instruction, but the interrupt request flag is held at “1”.

- Release by resetting

Release of all halt statuses is executed by resetting.

When the STOP mode is released by RESET, it is necessary to allow enough resetting time for operation of the oscillator to stabilize.

When releasing the halt mode by resetting, the internal RAM data keeps the state before the “HALT” instruction is executed. However the other settings contents are initialized. (Releasing due to interrupts keeps the state before the “HALT” instruction is executed.)

Table 3.4.5 Source of Halt state clearance and Halt clearance operation

Status of Received Interrupt			Interrupt Enabled (interrupt level) ≥ (interrupt mask)			Interrupt Disabled (interrupt level) < (interrupt mask)		
HALT mode			IDLE2	IDLE1	STOP	IDLE2	IDLE1	STOP
Source of Halt state clearance	Interrupt	INTWDT	◎	×	×	–	–	–
		INT0 to INT5 (Note1) INTKEY	◎	◎	◎*1	○	○	○*1
		INTUSB	◎	◎*2	×	○	○*2	×
		INT6 to INT7(PORT) (Note1)	◎	◎	◎*1	○	○	○*1
		INT6 to INT7(TMRB)	◎	×	×	×	×	×
		INTALM, INTRTC	◎	◎	×	○	○	×
		INTTA0 to 7, INTTP0 INTTB00 to INTTB01, INTTB10 to INTTB11 INTRX,INTTX, INTSBI INTI2S0, INTLCD, INTAD, INTADHP INTSPIRX,INTSPITX INTRSC, INTRDY INTDMA0 to INTDMA5	◎	×	×	×	×	×
	RESET		Reset initializes the LSI					

◎: After clearing the Halt mode, CPU starts interrupt processing.

○: After clearing the Halt mode, CPU resumes executing starting from instruction following the HALT instruction.

×: Cannot be used to release the halt mode.

–: The priority level (interrupt request level) of non-maskable interrupts is fixed to 7, the highest priority level. This combination is not available.

*1: Release of the HALT mode is executed after warm-up time has elapsed.

*2: 6 interrupts of all 24 INTUSB sources can release Halt state from IDLE1 mode, allowing for the construction of low power dissipation systems. However, the method of use is limited as below.

- Shift to IDLE1 mode:
  Execute Halt instruction when the flag of INT_SUS or INT_CLKSTOP is “1” (SUSPEND state)

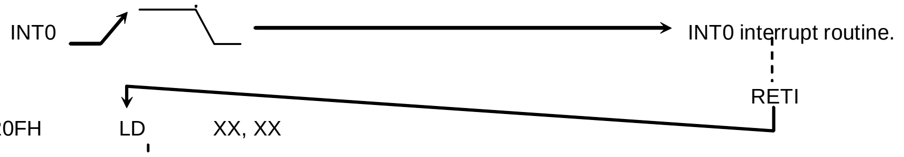
- Release from IDLE1 mode:
  Release Halt state by INT_RESUME or INT_CLKON request (release SUSPEND request)
  Release Halt state by INT_URST_STR or INT_URST_END request (RESET request)

Note: When the Halt mode is cleared by an INT0 interrupt of the level mode in the interrupt enabled status, hold level H until starting interrupt processing. If level L is set before holding level L, interrupt processing is correctly started.

(Example - releasing IDLE1 Mode)

An INT0 interrupt clears the Halt state when the device is in IDLE1 Mode.

```
Address
8200H    LD     (PCFC), 02H       ; Sets PC1 to INT0 interrupt.
8203H    LD     (IIMC0), 00H      ; Select INT0 interrupt rising edge.
8206H    LD     (INTE0), 06H      ; Sets INT0 interrupt level to 6.
8209H    EI     5                 ; Sets CPU interrupt level to 5.
820BH    LD     (SYSCR2), 28H     ; Sets Halt mode to IDLE1 mode.
820EH    HALT                     ; Halts CPU.
```




```
820FH    LD     XX, XX
```

(3) Operation

a. IDLE2 Mode

In IDLE2 Mode, only specific internal I/O operations, as designated by the IDLE2 Setting Register, can take place. Instruction execution by the CPU stops.

Figure 3.4.7 illustrates an example of the timing for clearance of the IDLE2 Mode Halt state by an interrupt.

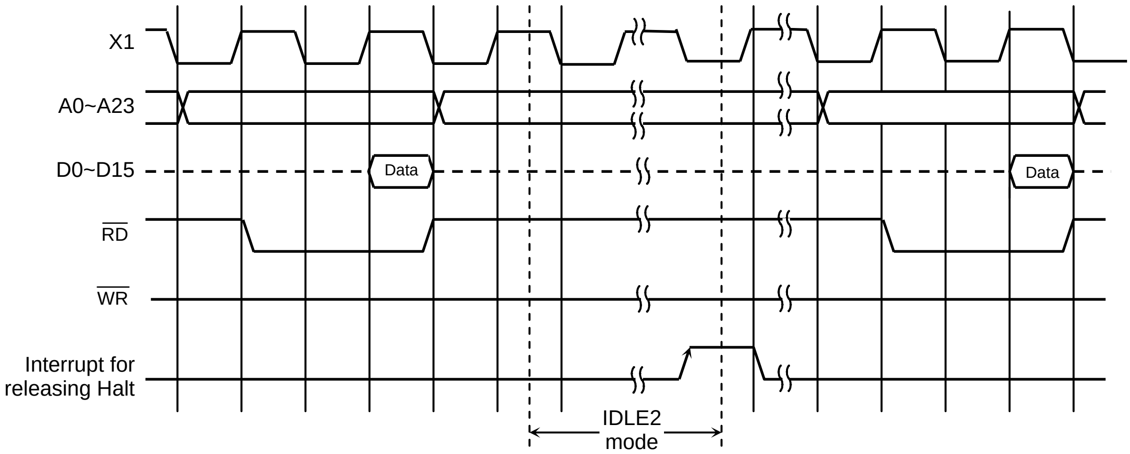


Figure 3.4.7 Timing chart for IDLE2 Mode Halt state cleared by interrupt

b. IDLE1 Mode

In IDLE1 Mode, only the internal oscillator and the RTC and MLD continue to operate. The system clock stops.

In the Halt state, the interrupt request is sampled asynchronously with the system clock; however, clearance of the Halt state (i.e. restart of operation) is synchronous with it.

Figure 3.4.8 illustrates the timing for clearance of the IDLE1 Mode Halt state by an interrupt.

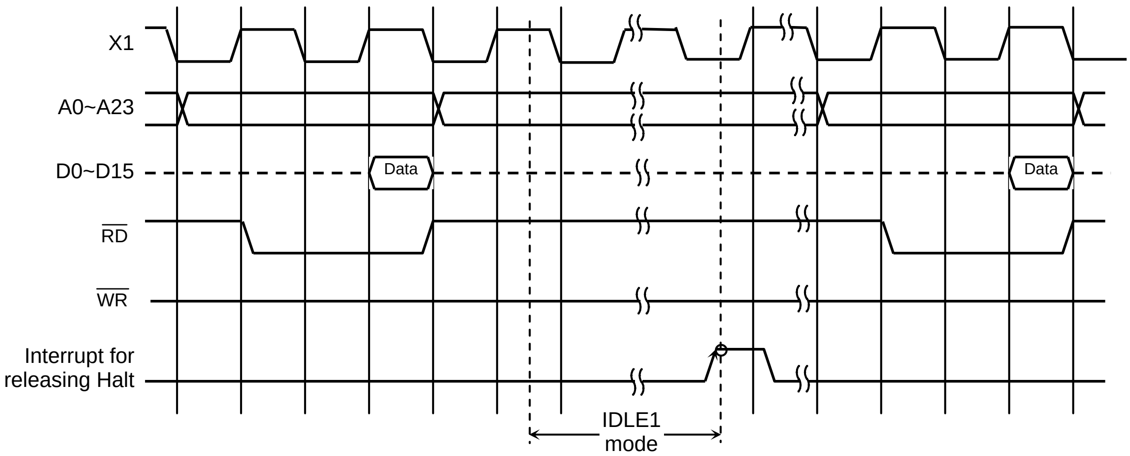


Figure 3.4.8 Timing chart for IDLE1 Mode Halt state cleared by interrupt

c. STOP Mode

When STOP Mode is selected, all internal circuits stop, including the internal oscillator.

After STOP Mode has been cleared system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize.

Figure 3.4.9 illustrates the timing for clearance of the STOP Mode Halt state by an interrupt.

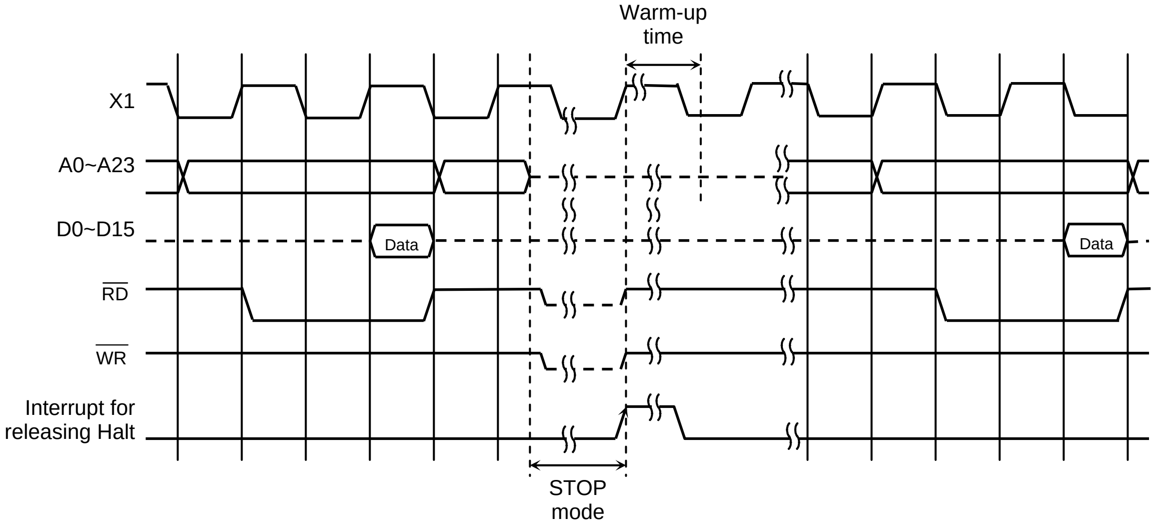


Figure 3.4.9 Timing chart for STOP Mode Halt state cleared by interrupt

Table 3.4.6 Example of warming-up time after releasing STOP-mode

at f_{OSCH} =10 MHz

SYSCR2<WUPTM1:0>		
01 (2^8)	10 (2^{14})	11 (2^{16})
25.6 μs	1.6384 ms	6.5536 ms

Table 3.4.7 Input Buffer State Table

<table>
<tr><th rowspan="4">Port Name</th><th rowspan="4">Input Function Name</th><th colspan="7">Input Buffer State</th></tr>
<tr><th rowspan="3">During Reset</th><th colspan="2" rowspan="2">When the CPU is operating</th><th colspan="4">In HALT mode (IDLE2/1/STOP)</th></tr>
<tr><th colspan="2"><PxDR>=1</th><th colspan="2"><PxDR>=0</th></tr>
<tr><th>When Used as function Pin</th><th>When Used as Input port</th><th>When Used as function Pin</th><th>When Used as Input port</th><th>When Used as function Pin</th><th>When Used as Input port</th></tr>
<tr><td>D0-D7</td><td>D0-D7</td><td>OFF</td><td rowspan="2">ON upon external read</td><td>–</td><td rowspan="2">OFF</td><td>–</td><td rowspan="2">OFF</td><td>–</td></tr>
<tr><td>P10-P17</td><td>D8-D15</td><td>16bit Start OFF
Boot Start ON</td><td rowspan="17">ON</td><td rowspan="17">ON</td><td rowspan="30">OFF</td></tr>
<tr><td>P60-P67</td><td>–</td><td>16bit Start OFF
Boot Start ON</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>P71-P74</td><td>–</td><td rowspan="15">ON</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>P75</td><td>NDR/$\overline{\text{B}}$</td><td rowspan="2">ON</td><td rowspan="2">ON</td><td rowspan="2">OFF</td></tr>
<tr><td>P76</td><td>$\overline{\text{WAIT}}$</td></tr>
<tr><td>P90</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>P91</td><td>RXD0</td><td rowspan="3">ON</td><td rowspan="3">ON</td><td rowspan="3">OFF</td></tr>
<tr><td>P92</td><td>$\overline{\text{CTS0}}$,SCLK0</td></tr>
<tr><td>P96 *1</td><td>INT4</td></tr>
<tr><td>P97</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>PA0-PA7 *1</td><td>KI0-KI7</td><td rowspan="5">ON</td><td rowspan="5">ON</td><td rowspan="5">OFF</td></tr>
<tr><td>PC0</td><td>INT0</td></tr>
<tr><td>PC1</td><td>INT1,TA0IN</td></tr>
<tr><td>PC2</td><td>INT2</td></tr>
<tr><td>PC3</td><td>INT3,TA2IN</td></tr>
<tr><td>PC4-PC7</td><td>–</td><td rowspan="2">–</td><td rowspan="2">–</td><td rowspan="2">–</td></tr>
<tr><td>PF0-PF2</td><td>–</td></tr>
<tr><td>PG0-PG2
PG4,PG5 *2</td><td>–</td><td rowspan="2">OFF</td><td>–</td><td rowspan="2">ON upon port read</td><td>–</td><td rowspan="2">OFF</td><td>–</td></tr>
<tr><td>PG3 *2</td><td>$\overline{\text{ADTRG}}$</td><td>ON</td><td>ON</td><td>ON</td></tr>
<tr><td>PJ5-PJ6</td><td>–</td><td rowspan="11">ON</td><td rowspan="3">–</td><td rowspan="11">ON</td><td rowspan="3">–</td><td rowspan="11">ON</td><td rowspan="3">–</td></tr>
<tr><td>PL0-PL7</td><td>–</td></tr>
<tr><td>PN0-PN7</td><td>–</td></tr>
<tr><td>PP3</td><td>INT5</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td rowspan="4">OFF</td></tr>
<tr><td>PP4</td><td>INT6,TB0IN0</td></tr>
<tr><td>PP5</td><td>INT7,TB1IN0</td></tr>
<tr><td>PR0</td><td>SPDI</td></tr>
<tr><td>PR1-PR3</td><td>–</td><td rowspan="2">–</td><td rowspan="2">–</td><td rowspan="2">–</td></tr>
<tr><td>PT0-PT7</td><td>–</td></tr>
<tr><td>PV6-PV7</td><td>SDA, SCL</td><td rowspan="2">ON</td><td rowspan="2">ON</td><td rowspan="2">OFF</td></tr>
<tr><td>PX5</td><td>X1USB</td></tr>
<tr><td>D+, D-</td><td>–</td><td colspan="7" rowspan="4">Always ON</td></tr>
<tr><td>$\overline{\text{RESET}}$</td><td>–</td></tr>
<tr><td>AM0,AM1</td><td>–</td></tr>
<tr><td>$\overline{\text{TEST}}$</td><td>–</td></tr>
<tr><td>X1,XT1</td><td>–</td><td colspan="3"></td><td colspan="4">IDLE2/DLE1: ON</td></tr>
</table>

ON: The buffer is always turned on. A current flows through the input buffer if the input pin is not driven.

OFF: The buffer is always turned off.

– : Not applicable

*1: Port having a pull-up/pull-down resistor.

*2: AIN input does not cause a current to flow through the buffer.

Table 3.4.8 Output buffer State Table (1/2)

<table>
<tr><th rowspan="4">Port Name</th><th rowspan="4">Output Function Name</th><th colspan="7">Output Buffer State</th></tr>
<tr><th rowspan="3">During Reset</th><th colspan="2" rowspan="2">When the CPU is operating</th><th colspan="4">In HALT mode (IDLE2/1/STOP)</th></tr>
<tr><th colspan="2"><PxDR> = 1</th><th colspan="2"><PxDR> = 0</th></tr>
<tr><th>When Used as function Pin</th><th>When Used as Output port</th><th>When Used as function Pin</th><th>When Used as Output port</th><th>When Used as function Pin</th><th>When Used as Output port</th></tr>
<tr><td>D0-7</td><td>D0-D7</td><td>OFF</td><td rowspan="2">ON upon external write</td><td>–</td><td rowspan="2">OFF</td><td>–</td><td rowspan="11">OFF</td><td>–</td></tr>
<tr><td>P10-17</td><td>D8-D15</td><td>16bit Start OFF
Boot Start OFF</td><td>ON</td><td>ON</td><td rowspan="22">OFF</td></tr>
<tr><td>P40-P47</td><td>A0-A7</td><td rowspan="2">ON</td><td rowspan="9">ON</td><td rowspan="21">ON</td><td rowspan="9">ON</td><td rowspan="21">ON</td></tr>
<tr><td>P50-P57</td><td>A8-A15</td></tr>
<tr><td>P60-67</td><td>A16-A23</td><td>16bit Start ON
Boot Start OFF</td></tr>
<tr><td>P70</td><td>$\overline{RD}$</td><td>ON</td></tr>
<tr><td>P71</td><td>$\overline{WRLL}$, $\overline{NDRE}$</td><td rowspan="6">OFF</td></tr>
<tr><td>P72</td><td>$\overline{WRLU}$, $\overline{NDWE}$</td></tr>
<tr><td>P73</td><td>EA24</td></tr>
<tr><td>P74</td><td>EA25</td></tr>
<tr><td>P75</td><td>R/ $\overline{W}$</td></tr>
<tr><td>P76</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>P80</td><td>$\overline{CS0}$</td><td rowspan="8">ON</td><td rowspan="9">ON</td><td rowspan="9">ON</td><td rowspan="9">OFF</td></tr>
<tr><td>P81</td><td>$\overline{CS1}$, $\overline{SDCS}$</td></tr>
<tr><td>P82</td><td>$\overline{CS2}$, $\overline{CSZA}$ $\overline{SDCS}$</td></tr>
<tr><td>P83</td><td>$\overline{CS3}$, $\overline{CSXA}$</td></tr>
<tr><td>P84</td><td>$\overline{CSZB}$</td></tr>
<tr><td>P85</td><td>$\overline{CSZC}$</td></tr>
<tr><td>P86</td><td>$\overline{CSZD}$, $\overline{ND0CE}$</td></tr>
<tr><td>P87</td><td>$\overline{CSXB}$, $\overline{ND1CE}$</td></tr>
<tr><td>P90</td><td>TXD0</td><td rowspan="3">OFF</td></tr>
<tr><td>P91</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>P92</td><td>SCLK0</td><td rowspan="3">ON</td><td rowspan="3">ON</td><td rowspan="3">OFF</td></tr>
<tr><td>P96</td><td>PX</td><td rowspan="10">OFF</td><td rowspan="2">–</td><td rowspan="2">–</td><td rowspan="2">–</td></tr>
<tr><td>P97</td><td>PY</td></tr>
<tr><td>PC0-PC3</td><td>–</td><td>–</td><td rowspan="9">ON</td><td>–</td><td rowspan="9">ON</td><td>–</td><td rowspan="9">OFF</td></tr>
<tr><td>PC4</td><td>EA26</td><td rowspan="27">ON</td><td rowspan="27">ON</td><td rowspan="27">OFF</td></tr>
<tr><td>PC5</td><td>EA27</td></tr>
<tr><td>PC6</td><td>EA28</td></tr>
<tr><td>PC7</td><td>KO8</td></tr>
<tr><td>PF0</td><td>I2S0CKO</td></tr>
<tr><td>PF1</td><td>I2S0DO</td></tr>
<tr><td>PF2</td><td>I2S0WS</td></tr>
<tr><td>PF7</td><td>SDCLK</td><td>ON</td></tr>
<tr><td>PG2</td><td>MX</td><td rowspan="2">OFF</td><td rowspan="2">–</td><td rowspan="2">–</td><td rowspan="2">–</td></tr>
<tr><td>PG3</td><td>MY</td></tr>
<tr><td>PJ0</td><td>$\overline{SDRAS}$, $\overline{SRLLB}$</td><td rowspan="5">ON</td><td rowspan="17">ON</td><td rowspan="17">ON</td><td rowspan="17">OFF</td></tr>
<tr><td>PJ1</td><td>$\overline{SDCAS}$, $\overline{SRLUB}$</td></tr>
<tr><td>PJ2</td><td>$\overline{SDWE}$, $\overline{SRWR}$</td></tr>
<tr><td>PJ3</td><td>SDLLDQM</td></tr>
<tr><td>PJ4</td><td>SDLUDQM</td></tr>
<tr><td>PJ5</td><td>NDALE</td><td rowspan="2">OFF</td></tr>
<tr><td>PJ6</td><td>NDCLE</td></tr>
<tr><td>PJ7</td><td>SDCKE</td><td rowspan="9">ON</td></tr>
<tr><td>PK0</td><td>LCP0</td></tr>
<tr><td>PK1</td><td>LLOAD</td></tr>
<tr><td>PK2</td><td>LFR</td></tr>
<tr><td>PK3</td><td>LVSYNC</td></tr>
<tr><td>PK4</td><td>LHSYNC</td></tr>
<tr><td>PK5</td><td>LGOE0</td></tr>
<tr><td>PK6</td><td>LGOE1</td></tr>
<tr><td>PK7</td><td>LGOE2</td></tr>
<tr><td>PL0-PL7</td><td>LD0-LD7</td><td>OFF</td></tr>
</table>

Table 3.4.9 Output buffer state table (2/2)

<table>
<tr><th rowspan="4">Port Name</th><th rowspan="4">Output Function Name</th><th colspan="7">Output Buffer State</th></tr>
<tr><th rowspan="3">During Reset</th><th colspan="2" rowspan="2">When the CPU is operating</th><th colspan="4">In HALT mode (IDLE2/1/STOP)</th></tr>
<tr><th colspan="2"><PxDR>=1</th><th colspan="2"><PxDR>=0</th></tr>
<tr><th>When Used as function Pin</th><th>When Used as Output port</th><th>When Used as function Pin</th><th>When Used as Output port</th><th>When Used as function Pin</th><th>When Used as Output port</th></tr>
<tr><td>PM1</td><td>MLDALM,TA1OUT</td><td rowspan="3">ON</td><td rowspan="5">ON</td><td rowspan="17">ON</td><td rowspan="5">ON</td><td rowspan="17">ON</td><td rowspan="5">OFF</td><td rowspan="17">OFF</td></tr>
<tr><td>PM2</td><td>$\overline{\text{MLDALM}}$, $\overline{\text{ALARM}}$</td></tr>
<tr><td>PM7</td><td>PWE</td></tr>
<tr><td>PN0-PN7</td><td>KO0-KO7</td><td rowspan="3">OFF</td></tr>
<tr><td>PP3</td><td>TA7OUT</td></tr>
<tr><td>PP4-PP5</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>PP6</td><td>TB0OUT0</td><td rowspan="2">ON</td><td rowspan="2">ON</td><td rowspan="2">ON</td><td rowspan="2">OFF</td></tr>
<tr><td>PP7</td><td>TB1OUT0</td></tr>
<tr><td>PR0</td><td>–</td><td rowspan="5">OFF</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>PR1</td><td>SPDO</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td rowspan="4">OFF</td></tr>
<tr><td>PR2</td><td>$\overline{\text{SPCS}}$</td></tr>
<tr><td>PR3</td><td>SPCLK</td></tr>
<tr><td>PT0-PT7</td><td>LD8-LD15</td></tr>
<tr><td>PV6</td><td>SDA</td><td rowspan="2">OFF</td><td rowspan="2">ON</td><td rowspan="2">ON</td><td rowspan="2">OFF</td></tr>
<tr><td>PV7</td><td>SCL</td></tr>
<tr><td>PX4</td><td>CLKOUT, LDIV</td><td>ON</td><td>ON</td><td>ON</td><td>OFF</td></tr>
<tr><td>PX5</td><td>–</td><td>OFF</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>D+, D–</td><td>–</td><td>OFF</td><td colspan="6">ON/OF depend on USBC operation</td></tr>
<tr><td>X2</td><td>–</td><td colspan="5" rowspan="2">Always ON</td><td colspan="2">IDLE2/1:ON,
STOP: output "H"</td></tr>
<tr><td>XT2</td><td>–</td><td colspan="2">IDLE2/1:ON,
STOP: output "HZ"</td></tr>
</table>

ON: The buffer is always turned on. When the bus is released, however, output buffers for some pins are turned off.

*1: Port having a pull-up/pull-down resistor.

OFF: The buffer is always turned off.

– : Not applicable

3.5 Boot ROM

The TMP92CF29A contains boot ROM for downloading a user program, and supports two kinds of downloading methods.

3.5.1 Operation Modes

The TMP92CF29A has two operation modes: MULTI mode and BOOT mode. The operation mode is selected according to the AM1 and AM0 pin levels when $\overline{\text{RESET}}$ is asserted.

(1) MULTI mode: After reset, the CPU fetches instructions from external memory and executes them.

(2) BOOT mode: After reset, the CPU fetches instructions from internal boot ROM and executes them. The boot ROM loads a user program into internal RAM from USB, or via UART, and then branches to the internal RAM. In this way the user program starts boot operation. Table 3.5.2 shows an outline of boot operation.

Table 3.5.1 Operation Modes

Mode Setting Pins			Operation Mode	
$\overline{\text{RESET}}$	AM1	AM0		
	0	1	MULTI	Start from external 16-bit bus memory
	1	0	TEST (Setting prohibited)	
	1	1	BOOT (Start from internal boot ROM)	
	0	0	TEST (Setting prohibited)	

Table 3.5.2 Outline of Boot Operation

Name	Priority	Loading			Operation after Loading
		Source	I/F	Destination	
(a)	1	PC (UART)	UART	Internal RAM	Branch to internal RAM
(b)	2	PC (USB_HOST)	USB		

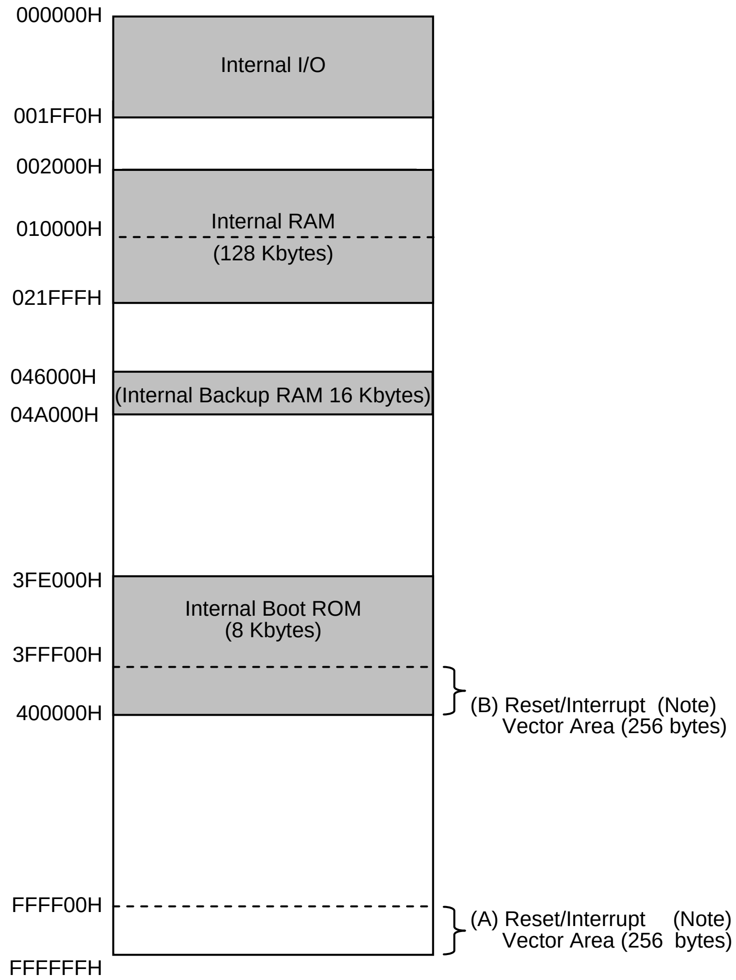
3.5.2 Hardware Specifications of Internal Boot ROM

(1) Memory map

Figure 3.5.1 shows a memory map of BOOT mode.

The boot ROM incorporated in the TMP92CF29A is an 8-Kbyte ROM area mapped to addresses 3FE000H to 3FFFFFH.

In MULTI mode, the boot ROM is not mapped and the above area is mapped as an external area.



Note: BROMCR<VACE> = "1" : (B) when booting

BROMCR<VACE> = "0" : (A) when multi mode

Figure 3.5.1 Memory Map of BOOT Mode

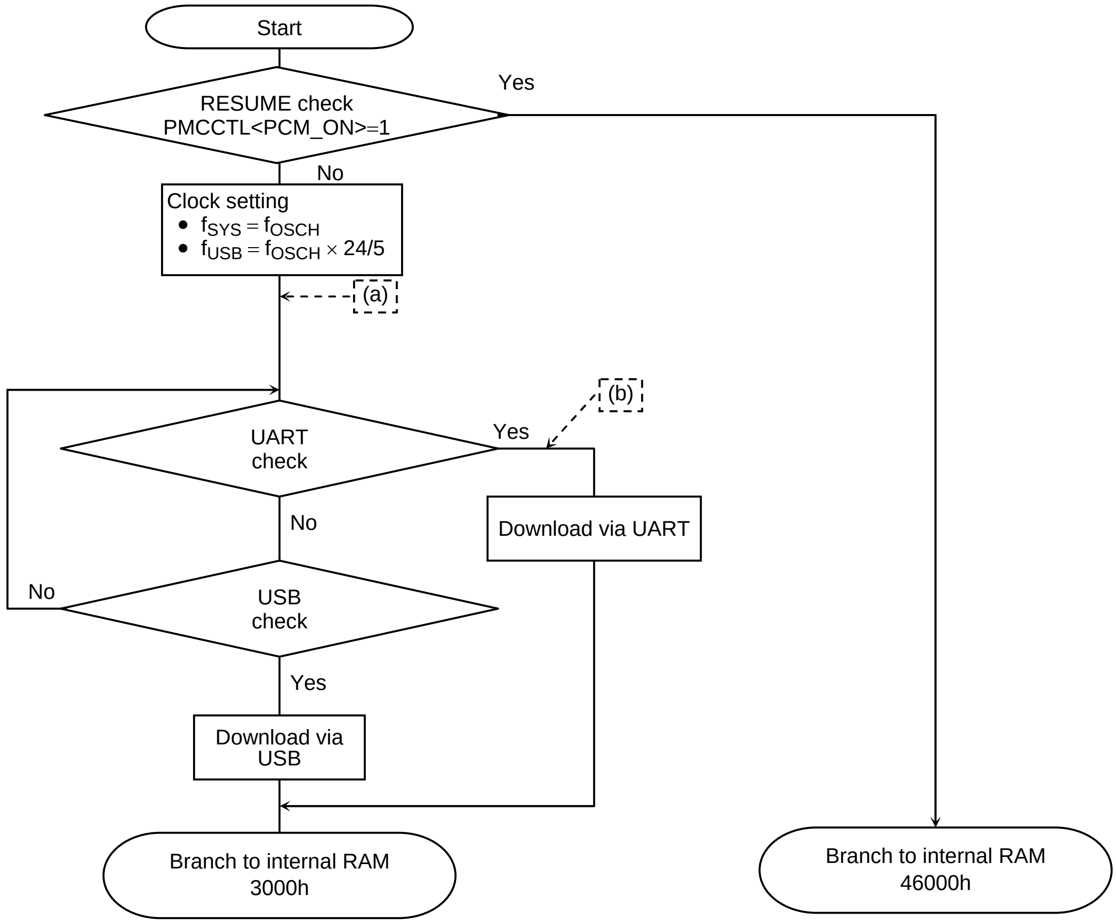
(2) Switching the boot ROM area to an external area

After the boot sequence is executed in BOOT mode, an application system program may start running without a reset being asserted. In this case, it is possible to switch the boot ROM area to an external area.

3.5.3 Outline of Boot Operation

The method for downloading a user program can be selected from two types: from UART, or via USB.

After reset, the boot program on the internal boot ROM executes as shown in Figure 3.5.2. Regardless of the downloading method used, the boot program downloads a user program into the internal RAM and then branches to the internal RAM. Figure 3.5.3 shows how the boot program uses the internal RAM (common to all the downloading methods).



Note 1: To download a user program via USB, a USB device driver and special application software are needed on the PC.

Note 2: To download a user program via UART, special application software is needed on the PC.

Note 3: The (a), (b) in the above flowchart indicate points where the settings of external port pins are changed. For details, see Table 3.5.3.

Figure 3.5.2 Flowchart for Internal Boot ROM Operation

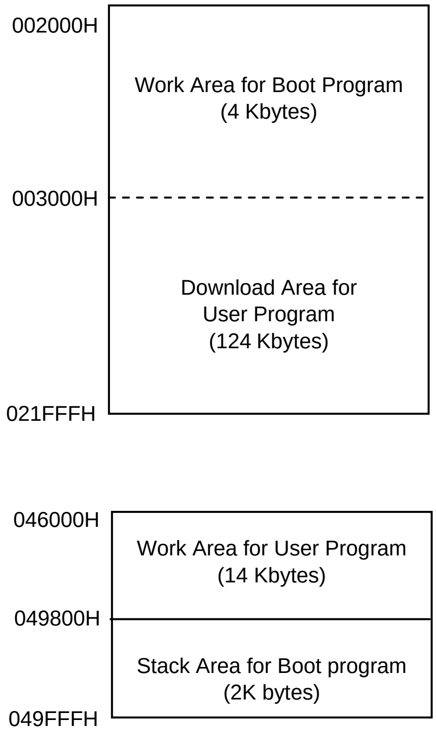


Figure 3.5.3 How the Boot Program Uses Internal RAM

(1) Port settings

Table 3.5.3 shows the port settings by the boot program. When designing your application system, please also refer to Table 3.5.4 for recommended pin connections for using the boot program.

The boot program only sets the ports shown in the table below; other ports are left as they are after reset or at startup of the boot program.

Table 3.5.3 Port Settings by the Boot Program

<table>
<tr><th colspan="2" rowspan="2">Port Name</th><th rowspan="2">Function Name</th><th rowspan="2">I/O</th><th colspan="3">Description</th></tr>
<tr><th>(a)</th><th>(b)</th><th>(c)</th></tr>
<tr><td rowspan="2">UART</td><td>P90</td><td>TXD0</td><td>Output</td><td>No change from after reset state (input port)</td><td rowspan="2">No change from (a)</td><td>Set as TXD0 output pin</td></tr>
<tr><td>P91</td><td>RXD0</td><td>Input</td><td>Set as RXD0 input pin</td><td>No change from (b)</td></tr>
<tr><td rowspan="3">USB</td><td>–</td><td>D+</td><td>I/O</td><td colspan="3" rowspan="2">No change</td></tr>
<tr><td>–</td><td>D–</td><td>I/O</td></tr>
<tr><td>PP6</td><td>PUCTL</td><td>Output</td><td>No change from after reset state (input port)</td><td>Set to “1”</td><td>No change from (b)</td></tr>
</table>

Table 3.5.4 Recommended Pin Connections

Port Name		Function Name	I/O	Recommended Pin Connections for Each Download Method	
				UART	USB
UART	P90	TXD0	Output	Connect to the level shifter.	No special setting is needed for booting via USB.
	P91	RXD0	Input		Add a pull-up resistor (100 kΩrecommended) to prevent transition to UART processing.
USB	–	D+	I/O	No special setting is needed for booting via UART.	Connect to the USB connector by adding a dumping resistor (27Ωrecommended) and a programmable pull-up resistor (1.5 kΩrecommended). When USB is not accessed, the pin level should be fixed with a resistor to prevent flow-through current.
	–	D–	I/O	If USB is not used, add a pull-up or pull-down resistor to prevent flow-through current on the D+/D- pins.	Connect to the USB connector by adding a dumping resistor (27Ω recommended). When USB is not accessed, the pin level should be fixed with a resistor to prevent flow-through current.
	PP6	PUCTL	Output	–	This pin is used to control ON/OFF of the D+ pin's pull-up resistor. Add a switch externally so that the pull-up is turned on when "1".

Note 1: When a user program is downloaded from UART and USB is used in the system, the pull-up resistor for USB's D+ pin should not be turned on in BOOT mode.

Note 2: When a user program is downloaded via USB, do not start the UART application software on the PC.

Note 3: When a user program is downloaded via UART, do not connect a USB connector.

Note 4: When USB is not used, the D+ and D- pins must be pulled up or down to prevent flow-through current.

(2) I/O register settings

Table 3.5.5 shows the I/O registers that are set by the boot program.

After the boot sequence, if execution moves to an application system program without a reset being asserted, the settings of these I/O registers must be taken into account. Also note that the registers in the CPU and the internal RAM remain in the state after execution of the boot program.

Table 3.5.5 I/O Register Settings by Boot Program

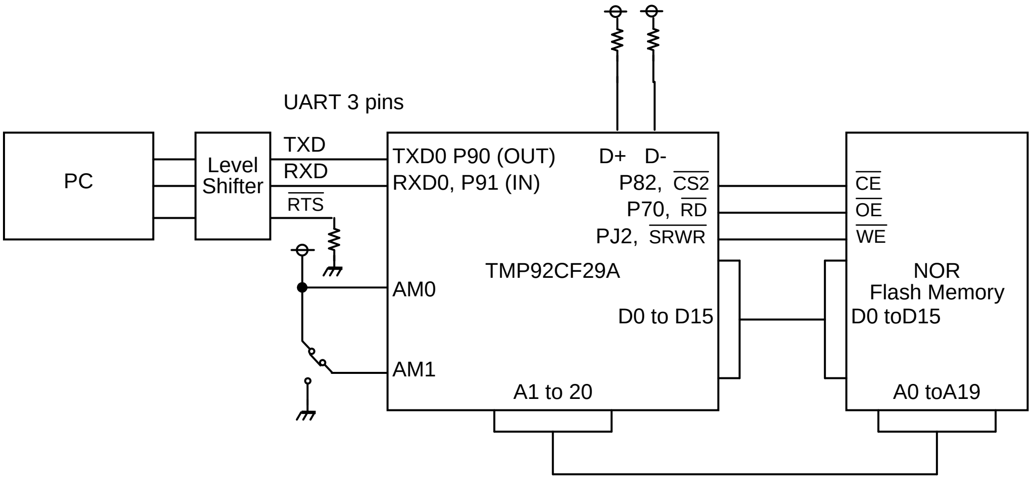
Register Name	Set Value	Description
WDMOD	00H	Watchdog timer not active
WDCR	B1H	Watchdog timer disabled
SYSCR0	70H	High-frequency and low-frequency oscillators operating
SYSCR1	00H	Clock gear = 1/1
SYSCR2	2CH	Initial value
PLLCR0	00H	PLL clock not used
PLLCR1	00H or 60H	Normally PLL is disabled. However, only in the case of booting via USB, PLL is activated for USB.
INTEUSB	04H	USB interrupt level setting
INTETC01	44H	INTTC interrupt level setting

Note: The values to be set in the I/O registers for UART and USB are not described here. If these functions are needed in a user program, set each I/O register as necessary.

3.5.4 Downloading a User Program via UART

(1) Connection example

Figure 3.5.4 shows an example of connections for downloading a user program via UART (using a 16-bit NOR Flash memory device as program memory).



Note: When USB is not used, add a pull-up or pull-down resistor to the D+ and D- pins to prevent flow-through current.

Figure 3.5.4 UART Connection Example

(2) UART interface specifications

SIO channel 0 is used for downloading a user program.

The UART communication format in BOOT mode is shown below. Before booting, the PC must also be set up with the same conditions.

Although the default baud rate is 9600 bps, this can be changed as shown in Table 3.5.8.

Serial transfer mode:	:	UART (asynchronous) mode, full-duplex
Data length	:	8 bits
Parity bit	:	None
STOP bit	:	1 bit
Handshake	:	None
Baud rate (default)	:	9600 bps

(3) UART data transfer format

Table 3.5.6 to Table 3.5.11 show the supported frequencies, data transfer format, baud rate modification command, operation command, and version management information, respectively.

Please also refer to the description of boot program operation later in this section.

Table 3.5.6 Supported Frequencies (X1)

6.00 MHz	8.00 MHz	9.00 MHz	10.00 MHz

Note: The built-in PLL (clock multiplier) is not used regardless of the oscillation frequency.

Table 3.5.7 Transfer Format

	Byte Number to Transfer	Transfer data from PC to TMP92CF29A	Baud Rate	Transfer data from TMP92CF29A to PC
Boot ROM	1st byte	Matching data (5AH)	9600 bps	– (Frequency measurement and baud rate auto setting)
	2nd byte	–		OK: Echo back data (5AH) Error: No transfer
	3rd byte to 6th byte	–		Version management information (See Table 3.5.10)
	7th byte	–		Frequency information
	8th byte	Baud rate modification command (See Table 3.5.8.)		–
	9th byte	–		OK: Echo back data Error: Error code × 3
	10th byte to (n – 4)th byte	User program Intel Hex format (binary)	New baud rate	NG: Operation stop by checksum error
	(n – 3)th byte	–		OK: SUM (High) (See (4)-c).)
	(n – 2)th byte	–		OK: SUM (Low)
	(n – 1)th byte	User program start command (C0H) (See Table 3.5.9.)		–
	n'th byte	–		OK: Echo back data (C0H) Error: Error code ×3
RAM	–	Branch to user program start address		

"Error code x 3" means that the error code is transmitted three times. For example, if the error code is 62H, the TMP92CF29A transmits 62H three times. For error codes, see (4)-b).

Table 3.5.8 Baud Rate Modification Command

Baud Rate (bps)	9600	19200	38400	57600	115200
Modification Command	28H	18H	07H	06H	03H

Note 1: If f_{OSCH} (oscillation frequency) is 10.0 MHz, 57600 and 115200 bps are not supported.

Note 2: If f_{OSCH} (oscillation frequency) is 6.00, 8.00, or 9.00 MHz, 38400, 57600, and 115200 bps are not supported.

Table 3.5.9 Operation Command

Operation Command	Operation
C0H	User program start

Table 3.5.10 Version Management Information

Version Information	ASCII Code
FRM1	46H, 52H, 4DH, 31H

Table 3.5.11 data of measuring frequency

X1-X2 oscillator frequency (MHz)	6.000	8.000	9.000	10.000
	09H	0AH	08H	0BH

(4) Description of the UART boot program operation

The boot program receives a user program sent from the PC via UART and transfers it to the internal RAM. If the transfer ends normally, the boot program calculates SUM and sends the result to the PC before executing the user program. The execution start address is the first address received. The boot program enables users to perform customized on-board programming.

When UART is used to download a user program, the maximum allowed program size is 124 Kbytes (3000H – 21FFFH). (The extended Intel Hex format is supported.)

a) Operation procedure

1. Connect the serial cable. This must be done before the microcontroller is reset.
2. Set the AM1 and AM0 pins to “1” and reset the microcontroller.
3. The receive data in the 1st byte is matching data (5AH). Upon starting in BOOT mode, the boot program goes to a state in which it waits for matching data. When matching data is received, the initial baud rate of the serial channel is automatically set to 9600 bps.
4. The 2nd byte is used to echo back 5AH to the PC upon completion of the automatic baud rate setting in the 1st byte. If automatic baud rate setting fails, the boot program stops operation.
5. The 3rd through 6th bytes are used to send the version management information of the boot program in ASCII code. The PC should check that the correct version of the boot program is used.
6. The 7th byte is used to send information on the measured frequency. The PC should check that the frequency of the resonator is measured correctly.

7. The receive data in the 8th byte is baud rate modification data. The five kinds of baud rate modification data shown in Table 3.5.8 are available. Even when the baud rate is not changed, the initial baud rate data (28H: 9600 bps) must be sent. Baud rate modification becomes effective after the echo back transmission is completed.
8. The 9th byte is used to echo back the received data to the PC when the data received in the 8th byte is one of the baud rate modification data corresponding to the operating frequency of the microcontroller. Then, the baud rate is changed. If the received baud rate data does not correspond to the operating frequency, the boot program stops operation after sending the baud rate modification error code (62H).
9. The receive data in the 10th to (n-4)th bytes is received as binary data in Intel Hex format. No echo back data is returned to the PC.
   The boot program ignores received data and does not send error code to the PC until it receives the start mark (3AH for ":") of Intel Hex format. After receiving the start mark, the boot program receives a range of data from record length to checksum and writes the received data to the specified RAM addresses successively.
   If a receive error or checksum error occurs, the boot program stops operation without sending error code to the PC.
   The boot program executes the SUM calculation routine upon detecting the end record. Thus, after sending the end record, the PC should be placed in a state in which it waits for SUM data.
10. The (n-3)th and (n-2)th bytes are used to send the SUM value to the PC in the order of upper byte and lower byte. For details on how to calculate SUM, see "SUM calculation" to be described later. SUM calculation is performed after detecting the end record only when no receives error or checksum error has occurred. Immediately after SUM calculation is completed, the boot program sends the SUM value to the PC. After sending the end record, the PC should determine whether or not writing to RAM has completed successfully based on whether or not the SUM value is received from the boot program.
11. After sending the SUM value, the boot program waits for the user program start command (C0H). If the SUM value is correct, the PC should send the user program start command in the (n-1)th byte.
12. The n'th byte is used to echo back the user program start command to the PC. After sending the echo back data, the boot program sets the stack pointer to 4A000H and jumps to the address that is received first as Intel Hex format data.
13. If the user program start command is not correct or a receive error has occurred, the boot program stops operation after sending the error code to the PC three times.

b) Error codes

The boot program uses the error codes shown in Table 3.5.12 to notify the PC of its processing status.

Table 3.5.12 Error Codes

Error Code	Meaning
62H	Unsupported baud rate
64H	Invalid operation command
A1H	Framing error in received data
A3H	Overrun error in received data

Note 1: If a receive error occurs while a user program is being received, no error code will be sent to the PC.

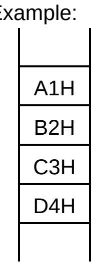
Note 2: After sending an error code, the boot program stops operation.

c) SUM calculation

1. Calculation method

SUM is calculated by adding data in bytes and is returned in words, as explained below.

Example:



If the data to be calculated consists of the 4 bytes shown to the left, SUM is calculated as follows:

A1H + B2H + C3H + D4H = 02EAH
SUM (HIGH) = 02H
SUM (LOW) = EAH

2. Data to be calculated

SUM is calculated from the data at the first received address through the last received address.

Even if received addresses are not continuous, unwritten addresses are also included in SUM calculation. The user program should not contain unwritten gaps.

d) Notes on Intel Hex format (binary)

1. After receiving the checksum of a record, the boot program waits for the start mark (3AH for ":") of the next record. If data other than 3AH is received between records, it is ignored.
2. Once the PC program has finished sending the checksum of an end record, it must wait for 2 bytes of data (upper and lower bytes of SUM) before sending any other data. This is because after receiving the checksum of an end record, the boot program calculates SUM and returns the result to the PC in 2 bytes.
3. Writing to areas other than internal RAM may cause incorrect operation. To transfer a record, set the paragraph address to 0000H.
4. Since the address pointer is initially set to 00H, the record type to be transferred first does not have to be an address record.
5. Addresses 3000H to 21FFFH are allocated as the user program download area.
6. A user program in Intel Hex format (ASCII codes) must be converted into binary data in advance, as explained in the example below.

Example: How to convert an Intel Hex file into binary format

The following shows how an Intel Hex format file is displayed on a text editor.

: 103000000607F100030000F201030000B1F16010B7
: 00000001FF

However, the actual data consists of ASCII codes, as shown below.

3A3130333030303030303630374631303030333030303046323031303330303030
423146313630313042370D0A3A303030303030303146460D0A

Thus, the ASCII codes must be converted into binary data based on the conversion rules shown in the table below.

ASCII Code	Binary Data
3A	3A (Only 3A remains the same.)
30 to 39	0 to 9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete

<u>Intel Hex format</u>

Data record: 3A 10 3000 00 0607F100030000F201030000B1F16010 B7

- 0607F100030000F201030000B1F16010: Data
- B7: Checksum
- 00: Record type
- 3000: Address
- 10: Record length
- 3A: : (Start mark)

End record: 3A 00 0000 01 FF

- FF: Data
- 01: Record type
- 0000: Address
- 00: Record length
- 3A: : (Start mark)

e) User program receive error

If either of the following error conditions occurs while a user program is being received, the boot program stops operation.

If the record type is other than 00H, 01H, or 02H

If a checksum error occurs

f) Measured frequency/baud rate error

When the boot program receives matching data, it measures the oscillation frequency. If an error is within plus or minus 3%, the boot program decides on that frequency.

Each baud rate includes a setting error as shown in Table 3.5.13. For example, in the case of 10.00 MHz /9600 bps, the baud rate is actually set at 9615.38 bps. To establish communication, the sum of the baud rate setting error and the measured frequency error must be within plus or minus 3 %.

Table 3.5.13 Baud Rate Setting Errors (%)

	9600 bps	19200 bps	38400 bps	57600 bps	115200 bps
6.000 MHz	0.2	0.2	–	–	–
8.000 MHz	0.2	0.2	–	–	–
9.000 MHz	0.2	–0.7	–	–	–
10.000 MHz	0.2	0.2	–1.4	–	–

–: Not supported

(5) Others

a) Handshake function

Although the $\overline{CTS}$ pin is available in the TMP92CF29A, the boot program does not use it for transfer control.

b) RS-232C connector

The RS-232C connector must not be connected or disconnected while the boot program is running.

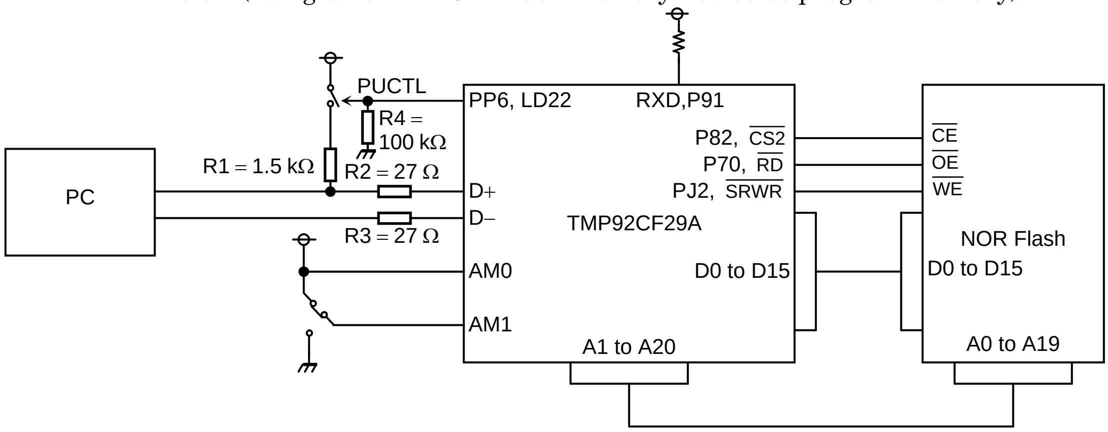
c) Software on the PC

When downloading a user program via UART, special application software is needed on the PC.

3.5.5 Downloading a User Program via USB

(1) Connection example

Figure 3.5.5 shows an example of connections for downloading a user program via USB (using a 16-bit NOR Flash memory device as program memory).



Note 1: The value of pull-up and pull-down resistors are recommended values.

Note 2: The PP6 pin is assigned as PUCTL (pull-up control) output for USB.

Note 3: Since the input gates of the D+ and D- pins are always open even at unused (unaccessed) times, these pins must be set to a fixed level to prevent flow-through current. Although the level setting is not specified in the above diagram, be sure to fix the level of the D+ and D- pins by referring to the chapter on USB.

Figure 3.5.5 USB Connection Example

(2) USB interface specifications

When a user program is downloaded via USB, the oscillation frequency should be set to 10.00 MHz. The transfer speed should be fixed to full speed (12 Mbps).

The boot program uses the following two transfer types.

Table 3.5.14 Transfer Types Used by the Boot Program

Transfer Type	Description
Control Transfer	Used for transmitting standard requests and vendor requests.
Bulk Transfer	Used for responding to vendor requests and transmitting a user program.

The following shows an overview of the USB communication flow.

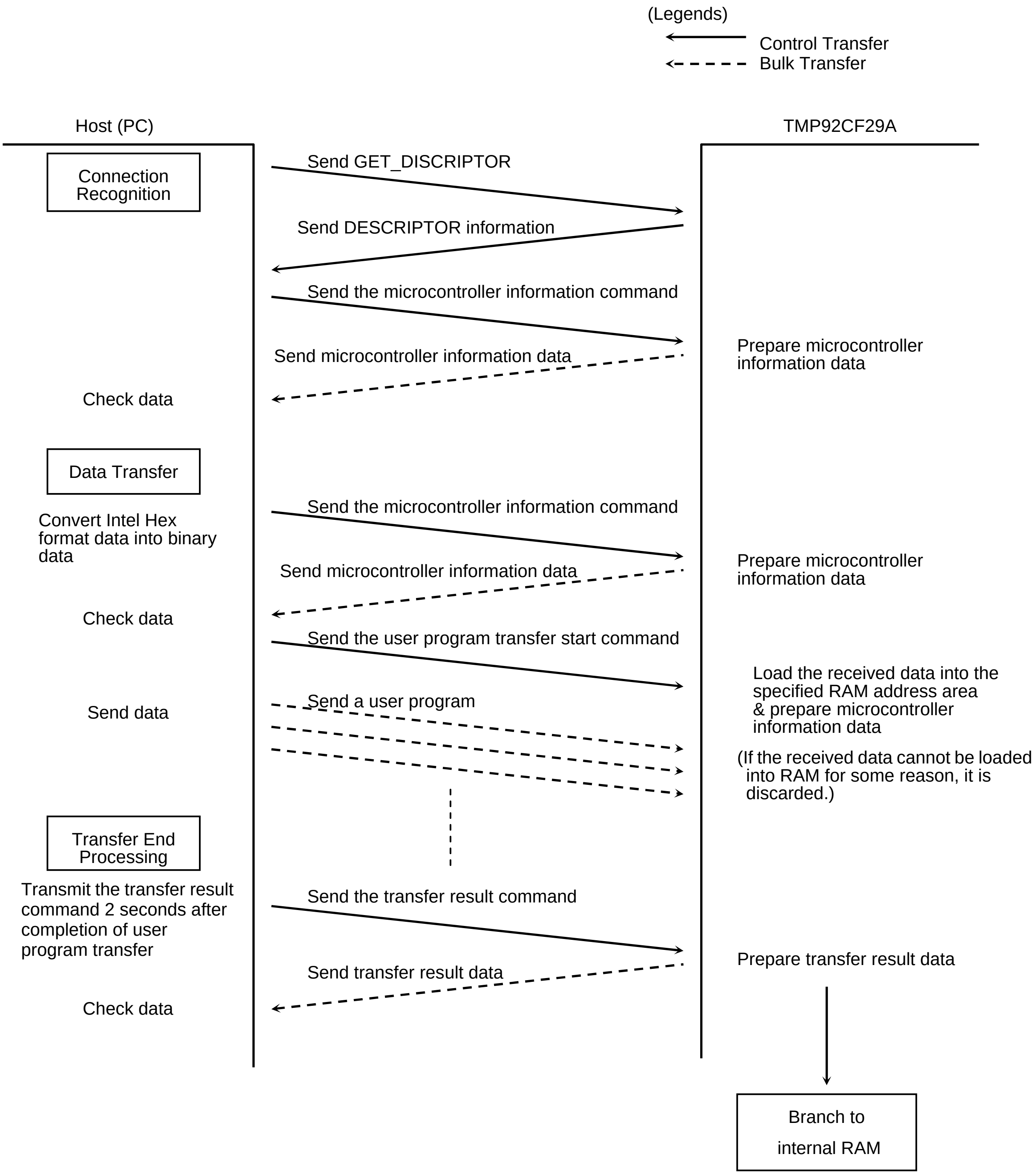


Figure 3.5.6 Overall Flowchart

Table 3.5.15 Vendor Request Commands

Command Name	Value of bRequest	Operation	Notes
Microcontroller information command	00H	Send microcontroller information	Microcontroller information data is sent by bulk IN transfer after the setup stage is completed.
User program transfer start command	02H	Receive a user program	Set the size of a user program in wIndex. The user program is received by bulk OUT transfer after the setup stage is completed.
User program transfer result command	04H	Send the transfer result	Transfer result data is sent by bulk IN transfer after the setup stage is completed.

Table 3.5.16 Setup Command Data Structure

Field Name	Value	Meaning
bmRequestType	40H	D7 0: Host to Device D6-D5 2: Vendor D4-D0 0: Device
bRequest	00H, 02H, 04H	00H: Microcontroller information 02H: User program transfer start 04H: User program transfer result
wValue	00H~FFFFH	Own data number (Not used by boot program)
wIndex	00H~FFFFH	User program size (Used when starting a user program transfer)
wLength	0000H	Fixed

Table 3.5.17 Standard Request Commands

Standard Request	Response Method
GET_STATUS	Automatic response by hardware
CLEAR_FEATURE	Automatic response by hardware
SET_FEATURE	Automatic response by hardware
SET_ADDRESS	Automatic response by hardware
GET_DISCRIPTOR	Automatic response by hardware
SET_DISCRIPTOR	Not supported
GET_CONFIGRATION	Automatic response by hardware
SET_CONFIGRATION	Automatic response by hardware
GET_INTERFACE	Automatic response by hardware
SET_INTERFACE	Automatic response by hardware
SYNCH_FRAME	Ignored

Table 3.5.18 Information Returned by GET_DISCRIPTOR

DeviceDescriptor

Field Name	Value	Meaning
Blength	12H	18 bytes
BdescriptorType	01H	Device descriptor
BcdUSB	0110H	USB Version 1.1
BdeviceClass	00H	Device class (Not in use)
BdeviceSubClass	00H	Sub command (Not in use)
BdeviceProtocol	00H	Protocol (Not in use)
BmaxPacketSize0	40H	EP0 maximum packet size (64 bytes)
IdVendor	0930H	Vendor ID
IdProduct	6504H	Product ID (0)
BcdDevice	0001H	Device version (v0.1)
Imanufacturer	00H	Index value of string descriptor indicating manufacturer name
Iproduct	00H	Index value of string descriptor indicating product name
IserialNumber	00H	Index value of string descriptor indicating product serial number
BnumConfigurations	01H	There is one configuration.

ConfigrationDescriptor

Field Name	Value	Meaning
bLength	09H	9 bytes
bDescriptorType	02H	Configuration descriptor
wTotalLength	0020H	Total length (32 bytes) which each descriptor of both configuration descriptor, interface and endpoint is added.
bNumInterfaces	01H	There is one interface.
bConfigurationValue	01H	Configuration number 1
iConfiguration	00H	Index value of string descriptor indicating configuration name (Not in use)
bmAttributes	80H	Bus power
MaxPower	31H	Maximum power consumption (49 mA)

InterfaceDescriptor

Field Name	Value	Meaning
bLength	09H	9 bytes
bDescriptorType	04H	Interface descriptor
bInterfaceNumber	00H	Interface number 0
bAlternateSetting	00H	Alternate setting number 0
bNumEndpoints	02H	There are two endpoints.
bInterfaceClass	FFH	Specified device
bInterfaceSubClass	00H	
bInterfaceProtocol	50H	Bulk only protocol
iInterface	00H	Index value of string descriptor indicating interface name (Not in use)

EndpointDescriptor

Field Name	Value	Meaning
<Endpoint1>		
blength	07H	7 bytes
bDescriptorType	05H	Endpoint descriptor
bEndpointAddress	01H	EP1= OUT
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)
<Endpoint2>		
bLength	07H	7 bytes
bDescriptor	05H	Endpoint descriptor
bEndpointAddress	82H	EP2 = IN
bmAttributes	02H	Bulk transfer
wMaxPacketSize	0040H	Payload 64 bytes
bInterval	00H	(Ignored for bulk transfer)

Table 3.5.19 Information Returned for the Microcontroller Information Command

Microcontroller Information	ASCII Code
TMP92CF29A	54H, 4DH, 50H, 39H, 32H, 43H, 46AH, 32H, 39H,20H, 20H, 20H, 20H, 20H, 20H

Table 3.5.20 Information Returned for the User Program Transfer Result Command

Transfer Result	Value	Error Conditions
No error	00H	
User program not received	02H	The user program transfer result is received without the user program transfer start command being received first.
Received file not in Intel Hex format	04H	The first data of a user program is not “:” (3AH).
User program size error	06H	The size of a received user program is larger than the value set in wIndex of the user program transfer start command.
Download address error	08H	The specified user program download address is not in the designated area.
Protocol error or other error	0AH	The user program transfer start or user program transfer result command is received first. A checksum error is detected in the Intel Hex file. A record type error is detected in the Intel Hex file. The length of an address record in the Intel Hex file is 3 or longer. The length of an end record in the Intel Hex file is other than 0.

(3) Description of the USB boot program operation

The boot program loads a user program in Intel Hex format sent from the PC into the internal RAM. When the user program has been loaded successfully, the user program starts executing from the first address received.

The boot program thus enables users to perform customized on-board programming.

a. Operation procedure

1. Connect the USB cable.
2. Set the AM0 and AM1 pins to “1” and reset the microcontroller.
3. After recognizing USB connection, the PC checks the information on the connected device using the GET_DISCRIPTOR command.
4. The PC sends the microcontroller information command by control transfer (vendor request). After the setup stage is completed, the PC checks microcontroller information data by bulk IN transfer.
5. Upon receiving the microcontroller information command, the boot program prepares microcontroller information in ASCII code.
6. The PC prepares the user program to be loaded by converting an Intel Hex file into binary format.
7. The PC sends the user program transfer start command by control transfer (vendor request). After the setup stage is completed, the PC transfers the user program by bulk OUT transfer.
8. After the user program has been transferred, the PC waits for about two seconds and then sends the user program transfer result command by control transfer (vendor request). After the setup stage is completed, the PC checks the transfer result by bulk IN transfer.
9. Upon receiving the user program transfer result command, the boot program prepares the transfer result value to be returned.
10. If the transfer result is other than OK, the boot program enters the error processing routine and will not automatically recover from it. In this case, terminate the device driver on the PC and retry from step 2.

b. Notes on the user program format (binary)

1. After receiving the checksum of a record, the boot program waits for the start mark (3AH for ":") of the next record. If data other than 3AH is received between records, it is ignored.
2. Since the address pointer is initially set to 00H, the record type to be transferred first does not have to be an address record.
3. Addresses 3000H to 21FFFH (124 Kbytes) are allocated as the user program download area. The user program should be contained within this area.

Note: In USB transfer, the size of program is set by wIndex from addresses 0000H to FFFFH. Therefore, the transferred Object size becomes 64K byte max. Please be careful.

4. A user program in Intel Hex format (normally written in ASCII code) must be converted into binary data before it can be transferred. See the example below for how to convert an Intel Hex file into binary format.

   When a user program is downloaded via USB, the maximum allowed record length is 250 bytes.

Example: Transfer data when writing 16-byte data in Intel Hex format from address 3000H

The following shows how an Intel Hex format file is displayed on a text editor.

```
: 103000000607F100030000F201030000B1F16010B7
: 00000001FF
```

However, the actual data consists of ASCII codes, as shown below.

```
3A3130333030303030303630374631303030333030303046323031303330303030
42314631363031304237 0D0A3A3030303030303031464 60D0A
```

Thus, the ASCII codes must be converted into binary data based on the conversion rules shown in the table below.

ASCII Code	Binary Data
3A	3A (Only 3A remains the same.)
30~39	0~9
41 or 61	A
42 or 62	B
43 or 63	C
44 or 64	D
45 or 65	E
46 or 66	F
0D0A	Delete

The above Intel Hex file is converted into binary data as follows:

```
Data record   3A  10  3000  00  0607F100030000F201030000B1F16010  B7
                                 ↓ Data                             Checksum
                          ↓ Record type
                   ↓ Address
               ↓ Record length
              : (Start mark\)

End record    3A  00  0000  01  FF
                                 ↓ Checksum
                          ↓ Record type
                   ↓ Address
               ↓ Record length
              : (Start mark)
```

(4) Others

a) USB connector

The USB connector must not be connected or disconnected while the boot program is running.

b) Software on the PC

To download a user program via USB, a USB device driver and special application software are needed on the PC.

3.6 Interrupts

Interrupts are controlled by the CPU Interrupt Mask Register <IFF2:0> (bits 12 to 14 of the Status Register) and by the built-in interrupt controller.

TMP92CF29A has a total of 57 interrupts divided into the following five types:

Interrupts generated by CPU: 9 sources
- Software interrupts: 8 sources
- Illegal Instruction interrupt: 1 source

Internal interrupts: 39 sources
- Internal I/O interrupts: 31 sources
- Micro DMA Transfer End interrupts /HDMA Transfer End interrupts: 6 sources
- Micro DMA Transfer End interrupts: 2 source

External interrupts: 9 sources
- Interrupts on external pins (INT0 to INT7, INTKEY)

A fixed individual interrupt vector number is assigned to each interrupt source. Any one of six levels of priority can also be assigned to each maskable interrupt. Non-maskable interrupts have a fixed priority level of 7, the highest level.

When an interrupt is generated, the interrupt controller sends the priority of that interrupt to the CPU. When more than one interrupt is generated simultaneously, the interrupt controller sends the priority value of the interrupt with the highest priority to the CPU. (The highest priority level is 7, the level used for non-maskable interrupts.)

The CPU compares the interrupt priority level which it receives with the value held in the CPU interrupt mask register <IFF2:0>. If the priority level of the interrupt is greater than or equal to the value in the interrupt mask register, the CPU accepts the interrupt.

However, software interrupts and illegal instruction interrupts generated by the CPU, and are processed irrespective of the value in <IFF2:0>.

The value in the interrupt mask register <IFF2:0> can be changed using the EI instruction (EI num sets <IFF2:0> to num). For example, the command EI3 enables the acceptance of all non-maskable interrupts and of maskable interrupts whose priority level, as set in the interrupt controller, is 3 or higher. The commands EI and EI0 enable the acceptance of all non-maskable interrupts and of maskable interrupts with a priority level of 1 or above (hence both are equivalent to the command EI1).

The DI instruction (Sets <IFF2:0> to 7) is exactly equivalent to the EI7 instruction. The DI instruction is used to disable all maskable interrupts (since the priority level for maskable interrupts ranges from 0 to 6). The EI instruction takes effect as soon as it is executed.

In addition to the general-purpose interrupt processing mode described above, there is also a micro DMA processing mode that can transfer data to internal/external memory and built-in I/O, and HDMA processing mode. In micro DMA mode the CPU, and in HDMA mode the DMA controller automatically transfers data in 1byte, 2byte or 4byte blocks. HDMA mode allows transfer faster than Micro DMA mode.

In addition, the TMP92CF29A also has a software start function in which micro DMA and HDMA processing is requested in software rather than by an interrupt. Figure 3.6.1 is a flowchart showing overall interrupts processing.

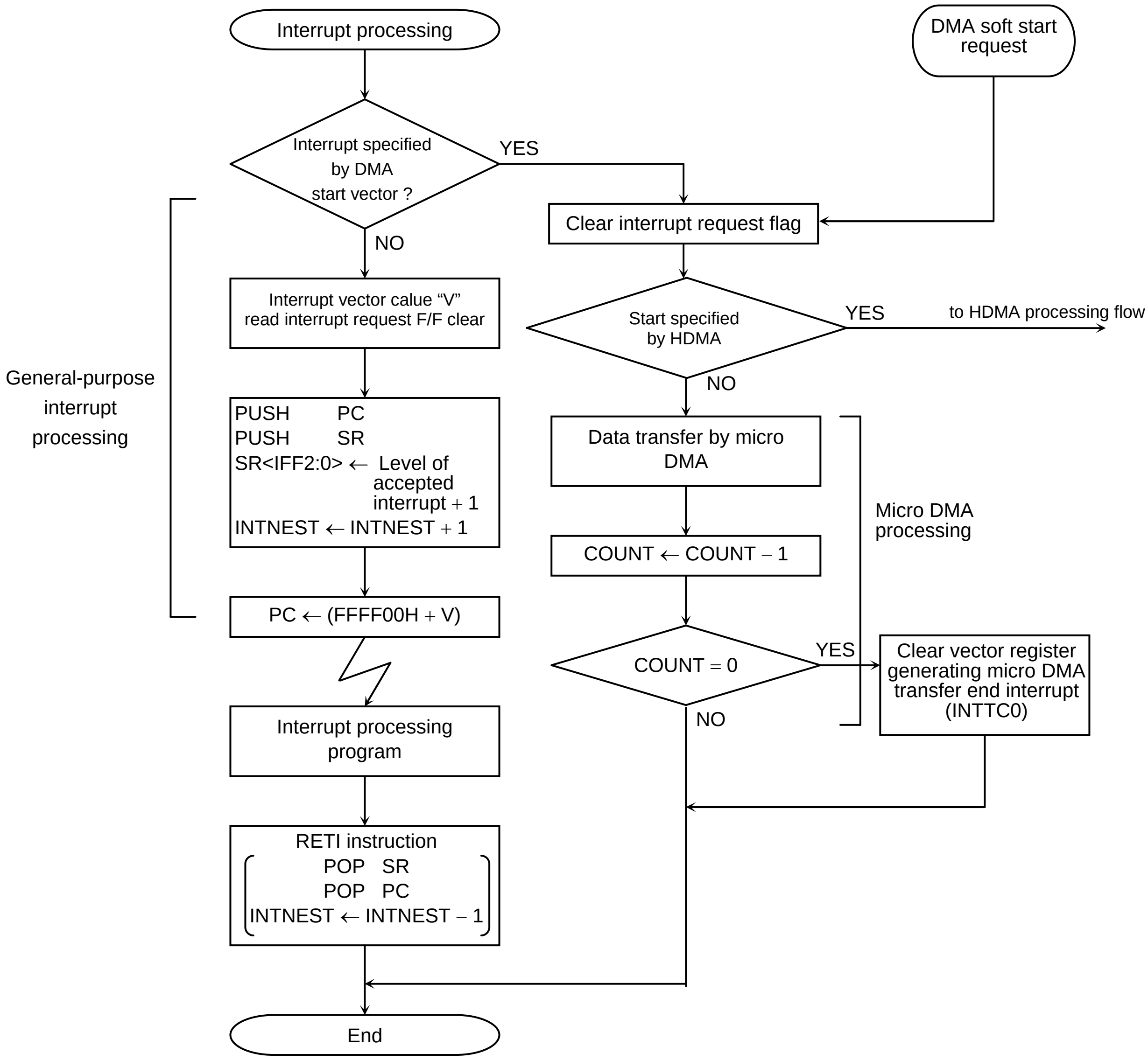


Figure 3.6.1 Interrupt processing Sequence

3.6.1 General-purpose Interrupt Processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. However, in the case of software interrupts and illegal instruction interrupts generated by the CPU, the CPU skips steps (1) and (3), and executes only steps (2), (4), and (5).

(1) The CPU reads the interrupt vector from the interrupt controller. When more than one interrupt with the same priority level has been generated simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt requests. (The default priority is determined as follows: The smaller the vector value, the higher the priority.)

(2) The CPU pushes the program counter (PC) and status register (SR) onto the top of the stack (Pointed to by XSP).

(3) The CPU sets the value of the CPU's interrupt mask register <IFF2:0> to the priority level for the accepted interrupt plus 1. However, if the priority level for the accepted interrupt is 7, the register's value is set to 7.

(4) The CPU increments the interrupt nesting counter INTNEST by 1.

(5) The CPU jumps to the address given by adding the contents of address FFFF00H + the interrupt vector, then starts the interrupt processing routine.

On completion of interrupt processing, the RETI instruction is used to return control to the main routine. RETI restores the contents of the program counter and the status register from the stack and decrements the interrupt nesting counter INTNEST by 1.

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source. (A priority level setting of 0 or 7 will disable an interrupt request.) If an interrupt request is received for an interrupt with a priority level equal to or greater than the value set in the CPU interrupt mask register <IFF2:0>, the CPU will accept the interrupt. The CPU interrupt mask register <IFF2:0> is then set to the value of the priority level for the accepted interrupt plus 1.

If during interrupt processing, an interrupt is generated with a higher priority than the interrupt currently being processed, or if, during the processing of a non-maskable interrupt processing, a non-maskable interrupt request is generated from another source, the CPU will suspend the routine which it is currently executing and accept the new interrupt. When processing of the new interrupt has been completed, the CPU will resume processing of the suspended interrupt.

If the CPU receives another interrupt request while performing processing steps (1) to (5), the new interrupt will be sampled immediately after execution of the first instruction of its interrupt processing routine. Specifying DI as the start instruction disables nesting of maskable interrupts.

A reset initializes the interrupt mask register <IFF2:0> to "111", disabling all maskable interrupts.

Table 3.6.1 shows the TMP92CF29A interrupt vectors and micro DMA start vectors. FFFF00H to FFFFFFH (256 bytes) is designated as the interrupt vector area.

Table 3.6.1 TMP92CF29A Interrupt Vectors and Micro DMA/HDMA Start Vectors

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address Refer to Vector	Micro DMA /HDMA Start Vector
1	Non maskable	Reset or [SWI0] instruction	0000H	FFFF00H	
2		[SWI1] instruction	0004H	FFFF04H	
3		Illegal instruction or [SWI2] instruction	0008H	FFFF08H	
4		[SWI3] instruction	000CH	FFFF0CH	
5		[SWI4] instruction	0010H	FFFF10H	
6		[SWI5] instruction	0014H	FFFF14H	
7		[SWI6] instruction	0018H	FFFF18H	
8		[SWI7] instruction	001CH	FFFF1CH	
9		(Reserved)	0020H	FFFF20H	
10		INTWD: Watchdog timer	0024H	FFFF24H	
–	Maskable	Micro DMA (Note 2)	–	–	–
11		INT0: INT0 pin input	0028H	FFFF28H	0AH(Note 1)
12		INT1: INT1 pin input	002CH	FFFF2CH	0BH
13		INT2: INT2 pin input	0030H	FFFF30H	0CH
14		INT3: INT3 pin input	0034H	FFFF34H	0DH
15		INT4: INT4 pin input (TSI)	0038H	FFFF38H	0EH
16		INTALM: ALM (8192Hz, 512Hz, 64Hz, 2Hz, 1Hz)	003CH	FFFF3CH	0FH
17		INTTA4: 8-bit timer 4	0040H	FFFF40H	10H
18		INTTA5: 8-bit timer 5	0044H	FFFF44H	11H
19		INTTA6: 8-bit timer 6	0048H	FFFF48H	12H
20		INTTA7: 8-bit timer 7	004CH	FFFF4CH	13H
21		INTP0: Protect 0 (Write to SFR)	0050H	FFFF50H	14H
22		(Reserved)	0054H	FFFF54H	15H
23		INTTA0: 0	0058H	FFFF58H	16H
24		INTTA1: 8-bit timer 1	005CH	FFFF5CH	17H
25		INTTA2: 8-bit timer 2	0060H	FFFF60H	18H
26		INTTA3: 8-bit timer 3	0064H	FFFF64H	19H
27		INTTB0: 16-bit timer 0	0068H	FFFF68H	1AH
28		INTTB1: 16-bit timer 0	006CH	FFFF6CH	1BH
29		INTKEY: Key wakeup	0070H	FFFF70H	1CH
30		INTRTC: RTC (Alarm interrupt)	0074H	FFFF74H	1DH
31		(Reserved)	0078H	FFFF78H	1EH
32		INTLCD: LCDC	007CH	FFFF7CH	1FH
33		INTRX0: Serial receive end	0080H	FFFF80H	20H (Note 1)
34		INTTX0: Serial transmission end	0084H	FFFF84H	21H
35		INTTB10: 16-bit timer 1	0088H	FFFF88H	22H
36		INTTB11: 16-bit timer 1	008CH	FFFF8CH	23H
37		INT5: INT5 pin input	0090H	FFFF90H	24H
38		INT6: INT6 pin input	0094H	FFFF94H	25H
39		INT7: INT7 pin input	0098H	FFFF98H	26H
40		INTI2S0: I^2S (Channel 0)	009CH	FFFF9CH	27H
41		(Reserved)	00A0H	FFFFA0H	28H
42		INTADM: AD Monitor function	00A4H	FFFFA4H	29H
43		INTSBI: SBI	00A8H	FFFFA8H	2AH
44		INTSPIRX: SPIC receive	00ACH	FFFFACH	2BH
45		INTSPITX: SPIC transmission	00B0H	FFFFB0H	2CH
46		INTRSC: NAND Flash controller	00B4H	FFFFB4H	2DH
47		INTRDY: NAND Flash controller	00B8H	FFFFB8H	2EH
48		INTUSB: USB	00BCH	FFFFBCH	2FH
49		INTRX1: Serial receive end	00C0H	FFFFC0H	30H
50		INTTX1: Serial transmission end	00C4H	FFFFC4H	31H

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value	Address Refer to Vector	Micro DMA /HDMA Start Vector
51	Maskable	INTADHP: AD most priority conversion end	00C8H	FFFFC8H	32H
52		INTAD: AD conversion end	00CCH	FFFFCCH	33H
53		INTTC0/INTDMA0: Micro DMA0 /HDMA0 end	00D0H	FFFFD0H	34H
54		INTTC1/INTDMA1: Micro DMA1 /HDMA1 end	00D4H	FFFFD4H	35H
55		INTTC2/INTDMA2: Micro DMA2 /HDMA2 end	00D8H	FFFFD8H	36H
56		INTTC3/INTDMA3: Micro DMA3 /HDMA3 end	00DCH	FFFFDCH	37H
57		INTTC4/INTDMA4: Micro DMA4 /HDMA4 end	00E0H	FFFFE0H	38H
58		INTTC5/INTDMA5: Micro DMA5 /HDMA5 end	00E4H	FFFFE4H	39H
59		INTTC6 : Micro DMA6 end	00E8H	FFFFE8H	3AH
60		INTTC7 : Micro DMA7 end	00ECH	FFFFECH	3BH
– to –		(Reserved)	00F0H : 00FCH	FFFFF0H : FFFFFCH	– to –

Note 1: When initiating micro DMA/HDMA , set at edge detect mode.

Note 2 : Micro DMA default priority.
Micro DMA initiation takes priority over other maskable interrupt.

3.6.2 Micro DMA processing

In addition to general-purpose interrupt processing, the TMP92CF29A also includes a micro DMA function and HDMA function. This section explains about Micro DMA function. For the HDMA function, please refer 3.7 DMA controller.

Micro DMA processing for interrupt requests set by micro DMA is performed at the highest priority level for maskable interrupts (Level 6), regardless of the priority level of the interrupt source.

Because the micro DMA function is implemented through the CPU, when the CPU is placed in a stand-by state (IDLE2, IDLE1, STOP) by a HALT instruction, the requirement of the micro DMA will be ignored (Pending).

Micro DMA supports 8 channels and can be transferred continuously by specifying the micro DMA burst function as below.

Note: When using the micro DMA transfer end interrupt, always write “1” to bit 7 of SIMC register.

(1) Micro DMA operation

When an interrupt request is generated by an interrupt source that specified by the micro DMA /HDMA start vector register, and Micro DMA start is specified by DMA selection register, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request. When IFF = 7, Micro DMA request cannot be accepted.

The 8 micro DMA channels allow micro DMA processing to be set for up to 8 types of interrupt at once.

When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared. Data in one-byte, two-byte or four-byte blocks is automatically transferred at once from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decremented by “1”. If the value of the counter after it has been decremented is not “0”, DMA processing ends with no change in the value of the micro DMA start vector register. If the value of the decremented counter is “0”, a micro DMA transfer end interrupt (INTTC0 to INTTC7) is sent from the CPU to the interrupt controller.

In addition, the micro DMA /HDMA start vector register is cleared to “0”, the next micro DMA operation is disabled and micro DMA processing terminates.

If an interrupt request is triggered for the interrupt source in use during the interval between the time at which the micro DMA /HDMA start vector is cleared and the next setting, general-purpose interrupt processing is performed at the interrupt level set. Therefore, if the interrupt is only being used to initiate micro DMA /HDMA (and not as a general-purpose interrupt), the interrupt level should first be set to 0 (i.e, interrupt requests should be disabled).

If micro DMA and general-purpose interrupts are being used together as described above, the level of the interrupt which is being used to initiate micro DMA processing should first be set to a lower value than all the other interrupt levels. (Note1) In this case, edge-triggered interrupts are the only kinds of general interrupts which can be accepted.

Note1: If the priority level of micro DMA is set higher than that of other interrupts, CPU operates as follows.
In case INTxxx interrupt is generated first and then INTyyy interrupt is generated between checking “Interrupt specified by micro DMA start vector” (in the Figure 3.6.1) and reading interrupt vector with setting below. The vector shifts to that of INTyyy at the time.
This is because the priority level of INTyyy is higher than that of INTxxx.
In the interrupt routine, CPU reads the vector of INTyyy because cheking of micro DMA has finished. And INTyyy is generated regardless of transfer counter of micro DMA.
INTxxx: level 1 without micro DMA
INTyyy: level 6 with micro DMA

If micro DMA requests are set simultaneously for more than one channel, priority is not based on the interrupt priority level but on the channel number: The lower the channel number, the higher the priority (Channel 0 thus has the highest priority and channel 7 the lowest).

Note2: Don’t start any micro DMAs by one interrupt. If any micro DMA are set by it, micro DMA that channel number is biggest (priority is lowest) is not started.(Because interrupt flag is cleared by micro DMA that priority is highest)

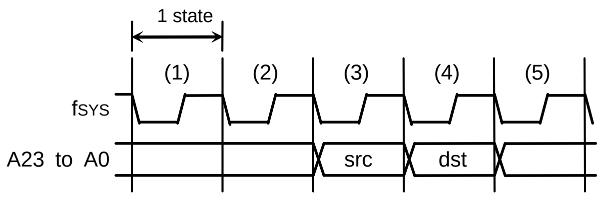
Although the control registers used for setting the transfer source and transfer destination addresses are 32 bits wide, this type of register can only output 24-bit addresses. Accordingly, micro DMA can only access 16 Mbytes (The upper 8 bits of a 32-bit address are not valid).

Three micro DMA transfer modes are supported: 1byte transfer, 2byte (One word) transfers and 4byte transfers. After a transfer in any mode, the transfer source and transfer destination addresses will either be incremented or decremented, or will remain unchanged. This simplifies the transfer of data from memory to memory, from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the various transfer modes, see section 3.6.2 (4) "Detailed description of the transfer mode register".

Since a transfer counter is a 16-bit counter, up to 65536 micro DMA processing operations can be performed per interrupt source (Provided that the transfer counter for the source is initially set to 0000H).

Micro DMA processing can be initiated by any one of 48 different interrupts – the 47 interrupts shown in the micro DMA start vectors in Table 3.6.1 and a micro DMA soft start.

Figure 3.6.2 shows a 2-byte transfer carried out using a micro DMA cycle in Transfer Destination Address INC Mode (micro DMA transfers are the same in every mode except Counter Mode). (The conditions for this cycle are as follows: both source and destination memory are internal-RAM and multiple of 4 numbered source and destination addresses).



Note: In fact, src and dst address are not outputted to A23-A0 pins because they are internal RAM address.

Figure 3.6.2 Timing for micro DMA cycle

States (1) and (2): Instruction fetch cycle (Prefetches the next instruction code)

State (3): Micro DMA read cycle.

State (4): Micro DMA writes cycle.

State (5): (The same as in state (1), (2).)

(2) Soft start function

The TMP92CF29A can initiate micro DMA/HDMA either with an interrupt or by using the micro DMA /HDMA soft start function, in which micro DMA or HDMA is initiated by a Write cycle which writes to the register DMAR.

Writing "1" to each bit of DMAR register causes micro DMA or HDMA to be performed once (If write "0" to each bit, micro DMA doesn't operate). On completion of the transfer, the bits of DMAR for the completed channel are automatically cleared to "0".

When writing again "1" to it, soft start can execute continuously until the DMA transfer counter (DMACn) or HDMA transfer counter B (HDMACBn) become "0".

When a burst is specified by the register DMAB, data is transferred continuously from the initiation of micro DMA until the value in the micro DMA transfer counter is "0". If execatee soft start during micro DMA transfer by interrupt source, micro DMA transfer counter doesn't change. Don't use Read-modify-write instruction to avoid writign to other bits by mistake.

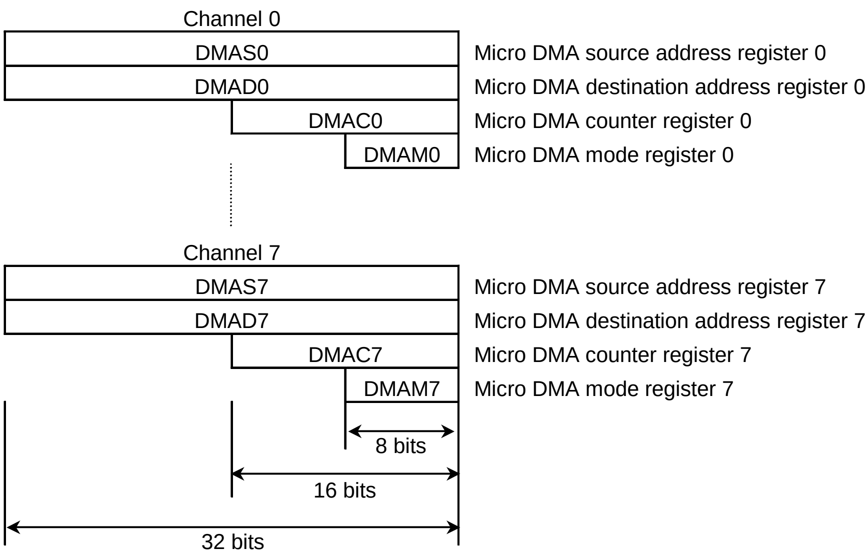
Note1: If it is started by software, don't set any channels to start in same time.

Note2: If be started sequentially, restart it after confirming micro DMA of all channels is completed (all micro DMA are set to "0").

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request	109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1	DREQ0
			R/W							
			0	0	0	0	0	0	0	0
			1: Start DMA							

(3) Transfer control registers

The transfer source address and the transfer destination address are set in the following registers. An instruction of the form LDC cr,r can be used to set these registers.



(4) Detailed description of the transfer mode register

0	0	0	Mode				

DMAM0 to DMAM7

DMAMn[4:0]	Mode Description	Execution Time
0 0 0 z z	Destination INC mode (DMADn +) ← (DMASn) DMACn ← DMACn - 1 If DMACn = 0 then INTTCn	5 states
0 0 1 z z	Destination DEC mode (DMADn -) ← (DMASn) DMACn ← DMACn - 1 If DMACn = 0 then INTTCn	5 states
0 1 0 z z	Source INC mode (DMADn) ← (DMASn +) DMACn ← DMACn - 1 If DMACn = 0 then INTTCn	5 states
0 1 1 z z	Source DEC mode (DMADn) ← (DMASn -) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5 states
1 0 0 z z	Source and destination INC mode (DMADn +) ← (DMASn +) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6 states
1 0 1 z z	Source and destination DEC mode (DMADn -) ← (DMASn -) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	6 states
1 1 0 z z	Destination and fixed mode (DMADn) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5 states
1 1 1 00	Counter mode DMASn ← DMASn + 1 DMACn ← DMACn – 1 If DMACn = 0 then INTTCn	5 states

ZZ: 00 = 1-byte transfer
01 = 2-byte transfer
10 = 4-byte transfer
11 = Reserved

Note 1: n stands for the micro DMA channel number (0 to 7).
DMADn+/DMASn+: Post increment (Register value is incremented after transfer).
DMADn–/DMASn–: Post decrement (Register value is decremented after transfer).
"I/O" signifies fixed memory addresses; "memory" signifies incremented or decremented memory addresses.

Note 2: The transfer mode register should not be set to any value other than those listed above.

Note 3: The execution state number shows number of best case (1-state memory access).

3.6.3 Interrupt Controller Operation

The block diagram in Figure 3.6.3 shows the interrupt circuits. The left-hand side of the diagram shows the interrupt controller circuit. The right-hand side shows the CPU interrupt request signal circuit and the halt release circuit.

For each of the 59 interrupts channels there is an interrupt request flag (consisting of a flip-flop), an interrupt priority setting register and a micro DMA /HDMA start vector register. The interrupt request flag latches interrupt requests from the peripherals.

The flag is cleared to "0" in the following cases: when a reset occurs, when the CPU reads the channel vector of an interrupt it has received, when the CPU receives a micro DMA request (when micro DMA is set), when the CPU receives a HDMA request (when HDMA is set), when a micro DMA burst transfer is terminated, and when an instruction that clears the interrupt for that channel is executed (by writing a micro DMA start vector to the INTCLR register).

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0 or INTE12). Six interrupt priorities levels (1 to 6) are provided. Setting an interrupt source's priority level to 0 (or 7) disables interrupt requests from that source.

If more than one interrupt request with a given priority level are generated simultaneously, the default priority (The interrupt with the lowest priority or, in other words, the interrupt with the lowest vector value) is used to determine which interrupt request is accepted first. The 3rd and 7th bits of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

If several interrupts are generated simultaneously, the interrupt controller sends the interrupt request for the interrupt with the highest priority and the interrupt's vector address to the CPU. The CPU compares the mask value set in <IFF2:0> of the status register (SR) with the priority level of the requested interrupt; if the latter is higher, the interrupt is accepted. Then the CPU sets SR<IFF2:0> to the priority level of the accepted interrupt + 1. Hence, during processing of the accepted interrupt, new interrupt requests with a priority value equal to or higher than the value set in SR<IFF2:0> (e.g., interrupts with a priority higher than the interrupt being processed) will be accepted.
When interrupt processing has been completed (e.g., after execution of a RETI instruction), the CPU restores to SR<IFF2:0> the priority value which was saved on the stack before the interrupt was generated.

The interrupt controller also includes eight registers which are used to store the micro DMA /HDMA start vector. Writing the start vector of the interrupt source for the micro DMA or /HDMA processing (See Table 3.6.1), enables the corresponding interrupt to be processed by micro DMA or HDMA processing. The values must be set in the micro DMA parameter registers (e.g., DMAS and DMAD) or HDMA parameter registers (e.g., HDMAS, and HDMAD) prior to micro DMA or HDMA processing.

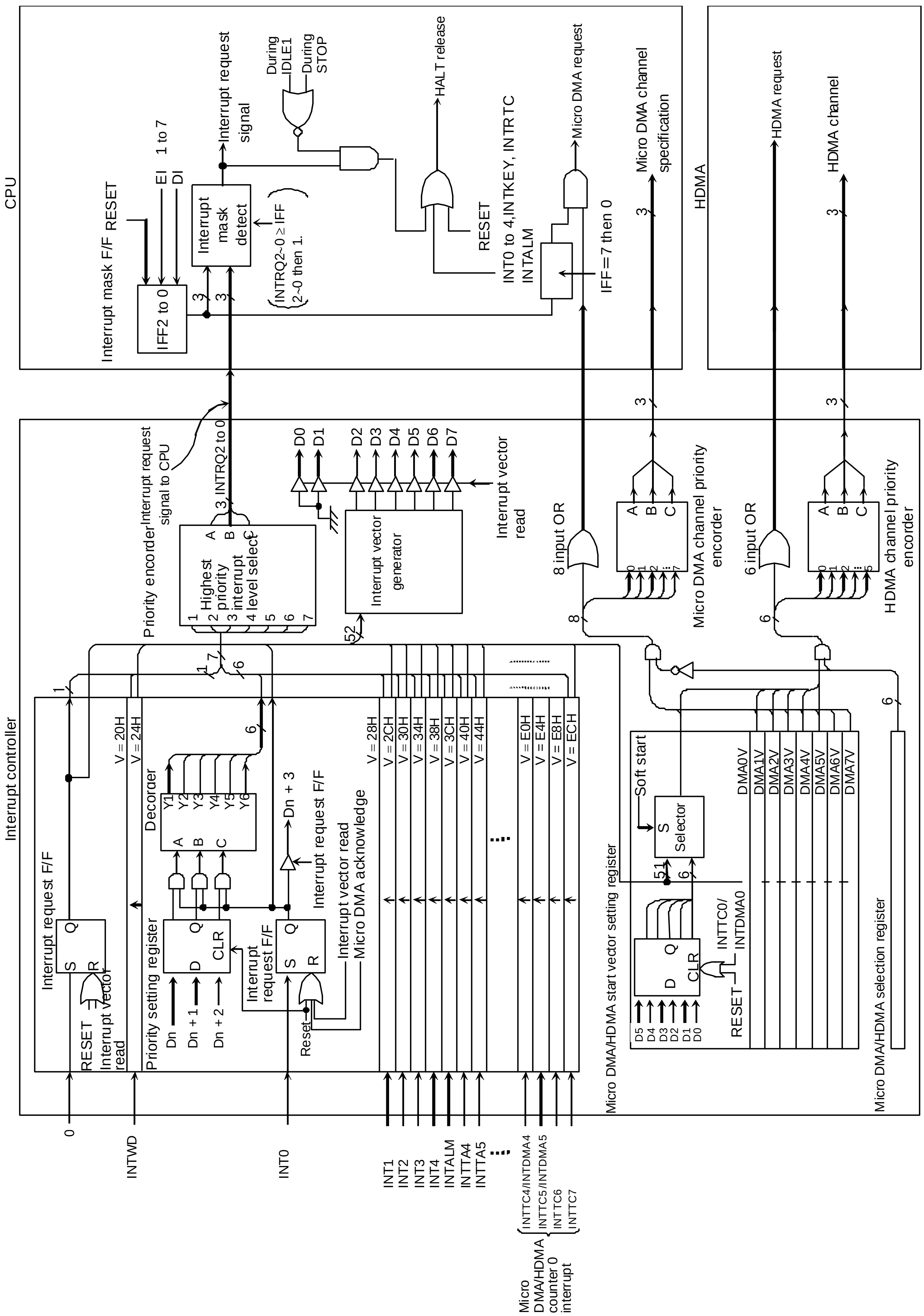


Figure 3.6.3 Block Diagram of Interrupt Controller

(1) Interrupt priority setting registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0	INT0 enable	F0H	–				INT0			
			–	–	–	–	I0C	I0M2	I0M1	I0M0
			–	–			R	R/W		
			Always write "0".				0	0	0	0
INTE12	INT1 & INT2 enable	D0H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	D1H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	D2H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE7	INT7 enable	D3H	–				INT7			
			–	–	–	–	I7C	I7M2	I7M1	I7M0
			–	–			R	R/W		
			Always write "0".				0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	D4H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	D5H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA45	INTTA4 & INTTA5 enable	D6H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA67	INTTA6 & INTTA7 enable	D7H	INTTA7 (TMRA7)				INTTA6 (TMRA6)			
			ITA7C	ITA7M2	ITA7M1	ITA7M0	ITA6C	ITA6M2	ITA6M1	ITA6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB0	INTTB00 & INTTB01 enable	D8H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	INTTB10 & INTTB11 enable	D9H	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES0	INTRX0 & INTTX0 enable	DBH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	INTRX1 & INTTX1 enable	DCH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESBIADM	INTSBI & INTADM enable	E0H	INTADM				INTSBI			
			IADM0C	IADMM2	IADMM1	IADMM0	ISBI0C	ISBIM2	ISBIM1	ISBIM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESPI	INTSPI enable	E1H	INTSPITX				INTSPIRX			
			ISPITC	ISPITM2	ISPITM1	ISPITM0	ISPIRC	ISPIRM2	ISPIRM1	ISPIRM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEUSB	INTUSB enable	E3H	–				INTUSB			
			–	–	–	–	IUSBC	IUSBM2	IUSBM1	IUSBM0
			–	–			R	R/W		
			Always write "0".				0	0	0	0
INTEALM	INTALM enable	E5H	–				INTALM			
			–	–	–	–	IALMC	IALMM2	IALMM1	IALMM0
			–	–			R	R/W		
			Always write "0".				0	0	0	0
INTERTC	INTRTC enable	E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
			–	–			R	R/W		
			Always write "0".				0	0	0	0

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEKEY	INTKEY enable	E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
			–	–			R	R/W		
			Always write “0”.				0	0	0	0
INTELCD	INTLCD enable	EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			–	–			R	R/W		
			Always write “0”.				0	0	0	0
INTEI2S0	INTI2S0 enable	EBH	–				INTI2S0			
			–	–	–	–	I I2S0C	II2S0M2	II2S0M1	II2S0M0
			–	–			R/W	R/W		
			Always write “0”.				0	0	0	0
INTENDFC	INTRSC & INTRDY enable	ECH	INTRSC				INTRDY			
			IRSCC	IRSCM2	IRSCM1	IRSCM0	IRDYC	IRDYM2	IRDYM1	IRDYM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
			–	–			R	R/W		
			Always write “0”.				0	0	0	
0INTEAD	INTAD & INTADHP enable	EFH	INTADHP				INTAD			
			IADHPC	IADHPM2	IADHPM1	IADHPM0	IADC	IADM2	IADM1	IADM0
			R	R/W			R/W	R/W		
			0	0	0	0	0	0	0	0

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETC01 /INTEDMA01	INTTC0/INTDMA0 & INTTC1/INTDMA1 enable	F1H	INTTC1/INTDMA1				INTTC0/INTDMA0			
			ITC1C /IDMA1C	ITC1M2 /IDMA1M2	ITC1M1 /IDMA1M1	ITC1M0 /IDMA1M0	ITC0C /IDMA0C	ITC0M2 /IDMA0M2	ITC0M1 /IDMA0M1	ITC0M0 /IDMA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23 /INTEDMA23	INTTC2/INTDMA2 & INTTC3/INTDMA3 enable	F2H	INTTC3/INTDMA3				INTTC2/INTDMA2			
			ITC3C /IDMA3C	ITC3M2 /IDMA3M2	ITC3M1 /IDMA3M1	ITC3M0 /IDMA3M0	ITC2C /IDMA2C	ITC2M2 /IDMA2M2	ITC2M1 /IDMA2M1	ITC2M0 /IDMA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC45 /INTEDMA45	INTTC4/INTDMA4 & INTTC5/INTDMA5 enable	F3H	INTTC5/INTDMA5				INTTC4/INTDMA4			
			ITC5C /IDMA5C	ITC5M2 /IDMA5M2	ITC5M1 /IDMA5M1	ITC5M0 /IDMA5M0	ITC4C /IDMA4C	ITC4M2 /IDMA4M2	ITC4M1 /IDMA4M1	ITC4M0 /IDMA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC67	INTTC6 & INTTC7 enable	F4H	INTTC7 (DMA7)				INTTC6 (DMA6)			
			ITC7C	ITC7M2	ITC7M1	ITC7M0	ITC6C	ITC6M2	ITC6M1	ITC6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTWDT	INTWD enable	F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
			–	–			R			
			Always write “0”.				0	–	–	–

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

(2) External interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC0	Interrupt input mode control 0	F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W						R/W	R/W
			0	0	0	0	0	0	0	0
			INT5EDGE 0: Rising 1: Falling	INT4EDGE 0: Rising 1: Falling	INT3EDGE 0: Rising 1: Falling	INT2EDGE 0: Rising 1: Falling	INT1EDGE 0: Rising 1: Falling	INT0EDGE 0: Rising 1: Falling	INT0 0:Edge mode 1: Level mode	Always write “0”
IIMC1	Interrupt input mode control 1	FAH (Prohibit RMW)							I7EDGE	I6EDGE
									W	
									0	0
									INT7EDGE 0: Rising 1: Falling	INT6EDGE 0: Rising 1: Falling

Note 1: Disable INT0 request before changing INT0 pin mode from level sense to edge sense. (change <I0LE>from “1” to “0”)

```
DI
LD      (IIMC0), XXXXXX0-B   ; Switches from level to edge.
LD      (INTCLR), 0AH        ; Clears interrupt request flag.

NOP                          ; Wait EI execution
NOP
NOP
EI
```

X: Don’t care, –: No change

Note 2: See electrical characteristics in section 4 for external interrupt input pulse width.

Note 3: In port setting, if 16 bit timer input is selected and capture control is executed, INT6 and INT7 don’t depend on IIMC1 register setting. INT6 and INT7 operate by setting TBnMOD<TBnCPM1:0>.

Settings of External Interrupt Pin Function

Interrupt	Pin Name	Mode	Setting Method
INT0	PC0	Rising edge	<I0LE> = 0,<I0EDGE> = 0
		Falling edge	<I0LE> = 0, <I0EDGE> = 1
		High level	<I0LE> = 1
INT1	PC1	Rising edge	<I1EDGE> = 0
		Falling edge	<I1EDGE> = 0
INT2	PC2	Rising edge	<I2EDGE> = 0
		Falling edge	<I2EDGE> = 1
INT3	PC3	Rising edge	<I3EDGE> = 0
		Falling edge	<I3EDGE> = 1
INT4	P96	Rising edge	<I4EDGE> = 0
		Falling edge	<I4EDGE> = 1
INT5	PP3	Rising edge	<I5EDGE> = 0
		Falling edge	<I5EDGE> = 1
INT6	PP4	Rising edge	<I6EDGE> = 0
		Falling edge	<I6EDGE> = 1
INT7	PP5	Rising edge	<I7EDGE> = 0
		Falling edge	<I7EDGE> = 1

(3) SIO receive interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
SIMC	SIO interrupt mode control	F5H (Prohibit RMW)	–	–					IR1LE	IR0LE
			W						W	
			0	0					1	1
			Always write "0" (Note)	Always write "0"					0:INTRX1 edge mode 1:INTRX1 level mode	0:INTRX0 edge mode 1:INTRX0 level mode

Note: When using the micro DMA transfer end interrupt, always write "1".

INTRX edge enable

0	Edge detect INTRX
1	"H" level INTRX

(4) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA /HDMA start vector, as given in Table 3.6.1 to the register INTCLR.

For example, to clear the interrupt flag INT0, perform the following register operation after execution of the DI instruction.

INTCLR ← 0AH ; Clears interrupt request flag INT0.

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTCLR	Interrupt clear control	F8H (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							

(5) Micro DMA start vector registers

These registers assign micro DMA/HDMA processing to sets which source corresponds to DMA. The interrupt source whose micro DMA /HDMA start vector value matches the vector set in one of these registers is designated as the micro DMA /HDMA start source.

When the micro DMA transfer counter (DMACn) or HDMA transfer counter B (HDMACBn) value reaches "0", the micro DMA /HDMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA /HDMA start vector register is cleared, and the micro DMA /HDMA start source for the channel is cleared. Therefore, in order for micro DMA /HDMA processing to continue, the micro DMA /HDMA start vector register must be set again during processing of the micro DMA /HDMA transfer end interrupt.

If the same vector is set in the micro DMA /HDMA start vector registers of more than one channel, the lowest numbered channel takes priority.

Accordingly, if the same vector is set in the micro DMA /HDMA start vector registers for two different channels, the interrupt generated on the lower-numbered channel is executed until micro DMA /HDMA transfer is complete. If the micro DMA /HDMA start vector for this channel has not been set in the channel's micro DMA /HDMA start vector register again, micro DMA /HDMA transfer for the higher-numbered channel will be commenced. (This process is known as micro DMA /HDMA chaining.)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	100H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	101H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	102H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	103H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
DMA4V	DMA4 start vector	104H			DMA4V5	DMA4V4	DMA4V3	DMA4V2	DMA4V1	DMA4V0
					R/W					
					0	0	0	0	0	0
					DMA4 start vector					
DMA5V	DMA5 start vector	105H			DMA5V5	DMA5V4	DMA5V3	DMA5V2	DMA5V1	DMA5V0
					R/W					
					0	0	0	0	0	0
					DMA5 start vector					
DMA6V	DMA6 start vector	106H			DMA6V5	DMA6V4	DMA6V3	DMA6V2	DMA6V1	DMA6V0
					R/W					
					0	0	0	0	0	0
					DMA6 start vector					
DMA7V	DMA7 start vector	107H			DMA7V5	DMA7V4	DMA7V3	DMA7V2	DMA7V1	DMA7V0
					R/W					
					0	0	0	0	0	0
					DMA7 start vector					

(6) Micro DMA/HDMA select register

This register selectable that is started either Micro DMA or HDMA processing.

Micro DMA /HDMA start vector register (DMAnV) shared with both functions. When interrupt which match with vector value that is set to DMA/HDMA start vector register generated, use this register.

Symbol	NAME	Address	7	6	5	4	3	2	1	0
DMASEL	Micro DMA/ HDMA select	10AH			DMASEL5	DMASEL4	DMASEL3	DMASEL2	DMASEL1	DMASEL0
					R/W					
					0	0	0	0	0	0
					0:Micro DMA5 1:HDMA5	0:Micro DMA4 1:HDMA4	0:Micro DMA3 1:HDMA3	0:Micro DMA2 1:HDMA2	0:Micro DMA1 1:HDMA1	0:Micro DMA0 1:HDMA0

(7) Specification of a micro DMA burst

Specifying the micro DMA burst function causes micro DMA transfer, once started, to continue until the value in the transfer counter register reaches “0”. Setting any of the bits in the register DMAB which correspond to a micro DMA channel (as shown below) to “1” specifies that any micro DMA transfer on that channel will be a burst transfer.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAB	DMA burst	108H	DBST7	DBST6	DBST5	DBST4	DBST3	DBST2	DBST1	DBST0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request on Burst mode							

(8) Notes

The instruction execution unit and the bus interface unit in this CPU operate independently. Therefore, if immediately before an interrupt is generated, the CPU fetches an instruction which clears the corresponding interrupt request flag, the CPU may execute this instruction in between accepting the interrupt and reading the interrupt vector. In this case, the CPU will read the default vector 0004H and jump to interrupt vector address FFFF04H.

To avoid this, an instruction which clears an interrupt request flag should always be preceded by a DI instruction. And in the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing and more than 3-instructions (e.g., “NOP” × 3 times). If placed EI instruction without waiting NOP instruction after execution of clearing instruction, interrupt will be enable before request flag is cleared.

In the case of changing the value of the interrupt mask register <IFF2:0> by execution of POP SR instruction, disable an interrupt by DI instruction before execution of POP SR instruction.

In addition, please note that the following two circuits are exceptional and demand special attention.

<table>
<tr><td>INT0 level mode</td><td>In level mode INT0 is not an edge-triggered interrupt. Hence, in level mode the interrupt request flip-flop for INT0 does not function. The peripheral interrupt request passes through the S input of the flip-flop and becomes the Q output. If the interrupt input mode is changed from edge mode to level mode, the interrupt request flag is cleared automatically.
If the CPU enters the interrupt response sequence as a result of INT0 going from 0 to 1, INT0 must then be held at 1 until the interrupt response sequence has been completed. If INT0 is set to level mode so as to release a halt state, INT0 must be held at 1 from the time INT0 changes from 0 to 1 until the halt state is released. (Hence, it is necessary to ensure that input noise is not interpreted as a 0, causing INT0 to revert to 0 before the halt state has been released.)
When the mode changes from level mode to edge mode, interrupt request flags which were set in level mode will not be cleared. Interrupt request flags must be cleared using the following sequence.

DI
LD (IIMC0), 00H ; Switches from level to edge.
LD (INTCLR), 0AH ; Clears interrupt request flag.
NOP ; Wait EI execution
NOP
NOP
EI</td></tr>
<tr><td>INTRX</td><td>In level mode (The register SIMC<IRxLE> set to “1”), the interrupt request flip-flop can only be cleared by a reset or by reading the serial channel receive buffer. It cannot be cleared by writing INTCLR register.</td></tr>
</table>

Note: The following instructions or pin input state changes are equivalent to instructions which clear the interrupt request flag.

INT0: Instructions which switch to level mode after an interrupt request has been generated in edge mode.

The pin input changes from high to low after an interrupt request has been generated in level mode. (“H” → “L”)

INTRX: Instructions which read the receive buffer.

3.7 DMAC (DMA Controller)

The TMP92CF29A incorporates a DMA controller (DMAC) having six channels. This DMAC can realize data transfer faster than the micro DMA function by the 900/H1 CPU.

The DMAC has the following features:

1) Six independent channels of DMA

2) Two types of transfer start requests

   Hardware request (using an interrupt source connected with the INTC) or software request can be selected for each channel.

3) Various source/destination combinations

   The combination of transfer source and destination can be selected for each channel from the following four types: memory to memory, memory to I/O, I/O to memory, I/O to I/O.

4) Transfer address mode

   Only the dual address mode is supported.

5) Dual-count mechanism and DMA end interrupt

   Two count registers are provided to execute multiple DMA transfers by one DMA request and to generate multiple DMA requests at a time. The DMA end interrupt (INTDMA0 to INTDMA5) is also provided so that a general-purpose interrupt routine can be used to prepare for the next processing.

6) Priorities among DMA channels (the same as the micro DMA acceptance specifications of the INTC)

   DMA requests are basically accepted in the order in which they are asserted. If more than one request is asserted simultaneously or it looks as if two requests were asserted simultaneously because one of the requests has been put on hold while other processing was being performed, the smaller-numbered channel is given a higher priority.

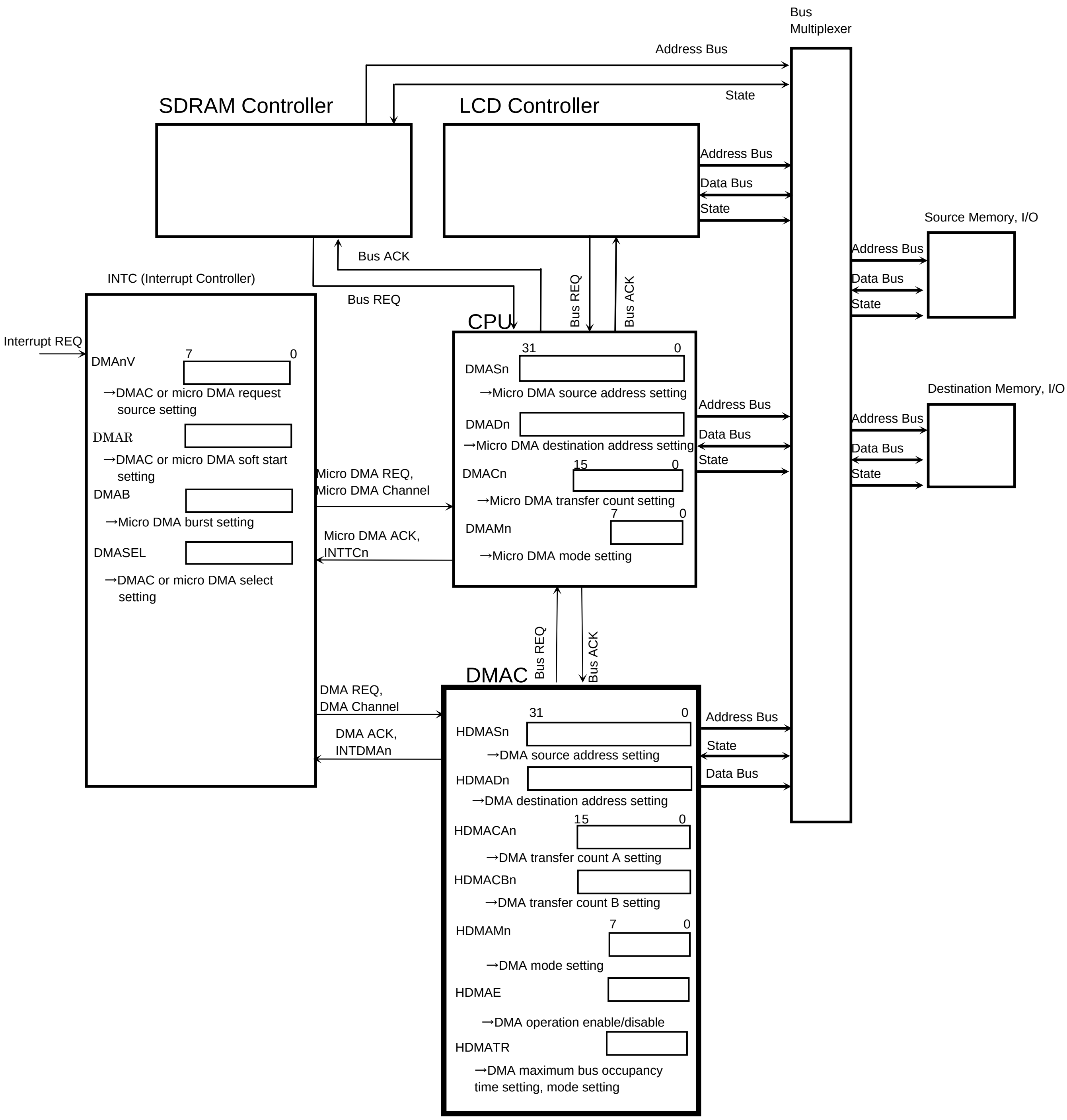
7) DMAC bus occupancy limiting function

   The DMAC incorporates a special timer for limiting its bus occupancy time to avoid excessive interference with the CPU or LCDC operation.

8) The DMAC can be used in HALT (IDLE2) mode.

3.7.1 Block Diagram

Figure 3.7.1 shows an overall block diagram for the DMAC.



Note: "n" denotes a channel number. Micro DMA has eight channels (0 to 7) and DMA has six channels (0 to 5).

Figure 3.7.1 Overall Block Diagram

3.7.2 SFRs

The DMAC has the following SFRs. These registers are connected to the CPU via a 16-bit data bus.

(1) HDMASn (DMA Transfer Source Address Setting Register)

The HDMASn register is used to set the DMA transfer source address. When the source address is updated by DMA execution, HDMASn is also updated.

HDMAS0 to HDMAS5 have the same configuration.

Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

HDMASn Register

HDMASn		7	6	5	4	3	2	1	0
	bit Symbol	DnSA7	DnSA6	DnSA5	DnSA4	DnSA3	DnSA2	DnSA1	DnSA0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Source address [7:0] for DMAn							
		15	14	13	12	11	10	9	8
	bit Symbol	DnSA15	DnSA14	DnSA13	DnSA12	DnSA11	DnSA10	DnSA9	DnSA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Source address [15:8] for DMAn							
		23	22	21	20	19	18	17	16
	bit Symbol	DnSA23	DnSA22	DnSA21	DnSA20	DnSA19	DnSA18	DnSA17	DnSA16
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Source address [23:16] for DMAn							

	Source address [23:16]	Source address [15:8]	Source address [7:0]
Channel 0	(0902H)	(0901H)	HDMAS0 (0900H)
Channel 1	(0912H)	(0911H)	HDMAS1 (0910H)
Channel 2	(0922H)	(0921H)	HDMAS2 (0920H)
Channel 3	(0932H)	(0931H)	HDMAS3 (0930H)
Channel 4	(0942H)	(0941H)	HDMAS4 (0940H)
Channel 5	(0952H)	(0951H)	HDMAS5 (0950H)

Note: Read-modify-write instructions can be used on all these registers.

Figure3.7.2 HDMASn Register

(2) HDMADn (DMA Transfer Destination Address Setting Register)

The HDMADn register is used to set the DMA transfer destination address. When the destination address is updated by DMA execution, HDMADn is also updated.

HDMAD0 to HDMAD5 have the same configuration.

Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

HDMADn Register

		7	6	5	4	3	2	1	0
HDMADn	bit Symbol	DnDA7	DnDA6	DnDA5	DnDA4	DnDA3	DnDA2	DnDA1	DnDA0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Destination address [7:0] for DMAn							
		15	14	13	12	11	10	9	8
	bit Symbol	DnDA15	DnDA14	DnDA13	DnDA12	DnDA11	DnDA10	DnDA9	DnDA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Destination address [15:8] for DMAn							
		23	22	21	20	19	18	17	16
	bit Symbol	DnDA23	DnDA22	DnDA21	DnDA20	DnDA19	DnDA18	DnDA17	DnDA16
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Destination address [23:16] for DMAn							

	Destination address [23:16]	Destination address [15:8]	Destination address [7:0]
Channel 0	(0906H)	(0905H)	HDMAD0 (0904H)
Channel 1	(0916H)	(0915H)	HDMAD1 (0914H)
Channel 2	(0926H)	(0925H)	HDMAD2 (0924H)
Channel 3	(0936H)	(0935H)	HDMAD3 (0934H)
Channel 4	(0946H)	(0945H)	HDMAD4 (0944H)
Channel 5	(0956H)	(0955H)	HDMAD5 (0954H)

Note: Read-modify-write instructions can be used on all these registers.

Figure3.7.3 HDMADn Register

(3) HDMACAn (DMA Transfer Count A Setting Register)

The HDMACAn register is used to set the number of times a DMA transfer is to be performed by one DMA request. HDMACAn contains 16 bits and can specify up to 65536 transfers (0001H = one transfer, FFFFH = 65535 transfers, 0000H = 65536 transfers). Even when the transfer count A is updated by DMA execution, HDMACAn is not updated.

HDMACA0 to HDMACA5 have the same configuration.

HDMACAn Register

		7	6	5	4	3	2	1	0
HDMACAn	bit Symbol	DnCA7	DnCA6	DnCA5	DnCA4	DnCA3	DnCA2	DnCA1	DnCA0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer count A [7:0] for DMAn							
		15	14	13	12	11	10	9	8
	bit Symbol	DnCA15	DnCA14	DnCA13	DnCA12	DnCA11	DnCA10	DnCA9	DnCA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer count A [15:8] for DMAn							

	Transfer count A [15:8]	Transfer count A [7:0]
Channel 0	(0909H)	HDMACA0 (0908H)
Channel 1	(0919H)	HDMACA1 (0918H)
Channel 2	(0929H)	HDMACA2 (0928H)
Channel 3	(0939H)	HDMACA3 (0938H)
Channel 4	(0949H)	HDMACA4 (0948H)
Channel 5	(0959H)	HDMACA5 (0958H)

Note: Read-modify-write instructions can be used on all these registers.

Figure3.7.4 HDMACAn Register

(4) HDMACBn (DMA Transfer Count B Setting Register)

The HDMACBn register is used to set the number of times a DMA request is to be made. HDMACBn contains 16 bits and can specify up to 65536 requests (0001H = one request, FFFFH = 65535 requests, 0000H = 65536 requests). When the transfer count B is updated by DMA execution, HDMACBn is also updated.

HDMACB0 to HDMACB5 have the same configuration.

HDMACBn Register

HDMACBn		7	6	5	4	3	2	1	0
	bit Symbol	DnCB7	DnCB6	DnCB5	DnCB4	DnCB3	DnCB2	DnCB1	DnCB0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer count B [7:0] for DMAn							
		15	14	13	12	11	10	9	8
	bit Symbol	DnCB15	DnCB14	DnCB13	DnCB12	DnCB11	DnCB10	DnCB9	DnCB8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer count B [15:8] for DMAn							

	Transfer count B [15:8]	Transfer count B [7:0]
Channel 0	(090BH)	HDMACB0 (090AH)
Channel 1	(091BH)	HDMACB1 (091AH)
Channel 2	(092BH)	HDMACB2 (092AH)
Channel 3	(093BH)	HDMACB3 (093AH)
Channel 4	(094BH)	HDMACB4 (094AH)
Channel 5	(095BH)	HDMACB5 (095AH)

Note: Read-modify-write instructions can be used on all these registers.

Figure3.7.5 HDMACBn Register

(5) HDMAMn (DMA Transfer Mode Setting Register)

The HDMAMn register is used to set the DMA transfer mode.

HDMAM0 to HDMAM5 have the same configuration.

HDMAMn Register

HDMAMn		7	6	5	4	3	2	1	0
	bit Symbol				DnM4	DnM3	DnM2	DnM1	DnM0
	Read/Write				R/W				
	Reset State				0	0	0	0	0
	Function				DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved (Note 2)			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

	Transfer mode [7:0]
Channel 0	HDMAM0 (090CH)
Channel 1	HDMAM1 (091CH)
Channel 2	HDMAM2 (092CH)
Channel 3	HDMAM3 (093CH)
Channel 4	HDMAM4 (094CH)
Channel 5	HDMAM5 (095CH)

Note 1: Read-modify-write instructions can be used on all these registers.

Note 2: INC: Post-increment

Dec: Post-decrement

I/O: Fixed memory address

MEM: Memory address to be incremented or decremented

Figure3.7.6 HDMAMn Register

(6) HDMAE (DMA Operation Enable Register)

The HDMAE register is used to enable or disable the DMAC operation.

Bits 0 to 5 correspond to channels 0 to 5. Unused channels should be set to “0”.

HDMAE Register

HDMAE (097EH)		7	6	5	4	3	2	1	0
	bit Symbol			DMAE5	DMAE4	DMAE3	DMAE2	DMAE1	DMAE0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			DMA channel operation 0: Disable 1: Enable					

Note: Read-modify-write instructions can be used on this register.

Figure3.7.7 HDMAE Register

(7) HDMATR (DMA Maximum Bus Occupancy Time Setting Register)

The HDMATR register is used to set the maximum duration of time the DMAC can occupy the bus. The TMP92CF29A does not have priority levels for bus arbitration. Therefore, once the DMAC owns the bus, other masters (such as the LCDC) must wait until the DMAC completes its transfer operation and releases the bus. This could lead to problems in the system. For example, if the LCDC cannot own the bus as required, the LCD display function may not work properly. To avoid such a situation, the DMAC limits the duration of its bus occupancy by using this timer register. When the DMAC occupies the bus for the duration of time set in this register, it releases the bus even if the specified DMA operation has not been completed yet. After waiting for 16 states, the DMAC asserts a bus request again to execute the rest of the DMA operation.

The DMAC counts the bus occupancy time regardless of which channel is occupying the bus. To set the maximum bus occupancy time, ensure that the HDMAE register is set to “00H” and set HDMATR<DMATE> to “1” and <DMATR6:0> to the desired value.

Note: In case of using S/W start with HDMA, transmission start is to set to “1” DMAR register. However DMAR register can't be used to confirm flag of transmission end. DMAR register reset to “0” when HDMA release bus occupation once with HDMATR function.

HDMATR Register

HDMATR (097FH)		7	6	5	4	3	2	1	0
	bit Symbol	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by “maximum bus occupancy time / (256/f_{SYS})”. “00H” cannot be set.						

Note: Read-modify-write instructions can be used on this register.

Figure3.7.8 HDMATR Register

3.7.3 DMAC Operation Description

(1) Overall flowchart

Figure 3.7.9 shows a flowchart for DMAC operation when an interrupt (DMA) is requested.

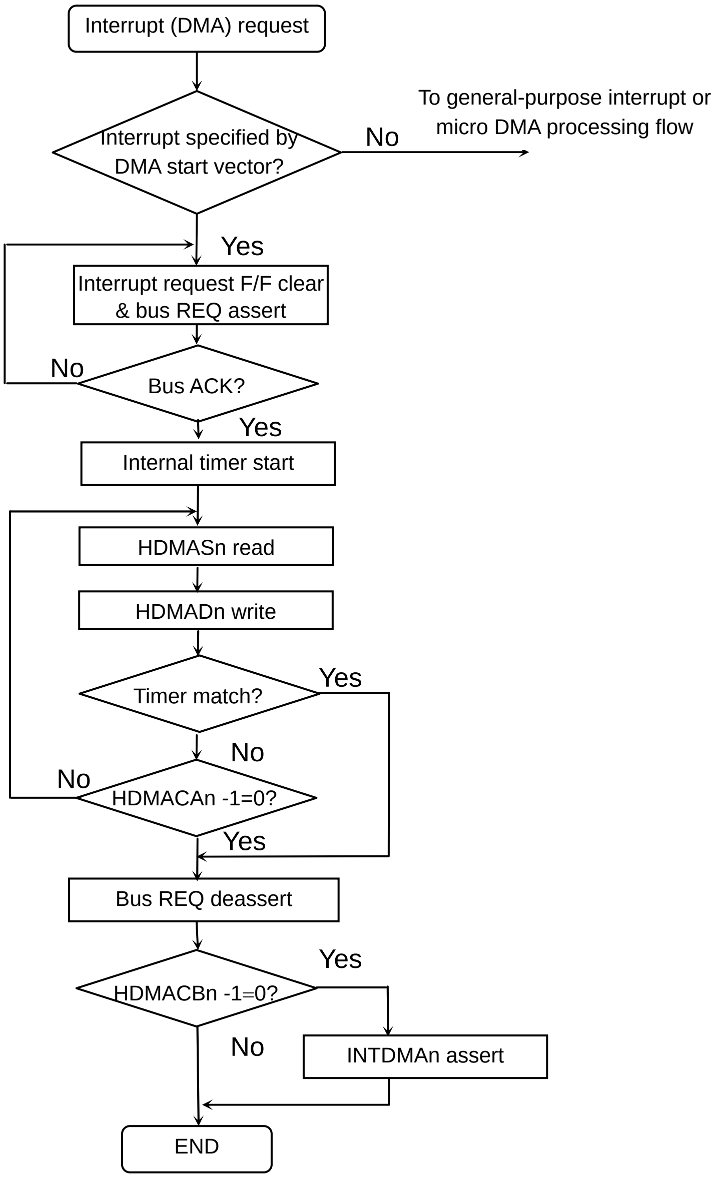


Figure 3.7.9 Overall Flowchart

(2) Bus arbitration

The TMP92CF29A includes three controllers (DMA controller, LCD controller, SDRAM controller) that function as bus masters apart from the CPU. These controllers operate independently and assert a bus request as required. The controller that receives a bus acknowledgement acts as the bus master. No priorities are assigned to these three controllers, and bus requests are processed in the order in which they are asserted. Once one of the controllers owns the bus, bus requests from other controllers are put on hold until the bus is released again. While one of the controllers is occupying the bus, CPU processing including non-maskable interrupt requests is also put on hold.

(3) Transfer source and destination memory setting

Either internal or external memory can be set as the source and destination memory or I/O to be accessed by the DMAC. Even when the MMU is used in external memory, the addresses to be accessed by the DMAC should be specified using logical addresses. The DMAC accesses the specified source and destination addresses according to the bus width and number of waits set in the memory controller and the bank settings made in the MMU.

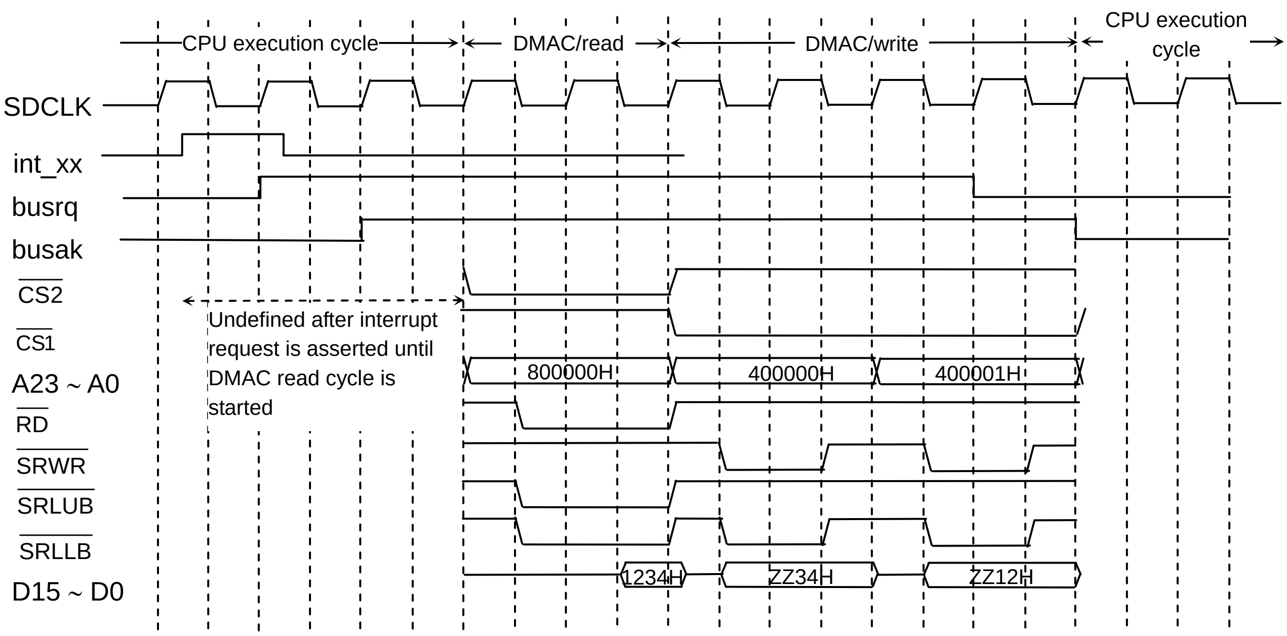
Although the bus sizing function is supported, the address alignment function is not supported. Therefore, specify an even-numbered address for transferring 2 bytes and an address that is an integral multiple of 4 for transferring 4 bytes.

Table 3.7.1 Difference point of address setting between HDMA and micro DMA

	Data Length	HDMA	Micro DMA
Source address	1byte	No restriction	No restriction
	2byte	Even address	
	4byte	Address in multiples of 4	
Destination address	1byte	No restriction	
	2byte	Even address	
	4byte	Address in multiples of 4	

(4) Operation timing

The following diagram shows an example of operation timing for transferring 2 bytes from 16-bit memory connected with the $\overline{CS2}$ area to 8-bit memory connected with the $\overline{CS1}$ area.



3.7.4 Setting Example

This section explains how to set the DMAC using an example.

(1) Transferring music data from internal RAM to I²S by DMA transfer

The 32 Kbytes of data stored in the internal RAM at addresses 2000H to 9FFFH shall be transferred to FIFO-RAM via I²S. Each time an INTI2S request is asserted, 64 bytes (4 bytes x 16 times) shall be transferred to FIFO-RAM using DMAC channel 0. Since INTI2S is an FIFO empty interrupt, the first data must be set in advance. Therefore, only the first 64 bytes shall be transferred by DMA soft start. After 32 Kbytes have been transferred, the INTDMA0 interrupt routine shall be activated to prepare for the next processing.

(a) Main routine

No	Instruction	Comments
1	ldl (hdmas0),2000H	; Source address = 2000H
2	ldl (hdmad0),i2sbuf	; Destination address = i2sbuf
3	ldw (hdmaca0),16	; Counter A = 16
4	ldw (hdmacb0),512	; Counter B = 512 (32768/64)
5	ldb (hdmam0),0AH	; Transfer mode = source INC, 4 bytes
6	set 0,(hdmae)	; Enable DMA channel 0.
7	ld (dmar),01H	; Transfer the first 64 bytes by DMA soft start.
8	nop	;
9	ld (dma0v),i2s_vector	; INTI2S = DMA0
10	ld (intedma01),xxH	; INTDMA level = x
11	ldw (i2sctl0),xxxxH	; Set operation mode for I²S.
12	ldw (i2sctl1),xxxxH	; Start I²S transmission.
13	ei xx	; Enable CPU interrupts.

(b) INTDMA0 interrupt routine

No	Instruction	Comments
1	res 0,(hdmae)	; Disable DMA channel 0.
2	:	
3	:	
4	:	
5	:	
6		
7		
8		
9		
10		
11	reti	;

3.7.5 Note

In case of using S/W start with HDMA, transmission start is to set to “1” DMAR register. However DMAR register can't be used to confirm flag of transmission end. DMAR register reset to “0” when HDMA release bus occupation once with HDMATR function. We recommend to use HDMACBn register (counter value) to confirm flag of transmission end.

3.7.6 Considerations for Using More Than One Bus Master

In the TMP92CF29A, the LCD controller, SDRAM controller, and DMA controller may act as the bus master apart from the CPU. Therefore, care must be exercised to enable each of these functions to operate smoothly.

To facilitate explanation of DMA operation performed by each bus master, the DMA transfer operation performed by the DMA controller is defined as “HDMA”, the display RAM read operation performed by the LCD controller as “LDMA”, and the SDRAM auto refresh operation performed by the SDRAM controller as “ARDMA”.

The following explains various cases where two or more bus masters may operate at the same time.

(1) CPU + HDMA

The DMA controller performs DMA transfer (HDMA) after issuing a bus request to the CPU and getting a bus acknowledgement. The DMA controller may be active while the CPU is in HALT mode (IDLE2 mode only), in which case HDMA does not interfere with the CPU operation. However, if HDMA is started while the CPU is active, the CPU cannot execute instructions while HDMA is being performed.

Before activating the DMA controller, therefore, it is necessary to estimate the CPU stop time (defined as “t_{STOP} (HDMA)”) based on the transfer time, transfer start interval, and number of channels to be used.

CPU bus stop rate = t_{STOP} (HDMA)[s] / HDMA start interval [s]

HDMA start interval [s] = HDMA start interrupt period [s]

Note: The HDMA start interval depends on the period of the HDMA start interrupt source. However, it is also possible to start HDMA by software.

t_{STOP} (HDMA) [s] = (Source read time + Destination write time) × Transfer count + α

state/byte

Memory Type / Read / Write	Internal RAM	External SDRAM 16-bit bus	External SRAM 16-bit bus	External SRAM 8-bit bus
Read	1 / 4 (Note 1)	Burst 1 / 2 (Note 2) 1 word 6 / 2 (Note 2)	2 / 2 (Note 3)	2 / 1 (Note 3)
Write	1 / 4	Burst 1 / 2 (Note 2) 1 word 3 / 2 (Note 2)	2 / 2 (Note 3)	2 / 1 (Note 3)

Note 1: 2-1-1-1 access. Each consecutive address can be accessed in 1 state.

Note 2: The transfer speed varies depending on the combination of source and destination.

a) When the source or destination is internal RAM or internal I/O (SFR), burst access (6-1-1-1 access) is possible. Only consecutive addresses on the same page can be accessed in 1 state. Additional 4 states are needed at the end of each burst access.

b) When the source or destination is other than internal RAM or internal I/O, 1-word access is used.

Note 3: In the case of 0 waits

state/byte

I/O Type / Read / Write	I^2S	NANDF	USB	SPI
Read	–	2 / 2	2 / 2	2 / 4
Write	2 / 4	2 / 2	2 / 2	2 / 4

Sample 1: Calculation example for CPU + HDMA

Conditions:

CPU operation speed (f_{SYS}) : 60 MHz

I^2S sampling frequency : 48 kHz (60 MHz/25/50 = 48 kHz)

I^2S data transfer bit length : 16 bits

DMAC channel 0 used to transfer 5 Kbytes from internal RAM to I^2S

Calculation example:

DMAC source data read time:

Internal RAM data read time

= 1 state/4 bytes (However, the first 1 byte requires 2 states.)

DMAC destination write time:

I^2S register write time = 2 states/4 bytes

Transfer count

To transfer 5 Kbytes of data in 4-byte units, the transfer count is calculated as follows:

5 Kbytes/4 bytes = 1280 [times]

Since I^2S generates an interrupt for every 64 bytes, the DMAC's counter A is set to 16 (64 bytes/4 bytes = 16 times) and counter B is set to 80.

Note: Since an interrupt is generated 80 times, the first read to internal RAM (which requires 1 additional state) occurs 80 times, requiring additional 80 states in total. In addition, from bus REQ to bus ACK, an overhead time of 2 states is also needed for each interrupt request, requiring additional 160 states in total.

t_{STOP} (HDMA) = (((1 + 2) × 16) × 80) + 80 + 160) / f_{SYS} [s] = 68 [μs]

HDMA start interval [s] = 1 / I^2S sampling frequency [Hz] × (64 / 16)

= 83.33 [ms]

CPU bus stop rate = t_{STOP} (HDMA) [s] / HDMA start interval [s]

= 68 [μs] / 83.33 [ms] = 0.08 [%]

(2) CPU + LDMA

The LCD controller performs DMA transfer (LDMA) after issuing a bus request to the CPU and getting a bus acknowledgement.

If LDMA is not performed properly, the LCD display function cannot work properly. Therefore, LDMA must have higher priority than the CPU. While LDMA is being performed, the CPU cannot execute instructions.

To display data on the LCD using the LCD controller, it is necessary to estimate to what degree LDMA would interfere with the CPU operation based on the display RAM type, display RAM bus width, LCDD type, display pixel count, and display quality.

The time the CPU stops operation while the LCD controller transfers data for one line is defined as "t_{STOP} (LDMA)", which is calculated as shown below for each display mode.

t_{STOP} (LDMA) = (SegNum × K / 8) × t_{LRD}

16-bit external SRAM	: t_{LRD} = (2 + wait count) / f_{SYS} [Hz] / 2
Internal RAM	: t_{LRD} = 1 / f_{SYS} [Hz] / 4
16-bit external SDRAM	: t_{LRD}= 1 / f_{SYS} [Hz] / 2
SegNum	: Number of segments to be displayed
K	: Number of bits needed for displaying 1 pixel

Monochrome	K = 1
4 gray scales	K = 2
16 gray scales	K = 4
256 colors	K = 8
4096 colors	K = 12
65536 colors	K = 16

Note 1: When SDRAM is used, the overhead time is added as shown below.

t_{STOP} [s] = (SegNum × K/8) × t_{LRD} + ((1/f_{SYS}) × 8)

Note 2: When internal RAM is used, the overhead time is added as shown below.

t_{STOP} [s] = (SegNum × K/8) × t_{LRD} + (1/f_{SYS})

The CPU bus stop rate indicates what proportion of the 1-line data update time t_{LP} is taken up by t_{STOP} (LDMA) and is calculated as follows:

CPU bus stop rate = t_{STOP} (LDMA) [s] / LHSYNC [period: s]

Sample2: Calculation examples for CPU + LDMA

Conditions 1:

CPU operation speed (f_{SYS}): 60 MHz
Display RAM : Internal RAM
Display size : QVGA (320seg × 240com)
Display quality : 65536 colors (TFT)
Refresh rate : 70 Hz (including 20 clocks of dummy cycles)

Calculation example 1:

t_{STOP} (LDMA) = ((SegNum × K / 8) × t_{LRD}) + (1 / f_{SYS} [Hz])
= ((320 × 16 / 8) × 1 / f_{SYS} [Hz] / 4) + (1 / f_{SYS} [Hz])
= ((640) × 16.67 [ns] / 4) + 16.67 [ns]
= 2.68 [μs]

LHSYNC [period: s] = 1/70 [Hz] / (COM+20=260) = 54.95 [μs]

CPU bus stop rate = t_{STOP} (LCD)[s] / LHSYNC [period: s]
= 2.68 [μs] / 54.95 [μs] = 4.88 [%]

Conditions 2:

CPU operation speed (f_{SYS}) : 10 MHz
Display RAM : 16-bit external SRAM (0 waits)
Display size : QVGA (240seg × 320com)
Display quality : 4096 colors (STN)
Refresh rate : 100 Hz (0 dummy cycles)

Calculation example 2:

t_{STOP} (LDMA) = (SegNum × K / 8) × t_{LRD}
= (240 × 12 / 8) × (2 + wait count) / f_{SYS} [Hz] / 2
= (360) × 200 [ns] / 2
= 36 [μs]

LHSYNC [period: s] = 1/100 [Hz] / (COM = 240) = 41.67 [μs]

CPU bus stop rate = t_{STOP} (LCD)[s] / LHSYNC [period: s]
= 36 [μs] / 41.67 [μs] = 86.40 [%]

(3) CPU + LDMA + ARDMA

The SDRAM controller owns the bus not only when SDRAM is used as the LCD display RAM but also when SDRAM is used as work, data, or stack area. The SDRAM controller occupies the bus (ARDMA) while it refreshes SDRAM data by the auto refresh function.

No special consideration is needed for the ARDMA time normally as it ends within several clocks per specified number of states. However, if the LCD controller occupies the bus continuously, ARDMA cannot be executed at normal intervals and refresh data is stored in a counter specifically provided in the SDRAM controller. In this case, ARDMA is executed successively after the LCD controller releases the bus.

The priorities among the three bus masters should be set in the order of LCDC > SDRAMC > CPU. The time the CPU stops operation while the LCD controller and SDRAM controller are transferring data for one line is defined as "t_{STOP} (LDMA・ARDMA)", which is calculated as follows:

t_{STOP} (LDMA・ARDMA) = t_{STOP} (LDMA)[s] − (t_{STOP} (LDMA)[s] / AR interval [s] × 2 / f_{SYS} [Hz])

CPU bus stop rate = t_{STOP} (LDMA・ARDMA)[s] / LHSYNC [period: s]

Auto Refresh Intervals

Unit: [μs]

SDRCR<SRS2:0>			Auto Refresh Interval (states)	Frequency (System Clock)					
SRS2	SRS1	SRS0		6 MHz	10MHz	20MHz	40MHz	60MHz	80MHz
0	0	0	47	7.8	4.7	2.4	1.18	0.78	0.59
0	0	1	78	13.0	7.8	3.9	1.95	1.30	0.98
0	1	0	156	26.0	15.6	7.8	3.90	2.60	1.95
0	1	1	312	52.0	31.2	15.6	7.80	5.20	3.90
1	0	0	468	78.0	46.8	23.4	11.70	7.80	5.85
1	0	1	624	104.0	62.4	31.2	15.60	10.40	7.80
1	1	0	936	156.0	93.6	46.8	23.40	15.60	11.70
1	1	1	1248	208.0	124.8	62.4	31.20	20.80	15.60

Sample3: Calculation example for CPU + LDMA + ARDMA

Conditions:

CPU operating speed (f_{SYS})	: 60 MHz
Display RAM	: 16-bit external SDRAM
Display size	: QVGA (320seg × 240com)
Display quality	: 65536 colors (TFT)
Refresh rate	: 70 Hz (including 20 clocks of dummy cycles)
SDRAM auto refresh	: Every 936 states (15.6 μs)

Calculation example:

t_{STOP} (LDMA) $=((SegNum \times K / 8) \times t_{LRD}) + (8 / f_{SYS}\ [Hz])$

$= ((320 \times 16 / 8) \times 1 / f_{SYS}\ [Hz] / 2) + (8 / f_{SYS}\ [Hz])$

$= ((640) \times 16.67\ [ns] / 2) + 133.33\ [ns]$

$= 5.47\ [\mu s]$

LHSYNC [period:s] = 1/70 [Hz] / (COM + 20 = 260) = 54.95 [μs]

Since SDRAM is auto-refreshed once or less in 5.47 [μs]:

t_{STOP} (ARDMA) $= 2 / f_{SYS}\ [Hz] = 33.33\ [ns]$

CPU bus stop rate $= t_{STOP}$ (LDMA・ARDMA) [s] / LHSYNC [period:s]

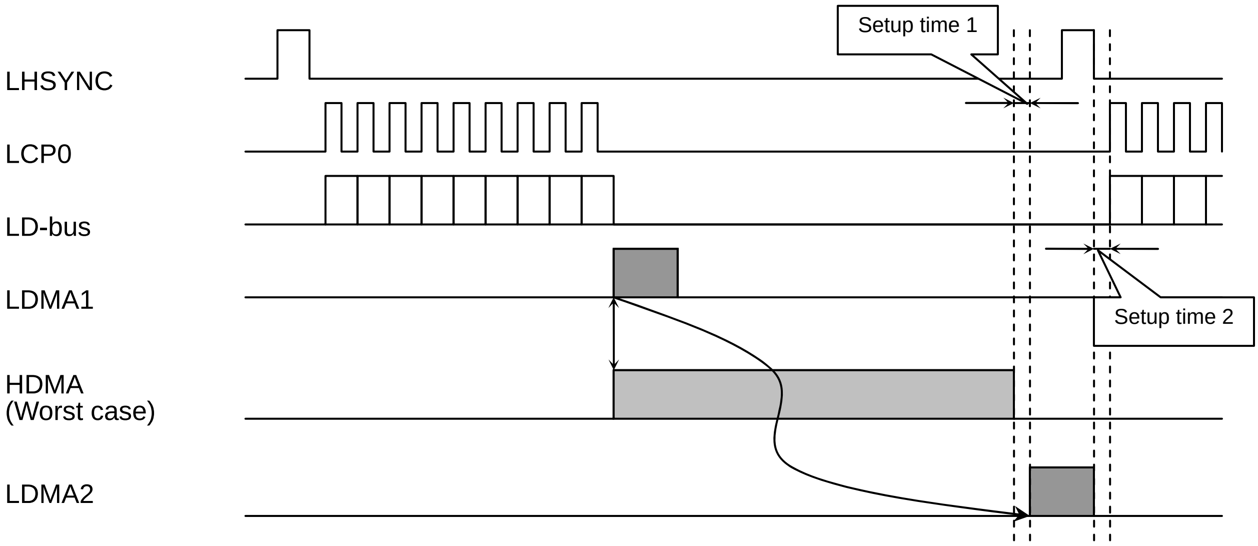
$= (5.47\ [\mu s] + 33.33\ [ns]) / 54.95\ [\mu s] = 10.01\ [\%]$

(4) CPU + LDMA+ ARDMA + HDMA

This is a case in which all the bus masters are active at the same time.

Since the LCD display function cannot work properly if the LCD controller cannot perform LDMA properly, the priorities among the four bus masters should be set in the order of LDMA > ARDMA > HDMA > CPU.

Before calculating the CPU bus stop rate, the conditions for proper LCD display shall be considered first.



The above diagram shows the LHSYNC signal, LCP0 signal, and LD-bus signal for transferring data from the LCD controller to the LCD driver, and the transfer operation (LDMA1) for reading data from the display RAM into the FIFO buffer in the LCD controller.

LDMA is started immediately after data has been transferred to the LCD driver. If HDMA is started immediately before LDMA1 is started, LDMA must wait until HDMA has finished before it can be started (LDMA2). LDMA2 must finish operation before the LCD driver output for the next stage is started.

LHSYNC [period: s] − LCD driver data transfer time [s] − t_{STOP} (LCD) [s]

= HDMA continuous time [s] + CPU operation time [s]

In the case of STN display

LCD driver data transfer time [s] = SegNum/8 × (1/f_{SYS}) × (LD bus transfer speed)

In the case of TFT display

LCD driver data transfer time [s] = SegNum × (1/ f_{SYS}) × (LD bus transfer speed)

Sample 4: Calculation example for CPU + LDMA+ ARDMA + HDMA

Conditions:

CPU operation speed (f_{SYS}): 60 MHz

Display RAM	: QVGA (320seg × 240com)
Display quality	: 65536 colors (TFT)
Refresh rate	: 70 Hz (including 20 clocks of dummy cycles)
SDRAM Auto Refresh	: Every 936 states (15.6 μs)
SDRAM	: 16-bit width
HDMA	: Transfers 5 Kbytes from internal RAM to I^2S

Calculation example:

t_{STOP} (LDMA) =((SegNum × K / 8) × t_{LRD}) + (1 / f_{SYS} [Hz])

= ((320 ×16 / 8) × 1 / f_{SYS} [Hz] / 4) + (1 / f_{SYS} [Hz])

= ((640) ×16.67 [ns] / 4) + 16.67 [ns]

= 2.68 [μs]

LHSYNC [period: s] = 1/70 [Hz] /(COM+20 = 260) = 54.95 [μs]

t_{STOP} (HDMA) = (((1 + 2) × 16) × 80) + 80 + 160) / f_{SYS} [s] = 68 [μs]

LCD driver data transfer time [s]

= SegNum × (1/ f_{SYS}) × (LD bus transfer speed)

= 320 × (1/60 MHz) × 16 = 85 [μs]

Since LHSYNC [period: s] < LCD driver data transfer time [s], this setting is not possible.

When the transfer speed is changed to × 4, the LCD driver data transfer time is calculated as follows:

(The transfer speed should be adjusted according to the required specifications.)

LCD driver data transfer time [s]

= SegNum × (1/ f_{SYS}) × (LD bus transfer speed)

= 320 × (1 / 60MHz) × 4 = 21.3 [μs]

LHSYNC [period: s] – LCD driver data transfer time [s] – t_{STOP} (LDMA)

= 54.95 [μs] – 21.3 [μs] – 2.68 [μs] = 30.94 [μs]

To realize proper LCD display, the maximum time HDMA can occupy the bus at a time (maximum HDMA time) must be set to 30.92 [μs] or less. Although transferring all 5Kbytes from the internal RAM to I^2S requires t_{STOP} (HDMA) = 68 [μs], the maximum HDMA time should be limited by using the HDMATR register.

HDMATR Register

HDMATR (097FH)		7	6	5	4	3	2	1	0
	bit Symbol	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by "Maximum bus occupancy time / (256/f_{SYS})". "00H" cannot be set.						

Note: Read-modify-write instructions can be used on this register.

By writing "87H" to the HDMATR register, the maximum HDMA time is set to 29.9 [μs] (256 × 7 × (1 / f_{SYS})). Since HDMA start interval [period:s] = 83.33 [ms] is longer than LHSYNC [period:s] = 54.95 [μs], it is assumed that HDMA transfer occurs once during LHSYNC [period:s].

Since SDRAM is auto-refreshed once or less in 5.47 [μs]:

t_{STOP} (ARDMA) = 2 / f_{SYS} [Hz] = 33.33 [ns]

The time LDMA, ARDMA, and HDMA all occupy the bus is defined as:

t_{STOP} (LDMA・ARDMA・HDMA)

Based on the above, the CPU bus stop rate is calculated as follows:

CPU bus stop rate = t_{STOP} (LDMA・ARDMA・HDMA) [s] / LHSYNC [period:s]

= (5.47 [μs] + 33.33 [ns] + 29.9 [μs]) / 54.95 [μs] = 64.42 [%]

Note: To be precise, the bus assert time and RAM access time are added each time the HDMA transfer time is forcefully terminated at 29.9 [μs].

3.8 Function of ports

The TMP92CF29A I/O port pins are shown in Table 3.8.1. In addition to functioning as general-purpose I/O ports, these pins are also used by the internal CPU and I/O functions. Table 3.8.2 lists the I/O registers and their specifications.

Table 3.8.1 Port Functions (1/2) (R: PD= with programmable pull-down resistor, U= with pull-up resistor)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for built-in function
Port 1	P10 to P17	8	I/O	–	bit	D8 to D15
Port 4	P40 to P47	8	Output	–	(Fixed)	A0 to A7
Port 5	P50 to P57	8	Output	–	(Fixed)	A8 to A15
Port 6	P60 to P67	8	I/O	–	bit	A16 to A23
Port 7	P70	1	Output	–	(Fixed)	$\overline{RD}$
	P71	1	I/O	–	bit	$\overline{WRLL}$, $\overline{NDRE}$
	P72	1	I/O	–	bit	$\overline{WRLU}$, $\overline{NDWE}$
	P73	1	I/O	–	bit	EA24
	P74	1	I/O	–	bit	EA25
	P75	1	I/O	–	bit	R/$\overline{W}$, NDR/$\overline{B}$
	P76	1	I/O	–	bit	$\overline{WAIT}$
Port 8	P80	1	Output	–	(Fixed)	$\overline{CS0}$
	P81	1	Output	–	(Fixed)	$\overline{CS1}$, $\overline{SDCS}$
	P82	1	Output	–	(Fixed)	$\overline{CS2}$, $\overline{CSZA}$, $\overline{SDCS}$
	P83	1	Output	–	(Fixed)	$\overline{CS3}$, $\overline{CSXA}$
	P86	1	Output	–	(Fixed)	$\overline{CSZD}$, $\overline{ND0CE}$
	P87	1	Output	–	(Fixed)	$\overline{CSXB}$, $\overline{ND1CE}$
Port 9	P90	1	I/O	–	bit	TXD0, TXD1
	P91	1	I/O	–	bit	RXD0, RXD1
	P92	1	I/O	–	bit	SCLK0, $\overline{CTS0}$, SCLK1, $\overline{CTS1}$
	P96	1	Input	PD	(Fixed)	INT4, PX
	P97	1	Input	–	(Fixed)	PY
Port A	PA0 to PA7	8	Input	U	(Fixed)	KI0 to KI7
Port C	PC0	1	I/O	–	bit	INT0
	PC1	1	I/O	–	bit	INT1, TA0IN
	PC2	1	I/O	–	bit	INT2
	PC3	1	I/O	–	bit	INT3, TA2IN
	PC4	1	I/O	–	bit	EA26
	PC5	1	I/O	–	bit	EA27
	PC6	1	I/O	–	bit	EA28
	PC7	1	I/O	–	bit	KO8
Port F	PF0	1	I/O	–	bit	I2S0CKO
	PF1	1	I/O	–	bit	I2S0DO
	PF2	1	I/O	–	bit	I2S0WS
	PF7	1	Output	–	(Fixed)	SDCLK
Port G	PG0 to PG1	2	Input	–	(Fixed)	AN0 to AN1
	PG2	1	Input	–	(Fixed)	AN2, MX
	PG3	1	Input	–	(Fixed)	AN3, $\overline{ADTRG}$, MY
	PG4 to PG5	2	Input	–	(Fixed)	AN4 to AN5

Table 3.8.1 Port Functions (2/2)

Port Name	Pin Name	Number of Pins	I/O	R	I/O Setting	Pin Name for built-in function
Port J	PJ0	1	Output	–	(Fixed)	$\overline{\text{SDRAS}}$, $\overline{\text{SRLLB}}$
	PJ1	1	Output	–	(Fixed)	$\overline{\text{SDCAS}}$, $\overline{\text{SRLUB}}$
	PJ2	1	Output	–	(Fixed)	$\overline{\text{SDWE}}$, $\overline{\text{SRWR}}$
	PJ3	1	Output	–	(Fixed)	SDLLDQM
	PJ4	1	Output	–	(Fixed)	SDLUDQM
	PJ5	1	I/O	–	bit	NDALE
	PJ6	1	I/O	–	bit	NDCLE
	PJ7	1	Output	–	(Fixed)	SDCKE
Port K	PK0	1	Output	–	(Fixed)	LCP0
	PK1	1	Output	–	(Fixed)	LLOAD
	PK2	1	Output	–	(Fixed)	LFR
	PK3	1	Output	–	(Fixed)	LVSYNC
	PK4	1	Output	–	(Fixed)	LHSYNC
	PK5	1	Output	–	(Fixed)	LGOE0
	PK6	1	Output	–	(Fixed)	LGOE1
	PK7	1	Output	–	(Fixed)	LGOE2
Port L	PL0 to PL7	8	I/O	–	bit	LD0 to LD7
Port M	PM1	1	Output	–	(Fixed)	MLDALM, TA1OUT
	PM2	1	Output	–	(Fixed)	$\overline{\text{ALARM}}$, $\overline{\text{MLDALM}}$
	PM7	1	Output	–	(Fixed)	PWE
Port N	PN0 to PN7	8	I/O	–	bit	KO0 to KO7
Port P	PP3	1	I/O	–	bit	INT5, TA7OUT, TXD0, TXD1
	PP4	1	I/O	–	bit	INT6, TB0IN0, RXD0, RXD1
	PP5	1	I/O	–	bit	INT7, TB1IN0, SCLK0, $\overline{\text{CTS0}}$ SCLK1, $\overline{\text{CTS1}}$
	PP6	1	Output	–	(Fixed)	TB0OUT0
Port R	PR0	1	I/O	–	bit	SPDI
	PR1	1	I/O	–	bit	SPDO
	PR2	1	I/O	–	bit	$\overline{\text{SPCS}}$
	PR3	1	I/O	–	bit	SPCLK
Port T	PT0 to PT7	8	I/O	–	bit	LD8 to LD15
Port V	PV6	1	I/O	–	bit	SDA
	PV7	1	I/O	–	bit	SCL
Port X	PX4	1	Output	–	(Fixed)	CLKOUT, LDIV
	PX5	1	I/O	–	bit	X1USB, X1D4

Table 3.8.2 I/O Port and Specifications (1/5) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 1	P10 toP17	Input port	X	0	0	None
		Output port	X	1		
		D8 to D15 bus	X	X	1	
Port 4	P40 to P47	Output port	X	None	0	None
		A0 to A7 Output	X	None	1	
Port 5	P50 to P57	Output port	X	None	0	None
		A8 to A15 Output	X	None	1	
Port 6	P60 to P67	Input port	X	0	0	None
		Output port	X	1		
		A16 to A23 Output	X	X	1	
Port 7	P70 to P76	Output port	X	1	0	None
	P71 to P76	Input port	X	0	0	
	P70	$\overline{RD}$ Output	X	None	1	
	P71	$\overline{WRLL}$ Output	1	1	1	
		$\overline{NDRE}$ Output	0			
	P72	$\overline{WRLU}$ Output	1	1	1	
		$\overline{NDWE}$ Output	0			
	P73	EA24 Output	X	1	1	
	P74	EA25 Output	X	1	1	
	P75	R/$\overline{W}$ Output	X	1	1	
		NDR/B Input	X	0	1	
	P76	$\overline{WAIT}$ Input	X	0	1	
Port 8	P80 to P87	Output port	X	None	0	0
	P80	$\overline{CS0}$ Output	X		1	None
	P81	$\overline{CS1}$ Output	X		1	0
		$\overline{SDCS}$ Output	X		X	1
	P82	$\overline{CS2}$ Output	X		1	0
		$\overline{CSZA}$ Output	X		0	1
		$\overline{SDCS}$ Output	X		1	1
	P83	$\overline{CS3}$ Output	X		1	0
		$\overline{CSXA}$ Output	X		X	1
	P86	$\overline{CSZD}$ Output	X		1	0
		$\overline{ND0CE}$ Output	X		1	1
	P87	$\overline{CSXB}$ Output	X		1	0
		$\overline{ND1CE}$ Output	X		1	1

Table3.8.2 I I/O Port and Specifications (2/5) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port 9	P90, P92	Input port	X	0	0	None
	P91	Input port, RXD0 Input	X	0	None	None
	P96	Input port	X	None	0	None
	P97	Input port	X	None	None	None
	P90 to P92	Output port	X	1	0	0
	P90	TXD0 Output	X	1	1	0
		TXD0 Output (Open-drain)	X	1	1	1
		TXD1 Output	X	1	1	0
		TXD1 Output (Open-drain)	X	1	1	1
	P92	SCLK0 Output	X	1	1	<P95F2> =0
		SCLK0, $\overline{CTS0}$ Input	X	0	0	0
		SCLK1 Output	X	1	1	<P95F2> =1
		SCLK1, $\overline{CTS1}$ Input	X	0	0	0
	P96	INT4 Input	X	None	1	None
Port A	PA0 to PA7	Input port	X	None	0	None
		KI0 to KI7 Input	X		1	None
Port C	PC0 to PC3 PC5 to PC7	Input port	X	0	0	None
		Output port	X	1	0	None
	PC4	Input port	X	0	0	X
		Output port	X	1	0	X
	PC0	INT0 Input	X	0	1	None
	PC1	INT1 Input	X	0	1	None
		TA0IN Input	X	1	1	None
	PC2	INT2 Input	X	0	1	None
	PC3	INT3 Input	X	0	1	None
		TA2IN Input	X	1	1	None
	PC4	EA26 Output	X	0	1	X
		(PC4) SPDI Input	X	1	1	1
		(PR0) SPDI Input	X	1	1	0
	PC5	EA27 Output	X	0	1	None
		SPDO Output	X	1	1	None
	PC6	EA28 Output	X	0	1	None
		SPCLK Output	X	1	1	None
	PC7	KO8 Output (Open-drain)	X	1	1	None
Port F	PF0 to PF2	Input port	X	0	0	None
	PF0 to PF2	Output port	X	1	0	None
	PF7	Output port	X	None	0	
	PF0	I2S0CKO Output	X	X	1	
	PF1	I2S0DO Output	X	X	1	
	PF2	I2S0WS Output	X	X	1	
	PF7	SDCLK Output	X	None	1	

Table3.8.2 I/O Port and Specifications (3/5) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port G	PG0 to PG5	Input port	X	None	None	None
		AN0 to AN5 Input				
	PG3	$\overline{ADTRG}$ Input			1	
	PG2	MX Output Note:			None	
	PG3	MY Output Note:				
Port J	PJ5 to PJ6	Input port	X	0	0	None
	PJ5 to PJ6	Output port	X	1	0	
	PJ0 to PJ4, PJ7	Output port	X	None	0	
	PJ0	$\overline{SDRAS}$, $\overline{SRLLB}$ Output	X	None	1	
	PJ1	$\overline{SDCAS}$, $\overline{SRLUB}$ Output	X		1	
	PJ2	$\overline{SDWE}$, $\overline{SRWR}$ Output	X		1	
	PJ3	SDLLDQM Output	X		1	
	PJ4	SDLUDQM Output	X		1	
	PJ5	NDALE Output	X	1	1	
	PJ6	NDCLE Output	X	1	1	
	PJ7	SDCKE Output	X	None	1	
Port K	PK0 to PK7	Output port	X	None	0	None
	PK0	LCP0 output	X		1	
	PK1	LLOAD output	X		1	
	PK2	LFR output	X		1	
	PK3	LVSYNC output	X		1	
	PK4	LHSYNC output	X		1	
	PK5	LGOE0 output	X		1	
	PK6	LGOE1 output	X		1	
	PK7	LGOE2 output	X		1	
Port L	PL0 to PL7	Input port	X	0	0	None
		Output port	X	1	0	
		LD0 to LD7 Output	X	X	1	
Port M	PM1, PM2, PM7	Output port	X	None	0	None
	PM1	TA1OUTOutput	0		1	
		MLDALM Output	1		1	
	PM2	$\overline{MLDALM}$ Output	0		1	
		$\overline{ALARM}$ Output	1		1	
	PM7	PWE Output	X		1	
Port N	PN0 to PN7	Input port	X	0	0	None
		Output port (CMOS Output)	X	1	0	
		KO Output (Open-drain Output)	X		1	

Note: Case of using touch screen.

Table 3.8.2 I/O Port and Specifications (4/5) X: Don’t care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port P	PP3 to PP5	Input port	X	0	0	<PP1F2:3F2>=0
		Output port	X	1	0	<PP1F2:3F2>=0
	PP6	Output port	X	None	0	None
	PP3	INT5 input	X	0	1	<PP1F2>=0
		TA7OUT Output	X	1	1	<PP1F2>=0
		TXD0 Output	X	X	X	<PP0F2>=0 <PP1F2>=1 <PP4F2>=1
		TXD0 Output (Open-drain)	X	X	X	<PP0F2>=1 <PP1F2>=1 <PP4F2>=1
		TXD1 Output	X	X	X	<PP0F2>=0 <PP1F2>=1 <PP4F2>=0
		TXD1 Output (Open-drain)	X	X	X	<PP0F2>=1 <PP1F2>=1 <PP4F2>=0
	PP6	INT6 Input	X	0	1	<PP2F2>=0
		TB0IN0 Input	X	1	1	<PP2F2>=0
		RXD1 (PP4/RXD1) Input	X	X	X	<PP2F2>=1 <PP5F2>=0
		RXD1(P91/RXD1) Input	X	X	X	<PP2F2>=1 <PP5F2>=1
	PP5	INT7 Input	X	0	1	<PP3F2>=0
		TB1IN0 Input	X	1	1	<PP3F2>=0
		SCLK1 (PP5/SCLK1) Input $\overline{\text{CTS1}}$ Input	X	0	X	<PP3F2>=1 <PP6F2>=0
		SCLK1 (P92/SCLK1) Input $\overline{\text{CTS1}}$ Input	X	0	X	<PP3F2>=1 <PP6F2>=1
		SCLK0 Output	X	1	X	<PP3F2>=1 <PP6F2>=1
		SCLK1 Output	X	1	X	<PP3F2>=1 <PP6F2>=0
	PP7	TB1OUT0 Output	X	None	1	None
Port R	PR0 to PR3	Input port	X	0	0	None
		Output port	X	1	0	
	PR0	SPDI_PR0 Input (to PC4)	X	0	1	
	PR1	SPDO Output	X	1	1	
	PR2	$\overline{\text{SPCS}}$ Output	X	1	1	
	PR3	SPCLK Output	X	1	1	
Port T	PT0 to PT7	Input port	X	0	0	None
		Output port	X	1	0	
		LD8 to LD15 Output	X	X	1	
Port V	PV6 to PV7	Input port	X	0	0	0
		Output port	X	1	0	0
		Output port (Open-drain)	X	1	0	1
	PV6	SDA I/O	X	1	1	0
		SDA I/O (Open-drain)	X	1	1	1
	PV7	SCL I/O	X	1	1	0
		SCL I/O (Open-drain)	X	1	1	1

Table 3.8.2 I/O Port and Specifications (5/5) X: Don't care

Port	Pin name	Specification	I/O register			
			Pn	PnCR	PnFC	PnFC2
Port X	PX5	Input port	X	0	0	None
	PX4	Output port	X	None	0	None
	PX5	Output port	X	1	0	None
	PX4	CLKOUT Output	0	None	1	None
		LDIV Output	1		1	None
	PX5	X1USB Input	X	0	1	None
		X1D4 Output (Output clock = ×1/8)	1	1	1	<PX5F2:4F2>=00
		X1D4 Output (Output clock = ×1/4)	1	1	1	<PX5F2:4F2>=01
		X1D4 Output (Output clock = ×1/2)	1	1	1	<PX5F2:4F2>=10
		X1D4 Output (Output clock = ×1/1)	1	1	1	<PX5F2:4F2>=11

3.8.1 Port 1 (P10 to P17)

Port1 is an 8-bit general-purpose I/O port. Bits can be individually set as either inputs or outputs by control register P1CR and function register P1FC.

In addition to functioning as a general-purpose I/O port, port1 can also function as a data bus (D8 to D15).

Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 1 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Data bus (D8 to D15)
1	0	Don't use this setting
1	1	Input port (P10 to P17)

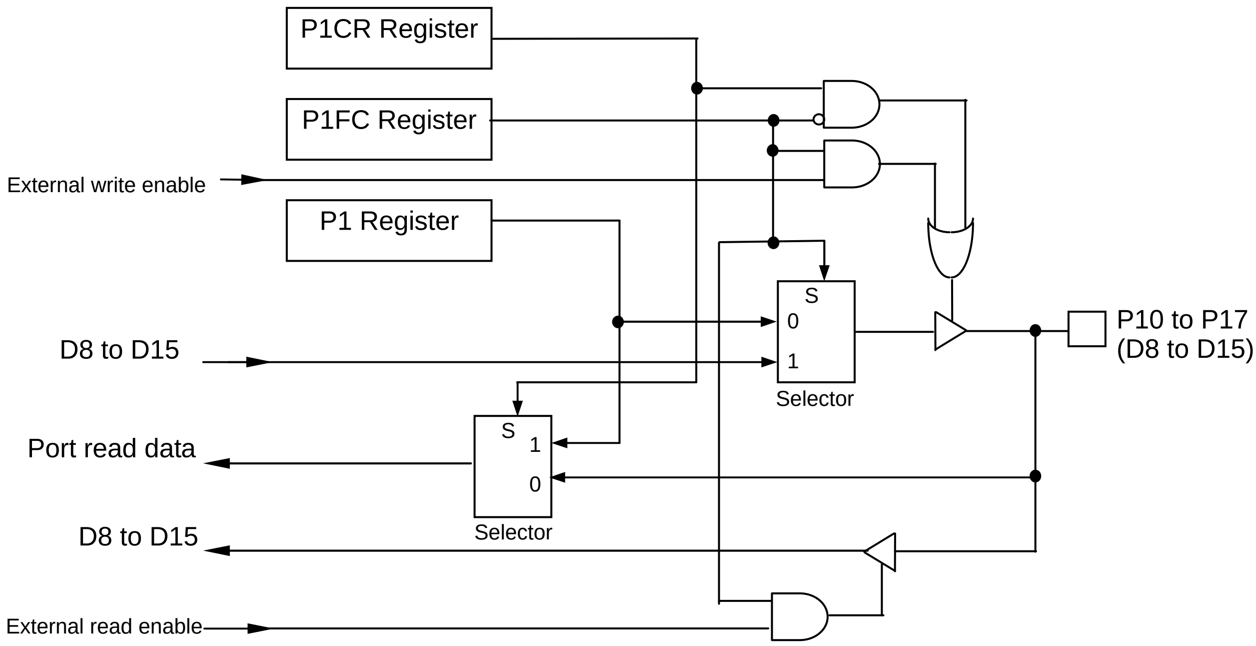


Figure 3.8.1 Port1

Port 1 register

P1 (0004H)		7	6	5	4	3	2	1	0
	bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
	Read/Write	R/W							
	System Reset State	Data from external port (Output latch register is cleared to “0”)							
	Hot Reset State	–							

Port 1 Control register

P1CR (0006H)		7	6	5	4	3	2	1	0
	bit Symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: Input 1: Output							

Port 1 Function register

P1FC (0007H)		7	6	5	4	3	2	1	0
	bit Symbol								P1F
	Read/Write								W
	System Reset State (Note2)								0/1
	Hot Reset State								–
	Function								0: Port 1:Data bus (D8 to D15)

Port 1 Drive register

P1DR (0081H)		7	6	5	4	3	2	1	0
	bit Symbol	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for P1CR, P1FC.

Note2: It is set to “Port” or “Data bus” by AM pins state.

Figure 3.8.2 Register for Port1

3.8.2 Port 4 (P40 to P47)

Port4 is an 8-bit general-purpose Output ports. In addition to functioning as a general-purpose Output port, port4 can also function as an address bus (A0 to A7). Each bit can be set individually for function. Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 4 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus (A0 to A7)
1	0	Don't use this setting
1	1	Output port (P40 to 47)

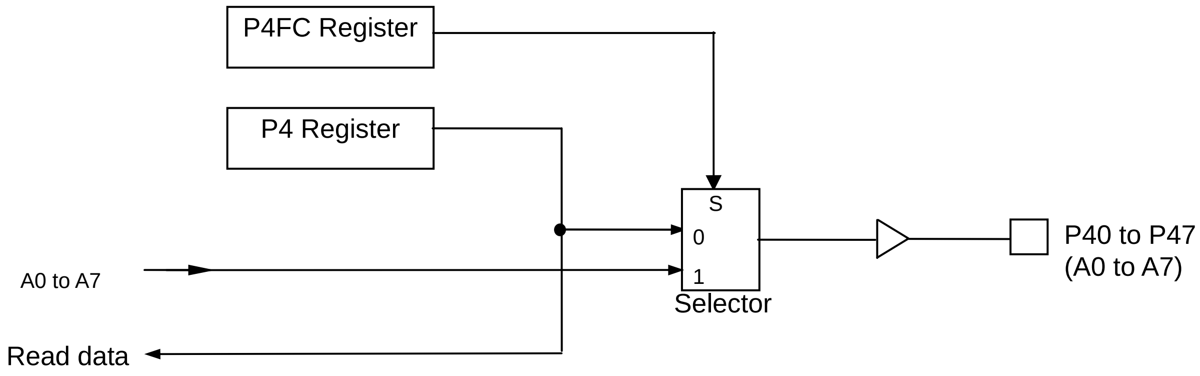


Figure 3.8.3 Port4

Port 4 register

P4 (0010H)		7	6	5	4	3	2	1	0
	bit Symbol	P47	P46	P45	P44	P43	P42	P41	P40
	Read/Write	R/W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–

Port 4 Function register

P4FC (0013H)		7	6	5	4	3	2	1	0
	bit Symbol	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
	Read/Write	W							
	System Reset State (Note2)	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0:Port 1:Address bus (A0 to A7)							

Port 4 Drive register

P4DR (0084H)		7	6	5	4	3	2	1	0
	bit Symbol	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for P4FC.

Note2: It is set to “Port” or “Address bus” by AM pins state.

Figure 3.8.4 Register for Port1

3.8.3 Port 5 (P50 to P57)

Port5 is an 8-bit general-purpose Output ports. In addition to functioning as a general-purpose I/O port, port5 can also function as an address bus (A8 to A15). Each bit can be set individually for function. Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 5 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus (A8 ~ A15)
1	0	Don't use this setting
1	1	Output port (P50 ~ P57)

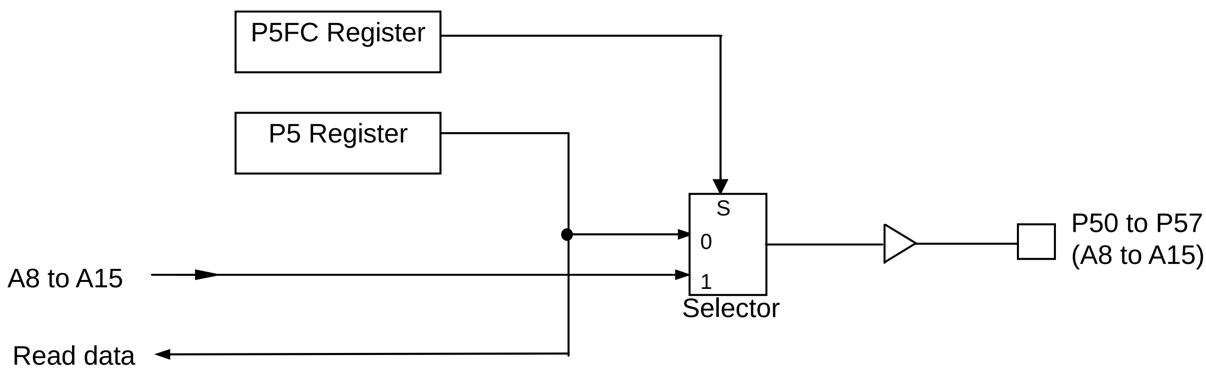


Figure 3.8.5 Port5

Port 5 register

		7	6	5	4	3	2	1	0
P5 (0014H)	bit Symbol	P57	P56	P55	P54	P53	P52	P51	P50
	Read/Write	R/W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–

Port 5 Function register

		7	6	5	4	3	2	1	0
P5FC (0017H)	bit Symbol	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
	Read/Write	W							
	System Reset State (Note2)	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0:Port 1:Address bus (A8 to A15)							

Port 5 Drive register

		7	6	5	4	3	2	1	0
P5DR (0085H)	bit Symbol	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for P5FC.

Note2: It is set to “Port” or “Address bus” by AM pins state.

Figure 3.8.6 Register for Port5

3.8.4 Port 6 (P60 to P67)

Port6 is an 8-bit general-purpose I/O ports. Bits can be individually set as either inputs or outputs and function by control register P6CR and function register P6FC.

In addition to functioning as a general-purpose I/O port, port6 can also function as an address bus (A16 to A23). Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 6 to the following function pins:

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	Address bus(A16 ~ A23)
1	0	Don't use this setting
1	1	Input port (P60 ~ P67)

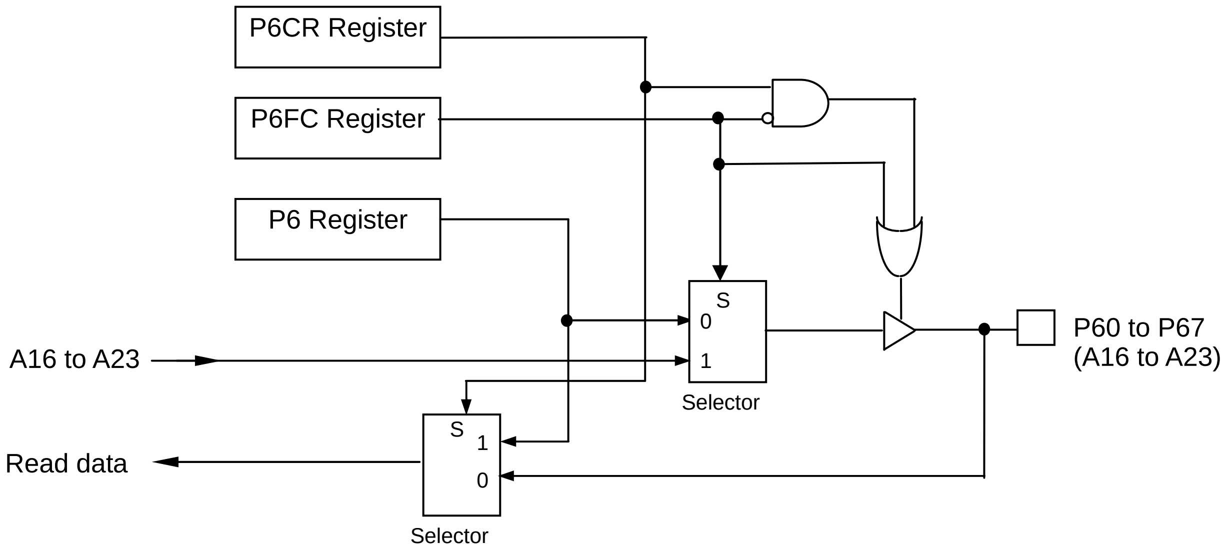


Figure 3.8.7 Port6

Port 6 register

P6 (0018H)

	7	6	5	4	3	2	1	0
bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R/W							
System Reset State	Data from external port (Output latch register is cleared to “0”)							
Hot Reset State	–							

Port 6 Control register

P6CR (001AH)

	7	6	5	4	3	2	1	0
bit Symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	0:Input 1:Output							

Port 6 Function register

P6FC (001BH)

	7	6	5	4	3	2	1	0
bit Symbol	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
Read/Write	W							
System Reset State (Note2)	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Hot Reset State	–	–	–	–	–	–	–	–
Function	0: Port 1:Address bus (A16 to A23)							

Port 6 Drive buffer register

P6DR (0086H)

	7	6	5	4	3	2	1	0
bit Symbol	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
Read/Write	R/W							
System Reset State	1	1	1	1	1	1	1	1
Hot Reset State	–	–	–	–	–	–	–	–
Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for P6CR, P6FC.

Note2: It is set to “Port” or “Address bus” by AM pins state.

Figure 3.8.8 Register for Port 6

3.8.5 Port 7 (P70 to P76)

Port7 is a 7-bit general-purpose I/O port (P70 is used for output only). Bits can be individually set as either inputs or outputs by control register P7CR and function register P7FC.

In addition to functioning as a general-purpose I/O port, P70 to P76 pins can also function as interface-pins for external memory.

A reset initializes P70 pin to output port mode, and P71 to P76 pins to input port mode.

Setting the AM1 and AM0 pins as shown below and resetting the device initialize port 7 to the following function pins:

Initial setting of P70 pin

AM1	AM0	Function Setting after reset is released
0	0	Don't use this setting
0	1	$\overline{RD}$ pin
1	0	Don't use this setting
1	1	Output port (P70)

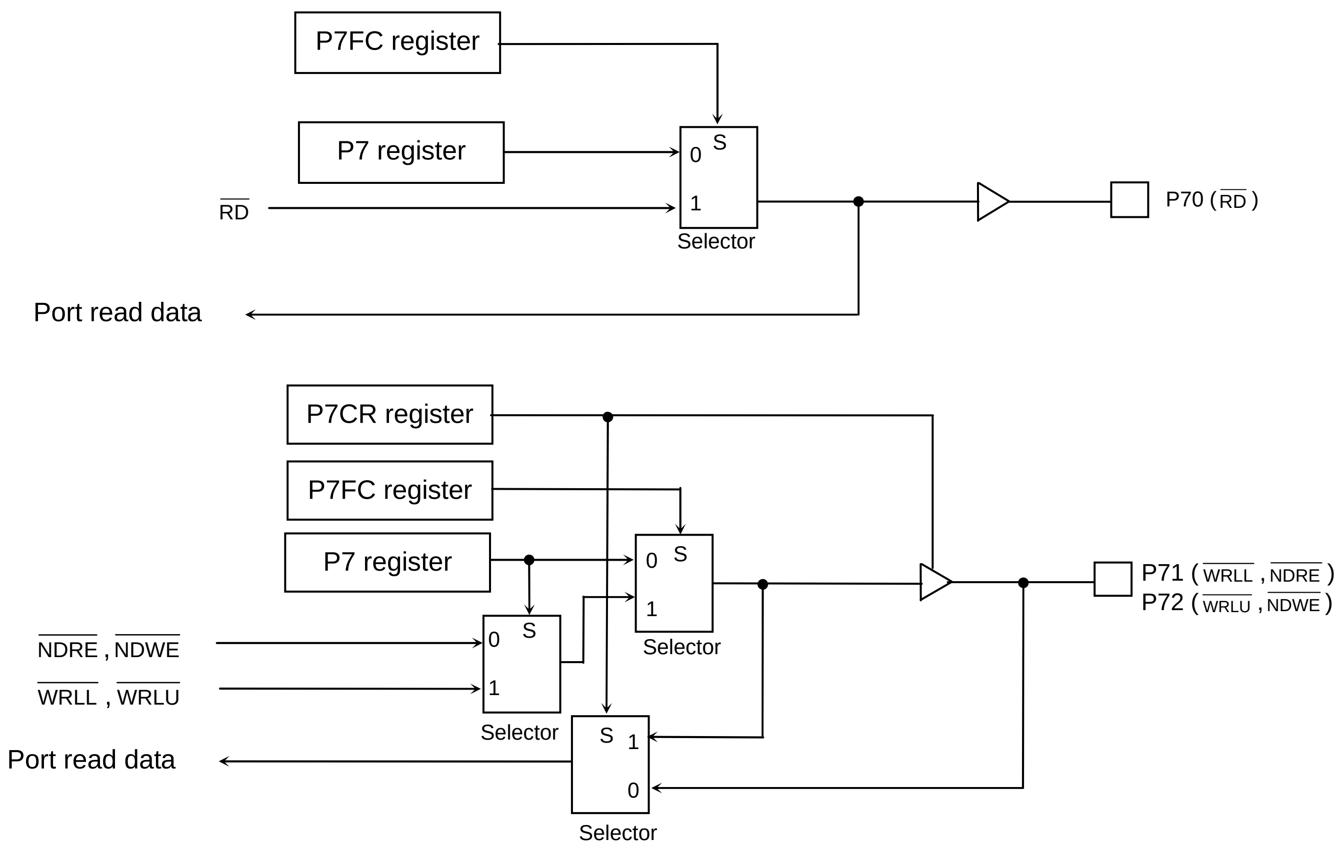
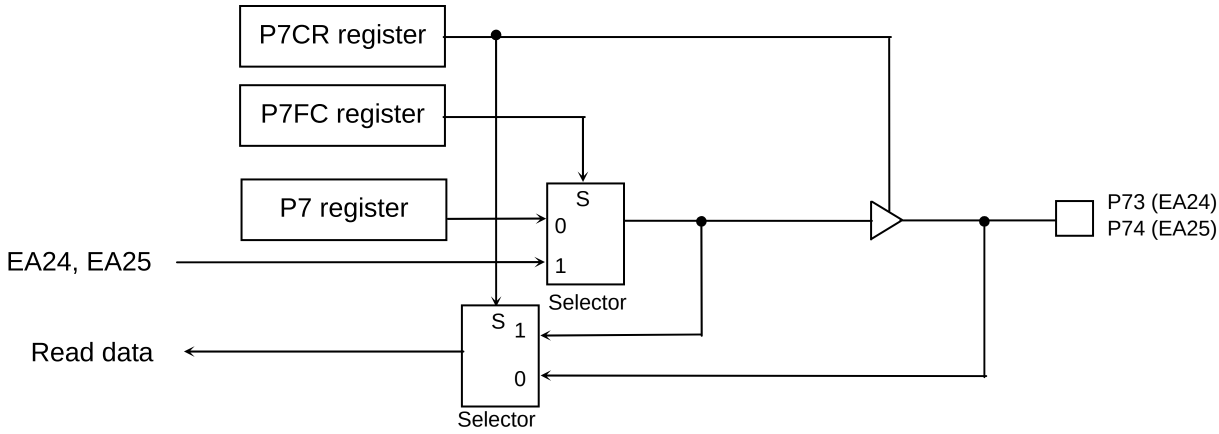
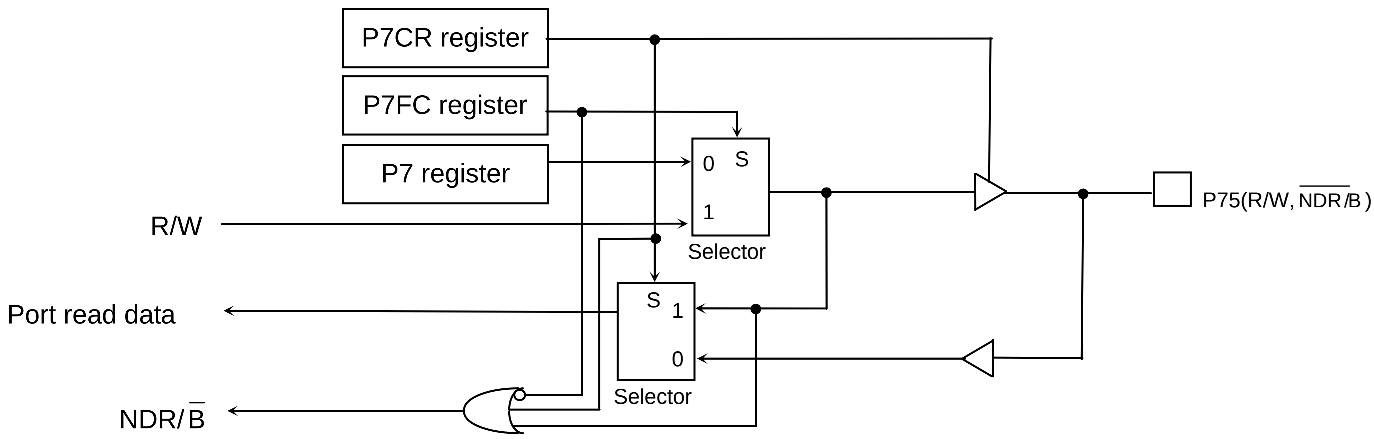


Figure 3.8.9 Port7





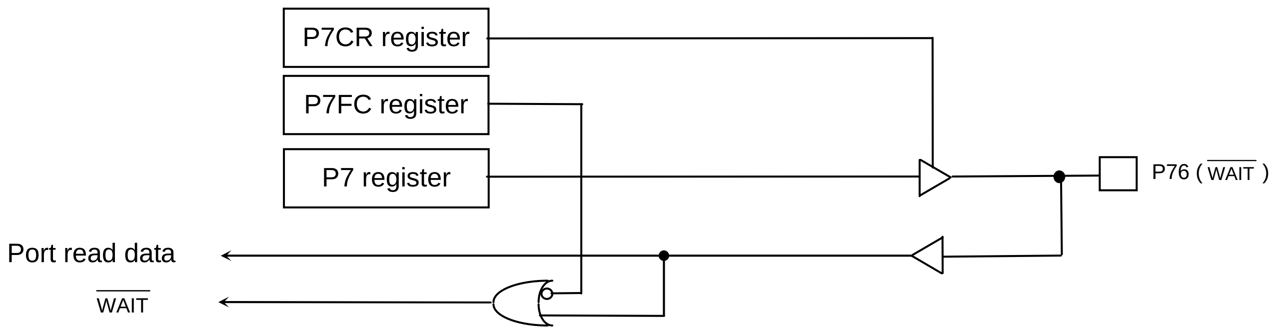


Figure 3.8.10 Port7

Port 7 register

P7 (001CH)		7	6	5	4	3	2	1	0
	bit Symbol		P76	P75	P74	P73	P72	P71	P70
	Read/Write		R/W						
	System Reset State		Data from external port (Output latch register is set to "1")		Data from external port (Output latch register is cleared to "0")		Data from external port (Output latch register is set to "1")		1
	Hot Reset State		–		–		–		–

Port 7 Control register

P7CR (001EH)		7	6	5	4	3	2	1	0
	bit Symbol		P76C	P75C	P74C	P73C	P72C	P71C	
	Read/Write		W						
	System Reset State		0	0	0	0	0	0	
	Hot Reset State		–	–	–	–	–	–	
	Function		0: Input 1: Output						

Port 7 Function register

P7FC (001FH)		7	6	5	4	3	2	1	0
	bit Symbol		P76F	P75F	P74F	P73F	P72F	P71F	P70F
	Read/Write		W						
	System Reset State		0	0	0	0	0	0	0/1 (Note3)
	Hot Reset State		–	–	–	–	–	–	–
	Function		0:Port 1: $\overline{\text{WAIT}}$	Refer to following table			0:Port 1: $\overline{\text{NDWE}}$ at <P72>=0 $\overline{\text{WRLU}}$ at <P72>=1	0:Port 1: $\overline{\text{NDRE}}$ at <P71>=0 $\overline{\text{WRLL}}$ at <P71>=1	0:Port 1: $\overline{\text{RD}}$

Port 7 Drive register

P7DR (0087H)		7	6	5	4	3	2	1	0
	bit Symbol		P76D	P75D	P74D	P73D	P72D	P71D	P70D
	Read/Write		R/W						
	System Reset State		1	1	1	1	1	1	1
	Hot Reset State		–	–	–	–	–	–	–
	Function		Input/Output buffer drive register for standby mode						

P73 setting

<P73F> \ <P73C>	0	1
0	Input Port	Output Port
1	Reserved	EA24Output

P72 setting

<P72F> \ <P72C>	0	1
0	Input Port	Output Port
1	Reserved	$\overline{\text{NDWE}}$ Output (at <P72>=0) $\overline{\text{WRLU}}$ Output (at <P72>=1)

P71 setting

<P71F> \ <P71C>	0	1
0	Input Port	Output Port
1	Reserved	$\overline{\text{NDRE}}$ Output (at <P71>=0) $\overline{\text{WRLL}}$ Output (at <P71>=1)

P76 setting

<P76F> \ <P76C>	0	1
0	Input Port	Output Port
1	$\overline{\text{WAIT}}$ Input	Reserved

P75 setting

<P75F> \ <P75C>	0	1
0	Input Port	Output Port
1	NDR/$\overline{\text{B}}$ Input	R/W Output

P74 setting

<P74F> \ <P74C>	0	1
0	Input Port	Output Port
1	Reserved	EA25Output

Note1: A read-modify-write operation cannot be performed for P7CR, P7FC.

Note2: When $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ are used, set registers in the following order to avoid outputting a negative glitch.

Order	Registser	bit2	bit1
(1)	P7	0	0
(2)	P7FC	1	1
(3)	P7CR	1	1

Note3: It is set to "Port" or $\overline{\text{RD}}$ pin by AM pins state.

Figure 3.8.11 Register for Port 7

3.8.6 Port 8 (P80 to P83, P86, P87)

Port 8 is 6-bit output ports. Resetting sets the output latch of P82 to “0” and the output latches of P80 to P81, P83, P86 and P87 to “1”. But if it is started at boot mode (AM [1:0]= “11”), output latch of P82 is set to “1”.

Port 8 can also be set to function as an interface-pin for external memory using function register P8FC.

Writing “1” in the corresponding bit of P8FC and P8FC2 enables the respective functions.

Resetting P8FC to “0” and P8FC2 to “0”, sets all bits to output ports.

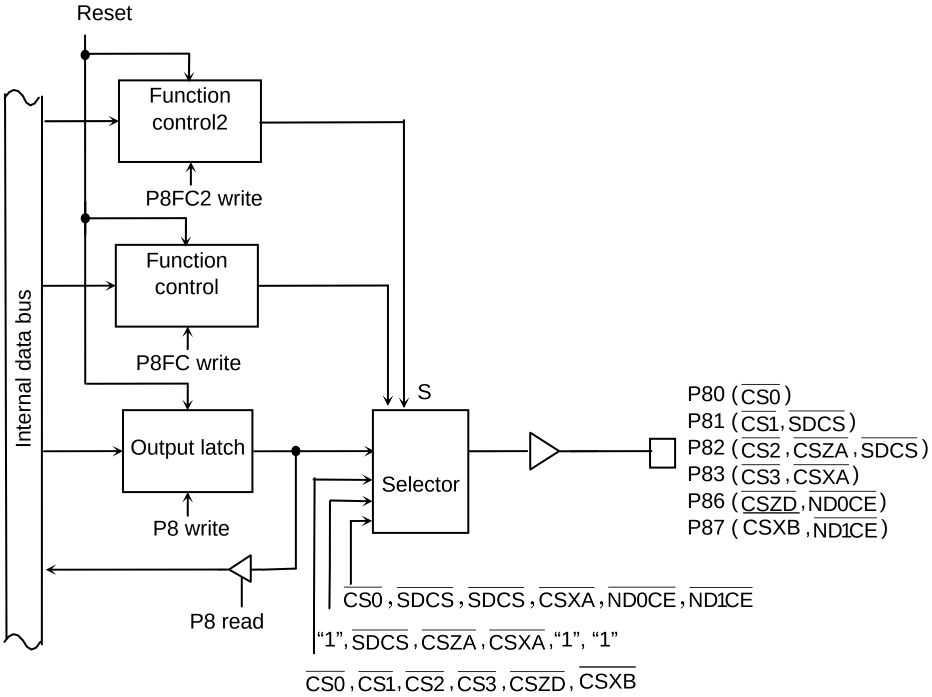


Figure 3.8.12 Port 8

Port 8 register

		7	6	5	4	3	2	1	0
P8 (0020H)	bit Symbol	P87	P86			P83	P82	P81	P80
	Read/Write	R/W				R/W			
	System Reset State	1	1			1	0 (Note3)	1	1
	Hot Reset State	–	–			–	–	–	–

Port 8 Function register

		7	6	5	4	3	2	1	0
P8FC (0023H)	bit Symbol	P87F	P86F			P83F	P82F	P81F	P80F
	Read/Write	W				W			
	System Reset State	0	0			0	0	0	0
	Hot Reset State	–	–			–	–	–	–
	Function	0: Port 1: <P87F2>	0: Port 1: <P86F2>			Refer to following table		0: Port 1: CS1	0: Port 1: CS0

Port 8 Function registers 2

		7	6	5	4	3	2	1	0
P8FC2 (0021H)	bit Symbol	P87F2	P86F2			P83F2	P82F2	P81F2	
	Read/Write	W				W			
	System Reset State	0	0			0	0	0	
	Hot Reset State	–	–			–	–	–	
	Function	0: CSXB 1: ND1CE	0: CSZD 1: ND0CE			Refer to following table		0: <P81F> 1: SDCS	

Port 8 Drive register

		7	6	5	4	3	2	1	0
P8DR (0088H)	bit Symbol	P87D	P86D			P83D	P82D	P81D	P80D
	Read/Write	R/W				R/W			
	System Reset State	1	1			1	1	1	1
	Hot Reset State	–	–			–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

P86 setting

<P86F2> \ <P86F>	0	1
0	Output port	CSZD Output
1	Don't setting	ND0CE Output

P83 setting

<P83F2> \ <P83F>	0	1
0	Output port	CS3 Output
1	CSXA Output	

P82 setting

<P82F2> \ <P82F>	0	1
0	Output port	CS2 Output
1	CSZA Output	SDCS Output

P87 setting

<P87F2> \ <P87F>	0	1
0	Output port	CSXB Output
1	Reserved	ND1CE Output

Note1: A read-modify-write operation cannot be performed for P8FC and P8FC2.

Note2: Do not write "1" to P8<P82> register before setting P82-pin to CS2 or CSZA because, on reset, P82-pin outputs "0" as CE for program memory.

Note3: If it is started at boot mode (AM [1:0] = "11"), output latch of P82 is set to "1".

Note4: When ND0CE and ND1CE are used, set registers by following order.

Order	Regigtser	bit2	bit1
(1)	P8	1	1
(2)	P8FC2	1	1
(3)	P8FC	1	1

Figure 3.8.13 Register for Port 8

3.8.7 Port 9 (P90 to P92, P96, P97)

P90 to P92 are 3-bit general-purpose I/O port. I/O can be set on a bit basis using the control register. Each bit can be set individually for input or output. Resetting sets P90 to P92 to input port and all bits of output latch to"1".

P96 to P97 are 2-bit general-purpose input port.

Writing "1" the corresponding bits of P9FC enables the respective functions.

Resetting resets the P9FC to "0", and sets all bits to input ports.

(1) Port 90 (TXD0), Port 91 (RXD0), Port 92 (SCLK0, $\overline{CTS0}$)

Ports 90 to 92 are general-purpose I/O port. They are also function as either SIO0 or SIO1. SIO0 and SIO1 functions are also used as PP3 to PP5 pins. When selecting SIO0 function (using Port 9 or Port P), set P9FC2<P93F2, P94F2, P95F2>. And when selection SIO1 function (using Port 9 or Port P), set PPFC2<PP4F2, PP5F2, PP6F2>.

	SIO mode (SIO0 module)	UART, IrDA mode (SIO0 module)
P90	TXD0, TXD1 (Data output)	TXD0 (Data output)
P91	RXD0, RXD1 (Data input)	RXD0 (Data input)
P92	SCLK0, SCLK1 (Clock input or output)	$\overline{CTS0}$, $\overline{CTS1}$ (Clear to send)

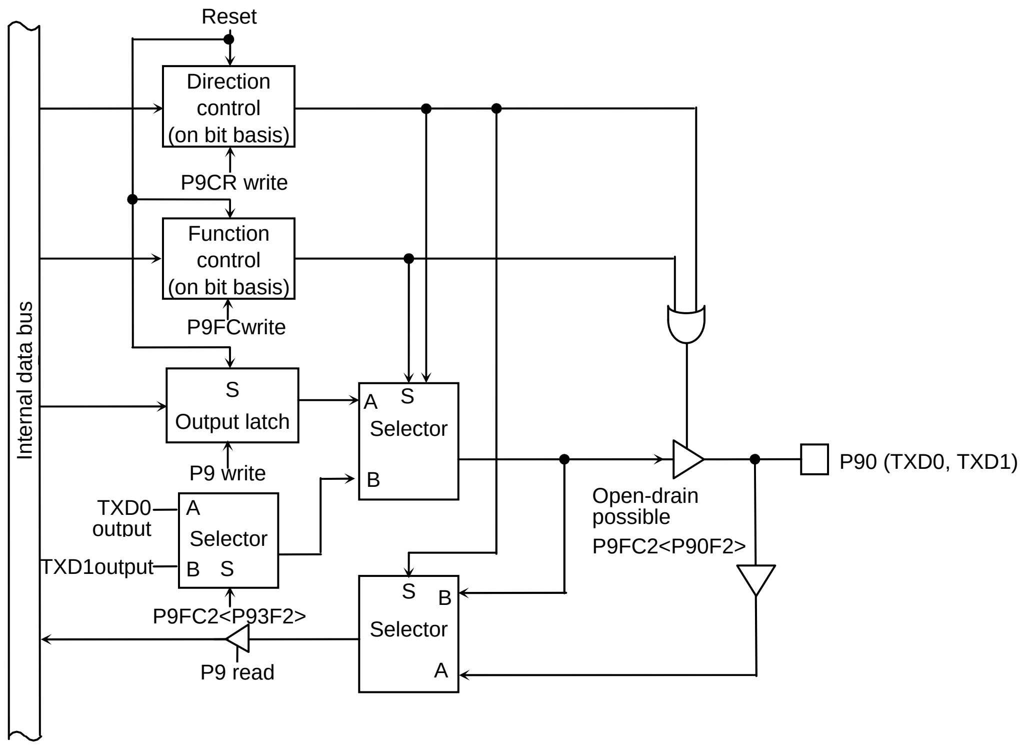


Figure 3.8.14 P90

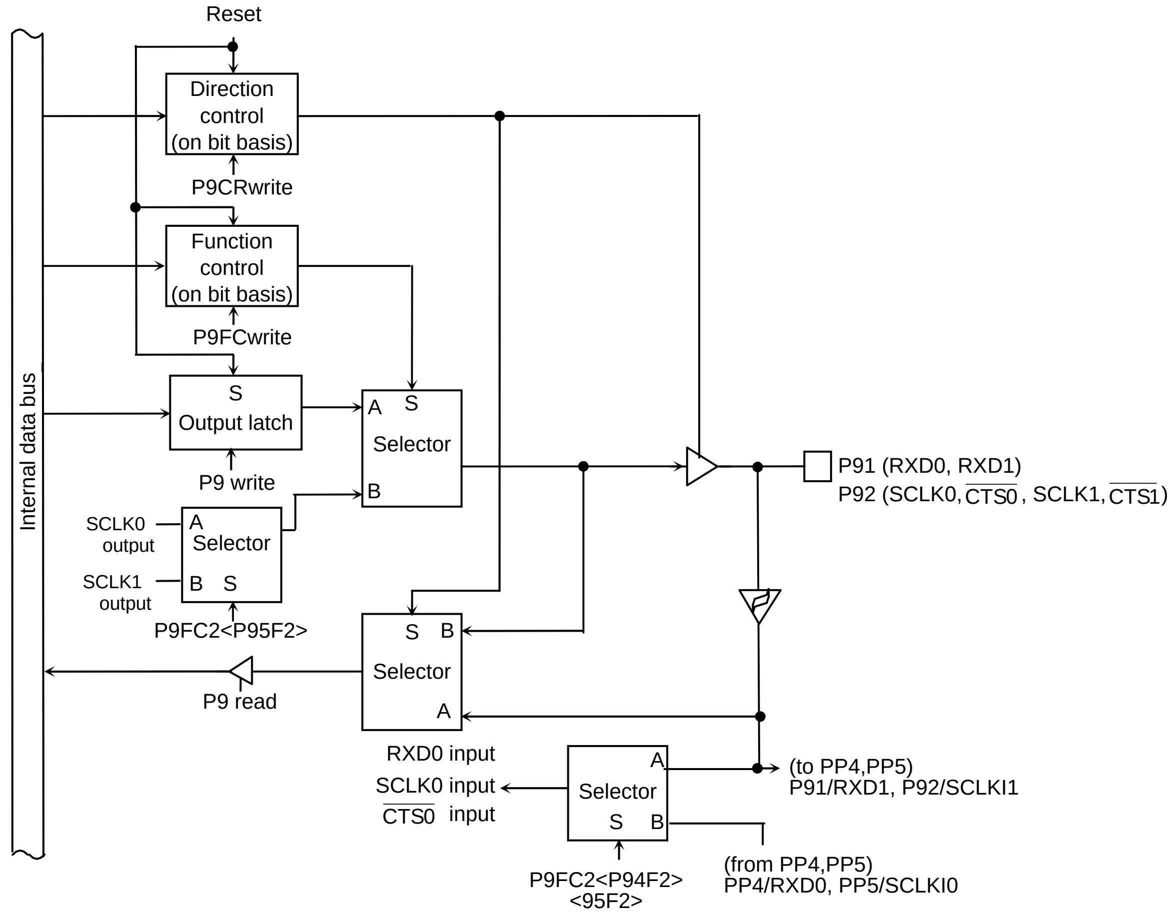


Figure 3.8.15 P91, 92

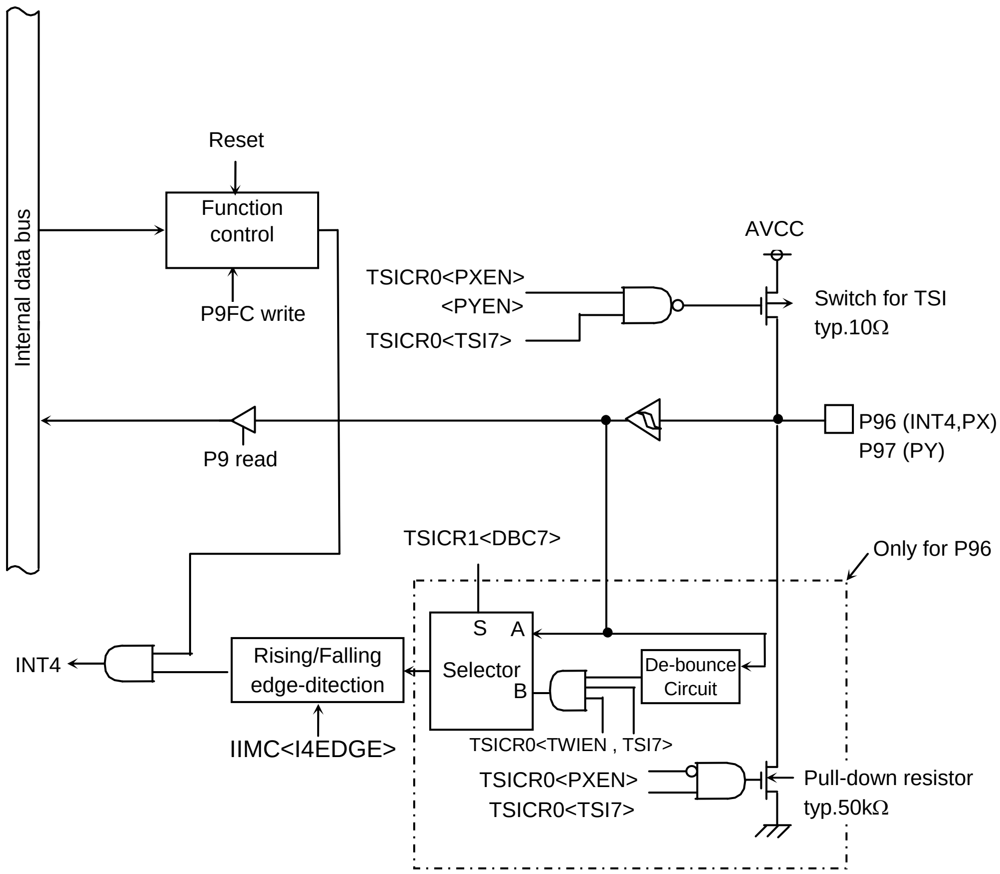


Figure 3.8.16 Port 96,97

Port 9 register

		7	6	5	4	3	2	1	0
P9 (0024H)	bit Symbol	P97	P96				P92	P91	P90
	Read/Write	R					R/W		
	System Reset State	Data from external port					Data from external port (Output latch register is set to "1")		
	Hot Reset State	–					–		

Port 9 control register

		7	6	5	4	3	2	1	0
P9CR (0026H)	bit Symbol						P92C	P91C	P90C
	Read/Write						W		
	System Reset State						0	0	0
	Hot Reset State						–	–	–
	Function						Refer to following table		

Port 9 function register

		7	6	5	4	3	2	1	0
P9FC (0027H)	bit Symbol		P96F				P92F		P90F
	Read/Write		W				W		W
	System Reset State		0				0		0
	Hot Reset State		–				–		–
	Function		0: Input port 1: INT4				Refer to following table		Refer to following table

Port 9 Function registers 2

		7	6	5	4	3	2	1	0
P9FC2 (0025H)	bit Symbol	–		P95F2	P94F2	P93F2	–		P90F2
	Read/Write	W		W	W	W	W		W
	System Reset State	0		0	0	0	0		0
	Hot Reset State	–		–	–	–	–		–
	Function	Always write "0"		P92 SCLK selection 0: SCLK0 1: SLCK1 SIO0 SCLK, $\overline{CTS}$ input selection 0: P92 1: PP5	SIO0 RXD selection 0: P91 1: PP4	P90 TXD selection 0: TXD0 1: TXD1	Always write "0"		0:CMOS 1: Open-drain

Port 9 drive register

P9DR (0089H)

	7	6	5	4	3	2	1	0
bit Symbol	P97D	P96D				P92D	P91D	P90D
Read/Write	R/W					R/W		
System Reset State	1	1				1	1	1
Hot Reset State	–	–				–	–	–
Function	Input/Output buffer drive register for standby mode							

P92 setting

<P92F> \ <P92C>	0	1
0	Input port, $\overline{CTS0}$, $\overline{CTS1}$ /SCLK0,SCLK1 Input	Output port
1	Don’t setting	SCLK0,SCLK1 Output

P91 setting

<P91C>	
0	1
Input port/ RXD0,RXD1 Input	Output port

P90 setting

<P90F> \ <P90C>	0	1
0	Input port	Output port
1	Don’t setting	TXD0,TXD1 Output

Note 1: A read-modify-write operation cannot be performed for P9CR, P9FC and P9FC2.

Note 2: When setting P96 pin to INT4 input, set P9DR<P96D> to “0” (prohibit input), and when driving P96 pin to “0”, execute HALT instruction. This setting generates INT4 inside. If don’t using external interrupt in HALT condition, set like an interrupt don’t generated. (e.g. change port setting)

Figure 3.8.17 Register for Port 9

3.8.8 Port A (PA0 to PA7)

Ports A0 to A7 are 8-bit general-purpose input ports with pull-up resistor. In addition to functioning as general-purpose I/O ports, ports A0 to A7 can also, as a Keyboard interface, operate a Key-on wake-up function. The various functions can each be enabled by writing a "1" to the corresponding bit of the Port A Function Register (PAFC).
Resetting resets all bits of the register PAFC to "0" and sets all pins to be input port.

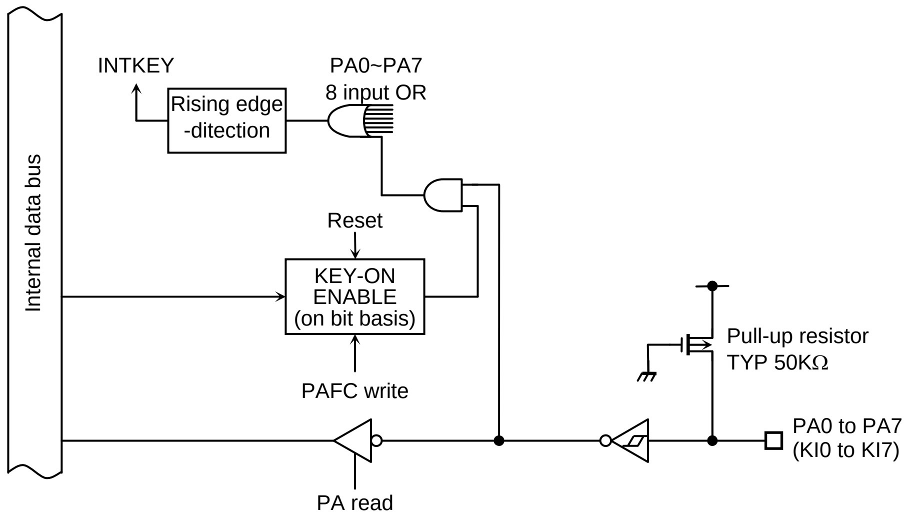


Figure 3.8.18 Port A

When PAFC = "1", if the input of any of KI0-KI7 pins falls down, an INTKEY interrupt is generated. An INTKEY interrupt can be used to release all HALT modes.

Port A register

PA (0028H)

	7	6	5	4	3	2	1	0
bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R							
System Reset State	Data from external port							
Hot Reset State	–							

Port A Function register

PAFC (002BH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	0: KEY IN disable 1: KEY IN enable							

Port A Drive register

PADR (008AH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
Read/Write	R/W							
System Reset State	1	1	1	1	1	1	1	1
Hot Reset State	–	–	–	–	–	–	–	–
Function	Input/Output buffer drive register for standby mode							

Note: A read-modify-write operation cannot be performed for PAFC.

Figure 3.8.19 Register for Port A

3.8.9 Port C (PC0 to PC7)

PC0 to PC7 are 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port C to an input port. It also sets all bits of the output latch register to “1”.

In addition to functioning as a general-purpose I/O port, Port C can also function as an input pin for timers (TA0IN, TA2IN), input pin for external interruption (INT0 to INT3), Extension address function (EA26, EA27, EA28) , output pin for SPI controller (SPDI, SPDO and SPCLK) and output pin for Key (KO8). These settings are mode using the function register PCFC. The edge select for external interruption is determined by the IIMC register in the interruption controller.

(1) PC0 (INT0), PC2 (INT2)

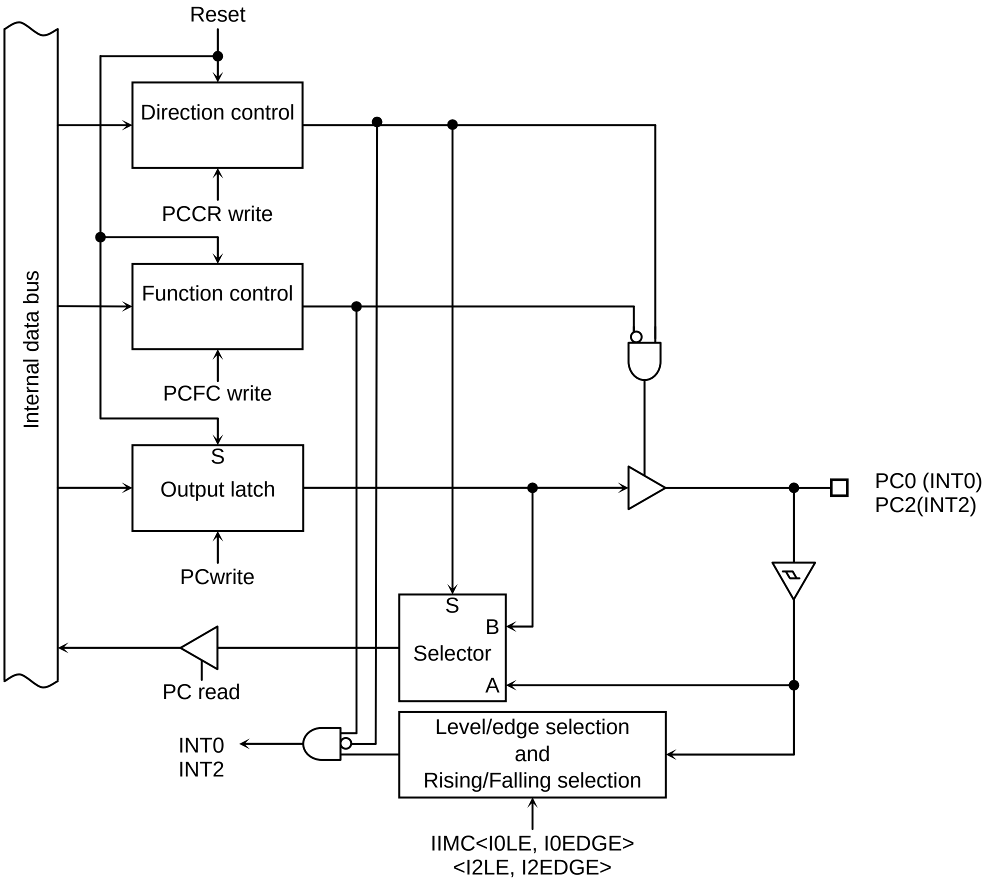


Figure 3.8.20 Port C0, C2

(2) PC1 (INT1, TA0IN), PC3 (INT3, TA2IN)

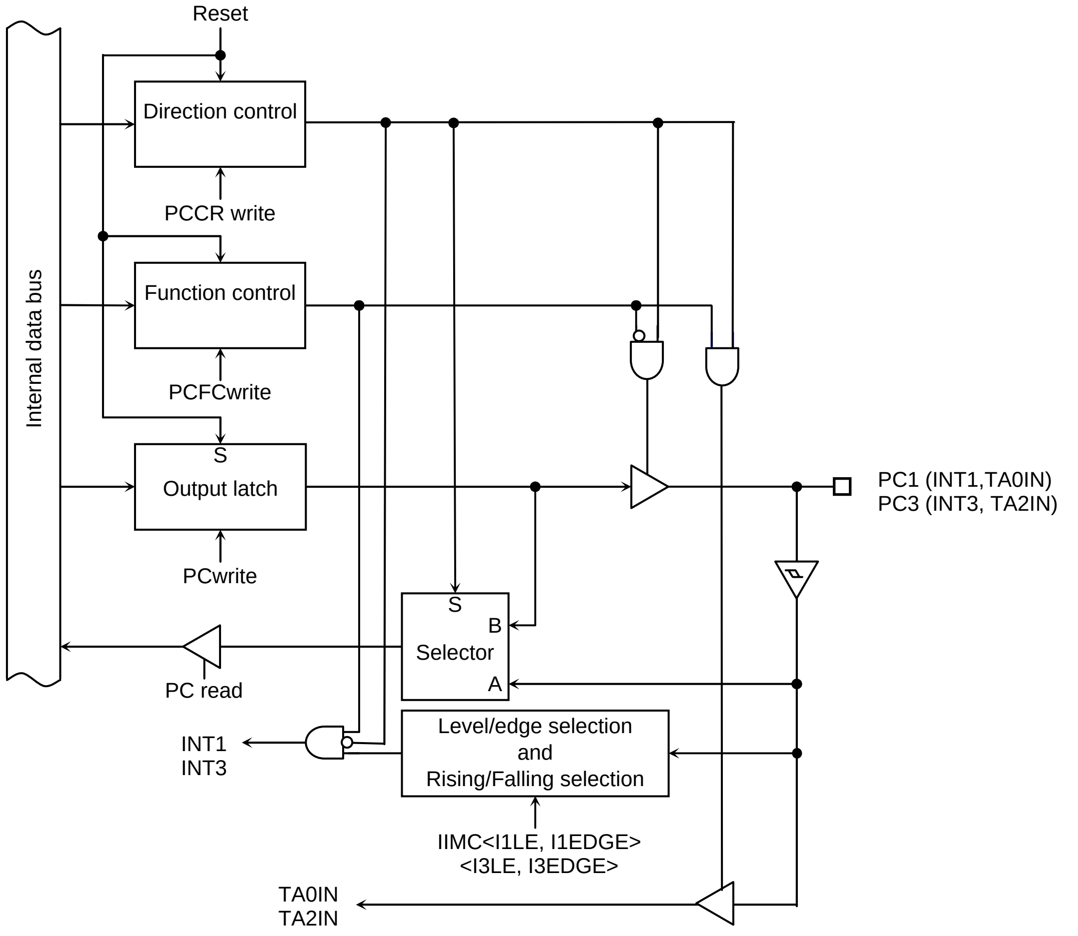


Figure 3.8.21 Port C1,C3

(3) PC4 (EA26, SPDI)

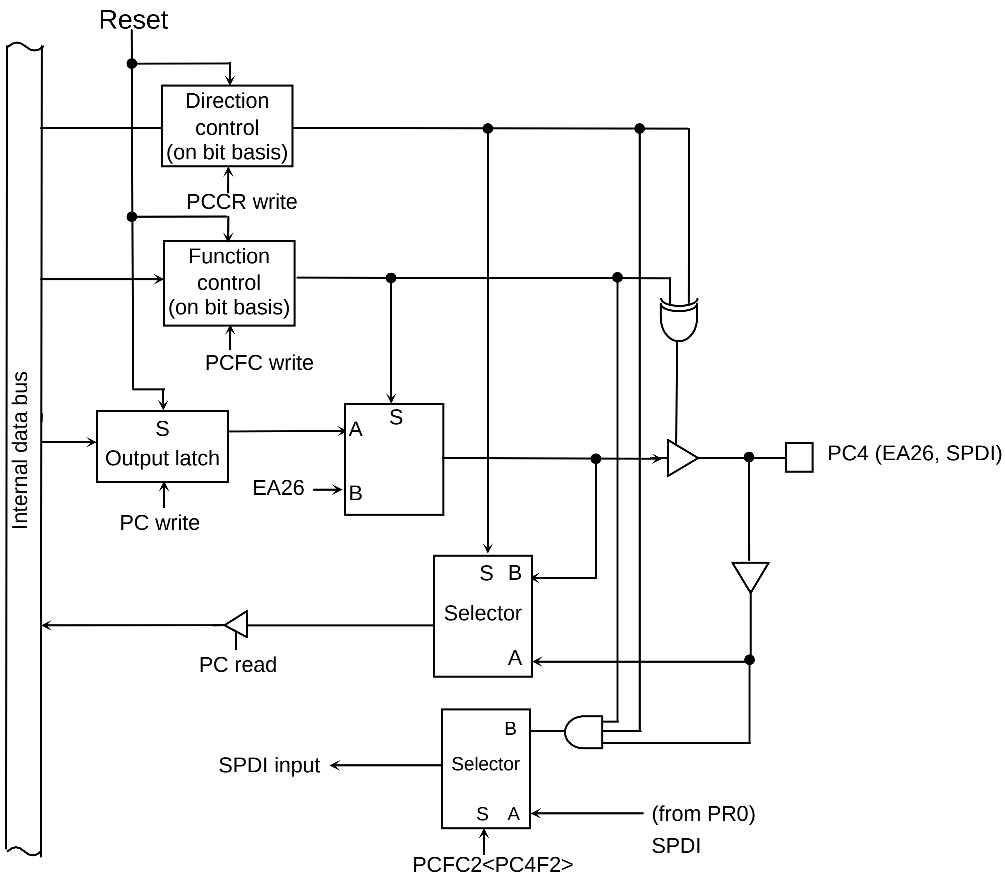


Figure 3.8.22 Port C4

(4) PC5 (EA27), PC6 (EA28)

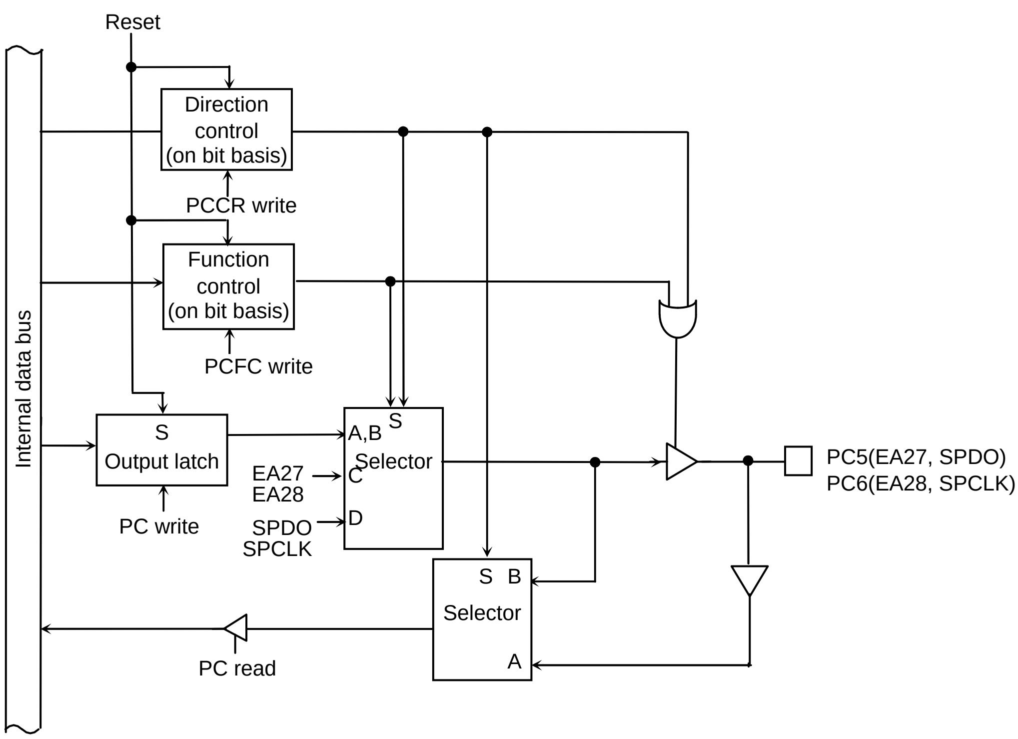


Figure 3.8.23 Port C5, C6

(5) PC7 (KO8)

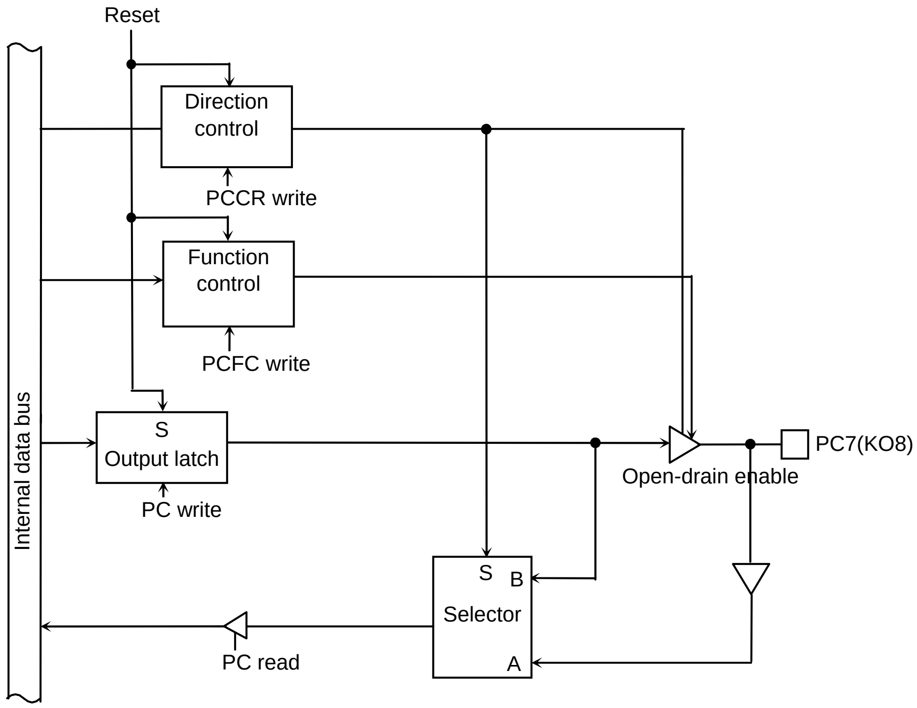


Figure 3.8.24 Port C7

Port C register

PC (0030H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Read/Write	R/W							
System Reset State	Data from external port (Output latch register is set to “1”)							
Hot Reset State	–							

Port C control register

PCCR (0032H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	0: Input 1: Output							

Port C function register

PCFC (0033H)

	7	6	5	4	3	2	1	0
bit Symbol	PC7F	PC6F	PC5F	PC4F	PC3F	PC2F	PC1F	PC0F
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	Refer to following table							

Port C function register 2

PCFC2 (0031H)

	7	6	5	4	3	2	1	0
bit Symbol				PC4F2				
Read/Write				W				
System Reset State				0				
Hot Reset State				–				
Function				SPDI pin selection 0: PR0 1: PC4				

Port C drive register

PCDR (008CH)		7	6	5	4	3	2	1	0
	bit Symbol	PC7D	PC6D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

PC2 setting

<PC2F> \ <PC2C>	0	1
0	Input port	Output port
1	INT2	Don't setting

PC1 setting

<PC1F> \ <PC1C>	0	1
0	Input port	Output port
1	INT1	TA0IN input

PC0 setting

<PC0F> \ <PC0C>	0	1
0	Input port	Output port
1	INT0	Don't setting

PC5 setting

<PC5F> \ <PC5C>	0	1
0	Input port	Output port
1	EA27 output	SPDO output

PC4 setting

<PC4F> \ <PC4C>	0	1
0	Input port	Output port
1	EA26 output	SPDI input

PC3 setting

<PC3F> \ <PC3C>	0	1
0	Input port	Output port
1	INT3	TA2IN input

PC7 setting

<PC7F> \ <PC7C>	0	1
0	Input port	Output port
1	Don't setting	KO8output (Open-drain)

PC6 setting

<PC6F> \ <PC6C>	0	1
0	Input port	Output port
1	EA28 output	SPCLK output

Note 1: A read-modify-write operation cannot be performed for the registers PCCR, PCFC.

Note 2: When setting PC3-PC0 pins to INT3-INT0 input, set PCDR<PC3D: PC0D> to "0000"(prohibit input), and when driving PC3-PC0 pins to "0", execute HALT instruction. This setting generates INT3-INT0 inside. If don't use external interrupt in HALT condition, set like an interrupt don't generated. (e.g. change port setting)

Figure 3.8.25 Register for Port C

3.8.10 Port F (PF0 to PF2, PF7)

Ports F0 to F2 are 3-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets PF0 to PF2 to be input ports. It also sets all bits of the output latch register to "1". In addition to functioning as general-purpose I/O port pins, PF0 to PF2 can also function as the output for I^2S0. A pin can be enabled for I/O by writing a "1" to the corresponding bit of the Port F Function Register (PFFC).

Port F7 is a 1-bit general-purpose output port. In addition to functioning as general-purpose output port, PF7 can also function as the SDCLK output. Resetting sets PF7 to be an SDCLK output port.

(1) Port F0 (I2S0CKO), Port F1 (I2S0DO), Port F2 (I2S0WS)

Ports F0 to F2 are general-purpose I/O port. They also function as either I^2S. Each pin is detailed below.

	I^2Smode (I2S0Module)
PF0	I2S0CKO (Clock output)
PF1	I2S0DO (Data output)
PF2	I2S0WS (Word-select output)

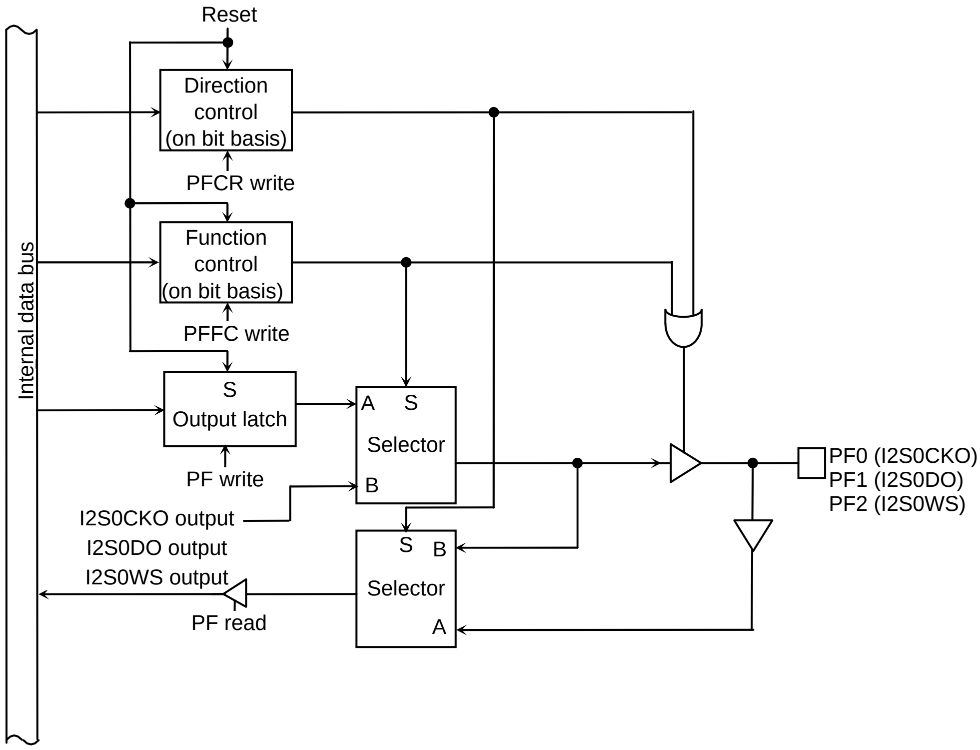


Figure 3.8.26 Port F0, F1, F2

(2) Port F7 (SDCLK)

Port F7 is general-purpose output port. In addition to functioning as general-purpose output port, PF7 can also function as the SDCLK output.

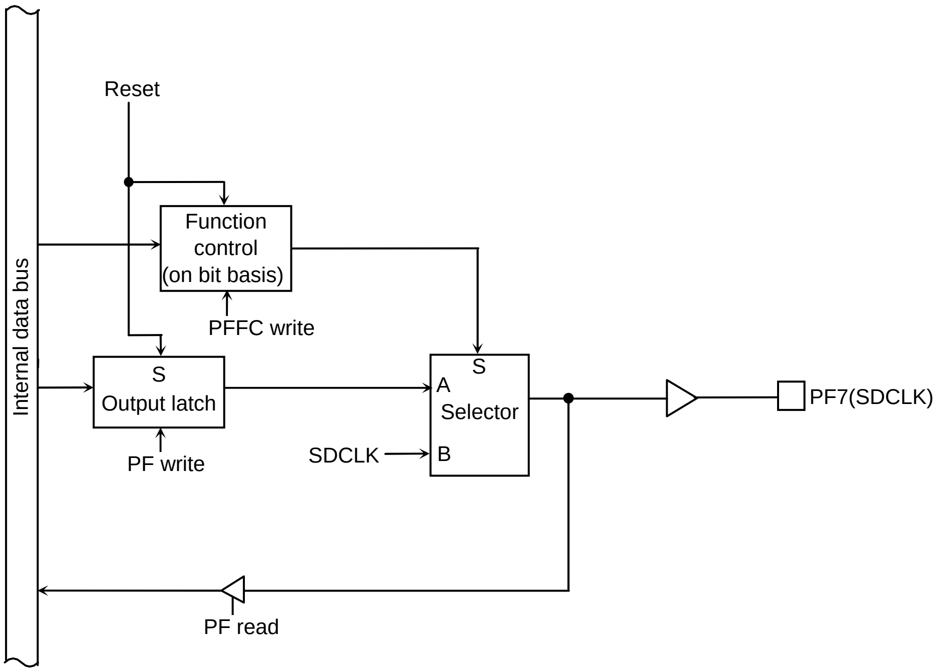


Figure 3.8.27 Port F7

Port F register

		7	6	5	4	3	2	1	0
PF (003CH)	bit Symbol	PF7					PF2	PF1	PF0
	Read/Write	R/W							
	System Reset State	1					Data from external port (Output latch register is set to "1")		
	Hot Reset State	–							

Port F control register

		7	6	5	4	3	2	1	0
PFCR (003EH)	bit Symbol						PF2C	PF1C	PF0C
	Read/Write						W		
	System Reset State						0	0	0
	Hot Reset State						–	–	–
	Function						Refer to following table		

Port F function register

		7	6	5	4	3	2	1	0
PFFC (003FH)	bit Symbol	PF7F					PF2F	PF1F	PF0F
	Read/Write	W					W		
	System Reset State	1					0	0	0
	Hot Reset State	–					–	–	–
	Function	0: Port 1: SDCLK					Refer to following table		

Port F drive register

		7	6	5	4	3	2	1	0
PFDR (008FH)	bit Symbol	PF7D					PF2D	PF1D	PF0D
	Read/Write	R/W					R/W		
	System Reset State	1					1	1	1
	Hot Reset State	–					–	–	–
	Function	Input/Output buffer drive register for standby mode							

PF2 setting

<PF2F> \ <PF2C>	0	1
0	Input port	Output port
1	I2S0WS output	

PF1 setting

<PF1F> \ <PF1C>	0	1
0	Input port	Output port
1	I2S0DO output	

PF0 setting

<PF0F> \ <PF0C>	0	1
0	Input port	Output port
1	I2S0CKOoutput	

Note: A read-modify-write operation cannot be performed for the registers PFCR, PFFC.

Figure 3.8.28 Register for Port F

3.8.11 Port G (PG0 to PG5)

PG0 to PG5 are 6-bit input ports and can also be used as the analog input pins for the internal AD converter. PG3 can also be used as the ADTRG pin for the AD converter.

PG2 and PG3 can also be used as the MX and MY pins for a Touch screen interface.

(PG) register is prohibited to access by byte. All the instruction (Arithmetic/Logical/

Bit operation and rotate/shift instruction) accesses by byte are prohibited. Word access is always needed.

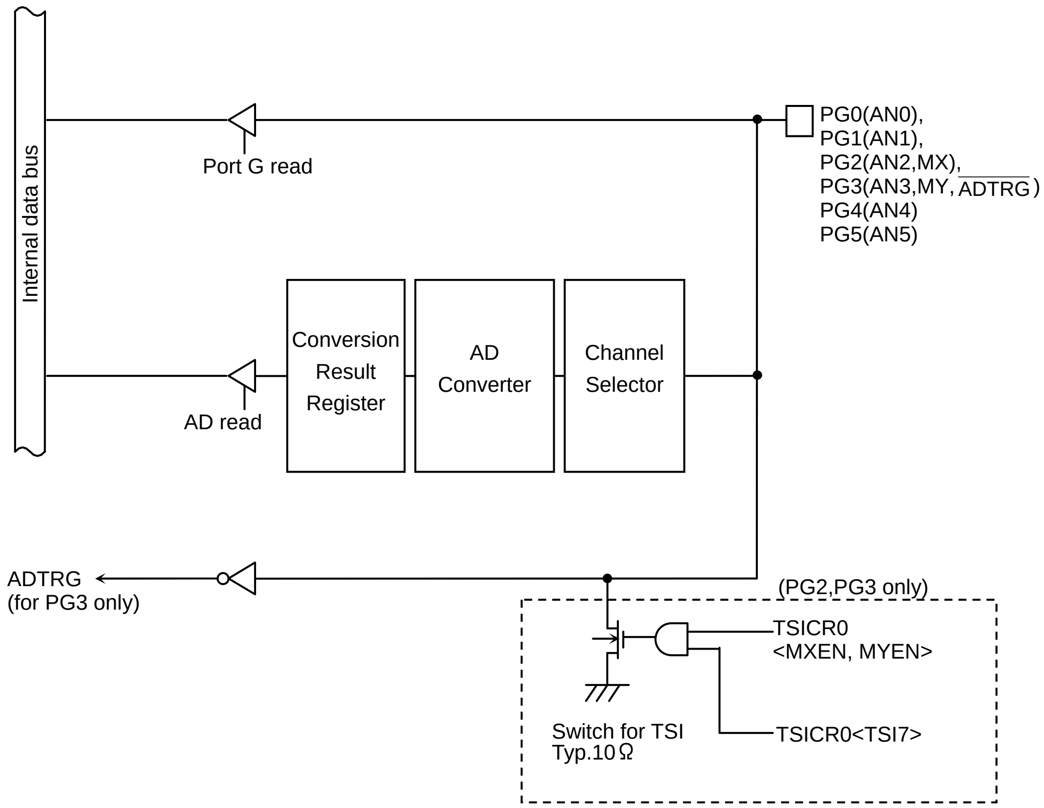


Figure 3.8.29 Port G

Port G register

		7	6	5	4	3	2	1	0
PG (0040H)	Bit Symbol			PG5	PG4	PG3	PG2	PG1	PG0
	Read/Write			R					
	System Reset State			Data from external port					
	Hot Reset State			–					

Note: The input channel selection of the AD converter and the permission of for ADTRG input are set by AD converter mode register ADMOD1.

Port G Function register

		7	6	5	4	3	2	1	0
PGFC (0043H)	Bit Symbol					PG3F			
	Read/Write					W			
	System Reset State					0			
	Hot Reset State					–			
	Function					0: Input port or AN3 1: $\overline{\text{ADTRG}}$			

Port G driver register

		7	6	5	4	3	2	1	0
PGDR (0090H)	Bit Symbol					PG3D	PG2D		
	Read/Write					R/W			
	System Reset State					1	1		
	Hot Reset State					–	–		
	Function					Input/Output buffer drive register for standby mode			

Note : A read-modify-write operation cannot be performed for the registers PGFC.

Figure 3.8.30 Register for Port G

3.8.12 Port J (PJ0 to PJ7)

PJ0 to PJ4 and PJ7 are 6-bit output port. Resetting sets the output latch PJ to "1", and they output "1". PJ5 to PJ6 are 2-bit input/output port. In addition to functioning as a port, Port J also functions as output pins for SDRAM ($\overline{SDRAS}$, $\overline{SDCAS}$, $\overline{SDWE}$, SDLLDQM, SDLUDQM, and SDCKE), SRAM ($\overline{SRWR}$, $\overline{SRLLB}$ and $\overline{SRLUB}$) and NAND-Flash(NDALE and NDCLE).

The above settings are made using the function register PJFC.

However, either SDRAM or SRAM output signal for PJ0 to PJ2 are selected automatically according to the setting of the memory controller.

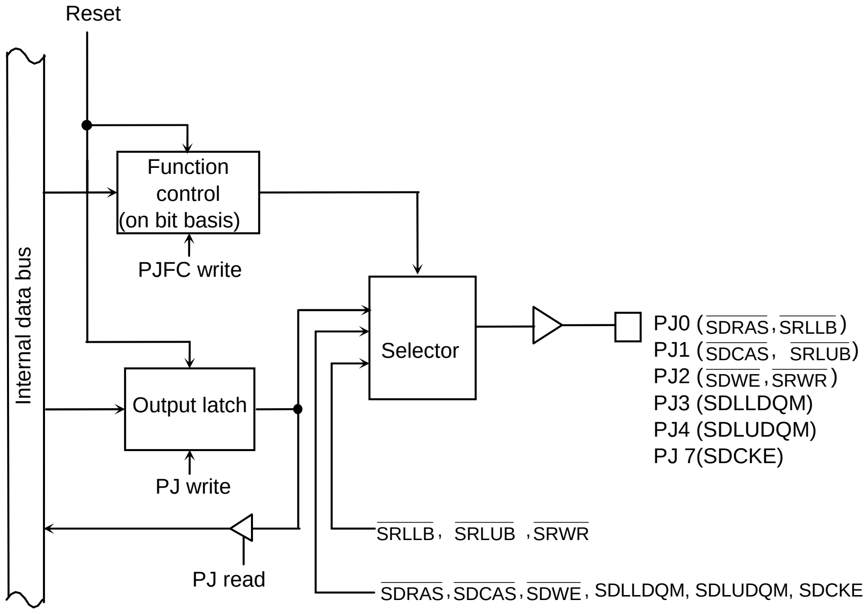


Figure 3.8.31 Port J0 to J4 and J7

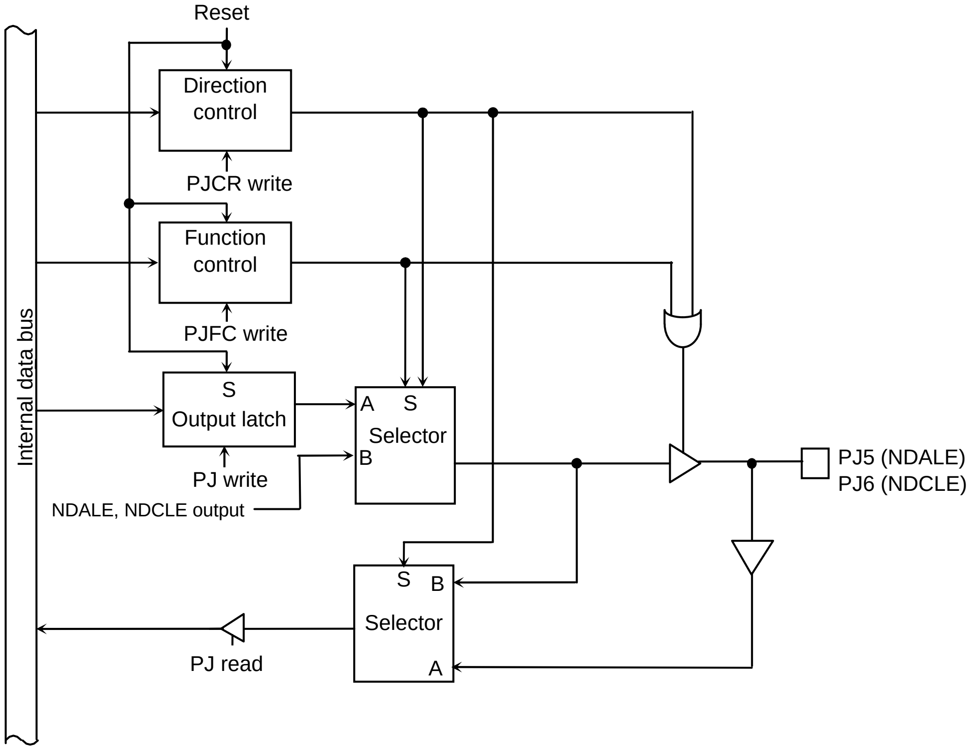


Figure 3.8.32 Port J5,J6

Port J register

		7	6	5	4	3	2	1	0
PJ (004CH)	bit Symbol	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
	Read/Write	R/W							
	System Reset State	1	Data from external port (Output latch register is set to "1")		1	1	1	1	1
	Hot Reset State	–	–		–	–	–	–	–

Port J control register

		7	6	5	4	3	2	1	0
PJCR (004EH)	bit Symbol		PJ6C	PJ5C					
	Read/Write		W						
	System Reset State		0	0					
	Hot Reset State		–	–					
	Function		0: Input, 1: Output						

Port J function register

		7	6	5	4	3	2	1	0
PJFC (004FH)	bit Symbol	PJ7F	PJ6F	PJ5F	PJ4F	PJ3F	PJ2F	PJ1F	PJ0F
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: Port 1: SDCKE	0: Port 1: NDCLE	0: Port 1: NDALE	0: Port 1:SDLUDQM	0: Port 1:SDLLDQM	0: Port 1: $\overline{SDWE}$, $\overline{SRWR}$	0: Port 1: $\overline{SDCAS}$, $\overline{SRLUB}$	0: Port 1: $\overline{SDRAS}$, $\overline{SRLLB}$

Port J drive register

		7	6	5	4	3	2	1	0
PJDR (0093H)	bit Symbol	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note: A read-modify-write operation cannot be performed for the registers PJCR and PJFC.

Figure 3.8.33 Register for Port J

3.8.13 Port K (PK0 to PK7)

PK0 to PK7 are 8-bit output ports. Resetting sets the output latch PK to "0", and PK0 to PK7 pins output "0".

In addition to functioning as an output port function, port K also functions as output pins for an LCD controller (LCP0, LHSYNC, LLOAD, LFR, LVSYNC, and LGOE0 to LGOE2).

The above settings are made using the function register PKFC.

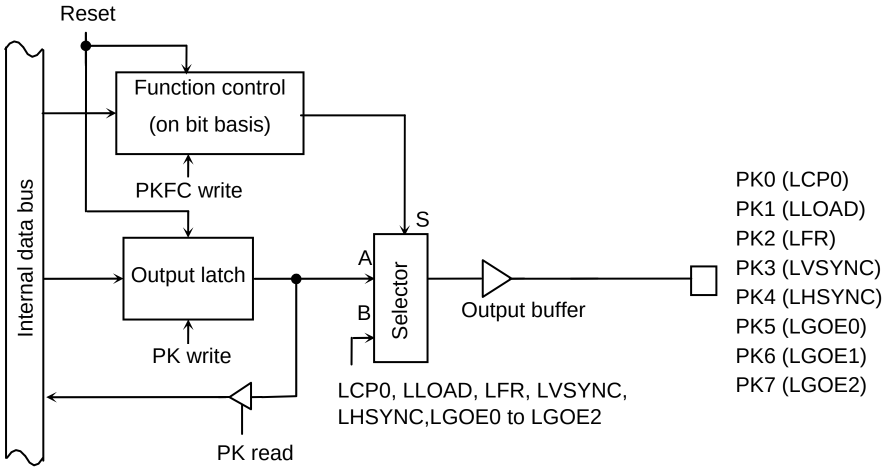


Figure 3.8.34 Port K0 to K7

Port K register

PK (0050H)		7	6	5	4	3	2	1	0
	bit Symbol	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
	Read/Write	R/W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–

Port K function register

PKFC (0053H)		7	6	5	4	3	2	1	0
	bit Symbol	PK7F	PK6F	PK5F	PK4F	PK3F	PK2F	PK1F	PK0F
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0:Port 1:LGOE2	0:Port 1:LGOE1	0:Port 1:LGOE0	0:Port 1: LHSYNC	0: Port 1: LVSYNC	0: Port 1: LFR	0: Port 1: LLOAD	0: Port 1: LCP0

Port K drive register

PKDR (0094H)		7	6	5	4	3	2	1	0
	bit Symbol	PK7D	PK6D	PK5D	PK4D	PK3D	PK2D	PK1D	PK0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note: A read-modify-write operation cannot be performed for the registers PKFC.

Figure 3.8.35 Register for Port K

3.8.14 Port L (PL0 to PL7)

PL0 to PL7 are 8-bit output ports. Resetting sets the output latch PL to "0", and PL0 to PL7 pins output "0". In addition to functioning as a general-purpose output port, port L can also function as a data bus for an LCD controller (LD0 to LD7). The above settings are made using the function register PLFC.

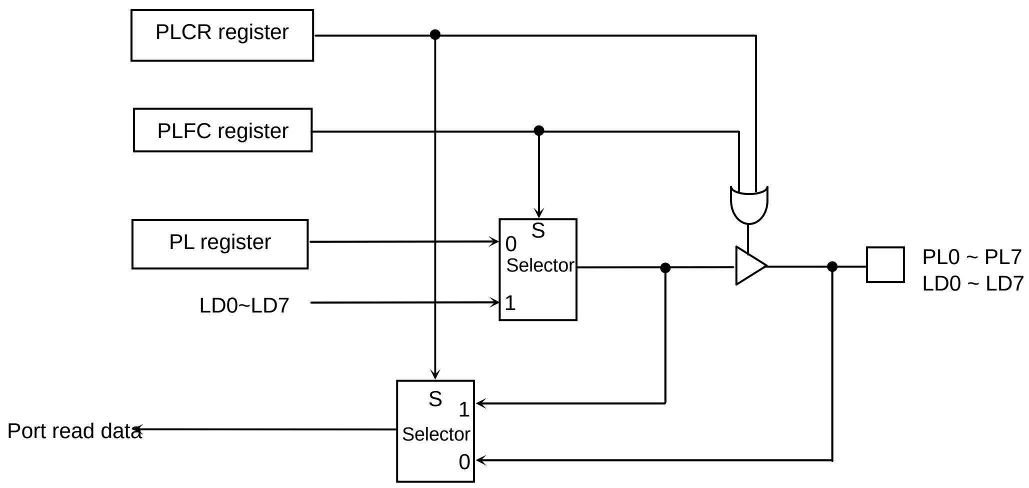


Figure 3.8.36 Port L0 to L7

Port L register

		7	6	5	4	3	2	1	0
PL (0054H)	bit Symbol	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
	Read/Write	R/W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–

Port L control register

		7	6	5	4	3	2	1	0
PLCR (0056H)	bit Symbol	PL7C	PL6C	PL5C	PL4C	PL3C	PL2C	PL1C	PL0C
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: Input 1: Output							

Port L function register

		7	6	5	4	3	2	1	0
PLFC (0057H)	bit Symbol	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: Port 1: Data bus for LCDC (LD7 toLD0)							

Port L drive register

		7	6	5	4	3	2	1	0
PLDR (0095H)	bit Symbol	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for the registers PLCR, PLFC.

Note2: When PL is used as LD7 to LD0, set applicable PLnC to"1".

<PLnF> \ <PLnC>	0	1
0	Input port	Output port
1	LDn	

Figure 3.8.37 Register for Port L

3.8.15 Port M (PM1, PM2, PM7)

PM1, PM2 and PM7 are 3-bit output ports. Resetting sets the output latch PM to “1”, and PM1, PM2 and PM7 pins output “1”.

In addition to functioning as an output ports, port M also functions as output pin for the timers (TA1OUT), output pins for the RTC alarm ($\overline{\text{ALARM}}$), and as the output pin for the melody/alarm generator (MLDALM, $\overline{\text{MLDALM}}$) and as the Power control pin (PWE). The above settings are made using the function register PMFC.

PM1 has two output function which MLDALM and TA1OUT, and PM2 has two output functions $\overline{\text{ALARM}}$ and $\overline{\text{MLDALM}}$. These are selected using PM<PM1>, PM<PM2>.

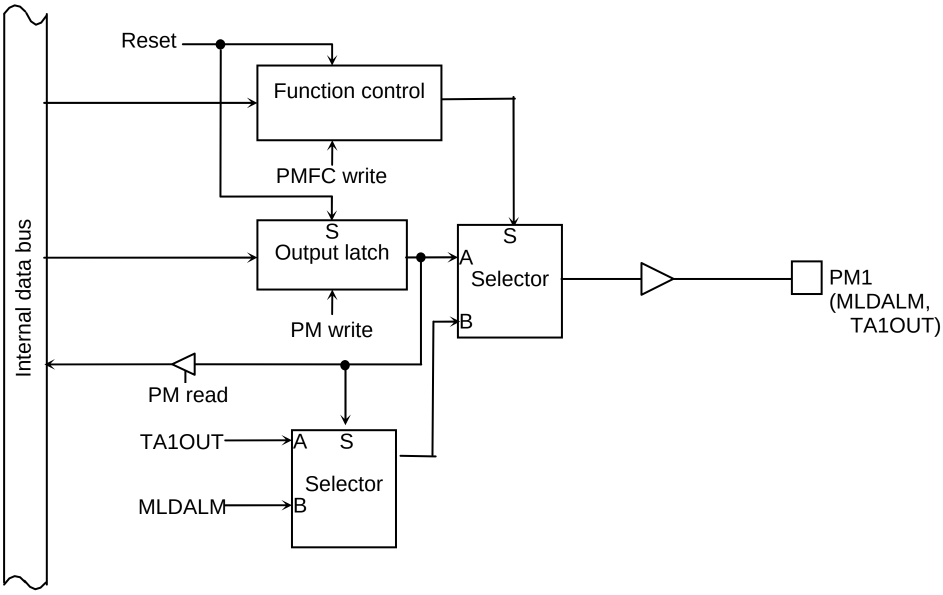


Figure 3.8.38 Port M1

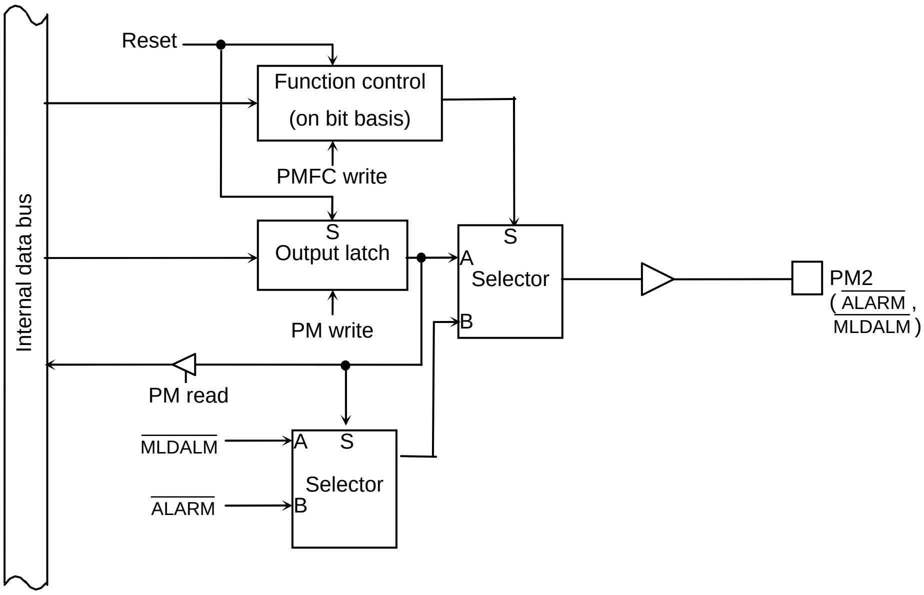


Figure 3.8.39 Port M2

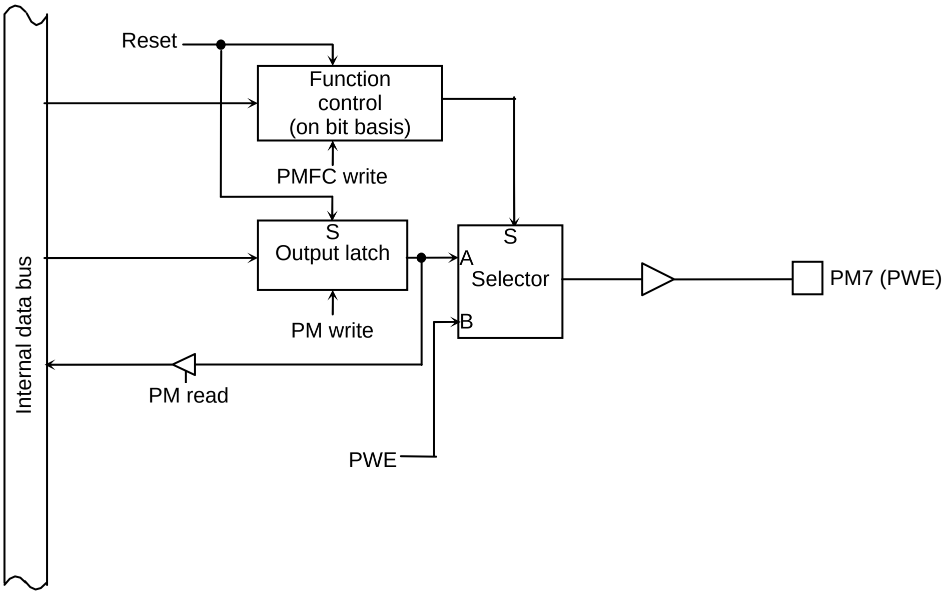


Figure 3.8.40 Port M7

Port M register

PM (0058H)		7	6	5	4	3	2	1	0
	bit Symbol	PM7					PM2	PM1	
	Read/Write	R/W					R/W		
	System Reset State	1					1	1	
	Hot Reset State	–					–	–	

Port M function register

PMFC (005BH)		7	6	5	4	3	2	1	0
	bit Symbol	PM7F					PM2F	PM1F	
	Read/Write	W					W		
	System Reset State	0					0	0	
	Hot Reset State	–					–	–	
	Function	0: Port 1: PWE					0: Port 1: $\overline{\text{ALARM}}$ at <PM2>=1, $\overline{\text{MLDALM}}$ at <PM2>=0	0: Port 1: MLDALM at <PM1>=1, TA1OUT at <PM1>=0	

Port M drive register

PMDR (0096H)		7	6	5	4	3	2	1	0
	bit Symbol	PM7D					PM2D	PM1D	
	Read/Write	R/W					R/W		
	System Reset State	1					1	1	
	Hot Reset State	–					–	–	
	Function	Input /Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		

Note: A read-modify-write operation cannot be performed for the registers PMFC.

Figure 3.8.41 Register for Port M

3.8.16 Port N (PN0 to PN7)

PN0 to PN7 are 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port N to an input port.

In addition to functioning as a general-purpose I/O port, Port N can also function as key-board interface pin (KO0 to KO7) which can be set to open-drain output buffer.

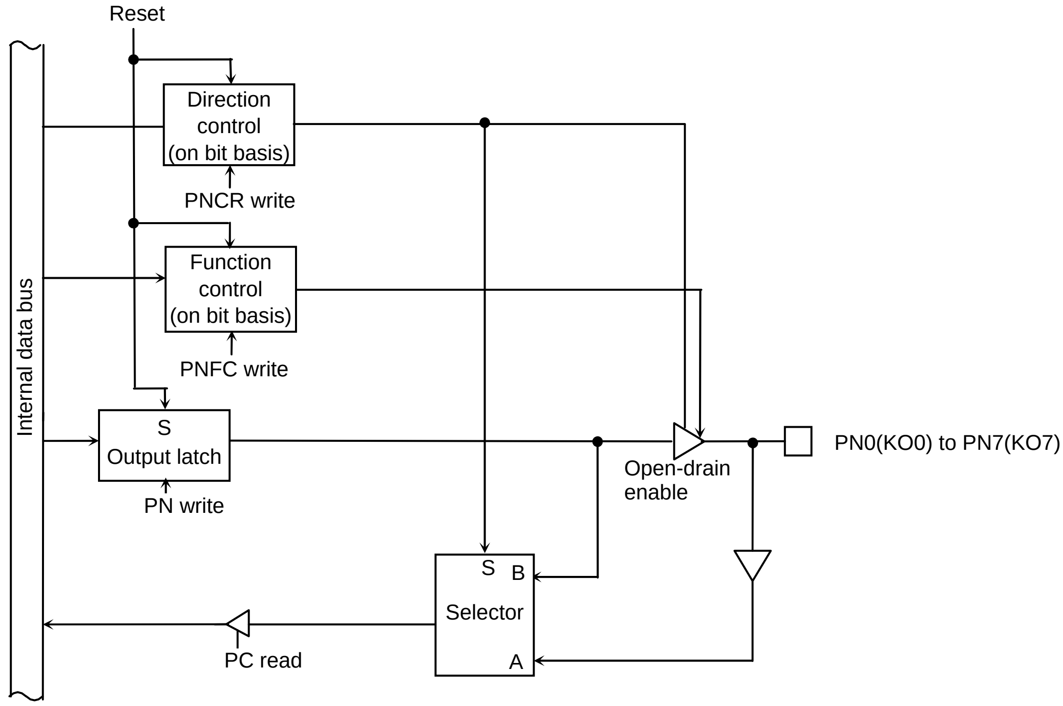


Figure 3.8.42 Port N

Port N register

PN (005CH)		7	6	5	4	3	2	1	0
	bit Symbol	PN7	PN6	PN5	PN4	PN3	PN2	PN1	PN0
	Read/Write	R/W							
	System Reset State	Data from external port (Output latch register is set to “1”)							
	Hot Reset State	–							

Port N control register

PNCR (005EH)		7	6	5	4	3	2	1	0
	bit Symbol	PN7C	PN6C	PN5C	PN4C	PN3C	PN2C	PN1C	PN0C
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: Input 1: Output							

Port N function register

PNFC (005FH)		7	6	5	4	3	2	1	0
	bit Symbol	PN7F	PN6F	PN5F	PN4F	PN3F	PN2F	PN1F	PN0F
	Read/Write	W							
	System Reset State	0	0	0	0	0	0	0	0
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	0: CMOS output 1: Open-drain output							

Port N drive register

PNDR (0097H)		7	6	5	4	3	2	1	0
	bit Symbol	PN7D	PN6D	PN5D	PN4D	PN3D	PN2D	PN1D	PN0D
	Read/Write	R/W							
	System Reset State	1	1	1	1	1	1	1	1
	Hot Reset State	–	–	–	–	–	–	–	–
	Function	Input/Output buffer drive register for standby mode							

Note : A read-modify-write operation cannot be performed for the registers PNCR and PNFC.

Figure 3.8.43 Register for Port N

3.8.17 Port P (PP3 to PP7)

Ports P3 to P5 are 3-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port P3 to P5 to input port and output latch to "0".

In addition to functioning as general-purpose I/O port, P3 to P5 can also function as an output pin for timer (TA7OUT), as an input pin for timers (TB0IN0, TB1IN0), and as an input pin for external interruption (INT5 to INT7), serial transfer SIO0 (TXD0, RXD0, SCLK0, $\overline{\text{CTS0}}$), SIO1 (TXD1, RXD1, SCLK1, $\overline{\text{CTS1}}$).

Port P6 is 1-bit output port. Resetting sets output latch to "0".

In addition to functioning as an output port, PP6 and PP7 can also function as an output pin for timer (TB0OUT0).

Setting in the corresponding bits of PPCR and PPFC enables the respective functions.

The edge select for external interruption is determined by the IIMC register in the interruption controller.

In port setting, if 16 bit timer input is selected and capture control is executed, INT6 and INT7 don't depend on IIMC1 register setting. INT6 and INT7 operate by setting TBnMOD<TBnCPM1:0>.

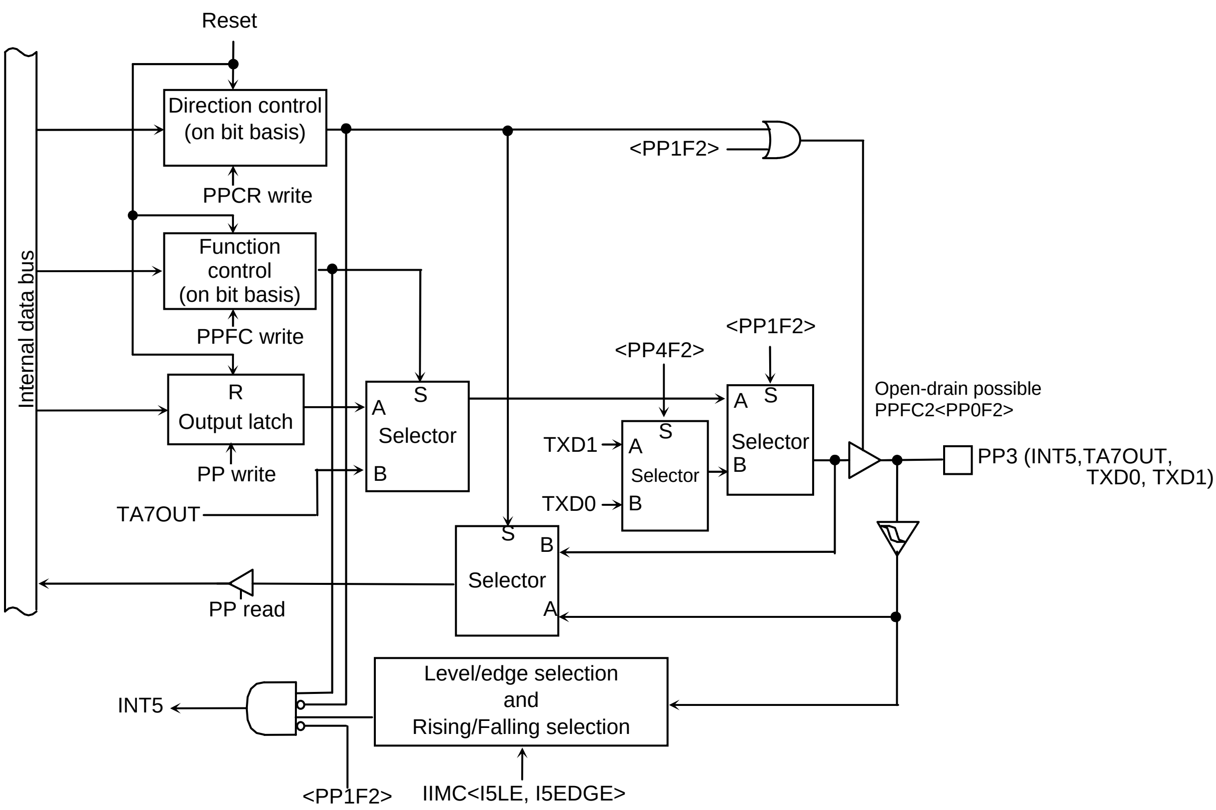


Figure 3.8.44 Port P3

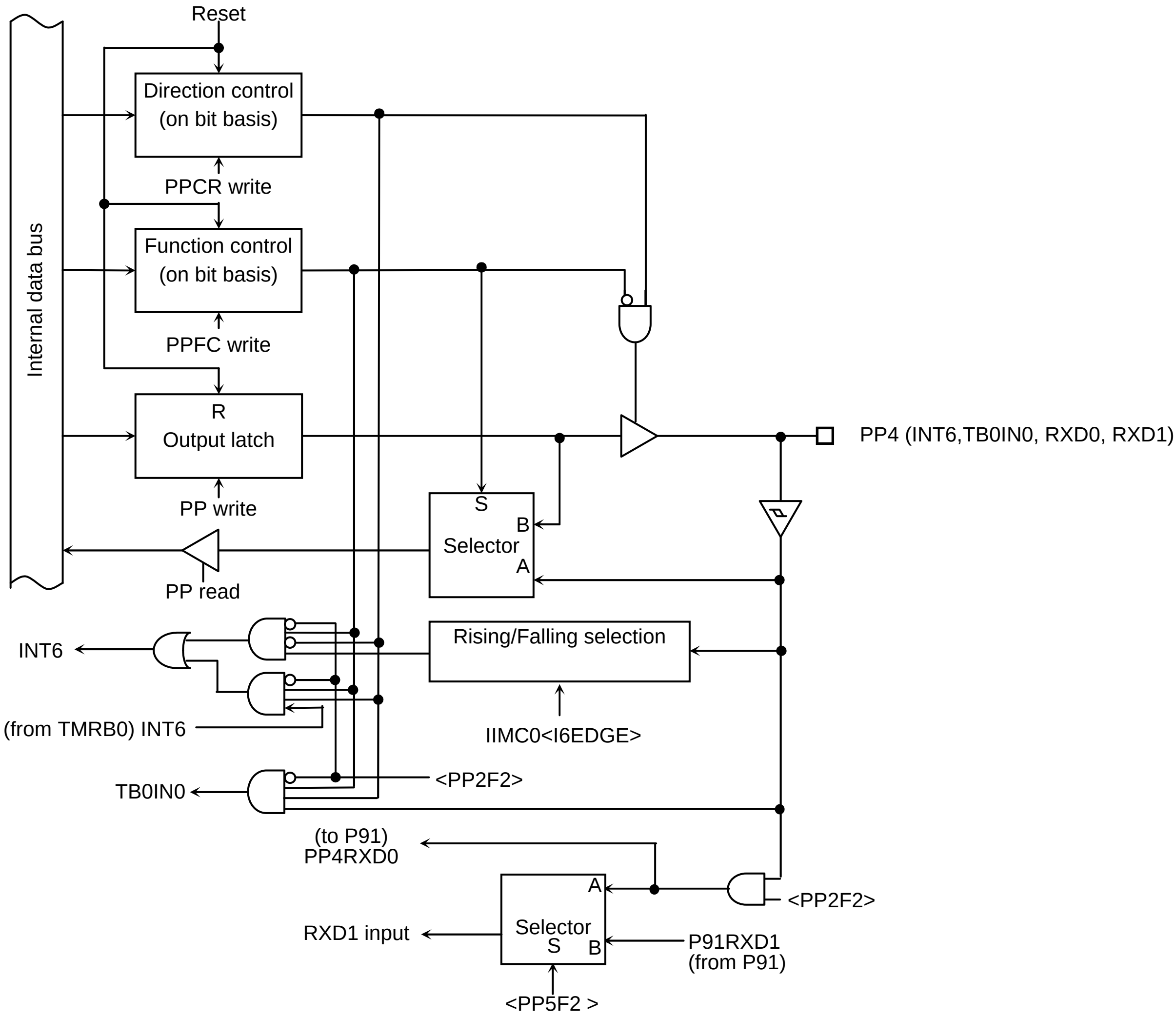


Figure 3.8.45 Port P4

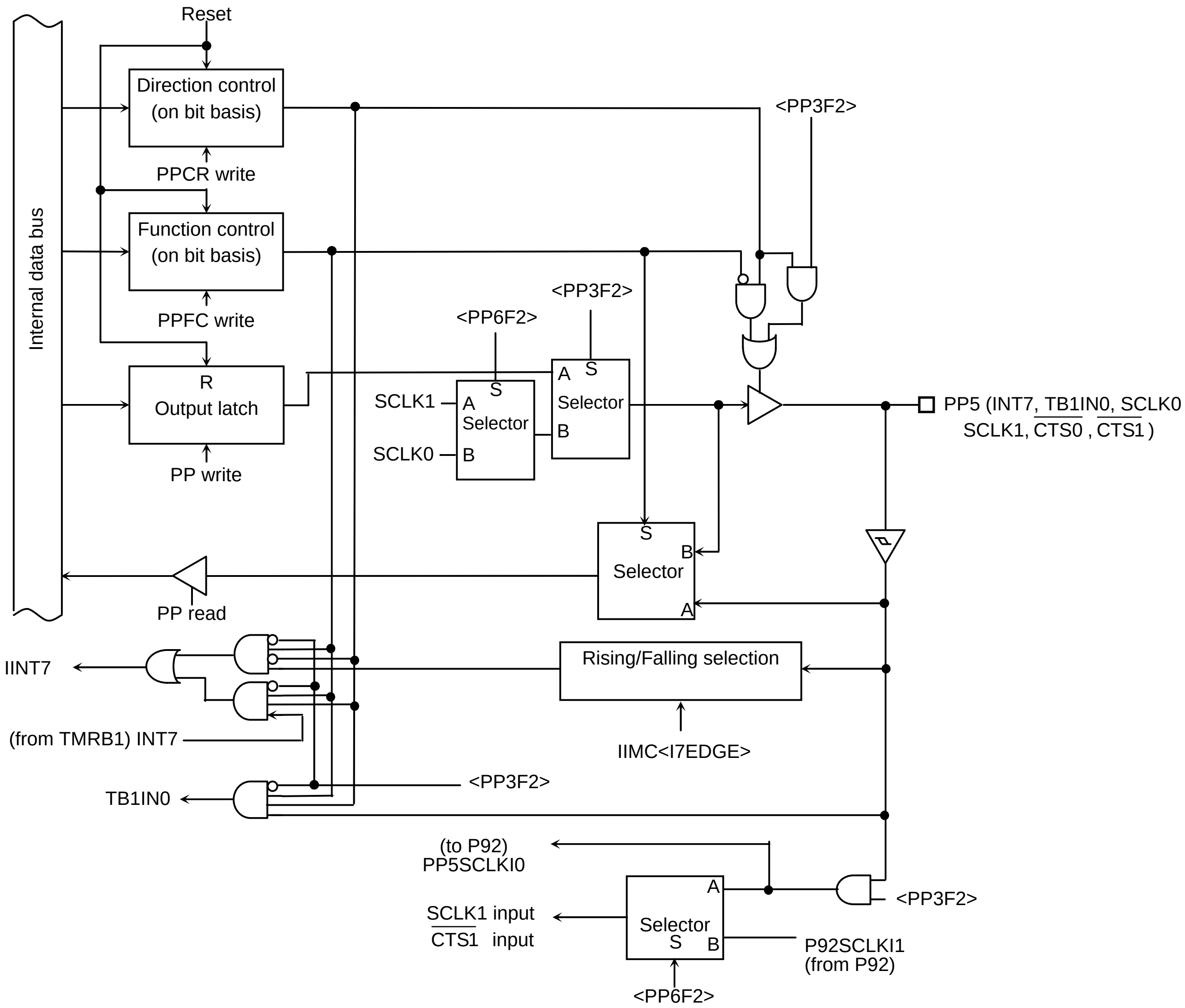


Figure 3.8.46 Port P5

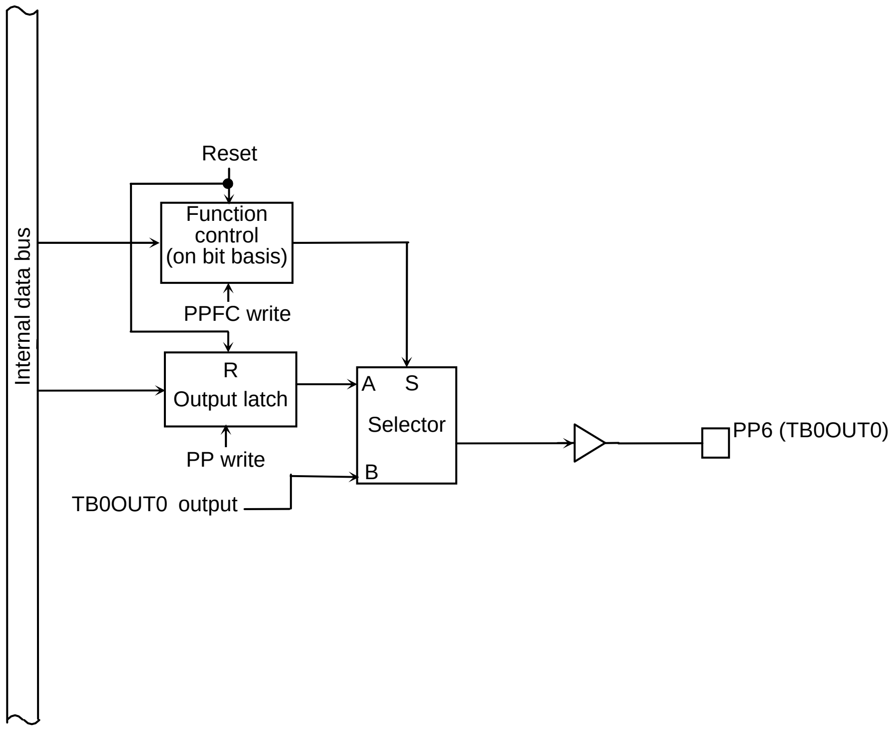


Figure 3.8.47 Port P6

Port P register

PP (0060H)

	7	6	5	4	3	2	1	0
bit Symbol		PP6	PP5	PP4	PP3			
Read/Write		R/W						
System Reset State		0	Data from external port (Output latch register is cleared to "0")					
Hot Reset State		–	–	–	–			

Port P control register

PPCR (0062H)

	7	6	5	4	3	2	1	0
bit Symbol			PP5C	PP4C	PP3C			
Read/Write			W					
System Reset State			0	0	0			
Hot Reset State			–	–	–			
Function			0: Input 1: Output					

Port P function register

PPFC (0063H)

	7	6	5	4	3	2	1	0
bit Symbol		PP6F	PP5F	PP4F	PP3F			
Read/Write		W						
System Reset State		0	0	0	0			
Hot Reset State		–	–	–	–			
Function		0:Port 1:TB0OUT0	Refer to following table					

Port P drive register

PPDR (0098H)

	7	6	5	4	3	2	1	0
bit Symbol		PP6D	PP5D	PP4D	PP3D			
Read/Write		R/W						
System Reset State		1	1	1	1			
Hot Reset State		–	–	–	–			
Function		Input/Output buffer drive register for standby mode						

Port P Function register 2

PPFC2 (0061H)		7	6	5	4	3	2	1	0
	bit Symbol		PP6F2	PP5F2	PP4F2	PP3F2	PP2F2	PP1F2	PP0F2
	Read/Write		W						
	System Reset State		0	0	0	0	0	0	0
	Hot Reset State		–	–	–	–	–	–	–
	Function		PP5 SCLK output 0: SCLK1 1: SCLK0 SIO1 SCLK, $\overline{\text{CTS}}$ input 0: PP5 1: P92	SIO1 RXD selection 0: PP4 1: P91	PP3 selection 0: TXD1 1: TXD0	PP5 selection 0: Others 1: SCLK, $\overline{\text{CTS}}$ input or SCLK output	PP4 selection 0: Others 1: RXD input	PP3 selection 0: Others 1: TXD output	PP3 selection 0: CMOS 1: Open -drain

PP3 setting (<PP1F2>=0)

<PP3F> \ <PP3C>	0	1
0	Input port	Output port
1	INT5 input	TA7OUT output

PP4 setting (<PP2F2>=0)

<PP4F> \ <PP4C>	0	1
0	Input port	Output port
1	INT5 input	TB0IN0 input

PP5 setting (<PP3F2>=0)

<PP5F> \ <PP5C>	0	1
0	Input port	Output port
1	INT7 input	TB1IN0 input

Note1: When setting <PP3F2, PP2F2, PP1F2> = "1", PP3~PP5 pins are set to SIO0 or SIO1 functions regardless PPCR, PPFC setting. PP3 is set to TXD, PP4 is set to RXD. PP5 is set to SCLK input or $\overline{\text{CTS}}$ input when <PP5C>=0. PP5 is set to SCLK output when <PP5C>=1.

Note2: A read-modify-write operation cannot be performed for the registers PPCR, PPFC.

Note3: When setting PP5, PP4, PP3 pins to INT7,INT6,INT5 input, set PPDR<PP5D:3D> to “0000” (prohibit input), and when driving PP5,PP4,PP3 pins to “0”, execute HALT instruction. This setting generates INT7, INT6, and INT5 inside. If don’t using external interrupt in HALT condition, set like an interrupt don’t generated.

Figure 3.8.48 Register for Port P

3.8.18 Port R (R0 to R3)

Ports R0 to R3 are 4-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port R0 to R3 to input port and output latch to "0".

In addition to functioning as general-purpose I/O port, PR0 to PR3 can also function as the SPI controller pin (SPCLK, $\overline{\text{SPCS}}$, SPDO and SPDI).

Setting in the corresponding bits of PFCR and PFFC enables the respective functions.

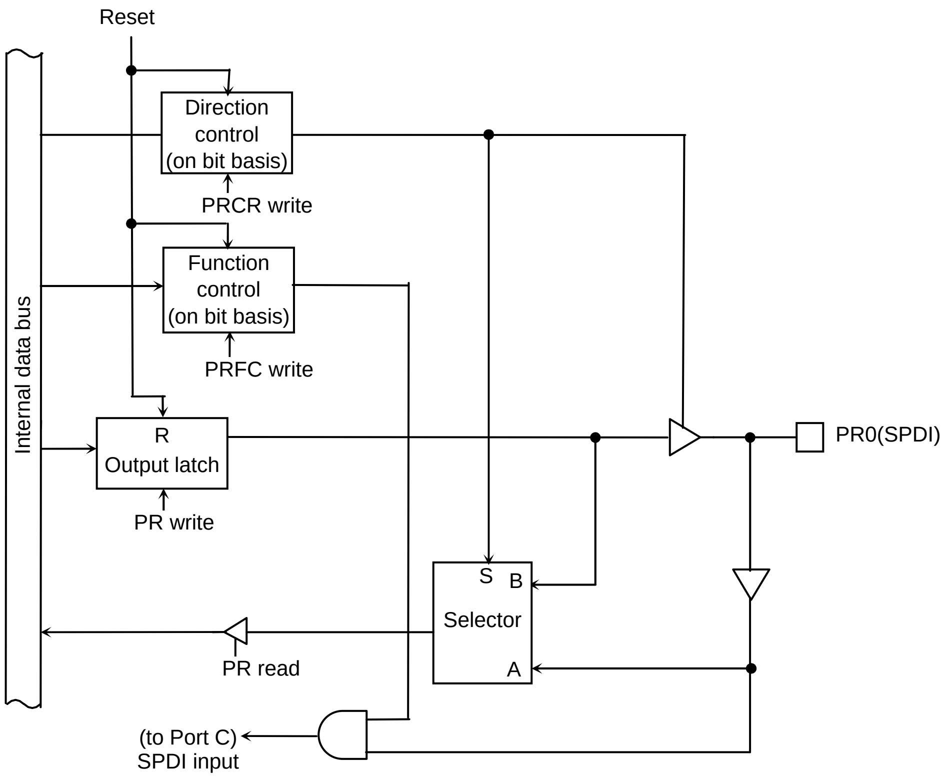


Figure 3.8.49 Port R0

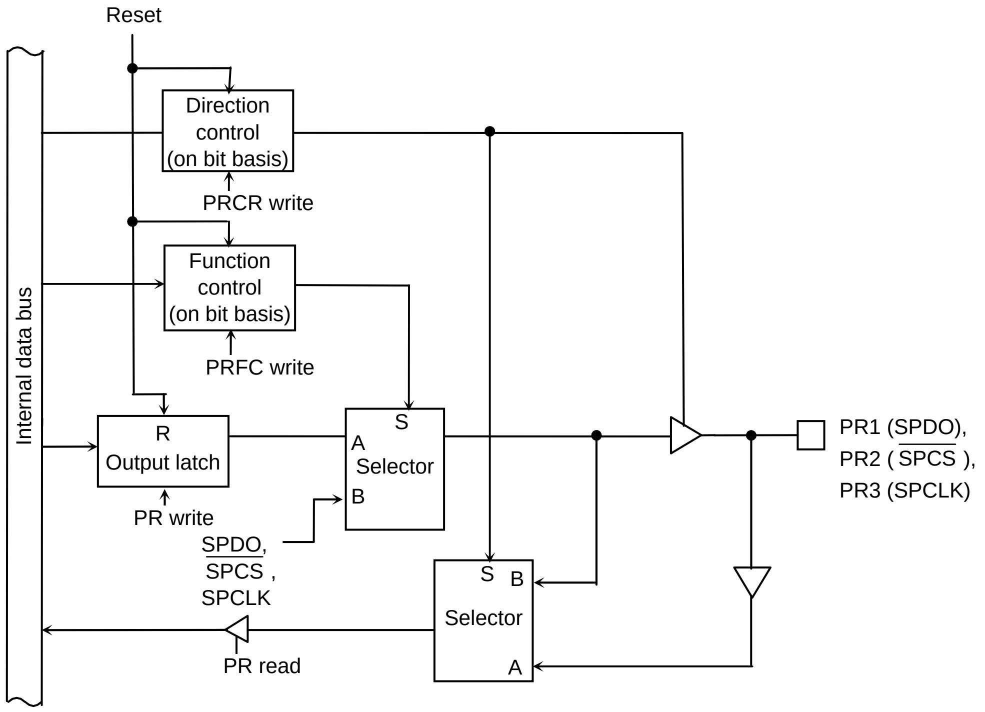


Figure 3.8.50 Port R1 to R3

Port R register

PR (0064H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3	PR2	PR1	PR0
Read/Write					R/W			
System Reset State					Data from external port (Output latch register is cleared to “0”)			
Hot Reset State					–			

Port R control register

PRCR (0066H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3C	PR2C	PR1C	PR0C
Read/Write					W			
System Reset State					0	0	0	0
Hot Reset State					–	–	–	–
Function					0: Input, 1: Output			

Port R function register

PRFC (0067H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3F	PR2F	PR1F	PR0F
Read/Write					W			
System Reset State					0	0	0	0
Hot Reset State					–	–	–	–
Function					0: Port 1: SPCLK	0: Port 1: $\overline{\text{SPCS}}$	0: Port 1: SPDO	0: Port 1: SPDI

Port R drive register

PRDR (0099H)

	7	6	5	4	3	2	1	0
bit Symbol					PR3D	PR2D	PR1D	PR0D
Read/Write					R/W			
System Reset State					1	1	1	1
Hot Reset State					–	–	–	–
Function					Input/Output buffer drive register for standby mode			

PR1 setting

<PR1F> \ <PR1C>	0	1
0	Input port	Output port
1	Reserved	SPDO output

PR0 setting

<PR0F> \ <PR0C>	0	1
0	Input port	Output port
1	SPDI input	Reserved

PR3setting

<PR3F> \ <PR3C>	0	1
0	Input port	Output port
1	Reserved	SPCLK output

PR2 setting

<PR2F> \ <PR2C>	0	1
0	Input port	Output port
1	Reserved	$\overline{\text{SPCS}}$ Output

Note: A read-modify-write operation cannot be performed for the registers PRCR, PRFC.

Figure 3.8.51 Register for Port R

3.8.19 Port T (PT0 to PT7)

Ports T0 to T7 are 8-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets ports T0 to T7 to input port and output latch to “0”.

In addition to functioning as general-purpose I/O port, PT0 to PT7 can also function as a data bus pin for LCD controller (LD8 to LD15).

Setting in the corresponding bits of PTCR and PTFC enables the respective functions.

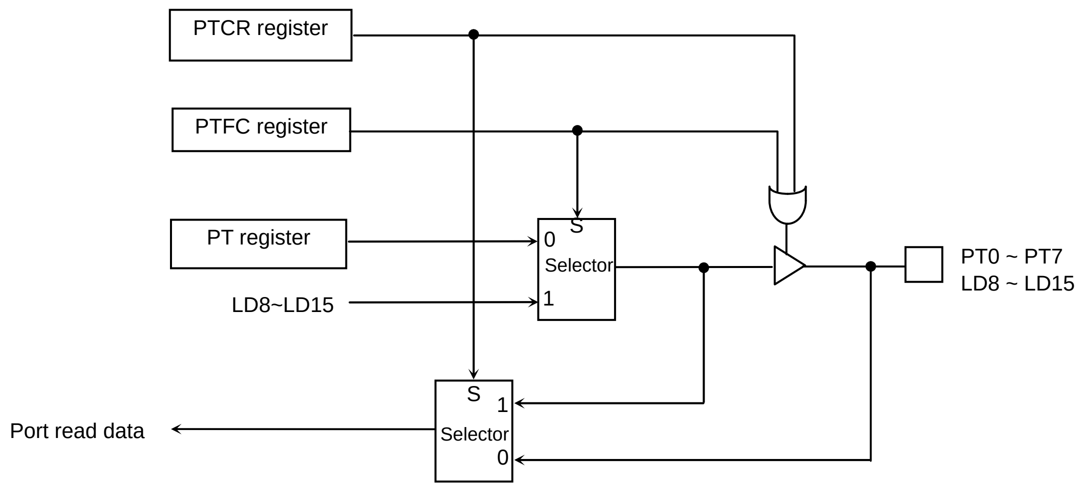


Figure 3.8.52 Port T0 to T7

Port T register

PT (00A0H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
Read/Write	R/W							
System Reset State	Data from external port (Output latch register is cleared to “0”)							
Hot Reset State	–							

Port T control register

PTCR (00A2H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7C	PT6C	PT5C	PT4C	PT3C	PT2C	PT1C	PT0C
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	0: Input 1: Output							

Port T function register

PTFC (00A3H)

	7	6	5	4	3	2	1	0
bit Symbol	PT7F	PT6F	PT5F	PT4F	PT3F	PT2F	PT1F	PT0F
Read/Write	W							
System Reset State	0	0	0	0	0	0	0	0
Hot Reset State	–	–	–	–	–	–	–	–
Function	0: Port 1: Data bus for LCDC (LD15 to LD8)							

Port T drive register

PTDR (009BH)

	7	6	5	4	3	2	1	0
bit Symbol	PT7D	PT6D	PT5D	PT4D	PT3D	PT2D	PT1D	PT0D
Read/Write	R/W							
System Reset State	1	1	1	1	1	1	1	1
Hot Reset State	–	–	–	–	–	–	–	–
Function	Input/Output buffer drive register for standby mode							

Note1: A read-modify-write operation cannot be performed for the registers PTCR, PTFC.

Note2: When PT is used as LD15 to LD8, set applicable PTnC to”1”.

<PTnF> \ <PTnC>	0	1
0	Input port	Output port
1	LDn	

Figure 3.8.53 Register for Port T

3.8.20 Port V (PV6, PV7)

Ports V6 and V7 are 2-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets port V6 and V7 to input port and output latch to “0”.

In addition to functioning as general-purpose I/O port, PV can also function as a input or output pin for SBI (SDA, SCL).

Setting in the corresponding bits of PVCR and PVFC enables the respective functions.

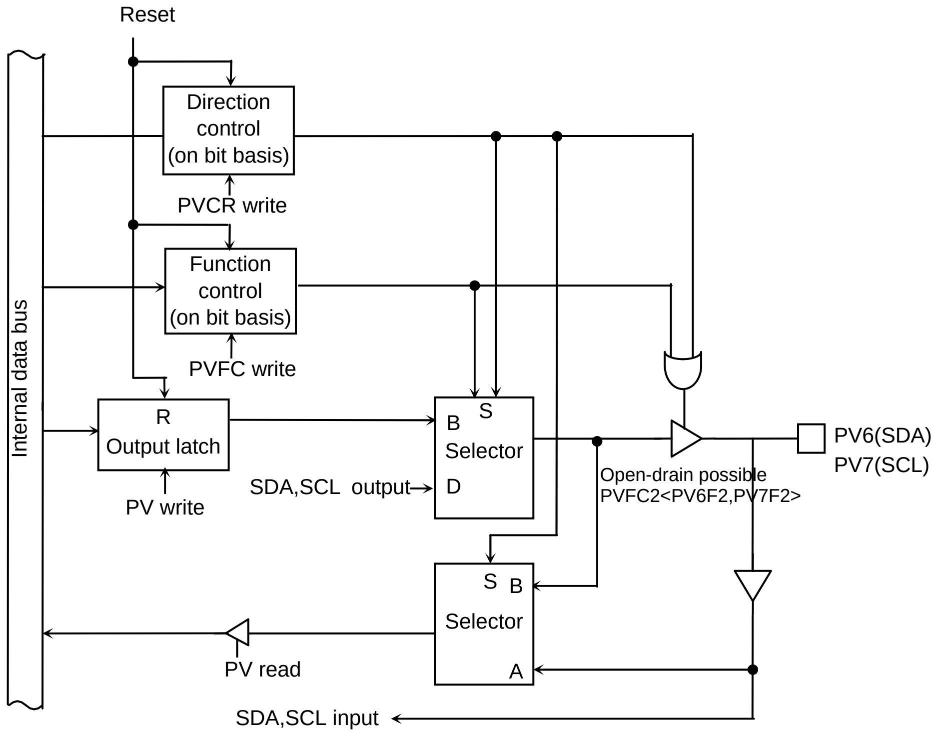


Figure 3.8.54 Port V6, V7

Port V register

PV (00A8H)

	7	6	5	4	3	2	1	0
bit Symbol	PV7	PV6						
Read/Write	R/W							
System Reset State	Data from external port (Output latch register is cleared to “0”)							
Hot Reset State	–							

Port V control register

PVCR (00AAH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7C	PV6C						
Read/Write								
System Reset State	0	0						
Hot Reset State	–	–						
Function	0: Input 1: Output							

Port V function register

PVFC (00ABH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7F	PV6F						
Read/Write	W							
System Reset State	0	0						
Hot Reset State	–	–						
Function	Refer to following table							

Port V function register 2

PVFC2 (00A9H)

	7	6	5	4	3	2	1	0
bit Symbol	PV7F2	PV6F2						
Read/Write	W							
System Reset State	0	0						
Hot Reset State	–	–						
Function	0: CMOS 1: Open -drain	0: CMOS 1: Open -drain						

Port V drive register

PVDR (009DH)

	7	6	5	4	3	2	1	0
bit Symbol	PV7D	PV6D						
Read/Write	R/W							
System Reset State	1	1						
Hot Reset State	–	–						
Function	Input/Output buffer drive register for standby mode							

PV7 setting

<PV7F> \ <PV7C>	0	1
0	Input port	Output port
1	Reserved	SCL I/O

PV6 setting

<PV6F> \ <PV6C>	0	1
0	Input port	Output port
1	Reserved	SDA I/O

Note: A read-modify-write operation cannot be performed for the registers PVCR, PVFC and PVFC2.

Figure 3.8.55 Register for Port V

3.8.21 Port X (PX4, PX5)

Port X5 is 1-bit general-purpose I/O ports. Each bit can be set individually for input or output. Resetting sets ports X5 to input port and output latch to “0”.

In addition to functioning as general-purpose I/O port, PX5 can also function as the USB clock input pin (X1USB) and dividing clock output of X1 and X2 oscillation clock (X1D4).

Setting in the corresponding bits of PXCR and PXFC enables the respective functions.

Port X4 is 1-bit general-purpose output port. Resetting sets output latch to “0”.

In addition to functioning as general-purpose output port, PX4 can also function as a system clock output pin (CLKOUT) and as an output pin (LDIV).

Setting in the corresponding bits of PX and PXFC enables the respective functions.

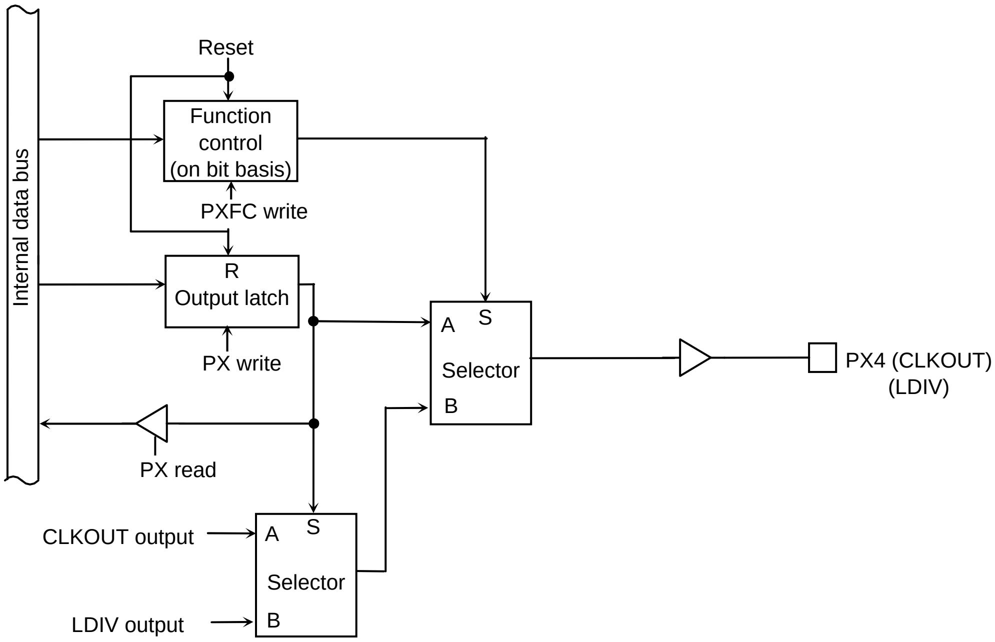


Figure 3.8.56 Port X4

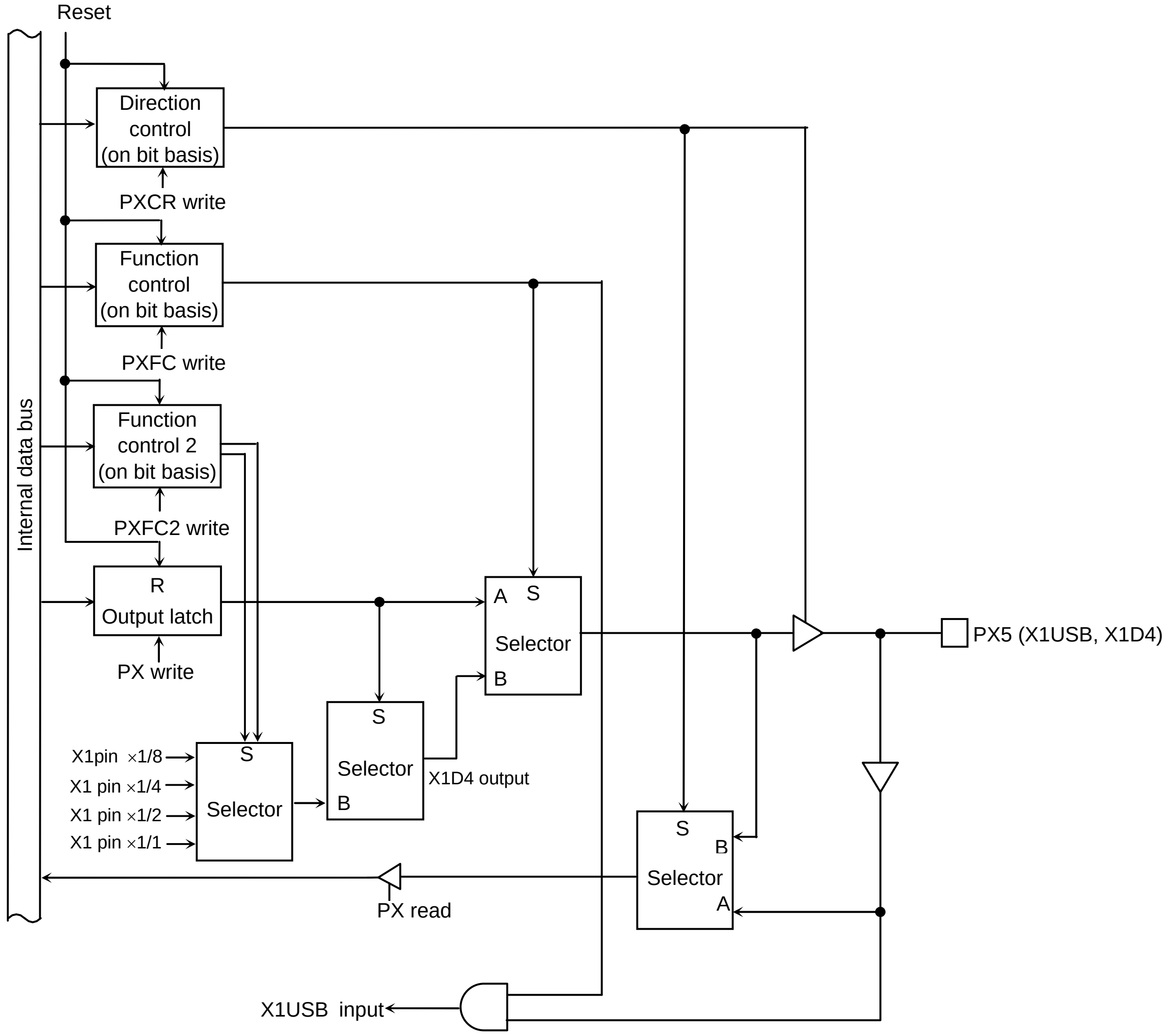


Figure 3.8.57 Port X5

Port X register

PX (00B0H)

	7	6	5	4	3	2	1	0
bit Symbol			PX5 Note3)	PX4 Note2)				
Read/Write			R/W					
System Reset State			Data from external port (Output latch register is cleared to “0”)					
Hot Reset State			–					

Port X control register

PXCR (00B2H)

	7	6	5	4	3	2	1	0
bit Symbol			PX5C					
Read/Write			W					
System Reset State			0					
Hot Reset State			–					
Function			0: Input 1: Output					

Port X function register

PXFC (00B3H)

	7	6	5	4	3	2	1	0
bit Symbol			PX5F	PX4F				
Read/Write			W					
System Reset State			0	0				
Hot Reset State			–	–				
Function			Refer to following table					

Port X function register 2

PXFC2 (00B1H)

	7	6	5	4	3	2	1	0
bit Symbol			PX5F2	PX4F2				
Read/Write			R/W					
System Reset State			0	0				
Hot Reset State			–	–				
Function			X1D4 output clock selection 00: X1 pin ×1/8 01: X1 pin ×1/4 10: X1 pin ×1/2 11: X1 pin ×1/1					

Port X drive register

PXDR (009FH)		7	6	5	4	3	2	1	0
	bit Symbol			PXD5	PXD4				
	Read/Write			R/W					
	System Reset State			1	1				
	Hot Reset State			–	–				
	Function			Input/Output buffer drive register for standby mode					

Note 1: A read-modify-write operation cannot be performed for the registers PXCR, PXFC and PXFC2.

Note 2: When PXFC<PX4F>= "1", Function is changed by PX<PX4> setting. Refer to following PX4 setting table.

Note 3: When PX5 is used as X1D4 pin, PX<PX5> must be set to "1". Refer to following PX5 setting table.

PX4 setting

<PX4F> \ <PX4>	0	1
0	Output port	
1	CLKOUT output	LDIV output

PX5 setting

<PX5F> \ <PX5C>	0	1
0	Input port	Output port
1	X1USB input	X1D4 output at <PX5>= "1"

Figure 3.8.58 Register for Port X

3.9 Memory Controller (MEMC)

3.9.1 Functional Overview

The TMP92CF29A has a memory controller with the following features to control four programmable address spaces:

(1) Four programmable address spaces

The MEMC can specify a start address and a block size for each of the four memory spaces (CS0 to CS3 spaces).

* SRAM or ROM: All CS spaces (CS0 to CS3) can be assigned.

* SDRAM: Either the CS1 or CS2 space can be assigned.

* Page-ROM: Only the CS2 space can be assigned.

* NAND-Flash: It is not required to setup the CS lines. However, when using NAND-Flash, set the BROMCR<CSDIS> bit to "1" to assign an external area to avoid data conflicts with CS spaces.

(2) Memory specification

The MEMC can specify the type of memory, SRAM, ROM and SDRAM to associate with the selected address spaces.

(3) Data bus width specification

The data bus width is selectable from 8 and 16 bits for the respective chip select spaces.

(4) Wait control

The number of wait states to be inserted into an external bus cycle is determined by the wait state bits of the control register and the $\overline{\text{WAIT}}$ input pin. The number of wait states of a read cycle and that of a write cycle can be specified individually. The number of wait states can be selected from the following 15 options:

0 to 10 wait states, 12 wait states,
16 wait states, 20 wait states
4+N wait states (controlled by the $\overline{\text{WAIT}}$ pin)

3.9.2 Control Registers and Memory Access Operations After Reset

This section describes the registers to control the memory controller, their reset states and the necessary settings after reset.

(1) Control Registers

The control registers of the memory controller are listed below.

- Control registers: BnCSH/BnCSL(n = 0 to 3, EX)
  Configures the basic settings of the memory controller, such as the memory type specification and the number of wait states to be inserted into a read or write cycle.
- Memory Start Address register: MSARn(n = 0 to 3)
  Specifies a start address fora selected address space.
- Memory Address Mask register: MAMR (n = 0 to 3)
  Specifies a block size for a selected address space.
- Page ROM Control register: PMEMCR
  Selects a method of accessing Page-ROM.
- Timing control registers: CSTMGCR, WRTMGCR, RDTMGCRn
  Adjust the timing of rising and falling edges of control signals.
- On-chip Boot ROM Control register: BROMCR
  Selects a method of accessing Boot-ROM.

Table 3.9.1 Control Registers

Register		7	6	5	4	3	2	1	0
B0CSL	Bit Symbol	B0WW3	B0WW2	B0WW1	B0WW0	B0WR3	B0WR2	B0WR1	B0WR0
(0140H)	Read/Write	R/W							
	Reset State	0	0	1	0	0	0	1	0
B0CSH	Bit Symbol	B0E			B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
(0141H)	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
MAMR0	Bit Symbol	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14-V9	M0V8
(0142H)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR0	Bit Symbol	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
(0143H)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B1CSL	Bit Symbol	B1WW3	B1WW2	B1WW1	B1WW0	B1WR3	B1WR2	B1WR1	B1WR0
(0144H)	Read/Write	R/W							
	Reset State	0	0	1	0	0	0	1	0
B1CSH	Bit Symbol	B1E			B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
(0145H)	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
MAMR1	Bit Symbol	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	M1V15-V9	M1V8
(0146H)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR1	Bit Symbol	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
(0147H)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B2CSL	Bit Symbol	B2WW3	B2WW2	B2WW1	B2WW0	B2WR3	B2WR2	B2WR1	B2WR0
(0148H)	Read/Write	R/W							
	Reset State	0	0	1	0	0	0	1	0
B2CSH	Bit Symbol	B2E	B2M		B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
(0149H)	Read/Write	R/W			R/W				
	Reset State	1	0		0	0	0	0	1
MAMR2	Bit Symbol	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
(014AH)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR2	Bit Symbol	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
(014BH)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
B3CSL	Bit Symbol	B3WW3	B3WW2	B3WW1	B3WW0	B3WR3	B3WR2	B3WR1	B3WR0
(014CH)	Read/Write	R/W							
	Reset State	0	0	1	0	0	0	1	0
B3CSH	Bit Symbol	B3E			B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
(014DH)	Read/Write	R/W			R/W				
	Reset State	0			0	0	0	0	0
MAMR3	Bit Symbol	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
(014EH)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1
MSAR3	Bit Symbol	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
(014FH)	Read/Write	R/W							
	Reset State	1	1	1	1	1	1	1	1

Table 3.9.2 Control Registers

Register		7	6	5	4	3	2	1	0
BEXCSL (0158H)	Bit Symbol	BEXWW3	BEXWW2	BEXWW1	BEXWW0	BEXWR3	BEXWR2	BEXWR1	BEXWR0
	Read/Write	R/W							
	Reset State	0	0	1	0	0	0	1	0
BEXCSH (0159H)	Bit Symbol				BEXREC	BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
	Read/Write				R/W				
	Reset State				0	0	0	0	0
PMEMCR (0166H)	Bit Symbol				OPGE	OPWR1	OPWR0	PR1	PR0
	Read/Write				R/W	R/W		R/W	
	Reset State				0	0	0	1	0
CSTMGCR (0168H)	Bit Symbol			TACSEL1	TACSEL0			TAC1	TAC0
	Read/Write			R/W				R/W	
	Reset State			0	0			0	0
WRTMGCR (0169H)	Bit Symbol			TCWSEL1	TCWSEL0	TCWS1	TCWS0	TCWH1	TCWH0
	Read/Write			R/W		R/W		R/W	
	Reset State			0	0	0	0	0	0
RDTMGCR0 (016AH)	Bit Symbol	B1TCRS1	B1TCRS0	B1TCRH1	B1TCRH0	B0TCRS1	B0TCRS0	B0TCRH1	B0TCRH0
	Read/Write	R/W		R/W		R/W		R/W	
	Reset State	0	0	0	0	0	0	0	0
RDTMGCR1 (016BH)	Bit Symbol	B3TCRS1	B3TCRS0	B3TCRH1	B3TCRH0	B2TCRS1	B2TCRS0	B2TCRH1	B2TCRH0
	Read/Write	R/W		R/W		R/W		R/W	
	Reset State	0	0	0	0	0	0	0	0
BROMCR (016CH)	Bit Symbol						CSDIS	ROMLESS	VACE
	Read/Write						R/W		
	Reset State						1	0/1	1/0
RAMCR (016DH)	Bit Symbol								–
	Read/Write								R/W
	Reset State								Must be written as “1”.

(2) Memory Access Operations After Reset

After reset, external memory is accessed using the initial data bus width that is determined by the AM1 and AM0 pins. The settings of the AM1 and AM0 pins and their corresponding operation modes are as follows:

AM1	AM0	Start Mode
0	0	Don’t use this setting
0	1	Boots from external memory using a16-bit data bus (Note)
1	0	Don’t use this setting
1	1	Boots from the on-chip boot ROM (32-bit on-chip-MROM)

Note: The memory that is used for booting after reset must be either NOR-Flash or Masked-ROM. NAND-Flash and SDRAM cannot be used.

The values of AM1 and AM0 are effective only upon reset. The data bus width is specified by the <BnBUS1:BnBUS0> bits of the control registers at any other timing.

Upon reset, only the control registers (B2CSH and B2CSL) for the CS2 space automatically becomes effective. (The B2CSH<B2E> bit is set to 1 upon reset.).Then, the AM1 and AM0 values that specify the data bus width are loaded into the data bus width specification bits of the control register for the CS2 space.At the same time, the address range ebtween 000000H and FFFFFFH is defined as the CS2 space. (The B2CSH<B2M> is cleared to 0.)

Then, the address spaces are configured by MSARn and MAMRn. The BnCSH and BnCSL registers are also set up. The BnCSH<BnE> must be set to 1 to enable these settings.

3.9.3 Basic Functions and Register Settings

This section describes some of the memory controller functions, such as setting the address range for each address space, associating memory to the selected space and setting the number of wait states to be inserted.

(1) Programming chip select spaces

The address ranges of CS0 to CS3 are specified by MSAR0 to MSAR3 and MAMR0 to MAMR3.

(a) Memory Start Address registers

Figure 3.9.1 shows the Memory Start Address registers. The MSAR0 to MSAR3 specify the start addresses for the CS0 to CS3 spaces. The bits S23 to S16 specify the upper 8 bits (A23 to A16) of the start address. The lower 16 bits of the start address (A15 to A0) are assumed to be 0. Accordingly, the start address can only be a multiple of 64 Kbytes, ranging from 000000H to FF0000H. Figure 3.9.2 shows the relationship between the start addresses and the Memory Start Address register values.

Memory Start Address Registers (for CS0 to CS3 spaces)

MSAR0 (0143H) / MSAR1 (0147H)
MSAR2 (014BH) / MSAR3 (014FH)

	7	6	5	4	3	2	1	0
Bit Symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
Reset State	1	1	1	1	1	1	1	1
Function	Determines A23 to A16 of the start address							

→ Specifies start addresses for CS0 to CS3 spaces

Figure 3.9.1 Memory Start Address Register

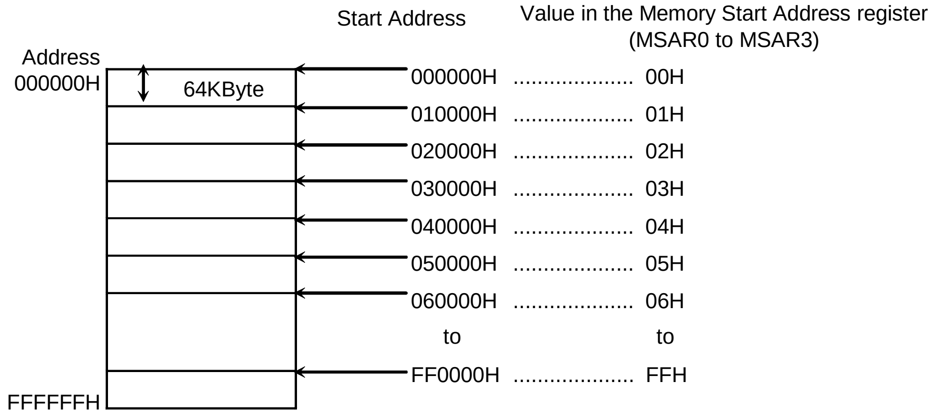


Figure 3.9.2 Relationship Between Start Addresses and the Memory Start Address Register Values

(b) Memory Address Mask Registers

Figure 3.9.3 shows the Memory Address Mask registers. MAMR0 to MAMR3 are used to determine the sizes of the CS0 to CS3 spaces by setting particular bits in MAMR0 to MAMR3 to mask the corresponding start address bits. The address compare logic uses only the address bits that are not masked (i.e., mask bit cleared to 0) to detect an address match in the CS0 to CS3 spaces. The upper bits are always compared.

Also, the address bits that can be masked by MAMR0 to MAMR3 differ between CS0 to CS3 spaces as follows:

CS0 space: A20 to A8

CS1 space: A21 to A8

CS2 and CS3 spaces: A22 to A15

Accordingly, the block size that can be assigned to each space is also different.

Note: After reset, only the control register for the CS2 space is effective. The control register for the CS2 space has the B2M bit. If the B2M bit is cleared to 0, the address range between 000000H and FFFFFFH is defined as the CS2 space. (The B2M bit is cleared to 0after reset.) By setting the B2CSH<B2M> bit to 1, the start address and the block size can be arbitrarily specified, as in the other spaces.

Memory Address Mask Register (for CS0 space)

MAMR0 (0142H)

	7	6	5	4	3	2	1	0
Bit Symbol	V20	V19	V18	V17	V16	V15	V14~9	V8
Read/Write	R/W							
Reset State	1	1	1	1	1	1	1	1
Function	CS0 block size 0: The address compare logic uses this address bit							

The CS0 block size can vary from 256 Bytes to 2 Mbytes

Memory Address Mask Register (for CS1 space)

MAMR1 (0146H)

	7	6	5	4	3	2	1	0
Bit Symbol	V21	V20	V19	V18	V17	V16	V15~9	V8
Read/Write	R/W							
Reset State	1	1	1	1	1	1	1	1
Function	CS1 block size 0: The address compare logic uses this address bit							

The CS1 block size can vary from 256 Bytes to 4 Mbytes

Memory Address Mask Register (for CS2 and CS3 spaces)

MAMR2 (014AH) / MSAR3 (014FH)

	7	6	5	4	3	2	1	0
Bit Symbol	V22	V21	V20	V19	V18	V17	V16	V15
Read/Write	R/W							
Reset State	1	1	1	1	1	1	1	1
Function	CS2 or CS3 block size 0: The address compare logic uses this address bit.							

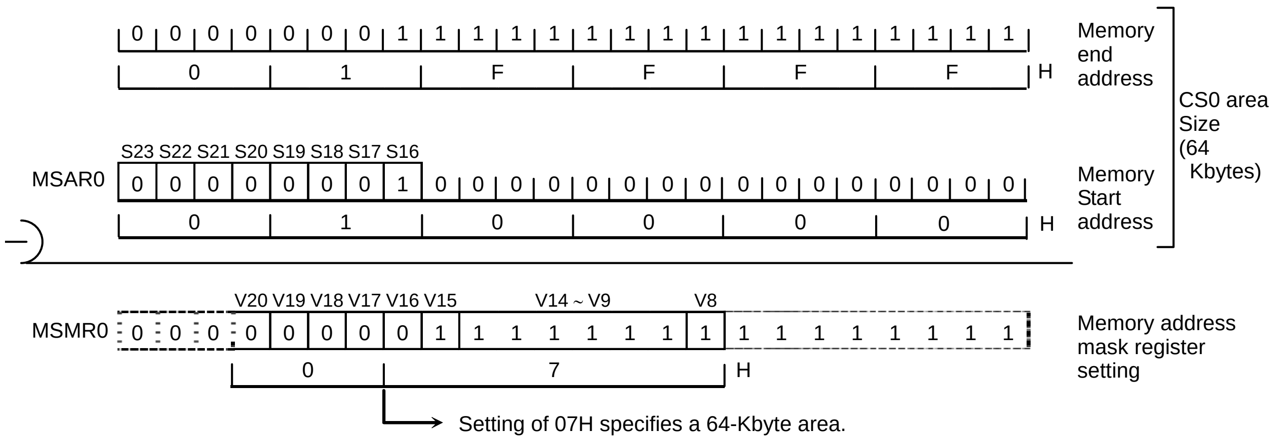
The CS2 and CS3 block sizes can vary from 32 Kbytes to 8 Mbytes

Figure 3.9.3 Memory Address Mask Registers

(c) Setting the start addresses and address ranges

An example of specifying a 64-Kbyte address space starting from 010000H for the CS0 space:

Set 01H in the MSAR0<S23:S16> bits that corresponds to the upper 8 bits of the start address. Then, calculate the difference between the start address and the anticipated end address (01FFFFH) based on the size of the CS0 space. Bits 20 to 8 of the calculation result correspond to the mask value to be set for the CS0 space. Setting this value in the MAMR0<V20:V8> bits specifies the block size. This example sets 07H in MAMR0 to allocate a 64-Kbyte address space for the CS0 space.



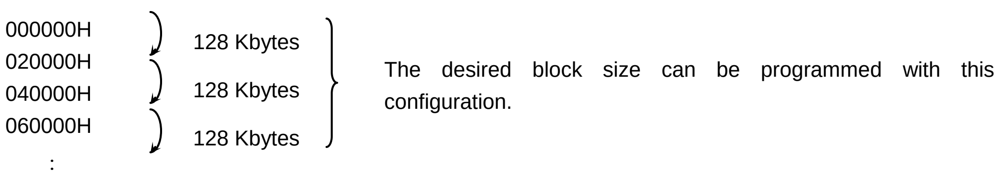
(d) Programming block sizes

Table 3.9.3 shows the relationship between CS spaces and their block sizes. The “Δ” symbol indicates the size that might not be programmable depending on the combination of the values of the Memory Start Address and Memory Address Mask registers. When specifying a block size indicated as “Δ”, set the start address register to a multiple of the desired block size starting from 000000H.

If the 16-Mbyte range is defined as CS2 space, or if two or more spaces overlap, the settings for the CS space with the smallest number overrides the settings for other spaces because of its highest priority.

Example: Defining 128 Kbyte area as the CS0 space:

a. Valid start addresses



b. Invalid start addresses

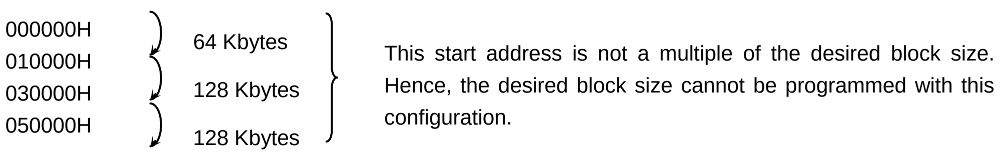


Table 3.9.3 Valid Block Sizes for Each CS Space

CS space \ Size (Byte)	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	Δ	Δ	Δ	Δ	Δ		
CS1	○	○		○	Δ	Δ	Δ	Δ	Δ	Δ	
CS2			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ
CS3			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ

Note: The "Δ" symbol indicates the sizes that may not be programmable depending on the combination of the values of the Memory Start Address and Memory Address Mask registers.

(e) Priorities of the address spaces

When the specified address space overlaps with the on-chip memory area, the priority order of the address spaces are as follows:

On-chip I/O > On-chip memory > CS0 space > CS1 space > CS2 space > CS3 space

(f) Specifying the number of wait states and the bus width for the address locations outside the CS0 to CS3 spaces

The BEXCSL and BEXCSH registers specify the data bus width and number of wait states when an address outside the CS0 to CS3 spaces ($\overline{\text{CSEX}}$ space) is accessed. These registers are always enabled for the CSEX space.

(2) Memory specification

Setting the BnCSH<BnOM1:BnOM0> bits specifies the memory type that is associated with each address spaces. The interface signal that corresponds to the specified memory type is generated. The memory type is specified as follows:

BnCSH<BnOM1:0>

BnOM1	BnOM0	Memory Type
0	0	SRAM/ROM (Default)
0	1	(Reserved)
1	0	(Reserved)
1	1	SDRAM

Note : SDRAM can be associated with the CS1 or CS2 space.

(3) Data bus width specification

The data bus width can be specified for each address space by the BnCSH<BnBUS1:BnBUS0> bits as follows:

BnCSH<BnBUS1:BnBUS0>

<BnBUS1>	<BnBUS0>	Bus Width
0	0	8-bit bus mode (Default)
0	1	16-bit bus mode
1	0	Reserved
1	1	Don't use this setting

Note: The data bus width for SDRAM should be defined as 16 bits by setting BnCSH<BnBUS1:BnBUS0> to 01.

As described above, the TMP92CF29A supports dinamic bus sizing, which allows the controller to transfer operands to or from the selected address spaces while automatically determining the data bus width. On which part of the data bus the data is actually placed is determined by the data size, bus width and start address. The table below provides a detailed description of the actual bus operation.

The TMP92CF29A has only 16 external data bus pins. Therefore, please ignore the setting information of when the memory bus width is set to be 32 bits in the table.

Note: If two memories with different bus widths are assigned to consecutive addresses, do not execute an instruction that accesses the addresses crossing the boundary between those memories. Otherwise, a read/write operation might not be performed correctly.

Operand Data Size (bit)	Operand Start Address	Memory Bus Width (bit)	CPU Address	CPU Data			
				D31 to D24	D23 to D16	D15 to D8	D7 to D0
8	4n + 0	8/16/32	4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
	4n + 1	8	4n + 1	xxxxx	xxxxx	xxxxx	b7 to b0
		16/32	4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
	4n + 2	8/16	4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
		32	4n + 2	xxxxx	b7 to b0	xxxxx	xxxxx
	4n + 3	8	4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
		16	4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
		32	4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
16	4n + 0	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
		16/32	4n + 0	xxxxx	xxxxx	b15 to b8	b7 to b0
	4n + 1	8	(1) 4n + 1	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 2	xxxxx	xxxxx	xxxxx	b15 to b8
		16	(1) 4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 2	xxxxx	xxxxx	xxxxx	b15 to b8
		32	4n + 1	xxxxx	b15 to b8	b7 to b0	xxxxx
	4n + 2	8	(1) 4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
		16	4n + 2	xxxxx	xxxxx	b15 to b8	b7 to b0
		32	4n + 2	b15 to b8	b7 to b0	xxxxx	xxxxx
	4n + 3	8	(1) 4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
		16	(1) 4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
		32	(1) 4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
32	4n + 0	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 2	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 3	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 0	xxxxx	xxxxx	b15 to b8	b7 to b0
			(2) 4n + 2	xxxxx	xxxxx	b31 to b24	b23 to b16
		32	4n + 0	b31 to b24	b23 to b16	b15 to b8	b7 to b0
	4n + 1	8	(1) 4n + 0	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 1	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 2	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 3	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 1	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 2	xxxxx	xxxxx	b23 to b16	b15 to b8
			(3) 4n + 4	xxxxx	xxxxx	xxxxx	b31 to b24
		32	(1) 4n + 1	b23 to b16	b15 to b8	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b31 to b24
	4n + 2	8	(1) 4n + 2	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 3	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 4	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 5	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 2	xxxxx	xxxxx	b15 to b8	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	b31 to b24	b23 to b16
		32	(1) 4n + 2	b15 to b8	b7 to b0	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	b31 to b24	b23 to b16
	4n + 3	8	(1) 4n + 3	xxxxx	xxxxx	xxxxx	b7 to b0
			(2) 4n + 4	xxxxx	xxxxx	xxxxx	b15 to b8
			(3) 4n + 5	xxxxx	xxxxx	xxxxx	b23 to b16
			(4) 4n + 6	xxxxx	xxxxx	xxxxx	b31 to b24
		16	(1) 4n + 3	xxxxx	xxxxx	b7 to b0	xxxxx
			(2) 4n + 4	xxxxx	xxxxx	b23 to b16	b15 to b8
			(3) 4n + 6	xxxxx	xxxxx	xxxxx	b31 to b24
		32	(1) 4n + 3	b7 to b0	xxxxx	xxxxx	xxxxx
			(2) 4n + 4	xxxxx	b31 to b24	b23 to b16	b15 to b8

xxxxx: The input data placed on the data bus indicated by this symbol is ignored during a read operation. During a write operation, the bus is in the high-impedance state, and the write strobe signal remains inactive.

(4) Wait control

The external bus cycle completes in two states at minimum (25 ns at f_{SYS} = 80 MHz) without inserting a wait state.

Setting up the BnCSL<BnWW3:BnWW0> bits specifies the number of wait states to be inserted in a write cycle, and setting the BnCSL<BnWR3:BnWR0> bits specifies the number of wait states to be inserted in a read cycle. The external bus cycle can be programmed as follows;

BnCSL<BnWW>/<BnWR>

<BnWW3> <BnWR3>	<BnWW2> <BnWR2>	<BnWW1> <BnWR1>	<BnWW0> <BnWR0>	Number of Wait States
0	0	0	1	2 states (0 wait state), fixed wait-state mode
0	0	1	0	3 states (1 wait state), fixed wait-state mode (Default)
0	1	0	1	4 states (2 wait states), fixed wait-state mode
0	1	1	0	5 states (3 wait states), fixed wait-state mode
0	1	1	1	6 states (4 wait states), fixed wait-state mode
1	0	0	0	7 states (5 wait states), fixed wait-state mode
1	0	0	1	8 states (6 wait states), fixed wait-state mode
1	0	1	0	9 states (7 wait states), fixed wait-state mode
1	0	1	1	10 states (8 wait states), fixed wait-state mode
1	1	0	0	11 states (9 wait states), fixed wait-state mode
1	1	0	1	12 states (10 wait states), fixed wait-state mode
1	1	1	0	14 states (12 wait states), fixed wait-state mode
1	1	1	1	18 states (16 wait states), fixed wait-state mode
0	1	0	0	22 states (20 wait states), fixed wait-state mode
0	0	1	1	6 states + $\overline{\text{WAIT}}$ pin input mode
Other than the above				Reserved

Note 1: For SDRAM, the above settings are not effective. Refer to Section 3.11, SDRAM controller.

Note 2: For NAND flash memory, the above settings are not effective.

(a) Fixed wait-state mode

The bus cycle is completed in the specified number of states. The number of states can be selected from 2 (0 wait state) through 12 (10 wait states), 14 (12 wait states), 18 (16 wait states) and 22 (20 wait states).

(b) $\overline{\text{WAIT}}$ pin input mode

In this mode, the $\overline{\text{WAIT}}$ signal is sampled. A wait state is continued to be inserted while the $\overline{\text{WAIT}}$ signal is sampled active. The minimum bus cycle in this mode is six states. The bus cycle is completed if the $\overline{\text{WAIT}}$ signal is sampled High at the rising edge of SDCLK in the sixth state. The bus cycle is extended as long as the $\overline{\text{WAIT}}$ signal remains active after sixth state.

(5) Recovery cycle (data hold time) control

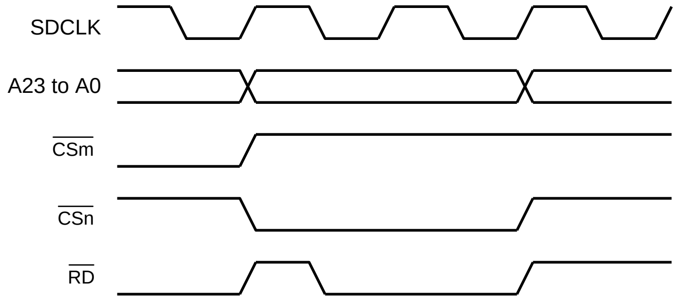
For some memory, the data hold time after when the $\overline{\mathrm{CE}}$ or $\overline{\mathrm{OE}}$ signal is asserted in a read cycle is defined by the AC specification. This may lead to data conflicts. Thus, to avoid this problem, a single dummy cycle can be inserted immediately after an access cycle for the CSm space by setting the BmCSH<BmREC> bit to 1.

This single dummy cycle is inserted when another CS space is accessed in the next bus cycle.

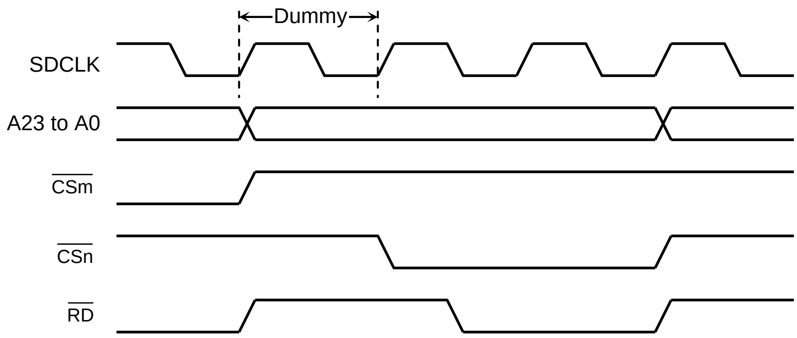
BnCSH<BnREC>

0	No dummy cycle is inserted (Default).
1	Dummy cycle is inserted.

- When no dummy cycle is inserted (0 wait state)



- When a single dummy cycle is inserted (0 wait state)



(6) Timing adjustment function for control signals

This function allows for the timing adjustment of the rising and falling edges of the $\overline{\text{CSn}}$, $\overline{\text{CSZx}}$, $\overline{\text{CSXx}}$, R/$\overline{\text{W}}$, $\overline{\text{RD}}$, $\overline{\text{WRxx}}$, $\overline{\text{SRWR}}$ and $\overline{\text{SRxxB}}$ signals based on the setup and hold time requirements of memories.

As for the $\overline{\text{CSn}}$, $\overline{\text{CSZx}}$, $\overline{\text{CSXx}}$ and R/$\overline{\text{W}}$ signals, and also for the $\overline{\text{WRxx}}$, $\overline{\text{SRWR}}$ and $\overline{\text{SRxxB}}$ signals (generated in a write cycle), their timing can be adjusted for only one CS space. As for the $\overline{\text{RD}}$ and $\overline{\text{SRxxB}}$ signals (generated in a read cycle), their timing can be adjusted individually for each of all CS spaces. As for the CS and EX spaces for which the timing adjustment is not performed, the buses connected to them operate with basic bus timing. (Refer to (7).)

This function can not be used while the BnCSH<BnREC> bit is enabled.

The control signals of SDRAM can be adjusted by setting up the SDRAM controller.

CSTMGCR<TxxSEL1:TxxSEL0>, WRTMGCR<TxxSEL1:TxxSEL0>

00	Change the bus timing for CS0 space
01	Change the bus timing for CS1 space
10	Change the bus timing for CS2 space
11	Change the bus timing for CS3 space

CSTMGCR<TAC1:TAC0>

00	TAC = 0 × 1/f_{SYS} (Default)
01	TAC = 1 × 1/f_{SYS}
10	TAC = 2 × 1/f_{SYS}
11	Reserved

TAC:The delay from A23-A0 to CSn, CSZx, CSXx, R/W.

WRTMGCR<TCWS/H1:TCWS/H0>

00	TCWS/H = 0.5 × 1/f_{SYS} (Default)
01	TCWS/H = 1.5 × 1/f_{SYS}
10	TCWS/H = 2.5 × 1/f_{SYS}
11	TCWS/H = 3.5 × 1/f_{SYS}

TCWS:The delay from CSn to WRxx,SRWR,SRxxB.

TCWH:The delay from WRxx,SRWR,SRxxB to CSn.

RDTMGCR0/1<BnTCRH1:BnTCRH0>

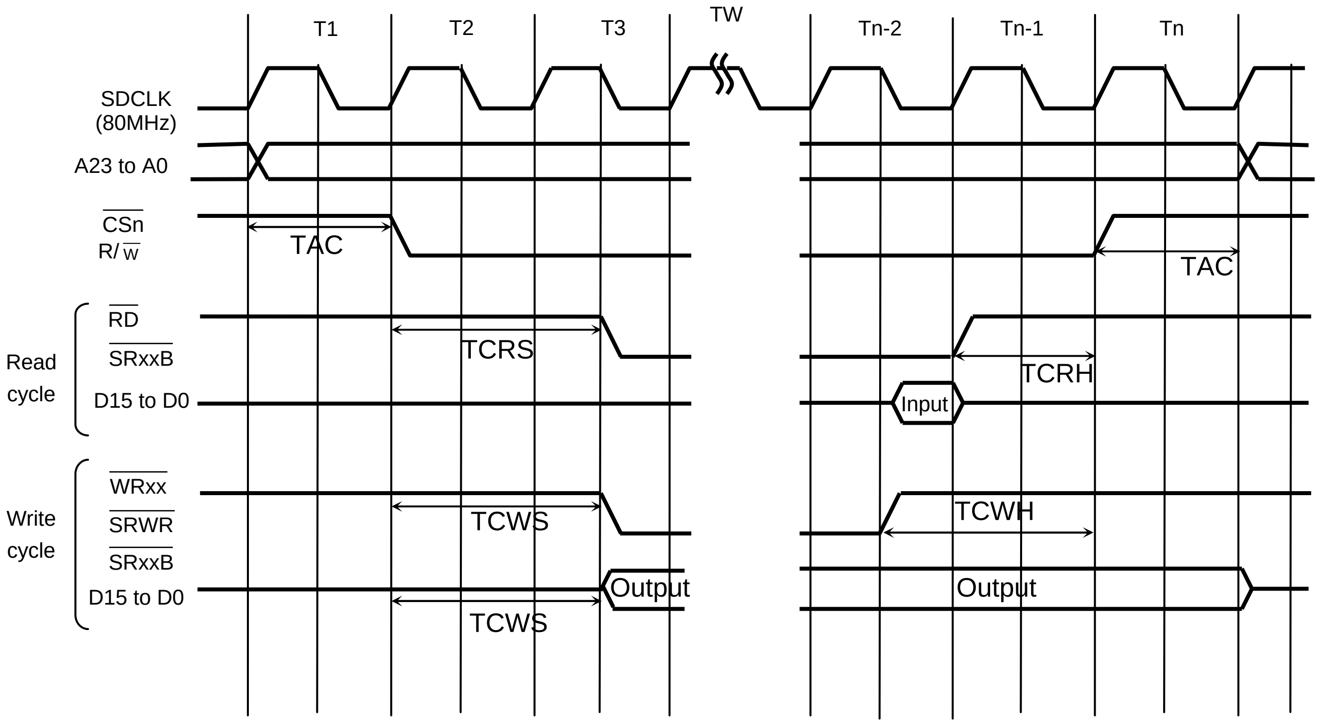
00	TCRH = 0 × 1/f_{SYS} (Default)
01	TCRH = 1 × 1/f_{SYS}
10	TCRH = 2 × 1/f_{SYS}
11	TCRH = 3 × 1/f_{SYS}

TCRH:The delay from RD,SRxxB to CSn.

RDTMGCR0/1<BnTCRS1:BnTCRS0>

00	TCRS = 0.5 × 1/f_{SYS} (Default)
01	TCRS = 1.5 × 1/f_{SYS}
10	TCRS = 2.5 × 1/f_{SYS}
11	TCRS = 3.5 × 1/f_{SYS}

TCRS:The delay from CSn to RD,SRxxB.

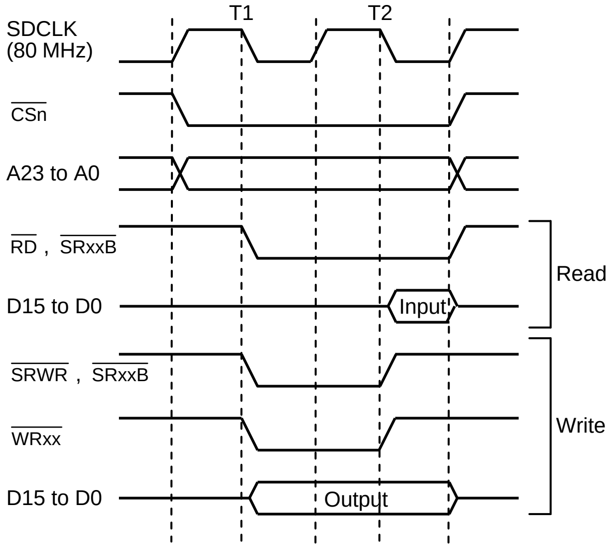


Note1: Wait states (TWs) are inserted as specified by the BnCSL register. No TW is inserted if the number of wait state is specified as zero.

Note2: Above diagram shows case of 16-bit bus access.

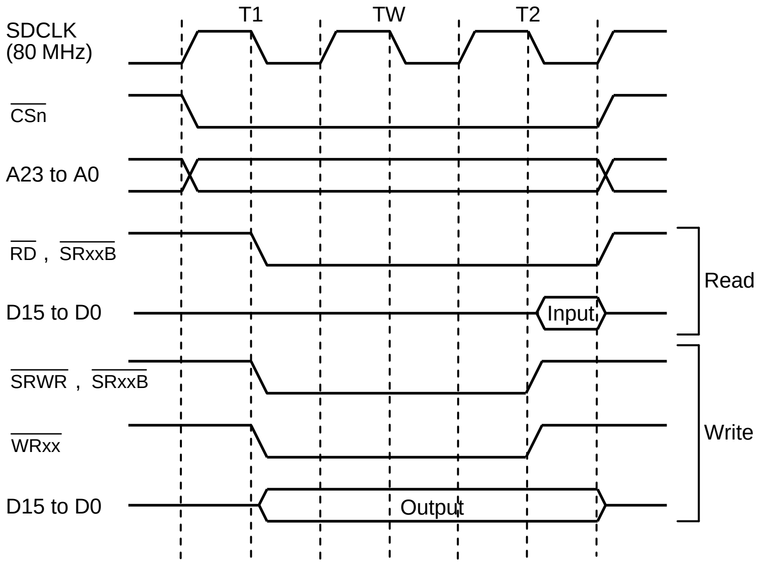
(7) Basic bus timing

(a) External bus read/write cycle (0 wait state)



Note: Above diagram shows case of 16-bit bus access.

(b) External bus read/write cycle (1 wait state)

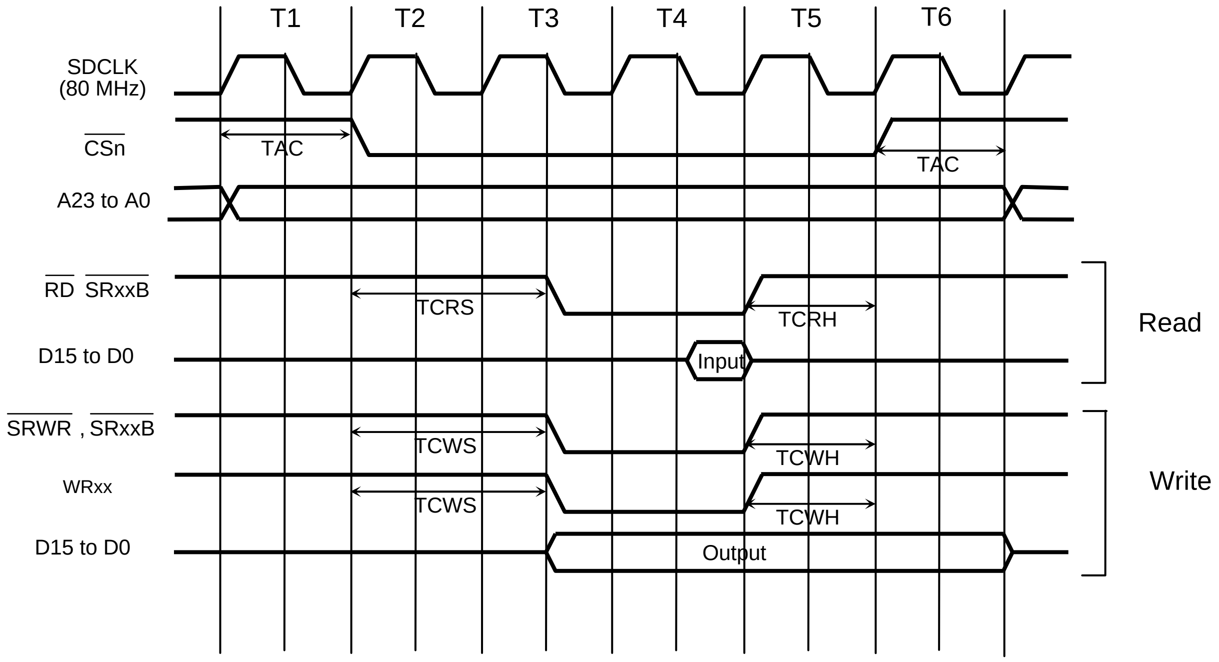


Note: Above diagram shows case of 16-bit bus access.

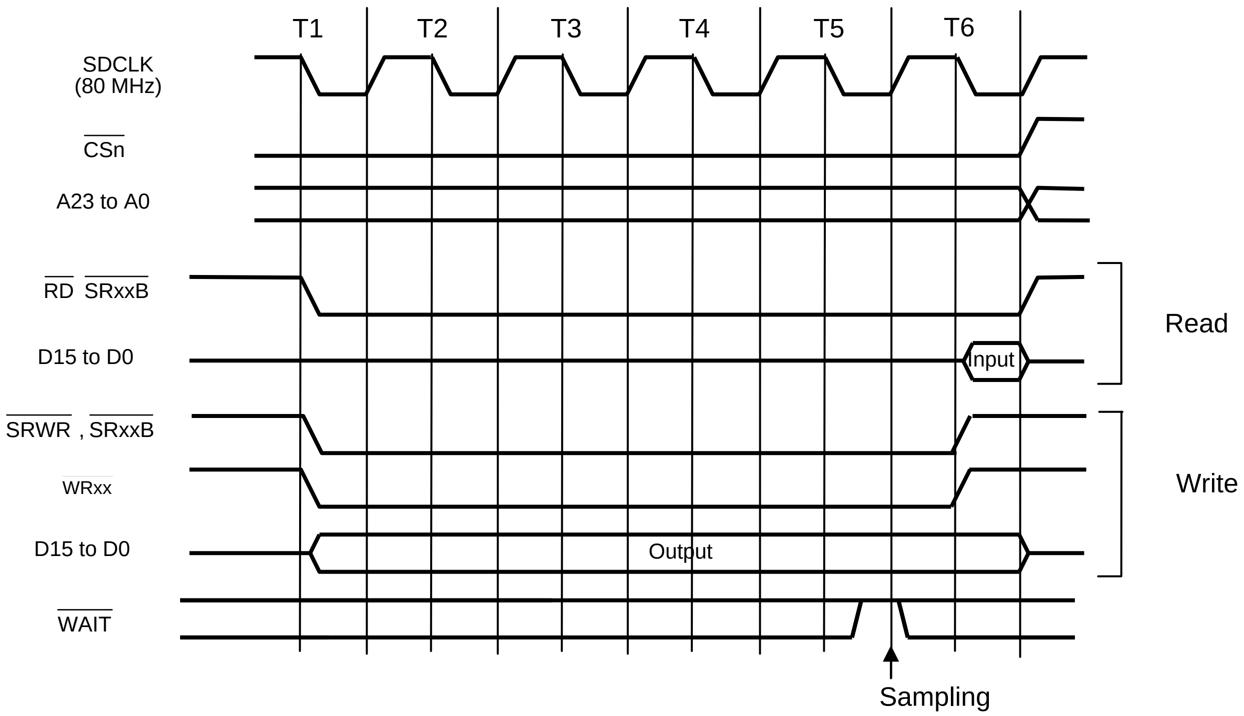
(c) External bus read cycle

(1 wait state + TAC: 1×1/f_{SYS} + TCRS: 1.5×1/f_{SYS} + TCRH: 1 ×1/f_{SYS})

External bus write cycle (1 wait state + TAC: 1×1/f_{SYS} + TCWS/H: 1.5×1/f_{SYS})

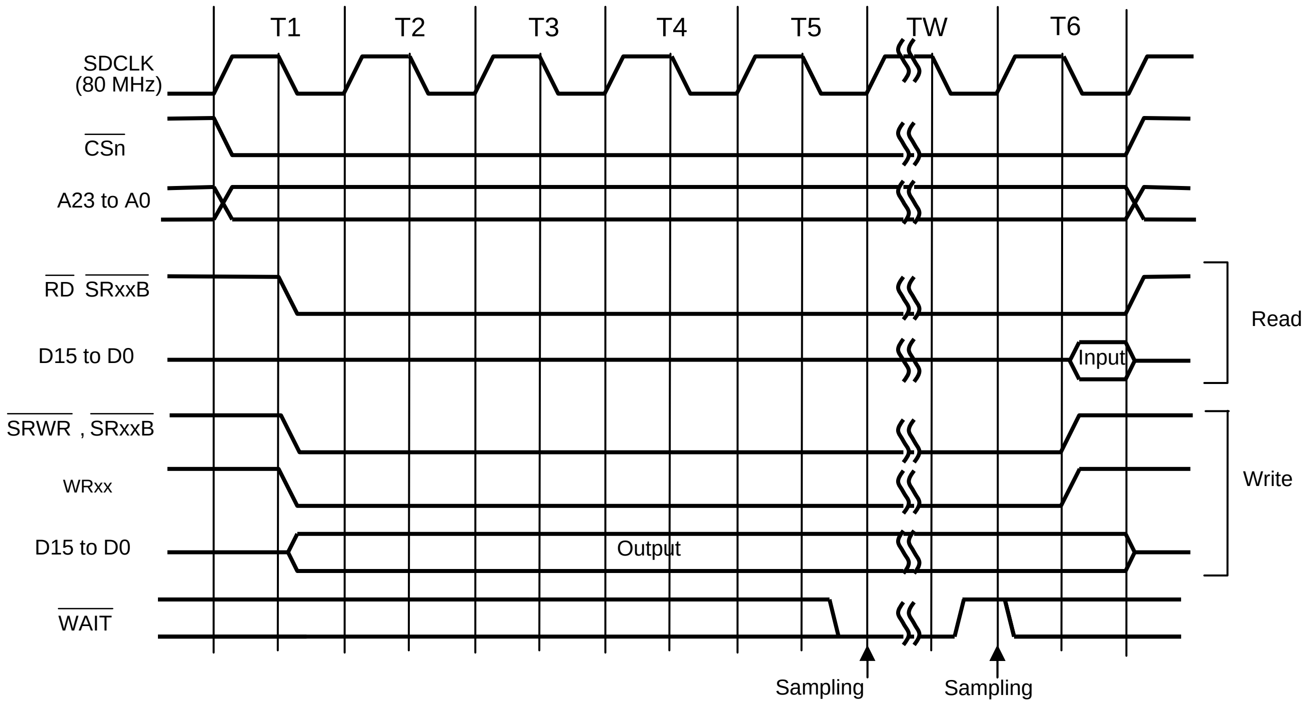


(d) External bus read/write cycle (4 wait states + $\overline{\text{WAIT}}$ pin input mode)



Note: Above diagram shows case of 16-bit bus access.

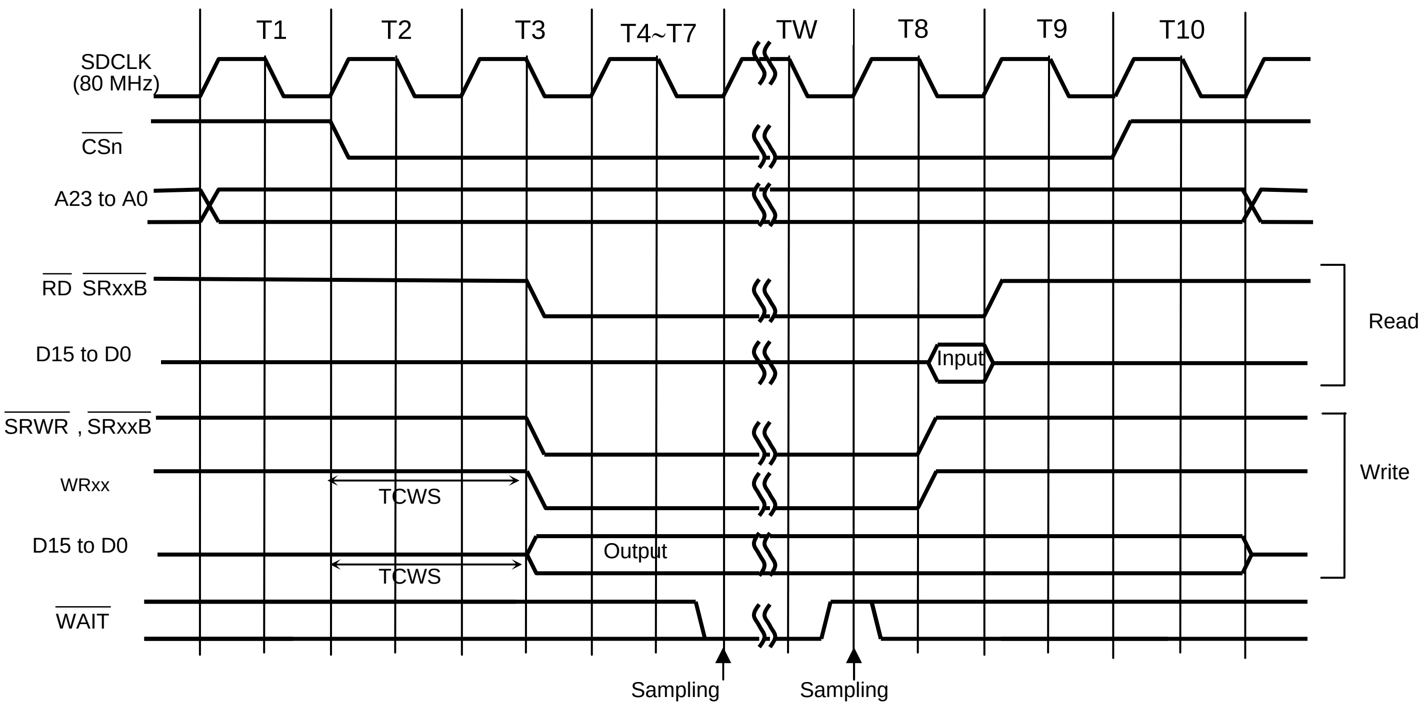
(e) External bus read/write cycle (4 wait states + $\overline{\text{WAIT}}$ pin input mode)



Note: Above diagram shows case of 16-bit bus access.

(f) External bus read cycle (4 wait states + $\overline{\text{WAIT}}$ pin input mode +TAC: 1×1/f_{SYS} + TCRS: 1.5×1/f_{SYS} + TCRH: 1×1/f_{SYS})

External bus write cycle (4 wait states + $\overline{\text{WAIT}}$ pin input mode + TAC: 1×1/f_{SYS} + TCWS/H: 1.5×1/f_{SYS})



Note: Above diagram shows case of 16-bit bus access.

(8) External memory connections

Figure 3.9.4 shows an example of how to connect external 16-bit SRAM and 16-bit NOR flash to the TMP92CF29A.

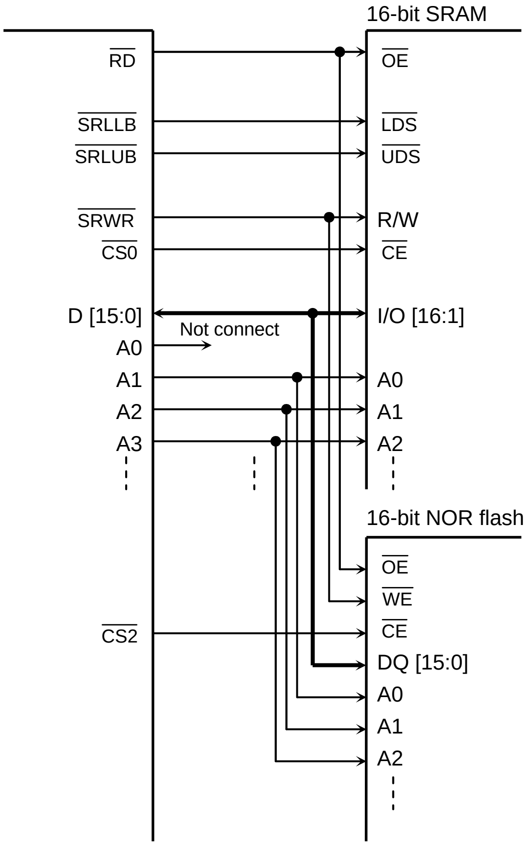


Figure 3.9.4 Example of External 16-Bit SRAM and NOR Flash Connection

3.9.4 Controlling the Page Mode Access to ROM

This section describes page mode access operations to ROM and the required register settings. The page mode operation to ROM is specified by PMEMCR.

(1) Operations and register settings

The TMP92CF29A supports page mode accesses to ROM. Only the CS2 space can be configured for this mode of access. The page mode operation to ROM is specified by the Page ROM Control register, PMEMCR.

Setting the PMEMCR<OPGE> bit to 1 sets the mode of memory access to the CS2 space to page mode.

The number of cycles required for a read cycle is specified by the PMEMCR<OPWR1:OPWR0> bits.

PMEMCR<OPWR1:OPWR0>

<OPWR1>	<OPWR0>	Number of Cycles in Page Mode
0	0	1 cycle (n-1-1-1 mode) (n ≥ 2)
0	1	2 cycles (n-2-2-2 mode) (n ≥ 3)
1	0	3 cycles (n-3-3-3 mode) (n ≥ 4)
1	1	4 cycles (n-4-4-4 mode) (n ≥ 5)

Note: Specify the number of wait states (n) using the control register (BnCSL) for each address space.

The page size (the number of bytes) of ROM as seen from the CPU is determined by PMEMCR<PR1:PR0>. When the specified page boundary is reached, the controller terminates the page read operation. The first data of the next page is read in the normal mode. Then, the following data is read again in page mode.

PMEMCR<PR1:PR0>

<PR1>	<PR0>	ROM Page Size
0	0	64 bytes
0	1	32 bytes
1	0	16 bytes (Default)
1	1	8 bytes

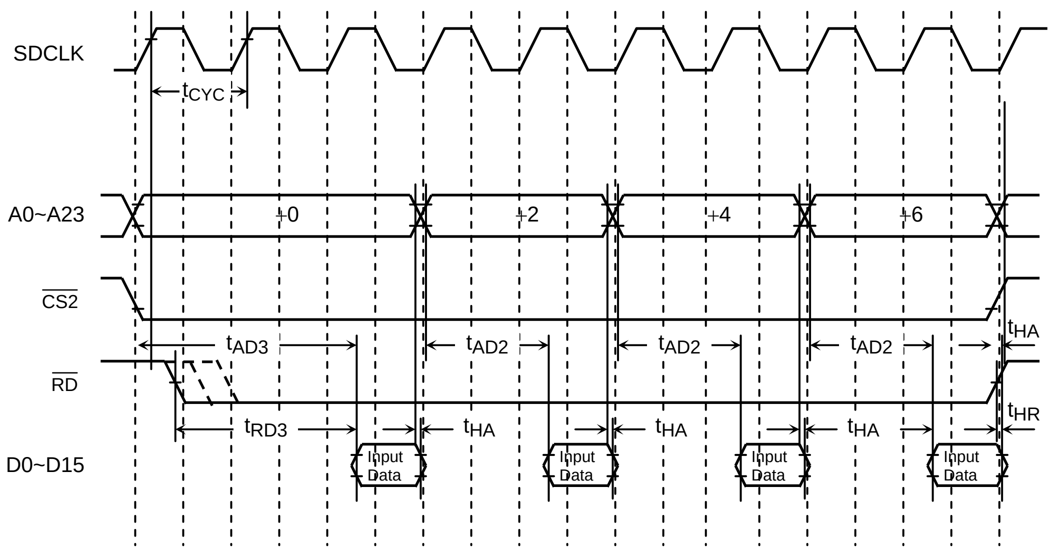


Figure 3.9.5 Page Mode Access Timing (when using a 8-byte page size)

3.9.5 On-Chip ROM Control

This section describes the on-chip ROM.

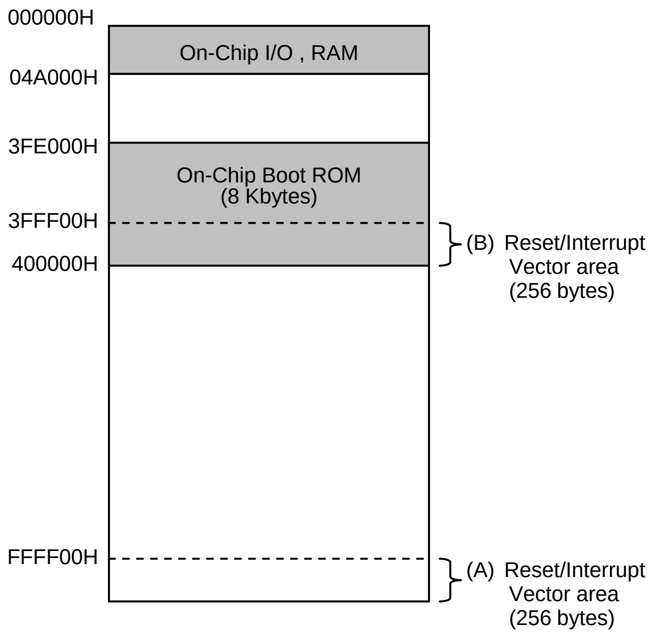
(1) BOOT mode

The TMP92CF29A boots in BOOT mode following the AM1 and AM0 settings upon reset.

AM1	AM0	Start mode
0	0	Don’t use this setting
0	1	Boots from external memory using a 16-bit data bus
1	0	Don’t use this setting
1	1	Boots from the on-chip Boot ROM (32-bit on-chip MROM)

(2) Memory map of the On-Chip ROM

The On-Chip ROM consists of 8-Kbyte masked ROM and is located in the memory area from 3FE000H to 3FFFFFH.



(3) Reset/interrupt address select circuitry

The reset/interrupt vector area is located in the memory area from FFFF00H to FFFFEFH (area (A)) in the TLCS-900/H1.

Since the boot ROM is located in the different area, the TMP92CF29A supports reset/interrupt vector address select circuitry.

In BOOT mode, the reset/interrupt vector area is located in the memory area from 3FFF00H to 3FFFEFH (area (B)). By clearing the BROMCR<VACE> bit to 0 after the boot sequence, the vector area can be remapped to the area (A). Therefore, the area (A) can be used only for the system routine.

This BROMCR<VACE> bit is initialized to 1 in BOOT mode. In any other start mode, this register has no effect.

Note: Since the last 16-byte area (FFFFF0H to FFFFFFH) is reserved for an emulator, this area is not remapped by clearing the BROMCR<VACE> bit.

(4) Bypassing boot ROM

The application system program may continue to run without asserting a reset signal even after completing the boot sequence in BOOT mode. In this case, the external memory area from 3FE000H to 3FFFFFH can not be accessed because the boot ROM already resides in the same area.

To avoid such a situation, the on-chip boot ROM can be bypassed by setting the BROMCR<ROMLESS> bit to 1.

This BROMCR<ROMLESS> bit is initialized to 0 in BOOT mode, while it is initialized to 1 in other start modes.

If this bit has been set to 1, writing a 0 to this bit is prohibited.

BROMCR (016CH)		7	6	5	4	3	2	1	0
	Bit Symbol						CSDIS	ROMLESS	VACE
	Read/Write						R/W		
	Reset State						1	0/1 (note)	1/0 (note)
	Function						Nand_Flash area CS output 0: Enable 1: Disable	Boot ROM 0: Use 1: Bypass	Vector address conversion 0: Disable 1: Enable

Note: Reset states differ depending on start modes.

3.9.6 Notes

(1) Timing for the $\overline{\text{CS}}$ and $\overline{\text{RD}}$ signals

If the load capacitance of the $\overline{\text{RD}}$ (Read) signal line is greater than that of the $\overline{\text{CS}}$ (Chip Select) signal line, the deassertion timing of the read signal is delayed, which may lead to an unintentional extension of a read cycle. Such an unintended read cycle extension, which is indicated as (a) in Figure 3.9.6, may cause a problem.

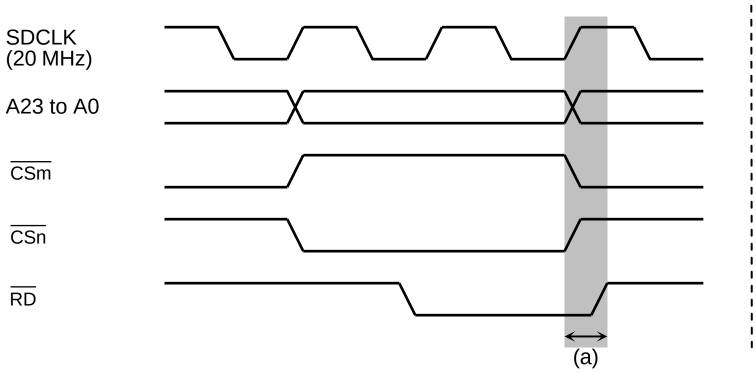


Figure 3.9.6 Read Cycle of When the Read Signal is Delayed

Example: When using an externally connected NOR flash whose commands are compatible with the standard JEDEC commands, the toggle bit may not be read correctly. If the rising edge of the read signal in the cycle immediately preceding the NOR flash access cycle does not occur in time, a read cycle may be extended unintentilnally as indicated as (b) in Figure 3.9.7.

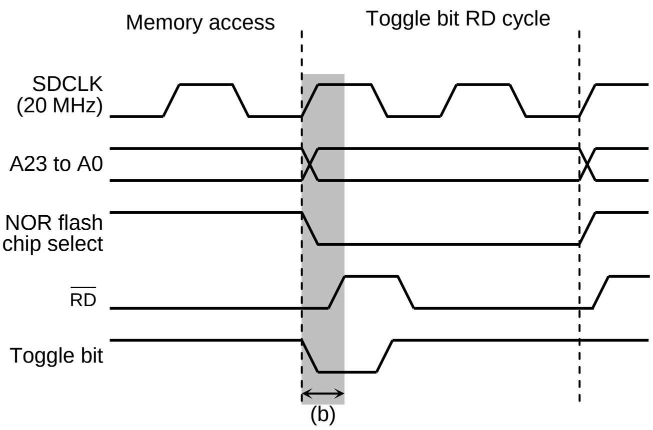


Figure 3.9.7 NOR Flash Toggle Bit Read Cycle

When the toggle bit is inverted due to this unexpected read cycle extension, the CPU cannot read the toggle bit properly and it always reads the same value from the toggle bit.

To avoid this situation, it is recommended to perform data polling or to use the timing adjustment function for the rising edge of the $\overline{\text{RD}}$ signal (RDTMGCRn <BnTCRH1:BnTCRH0>).

(2) Setting up the NAND flash area

Figure 3.9.8 shows a memory map for the NAND flash memory.

Since it is recommended that the CS3 space be located in the memory area from 000000H to 3FFFFFH, the following description is provided for such condition.In this case, the NAND flash area overlaps with the CS3 space. However, the $\overline{CS3}$ pin is not asserted by setting the BROMCR<CSDIS> bit to 1. Likewise, the $\overline{CS0}$ through $\overline{CS3}$ pins, the $\overline{CSXA}$ through $\overline{CSXB}$ pins and the $\overline{CSZA}$ through $\overline{CSZD}$ pins are not asserted either.

Note 1: In the above setting, 296 Kbytes out of the memory area for the CS3 (000000H to 049FFFH) cannot be used.

Note 2: The 16-byte area (001FF0H to 001FFFH) is predefined asNAND Flash area as shown below regardless of which CS space is selected. Therefore, the setting of the CS3 space does not affect the NAND flash area.

(NAND-Flash area specification)

1. Bus width : Specified by NDFMCR1<BUSW> in the NAND Flash controller.
2. Wait control : Specified by NDFMCR<SPLW1:SPLW0> and NDFMCR<SPHW1:SPHW0> in the NAND Flash controller

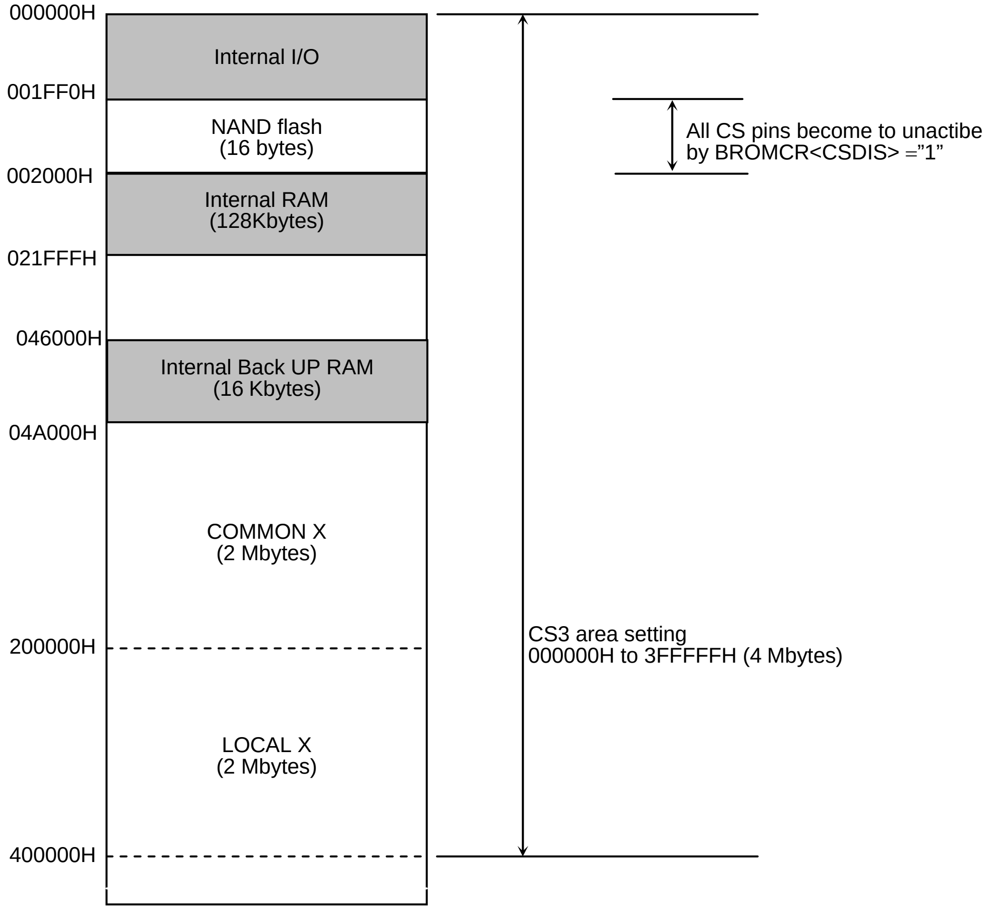


Figure 3.9.8 Recommended CS3 Space Assignment

(3) Setting up the NAND flash area

In case of using SDRAM (SDCS) and NAND flash together, the BROMCR<CSDIS> bit cannot be used. This section provides an example of such cases.

It is recommended that the memory area from 000000H to 3FFFFFH be assigned to the CS2 or CS1 (SDCS) space. A detailed description is provided below.

In this case, the NAND flash area overlaps with the CS2 or CS1 (SDCS) space.

So, if a program accesses NAND flash, the CS2 or CS1 space and NAND flash space are accessed at the same time, which leads to problems such as a data conflict.

To avoid this, it is recommended that the 32-Kbyte memory area from 000000H to 007FFFH be assigned to the CS0 space. (The $\overline{\text{CS0}}$ pin is not required.)

Since the CS0 setting has higher priority over the settings of the CS2 and CS1 spaces, only NAND flash will be accessed without causing data conflicts.

Note: In this case, the 32-Kbyte memory area from 000000H to 007FFFH within the SDCS space cannot be used.

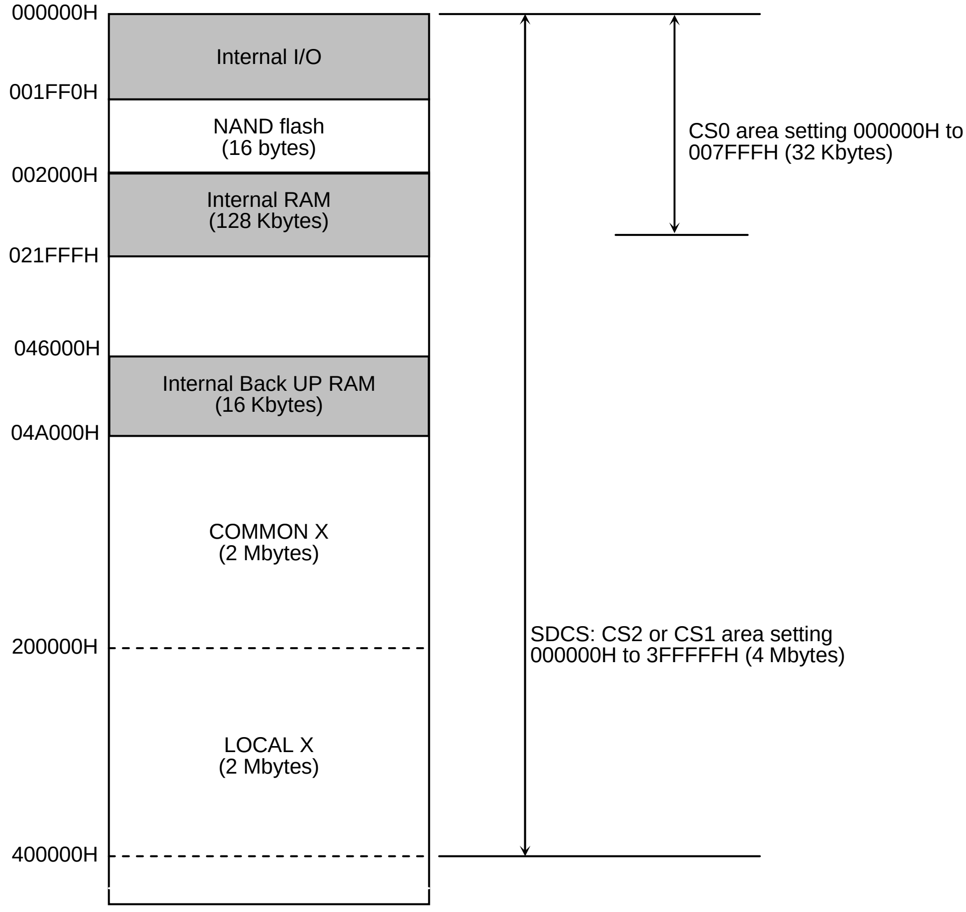


Figure 3.9.9 Recommended Assignment for the SDCS and CS0 Spaces

3.10 External Memory Extension (MMU)

The MMU allows for memory expansion by providing three local memory areas, the MMU function allows for the expansion of the program/data area to 2.1Gbytes.

For recommended address memory maps, refer to Figure 3.10.1.

However, when the amount of memory being used is less than 16 Mbytes, it is not necessary to configure the MMU register. For such cases, please refer to the section on the Memory controller.

A memory area which can be configured into banks is called the LOCAL area. The address range assigned to the LOCAL area is predefined and cannot be changed.

And the rest of the memory area is called the COMMON area.

Basically, a series of program routines should be stored entirely within one bank. The program execution cannot be branched between different banks of the same LOCAL area using the JP instruction. For more details, refer to the following programming examples.

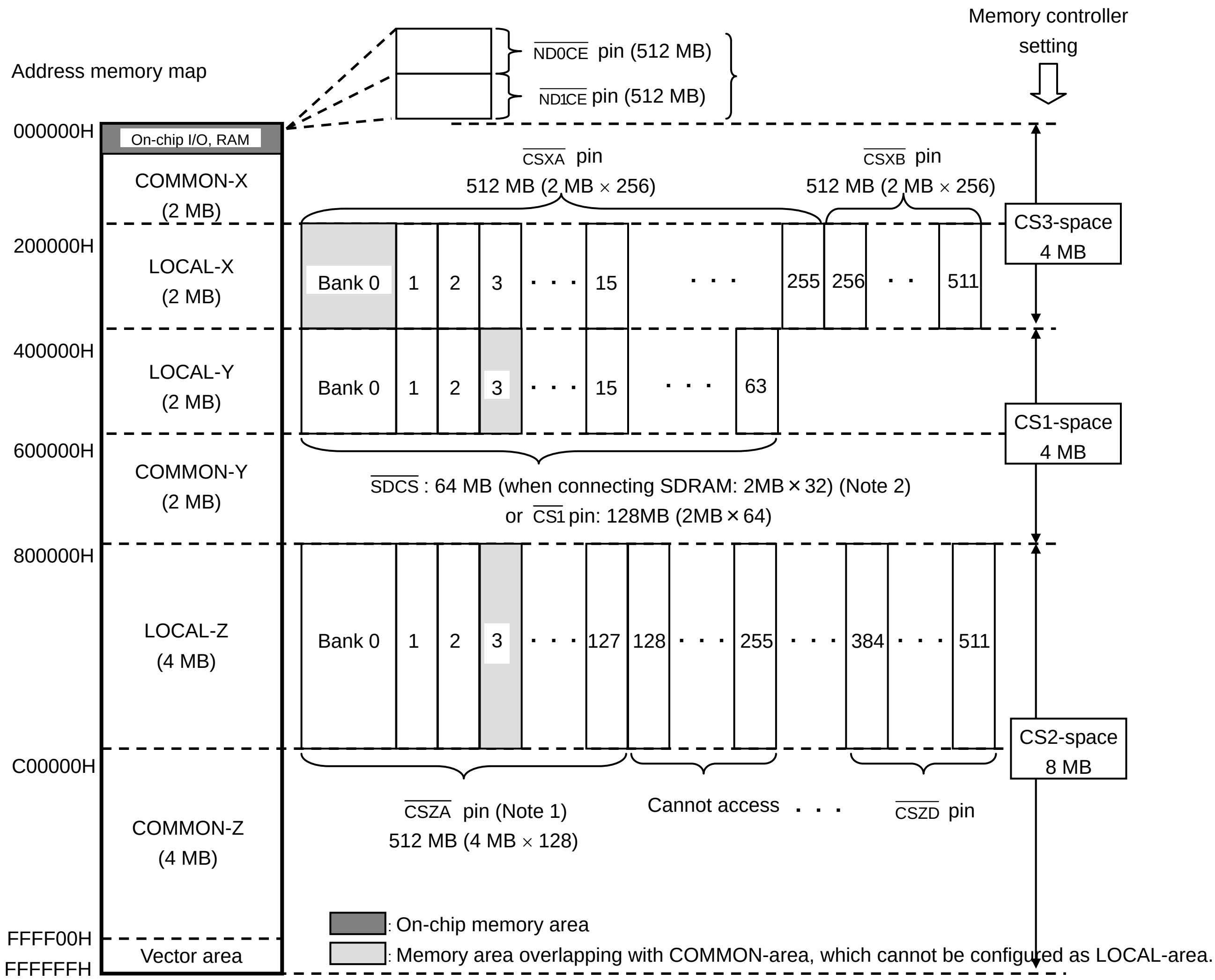
The TMP92CF29A has the following external pins for connecting external memory.

- Address bus: EA28, EA27, EA26, EA25, EA24 and A23 to A0
- Chip Select: $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, $\overline{\text{CSXA}}$ to $\overline{\text{CSXB}}$, $\overline{\text{CSZA}}$ to $\overline{\text{CSXD}}$, $\overline{\text{SDCS}}$, $\overline{\text{ND0CE}}$ and $\overline{\text{ND1CE}}$
- Data bus: D15 to D0

Note: This device is a subset microcontroller of 900H1 series microcontroller: TMP92CZ26AXBG and TMP92CF26A. The total memory size of this device was cut from 3.1GByte to 2.1G byte because number of pins was cut, and BANK of Z-area was cut from 512 banks to 256 banks.

3.10.1 Recommended Memory Map

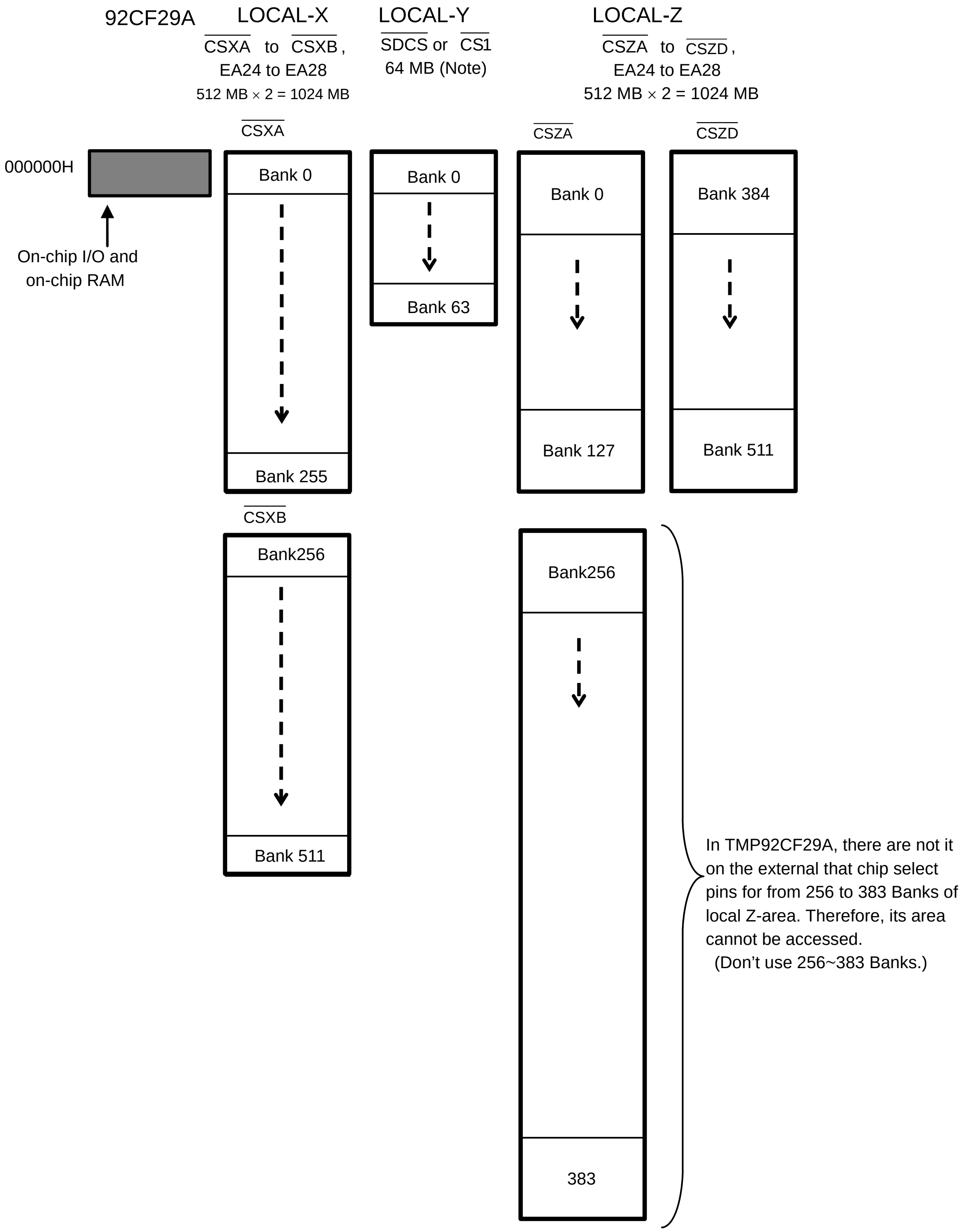
Figure 3.10.1 shows one of recommended address memory maps. This is an example of when memory is expanded to the maximum size.



Note1: CSZA is a chip-select signal for not only bank 0 through bank 127 of the LOCAL-Z area, but also for the COMMON-Z area.

Note2: In case of connecting SDRAM to the Y-area, the maximum expanded memory size is 64 MB (2 MB × 32).

Figure 3.10.1 Recommended Memory Map for the Maximum Expansion (Logical address)



Note: In case of connecting SDRAM to the Y-area, the maximum expanded memory size is 64MB (2MB×32).

Figure 3.10.2 Recommended Memory Map for the Maximum Expansion (Physical address)

3.10.2 Control registers

The TMP92CF29A MMU has 24 registers. These registers are used for storing eight types of data (program, read data, write data, LCD-display data, source data for DMA channels of odd/even number, destination-data for DMA channels of odd/even number) for each of three-LOCAL areas (LOCAL-X through LOCAL-Z). These registers allow for easy data access.

(How to use the control registers)

First, load the control registers for each LOCAL area with the desired bank number and enable/disable the specified bank. Then, configure the external pins to be used and also the Memory Controller. Then, when the CPU or LCDC accesses a logical address in the LOCAL area, the MMU translates the logical address to the corresponding physical address according to the programmed bank configuration. The physical address is then placed on the external address bus pin, which enables external memory accesses. Thus, even when a program accesses the same logical address, its physical address changes depending on the bank specified by the program bank register. This enables memory accesses to the different memory banks.

Note1: When programming the bank registers, the bank area that is overlapping with the COMMON area must not be specified (because addresses of those areas are converted to the same physical addresses).

Note2: In the LOCAL area, changing Program bank number (LOCALPX, Y or Z) is disabled. Program bank setting of each LOCAL area must change in COMMON area. (But bank setting of data-Read, data-Write and LCDC-display data can change also in LOCAL area.)

Note3: After setting values specifying the data bank number into bank registers for the read, write, DMA and LCD display data (LOCALRn, LOCALWn or LOCALLn, LOCALESn, LOCALEDn, LOCALOSn, LOCALODn; the symbol “n” indicates X, Y or Z), the specified bank requires a certain setup time to be enabled. Thus, the bank cannot be accessed by an instruction immediately following the register setting instructions. In this case, insert a dummy instruction which accesses SFR or another memory area as shown in the following example.

(Example)

```
ld      xix, 200000h         ;
ldw     (localrx), 8001h     ;  Specify the read-data bank number
ldw     wa, (localrx)        ;  ← Inserted dummy instruction which accesses SFR
------------------------------------------------------------------------------
ldw     wa, (xix)            ;  instruction which reads bank 1 of the LOCAL-X area.
```

Note4: When the LOCAL-Z area is used, pin P82 should be assigned as the chip select signal $\overline{CSZA}$. In this case, $\overline{CSZA}$ works as the chip select signal for the bank 0 through the bank 15, and also for the COMMON-Z area. After reset, pin P82 should be properly configured following the procedure below.

```
ldw     (localpz), 8000h      ; Enable the banks in LOCAL-Z for program
ldw     (localrz), 8000h      ; Enable the banks in LOCAL-Z for read data
ldw     (localwz), 8000h      ; Enable the banks in LOCAL-Z for write data (*1)
ldw     (locallz), 8000h      ; Enable the banks in LOCAL-Z for LCD display memory
                                (*2)
ld      (P8FC),  -----0--B    ; Assign P82 as the CSZA output
ld      (P8FC2), -----1--B    ;
```

(*1) This setting is not required if the COMMON-Z area is not used to store write data.

(*2) This setting is not required if the COMMON-Z area is not used to store display data for LCD.

3.10.2.1 Program bank registers

These registers should be loaded with bank number values to specify the bank to be used as program memory. As described above, the program execution cannot be directly branched to a different bank in the same LOCAL area. The bank switching within the same LOCAL area is prohibited.

LOCAL-X Register for Program

		7	6	5	4	3	2	1	0
LOCALPX (0880H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(0881H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for Program

		7	6	5	4	3	2	1	0
LOCALPY (0882H)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(0883H)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for Program

		7	6	5	4	3	2	1	0
LOCALPZ (0884H)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(0885H)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

3.10.2.2 LCD Display Data Bank Registers

These registers should be loaded with bank number values to specify the bank to be used as LCD display data memory. Since the data bank registers for CPU and LCDC are prepared independently, the banks that are accessed by the CPU (for program, read and write data) can be switched while the LCD display is on.

LOCAL-X Register for LCD Data

		7	6	5	4	3	2	1	0
LOCALLX (0888H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(0889H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for LCD Data

		7	6	5	4	3	2	1	0
LOCALLY (088AH)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(088BH)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for LCD Data

		7	6	5	4	3	2	1	0
LOCALLZ (088CH)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(088DH)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

3.10.2.3 Read-Data Bank Registers

These registers should be loaded with bank number values to specify the banks to be used as read-data memory. The following example shows how to specify bank 1 for storing read data in the LOCAL-X area. The instruction, "ldw wa, (xix),"reads the data from the memory location at the address xix and stores it into the wa register of the CPU. When loading the address xix into the read-data bank register, the bank is only enabled upon a data (operand) read operation for the memory location at the address xix.

(Example)

```
ld      xix, 200000h       ;
ld      (localrx), 8001h   ;  Specify the read-data bank number.
ldw     wa, (localrx)      ;  ← Insert a dummy instruction that accesses SFR
-----------------------------------------------------------------------------
ldw     wa, (xix)          ;  Read bank 1 of the LOCAL-X area
```

LOCAL-X Register for Read Data

		7	6	5	4	3	2	1	0
LOCALRX (0890H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(0891H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for Read Data

		7	6	5	4	3	2	1	0
LOCALRY (0892H)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(0893H)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for Read Data

LOCALRZ (0894H)		7	6	5	4	3	2	1	0
	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(0895H)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

3.10.2.4 Write-Data Bank Registers

These registers should be loaded with bank number values to specify the banks to be used as write data memory. The following example shows how to specify bank 1 for storing write data in the LOCAL-X area. The instruction, "ldw (xix), wa," writes the wa register value of the CPU into the memory location at the address xix. When loading the address xix into the read-data bank register, the bank is only enabled upon a data (operand) write operation for the memory location at the address xix.

(Example)

```
ld    xix, 200000h       ;
ld    (localwx), 8001h   ;  Specify the write-data bank number.
ldw   wa, (localwx)      ;  ← Insert a dummy instruction that accesses SFR
-----------------------------------------------------------------------
ldw   (xix), wa          ;  Write to bank 1 of the LOCAL-X area
```

LOCAL-X Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWX (0898H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(0899H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWY (089AH)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(089BH)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for Write Data

		7	6	5	4	3	2	1	0
LOCALWZ (089CH)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(089DH)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

3.10.2.5 DMA-Function Bank Registers

The TMP92CF29A supports not only the read and write operations of the CPU, but also the high-speed data transfer by enabling the internal DMAC to become the bus master. (Please refer to Section 3.7, “DMA Controller”.)

These registers are provided specially for the DMA operation, separately from the bank registers for the CPU and LCDC. Regardless of the settings of the bank registers for program, read and write data of the CPU, the banks to be used as source address memory and destination address memory are specified individually during DMA operations.

The DMAC of the TMP92CF29A supports six channels, and the bank control is performed by dividing those channels into 2 groups. The DMA channels with the even-channel number, 0, 2 and 4, are classified into the E-group (ES and ED groups); while the channels with the odd-channel number, 1 and 3, are classified into the O-group (OS and OD groups). These registers cannot specify bank numbers for each channel, but specifies one bank number for all the channels in the same group.

The following example shows how to specify bank 1 for storing DMA-source addresses in the LOCAL-X area, and also specify bank 2 for storing DMA-destination addresses in the LOCAL-Y area. If the DMA operation for channel 0 is initiated Assume that the source and destination addresses specified by the DMA operation, which is described in Section 3.7, are set into the LOCAL-X and LOCAL-Y areas, respectively. Then, if the DMA operation for channel 0 is initiated, bank 1 in the LOCAL-X area is configured as the source address memory, and bank 2 in the LOCAL-Y area is configured as the destination address memory.

(Example)

```
        ldw     (localesx), 8001h       ;   Specify DMA-source bank number for channel 0
        ldw     (localedy), 8002h       ;   Specify DMA-destination bank number for channel 0

    DMA operation for channel 0 is started
```

LOCAL-X Register for the E-group DMA Source

		7	6	5	4	3	2	1	0
LOCALESX (08A0H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(08A1H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for the E-group DMA Source

		7	6	5	4	3	2	1	0
LOCALESY (08A2H)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(08A3H)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for the E-group DMA Source

		7	6	5	4	3	2	1	0
LOCALESZ (08A4H)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3)							
		15	14	13	12	11	10	9	8
(08A5H)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	BANK for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

LOCAL-X Register for the E-group DMA Destination

LOCALEDX (08A8H)

	7	6	5	4	3	2	1	0
Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							

(08A9H)

	15	14	13	12	11	10	9	8
Bit Symbol	LXE							X8
Read/Write	R/W							R/W
Reset State	0							0
Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for the E-group DMA Destination

LOCALEDY (08AAH)

	7	6	5	4	3	2	1	0
Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
Read/Write			R/W					
Reset			0	0	0	0	0	0
Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					

(08ABH)

	15	14	13	12	11	10	9	8
Bit Symbol	LYE							
Read/Write	R/W							
Reset	0							
Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for the E-group DMA Destination

LOCALEDZ (08ACH)

	7	6	5	4	3	2	1	0
Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							

(08ADH)

	15	14	13	12	11	10	9	8
Bit Symbol	LZE							Z8
Read/Write	R/W							R/W
Reset State	0							0
Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA; 100000000 to 101111111 Setting prohibited 010000000 to 011111111 Setting prohibited; 110000000 to 111111111 CSZD						

LOCAL-X Register for the O-group DMA Source

		7	6	5	4	3	2	1	0
LOCALOSX (08B0H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(08B1H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for the O-group DMA Source

		7	6	5	4	3	2	1	0
LOCALOSY (08B2H)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(08B3H)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	Bank for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for the O-group DMA Source

		7	6	5	4	3	2	1	0
LOCALOSZ (08B4H)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(08B5H)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

LOCAL-X Register for the O-group DMA Destination

		7	6	5	4	3	2	1	0
LOCALODX (08B8H)	Bit Symbol	X7	X6	X5	X4	X3	X2	X1	X0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		15	14	13	12	11	10	9	8
(08B9H)	Bit Symbol	LXE							X8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						

LOCAL-Y Register for the O-group DMA Destination

		7	6	5	4	3	2	1	0
LOCALODY (08BAH)	Bit Symbol			Y5	Y4	Y3	Y2	Y1	Y0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		15	14	13	12	11	10	9	8
(08BBH)	Bit Symbol	LYE							
	Read/Write	R/W							
	Reset State	0							
	Function	BANK for LOCAL-Y 0: Disable 1: Enable							

LOCAL-Z Register for the O-group DMA Destination

		7	6	5	4	3	2	1	0
LOCALODZ (08BCH)	Bit Symbol	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		15	14	13	12	11	10	9	8
(08BDH)	Bit Symbol	LZE							Z8
	Read/Write	R/W							R/W
	Reset State	0							0
	Function	Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

3.10.3 Programming example

The conditions listed in this table apply the following programming examples.

No.	Used as	Memory	Setting	MMU area	Logical address	Physical address
(a)	Main Routine	NOR-Flash (16 MB, 1 pcs)	$\overline{CSZA}$, 16 bit, 1 wait state	COMMON-Z	C00000H to FFFFFFH	
(b)	Character-ROM			Bank 0 in LOCAL-Z	800000H to BFFFFFH	000000H to 3FFFFFH
(c)	Subroutine	SRAM (16 MB, 1 pcs)	$\overline{CS1}$, 16 bit, 0 wait state	Bank 0 in LOCAL-Y	400000H to 5FFFFFH	000000H to 1FFFFFH
(d)	LCD Display-RAM			Bank 1 in LOCAL-Y		200000H to 3FFFFFH
(e)	Stack-RAM	On-chip-RAM (144KB)	– (32 bit, 2-1-1-1clk)	Bank 2 in LOCAL-Y	002000H to 049FFFH	

(a) Main Routine (COMMON-Z)

Logical Address	Physical Address	Instruction No.	Instruction	Comment
		1	org C00000H	;
C00000H	<-(Same)	2	ldw (mamr2),80FFH	; CS2 800000-FFFFFF/8MB
C000xxH	<-	3	ldw (b2csl), C122H	; CS2 16-bit ROM, 1 wait state
		4	ldw (mamr1),40FFH	; CS1 400000-7FFFFF/4MB
		5	ldw (b1csl), 8111H	; CS1 16-bit RAM, 0 wait state
		5.1	ldw (localpz),8000H	; Enable LOCAL-Z bank for program
		5.2	ldw (localrz),8000H	; Enable LOCAL-Z bank for read-data
		6	ld (p8fc), 02H	; P81: $\overline{CS1}$
		7	ld (p8fc2), 04H	; P82:
		9	ld xsp,48000H	; Stack Pointer = 48000H
		10	ldw (localpy),8000H	; Bank 0 in LOCAL-Y is configured as the program bank for subroutines
		11	:	;
C000yyH	<-	12	call 400000H	; Call a subroutine
		13	:	;
		14	:	;
		15	:	;

- The instructions No.2 through No.8 configure external pins and the Memory Controller.
- The instruction No.9 specifies the stack pointer value. The stack pointer is herein specified to point to the memory location in on-chip RAM.
- The instruction No.10 configures the setting used for a subroutine call instruction of No.12.
- The instruction No.12 calls a subroutine. When the CPU generates the address 400000H, the MMU translates it to the physical address 000000H, which is then placed onto the external address bus: A23 to A0. Since the logical address is within the address range of the CS1 space, $\overline{CS1}$ for SRAM is asserted at the same time. By using these instructions, the program execution of the CPU can be branched to the subroutine.

Note: This example assumes that the subroutine program is already written into SRAM.

(b) Subroutine (Bank 0 in LOCAL-Y)

Logical address	Physical address	Instruction No.	Instruction	Comment
		16	org 400000H	;
400000H	000000H	17	ldw (localwy),8001H	; Bank 1 in LOCAL-Y is configured as write-data memory for LCD Display RAM
4000xxH	0000xxH	18	ldw (locally), 8001H	; Bank 1 in LOCAL-Y is configured as LCD display RAM
		19	ldw (localrz), 8001H	; Bank 0 in LOCAL-Z is configured as read-data memory for Character-RAM
		20	ld xiy,800000H	; Index address register for reading Character-ROM
		21	ld wa,(xiy)	; Read Character-ROM
		22	:	; Convert the read data to display-data
		23	~~ld (localpy), 82H~~	;
		24	ld xix, 400000H	; Index address register for writing LCD Display data
		25	ld (xix), bc	; Write LCD Display data
		26	:	; Configure the LCD Controller
		27	:	;
		28	ld xiz, 400000H	; Load the LCD Start address into LCDC
		29	ld (lsarcl), xiz	;
		30	ld (lcdctl0),01H	; Start LCD Display operation
		31	:	;
5000yyH	1000yyH	32	ret	;

- The instructions No.17 and No.18 configure bank 1 of the LOCAL-Y area. In this case, the CPU writes the LCD Display data to Display RAM, and the data is then read by the LCDC. Thus, the LOCALWY and LOCALLY registers should be programmed to specify the same bank, bank1.
- The instruction No.19 configures Bank 0 of the LOCAL-Z area to read data from character-ROM.
- The instructions No.20 and No.21 are used to read data from character-ROM. When the CPU generates the address 800000H, the MMU translates it to the physical address 000000H, which is then placed onto the external address bus: A23 to A0. Since the logical address is within the address range of the CS2 space, $\overline{\text{CSZA}}$ for NOR-Flash is asserted at the same time. By using these instructions, the CPU can read data from character ROM.
- The instruction No.23 switches the program bank in the LOCAL area. Since the program bank switching within the same LOCAL area is prohibited, this is a bad example.
- The instructions No.24 and No.25 are used to write data to SRAM. When the CPU generates the address 400000H, the MMU translates it to the physical address 200000H, which is then placed onto the external address bus: A23 to A0. Since the logical address is within the address range of the CS1 space, $\overline{\text{CS1}}$ for SRAM is asserted at the same time. By using these instructions, the CPU can write data to SRAM.
- The instructions No.28 and No.29 load the LCD starting address into the LCD Controller. When the LCDC generates the address 400000H in a DMA cycle, the MMU translates it to the physical address 200000H, which is then placed onto the external address bus: A23 to A0. Since the logical address is within the address range of the CS1 space, $\overline{\text{CS1}}$ for SRAM is asserted at the same time. By using these instructions, the LCDC can read data from SRAM.
- The instruction No.30 starts LCD display operation.

3.11 SDRAM Controller (SDRAMC)

The TMP92CF29A incorporates an SDRAM controller (SDRAMC) for accessing SDRAM that can be used as data memory, program memory, or display memory.

The SDRAMC has the following features:

(1) Supported SDRAM

Data rate type : SDR (single data rate) type only

Memory capacity : 16 / 64 / 128 / 256 / 512 Mbits

Number of banks : 2 banks / 4 banks

Data bus width : 16 bits

Read burst length : 1 word / full page

Write mode : Single mode / Burst mode

(2) Supported initialization sequence commands

Precharge All command

Eight Auto Refresh commands

Mode Register Set command

(3) Access mode

	CPU Cycle	HDMA Cycle	LCDC Cycle
Burst length	1 word	1 word or full page selectable	Full page
Addressing mode	Sequential	Sequential	Sequential
CAS latency (clock)	2	2	2
Write mode	Single	Single or burst selectable	

(4) Access cycles

- CPU access cycles

  Read cycle : 1 word, 4-3-3-3 states (minimum)

  Write cycle : Single, 3-2-2-2 states (minimum)

  Data size : 1 byte / 1 word / 1 long-word

- HDMA access cycles

  Read cycle : 1 word, 4-3-3-3 states / full page, 4-1-1-1 states (minimum)

  Write cycle : Single, 3-2-2-2 states (minimum) / burst, 2-1-1-1 states (minimum)

  Data size : 1 byte / 1 word / 1 long-word

- LCDC access cycles

  Read cycle : Full page, 4-1-1-1 states (minimum)

  Data size : 1 word

(5) Auto generation of refresh cycles

- Auto Refresh is performed while the SDRAM is not being accessed.
- The Auto Refresh interval is programmable.
- The Self Refresh function is also supported.

Note: The SDRAM address area is determined by the CS1 or CS2 setting of the memory controller. However, the number of bus cycle states is controlled by the SDRAMC.

3.11.1 Control Registers

The SDRAMC has the following control registers.

SDRAM Access Control Register

		7	6	5	4	3	2	1	0
SDACR	Bit symbol	SRDS	–	SMUXW1	SMUXW0	SPRE			SMAC
(0250H)	Read/Write	R/W							R/W
	Reset State	1	0	0	0	0			0
	Function	Read data shift function 0: Disable 1: Enable	Always write "0"	Address multiplex type 00: Type A (A9-) 01: Type B (A10-) 10: Type C (A11-) 11: Reserved		Read/Write commands 0: Without auto precharge 1: With auto precharge			SDRAM controller 0: Disable 1: Enable

SDRAM Command Interval Setting Register

		7	6	5	4	3	2	1	0
SDCISR	Bit symbol		STMRD	STWR	STRP	STRCD	STRC2	STRC1	STRC0
(0251H)	Read/Write		R/W						
	Reset State		1	1	1	1	1	0	0
	Function		TMRD 0: 1 CLK 1: 2 CLK	TWR 0: 1 CLK 1: 2 CLK	TRP 0: 1 CLK 1: 2 CLK	TRCD 0: 1 CLK 1: 2 CLK	TRC 000: 1 CLK 100: 5 CLK 001: 2 CLK 101: 6 CLK 010: 3 CLK 110: 7 CLK 011: 4 CLK 111: 8 CLK		

SDRAM Refresh Control Register

		7	6	5	4	3	2	1	0
SDRCR	Bit symbol	–			SSAE	SRS2	SRS1	SRS0	SRC
(0252H)	Read/Write	R/W			R/W				
	Reset State	0			1	0	0	0	0
	Function	Always write "0"			Self Refresh auto exit function 0:Disable 1:Enable	Refresh interval 000: 47 states 100: 468 states 001: 78 states 101: 624 states 010: 156 states 110: 936 states 011: 312 states 111: 1248 states			Auto Refresh 0:Disable 1:Enable

SDRAM Command Register

SDCMM (0253H)		7	6	5	4	3	2	1	0
	Bit symbol						SCMM2	SCMM1	SCMM0
	Read/Write						R/W		
	Reset State						0	0	0
	Function						Command issue (Note 1) (Note 2) 000: Don't care 001: Initialization sequence a. Precharge All command b. Eight Auto Refresh commands c. Mode Register Set command 010: Precharge All command 100: Reserved 101: Self Refresh Entry command 110: Self Refresh Exit command Others: Reserved		

Note 1: <SCMM2:0> is automatically cleared to "000" after the specified command is issued. Before writing the next command, make sure that <SCMM2:0> is "000". In the case of the Self Refresh Entry command, however, <SCMM2:0> is not cleared to "000" by execution of this command. Thus, this register can be used as a flag for checking whether or not Self Refresh is being performed.

Note 2: The Self Refresh Exit command can only be specified while Self Refresh is being performed.

SDRAM HDMA Burst Length Select Register

SDBLS (0254H)		7	6	5	4	3	2	1	0
	Bit symbol			SDBL5	SDBL4	SDBLS	SDBL2	SDBL1	SDBL0
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			For HDMA5	For HDMA4	For HDMA3	For HDMA2	For HDMA1	For HDMA0
				HDMA burst length 0: 1 Word read / Single write 1: Full page read / Burst write					

Figure 3.11.1 Control Registers

3.11.2 Operation Description

(1) Memory access control

The SDRAMC is enabled by setting SDACR<SMAC> to “1”.

When one of the bus masters (CPU, LCDC, DMAC) generates a cycle to access the SDRAM address area, the SDRAMC outputs SDRAM control signals.

Figure 3.11.2 to Figure3.11.5 shows the timing for accessing the SDRAM. The number of SDRAM access cycles is controlled by the SDRAMC and does not depend on the number of waits controlled by the memory controller.

(a) Command issue function

The SDRAMC issues commands as specified by the SDCMM register. The SDRAMC also issues commands automatically for each SDRAM access cycle generated by each bus master.

Table 3.11.1 shows the commands that are issued by the SDRAMC.

Table 3.11.1 Commands Issued by the SDRAMC

Command	CKEn-1	CKEn	SDxxDQM	A10	A15-11 A9-0	$\overline{\text{SDCS}}$	$\overline{\text{SDRAS}}$	$\overline{\text{SDCAS}}$	$\overline{\text{SDWE}}$
Bank Activate	H	H	H	RA	RA	L	L	H	H
Precharge All	H	H	H	H	X	L	L	H	L
Read	H	H	L	L	CA	L	H	L	H
Read with Auto Precharge	H	H	L	H	CA	L	H	L	H
Write	H	H	L	L	CA	L	H	L	L
Write with Auto Precharge	H	H	L	H	CA	L	H	L	L
Mode Register Set	H	H	H	L	M	L	L	L	L
Burst Stop	H	H	H	X	X	L	H	H	L
Auto Refresh	H	H	H	X	X	L	L	L	H
Self Refresh Entry	H	L	H	X	X	L	L	L	H
Self Refresh Exit	L	H	H	X	X	H	H	H	H

Note 1: H = High level, L = Low level, RA = Row address, CA = Column address, M = Mode data, X = Don’t care

Note 2: CKE$_n$ = CKE level in the command input cycle

CKE$_{n-1}$ = CKE level in a cycle immediately before the command input cycle

(b) Address multiplex function

In access cycles, the A0 to A15 pins output low/column multiplexed addresses. The multiplex width is set by SDACR<SMUXW1:0>. Table 3.11.2 shows the relationship between the multiplex width and low/column addresses.

Table 3.11.2 Address Multiplex

92CF29A Pin Name	SDRAM Access Cycle Address			
	Row Address			Column Address
	Type A <SMUXW> = 00	Type B <SMUXW> = 01	Type C <SMUXW> = 10	
A0	A9	A10	A11	A1
A1	A10	A11	A12	A2
A2	A11	A12	A13	A3
A3	A12	A13	A14	A4
A4	A13	A14	A15	A5
A5	A14	A15	A16	A6
A6	A15	A16	A17	A7
A7	A16	A17	A18	A8
A8	A17	A18	A19	A9
A9	A18	A19	A20	A10
A10	A19	A20	A21	AP *
A11	A20	A21	A22	Row Address
A12	A21	A22	A23	
A13	A22	A23	EA24	
A14	A23	EA24	EA25	
A15	EA24	EA25	EA26	

*AP: Auto Precharge

(c) Burst length

When the CPU accesses the SDRAM, the burst length is fixed to 1-word read/single write. When the LCDC accesses the SDRAM, the burst length is fixed to full page.

The burst length can be selected for SDRAM read and write accesses by HDMA if the following conditions are satisfied:

- The HDMA transfer mode is an increment mode.
- Transfers are made between the SDRAM and internal RAM or internal I/O.

In other cases, HDMA operation can only be performed in 1-word read/single write mode. Use SDBLS<SDBL5:0> to set the burst length for each HDMA channel.

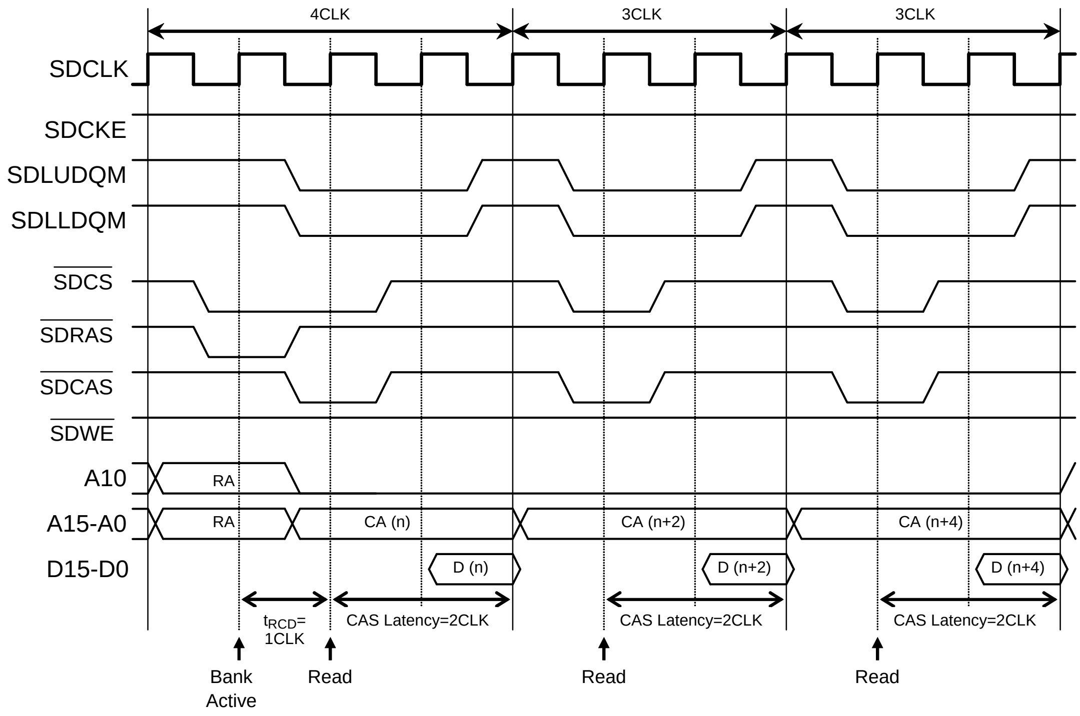


Figure 3.11.2 1-Word Read Cycle Timing

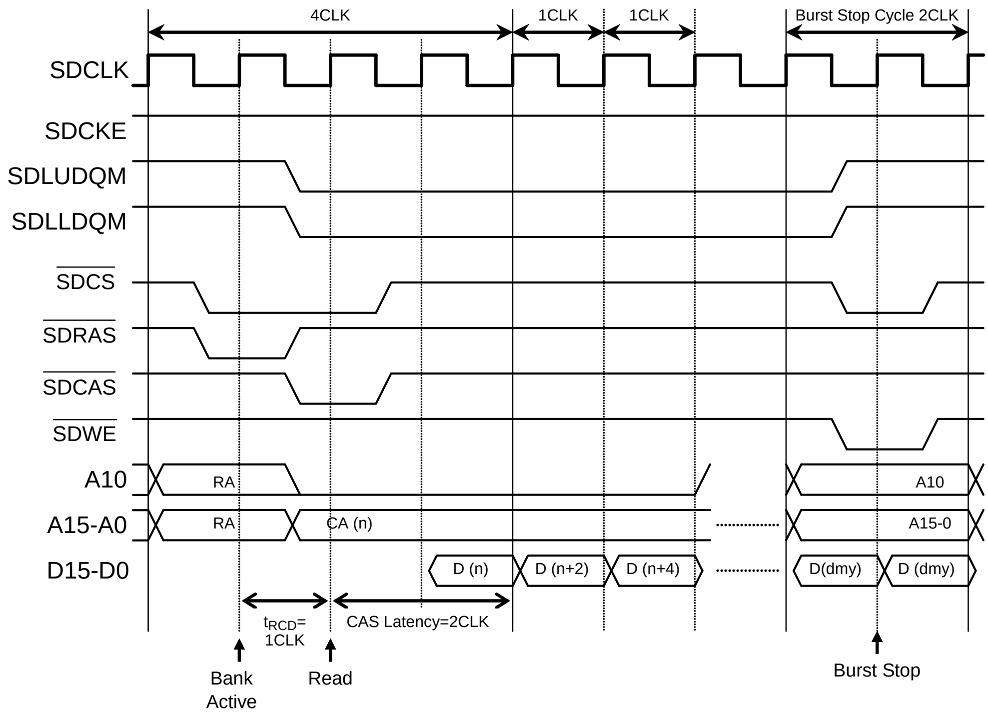


Figure 3.11.3 Full-Page Read Cycle Timing

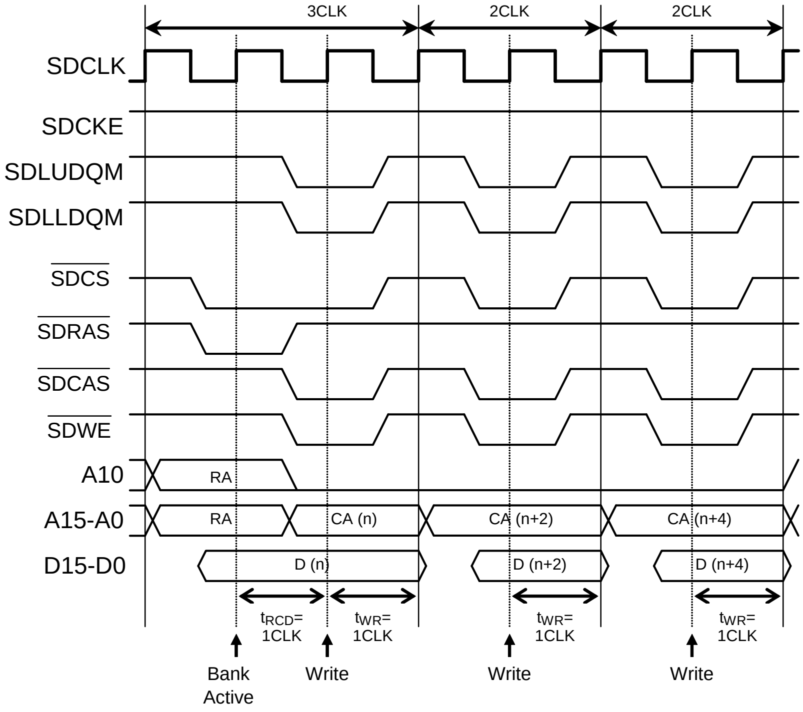


Figure 3.11.4 Single Write Cycle Timing

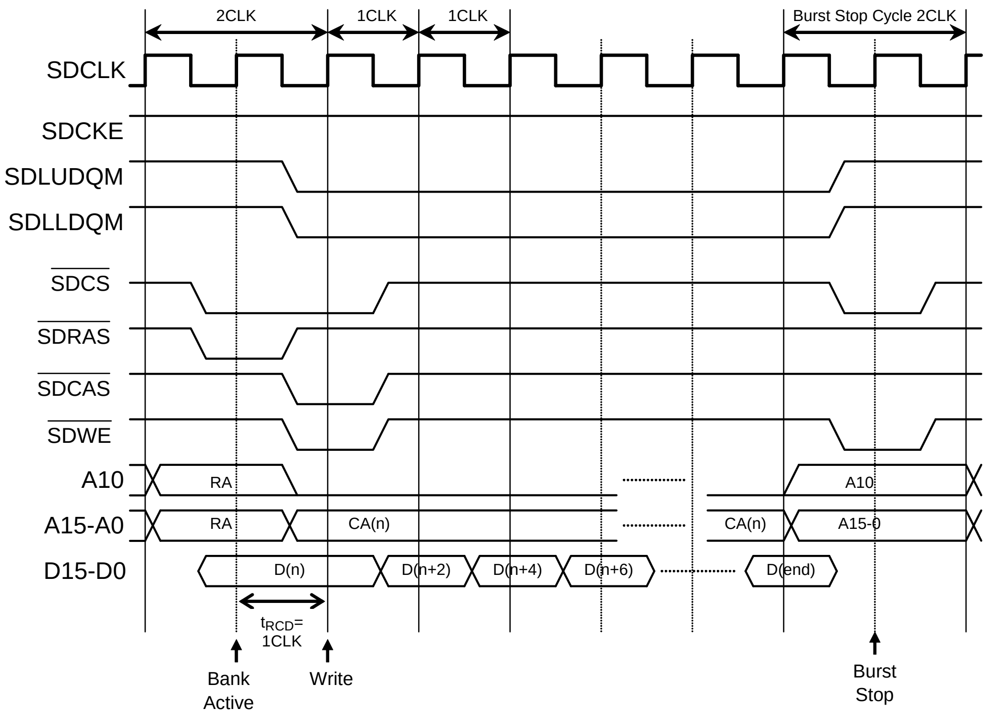


Figure3.11.5 Burst Write Cycle Timing

(2) Execution of instructions on SDRAM

The CPU can execute instructions that are stored in the SDRAM. However, the following operations cannot be performed.

a) Executing the HALT instruction

b) Changing the clock gear setting

c) Changing the settings in the SDACR, SDCMM, and SDCISR registers

These operations, if needed, must be executed by branching to other memory such as internal RAM.

(3) Command interval adjustment function

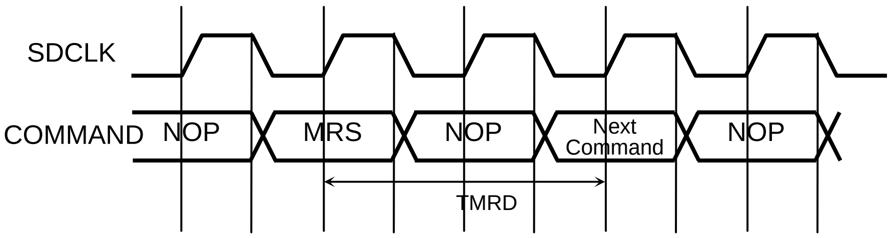
Command execution intervals can be adjusted for each command. This function enables the SDRAM to be accessed at optimum cycles even if the operation frequency is changed by clock gear.

Command intervals should be set in the SDCISR register according to the operating frequency of the TMP92CF29A and the AC specifications of the SDRAM.

The SDCICR register must not be changed while the SDRAM is being accessed.

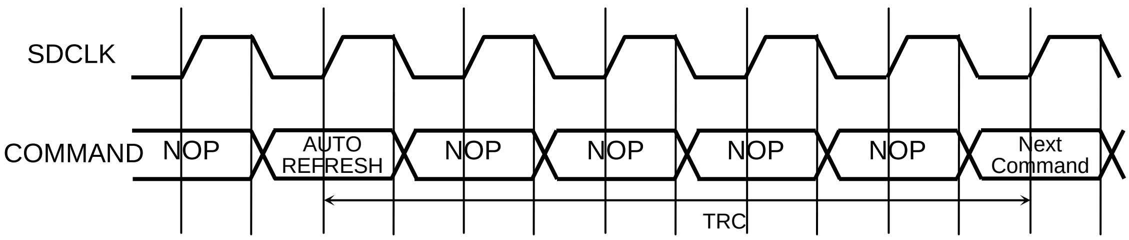
The timing waveforms for various cases are shown below.

(a) Mode Register Set command



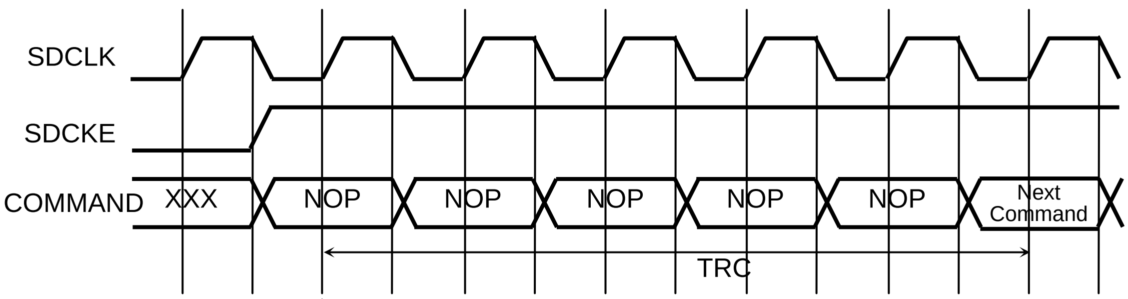
*TMRD=2CLK (SDCISR<STMRD>= "1")

(b) Auto Refresh command



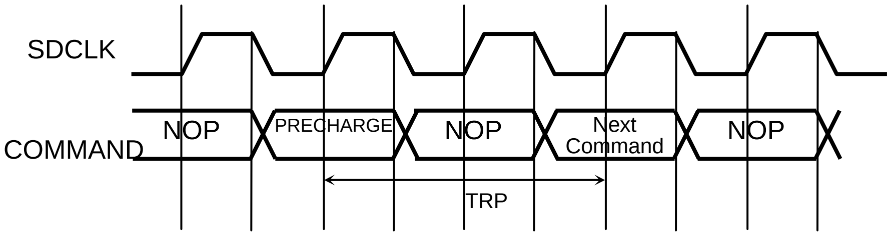
*TRC=5CLK (SDCISR<STRC2:0>= "100")

(c) Self Refresh Exit



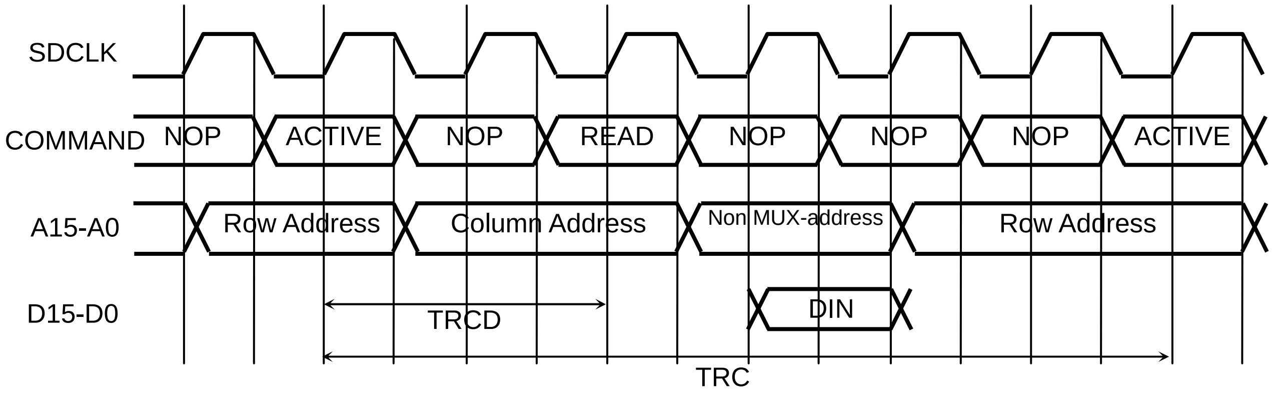
*TRC=5CLK (SDCISR<STRC2:0>= "100")

(d) Precharge command



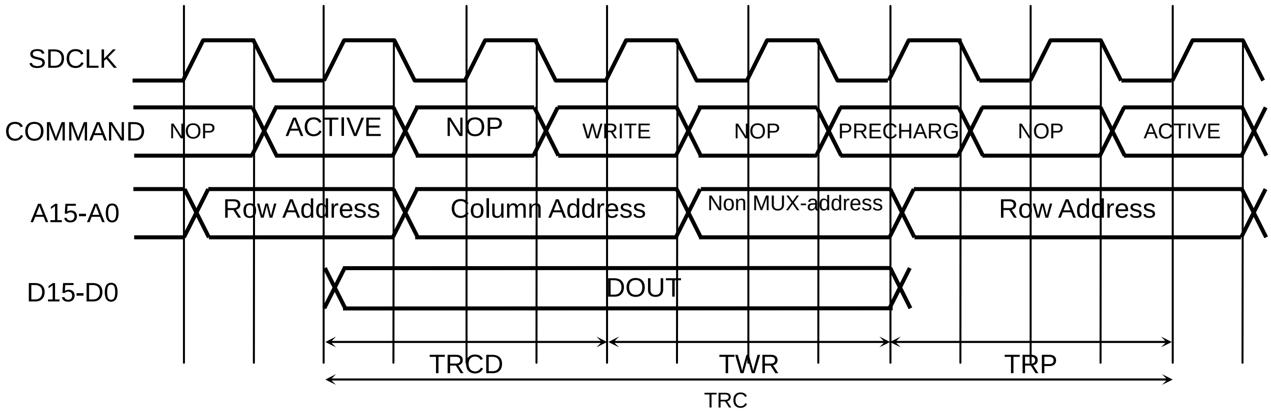
*TRP=2CLK (SDCISR<STRP>= "1")

(e) Read cycle



*TRCD=2CLK (SDCISR<STRCD>= "1")
*TRC=6CLK (SDCISR<STRC2:0>= "101")

(f) Write cycle

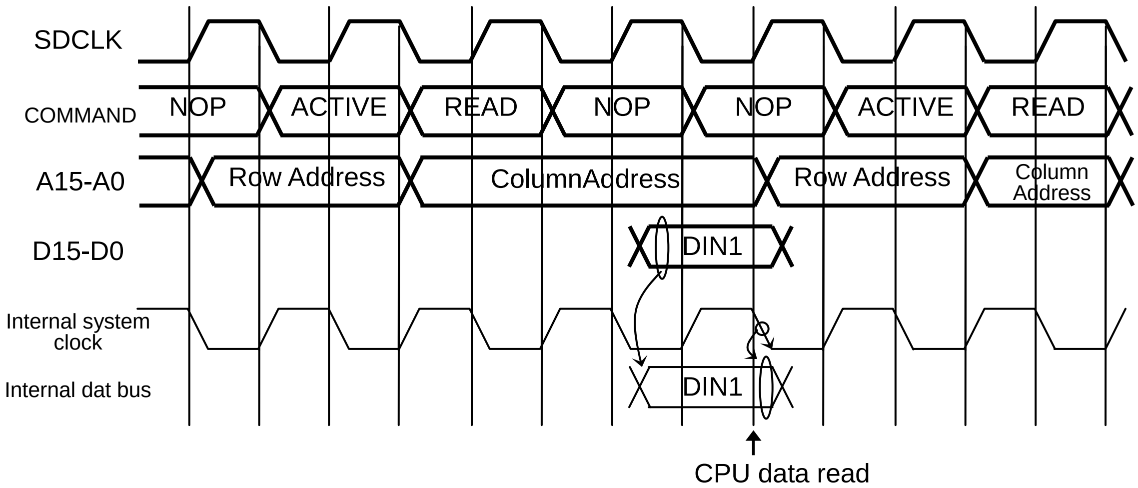


*TRCD=2CLK (SDCISR<STRCD>= "1")
*TWR=2CLK (SDCISR<STWR>= "1")
*TRP=2CLK (SDCISR<STRP>= "1")
*TRC=6CLK (SDCISR<STRC2:0>= "101")

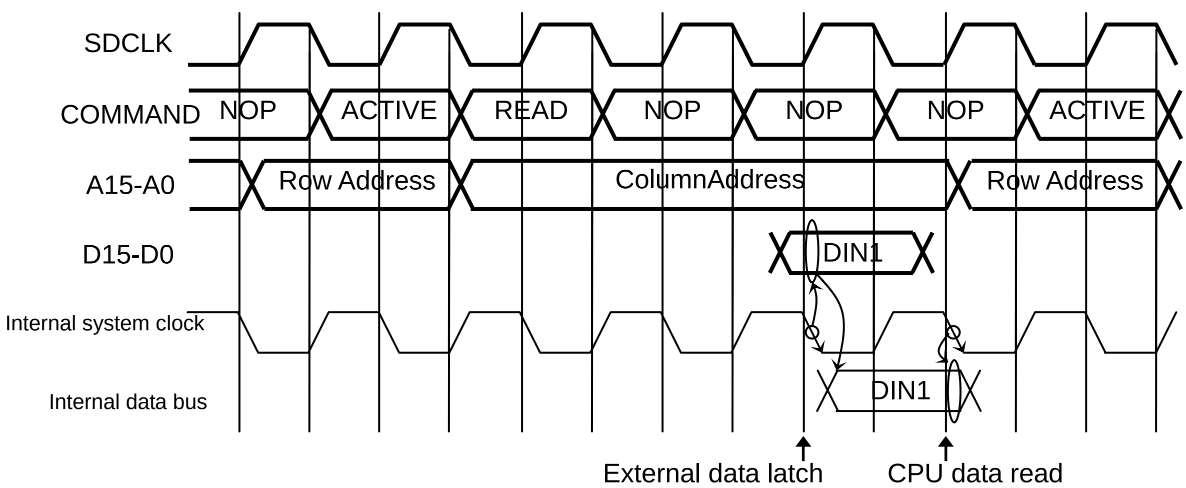
(4) Read data shift function

If the AC specifications of the SDRAM cannot be satisfied when data is read from the SDRAM, the read data can be latched in a port circuit so that the CPU can read the data in the next state. When this read data shift function is used, the read cycle requires additional one state. The write cycle is not affected. The timing waveforms for various cases are shown below.

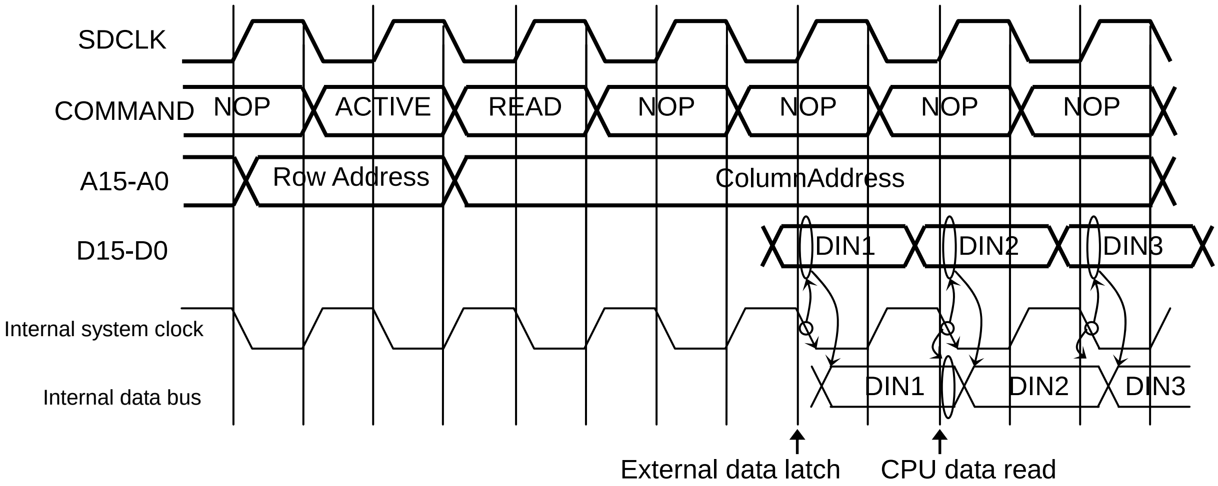
(a) 1-word read, the read data shift function disabled (SDACR<SRCS> = "0")



(b) 1-word read, the read data shift function enabled (SDACR<SRDS> = "1", <SRDSCK>= "0")



(c) Full-page read, the read data shift function enabled (SDACR<SRDS> = “1”, <SRDSCK> = “0”)



(5) Read/Write commands

The Read/Write commands to be used in 1-word read/single write mode can be specified by using SDACR<SPRE>.

When SDACR<SPRE> is set to “1”, the Read/Write commands are executed with Auto Precharge. When Auto Precharge is enabled, the SDRAM is automatically precharged internally at every access cycle. Thus, the SDRAM is always in a “bank idle” state while it is not being accessed. This helps reduce the power consumption of the SDRAM but at the cost of degradation in performance as the Bank Active command is needed at every access cycle.

When SDACR<SPRE> is set to “0”, the Read/Write commands are executed without Auto Precharge. In this case, the SDRAM is not precharged at every access cycle and is always in a “bank active” state. This increases the power consumption of the SDRAM, but improves performance as there is no need to issue the Bank Active command at every access cycle. If an access is made to outside the SDRAM page boundaries or if the Auto Refresh command is issued, the SDRAMC automatically issues the Precharge All command.

And this micro has LCD controller and DMA controller, in case of using below condition, there is one limitation. When SDRAM is set as VRAM for LCD controller and DMA controller is operated at the same time, always set to “1” to SDACR<SPRE>.

(6) Refresh control

The TMP92CF29A supports two kinds of refresh commands: Auto Refresh and Self Refresh.

(a) Auto Refresh

When SDRCR<SRC> is set to "1", the Auto Refresh command is automatically issued at intervals specified by SDRCR<SRS2:0>. The Auto Refresh interval can be specified in a range of 47 states to 1248 states (0.78 μs to 20.8 μs at f_{SYS} = 60 MHz).

The CPU operation (instruction fetch and execution) is halted while the Auto Refresh command is being executed. Figure 3.11.6 shows the Auto Refresh cycle timing, and Table 3.11.3 shows the Auto Refresh interval settings. The Auto Refresh function cannot be used in IDLE1 and STOP modes. In these modes, use the Self Refresh function to be explained next.

Note: A system reset disables the Auto Refresh function.

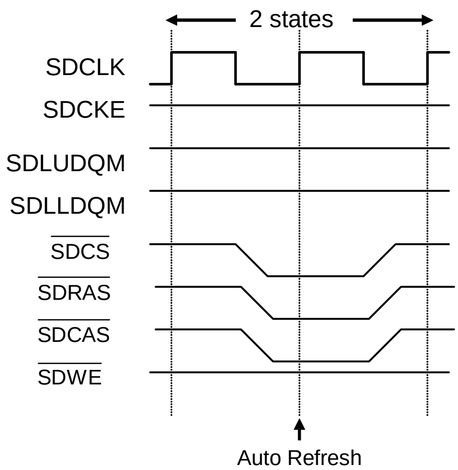


Figure 3.11.6 Auto Refresh Cycle Timing

Note1: Set the interval of Auto Refresh as below table for your reference.

Note2: Take care SDRAM specification and CPU operation speed, please.

Table 3.11.3 System clock speed & auto refresh interval

SDCR<SRS2:0>			interval state	Frequency: system clock [MHz]											
SRS2	SRS1	SRS0		1	2	3	4	6	8	10	20	30	40	60	80
				Time: auto refresh interval [μs]											
0	0	0	47	47.0	23.5	15.67	11.75	7.83	5.88	4.70	2.35	1.57	1.18	0.78	0.59
0	0	1	78	78.0	39.0	26.0	19.5	13.0	9.75	7.80	3.9	2.60	1.95	1.30	0.98
0	1	0	156	156.0	78.0	52.0	39.0	26.0	19.5	15.60	7.8	5.20	3.90	2.60	1.95
0	1	1	312	312.0	156.0	104.0	78.0	52.0	39.0	31.2	15.60	10.4	7.80	5.20	3.90
1	0	0	468	468.0	234.0	156.0	117.0	78.0	58.5	46.8	23.4	15.60	11.7	7.80	5.85
1	0	1	624	624.0	312.0	208.0	156.0	104.0	78.0	62.4	31.2	20.8	15.60	10.4	7.80
1	1	0	936	936.0	468.0	312.0	234.0	156.0	117.0	93.6	46.8	31.2	23.4	15.60	11.70
1	1	1	1248	1248.0	624.0	416.0	312.0	208.0	156.0	124.8	62.4	41.6	31.2	20.8	15.60

Note: Above gray zone is prohibited to set. SDRAM request: 4096 times per 64ms.

(b) Self Refresh

The Self Refresh Entry command is issued by setting SDCMM<SCMM2:0> to "101". Figure 3.11.7 shows the Self Refresh cycle timing. Before entering Self-refresh mode, issue the all Bank Pre-charge Command. Once Self Refresh is started, the SDRAM is refreshed internally without the need to issue the Auto Refresh command.

Note 1: When standby mode is released by a system reset, the I/O registers are initialized and the Self Refresh state is exited. Note that the Auto Refresh function is also disabled at this time.

Note 2: The SDRAM cannot be accessed while it is in the Self Refresh state.

Note 3: To execute the HALT instruction after the Self Refresh Entry command, insert at least 10 bytes of NOP or other instructions between the instruction to set SDCMM<SCMM2:0> to "101" and the HALT instruction.

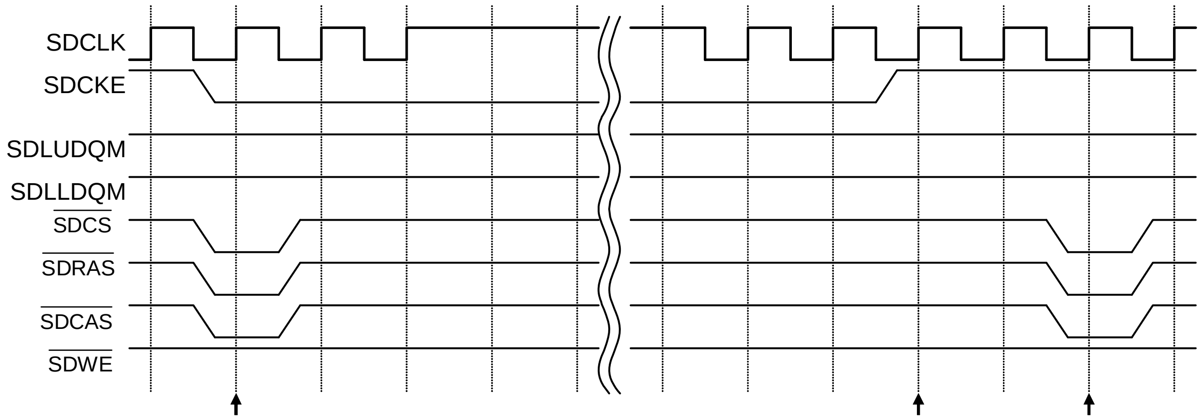


Figure 3.11.7 Self Refresh Cycle Timing

Setting Example

```
org   0x2000          ; Internal RAM
ld    (sdcmm),0x02    ; All Bank Precharge Command
ld    (sdcmm),0x05    ; Self Refresh Entry Command
NOP×10                ; Setup time
halt                  ;
```

The Self Refresh state can be exited by the Self Refresh Exit command. The Self Refresh Exit command is executed when SDCMM<SCMM2:0> is set to “110”. It is also executed automatically in synchronization with HALT mode release. In either of these two cases, Auto Refresh is performed immediately after the Self Refresh state is exited. Then, Auto Refresh is executed at specified intervals. Exiting the Self Refresh state clears SDCMM<SCMM2:0> to “000”.

SDRAM Refresh Control Register

		7	6	5	4	3	2	1	0
SDRCR (0252H)	Bit symbol	–			SSAE	SRS2	SRS1	SRS0	SRC
	Read/Write	R/W			R/W				
	Reset State	0			1	0	0	0	0
	Function	Always write “0”			Self Refresh auto exit function 0:Disable 1:Enable	Refresh interval 000: 47 states 001: 78 states 010: 156 states 011: 312 states	100: 468 states 101: 624 states 110: 936 states 111: 1248 states		Auto Refresh 0:Disable 1:Enable

Setting SDRCR<SSAE> to “1” enables automatic execution of the Self Refresh Exit command in synchronization with HALT release.

Setting SDRCR<SSAE> to “0” disables automatic execution of the Self Refresh Exit command in synchronization with HALT release. The auto exit function should also be disabled in cases where the SDRAM operation requirements cannot be met as the operation clock frequency is reduced by clock gear down, as shown in Figure 3.11.8.

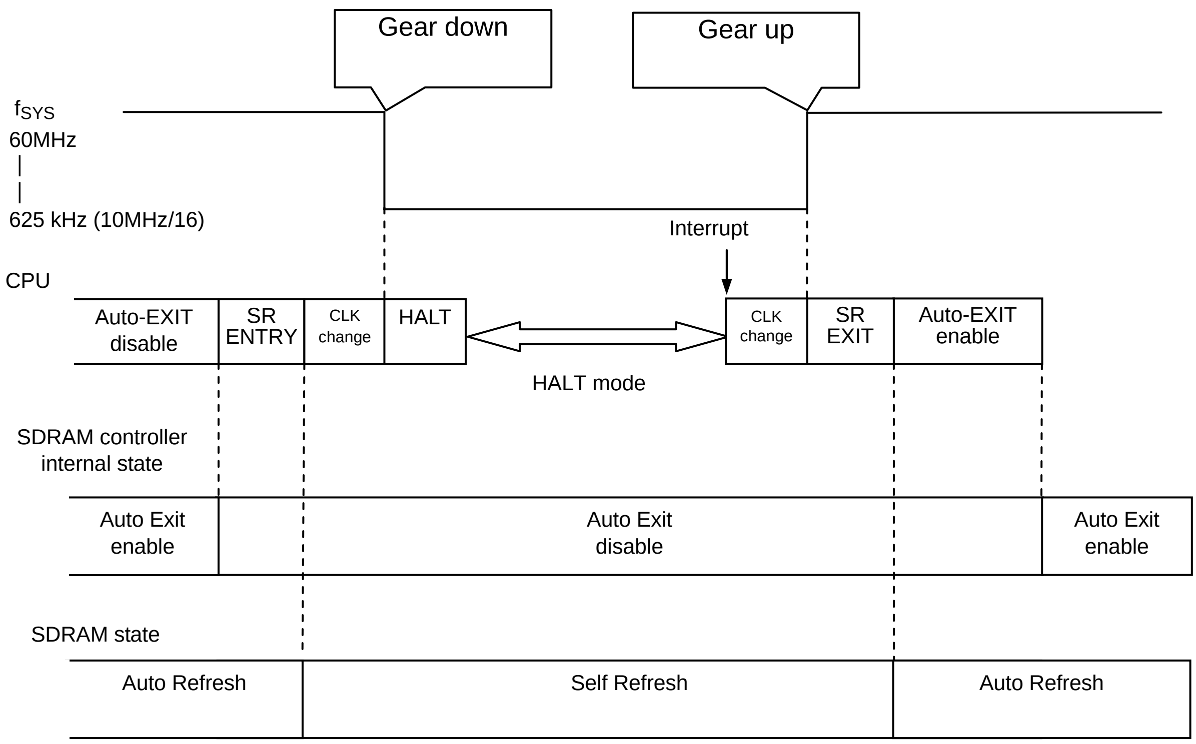


Figure 3.11.8 Execution Flow for Executing HALT Instruction after Clock Gear Down

(7) SDRAM initialization sequence

After reset release, the following sequence of commands can be executed to initialize the SDRAM.

Precharge All command

Eight Auto Refresh commands

Mode Register Set command

The above commands are issued by setting SDCMM<SCMM2:0> to “001”. While these commands are issued, the CPU operation (instruction fetch, execution) is halted. Before executing the initialization sequence, appropriate port settings must be made to enable the SDRAM control signals and address signals (A0 to A15).

After the initialization sequence is completed, SDCMM<SCMM2:0> is automatically cleared to “000”.

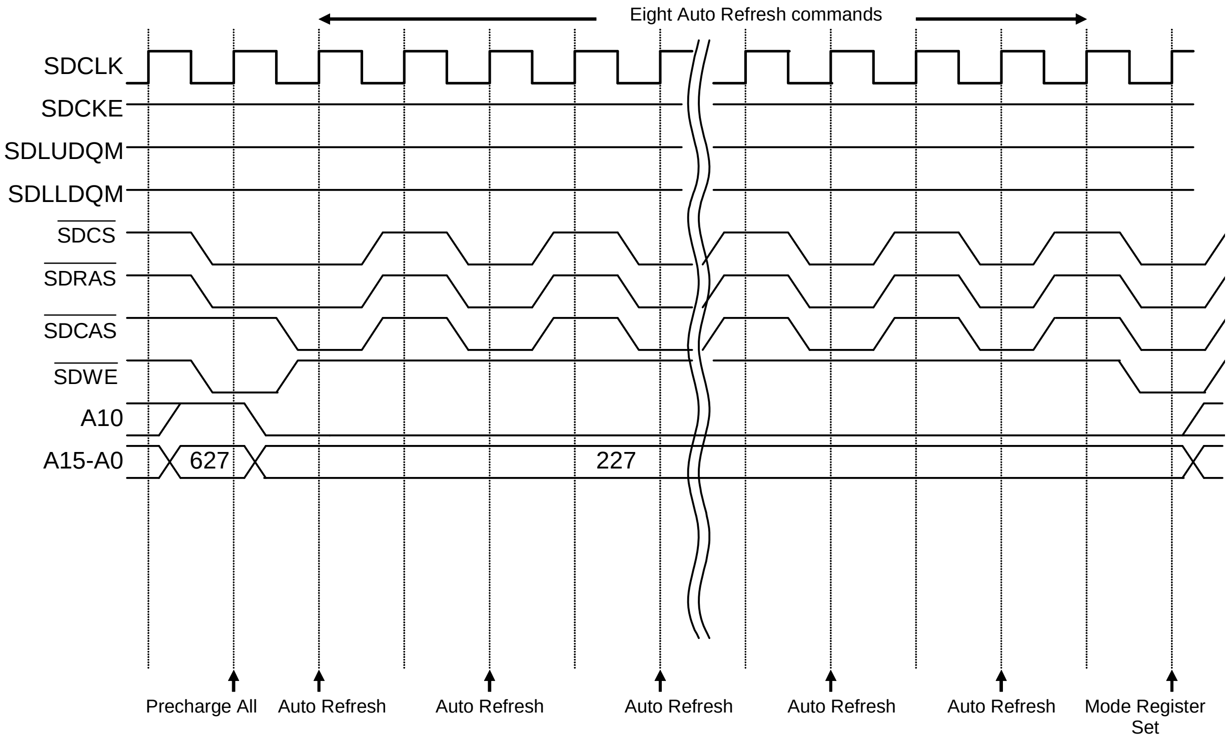


Figure3.11.9 Initialization Sequence Timing

(8) Connection example

Figure 3.11.10 shows an example of connections between the TMP92CF29A and SDRAM.

Table 3.11.4 Pin Connections

92CF29A Pin Name	SDRAM Pin Name				
	Data Bus Width 16 bits				
	16M	64M	128M	256M	512M
A0	A0	A0	A0	A0	A0
A1	A1	A1	A1	A1	A1
A2	A2	A2	A2	A2	A2
A3	A3	A3	A3	A3	A3
A4	A4	A4	A4	A4	A4
A5	A5	A5	A5	A5	A5
A6	A6	A6	A6	A6	A6
A7	A7	A7	A7	A7	A7
A8	A8	A8	A8	A8	A8
A9	A9	A9	A9	A9	A9
A10	A10	A10	A10	A10	A10
A11	BS	A11	A11	A11	A11
A12	–	BS0	BS0	A12	A12
A13	–	BS1	BS1	BS0	BS0
A14	–	–	–	BS1	BS1
A15	–	–	–	–	–
$\overline{\text{SDCS}}$	CS	CS	CS	CS	CS
SDLUDQM	UDQM	UDQM	UDQM	UDQM	UDQM
SDLLDQM	LDQM	LDQM	LDQM	LDQM	LDQM
$\overline{\text{SDRAS}}$	RAS	RAS	RAS	RAS	RAS
$\overline{\text{SDCAS}}$	CAS	CAS	CAS	CAS	CAS
$\overline{\text{SDWE}}$	WE	WE	WE	WE	WE
SDCKE	CKE	CKE	CKE	CKE	CKE
SDCLK	CLK	CLK	CLK	CLK	CLK
SDACR <SMUXW>	00: TypeA	00: TypeA	01: TypeB	01: TypeB	10: TypeC

(shaded): Command address pin of SDRAM

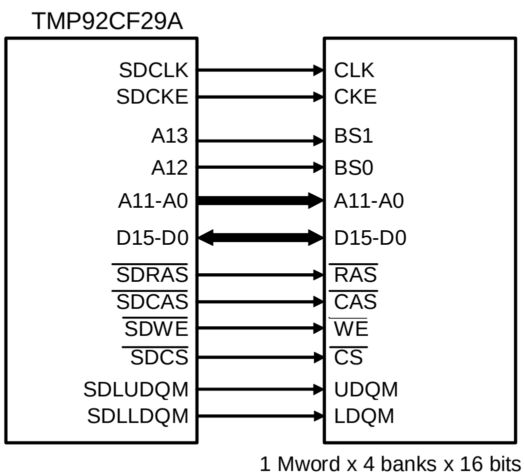


Figure 3.11.10 An Example of Connections between TMP92CF29A and SDRAM

3.11.3 An Example of Calculating HDMA Transfer Time

The following shows an example of calculating the HDMA transfer time when SDRAM is used as the transfer source.

- Transfer from SDRAM to internal SRAM

Conditions:

System clock (f_{SYS}) : 60 MHz
SDRAM read cycle : Full page (5-1-1-1), 16-bit data bus
16-bit data bus
SDRAM Auto Refresh interval: 936 states (15.6 μs)
Internal RAM write cycle : 1 state, 32-bit data bus
Number of bytes to transfer : 512 bytes

Calculation example:

Transfer time = (SDRAM read time + SRAM write time) × transfer count
+ (SDRAM burst start + stop time)
+ (Precharge time + Auto Refresh time) × Auto Refresh count

(a) Read/write time
(SDRAM read 1 state × 2 + Internal RAM write 1 state) × 512 bytes/4 bytes
= 384 states × 1/60 MHz
= 6.4 μs

(b) Burst start/stop time
Start (TRCD: 2CLK) 5 states + Stop 2 states
= 7states/60 MHz
= 0.117 μs

(c) Auto Refresh time
Based on the above (a), Auto Refresh occurs once or zero times in 384 states. It is assumed that Auto Refresh occurs once here.

(Precharge (TRP: 2CLK) 2 states + AREF (TRC: 5CLK) 5 states) ×AREF once
= 7 states × 1/60 MHz
= 0.117 μs

Total transfer time = (a) + (b) + (c)
= 6.4 μs + 0.117 μs + 0.117 μs
= 6.634 μs

3.11.4 Considerations for Using the SDRAMC

This section describes the points that must be taken into account when using the SDRAMC. Please carefully read the following to ensure proper use of the SDRAMC.

1) WAIT access

When SDRAM is used, the following restriction applies to memory access to other than the SDRAM.

In the external WAIT pin input setting of the memory controller, the maximum external WAIT period that can be set is limited to “Auto Refresh interval × 8190”.

2) Execution of the Self Refresh Entry, Initialization Sequence, or Precharge All command before the HALT instruction

Execution of the commands issued by the SDRAMC (Self Refresh Entry, Initialization Sequence, Precharge All) requires several states after the SDCMM register is set.

Therefore, to execute the HALT instruction after one of these commands, be sure to insert at least 10 bytes of NOP or other instructions.

3) Auto Refresh interval setting

When SDRAM is used, the system clock frequency must be set to satisfy the minimum operation frequency and minimum Auto Refresh interval of the SDRAM to be used.

In a system in which SDRAM is used and the clock is geared up and down, the Auto Refresh interval must be set carefully.

Before changing the Auto Refresh interval, ensure that SDRCR<SRC> is set to “0” to disable the Auto Refresh function.

4) Changing SFR settings

Before changing the settings of the SDACR<SPRE> and SDCISR registers, ensure that the SDRAMC is disabled (SDACR<SMAC> =“0”).

5) Disabling the SDRAMC

Set the following procedure, when disable the SDRAMC.

```
        LD   (SDCMM),0x02   ;   Issue to All Bank Precharge
LOOP:   LD   A,(SDCMM)      ;   Read SDCMM
        CP   A,0x00         ;   Palling it until the All Bank Precharge command  is finished
        JP   NZ,LOOP        ;
        LD   (SDACR),0x00   ;   Stop the SDRAM controller
```

6) Using LCDC, DMAC with SDRAMC

And this micro has LCD controller and DMA controller, in case of using below condition, there is one limitation. When SDRAM is set as VRAM for LCD controller and DMA controller is operated at the same time, always set to “1” to SDACR<SPRE>.

3.12 NAND Flash Controller (NDFC)

3.12.1 Features

The NAND Flash Controller (NDFC) is provided with dedicated pins for connecting with NAND Flash memory.

The NDFC also has an ECC calculation function for error correction and supports two types of ECC calculation methods. The ECC calculation method using Hamming codes can be used for NAND Flash memory of SLC (Single Level Cell) type and is capable of detecting a single-bit error for every 256 bytes. The ECC calculation method using Reed-Solomon codes can be used for NAND Flash memory of MLC (Multi Level Cell) type and is capable of detecting four error addresses for every 518 bytes.

Although the NDFC has two channels (channel 0, channel 1), all pins except for Chip Enable are shared between the two channels. Only the operation of channel 0 is explained here.

The NDFC has the following features:

1) Controls the NAND Flash memory interface through registers.
2) Supports 8-bit and 16-bit NAND Flash memory devices.
3) Supports page sizes of 512 bytes and 2048 bytes.
4) Supports large-capacity block sizes over 256 Kbytes.
5) Includes an ECC generation circuit using Hamming codes (for SLC type).
6) Includes a 4-address (4-byte) error detection circuit using Reed-Solomon coding/ encoding techniques (for MLC type).

Note 1: The $\overline{\text{WP}}$ (Write Protect) pin of NAND Flash is not supported. If this function is needed, prepare it on an external circuit.

Note 2: The two channels cannot be accessed simultaneously. It is necessary to switch between the two channels.

3.12.2 Block Diagram

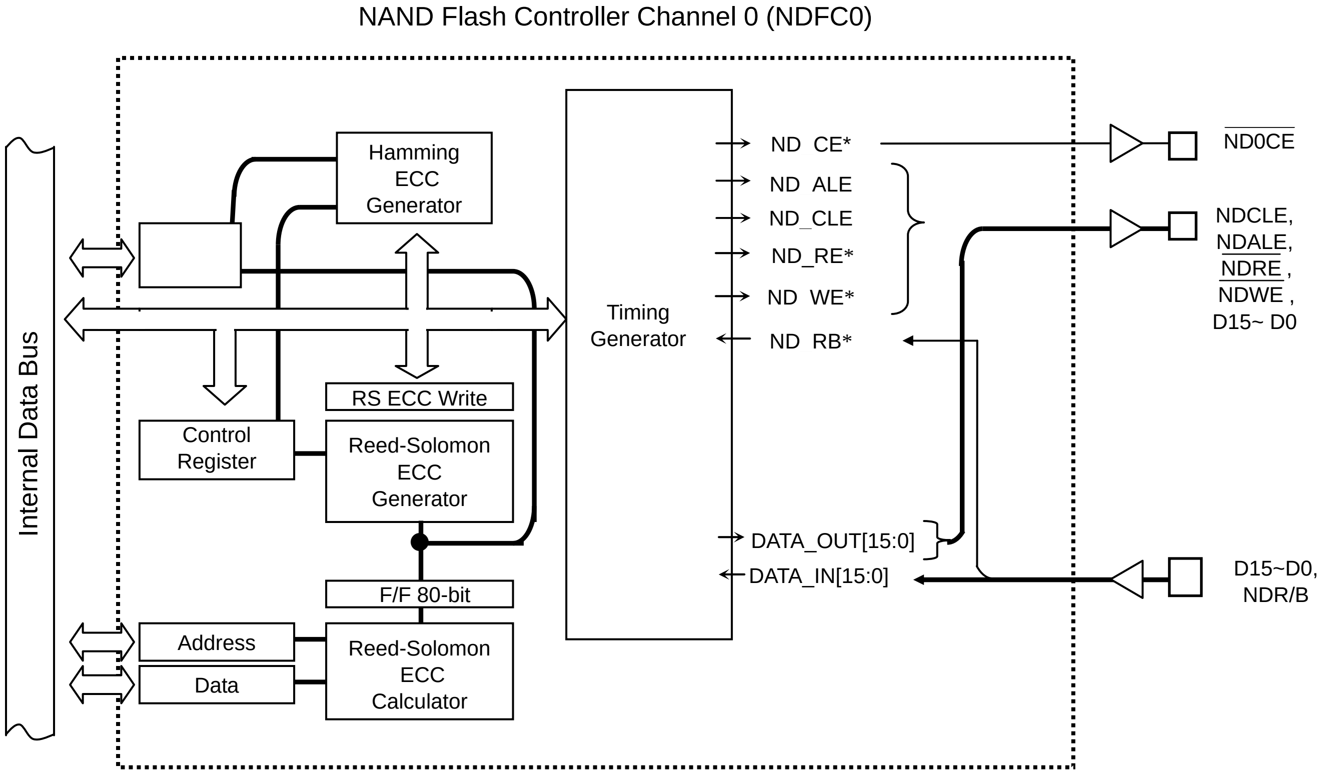


Figure 3.12.1 Block Diagram for NAND Flash Controller

3.12.3 Operation Description

3.12.3.1 Accessing NAND Flash Memory

The NDFC accesses data on NAND Flash memory indirectly through its internal registers. This section explains the operations for accessing the NAND Flash.

Since no dedicated sequencer is provided for generating commands to the NAND Flash, the levels of the NDCLE, NDALE, and $\overline{\text{NDCE}}$ pins must be controlled by software.

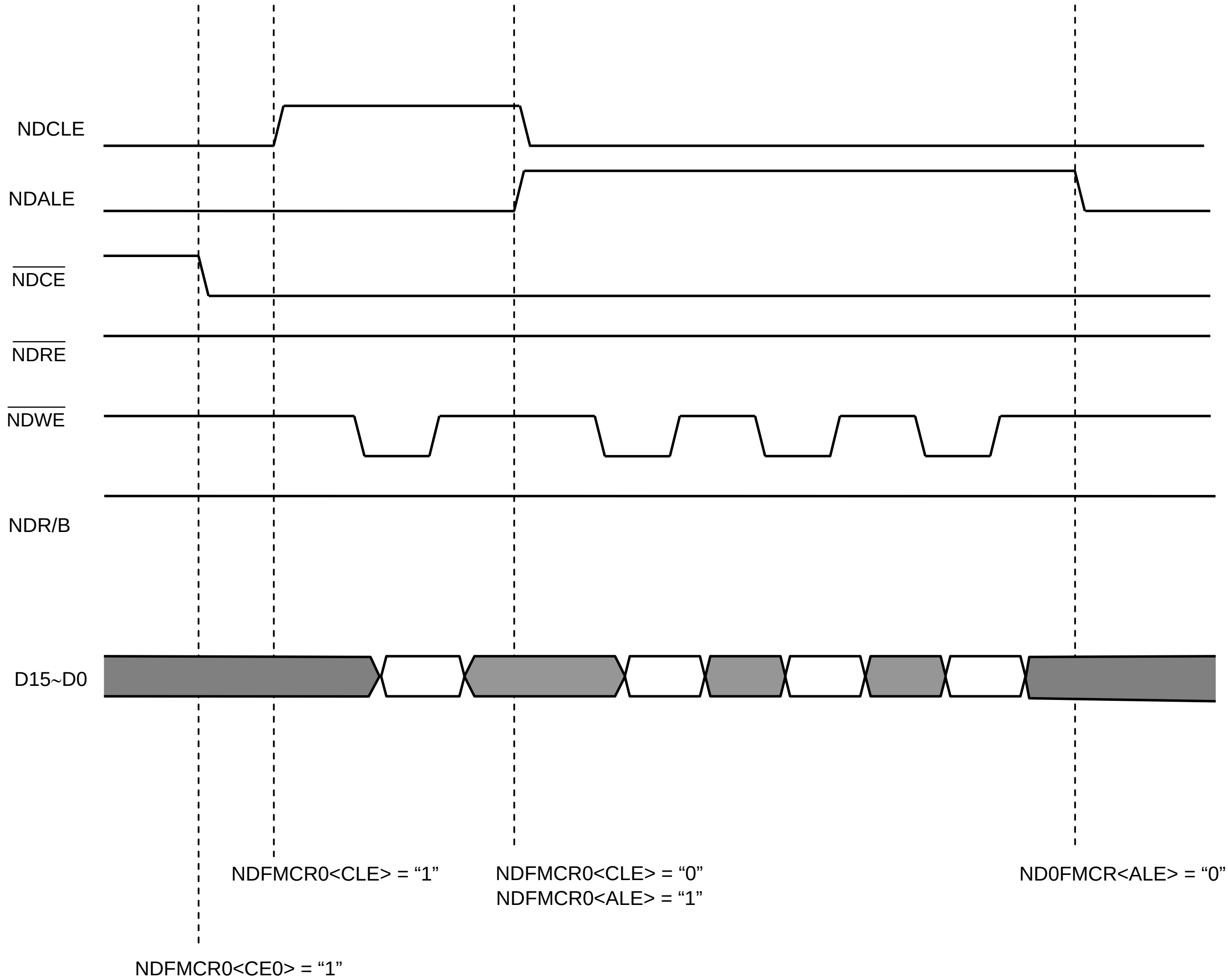


Figure3.12.2 Basic Timing for Accessing NAND Flash

The $\overline{NDRE}$ and $\overline{NDWE}$ signals are explained next. Write and read operations to and from the NAND Flash are performed through the ND0FDTR register. The actual write operation completes not when the ND0FDTR register is written to but when the data is written to the external NAND Flash. Likewise, the actual read operation completes not when the ND0FDTR register is read but when the data is read from the external NAND Flash.

At this time, the Low and High widths of $\overline{NDRE}$ and $\overline{NDWE}$ can be adjusted according to the CPU operating speed (f_{SYS}) and the access time of the NAND Flash. (For details, refer to the electrical characteristics.)

The following shows an example of accessing the NAND Flash in 6 clocks by setting NDFMCR0<SPLW1:0>=2 and NDFMCR0<SPHW1:0>=2. (In write cycles, the data drive time also becomes longer.)

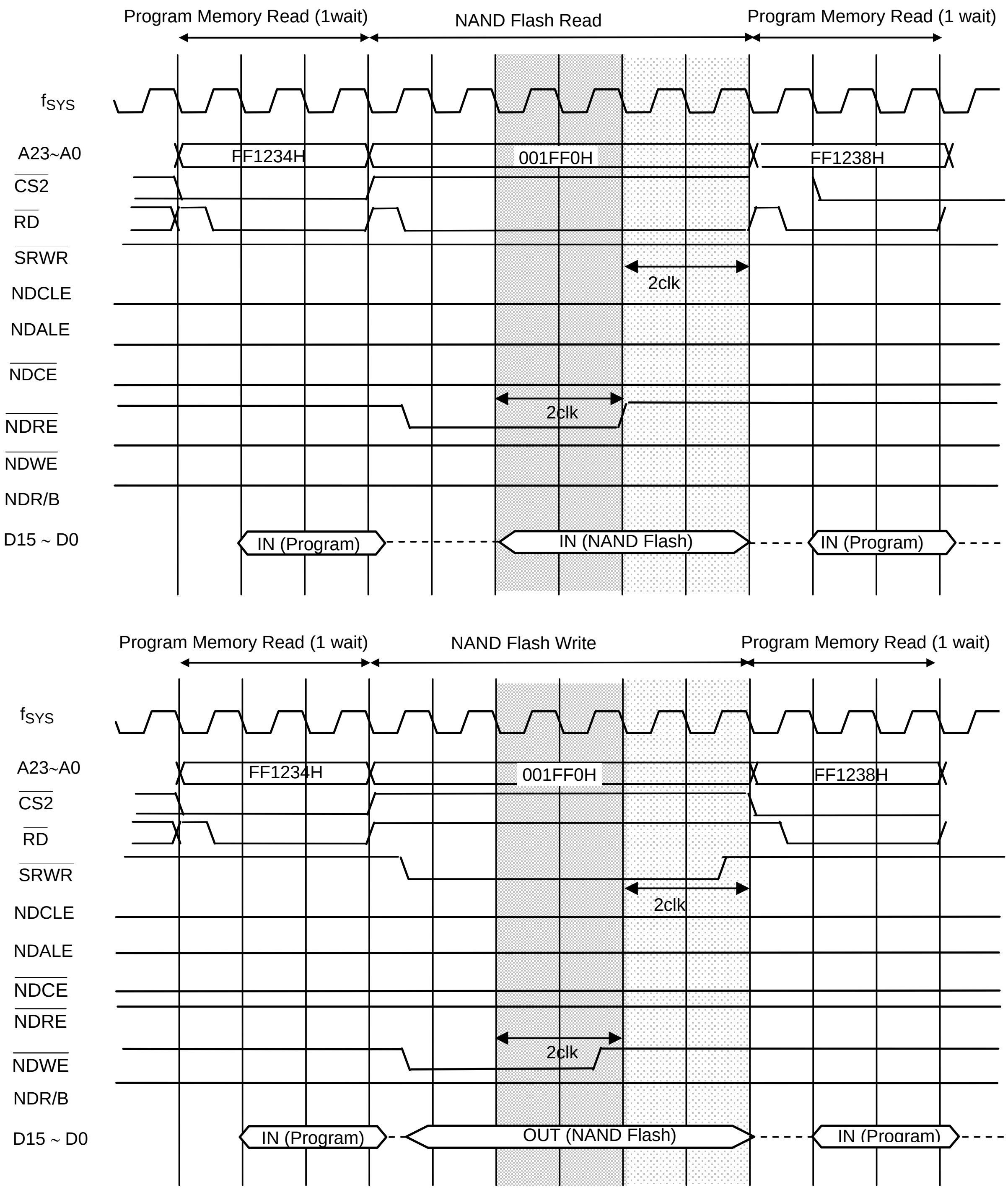


Figure3.12.3 Read/Write Access to NAND Flash

3.12.4 ECC Control

NAND Flash memory devices may inherently include error bits. It is therefore necessary to implement the error correction processing using ECC (Error Correction Code).

Figure3.12.4 shows a basic flowchart for ECC control.

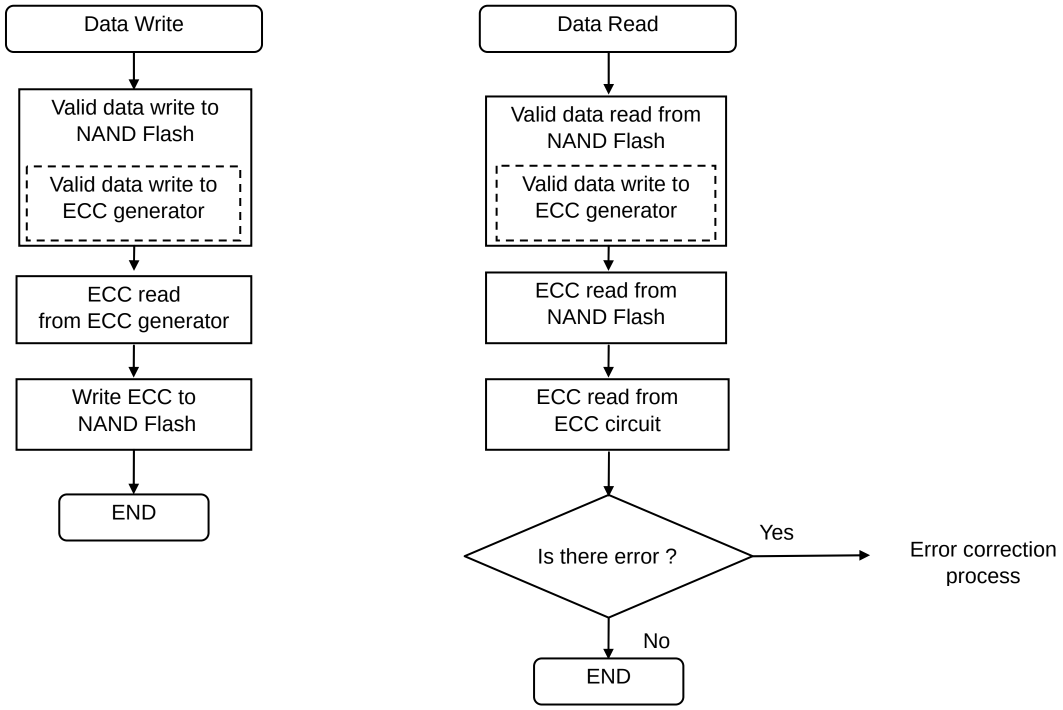


Figure3.12.4 Basic Flow of ECC Control

Write:

1. When data is written to the actual NAND Flash memory, the ECC generator in the NDFC simultaneously generates ECC for the written data.
2. The ECC is written to the redundant area in the NAND Flash separately from the valid data.

Read:

1. When data is read from the actual NAND Flash memory, the ECC generator in the NDFC simultaneously generates ECC for the read data.
2. The ECC for the written data and the ECC for the read data are compared to detect and correct error bits.

3.12.4.1 Differences between Hamming Codes and Reed-Solomon Codes

The NDFC includes an ECC generator supporting NAND Flash memory devices of SLC (or 2LC: two states) type and MLC (or 4LC: four states) type.

The ECC calculation using Hamming codes (supporting SLC) generates 22 bits of ECC for every 256 bytes of valid data and is capable of detecting and correcting a single-bit error for every 256 bytes. Error bit detection calculation and correction must be implemented by software. When using SmartMedia™, Hamming codes should be used.

The ECC calculation using Reed-Solomon codes (supporting MLC) generates 80 bits of ECC for every 1 byte to 518 bytes of valid data and is capable of detecting and correcting error bits at four addresses for every 518 bytes. When using Reed-Solomon codes, error bit detection calculation is supported by hardware and only error bit correction needs to be implemented by software.

The differences between Hamming codes and Reed-Solomon codes are summarized in Table 3.12.1.

Table 3.12.1 Differences between Hamming Codes and Reed-Solomon Codes

	Hamming	Reed-Solomon
Maximum number of correctable errors	1 bit	4 addresses (All the 8 bits at one address are correctable.)
Number of ECC bits	22 bits/256 bytes	80 bits/up to 518 bytes
Error bit detection method	Software	Hardware
Error bit correction method	Software	Software
Error bit detection time	Depends on the software to be used.	See the table below.
Others	Supports SmartMedia™.	–

Number of Error Bits	Reed-Solomon Error Bit Detection Time (Unit: Clocks)	Notes
4	813 (max)	These values indicate the total number of clocks for detecting error bit(s) not including the register read/write time by the CPU.
3	648 (max)	
2	358 (max)	
1	219 (max)	
0	1	

3.12.4.2 Error Correction Methods

Hamming ECC

- The ECC generator generates 44 bits of ECC for a page containing 512 bytes of valid data. The error correction process must be performed in units of 256 bytes (22 bits of ECC). The following explains how to implement error correction on 256 bytes of valid data using 22 bits of ECC.
- If the NAND Flash to be used has a large-capacity page size (e.g. 2048 bytes), the error correction process must be repeated several times to cover the entire page.

1) The calculated ECC and the ECC in the redundant area are rearranged, respectively, so that the lower 2 bytes represent line parity (LPR15:0) and the upper 1 byte (of which the upper 6 bits are valid) represents column parity (CPR7:2).
2) The two rearranged ECCs are XORed.
3) If the XOR result is 0 indicating an ECC match, the error correction process ends normally (no error). If the XOR result is other than 0, it is checked whether or not the error data can be corrected.
4) If the XOR result contains only one ON bit, it is determined that a single-bit error exists in the ECC data itself and the error correction process terminates here (error not correctable).
5) If each pair of bits 0 to 21 of the XOR result is either 01B or 10B, it is determined that the error data is correctable and error correction is performed accordingly. If the XOR result contains either 00B or 11B, it is determined that the error data is not correctable and the error correction process terminates here.

	An Example of Correctable XOR Result	An Example of Uncorrectable XOR Result
Binary	10 01 10 00 Column parity 10 10 01 10 Line parity 01 01 10 10	10 11 10 00 Column parity 10 10 01 10 Line parity 01 01 10 10

6) The line and bit positions of the error are detected using the line parity and column parity of the XOR result, respectively. The error bit thus detected is then inverted. This completes the error correction process.

Example: When the XOR result is 1001101010011001011010

Convert two bytes of line parity into one byte (10→1, 01→0).

Convert six bits of column parity into three bits (10→1, 01→0).

Line parity: 10 10 01 10 01 01 10 10

⇩ ⇩ ⇩ ⇩ ⇩ ⇩ ⇩ ⇩

1 1 0 1 0 0 1 1 = D3H *Error at D3/FF H

Column parity: 10 01 10

⇩ ⇩ ⇩

1 0 1 = 5 *Error in bit 5

Based on the above, error correction is performed by inverting the data in bit 5 at address 212.

Reed-Solomon ECC

- The ECC generator generates 80 bits of ECC for up to 518 bytes of valid data. If the NAND Flash to be used has a large-capacity page size (e.g. 2048 bytes), the error correction process must be repeated several times to cover the entire page.
- Basically no calculation is needed for error correction. If error detection is performed properly, the NDFC only needs to refer to the error address and error bit. However, it may be necessary to convert the error address, as explained below.

1) If the error address indicated by the NDRSCAn register is in the range of 000H to 007H, this error exists in the ECC area and no correction is needed in this case.

   (It is not able to correct the error in the ECC area. However, if the error exists in the ECC area, only 4symbol (include the error in the ECC area) can correct the error to this LSI. Please be careful.)

2) If the error address indicated by the NDRSCAn register is in the range of 008H to 20DH, the actual error address is obtained by subtracting this address from 20 DH.

   (If the valid data is processed as 512 byte, the actual error address is obtained by subtracting this address from 207H when the error address in the range of 008H to 207H.)

Example 1:

NDRSCAn = 005H, NDRSCDn = 04H = 00000100B

As the error address (005H) is in the range of 000H to 007H, no correction is needed.

(Although an error exists in bit 2, no correction is needed.)

Example 2:

NDRSCAn = 083H, NDRSCDn = 81H = 10000001B

The actual error address is obtained by subtracting 083H from 20DH. Thus, the error correction process inverts the data in bits 7 and 0 at address 18AH.

(If the valid data is 512 byte, the actual error address is obtained by subtracting 083H from 207H. Thus, the error correction process inverts the data in bits 7 and 0 at address 184H.)

Note: If the error address (after converted) is in the range of 000H to 007H, it indicates that an error bit exists in redundant area (ECC). In this case, no error correction is needed. If the number of error bits is not more than 4 symbols, Reed-Solomon codes calculate each error bit precisely even if it is the redundant area (ECC).

3.12.5 Description of Registers

NAND Flash Control 0 Register

		7	6	5	4	3	2	1	0
NDFMCR0 (08C0H)	bit Symbol	WE	ALE	CLE	CE0	CE1	ECCE	BUSY	ECCRST
	Read/Write	R/W						R	W
	Reset State	0	0	0	0	0	0	0	0
A read-modify -write operation cannot be performed	Function	WE enable 0: Disable 1: Enable	ALE control 0: "L" out 1: "H" out	CLE control 0: "L" out 1: "H" out	$\overline{\text{CE0}}$ control 0: "H" out 1: "L" out	$\overline{\text{CE1}}$ control 0: "H" out 1: "L" out	ECC circuit control 0: Disable 1: Enable	NAND Flash state 1: Busy 0: Ready	ECC reset control 0: – 1: Reset *Always read as "0".

		15	14	13	12	11	10	9	8
(08C1H)	bit Symbol	SPLW1	SPLW0	SPHW1	SPHW0	RSECCL	RSEDN	RSESTA	RSECGW
	Read/Write	R/W						W	R/W
	Reset State	0	0	0	0	0	0	0	0
A read-modify-write operation cannot be performed	Function	Strobe pulse width (Low width of $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$) Inserted width = (f_{SYS}) × (set value)		Strobe pulse width (High width of $\overline{\text{NDRE}}$, $\overline{\text{NDWE}}$) Inserted width = (f_{SYS}) × (set value)		Reed-Solomon ECC latch 0: Disable 1: Enable	Reed-Solomon operation 0: Encode (Write) 1: Decode (Read)	Reed-Solomon error calculation start 0: – 1: Start *Always read as "0".	Reed-Solomon ECC generator write control 0: Disable 1: Enable

Figure3.12.5 NAND Flash Mode Control 0 Register

(a) <ECCRST >

The <ECCRST> bit is used for both Hamming and Reed-Solomon codes.

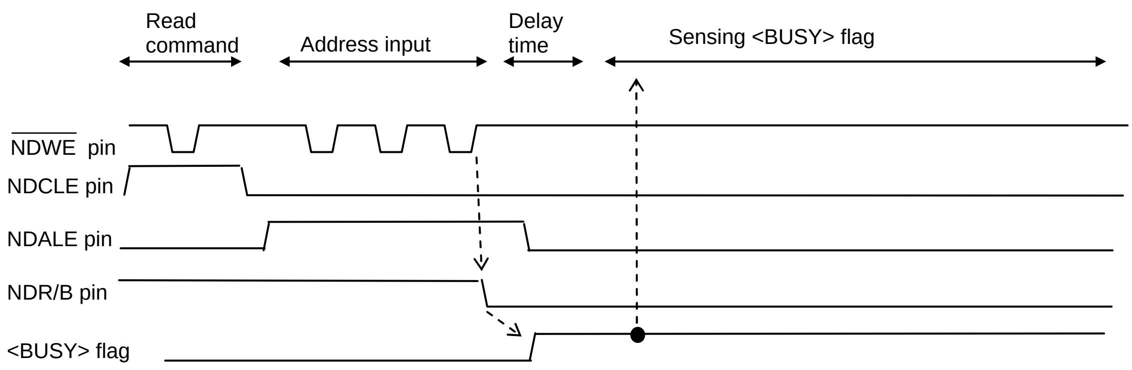
When NDFMCR1<ECCS>="0", setting this bit to "1" clears the Hamming ECC in the ECC generator. When NDFMCR1<ECCS>="1", setting this bit to "1" clears the Reed-Solomon ECC. Note that this bit is ineffective when NDFMCR0<ECCE>="0". Before writing to this bit, ensure that NDFMCR0<ECCE>="1".

(b) <BUSY>

The <BUSY> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to check the state of the NAND Flash memory (NDR/B pin). It is set to “1” when the NAND Flash is “busy” and to “0” when it is “ready”.

Since the NDFC incorporates a noise filter of several states, a change in the NDR/B pin state is reflected on the <BUSY> flag after some delay. It is therefore necessary to inert a delay time by software (e.g. ten NOP instructions) before checking this flag.



(c) <ECCE>

The <ECCE> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to enable or disable the ECC generator. To reset the ECC in the ECC generator (to set <ECCRST> to “1”), the ECC generator must be enabled (<ECCE> = “1”).

(d) <CE1:0>, <CLE>, <ALE>

The <CE1:0>, <CLE>, and <ALE> bits are used for both Hamming and Reed-Solomon codes to control the pins of the NAND Flash memory.

(e) <WE>

The <WE> bit is used for both Hamming and Reed-Solomon codes to enable or disable write operations.

(f) <RSECGW>

The <RSECGW> bit is used only for Reed-Solomon codes. When Hamming codes are used, this bit should be set to “0”.

Since valid data and ECC are processed differently, the NDFC needs to know whether valid data or ECC is to be read. This control is implemented by software using this bit.

To read valid data from the NAND Flash, set <RSECGW> to “0”. To read ECC written in the redundant area in the NAND Flash, set <RSECGW> to “1”.

Note 1: Valid data and ECC cannot be read continuously by DMA transfer. After valid data has been read, DMA transfer should be stopped once to change the <RSECGW> bit from “0” to “1” before ECC can be read.

Note 2: Immediately after ECC is read from the NAND Flash, the NAND Flash access operation or error bit calculation cannot be performed for a duration of 20 system clocks (f_{SYS}). It is necessary to insert 20 NOP instructions or the like.

(g) <RSESTA>

The <RSESTA> bit is used only for Reed-Solomon codes.

The error address and error bit position are calculated using an intermediate code generated from the ECC for written data and the ECC for read data. Setting <RSESTA> to "1" starts this calculation.

(h) <RSEDN>

The <RSEDN> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to "0".

For a write operation, this bit should be set to "0" (encode) to generate ECC. The ECC read from the NDECCRDn register is written to the redundant area in the NAND Flash. For a read operation, this bit should be set to "1" (decode). In this case, valid data is read from the NAND Flash and the ECC written in the redundant area is also read to generate an intermediate code for calculating the error address and error bit position.

(i) <RSECCL>

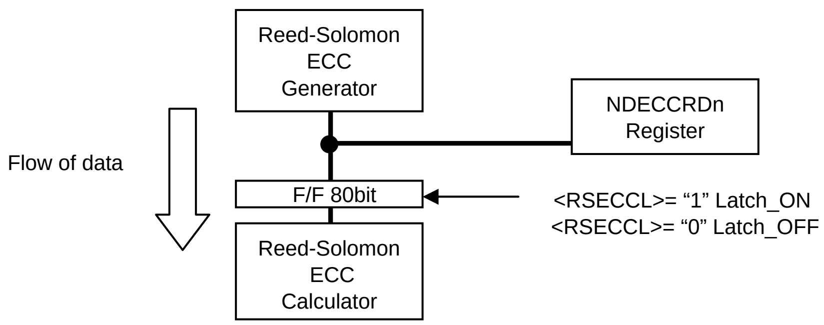
The <RSECCL> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to "0".

The Reed-Solomon processing unit is comprised of two elements: an ECC generator and an ECC calculator. The latter is used to calculate the error address and error bit position.

The error address and error bit position are calculated using an intermediate code generated from the ECC for written data and the ECC for read data. At this time, no special care is needed if ECC generation and error calculation are performed serially. If these operations need to be performed parallely, the intermediate code used for error calculation must be latched while the calculation is being performed. The <RSECCL> bit is provided to enable this latch operation.

When <RSECCL> is set to "1", the intermediate code is latched so that the ECC generator can generate the ECC for another page without problem while the ECC calculator is calculating the error address and error bit position. At this time, the ECC generator can perform both encode (write) and decode (read) operations.

When <RSECCL> is set to "0", the latch is released and the contents of the ECC calculator are updated as the data in the ECC generator is updated.



(j) <SPHW1:0>

The <SPHW1:0> bits are used for both Hamming and Reed-Solomon codes.

These bits are used to specify the High width of the $\overline{NDRE}$ and $\overline{NDWE}$ signals. The High width to be inserted is obtained by multiplying the value set in these bits by f_{SYS}.

(k) <SPLW1:0>

The <SPLW1:0> bits are used for both Hamming and Reed-Solomon codes.

These bits are used to specify the Low width of the $\overline{NDRE}$ and $\overline{NDWE}$ signals. The Low width to be inserted is obtained by multiplying the value set in these bits by f_{SYS}.

NAND Flash Control 1 Register

			7	6	5	4	3	2	1	0
NDFMCR1 (08C2H)	bit Symbol		INTERDY	INTRSC				BUSW	ECCS	SYSCKE
	Read/Write		R/W					R/W		
	Reset State		0	0				0	0	0
	Function		Ready interrupt 0: Disable 1: Enable	Reed-Solomon calculation end interrupt 0: Disable 1: Enable				Data bus width 0: 8-bit 1: 16-bit	ECC calculation 0:Hamming 1: Reed-Solomon	Clock control 0: Disable 1: Enable
(08C3H)			15	14	13	12	11	10	9	8
	bit Symbol		STATE3	STATE2	STATE1	STATE0	SEER1	SEER0		
	Read/Write		R							
	Reset State		0	0	0	0	Undefined	Undefined		
	Function		Status read (See the table below.)							

Table3.12.2 Reed-Solomon Calculation Result Status Table

STATE<3:0>	Meaning
0000	Calculation ended 0 (No error)
0001	Calculation ended 1(5 or more symbols in error; not correctable)
0010	Calculation ended 2 (Error found)
0011	
0100~1111	Calculation in progress

Note: The <STATE3:0> value becomes effective after the calculation has started.

SEER<1:0>	Meaning
00	1-address error
01	2-address error
10	3-address error
11	4-address error

Note: The <SEER1:0> value becomes effective after the calculation has ended.

(a) <SYSCKE>

The <SYSCKE> bit is used for both Hamming and Reed-Solomon codes.

When using the NDFC, this bit must be set to “1” to enable the system clock. When not using the NDFC, power consumption can be reduced by setting this bit to “0”.

(b) <ECCS>

The <ECCS> bit is used to select whether to use Hamming codes or Reed-Solomon codes. This bit is set to “0” for using Hamming codes and to “1” for using Reed-Solomon codes. It is also necessary to set this bit for clearing ECC.

(c) <BUSW>

The <BUSW> bit is used for both Hamming and Reed-Solomon codes.

This bit specifies the bus width of the NAND Flash to be accessed (“0” = 8 bits, “1” = 16 bits). No other setting is required in the memory controller.

(d) <INTRSC>

The <INTRSC> bit is used only for Reed-Solomon codes. When using Hamming codes, this bit should be set to “0”.

This bit is used to enable or disable the interrupt to be generated when the calculation of error address and error bit position has ended.

The interrupt is enabled when this bit is set to “1” and disabled when “0”.

(e) <INTRDY>

The <INTRDY> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to enable or disable the interrupt to be generated when the status of the NDR/B pin of the NAND Flash changes from “busy” (0) to “ready” (1). The interrupt is enabled when this bit is set to “1” and disabled when “0”.

(f) <STATE3:0>、<SEER1:0>

The <STATE3:0> and <SEER1:0> bits are used only for Reed-Solomon codes. When using Hamming codes, they have no meaning.

These bits are used as flags to indicate the result of error address and error bit calculation. For details, see Table3.12.2.

NAND Flash Data Register 0

		7	6	5	4	3	2	1	0
NDFDTR0 (1FF0H)	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	R/W							
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (7-0)							
		15	14	13	12	11	10	9	8
(1FF1H)	bit Symbol	D15	D14	D13	D12	D11	D10	D9	D8
	Read/Write	R/W							
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (15-8)							

NAND Flash Data Register 1

		7	6	5	4	3	2	1	0
NDFDTR1 (1FF2H)	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	R/W							
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (7-0)							
		15	14	13	12	11	10	9	8
(1FF3H)	bit Symbol	D15	D14	D13	D12	D11	D10	D9	D8
	Read/Write	R/W							
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
	Function	NAND Flash Data Register (15-8)							

Note: Although these registers allow both read and write operations, no flip-flop is incorporated. Since write and read operations are performed in different manners, it is not possible to read out the data that has been just written.

Figure3.12.6 NAND Flash Data Registers (NDFDTR0, NDFDTR1)

Write and read operations to and from the NAND Flash memory are performed by accessing the NDFDTR0 register. When you write to this register, the data is written to the NAND Flash. When you read from this register, the data is read from the NAND Flash. The NDFDTR0 register is used for both channel 0 and channel 1.

A total of 4 bytes are provided as data registers to enable 4-byte DMA transfer. For example, 4 bytes of data can be transferred from 32-bit internal RAM to 8-bit NAND Flash memory by DMA operation by setting the destination address as NDFDTR0. (NDFDTR1 cannot be set as the destination address.) The actual DMA operation is performed by first reading 4 bytes from the internal RAM and then writing 1 byte to the NAND Flash four times from the lowest address.

To access data in the NAND Flash, be sure to access NDFDTR0 (at address 1FF0). For details, see Table3.12.3.

Table3.12.3 How to Access the NAND Flash Data Register

Write

Access Data Size	Example of instruction	8-bit NAND Flash	16-bit NAND Flash
1-byte access	ld (0x1FF0),a	Supported	Not supported
2-byte access	ld (0x1FF0),wa	Supported	Supported
4-byte access	ld (0x1FF0),xwa	Supported	Supported

Read

Access Data Size	Example of instruction	8-bit NAND Flash	16-bit NAND Flash
1-byte access	ld a,(0x1FF0)	Supported	Not supported
2-byte access	ld wa,(0x1FF0)	Supported	Supported
4-byte access	ld xwa,(0x1FF0)	Supported	Supported

NAND Flash ECC Register 0

		7	6	5	4	3	2	1	0
NDECCRD0 (08C4H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C5H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 1

		7	6	5	4	3	2	1	0
NDECCRD1 (08C6H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C7H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 2

		7	6	5	4	3	2	1	0
NDECCRD2 (08C8H)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08C9H)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 3

		7	6	5	4	3	2	1	0
NDECCRD3 (08CAH)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08CBH)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

NAND Flash ECC Register 4

		7	6	5	4	3	2	1	0
NDECCRD4 (08CCH)	bit Symbol	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (7-0)							
		15	14	13	12	11	10	9	8
(08CDH)	bit Symbol	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash ECC Register (15-8)							

Figure3.12.7 NAND Flash ECC Registers

The NAND Flash ECC register is used to read ECC generated by the ECC generator.

After valid data has been written to or read from the NAND Flash, setting NDFMCR0<ECCE> to “0” causes the corresponding ECC to be set in this register. (The ECC in this register is updated when NDFMCR0<ECCE> changes from “1” to “0”.)

When Hamming codes are used, 22 bits of ECC are generated for up to 256 bytes of valid data. In the case of Reed-Solomon codes, 80 bits of ECC are generated for up to 518 bytes of valid data. A total of 80 bits of registers are provided, arranged as five 16-bit registers. These registers must be read in 16-bit units and cannot be accessed in 32-bit units.

After ECC calculation has completed, in the case of Hamming codes, the 16-bit line parity for the first 256 bytes is stored in the NDECCRD0 register, the 6-bit column parity for the first 256 bytes in the NDECCRD1 register (<ECCE7:2>), the 16-bit line parity for the second 256 bytes in the NDECCRD2 register, and the 6-bit column parity for the second 256 bytes in the NDECCRD3 register (<ECCD7:2>). In this case, the NDECCRD4 register is not used.

In the case of Reed-Solomon codes, 80 bits of ECC are stored in the NDECCRD0, NDECCRD1, NDECCRD2, NDECCRD3 and NDECCRD4 registers.

Note: Before reading ECC from the NAND Flash ECC register, be sure to set NDFMCR0<ECCE> to “0”. The ECC in the NAND Flash ECC register is updated when NDFMCR0<ECCE> changes from “1” to “0”. Also note that when the ECC in the ECC generator is reset by NDFMCR0<ECCRST>, the contents of this register are not reset.

Register Name	Hamming	Reed-Solomon
NDECCRD0	[15:0] Line parity (for the first 256 bytes)	[15:0] Reed-Solomon ECC code 79:64
NDECCRD1	[7:2] Column parity (for the first 256 bytes)	[15:0] Reed-Solomon ECC code 63:48
NDECCRD2	[15:0] Line parity (for the second 256 bytes)	[15:0] Reed-Solomon ECC code 47:32
NDECCRD3	[7:2] Column parity (for the second 256 bytes)	[15:0] Reed-Solomon ECC code 31:16
NDECCRD4	Not in use	[15:0] Reed-Solomon ECC code 15:0

The table below shows an example of how ECC is written to the redundant area in the NAND Flash memory when using Reed-Solomon codes.

When using Hamming codes with SmartMedia™, the addresses of the redundant area are specified by the physical format of SmartMedia™. For details, refer to the SmartMedia™ Physical Format Specifications.

Register Name	Reed-Solomon	NAND Flash Address
NDECCRD0	[15:0] Reed-Solomon ECC code 79:64	Upper 8 bits [79:72]→ address 518 Lower 8 bits [71:64] → address 519
NDECCRD1	[15:0] Reed-Solomon ECC code 63:48	Upper 8 bits [63:56] → address 520 Upper 8 bits [55:48] → address 521
NDECCRD2	[15:0] Reed-Solomon ECC code 47:32	Upper 8 bits [47:40] → address 522 Lower 8 bits [39:32] → address 523
NDECCRD3	[15:0] Reed-Solomon ECC code 31:16	Upper 8 bits [31:24] → address 524 Lower 8 bits [23:16] → address 525
NDECCRD4	[15:0] Reed-Solomon ECC code 15:0	Upper 8 bits [15:8] → address 526 Lower 8 bits [7:0] → address 527

NAND Flash Reed-Solomon Calculation Result Address Register

		7	6	5	4	3	2	1	0
NDRSCA0 (08D0H)	bit Symbol	RS0A7	RS0A6	RS0A5	RS0A4	RS0A3	RS0A2	RS0A1	RS0A0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D1H)	bit Symbol							RS0A9	RS0A8
	Read/Write							R	
	Reset State							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA1 (08D4H)	bit Symbol	RS1A7	RS1A6	RS1A5	RS1A4	RS1A3	RS1A2	RS1A1	RS1A0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D5H)	bit Symbol							RS1A9	RS1A8
	Read/Write							R	
	Reset State							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA2 (08D8H)	bit Symbol	RS2A7	RS2A6	RS2A5	RS2A4	RS2A3	RS2A2	RS2A1	RS2A0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08D9H)	bit Symbol							RS2A9	RS2A8
	Read/Write							R	
	Reset State							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
		7	6	5	4	3	2	1	0
NDRSCA3 (08DCH)	bit Symbol	RS3A7	RS3A6	RS3A5	RS3A4	RS3A3	RS3A2	RS3A1	RS3A0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		15	14	13	12	11	10	9	8
(08DDH)	bit Symbol							RS3A9	RS3A8
	Read/Write							R	
	Reset State							0	0
	Function							NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	

Figure3.12.8 NAND Flash Reed-Solomon Calculation Result Address Register

If error is found at only one address, the error address is stored in the NDRSCA0 register. If error is found at two addresses, the NDRSCA0 and NDRSCA1 registers are used to store the error addresses. In this manner, up to four error addresses can be stored in the NDRSCA0 to NDRSCA3 registers.

The number of error addresses can be checked by NDFMCR1<SEER1:0>.

NAND Flash Reed-Solomon Calculation Result Data Register

		7	6	5	4	3	2	1	0
NDRSCD0 (08D2H)	bit Symbol	RS0D7	RS0D6	RS0D5	RS0D4	RS0D3	RS0D2	RS0D1	RS0D0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD1 (08D6H)	bit Symbol	RS1D7	RS1D6	RS1D5	RS1D4	RS1D3	RS1D2	RS1D1	RS1D0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD2 (08DAH)	bit Symbol	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
		7	6	5	4	3	2	1	0
NDRSCD3 (08DEH)	bit Symbol	RS3D7	RS3D6	RS3D5	RS3D4	RS3D3	RS3D2	RS3D1	RS3D0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							

Figure3.12.9 NAND Flash Reed-Solomon Calculation Result Data Register

If error is found at only one address, the error data is stored in the NDRSCD0 register. If error is found at two addresses, the NDRSCD0 and NDRSCD1 registers are used to store the error data. In this manner, the error data at up to four addresses can be stored in the NDRSCD0 to NDRSCD3 registers.

The number of error addresses can be checked by NDFMCR1<SEER1:0>.

3.12.6 An Example of Accessing NAND Flash of SLC Type

1. Initialization

```
;
; ***** Initialize NDFC *****
;        Conditions: 8-bit bus, CE0, SLC, 512 (528) bytes/page, Hamming codes
;
         ld      (ndfmcr1),0001h   ; 8-bit bus, Hamming ECC, SYSCK-ON
         ld      (ndfmcr0),2000h   ; SPLW1:0=0, SPHW1:0=2
```

2. Write

Writing valid data

```
; ***** Write valid data*****
;
         ldw     (ndfmcr0),2010h   ; CE0 enable
         ldw     (ndfmcr0),20B0h   ; WE enable, CLE enable
         ld      (ndfdtr0),80h     ; Serial input command
         ldw     (ndfmcr0),20D0h   ; ALE enable
         ld      (ndfdtr0),xxh     ; Address write (3 or 4 times)
         ldw     (ndfmcr0),2095h   ; Reset ECC, ECCE enable, CE0 enable
         ld      (ndfdtr0),xxh     ; Data write (512 times)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
         ldw     (ndfmcr0),2010h   ; ECC circuit disable

         ldw     xxxx,(ndeccrd0)   ; Read ECC from internal circuit
;                1'st Read:        D15-0 > LPR15:0            For first 256 bytes
         ldw     xxxx,(ndeccrd1)   ; Read ECC from internal circuit
;                2'nd Read:        D15-0 > FFh+CPR5:0+11b     For first 256 bytes
         ldw     xxxx,(ndeccrd0)   ; Read ECC from internal circuit
;                3'rd Read:        D15-0 > LPR15:0            For second 256 bytes
         ldw     xxxx,(ndeccrd1)   ; Read ECC from internal circuit
;                4'th Read:        D15-0 > FFh+CPR5:0+11b     For second 256 bytes
```

Writing ECC to NAND Flash

```
; ***** Write dummy data & ECC*****
;
         ldw     (ndfmcr0),2090h   ; ECC circuit disable, data write mode
         ld      (ndfdtr0),xxh     ; Redundancy area data write (16 times)
;                Write to D520:    LPR7:0        > D7-0   For second 256 bytes
;                Write to D521:    LPR15:8       > D7-0   For second 256 bytes
;                Write to D522:    CPR5:0+11b    > D7-0   For second 256 bytes
;                Write to D525:    LPR7:0        > D7-0   For first 256 bytes
;                Write to D526:    LPR15:8       > D7-0   For first 256 bytes
;                Write to D527:    CPR5:0+11b    > D7-0   For first 256 bytes
```

Executing page program

```
; ***** Set auto page program*****
;
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),10h    ; Auto page program command
        ldw     (ndfmcr0),2010h  ; WE disable, CLE disable
;
;       Wait setup time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;
```

Reading status

```
; ***** Read Status*****
;
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),70h    ; Status read command
        ldw     (ndfmcr0),2010h  ; WE disable, CLE disable
        ld      xx,(ndfdtr0)     ; Status read
```

3. Read

Reading valid data

```
; ***** Read valid data*****
;
        ldw     (ndfmcr0),2010h  ; CE0 enable
        ldw     (ndfmcr0),20B0h  ; WE enable, CLE enable
        ld      (ndfdtr0),00h    ; Read command
        ldw     (ndfmcr0),20D0h  ; ALE enable
        ld      (ndfdtr0),xxh    ; Address write (3 or 4 times)
;
;       Wait setup time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;
        ldw     (ndfmcr0),2015h  ; Reset ECC, ECCE enable, CE0 enable
        ld      xx,(ndfdtr0)     ; Data read (512 times)
        ldw     (ndfmcr0),2010h  ; ECC circuit disable
        ld      xx,(ndfdtr0)     ; Redundancy data read (8 times)
        ld      xx,(ndfdtr0)     ; ECC data read (3 times)
        ld      xx,(ndfdtr0)     ; Redundancy data read (2 times)
        ld      xx,(ndfdtr0)     ; ECC data read (3 times)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
        ldw     (ndfmcr0),2010h  ; ECC circuit disable
        ldw     xxxx,(ndeccrd0)  ; Read ECC from internal circuit
;               1'st Read:       D15-0 > LPR15:0          For first 256 bytes
        ldw     xxxx,(ndeccrd1)  ; Read ECC from internal circuit
;               2'nd Read:       D15-0 > FFh+CPR5:0+11b   For first 256 bytes
        ldw     xxxx,(ndeccrd0)  ; Read ECC from internal circuit
;               3'rd Read:       D15-0 > LPR15:0          For second 256 bytes
        ldw     xxxx,(ndeccrd1)  ; Read ECC from internal circuit
;               4'th Read:       D15-0 > FFh+CPR5:0+11b   For second 256 bytes
```

Software processing

The ECC data generated for the read operation and the ECC in the redundant area in the NAND Flash are compared. If any error is found, the error processing routine is performed to correct the error data. For details, see 3.12.4.2 "Error Correction Methods".

4. ID Read

The ID read routine is as follows:

```
ldw    (ndfmcr0),20B0h   ; WE Enable, CLE enable
ld     (ndfdtr0),90h     ; Write ID read command
ldw    (ndfmcr0),20D0h   ; ALE enable, CLE disable
ld     (ndfdtr0),00h     ; Write 00
ldw    (ndfmcr0),2010h   ; WE disable, CLE disable
ld     xx,(ndfdtr0)      ; Read 1'st ID maker code
ld     xx,(ndfdtr0)      ; Read 2'nd ID device code
```

3.12.7 An Example of Accessing NAND Flash of MLC Type (When the valid data is processed as 518byte)

1. Initialization

```
;
; ***** Initialize NDFC *****
;           Conditions: 16-bit bus, CE1, MLC, 2048 (2112) bytes/page, Reed-Solomon codes
;
            ld      (ndfmcr1),0007h   ; 16-bit bus, Reed-Solomon ECC, SYSCK-ON
            ld      (ndfmcr0),5000h   ; SPLW1:0=1, SPHW1:0=1
```

2. Write

Writing valid data

```
; ***** Write valid data*****
;
            ldw     (ndfmcr0),5008h   ; CE1 enable
            ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
            ldw     (ndfdtr0),0080h   ; serial input command
            ldw     (ndfmcr0),50C8h   ; ALE enable
            ldw     (ndfdtr0),00xxh   ; Address write ( 4 or 5 times)
            ldw     (ndfmcr0),508Dh   ; Reset ECC code, ECCE enable
            ldw     (ndfdtr0),xxxxh   ; Data write (259-times/:518byte)
                                                    (256-times/512byte)
```

Generating ECC → Reading ECC

```
; ***** Read ECC *****
;
            ldw     (ndfmcr0),5008h   ; ECC circuit disable
            ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
            ldw     (ndfdtr0),0080h   ; serial input command
            ldw     (ndfmcr0),50C8h   ; ALE enable
            ldw     (ndfdtr0),00xxh   ; Address write ( 4 or 5 times)

            ldw     xxxx,(ndeccrd0)   ; Read ECC from internal circuit
;                   Read:   D79-64
            ldw     xxxx,(ndeccrd1)   ; Read ECC from internal circuit
;                   Read:   D63-48
            ldw     xxxx,(ndeccrd2)   ; Read ECC from internal circuit
;                   Read:   D47-32
            ldw     xxxx,(ndeccrd3)   ; Read ECC from internal circuit
;                   Read:   D31-16
            ldw     xxxx,(ndeccrd4)   ; Read ECC from internal circuit
;                   Read:   D15-0
```

Writing ECC to NAND Flash

```
; ***** Write dummy data & ECC *****
;
        ldw     (ndfmcr0),5088h  ; ECC circuit disable, data write mode
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 207-206hex address:      > D79-64
        ldw     (ndfdtr1),xxxxh  ; Redundancy area data write
;               Write to 209-208hex address:      > D63-48
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 20B-20Ahex address:      > D47-32
        ldw     (ndfdtr1),xxxxh  ; Redundancy area data write
;               Write to 20D-20Chex address:      > D31-16
        ldw     (ndfdtr0),xxxxh  ; Redundancy area data write
;               Write to 20F-20Ehex address:      > D15-0
;
;       The write operation is repeated four times to write 2112 bytes.
```

Executing page program

```
; ***** Set auto page program*****
;
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0010h  ; Auto page program command
        ldw     (ndfmcr0),5008h  ; WE disable, CLE disable
;
;       Wait set up time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
```

Note: In case of LB type NANDF, programming page size is normally each 2112 bytes and ECC calculation is processed each 518 (512) bytes. Please take care of programming flow. In details, refer the NANDF memory specifications.

Reading status

```
; ***** Read status*****
;
        ldw     (ndfmcr0),50A8h  ; WE enable, CLE enable
        ldw     (ndfdtr0),0070h  ; Status read command
        ldw     (ndfmcr0),5008h  ; WE disable, CLE disable
        ldw     xxxx,(ndfdtr0)   ; Status read
```

3. Read (including ECC data read)

Reading valid data

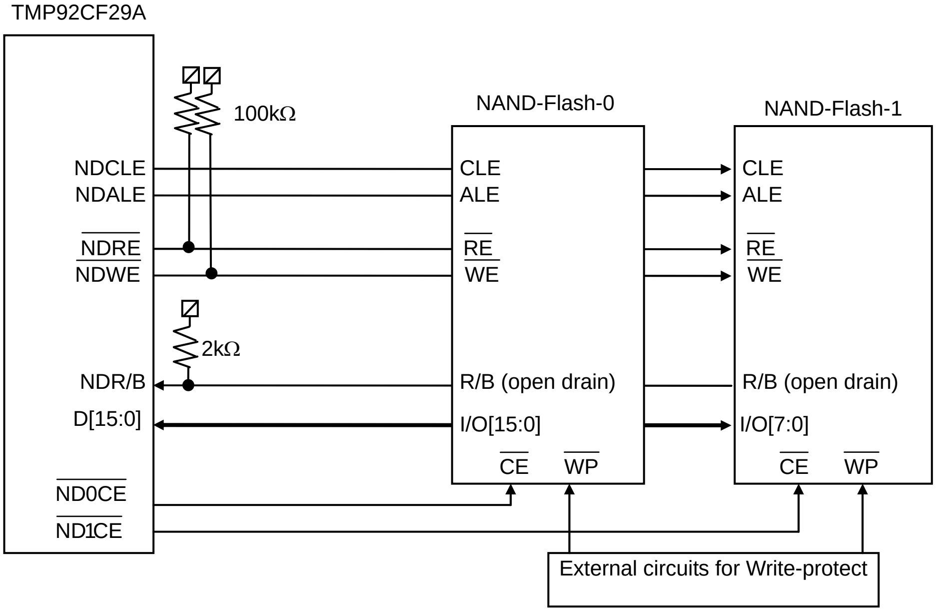
```
; ***** Read valid data*****
;
        ldw     (ndfmcr0),5008h   ; CE1 enable
        ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
        ldw     (ndfdtr0),0000h   ; Read command 1
        ldw     (ndfmcr0),50C8h   ; ALE enable
        ldw     (ndfdtr0),00xxh   ; Address write (4 or 5 times)
        ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
        ldw     (ndfdtr0),0030h   ; Read command 2
;
;       Wait set up time (from Busy to Ready)
;               1. Flag polling
;               2. Interrupt
;
        ldw     (ndfmcr0),540Dh   ; ECC reset, ECC circuit enable, decode mode
        ldw     xxxx,(ndfdtr0)    ; Data read (259 times: 518 bytes)
                                            (256-times:512 byte)
        ldw     (ndfmcr0),550Ch   ; RSECGW enable
        ldw     xxxx,(ndfdtr0)    ; Read ECC (5 times: 80 bits)
;
;       Wait set up time (20 system clocks)
;
(1) Error bit calculation
        ldw     (ndfmcr1),0047h   ; Error bit calculation interrupt enable
        ldw     (ndfmcr0),560Ch   ; Error bit calculation circuit start
;
;       Wait set up time
;       Interrupt routine (End of calculation for Reed-Solomon Error bit)
;
INT:    ldw     xxxx,(ndfmcr1)    ; Check error status "STATE3:0, SEER1:0"
;
;       If error is found, the error processing routine is performed to
;       correct the error data. For details see 3.12.4.2 "Error Correction
;       Methods".
;
;       The read operation is repeated four times to read 2112 bytes.
;
```

4. ID Read

The ID read routine is as follows:

```
ldw     (ndfmcr0),50A8h   ; WE enable, CLE enable
ldw     (ndfdtr0),0090h   ; Write ID read command
ldw     (ndfmcr0),50C8h   ; ALE enable, CLE disable
ldw     (ndfdtr0),0000h   ; Write 00
ldw     (ndfmcr0),5008h   ; WE disable, CLE disable
ldw     xxxx,(ndfdtr0)    ; Read 1'st ID maker code
ldw     xxxx,(ndfdtr1)    ; Read 2'ndID device code
```

3.12.8 An Example of Connections with NAND Flash



Note 1: A reset sets the $\overline{\text{NDRE}}$ and $\overline{\text{NDWE}}$ pins as input ports, so pull-up resistors are needed.

Note 2: The pull-up resistor value for the NDR/B pin must be set appropriately according to the NAND Flash memory to be used and the capacity of the board (typical: 2 kΩ).

Note 3: The $\overline{\text{WP}}$ (Write Protect) pin of NAND Flash is not supported. When this function is needed, prepare it on an external circuit.

Figure3.12.10 An Example of Connections with NAND Flash

3.13 8 Bit Timer (TMRA)

The TMP92CF29A features 8 channel built-in 8-bit timers (TMRA0 to TMRA7).

These timers are paired into 4 modules: TMRA01, TMRA23, TMRA45 and TMRA67. Each module consists of 2 channels and can operate in any of the following 4 operating modes.

- 8-bit interval timer mode
- 16-bit interval timer mode
- 8-bit programmable square wave pulse generation output mode (PPG: Variable duty cycle with variable period)
- 8-bit pulse width modulation output mode (PWM – Variable duty cycle with constant period)

Figure 3.13.1 to Figure 3.13.4 show block diagrams for TMRA01 to TMRA67.

Each channel consists of an 8-bit up counter, an 8-bit comparator and an 8-bit timer register. In addition, a timer flip-flop and a prescaler are provided for each pair of channels.

The operation mode and timer flip-flops are controlled by a 5bytes registers SFRs (Special-function registers).

Each of the 4 modules (TMRA01 to TMRA67) can be operated independently. All modules operate in the same manner; hence only the operation of TMRA01 is explained here.

The contents of this chapter are as follows.

Table 3.13.1 Registers and Pins for Each Module

Specification		TMRA01	TMRA23	TMRA45	TMRA67
External pin	Input pin for external clock	TA0IN (Shared with PC1)	TA2IN (Shared with PC3)	Low-frequency clock fs	Low-frequency clock fs
	Output pin for timer flip-flop	TA1OUT (Shared with PM1)	–	–	TA7OUT (Shared with PP3)
SFR (Address)	Timer run register	TA01RUN (1100H)	TA23RUN (1108H)	TA45RUN (1110H)	TA67RUN (1118H)
	Timer register	TA0REG (1102H) TA1REG (1103H)	TA2REG (110AH) TA3REG (110BH)	TA4REG (1112H) TA5REG (1113H)	TA6REG (111AH) TA7REG (111BH)
	Timer mode register	TA01MOD (1104H)	TA23MOD (110CH)	TA45MOD (1114H)	TA67MOD (111CH)
	Timer flip-flop control register	TA1FFCR (1105H)	TA3FFCR (110DH)	–	TA7FFCR (111DH)

3.13.1 Block Diagram

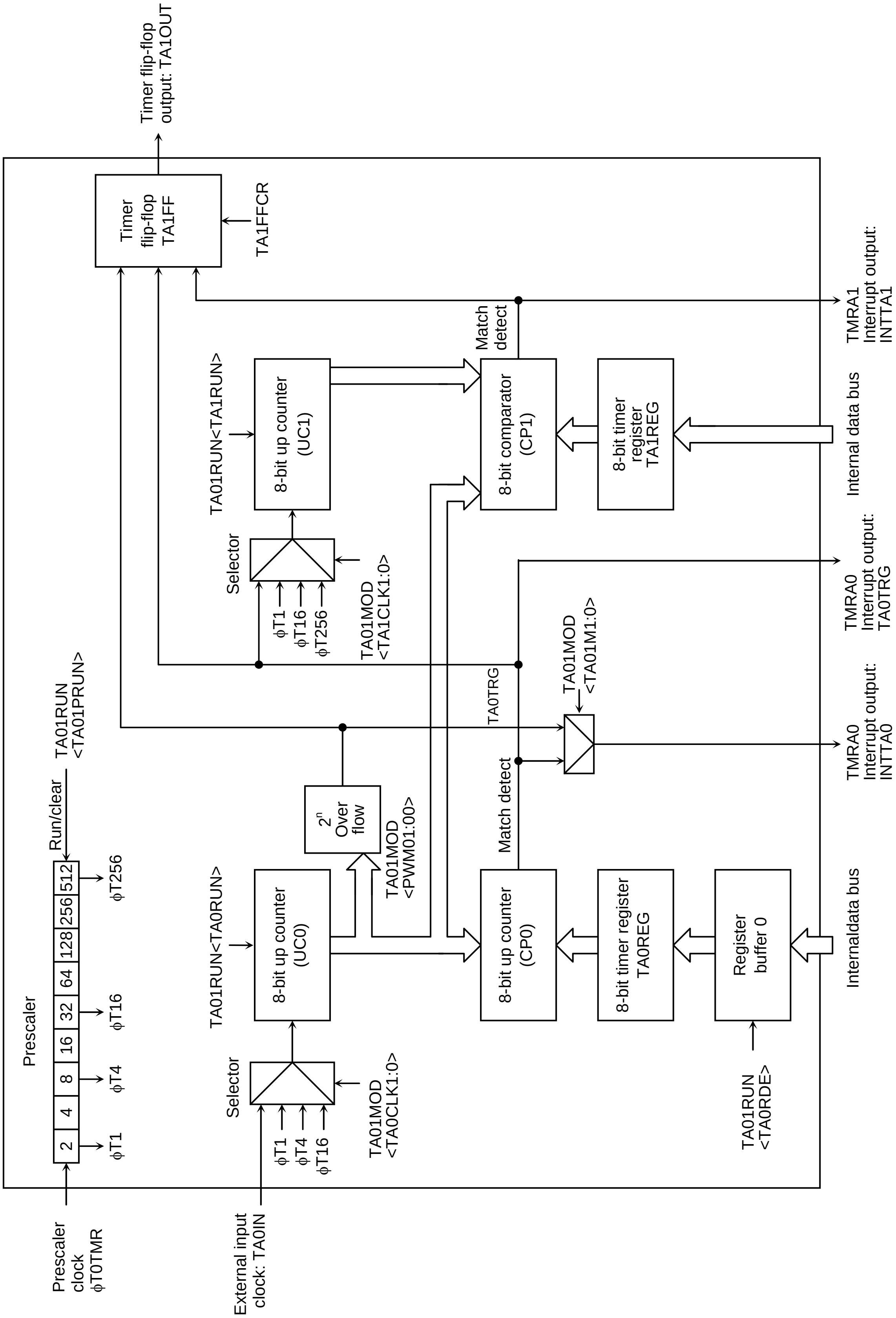


Figure 3.13.1 TMRA01 Block Diagram

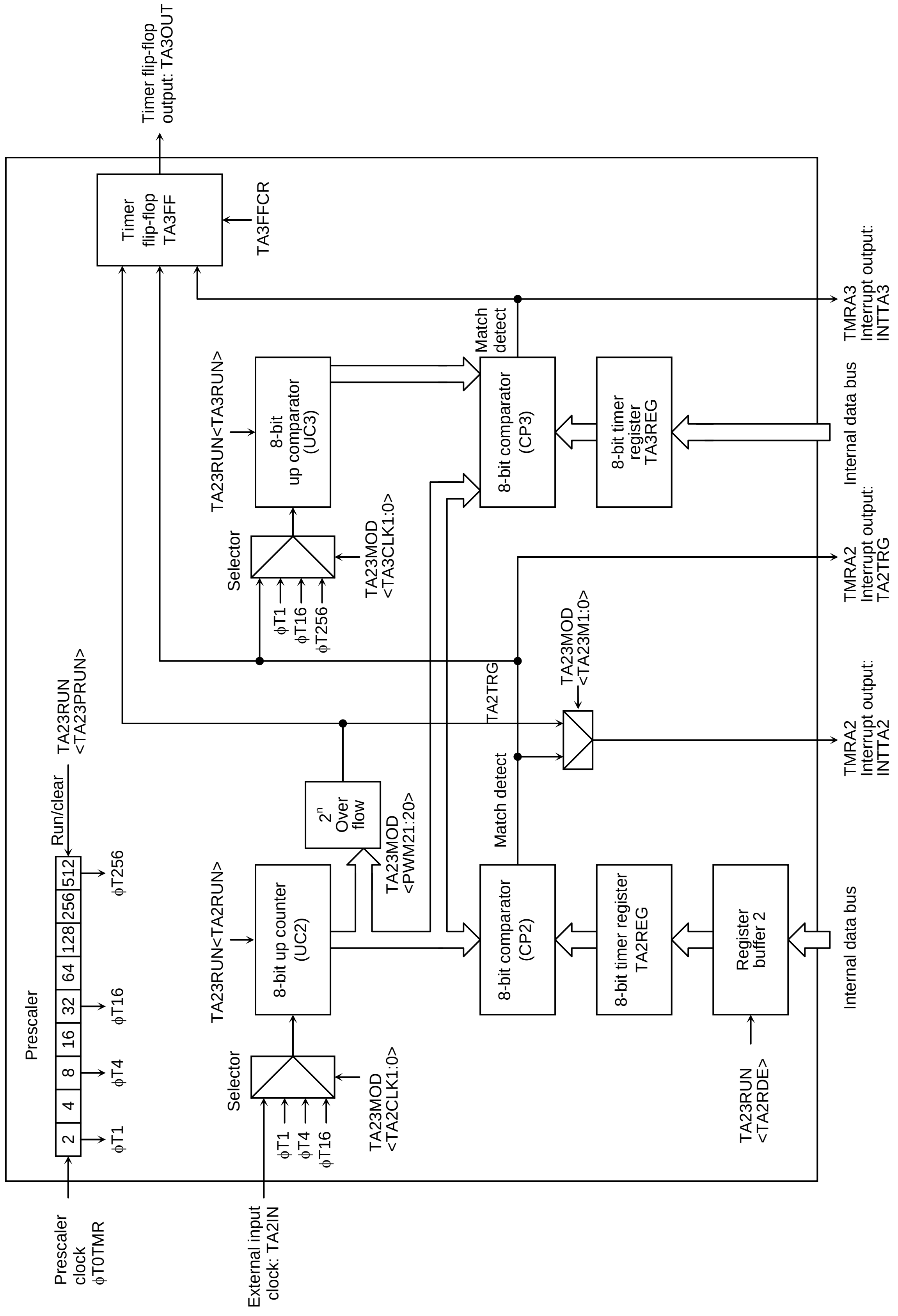


Figure 3.13.2 TMRA23 Block Diagram

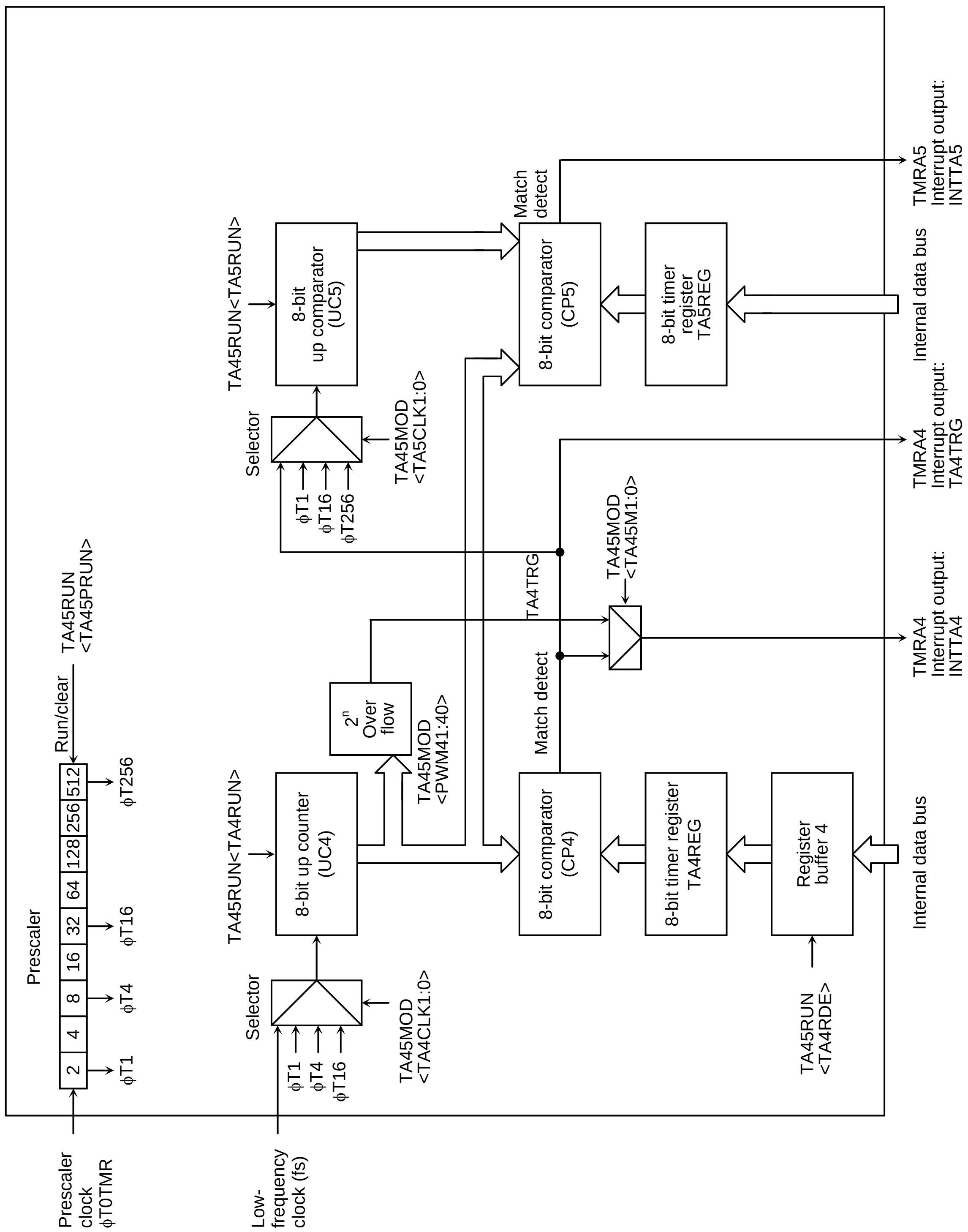


Figure 3.13.3 TMRA45 Block Diagram

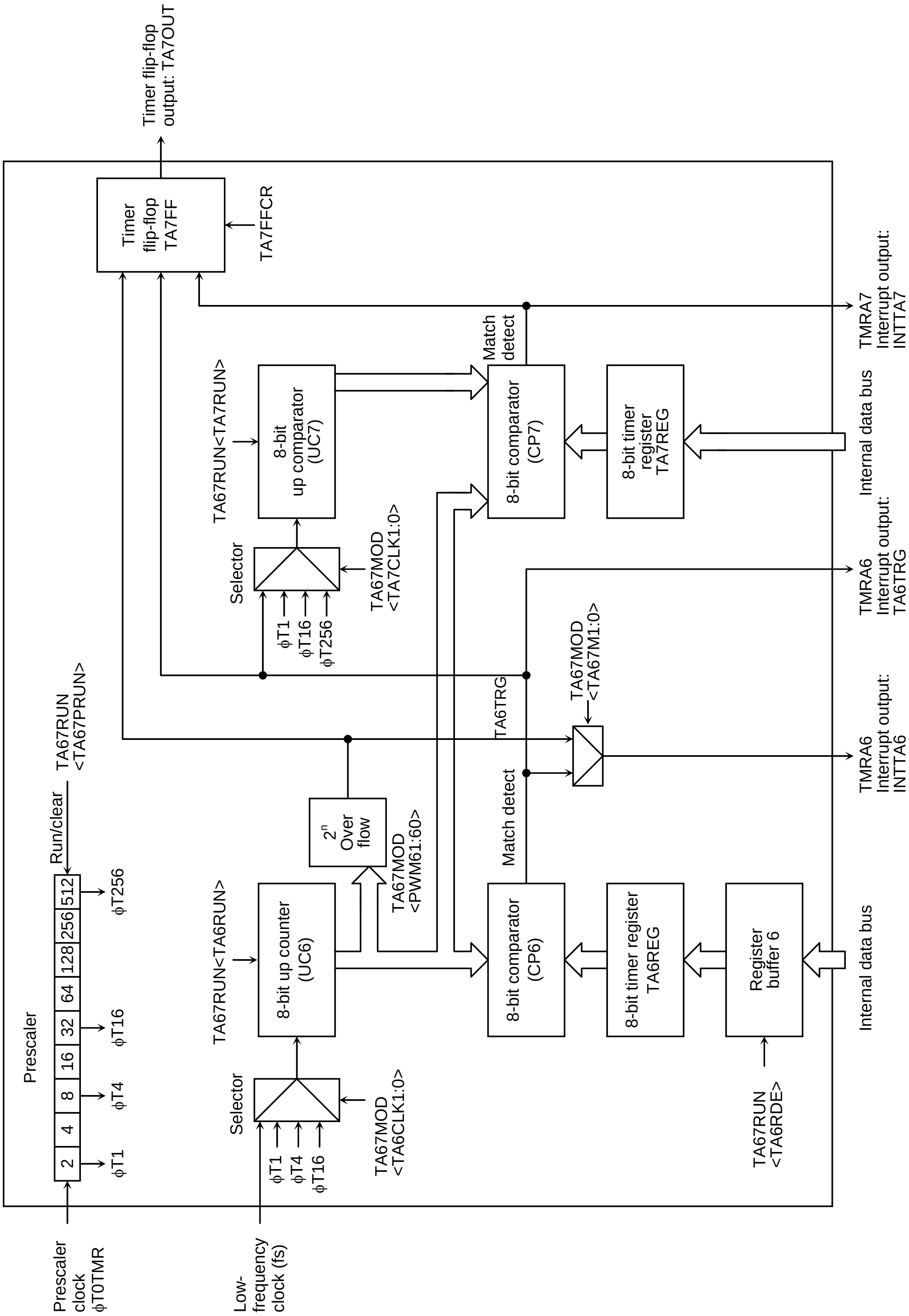


Figure 3.13.4 TMRA67 Block Diagram

3.13.2 Operation of Each Circuit

(1) Prescaler

A 9-bit prescaler generates the input clock to TMRA01.The clock φT0TMR is selected using the prescaler clock selection register SYSCR0<PRCK>.

The prescaler operation can be controlled using TA01RUN<TA0PRUN> in the timer control register. Setting <TA01PRUN> to "1" starts the count; setting <TA01PRUN> to "0" clears the prescaler to "0" and stops operation. Table 3.13.2 shows the various prescaler output clock resolutions.

(Although the prescaler and the timer counter can be started separately, the timer counter's operation depends on the prescaler's input timing.)

Table 3.13.2 Prescaler Output Clock Resolution

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>	–	Timer counter input clock Prescaler of TMRA TAxxMOD<TAxCLK1:0>			
				φT1(1/2)	φT4(1/8)	φT16(1/32)	φT256(1/512)
fc	000(1/1)	0(1/2)	1/2	fc/8	fc/32	fc/128	fc/2048
	001(1/2)			fc/16	fc/64	fc/256	fc/4096
	010(1/4)			fc/32	fc/128	fc/512	fc/8192
	011(1/8)			fc/64	fc/256	fc/1024	fc/16384
	100(1/16)			fc/128	fc/512	fc/2048	fc/32768
	000(1/1)	1(1/8)		fc/32	fc/128	fc/512	fc/8192
	001(1/2)			fc/64	fc/256	fc/1024	fc/16384
	010(1/4)			fc/128	fc/512	fc/2048	fc/32768
	011(1/8)			fc/256	fc/1024	fc/4096	fc/65536
	100(1/16)			fc/512	fc/2048	fc/8192	fc/131072

(2) Up counters (UC0 and UC1)

These are 8-bit binary counters which count up the input clock pulses for the clock specified by TA01MOD.

The input clock for UC0 is selectable and can be either the external clock input via the TA0IN pin or one of the three internal clocks φT1, φT4 or φT16. The clock setting is specified by the value set in TA01MOD<TA01CLK1:0>.

The input clock for UC1 depends on the operation mode. In 16-bit timer mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-bit timer mode, the input clock is selectable and can either be one of the internal clocks φT1, φT16 or φT256, or the comparator output (The match detection signal) from TMRA0.

For each interval timer the timer operation control register bits TA01RUN <TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up counters and to control their count. A reset clears both up counters, stopping the timers.

Note: TMR45 and TMR67 can be selected low-frequency clock(fs) instead of external clock input.

(3) Timer registers (TA0REG and TA1REG)

These are 8-bit registers, which can be used to set a time interval. When the value set in the timer register TA0REG or TA1REG matches the value in the corresponding up counter, the comparator match detect signal goes active. If the value set in the timer register is 00H, the signal goes active when the up counter overflows.

TA0REG has a double buffer structure, making a pair with the register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = "0" and enabled if <TA0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a 2^n overflow occurs in PWM mode, or at the start of the PPG cycle in PPG mode. Hence the double buffer cannot be used in timer mode.

(When using the double buffer, method of renewing timer register is only overflow in PWM mode or frequency agreement in PPG mode.)

A reset initializes <TA0RDE> to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TA0RDE> to "1", and write the following data to the register buffer. Figure 3.13.5 shows the configuration of TA0REG.

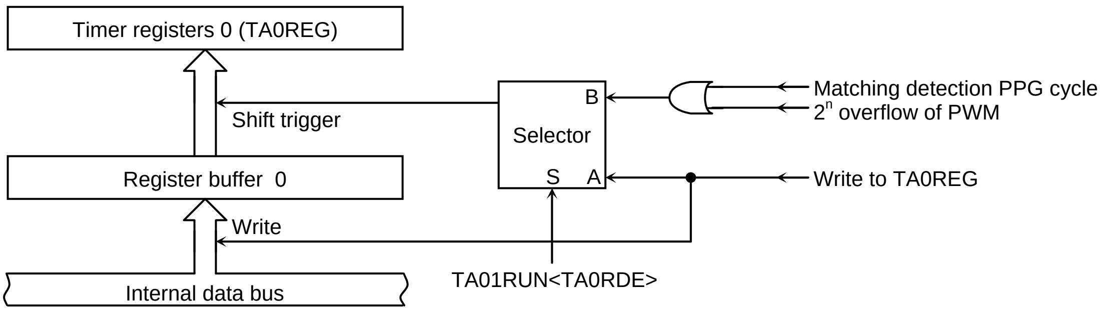


Figure 3.13.5 Configuration of timer register (TA0REG)

Note: The same memory address is allocated to the timer register and the register buffer 0. When <TA0RDE> = "0", the same value is written to the register buffer 0 and the timer register; when <TA0RDE> = "1", only the register buffer 0 is written to.

(4) Comparator (CP0, CP1)

The comparator compares the value in an up counter with the value set in a timer register. If they match, the up counter is cleared to "0" and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

Note: If a value smaller than the up-counter value is written to the timer register while the timer is counting up, this will cause the timer to overflow and an interrupt cannot be generated at the expected time. (The value in the timer register canbe changed without any problem if the new value is larger than the up-counter value.) In 16-bit interval timer mode, be sure to write to both TA0REG and TA1REG in this order (16 bits in total), The compare circuit will not function if only the lower 8 bits are set.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detect signals (8-bit comparator output) of each interval timer.

Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TA1FFIE> in the timer flip-flops control register. A reset clears the value of TA1FF to "0". Writing "01" or "10" to TA1FFCR<TA1FFC1:0> sets TA1FF to "0" or "1". Writing "00" to these bits inverts the value of TA1FF. (This is known as software inversion.)

The TA1FF signal is output via the TA1OUT pin. When this pin is used as the timer output, the timer flip-flop should be set beforehand using the port function registers.

The condition for TA1FF inversion varies with mode as shown below

8-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG (Select either one of the two)
16-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG
80bit PWM mode	: UC0 matches TA0REG or a 2^n overflow occurs
8-bit PPG mode	: UC0 matches TA0REG or UC0 matches TA1REG

Note: If an inversion by the match-detect signal and a setting change via the TMRA1 flip-flopcontrol register occur simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect signal and an attempt to set the flip-flop to 1 via the register occur simultaneously, the timer flip-flop will be set to 1.
- If an inversion by the match-detect signal and an attempt to clear the flip-flop to 0 via the register occur simultaneously the flip-flop will be cleared to 1.

Be sure to stop the timer before changing the flip-flop inversion setting.

If the setting is changed while the timer is counting, proper operation cannot be obtained.

3.13.3 SFR

TMRA01 RUN Register

TA01RUN (1100H)		7	6	5	4	3	2	1	0
	Bit symbol	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
	Read/Write	R/W				R/W			
	Reset State	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA01 prescaler	Up counter (UC1)	Up counter (UC0)
							0: Stop and clear 1: Run (Count up)		

TA0REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA01RUN are "1" when read.

TMRA23 RUN Register

TA23RUN (1108H)		7	6	5	4	3	2	1	0
	Bit symbol	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
	Read/Write	R/W				R/W			
	Reset State	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA23 prescaler	Up counter (UC3)	Up counter (UC2)
							0: Stop and clear 1: Run (Count up)		

TA3REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA23RUN are "1" when read.

Figure 3.13.6 Register for TMRA

TMRA45 RUN Register

TA45RUN (1110H)

	7	6	5	4	3	2	1	0
Bit symbol	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
Read/Write	R/W				R/W			
Reset State	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA45 prescaler	Up counter (UC5)	Up counter (UC4)
						0: Stop and clear 1: Run (Count up)		

TA4REG double buffer control

0	Disable
1	Enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA45RUN are "1" when read.

TMRA67RUN Register

TA67RUN (1118H)

	7	6	5	4	3	2	1	0
Bit symbol	TA6RDE				I2TA67	TA67PRUN	TA7RUN	TA6RUN
Read/Write	R/W				R/W			
Reset State	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				In IDLE2 mode 0: Stop 1: Operate	TMRA67 prescaler	Up counter (UC7)	Up counter (UC6)
						0: Stop and clear 1: Run (Count up)		

TA6REG double buffer control

0	Disable
1	enable

Count control

0	Stop and clear
1	Run (Count up)

Note: The values of bits 4 to 6 of TA67RUN are "1" when read.

Figure 3.13.7 Register for TMRA

TMRA01 Mode Register

		7	6	5	4	3	2	1	0
TA01MOD (1104H)	Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA0 input clock

<TA0CLK1:0>	00	TA0IN (External input)
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA1 input clock

		TA01MOD<TA01M1:0>≠01	TA01MOD<TA01M1:0>=01
<TA1CLK1:0>	00	Comparator output from TMRA0	Overflow output from TMRA0 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM01:00>	00	Reserved
	01	$2^6 \times$ Source clock
	10	$2^7 \times$ Source clock
	11	$2^8 \times$ Source clock

TMRA01 operation mode selection

<TA01MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA0), 8-bit timer (TMRA1)

Figure 3.13.8 Register for TMRA

TMRA23 Mode Register

		7	6	5	4	3	2	1	0
TA23MOD (110CH)	Bit symbol	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA3 clock for TMRA3 00: TA2TRG 01: φT1 10: φT16 11: φT256		TMRA2 clock for TMRA2 00: TA2IN pin 01: φT1 10: φT4 11: φT16	

TMRA2 input clock

<TA2CLK1:0>	00	TA2IN (External input)
	01	φT1
	10	φT4
	11	φT16

TMRA3 input clock

		TA23MOD<TA23M1:0>≠01	TA23MOD<TA23M1:0>=01
<TA3CLK1:0>	00	Comparator output from TMRA2	Overflow output from TMRA2 (16-bit timer mode)
	01	φT1	
	10	φT16	
	11	φT256	

PWM cvcle selection

<PWM21:20>	00	Reserved
	01	$2^6 \times$ Source clock
	10	$2^7 \times$ Source clock
	11	$2^8 \times$ Source clock

TMRA23 operation mode selection

<TA23MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA2), 8-bit timer (TMRA3)

Figure 3.13.9 Register for TMRA

TMRA45 Mode Register

TA45MOD (1114H)		7	6	5	4	3	2	1	0
	Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA5 clock for TMRA5 00: TA4TRG 01: φT1 10: φT16 11: φT256		TMRA4 clock for TMRA4 00: low-frequency clock 01: φT1 10: φT4 11: φT16	

TMRA4 input clock

<TA4CLK1:0>	00	low-frequency clock(fs)
	01	φT1
	10	φT4
	11	φT16

TMRA5 input clock

		TA45MOD<TA45M1:0>≠01	TA45MOD<TA45M1:0>=01
<TA5CLK1:0>	00	Comparator output from TMRA4	Overflow output from TMRA4 (16-bit timer mode)
	01	φT1	
	10	φT16	
	11	φT256	

PWM cycle selection

<PWM41:40>	00	Reserved
	01	$2^6 \times$ Source clock
	10	$2^7 \times$ Source clock
	11	$2^8 \times$ Source clock

TMRA45 operation mode selection

<TA45MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA4), 8-bit timer (TMRA5)

Figure 3.13.10 Register for TMRA

TMRA67 Mode Register

		7	6	5	4	3	2	1	0
TA67MOD (111CH)	Bit symbol	TA67M1	TA67M0	PWM61	PWM60	TA7CLK1	TA7CLK0	TA6CLK1	TA6CLK0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA7 clock for TMRA7 00: TA6TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA6 clock for TMRA6 00: low-frequency clock 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA6 input clock

<TA6CLK1:0>	00	low-frequency clock(fs)
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA1 input clock

		TA67MOD<TA67M1:0>≠01	TA67MOD<TA67M1:0>=01
<TA7CLK1:0>	00	Comparator output from TMRA6	Overflow output from TMRA6 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM61:60>	00	Reserved
	01	$2^6 \times$ Source clock
	10	$2^7 \times$ Source clock
	11	$2^8 \times$ Source clock

TMRA67 operation mode selection

<TA67MA1:0>	00	8 timer × 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA6), 8-bit timer (TMRA7)

Figure 3.13.11 Register for TMRA

TMRA1 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR	Bit symbol					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
(1105H)	Read/Write					R/W		R/W	
A read-modify-write operation cannot be performed	Reset State					1	1	0	0
	Function					00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1

Inversion signal for timer flip-flop 1 (TA1FF)
(Don't care except in 8-bit timer mode)

TA1FFIS	0	Inversion by TMRA0
	1	Inversion by TMRA1

Inversion of TA1FF

TA1FFIE	0	Disabled
	1	Enabled

Control of TA1FF

<TA1FFC1:0>	00	Inverts the value of TA1FF (Software inversion)
	01	Sets TA1FF to "1"
	10	Clears TA1FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA1FFCR are "1" when read.

Figure 3.13.12 Register for TMRA

TMRA3 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA3FFCR (110DH)	Bit symbol					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
A read-modify-write operation cannot be performed	Read/Write					R/W		R/W	
	Reset State					1	1	0	0
	Function					00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

Inversion signal for timer flip-flop 3 (TA3FF)
(Don't care except in 8-bit timer mode)

TA3FFIS	0	Inversion by TMRA2
	1	Inversion by TMRA3

Inversion of TA3FF

TA3FFIE	0	Disabled
	1	Enabled

Control of TA3FF

<TA3FFC1:0>	00	Inverts the value of TA3FF (Software inversion)
	01	Sets TA3FF to "1"
	10	Clears TA3FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA3FFCR are "1" when read.

Figure 3.13.13 Register for TMRA

TMRA7 Flip-Flop Control Register

TA7FFCR (111DH)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol					TA7FFC1	TA7FFC0	TA7FFIE	TA7FFIS
Read/Write					R/W		R/W	
Reset State					1	1	0	0
Function					00: Invert TA7FF 01: Set TA7FF 10: Clear TA7FF 11: Don't care		TA7FF control for inversion 0: Disable 1: Enable	TA7FF inversion select 0: TMRA6 1: TMRA7

Inversion signal for timer flip-flop 7 (TA7FF)
(Don't care except in 8-bit timer mode)

TA7FFIS Inversion of TA7FF	0	Inversion by TMRA6
	1	Inversion by TMRA7

TA7FFIE Control of TA7FF	0	Disabled
	1	Enabled

<TA7FFC1:0>	00	Inverts the value of TA7FF (Software inversion)
	01	Sets TA7FF to "1"
	10	Clears TA7FF to "0"
	11	Don't care

Note: The values of bits 4 to 6 of TA7FFCR are "1" when read.

Figure 3.13.14 Register for TMRA

Timer Registers

		7	6	5	4	3	2	1	0
TA0REG (1102H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA1REG (1103H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA2REG (110AH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA3REG (110BH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA4REG (1112H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA5REG (1113H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA6REG (111AH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TA7REG (111BH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							

Note: A read-modify-write operation cannot be performed for All registers.

Figure 3.13.15 TMRA Registers

3.13.4 Operation in Each Mode

(1) 8-bit timer mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

a. Generating interrupts at a fixed interval (Using TMRA1)

To generate interrupts at constant intervals using TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 20 μs at f_{SYS}= 50 MHz, set each register as follows;

* Clock state: Clcok gear : 1/1; Prescaler of clock gear :1/2

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	←	0	0	X	X	0	1	X	X	Select 8-bit timer mode and select ϕT1 (0.16 μs at f_{SYS} = 50 MHz) as the input clock.
TA1REG	←	0	1	1	1	1	1	0	1	Set TA1REG to 20 μs ÷ ϕT1 = 125(7DH)
INTETA1	←	X	1	0	1	X	–	–	–	Enable INTTA1 and set it to level 5.
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't Care, –: No change

Select the input clock using Table 3.13.2.

Note: The input clocks for TMRA0 and TMRA1 are different from as follows.
TMRA0: TA0IN input, ϕT1, ϕT4 or ϕT16.
TMRA1: Matches output of TMRA0, ϕT1, ϕT16, and ϕT256.

b. Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 3.2μs square wave pulse from the TA1OUT pin at f_{SYS} = 50 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

* Clock state — Clcok gear : 1/1; Prescaler of clock gear : 1/2

Register		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to "0".
TA01MOD	←	0	0	X	X	0	1	X	X	Select 8-bit timer mode and select φT1 (0.16 μs at f_{SYS} = 50 MHz) as the input clock.
TA1REG	←	0	0	0	0	1	0	1	0	Set the timer register to 3.2 μs ÷ φT1 ÷ 2 = 0AH
TA1FFCR	←	X	X	X	X	1	0	1	1	Clear TA1FF to "0" and set it to invert on the match detect signal from TMRA1.
PM	←	–	X	X	X	X	–	0	X	Set PM1 to function as the TA1OUT pin.
PMFC	←	–	X	X	X	X	–	1	X	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't care, –: No change

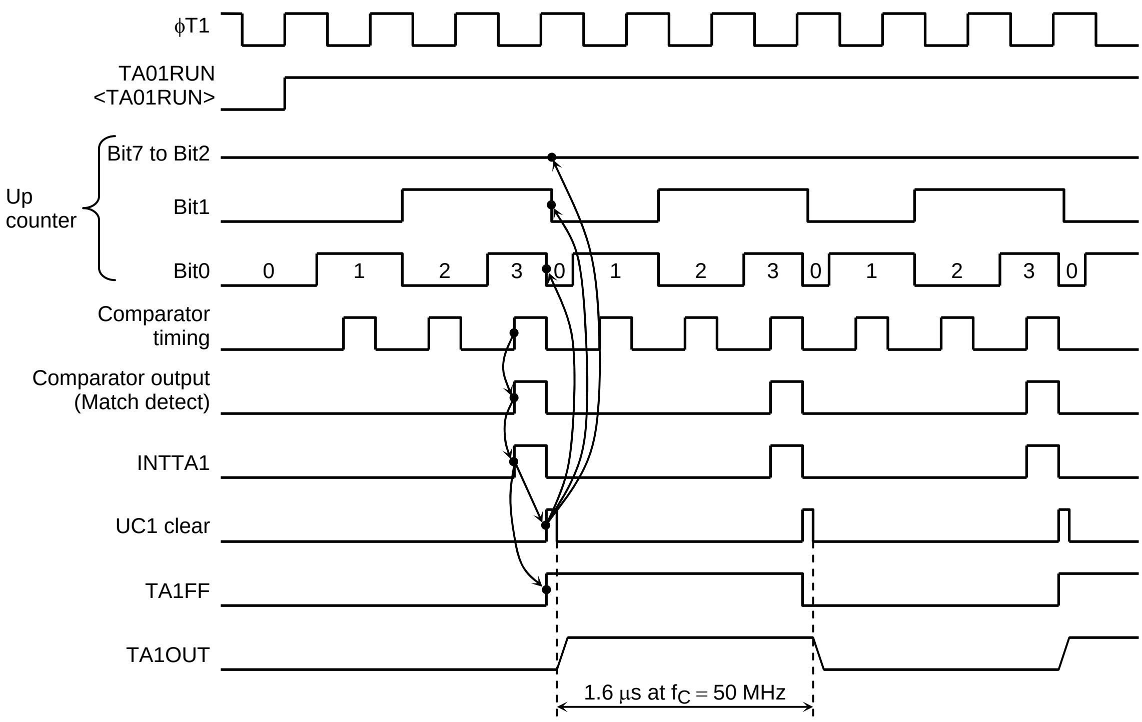


Figure 3.13.16 Square Wave Output Timing Chart (50% duty)

c. Making TMRA1 count up on the match signal from the TMRA0 comparator

Select 8-bit timer mode and set the comparator output from TMRA0 to be the input clock to TMRA1.

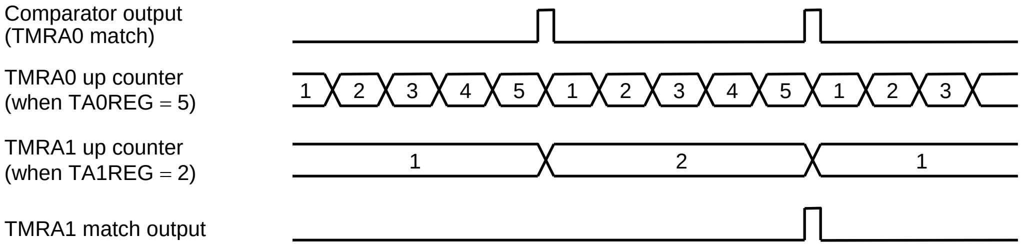


Figure 3.13.17 TMRA1 Count Up on Signal from TMRA0

(2) 16 bit timer mode

Pairing the two 8-bit timers TMRA0 and TMRA1 configures a 16-bit interval timer. To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD<TA01M1:0> to "01".

In 16-bit timer mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA01CLK1:0>. Table 3.13.2 shows the relationship between the timer (Interrupt) cycle and the input clock selection.

Example: To generate an INTTA1 interrupt every 0.13 s at f_{SYS} = 50 MHz, set the timer registers TA0REG and TA1REG as follows:

* Clock state: Clcok gear : 1/1; Prescaler of clock gear : 1/2

If ϕT16 (2.6 μs at f_{SYS} = 50 MHz) is used as the input clock for counting, set the following value in the registers: 0.13 s ÷ 2.6 μs = 50000 = C350H; e.g. set TA1REG to C3H and TA0REG to 50H.

The comparator match signal is output from TMRA0 each time the up counter UC0 matches TA0REG, though the up counter UC0 is not cleared.

In the case of the TMRA1 comparator, the match detect signal is output on each comparator pulse on which the values in the up counter UC1 and TA1REG match. When the match detect signal is output simultaneously from both the comparator TMRA0 and TMRA1, the up counters UC0 and UC1 are cleared to 0 and the interrupt INTTA1 is generated. Also, if inversion is enabled, the value of the timer flip-flop TA1FF is inverted.

Example: When TA1REG = 04H and TA0REG = 80H

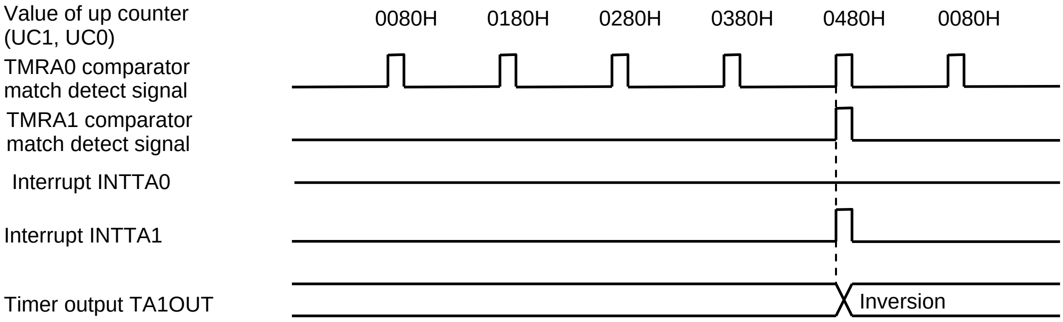


Figure 3.13.18 Timer Output by 16-Bit Timer Mode

(3) 8-bit PPG (Programmable pulse generation) output mode

Square wave pulses can be generated at any frequency and duty ratio by TMRA0. The output pulses may be active-low or active-high. In this mode TMRA1 cannot be used.

TMRA0 outputs pulses on the TA1OUT pin.

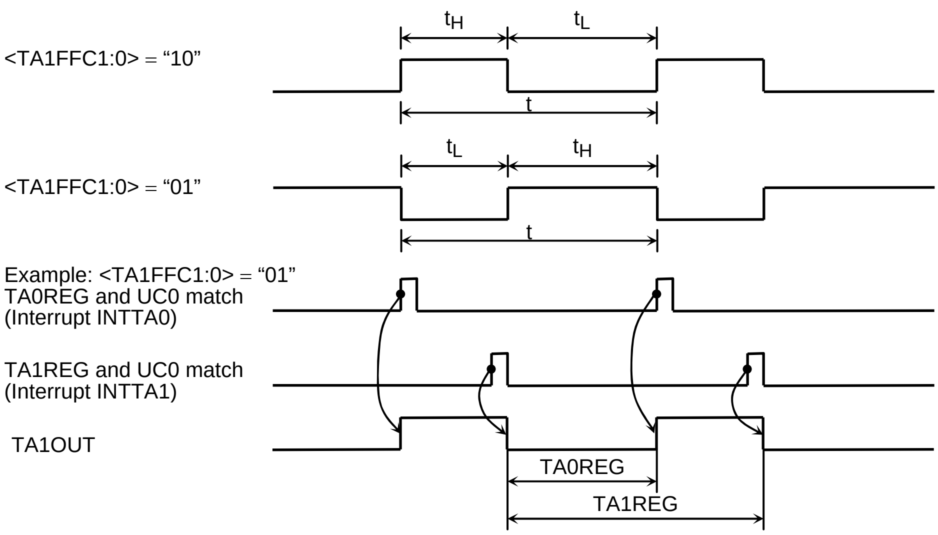


Figure 3.13.19 8-Bit PPG Output Waveforms

In this mode a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to 1 so that UC1 is set for counting.

Figure 3.13.20 shows a block diagram representing this mode.

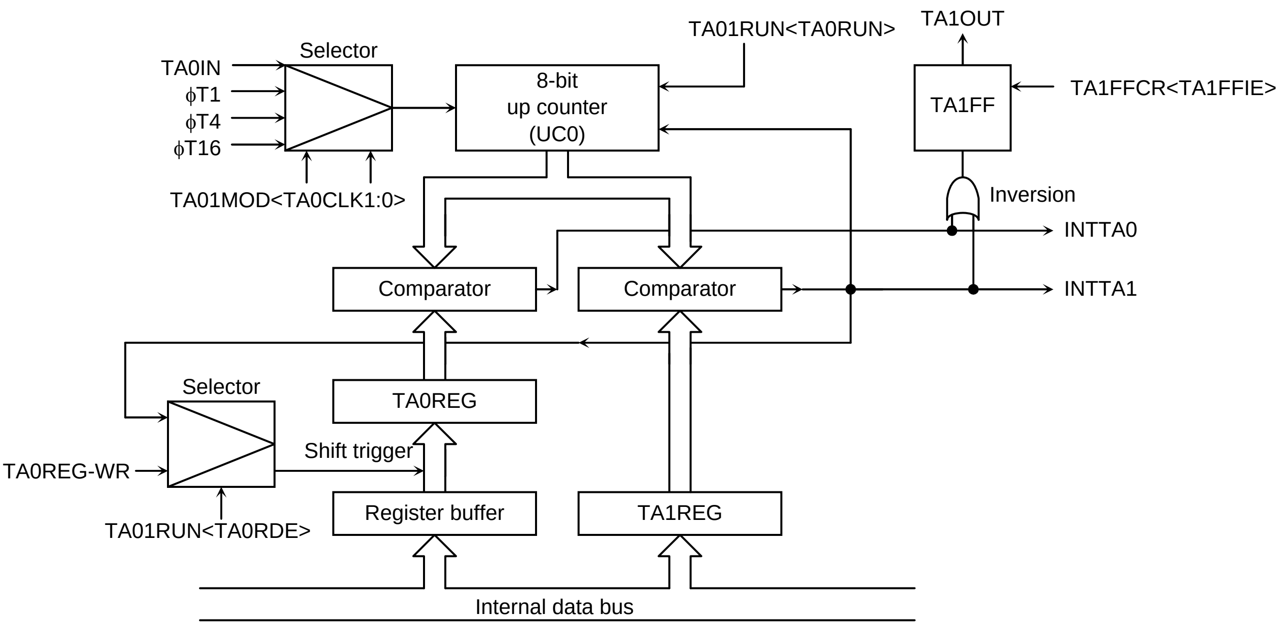


Figure 3.13.20 Block Diagram of 8-Bit PPG Output Mode

If the TA0REG double buffer is enabled in this mode, the value of the register buffer will be shifted into TA0REG each time TA1REG matches UC0.

Use of the double buffer facilitates the handling of low-duty waves (when duty is varied).

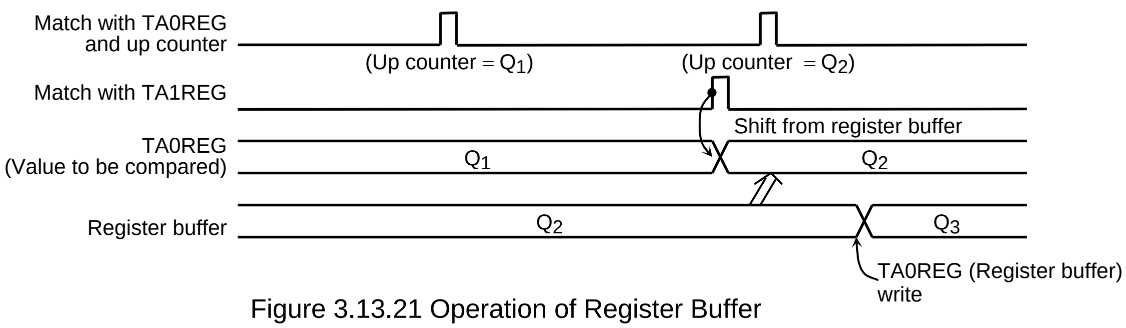
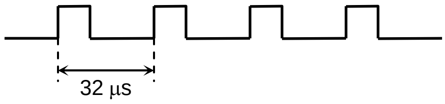


Figure 3.13.21 Operation of Register Buffer

Note: The values that can be set in TAxREG renge from 01h to 00h (equivalent to 100h). If the maximum value 00h is set , the match-detect signal goes active when the up-counter overfolws.

Example: To generate 1/4 duty 31.25 kHz pulses (at f_{SYS}= 50 MHz)



* Clock state
- Clcok gear : 1/1
- Prescaler of clock gear : 1/2

Calculate the value which should be set in the timer register.

To obtain a frequency of 31.25 kHz, the pulse cycle t should be: t = 1/31.25kHz = 32 µs

ϕT1 = 0.16 µs (at 50 MHz);

32 µs ÷ 0.16 µs = 200

Therefore set TA1REG to 200 (C8H)

The duty is to be set to 1/4: t × 1/4 = 32 µs × 1/4 = 8 µs

8 µs ÷ 0.16 µs = 50

Therefore, set TA0REG = 50 = 32H.

		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	–	0	0	Stop TMRA0 and TMRA1 and clear it to "0".
TA01MOD	←	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select ϕT1 as input clock.
TA0REG	←	0	0	0	0	1	0	1	0	Write 32H.
TA1REG	←	1	1	0	0	1	0	0	0	Write C8H.
TA1FFCR	←	X	X	X	X	0	1	1	X	Set TA1FF, enabling both inversion and the double buffer. Writing 10 provides negative logic pulse.
PM	←	–	X	X	X	X	–	0	X	Set PM1 as the TA1OUT pin.
PMFC	←	–	X	X	X	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA1 counting.

X: Don't care, –: No change

(4) 8-bit PWM (Pulse width modulation) output mode

This mode is only valid for TMRA0. In this mode, a PWM pulse with the maximum resolution of 8 bits can be output.

When TMRA0 is used the PWM pulse is output on the TA1OUT pin (Shared with PM1). TMRA1 can also be used as an 8-bit timer.

The timer output is inverted when the up counter (UC0) matches the value set in the timer register TA0REG or when 2^n counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01:00>). The up counter UC0 is cleared when 2^n counter overflow occurs.

The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < Value set for 2n counter overflow

Value set in TA0REG ≠ 0

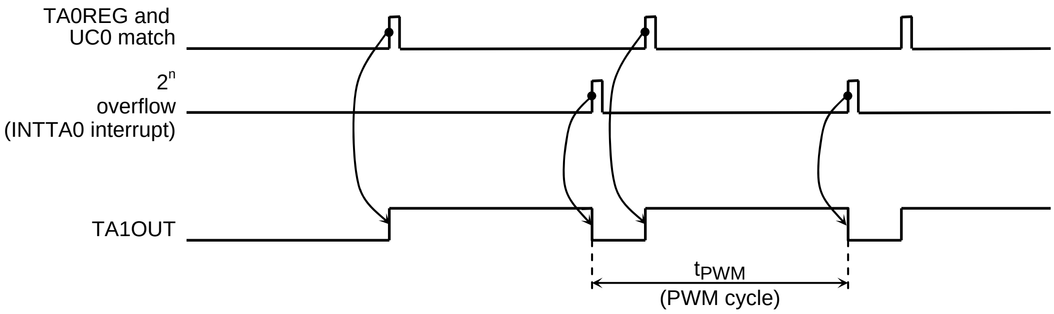


Figure 3.13.22 8-Bit PWM Waveforms

Figure 3.13.23 shows a block diagram representing this mode.

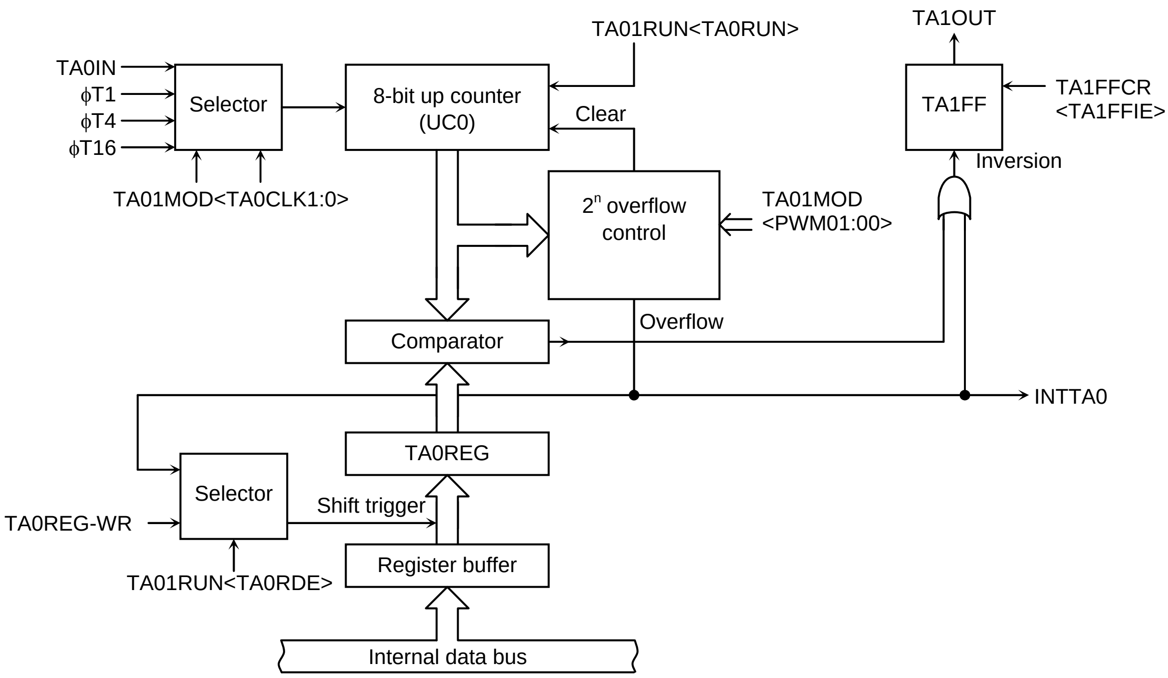


Figure 3.13.23 Block Diagram of 8-Bit PWM Mode

In this mode the value of the register buffer will be shifted into TA0REG if 2^n overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

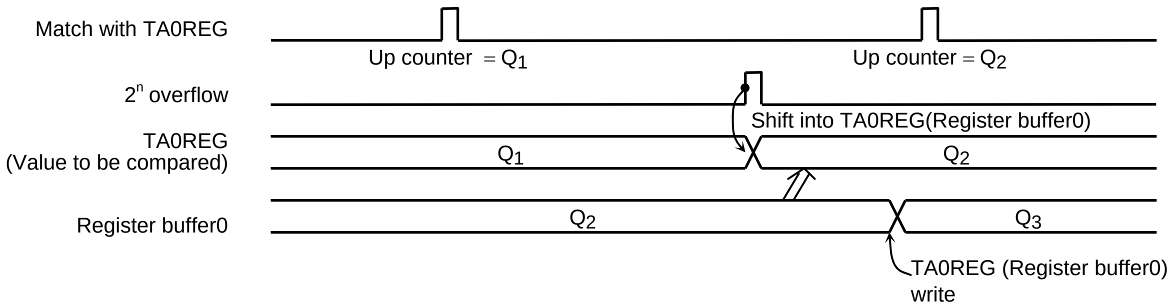
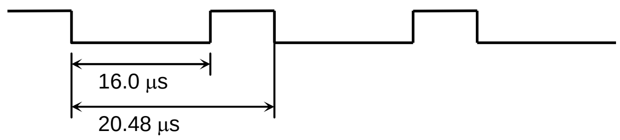


Figure 3.13.24 Register Buffer Operation

Example: To output the following PWM waves on the TA1OUT pin (at f_{SYS} = 50 MHz).



* Clock state: Clcok gear : 1/1; Prescaler of clock gear : 1/2

To achieve a 20.48μs PWM cycle by setting ϕT1 to 0.16 μs (at f_{SYS} = 50 MHz):

20.48 μs ÷ 0.16 μs = 128

2^n = 128

Therefore n should be set to 7.

Since the low level period is 16.0 μs when ϕT1 = 0.16 μs,

set the following value for TAREG:

16.0 μs ÷ 0.16 μs = 100 = 64H

		7 (MSB)	6	5	4	3	2	1	0 (LSB)	
TA01RUN	←	–	X	X	X	–	–	–	0	Stop TMRA0 and clear it to 0
TA01MOD	←	1	1	1	0	X	X	0	1	Select 8-bit PWM mode (cycle: 2^7) and select ϕT1 as the input clock.
TA0REG	←	0	1	1	0	0	1	0	0	Write 64H.
TA1FFCR	←	X	X	X	X	1	0	1	X	Clear TA1FF to 0, enable the inversion and double buffer.
PM	←	–	X	X	X	X	–	0	X	Set PM1 as the TA1OUT pin.
PMFC	←	–	X	X	X	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	–	1	Start TMRA0 counting.

X: Don't care, –: No change

Table 3.13.3 PWM Cycle

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>		PWM cycle TAxxMOD<PWMx1:0>								
				2^6 (x64)			2^7(x128)			2^8(x256)		
				TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>			TAxxMOD<TAxCLK1:0>		
				φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)	φT1(x2)	φT4(x8)	φT16(x32)
1/fc	000(x1)	0(x2)	x2	512/fc	2048/fc	8192/fc	1024/fc	4096/fc	16384/fc	2048/fc	8192/fc	32768/fc
	001(x2)			1024/fc	4096/fc	16384/fc	2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc
	010(x4)			2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc
	011(x8)			4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc
	100(x16)			8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc
	000(x1)	1(x8)		2048/fc	8192/fc	32768/fc	4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc
	001(x2)			4096/fc	16384/fc	65536/fc	8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc
	010(x4)			8192/fc	32768/fc	131072/fc	16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc
	011(x8)			16384/fc	65536/fc	262144/fc	32768/fc	131072/fc	524288/fc	65536/fc	262144/fc	1048576/fc
	100(x16)			32768/fc	131072/fc	524288/fc	65536/fc	262144/fc	1048576/fc	131072/fc	524288/fc	2097152/fc

(5) Settings for each mode

Table 3.13.4 shows the SFR settings for each mode.

Table 3.13.4 Timer Mode Setting Registers

Register Name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1:0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	TA1FFIS
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
8-bit timer × 2 channels	00	–	Lower timer match φT1, φT16, φT256 (00, 01, 10, 11)	External clock φT1, φT4, φT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	–	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PPG × 1 channel	10	–	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit PWM × 1 channel	11	2^6, 2^7, 2^8 (01, 10, 11)	–	External clock φT1, φT4, φT16 (00, 01, 10, 11)	–
8-bit timer × 1 channel	11	–	φT1, φT16, φT256 (01, 10, 11)	–	Output disabled

–: Don't care

3.14 16 bit timer / Event counter (TMRB)

The TMP92CF29A incorporates two multifunctional 16-bit timer/event counter (TMRB0, TMRB1) which have the following operation modes:

- 16 bit interval timer mode
- 16 bit event counter mode
- 16 bit programmable pulse generation mode (PPG)

Can be used following operation modes by capture function.

- Frequency measurement mode
- Pulse width measurement mode

The timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (one of them with a double-buffer structure), a 16-bit capture registers two comparators, a capture input controller, a timer flip-flop and a control circuit.

The timer/event counter is controlled by an 11-byte control SFR.

Each channel (TMRB0,TMRB1) operate independently. In this section, the explanation describes only for TMRB0 because each channel is identical operation except for the difference as follows;

Table 3.14.1 Difference between TMRB0 and TMRB1

Specification \ Channel		TMRB0	TMRB1
External pins	External clock/ capture trigger input pins	TB0IN0 (Shared with PP4)	TB1IN0 (Shared with PP5)
	Timer flip-flop output pins	TB0OUT0 (Shared with PP6)	–
SFR (Address)	Timer run register	TB0RUN (1180H)	TB1RUN (1190H)
	Timer mode register	TB0MOD (1182H)	TB1MOD (1192H)
	Timer flip-flop control register	TB0FFCR (1183H)	–
	Timer register	TB0RG0L (1188H) TB0RG0H (1189H) TB0RG1L (118AH) TB0RG1H (118BH)	TB1RG0L (1198H) TB1RG0H (1199H) TB1RG1L (119AH) TB1RG1H (119BH)
	Capture register	TB0CP0L (118CH) TB0CP0H (118DH) TB0CP1L (118EH) TB0CP1H (118FH)	TB1CP0L (119CH) TB1CP0H (119DH) TB1CP1L (119EH) TB1CP1H (119FH)

3.14.1 Block diagram

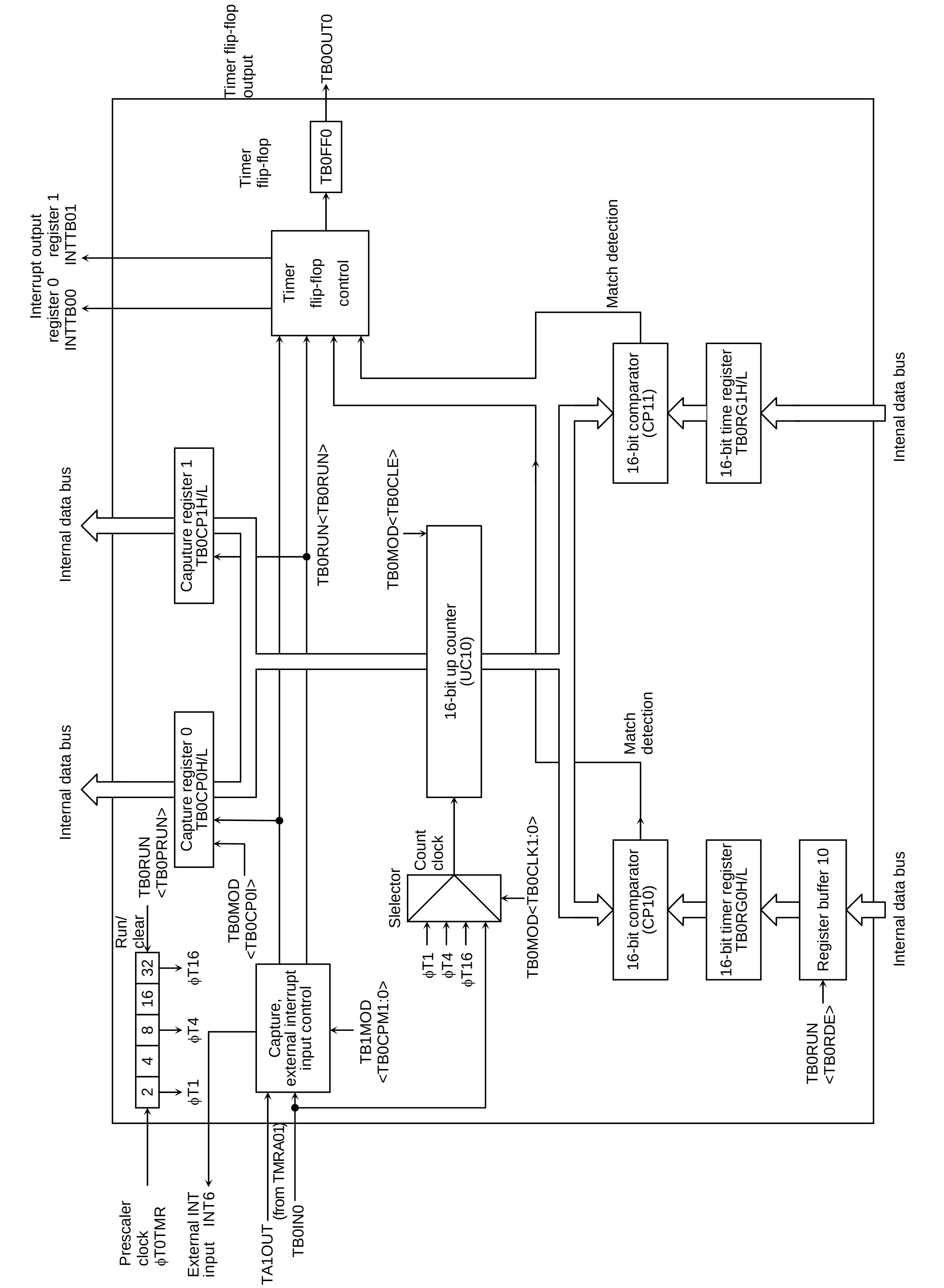


Figure 3.14.1 Block diagram of TMRB0

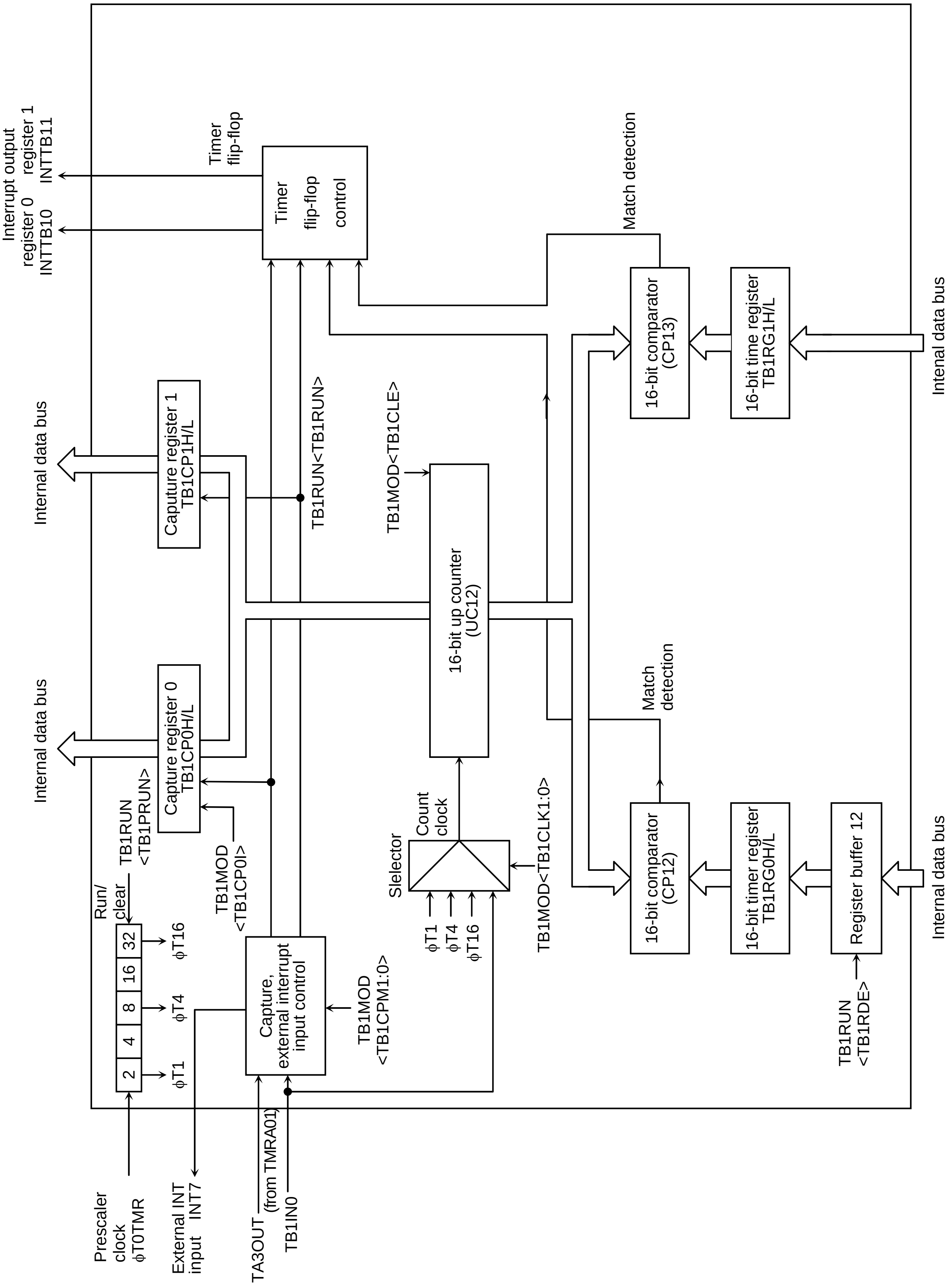


Figure 3.14.2 Block diagram of TMRB1

3.14.2 Operation

(1) Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (ϕT0TMR) is selected by the register SYSCR0<PRCK> of clock gear. This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0RUN> is set to “1”; the prescaler is cleared to “0” and stops operation when <TB0RUN> is cleared to “0”.

The resolution of prescaler is showed in the Table 3.14.2.

Table 3.14.2 Prescaler Clock Resolution

	Clock gear selection SYSCR1 <GEAR2:0>	Prescaler of clock gear SYSCR0 <PRCK>	–	Timer counter input clock Prescaler of TMRB TBxMOD<TBxCLK1:0>		
				ϕT1(1/2)	ϕT4(1/8)	ϕT16(1/32)
fc	000(1/1)	0(1/2)	1/2	fc/8	fc/32	fc/128
	001(1/2)			fc/16	fc/64	fc/256
	010(1/4)			fc/32	fc/128	fc/512
	011(1/8)			fc/64	fc/256	fc/1024
	100(1/16)			fc/128	fc/512	fc/2048
	000(1/1)	1(1/8)		fc/32	fc/128	fc/512
	001(1/2)			fc/64	fc/256	fc/1024
	010(1/4)			fc/128	fc/512	fc/2048
	011(1/8)			fc/256	fc/1024	fc/4096
	100(1/16)			fc/512	fc/2048	fc/8192

(2) Up counter (UC10)

UC10 is a 16-bit binary counter which counts up pulses input from the clock specified by TB0MOD<TB0CLK1:0>.

Any one of the prescaler internal clocks ϕT1, ϕTB0 and ϕT16 or an external clock input via the TB0IN0 pin can be selected as the input clock. Counting or stopping and clearing of the counter is controlled by TB0RUN<TB0RUN>.

When clearing is enabled, the up counter UC10 will be cleared to “0” each time its value matches the value in the timer register TB0RG1H/L. If clearing is disabled, the counter operates as a free running counter.

Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

(3) Timer registers (TB0RG0H/L, TB0RG1H/L)

These two 16-bit registers are used to set the interval time. When the value in the up counter UC10 matches the value set in this timer register, the comparator match detect signal will go active.

Setting data for both upper and lower timer registers is always needed. For example, eithre using a 2-byte data transfer instruction or using a 1-byte data transfer instruction twice for the lower 8 bits and upper 8 bits in order.

(The compare circuit will not operate if only the lower 8 bits are written. Be sure to write to both timer registers (16 bits) from the lower 8 bits followed by the upper 8 bits.)

The TB0RG0H/L timer register has a double-buffer structure, which is paired with a register buffer 10. The value set in TB0RUN<TB0RDE> determines whether the double-buffer structure is enabled or disabled: it is disabled when <TB0RDE> = "0", and enabled when <TB0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer 10 to the timer register when the values in the up counter (UC10) and the timer register TB0RG1H/L match.

The double buffer circuit incorporates two flags to indicate whether or not data is written to the lower 8 bits and the upper 8 bits of the register buffer, respectively. Only when both flags are set can data be transferred from the register buffer to the timer register by a match between the up-counter UC10 and the timer register TB0RG1H/L. This data transfer is performed so long as 16-bit data is written in the register buffer regardless of the register buffer to the timer register unexpectedly as explained below.

For example, let us assume that an interrupt occurs when only the lower 8 bits (L1) of the register buffer data (H1L1) have been written and the interrupt routine includes writes to all 16 bits in the register buffer and a transfer of the data to the timer register. In this case, if the higher 8 bits (H1) are written after the interrupt routine is completed, only the flag for the higher 8 bits will be set, the flag for the lower 8 bits having been cleared in the interrupt routine. Therefore, even if a match occurs between UC10 and TB0RG1H/L, no data transfer will be performed.

Then, in an attempt to set the next set of data (H2L2) in the register buffer, when the lower 8 bits (L2) are written, this will cause the flag for the lower 8 bits to be set as well as the flag for the higher 8 bits which has been set by writing the previous data (H1). If a match between UC10 and TB0RG1H/L occurs before the higher 8 bits (H2) are written, this will cause unexpected data (H1L2) to be sent to the timer register instead of the intended data (H2L2).

To avoid such transfer timing problems due to interrupts, the DI instruction (disable interrupts) and the EI (enable interrupts) can be executed before and after setting data in the register buffer, respectively.

After a reset, TB0RG0H/L and TB0RG1H/L are undefined. If the 16-bit timer is to be used after a reset, data should be written to it beforehand.

On a reset <TB0RDE> is initialized to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TB0RDE> to "1", then write data to the register buffer 10 as shown below.

TB0RG0H/L and the register buffer 10 both have the same memory addresses (1188H and 1189H) allocated to them. If <TB0RDE> = “0”, the value is written to both the timer register and the register buffer 10. If <TB0RDE> = “1”, the value is written to the register buffer 10 only.

The addresses of the timer registers are as follows:

TMRB0

TB0RG0H/L		TB0RG1 H/L	
Upper 8 bits (TB0RG0H)	Lower 8 bits (TB0RG0L)	Upper 8 bits (TB0RG1H)	Lower 8 bits (TB0RG1L)
1189H	1188H	118BH	118AH

TMRB1

TB1RG0 H/L		TB1RG1 H/L	
Upper 8 bits (TB1RG0H)	Lower 8 bits (TB1RG0L)	Upper 8 bits (TB1RG1H)	Lower 8 bits (TB1RG1L)
1199H	1198H	119BH	119AH

The timer registers are write-only registers and thus cannot be read.

(4) Capture registers (TB0CP0H/L, TB0CP1H/L)

These 16-bit registers are used to latch the values in the up counter (UC10).

All 16 bits of data in the capture registers should be read. For example, using a 2-byte data load instruction or two 1-byte data load instructions. The least significant byte is read first, followed by the most significant byte.

(during capture is read, capture operation is prohibited. In that case, the lower 8 bits should be read first, followed by the 8 bits.)

The addresses of the capture registers are as follows;

TMRB0

TB0CP0H/L		TB0CP1H/L	
Upper 8 bits (TB0CP0H)	Lower 8 bits (TB0CP0L)	Upper 8 bits (TB0CP1H)	Lower 8 bits (TB0CP1L)
118DH	118CH	118FH	118EH

TMRB1

TB1CP0H/L		TB1CP1H/L	
Upper 8 bits (TB1CP0H)	Lower 8 bits (TB1CP0L)	Upper 8 bits (TB1CP1H)	Lower 8 bits (TB1CP1L)
119DH	119CH	119FH	119EH

The capture registers are read-only registers and thus cannot be written to.

(5) Capture input and external interrupt control

This circuit controls the timing to latch the value of the up-counter UC10 into TB0CP0H/L and TB0CP1H/L, and generates external interrupt. The latch timing of capture register and selection of edge for external interrupt is controlled by TB0MOD<TB0CPM1:0>.

The value in the up-counter (UC10) can be loaded into a capture register by software. Whenever "0" is written to TB0MOD<TB0CP0I>, the current value in the up counter (UC10) is loaded into capture register TB0CP0H/L. It is necessary to keep the prescaler in RUN mode (i.e., TB0RUN<TB0PRUN> must be held at a value of "1").

(6) Comparators (CP10, CP11)

CP10 and CP11 are 16-bit comparators which compare the value in the up counter UC10 with the value set in TB0RG0H/L or TB0RG1H/L respectively, in order to detect a match. If a match is detected, the comparator generates an interrupt (INTTB00 or INTTB01 respectively).

(7) Timer flip-flops (TB0FF0)

These flip-flops are inverted by the match detect signals from the comparators and the latch signals to the capture registers. Inversion can be enabled and disabled for each element using TB0FFCR<TB0C1T1, TB0C0T1, TB0E1T1, TB0E0T1>.

After a reset the value of TB0FF0 is undefined. If “00” is written to TB0FFCR <TB0FF0C1:0>, TB0FF0 will be inverted. If “01” is written to the capture registers, the value of TB0FF0 will be set to “1”. If “10” is written to the capture registers, the value of TB0FF0 will be set to “0”.

Note: If an inversion by the match-detect signal and a setting change via the TB0FFCR register occurs simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect siganl and an attempt to set the flip-flop to “1” via the register occur simultaneously, the flip-flop will be set to “1”.
- If an inversion by the match-detect signal and an attmept to cleare the flip-flop to “0” via the register occur simultanerously, the flip-flop will be cleared to “0”.

If an inversion by match-detect signal and inversion disable setting occur simultaneously, two case (it is inverted and it is not inverted) are occurred. Therefore, if changing inversion control (inversion enable/disable), stop timer operation beforehand.

The values of TB0FF0 can be output via the timer output pins TB0OUT0 (which is shared with PP6) and TB0OUT1 (which is shared with PP7). Timer output should be specified using the port P function register.

3.14.3 SFR

TMRB0 RUN Register

TB0RUN (1180H)		7	6	5	4	3	2	1	0
	Bit symbol	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
	Read/Write	R/W	R/W			R/W	R/W		R/W
	Reset State	0	0			0	0		0
	Function	Double buffer 0: disable 1: enable	Always write "0"			In IDLE2 mode 0: Stop 1: Operate	TMRB0 prescaler 0: Stop and clear 1: Run (Count up)		Up counter (UC10) 0: Stop and clear 1: Run (Count up)

Count operation

<TB0PRUN>, <TB0RUN>	0	Stop and clear
	1	Count up

Note: 1, 4 and 5 of TB0RUN are read as "1" values.

TMRB1 RUN Register

TB1RUN (1190H)		7	6	5	4	3	2	1	0
	Bit symbol	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
	Read/Write	R/W	R/W			R/W	R/W		R/W
	Reset State	0	0			0	0		0
	Function	Double buffer 0: disable 1: enable	Always write "0"			In IDLE2 mode 0: Stop 1: Operate	TMRB1 prescaler 0: Stop and clear 1: Run (Count up)		Up counter (UC12) 0: Stop and clear 1: Run (Count up)

Count operation

<TB1PRUN>, <TB1RUN>	0	Stop and clear
	1	Count up

Note: 1, 4 and 5 of TB1RUN are read as "1" values.

Figure 3.14.3 Register for TMRB

TMRB0 Mode Register

TB0MOD (1182H)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
Read/Write	R/W		W*	R/W				
Reset State	0	0	1	0	0	0	0	0
Function	Always write “0”.		Software capture control 0: Software capture 1:Undefined	Capture timing 00:Disable INT6 occurs at rising edge 01:TB0IN0 ↑ INT6 occurs at rising edge 10: TB0IN0 ↑ TB0IN0 ↓ INT6 occurs at falling edge 11: TA1OUT ↑ TA1OUT ↓ INT6 occurs at rising edge		Control Up counter 0:Disable 1:Enable	TMRB0 source clock 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	

TMRB0 source clock

<TB0CLK1:0>	00	TB0IN0 pin input
	01	ϕT1
	10	ϕT4
	11	ϕT16

Control clearing for up counter (UC10)

<TB0CLE>	0	Disable
	1	Enable clearing by match with TB0RG1H/L

Capture/interrupt timing

		Capture control	INT6 control
<TB0CPM1:0>	00	Disable	INT6 occurs at the rising edge of TB0IN0 ↑
	01	Capture to TB0CP0H/L at rising edge of TB0IN0	
	10	Capture to TB0CP0H/L at rising edge of TB0IN0 Capture to TB0CP1H/L at falling edge of TB0IN0	INT6 occurs at the rising edge of TB0IN0 ↓
	11	Capture to TB0CP0H/L at rising edge of TA1OUT Capture to TB0CP1H/L at falling edge of TA1OUT	INT6 occurs at the rising edge of TB0IN0 ↑

Software capture

<TB0CP0I>	0	The value of up counter is captured to TB0CP0H/L
	1	Undefined

Figure 3.14.4 Register for TMRB

TMRB1 Mode Register

TB1MOD (1192H)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	–	–	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
Read/Write	R/W		W*	R/W				
Reset State	0	0	1	0	0	0	0	0
Function	Always write “0”.		Software capture control 0: Software capture 1:Undefined	Capture timing 00:Disable INT7 occurs at rising edge 01:TB1IN0 ↑ INT7 occurs at rising edge 10: TB1IN0 ↑ TB1IN0 ↓ INT7 occurs at falling edge 11: TA3OUT ↑ TA3OUT ↓ INT7 occurs at rising edge		Control Up counter 0:Disable 1:Enable	TMRB1 source clock 00: TB1IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	

TMRB1 source clock

<TB1CLK1:0>	00	TB1IN0 pin input
	01	ϕT1
	10	ϕT4
	11	ϕT16

Control clearing for up counter (UC12)

<TB1CLE>	0	Disable
	1	Enable clearing by match with TB1RG1H/L

Capture/interrupt timing

		Capture control	INT7 control
<TB1CPM1:0>	00	Disable	INT7 occurs at the rising edge of TB1IN0
	01	Capture to TB1CP0H/L at rising edge of TB1IN0	
	10	Capture to TB1CP0H/L at rising edge of TB1IN0 Capture to TB1CP1H/L at falling edge of TB1IN0	INT7 occurs at the rising edge of TB1IN0
	11	Capture to TB1CP0H/L at rising edge of TA3OUT Capture to TB1CP1H/L at falling edge of TA3OUT	INT7 occurs at the rising edge of TB1IN0

Software capture

<TB1CP0I>	0	The value of up counter is captured to TB1CP0H/L
	1	Undefined

Figure 3.14.5 Register for TMRB

TMRB0 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TB0FFCR (1183H)	Bit symbol	–	–	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
A read -modify-write operation cannot be performed	Read/Write	W*		R/W				W*	
	Reset State	1	1	0	0	0	0	1	1
	Function	Always write “11” *Always read as “11”.		TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB0FF0 00: Invert 01: Set 10: Clear 11: Undefined *Always read as “11”.	
				When capture UC10 to TB0CP1H/L	When capture UC10 to TB0CP0H/L	When UC10 matches with TB0RG1H/L	When UC10 matches with TB0RG0H/L		

Timer flip-flop control(TB0FF0)

<TB0FF0C1:0>	00	Invert
	01	Set to “11”
	10	Clear to “00”
	11	Undefined (Always read as “11”)

TB0FF0 control
Inverted when UC10 value matches the valued in TB0RG0H/L

<TB0E0T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value matches the valued in TB0RG1H/L

<TB0E1T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value is captured into TB0CP0H/L

<TB0C0T1>	0	Disable trigger
	1	Enable trigger

TB0FF0 control
Inverted when UC10 value is captured into TB0CP1H/L

<TB0C1T1>	0	Disable trigger
	1	Enable trigger

Figure 3.14.6 Register for TMRB

TMRB0 register

Register		7	6	5	4	3	2	1	0
TB0RG0L (1188H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB0RG0H (1189H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB0RG1L (118AH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB0RG1H (118BH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB1RG0L (1198H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB1RG0H (1199H)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB1RG1L (119AH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							
TB1RG1H (119BH)	bit Symbol	–							
	Read/Write	W							
	Reset State	0							

Note: A read-modify-write operation cannot be performed for All registers.

Figure 3.14.7 Register for TMRB

3.14.4 Operation in Each Mode

(1) 16 bit timer mode

Generating interrupts at fixed intervals

In this example, the interrupt INTTB01 is set to be generated at fixed intervals. The interval time is set in the timer register TB0RG1H/L.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	–	X	0	Stop TMRB0
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01and set interrupt level 4. Disable INTTB00
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable the trigger
TB0MOD	←	0	0	1	0	0	1	*	*	Select internal clock for input and
						(** = 01, 10, 11)				disable the capture function.
TB0RG1H/L	←	*	*	*	*	*	*	*	*	Set the interval time
		*	*	*	*	*	*	*	*	(16 bits).
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

(2) 16 bit event counter mode

In 16 bit timer mode as described in above, the timer can be used as an event counter by selecting the external clock (TB0IN0 pin input) as the input clock. Up counter (UC10) counts up at the rising edge of TB0IN0 input. To read the value of the counter, first perform "software capture" once and read the captured value.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	–	X	0	Stop TMRB0
PPCR	←	X	X	–	1	–	–	–	X	Set PP4 to input mode for TB0IN0
PPFC	←	–	–	–	1	–	–	–	X	
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and sets interrupt level 4 Disable INTTB00
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable trigger
TB0MOD	←	0	0	1	0	0	1	0	0	Select TB0IN0 as the input clock
TB0RG1H/L	←	*	*	*	*	*	*	*	*	Set the number of counts
		*	*	*	*	*	*	*	*	(16 bit)
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0

X: Don't care, –: No change

When used as an event counter, set the prescaler in RUN mode.
(TB0RUN <TB0PRUN> = "1")

(3) 16-bit programmable pulse generation (PPG) output mode

Square wave pulses can be generated at any frequency and duty ratio. The output pulse may be either low active or high active.

The PPG mode is obtained by inversion of the timer flip-flop TB0FF0 that is enabled by the match of the up counter UC10 with timer register TB0RG0H/L or TB0RG1H/L and is output to TB0OUT0. In this mode the following conditions must be satisfied.

(Value set in TB0RG0H/L) < (Value set in TB0RG1H/L)

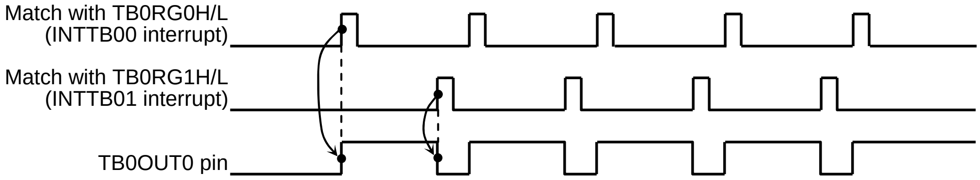


Figure 3.14.8 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0H/L double buffer is enabled in this mode, the value of register buffer 10 will be shifted into TB0RG0H/L at match with TB0RG1H/L. This feature facilitates the handling of low-duty waves.

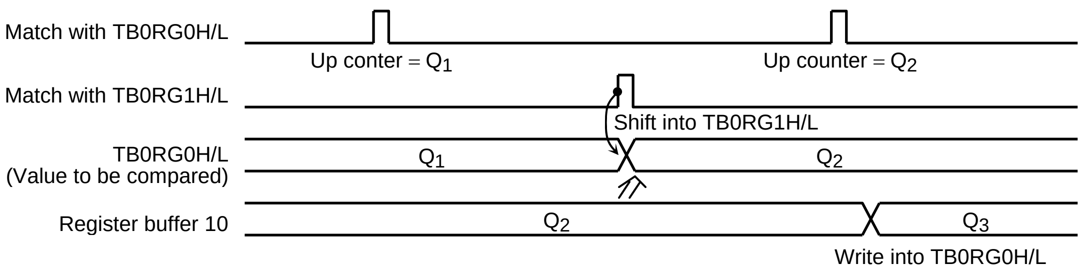


Figure 3.14.9 Operation of double buffer

Note: The values that can be set in TBxRGxH/L range from 0001h to 0000h (equivalent to 10000h). If the maximum value 000h is set, the match-detect signal goes active when the up-counter overflows.

The following block diagram illustrates this mode.

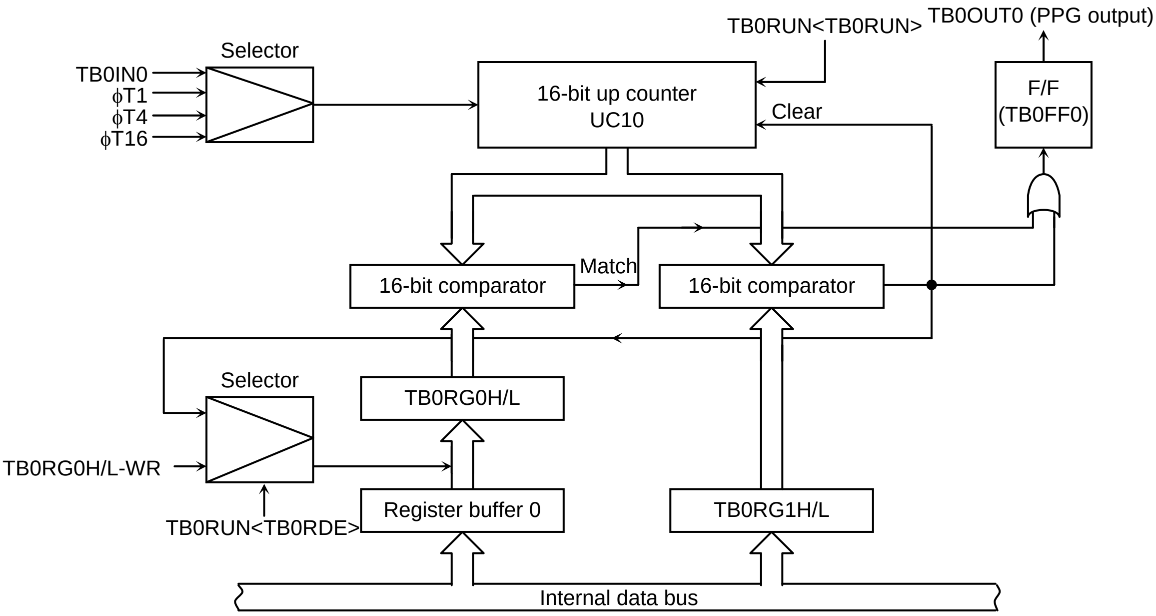


Figure 3.14.10 Block Diagram of 16-Bit Mode

The following example shows how to set 16-bit PPG output mode:

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	–	X	0	Disable the TB0RG0H/L double buffer and stop TMRB0.
TB0RG0H/L	←	*	*	*	*	*	*	*	*	Set the duty ratio
		*	*	*	*	*	*	*	*	(16 bit)
TB0RG1H/L	←	*	*	*	*	*	*	*	*	Set the frequency
		*	*	*	*	*	*	*	*	(16 bit)
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0H/L double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0H/L/TB0RG1H/L. Set TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Select the internal clock as the input clock and disable the capture function.
						(** = 01, 10, 11)				
PPFC	←	–	1	–	–	–	–	–	X	Set PP6 to function as TB0OUT0
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

(4) Application examples of capture function

Used capture function, they can be applied in many ways, for example;

1. One-shot pulse output from external trigger pulse
2. Frequency measurement
3. Pulse width measurement

1. One-shot pulse output from external trigger pulse

Set the up counter UC10 in free-running mode with the internal input clock, input the external trigger pulse from TB0IN0 pin, and load the value of up counter into capture register TB0CP0H/L at the rising edge of the TB0IN0 pin.

When the interrupt INT6 is generated at the rising edge of TB0IN0 input, set the TB0CP0H/L value (c) plus a delay time (d) to TB0RG0H/L (=c+d), and set the above set value (c+d) plus a one-shot pulse width (p) to TB0RG1H/L (=c+d+p).

The TB0FFCR<TB0E1T1, TB0E0T1> register should be set “11” and that the TB0FF0 inversion is enabled only when the up counter value matches TB0RG0H/L or TB0RG1H/L. When interrupt INTTB01 occurs, this inversion will be disabled after one-shot pulse is output.

The (c), (d) and (p) correspond to c, d, and p in the Figure 3.14.11.

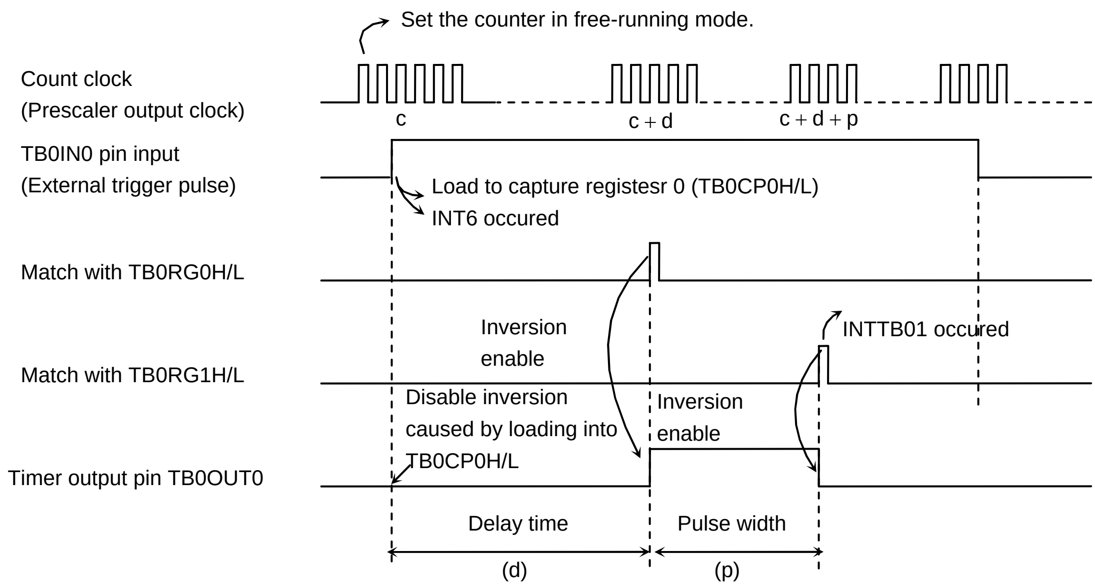


Figure 3.14.11 One-shot Pulse Output (with delay)

Example: To output 2ms one-shot pulse with 3ms delay to the external trigger pulse to TB0IN0 pin

*Clock state
- System clock : f_{SYS}
- Prescaler clock : $f_{SYS}/4$

Main setting

Register		7	6	5	4	3	2	1	0	
TB0MOD	←	X	X	1	0	1	0	0	1	Free-running; Count with ϕT1; Load to TB0CP0H/L at the rising edge of TB0IN0
TB0FFCR	←	X	X	0	0	0	0	1	0	Clear TB0FF0 to "0"; Disable TB0FF0 inversion
PPFC	←	–	1	–	–	–	–	–	X	Select PP6 as TB0OUT0 pin (port setting)
INTE56	←	X	1	0	0	X	–	–	–	Enable INT6
INTETB0	←	X	0	0	0	X	0	0	0	Disable INTTB00, INTTB01
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0

Setting in INT6 routine

TB0RG0H/L ← TB0CP0H/L + 3ms/ϕT1

TB0RG1H/L ← TB0RG0H/L + 2ms/ϕT1

Register		7	6	5	4	3	2	1	0	
TB0FFCR	←	X	X	–	–	1	1	–	–	Enable TB0FF0 inversion when the up counter value matches TB0RG0H/L or TB0RG1H/L
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01

Setting in INTTB01 routine

Register		7	6	5	4	3	2	1	0	
TB0FFCR	←	X	X	–	–	0	0	–	–	Disable TB0FF0 inversion when the up counter value matches TB0RG0H/L or TB0RG1H/L
INTETB0	←	X	0	0	0	X	0	0	0	Disable INTTB01

X: Don't care, –: No change

When delay time is unnecessary, invert timer flip-flop TB0FF0 when the up counter value is loaded into capture register (TB0CP0H/L), and set the TB0CP0H/L value (c) plus the one –shot pulse width (p) to TB0RG1H/L when the interrupt INT6 occurs. The TB0FF0 inversion should be enabled when the up counter (UC10) value matched TB0RG1H/L, and disabled when generating the interrupt INTTB01.

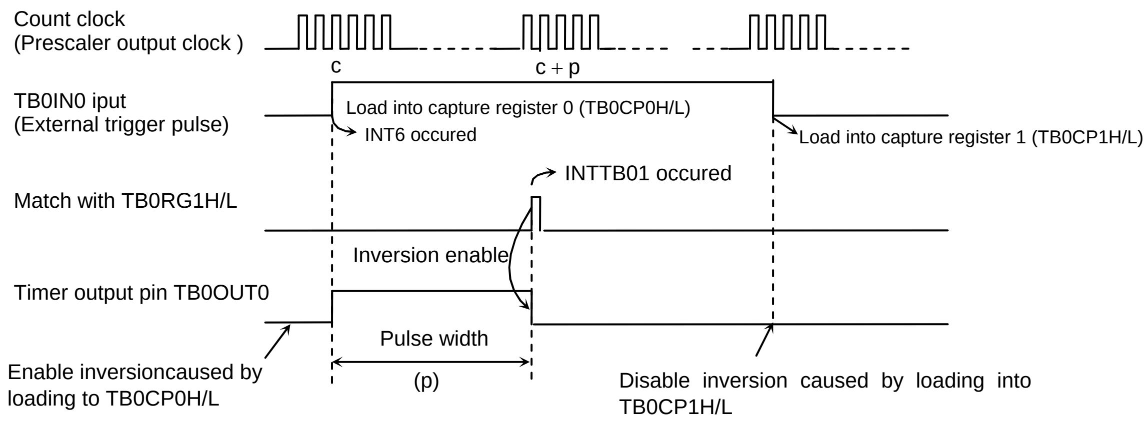


Figure 3.14.12 One-shot Pulse Output (without delay)

2. Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TB0IN0 pin, and its frequency is measured by the 8 bit timers TMRA01 and the 16 bit timer/event counter (TMRB0).

The TB0IN0 pin input should be selected for the input clock of TMRB0. Set to TB0MOD<TB0CPM1:0>=”11”. The value of the up counter is loaded into the capture register TB0CP0H/L at the rising edge of the timer flip-flop TA1FF of 8bit timers (TMRA01), and TB0CP1H/L at its falling edge.

The frequency is calculated by the difference between the loaded values in TB0CP0H/L and TB0CP1H/L when the interrupt (INTTA0 or INTTA1) is generated by either 8 bit timer.

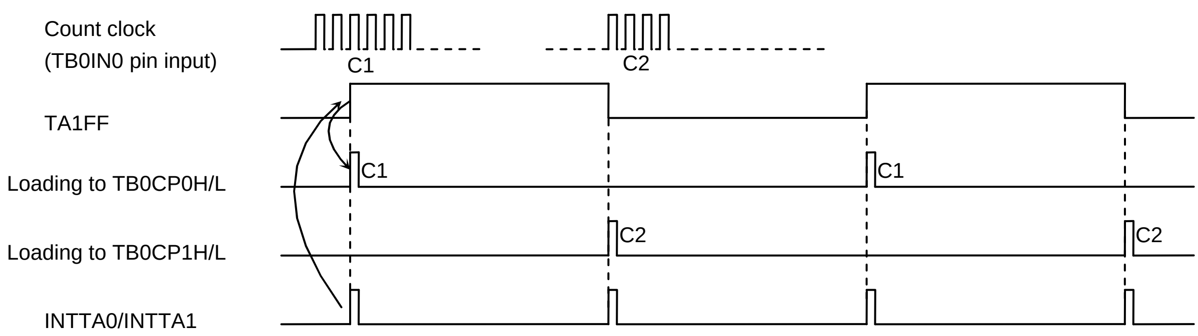


Figure 3.14.13 Frequency Measurement

For example, if the value for the level 1 width of TA1FF of the 8 bit timer is set to 0.5[s] and the difference between TB0CP0H/L and TB0CP1H/L is 100, the frequency will be 100/0.5[s] =200[Hz].

Note: The frequency in this examole is calculated with 50% duty.

3. Pulse width measurement

This mode allows measuring the H level width of an external pulse. While keeping the 16 bit timer/event counter counting (free-running) with the internal clock input, the external pulse is input through the TB0IN0 pin. Then the capture function is used to load the UC10 values into TB0CP0H/L and TB0CP1H/L at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT6 occurs at the falling edge of TB0IN0.

The pulse width is obtained from the difference between the values of TB0CP0H/L and TB0CP1H/L and the internal clock cycle.

For example, if the internal clock is 0.8[us] and the difference between TB0CP0H/L and TB0CP1H/L is 100, the pulse width will be 100 × 0.8[μs] =80μs

Additionally, the pulse width which is over the UC10 maximum count time specified by the clock source can be measured by changing software.

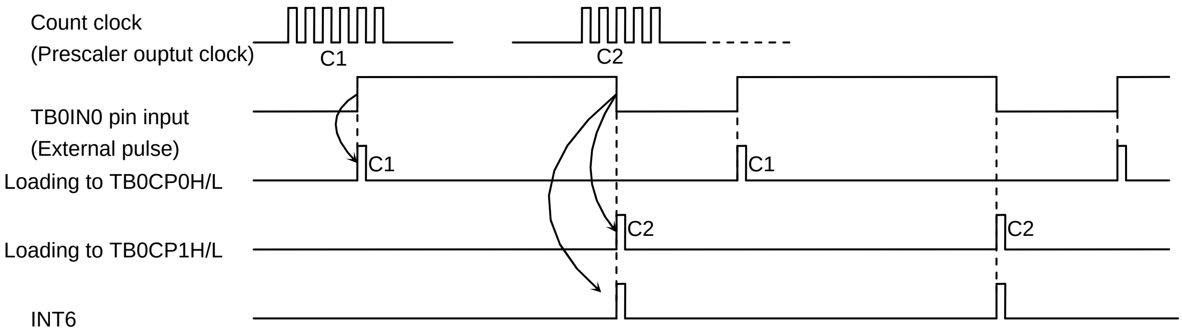


Figure 3.14.14 Pulse Width Measurement

Note: Only in this pulse width measuring mode(TB0MOD<TB0CPM1:0>= “10”), external interrupt INT6 occurs at the falling edge of TB0IN0 pin input. In other modes, it occurs at the rising edge.

The width of L level can be measured by multiplying the difference between the first C1 and the second C0 at the second INT6 interrupt and the internal clock cycle together.

3.15 Serial Channels (SIO)

The TMP92CF29A include 2 serials I/O channel (SIO0 and SIO1). For channels either UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission) can be selected. And, SIO0 and SIO1 include data modulator that supports the IrDA 1.0 infrared data communication specification.

- I/O interface mode —— Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending I/O.
- UART mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

In mode 1 and mode 2, a parity bit can be added. Mode 3 has a wakeup function for making the master controller start slave controllers via a serial link (A multi-controller system).

Figure 3.15.1 is block diagrams for each channel.

Each channel is compounded mainly prescaler, serial clock generation circuit, receiving buffer and control circuit, transmission buffer and control circuit.

Each channel can be used independently.

Each channel operates in the same fashion except for the following points; hence only the operation of channel 0 is explained below.

Table 3.15.1 Differences between Channels 0 to 1

	Channel 0	Channel 1
Pin name	TXD0 (P90 or PP3) RXD0 (P91 or PP4) $\overline{CTS0}$, SCLK0 (P92 or PP5)	TXD1 (P90 or PP3) RXD1 (P91 or PP4) $\overline{CTS1}$, SCLK1 (P92 or PP5)
IrDA mode	Yes	Yes

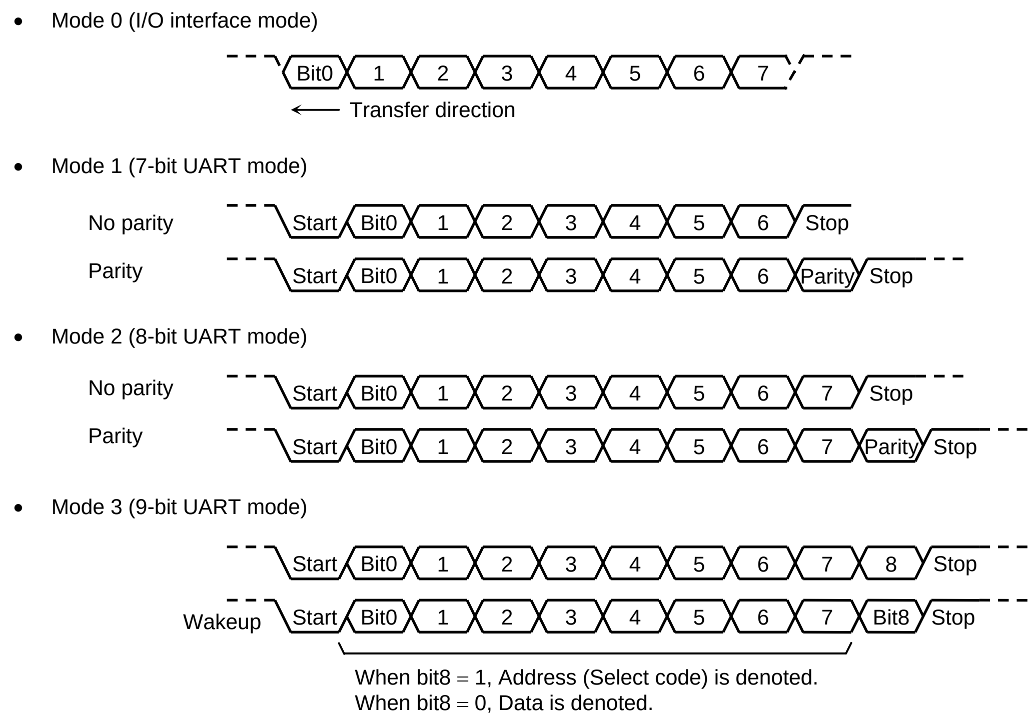


Figure 3.15.1 Data Formats

3.15.1 Block Diagram

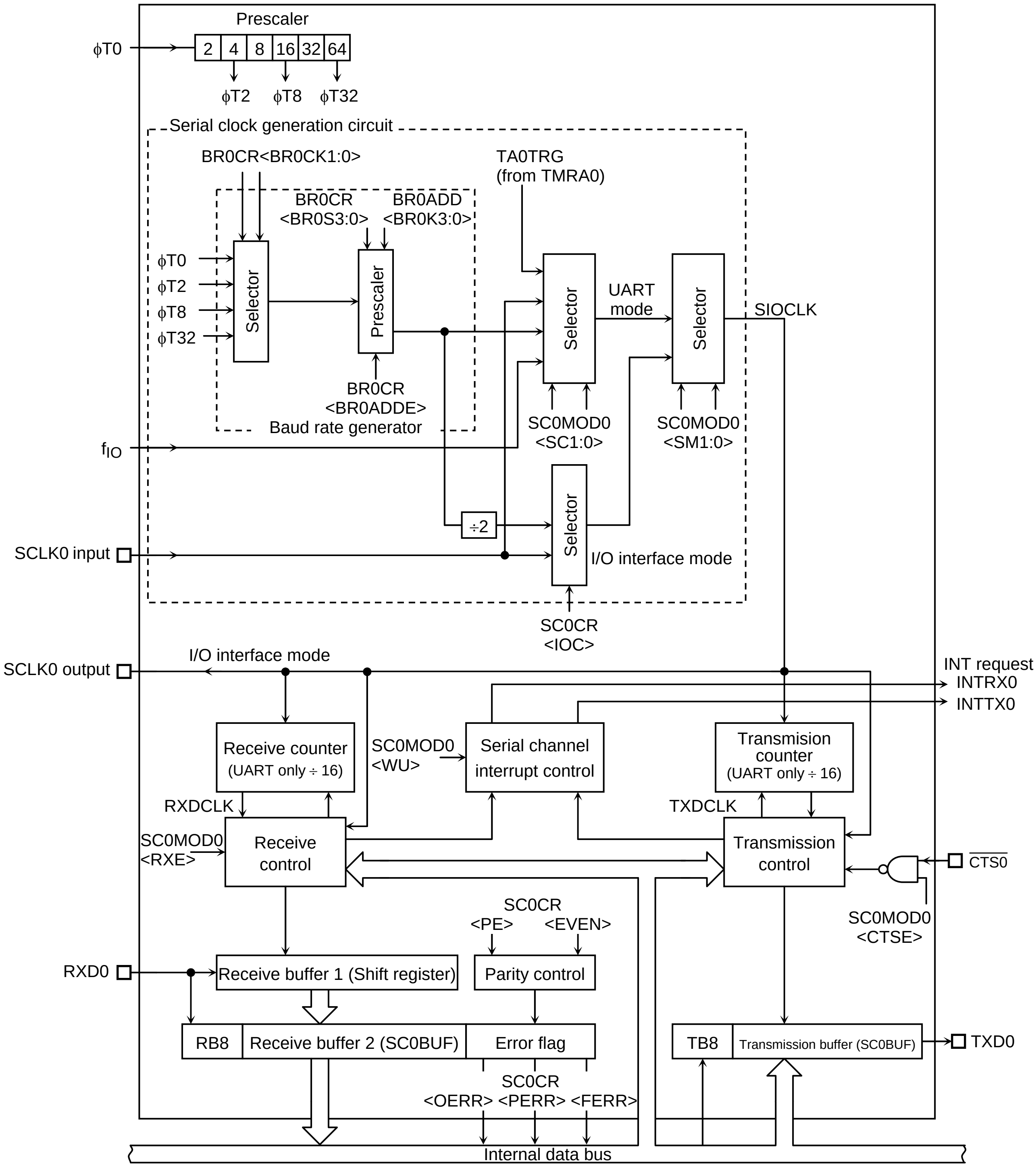


Figure 3.15.2 Block Diagram

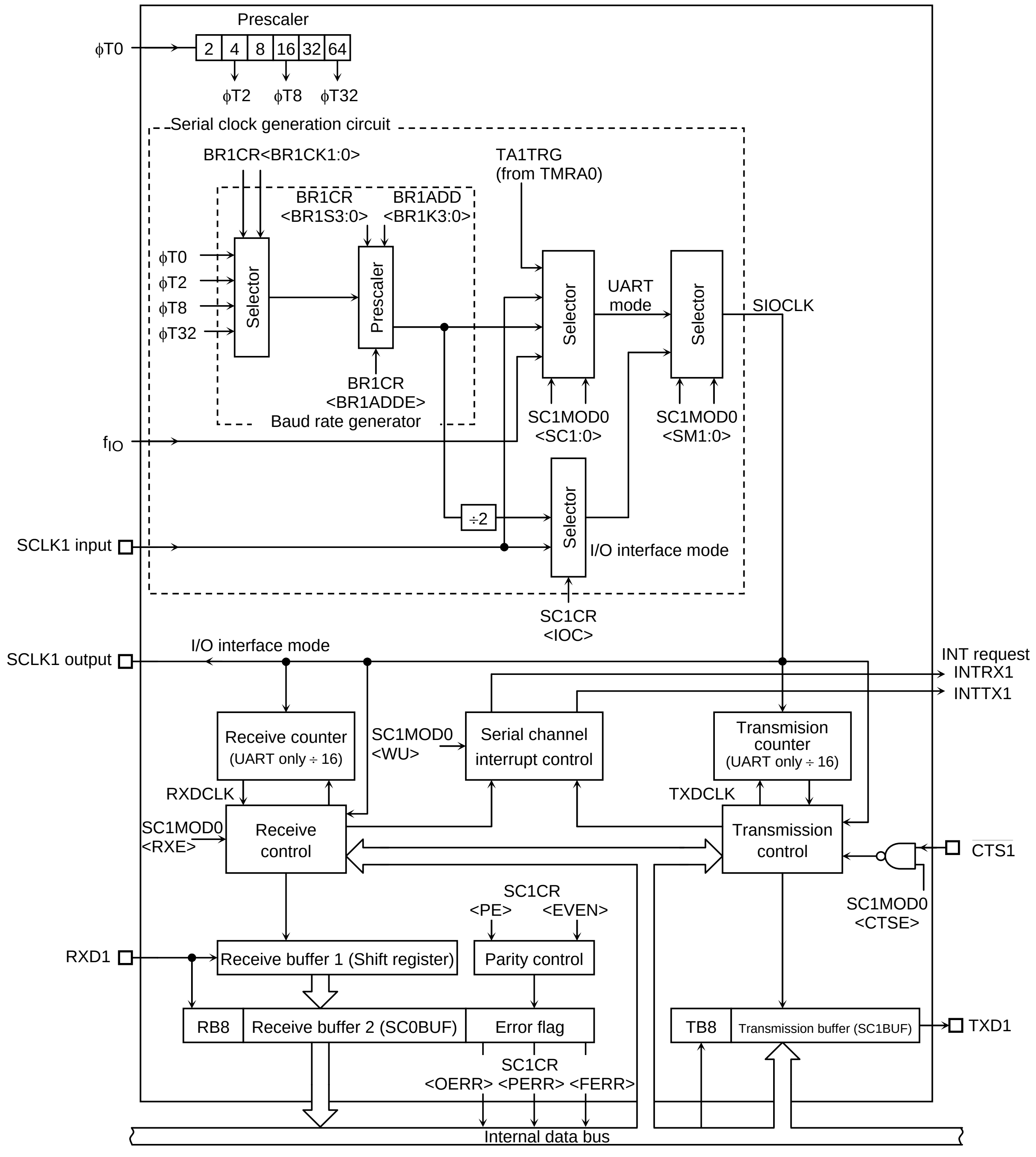


Figure 3.15.3 SIO1 Block Diagram

3.15.1.1 Block Diagram

Figure 3.15.4 shows the connection image for SIO circuits in TMP92CF29A.

SIO circuit are built-in 2ch, it can set each signal to P90, P91, P92 or PP3, PP4, PP5.

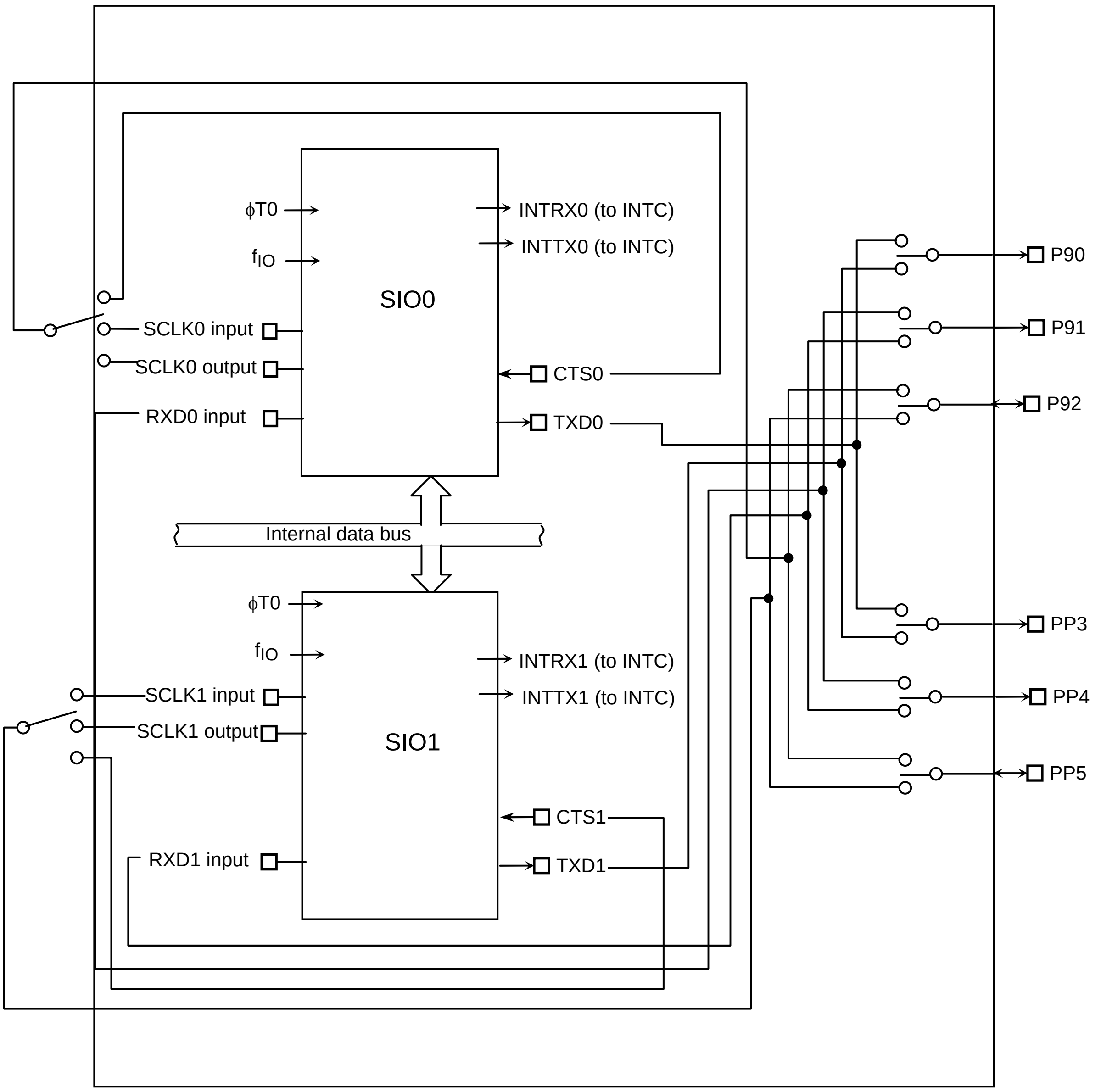


Figure 3.15.4 Connection images of internal circuit and external port

Note1: Figure 3.15.4 shows connection image. The circuit compounds and a setting procedure Refer to section of Port.

Note2: When shifting extrernal port, shift port after stop internal circuits completely.

3.15.2 Operation of Each Circuit

(1) Prescaler

There is a 6-bit prescaler for generating a clock to SIO0. The prescaler can be run by selecting the baud rate generator as the serial transfer clock.

Table 3.15.2 shows prescaler clock resolution into the baud rate generator.

Table 3.15.2 Prescaler Clock Resolution to Baud Rate Generator

–	Clock gear SYSCR1 <GEAR2:0>	–	Baud Rate Generator input clock SIO Prescaler BR0CR<BR0CK1:0>			
			φT0 (1/1)	φT2(1/4)	φT8(1/16)	φT32(1/64)
fc	000(1/1)	1/4	fc/4	fc/16	fc/64	fc/256
	001(1/2)		fc/8	fc/32	fc/128	fc/512
	010(1/4)		fc/16	fc/64	fc/256	fc/1024
	011(1/8)		fc/32	fc/128	fc/512	fc/2048
	100(1/16)		fc/64	fc/256	fc/1024	fc/4096

The baud rate generator selects between 4-clock inputs: φT0, φT2, φT8, and φT32 among the prescaler outputs.

(2) Baud rate generator

The baud rate generator is the circuit which generates transmission/receiving clock and determines the transfer rate of the serial channels.

The input clock to the baud rate generator, φT0, φT2, φT8 or φT32, is generated by the 6-bit prescaler which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1:0> field in the baud rate generator control register.

The baud rate generator includes a frequency divider, which divides the frequency by 1 or N + (16 − K)/16 to 16 values, thereby determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3:0> and BR0ADD<BR0K3:0>.

In UART mode

When BR0CR<BR0ADDE> = "0"

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3:0>. (N = 1, 2, 3 ... 16)

When BR0CR<BR0ADDE> = "1"

The N + (16 − K)/16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 − K)/16 using the value of N set in BR0CR<BR0S3:0> (N = 2, 3 ... 15) and the value of K set in BR0ADD<BR0K3:0> (K = 1, 2, 3 ... 15)

Note: If N = 1 or N = 16, the N + (16 − K)/16 division function is disabled. Clear BR0CR<BR0ADDE> to "0".

In I/O interface mode

The N + (16 − K)/16 division function is not available in I/O interface mode. Clear BR0CR<BR0ADDE> to "0" before dividing by N.

The method for calculating the transfer rate when the baud rate generator is used is explained below.

- In UART mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

- In I/O interface mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 2$$

Integer divider (N divider)

For example, when the source clock frequency (f_C) is 19.6608 MHz, the input clock is ϕT2, the frequency divider N (BR0CR<BR0S3:0>) = 8, and BR0CR<BR0ADDE> = "0", the baud rate in UART Mode is as follows:

*Clock state [Clock gear : 1/1

$$\text{Baud Rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

$$= \frac{f_C/16}{8} \div 16$$

$$= 19.6608 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 – K) / 16 division function is disabled and setting BR0ADD <BR0K3:0> is invalid.

N+(16-K)/16 divider (UART Mode only)

Accordingly, when the source clock frequency (fc) = 15.9744 MHz, the input clock is ϕT2, the frequency divider N (BR0CR<BR0S3:0>) = 6, K (BR0ADD<BR0K3:0>) = 8, and BR0CR <BR0ADDE> = "1", the baud rate in UART Mode is as follows:

* Clock state [Clock gear : 1/1

$$\text{Baud Rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

$$= \frac{f_C/16}{6 + \frac{(16-8)}{16}} \div 16$$

$$= 15.9744 \times 10^6 \div 16 \div (6 + \frac{8}{16}) \div 16 = 9600 \text{ (bps)}$$

Table 3.15.3 show examples of UART Mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial Channel 0). The method for calculating the baud rate is explained below:

- In UART Mode

  Baud rate = external clock input frequency ÷ 16

  It is necessary to satisfy (external clock input cycle) ≥ $4/f_{SYS}$
- In I/O Interface Mode

  Baud rate = external clock input frequency

  It is necessary to satisfy (external clock input cycle) ≥ $16/f_{SYS}$

Table 3.15.3 Transfer Rate Selection
(When baud rate generator is used and BR0CR<BR0ADDE> = "0") Unit (kbps)

f_{SYS} [MHz]	Input Clock / Frequency Divider N	ϕT0 (f_{SYS}/4)	ϕT2 (f_{SYS}/16)	ϕT8 (f_{SYS}/64)	ϕT32 (f_{SYS}/256)
7.3728	1	115.200	28.800	7.200	1.800
↑	3	38.400	9.600	2.400	0.600
↑	6	19.200	4.800	1.200	0.300
↑	A	11.520	2.880	0.720	0.180
↑	C	9.600	2.400	0.600	0.150
↑	F	7.680	1.920	0.480	0.120
9.8304	1	153.600	38.400	9.600	2.400
↑	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	5	30.720	7.680	1.920	0.480
↑	8	19.200	4.800	1.200	0.300
↑	0	9.600	2.400	0.600	0.150
44.2368	6	115.20	28.800	7.200	1.800
↑	9	76.800	19.200	4.800	1.200
58.9824	2	460.800	115.200	28.800	7.200
↑	3	307.200	76.800	19.200	4.800
↑	5	184.320	46.080	11.520	2.880
↑	6	153.600	38.400	9.600	2.400
↑	8	115.200	28.800	7.200	1.800
↑	C	76.800	19.200	4.800	1.200
↑	F	61.440	15.360	3.840	0.960
73.728	1	1152.000	288.000	72.000	18.000
↑	3	384.000	96.000	24.000	6.000
↑	6	192.000	48.000	12.000	3.000
↑	A	115.200	28.800	7.200	1.800
↑	C	96.000	24.000	6.000	1.500
↑	F	76.800	19.200	4.800	1.200

Note1: Transfer rates in I/O interface mode are eight times faster than the values given above.

In UART mode, TMRA match detect signal (TA0TRG) can be used for serial transfer clock.

Method for calculating the timer output frequency which is needed when outputting trigger of timer

TA0TRG frequency = Baud rate × 16

Note2: The TMRA0 match detect signal cannot be used as the transfer clock in I/O Interface mode.

(3) Serial clock generation circuit

This circuit generates the basic clock for transmitting and receiving data.

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = “0”, the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.

In SCLK Input Mode with the setting SC0CR<IOC> = “1”, the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

- In UART Mode

The SC0MOD0 <SC1:0> setting determines whether the baud rate generator clock, the internal clock f_{IO}, the match detect signal from timer TMRA0 or the external clock (SCLK0) is used to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART Mode, which counts up the pulses of the SIOCLK clock. It takes 16 SIOCLK pulses to receive 1 bit of data; each data bit is sampled three times - on the 7th, 8th and 9th clock cycles.

The value of the data bit is determined from these three samples using the majority rule.

For example, if the data bit is sampled respectively as “1”, “0” and “1” on 7th, 8th and 9th clock cycles, the received data bit is taken to be 1. A data bit sampled as “0”, “0” and “1” is taken to be “0”.

(5) Receiving control

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = “0”, the RXD0 signal is sampled on the rising or falling edge of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.

In SCLK Input Mode with the setting SC0CR<IOC> = “1”, the RXD0 signal is sampled on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting

- In UART Mode

The receiving control block has a circuit, which detects a start bit using the majority rule. Received bits are sampled three times; when two or more out of three samples are 0, the bit is recognized as the start bit and the receiving operation commences.

The values of the data bits that are received are also determined using the majority rule.

(6) The Receiving Buffers

To prevent Overrun errors, the Receiving Buffers are arranged in a double-buffer structure.
Received data is stored one bit at a time in Receiving Buffer 1 (which is a shift register). When 7 or 8 bits of data have been stored in Receiving Buffer 1, the stored data is transferred to Receiving Buffer 2 (SC0BUF); this causes an INTRX0 interrupt to be generated. The CPU only reads Receiving Buffer 2 (SC0BUF). Even before the CPU reads receiving Buffer 2 (SC0BUF), the received data can be stored in Receiving Buffer 1. However, unless Receiving Buffer 2 (SC0BUF) is read before all bits of the next data are received by Receiving Buffer 1, an overrun error occurs. If an Overrun error occurs, the contents of Receiving Buffer 1 will be lost, although the contents of Receiving Buffer 2 and SC0CR<RB8> will be preserved.
SC0CR<RB8> is used to store either the parity bit - added in 8-Bit UART Mode - or the most significant bit (MSB) - in 9-Bit UART Mode.
In 9-Bit UART Mode the wake-up function for the slave controller is enabled by setting SC0MOD0<WU> to “1”; in this mode INTRX0 interrupts occur only when the value of SC0CR<RB8> is “1”.

SIO interrupt mode is selectable by the register SIMC.

Note1: The double buffer structure does not support SC0CR<RB8>.

Note2: If the CPU reads receive buffer 2 while data is being transferred from receive buffer 1 to receive buffer 2, the data may not be read properly. To avoid this situation, a read of receive buffer 2 should be triggered by a receive interrupt.

(7) Notes for Using Receive Interrupts

- Receive interrupts can be detected either in level or edge mode. For details, see the description of the SIO/SEI receive interrupt mode select register SIMC in the section on interrupts.
- When receive interrupts are set to level mode, once an interrupt occurs, the same interrupt will occur repeatedly even after control has jumped to the interrupt routine unless interrupts are disabled.

(8) Transmission counters

The transmission counter is a 4-bit binary counter used in UART Mode and which, like the receiving counter, counts the SIOCLK clock pulses; a TXDCLK pulse is generated every 16 SIOCLK clock pulses.

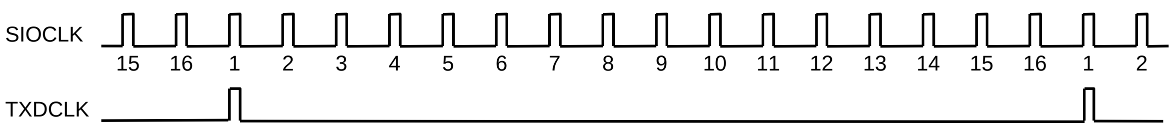


Figure 3.15.5 Generation of the transmission clock

(9) Transmission controller

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = "0", the data in the Transmission Buffer is output one bit at a time to the TXD0 pin on the rising edge or falling of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.
In SCLK Input Mode with the setting SC0CR<IOC> = "1", the data in the Transmission Buffer is output one bit at a time on the TXD0 pin on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

- In UART Mode

When transmission data sent from the CPU is written to the Transmission Buffer, transmission starts on the rising edge of the next TXDCLK.

Handshake function

Use of $\overline{CTS0}$ pin allows data can to be sent in units of one frame; thus, overrun errors can be avoided. The handshake functions is enabled or disabled by the SC0MOD<CTSE> setting.

When the $\overline{CTS0}$ pin goes high on completion of the current data send, data transmission is halted until the $\overline{CTS0}$ pin goes low again. However, the INTTX0 interrupt is generated, and it requests the next data send to from the CPU. The next data is written in the transmission buffer and data sending is halted.

Though there is no $\overline{RTS}$ pin, a handshake function can be easily configured by setting any port assigned to be the $\overline{RTS}$ function. The $\overline{RTS}$ should be output "high" to request send data halt after data receive is completed by software in the RXD interrupt routine.

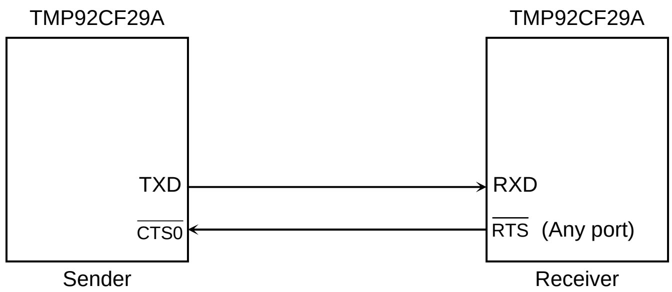
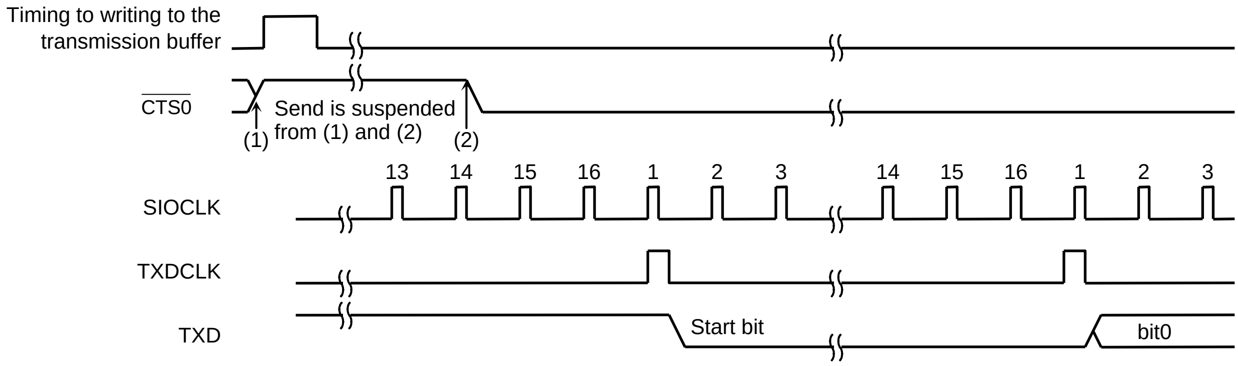


Figure 3.15.6 Handshake function



Note 1: If the $\overline{CTS0}$ signal goes High during transmission, no more data will be sent after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{CTS0}$ signal has fallen.

Figure 3.15.7 $\overline{CTS0}$ (Clear to send) Timing

(10) Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU in order from the least significant bit (LSB). When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

(11) Parity control circuit

When SC0CR<PE> in the serial channel control register is set to “1”, it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> field in the serial channel control register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the transmission buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-bit UART mode or in SC0MOD0<TB8> in 8-bit UART mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the transmission buffer.

In the case of receiving, data is shifted into receiving buffer 1, and the parity is added after the data has been transferred to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-bit UART mode or with SC0CR<RB8> in 8-bit UART mode. If they are not equal, a parity error is generated and the SC0CR<PERR> flag is set.

(12) Error flags

Three error flags are provided to increase the reliability of data reception.

1. Overrun error <OERR>

   If all the bits of the next data item have been received in receiving buffer 1 while valid data still remains stored in receiving buffer 2 (SC0BUF), an overrun error is generated.

   The below is a recommended flow when the overrun error is generated.

   (INTRX interrupt routine)

   1) Read receiving buffer

   2) Read error flag

   3) If <OERR> = “1”

      then

      a) Set to disable receiving (Write “0” to SC0MOD0<RXE>)

      b) Wait to terminate current frame

      c) Read receiving buffer

      d) Read error flag

      e) Set to enable receiving (Write “1” to SC0MOD0<RXE>)

      f) Request to transmit again

   4) Others

Note: Overrun errors are generated only with regard to receive buffer 2 (SC0BUF). Thus, if SC0CR<RB8> is not read, no overrun error will occur.

2. Parity error <PERR>

The parity generated for the data shifted into receiving buffer 2 (SC0BUF) is compared with the parity bit received via the RXD pin. If they are not equal, a parity error is generated.

Note: The parity error flag is cleared every time it is read. However, if a parity error is detected w¥twice in succession and the parity error flag is read between the two parity errors, it may seem as if the flag had not been cleared. To avoid this situation, a read of the parity error flag should be riggered by a receive interrupt.

3. Framing error <FERR>

The stop bit for the received data is sampled three times around the center. If the majority of the samples are "0", a Framing error is generated.

(13) Timing generation

a. In UART Mode

Receiving

Mode	9-Bit (Note)	8-Bit + Parity (Note)	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	—	Center of last bit (parity bit)	Center of stop bit
Overrun error timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit

Note: In 9-Bit and 8-Bit + Parity Modes, interrupts coincide with the ninth bit pulse. Thus, when servicing the interrupt, it is necessary to wait for a 1-bit period (to allow the stop bit to be transferred) to allow checking for a framing error.

Transmitting

Mode	9-Bit	8-Bit + Parity	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Just before stop bit is transmitted	Just before stop bit is transmitted	Just before stop bit is transmitted

b. I/O interface

Transmission Interrupt timing	SCLK Output Mode	Immediately after last bit. (See Figure 3.15.20.)
	SCLK Input Mode	Immediately after rise of last SCLK signal Rising Mode, or immediately after fall in Falling Mode. (See Figure 3.15.21.)
Receiving Interrupt timing	SCLK Output Mode	Timing used to transfer received to data Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.15.22.)
	SCLK Input Mode	Timing used to transfer received data to Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.15.23.)

3.15.3 SFR

		7	6	5	4	3	2	1	0
SC0MOD0 (1202H)	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer data bit 8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up function 0: disable 1: enable	Serial Transmission Mode 00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{IO} 11: External clock (SCLK0 input)	

Serial transmission clock source (UART)

00	TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{IO}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial bontrol register (SC0CR).

Serial Transmission Mode

00	I/O Interface Mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wake-up function

	9-Bit UART	Other Modes
0	Interrupt generated whenever data is received	Don't care
1	Interrupt generated only when <RB8> = "1"	

Receiving Function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{CTS0}$ pin)

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.15.8 Serial Mode Control Register (channel 0, SC0MOD0)

SC1MOD0 (120AH)		7	6	5	4	3	2	1	0
	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transfer data bit 8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up function 0: disable 1: enable	Serial Transmission Mode 00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{IO} 11: External clock (SCLK1 input)	

Serial transmission clock source (UART)

00	TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{IO}
11	External clock (SCLK1 input)

Note: The clock selection for the I/O interface mode is controlled by the serial bontrol register (SC1CR).

Serial Transmission Mode

00	I/O Interface Mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wake-up function

	9-Bit UART	Other Modes
0	Interrupt generated whenever data is received	Don't care
1	Interrupt generated only when <RB8> = "1"	

Receiving Function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{CTS1}$ pin)

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Note: SIO1 can input SIO1 source clock from timer, however, it is possible to use only TMRA0 same with timer of SIO0. Timer differ with SIO0 cannot use. Please be careful.

Figure 3.15.9 Serial Mode Control Register (channel 1, SC1MOD0)

SC0CR (1201H)

A read -modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
Reset State	Undefined	0	0	0	0	0	0	0
Function	Received data bit 8	Parity 0: odd 1: even	Parity addition 0: disable 1: enable	1: error Overrun	Parity	Framing	0: SCLK0 (rising edge) 1: SCLK0 (falling edge)	0: baud rate generator 1: SCLK0 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK pin (Input / Output Mode)

0	Transmits and receives data on rising edge of SCLK0.
1	Transmits and receives data on falling edge SCLK0.

Framing Error flag
Parity Error flag
Overrun Error flag
} Cleared to "0" when read

Parity addition enables

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.15.10 Serial Control Register (channel 0, SC0CR)

SC1CR (1209H)

A read -modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
Reset State	Undefined	0	0	0	0	0	0	0
Function	Received data bit 8	Parity 0: odd 1: even	Parity addition 0: disable 1: enable	1: error Overrun	Parity	Framing	0: SCLK1 (rising edge) 1: SCLK1 (falling edge)	0: baud rate generator 1: SCLK1 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK1 pin input

Edge selection for SCLK pin (Input / Output Mode)

0	Transmits and receives data on rising edge of SCLK1.
1	Transmits and receives data on falling edge SCLK1.

Framing Error flag
Parity Error flag
Overrun Error flag
} Cleared to “0” when read

Parity addition enables

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.15.11 Serial Control Register (channel 1, SC1CR)

BR0CR (1203H)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Always write "0"	+(16−K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency setting			

+(16−K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR0ADD (1204H)

	7	6	5	4	3	2	1	0
bit Symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write					R/W			
Reset State					0	0	0	0
Function					Sets frequency divisor "K" (divided by N + (16-K) / 16)			

Sets baud rate generator frequency divisor

	BR0CR<BR0ADDE> = "1"		BR0CR<BR0ADDE> = "0"
BR0CR <BR0S3:0> / BR0ADD <BR0K3:0>	0000(N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111(N = 15) 0000(N = 16)
0000	Disable	Disable	Divided by N
0001(K = 1) to 1111(K = 15)	Disable	Divided by N + (16-K) /16	

Note1:Availability of +(16-K)/16 division function

N	UART mode	I/O mode
2 to 15	○	×
1 , 16	×	×

The baud rate generator can be set to "1" in UART mode only when the +(16-K)/16 division function is not used. Do not use in I/O interface mode.

Note2:Set BR0CR <BR0ADDE> to "1" after setting K (K = 1 to 15) to BR0ADD<BR0K3:0> when the +(16-K)/16 division function is used. If the unused bits in the BR0ADD register is written, it does not affect operation. If that bits is read, it becomes undefined.

Figure 3.15.12 Baud rate generator control (channel 0, BR0CR, BR0ADD)

BR1CR (120BH)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Always write "0"	+(16–K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency setting			

+(16–K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR1ADD (120CH)

	7	6	5	4	3	2	1	0
bit Symbol					BR1K3	BR1K2	BR1K1	BR1K0
Read/Write					R/W			
Reset State					0	0	0	0
Function					Sets frequency divisor "K" (divided by N + (16-K) / 16)			

Sets baud rate generator frequency divisor

BR1ADD <BR1K3:0> \ BR1CR <BR1S3:0>	BR1CR<BR1ADDE> = "1"		BR1CR<BR1ADDE> = "0"
	0000(N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111(N = 15) 0000(N = 16)
0000	Disable	Disable	Divided by N
0001(K = 1) to 1111(K = 15)	Disable	Divided by N + (16-K) /16	Divided by N

Note1:Availability of +(16-K)/16 division function

N	UART mode	I/O mode
2 to 15	○	×
1 , 16	×	×

The baud rate generator can be set to "1" in UART mode only when the +(16-K)/16 division function is not used. Do not use in I/O interface mode.

Note2:Set BR1CR <BR1ADDE> to "1" after setting K (K = 1 to 15) to BR1ADD<BR1K3:0> when the +(16-K)/16 division function is used. If the unused bits in the BR1ADD register is written, it does not affect operation. If that bits is read, it becomes undefined.

Figure 3.15.13 Baud rate generator control (channel 1, BR1CR, BR1ADD)

SC0BUF (1200H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read-modify-write for SC0BUF.

Figure 3.15.14 Serial Transmission/Receiving Buffer Registers (channel 0, SC0BUF)

SC0MOD1 (1205H)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0						
Read/Write	R/W	R/W						
Reset State	0	0						
Function	IDLE2 0: Stop 1: Run	duplex 0: half 1: full						

Figure 3.15.15 Serial Mode Control Register 1 (channel 0, SC0MOD1)

SC1BUF (1208H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read-modify-write for SC1BUF.

Figure 3.15.16 Serial Transmission/Receiving Buffer Registers (channel 1, SC1BUF)

SC1MOD1 (120DH)

	7	6	5	4	3	2	1	0
Bit symbol	I2S1	FDPX1						
Read/Write	R/W	R/W						
Reset State	0	0						
Function	IDLE2 0: Stop 1: Run	duplex 0: half 1: full						

Figure 3.15.17 Serial Mode Control Register 1 (channel 1, SC1MOD1)

3.15.4 Operation in each mode

(1) Mode 0 (I/O Interface Mode)

This mode allows an increase in the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode includes the SCLK output mode to output synchronous clock SCLK and SCLK input mode to input external synchronous clock SCLK.

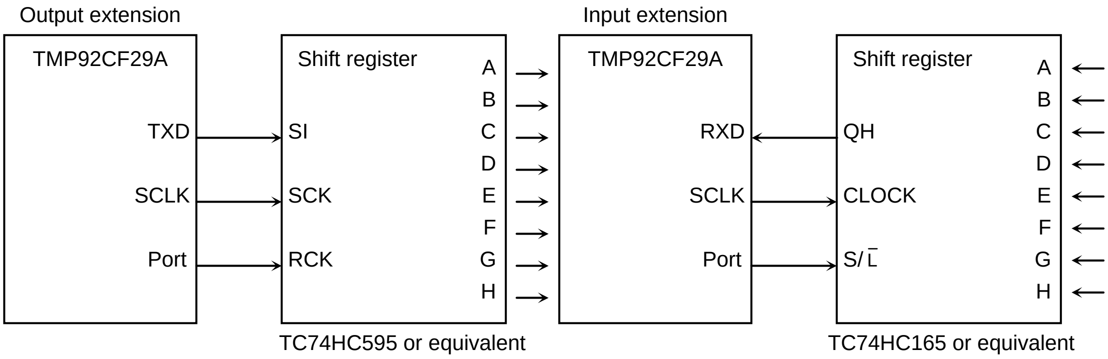


Figure 3.15.18 SCLK Output Mode connection example

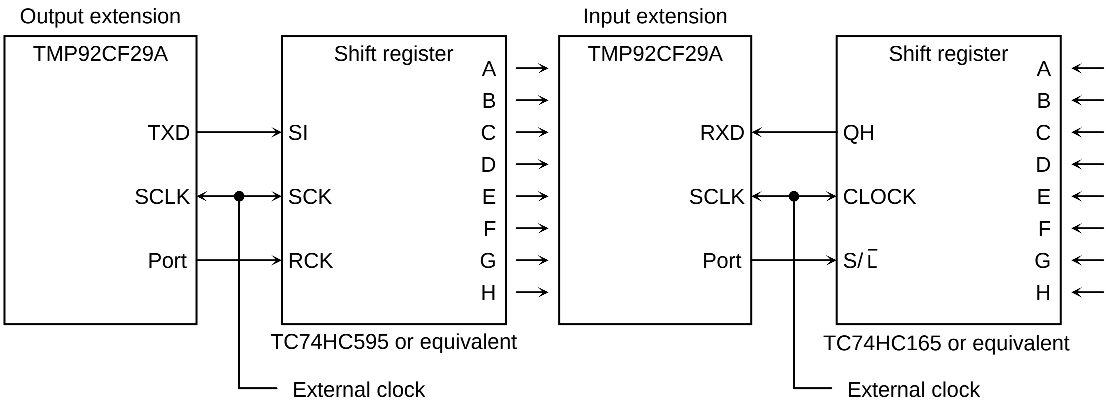


Figure 3.15.19 Example of SCLK Input Mode Connection

a. Transmission

In SCLK output mode 8-bit data and a synchronous clock are output on the TXD0 and SCLK0 pins respectively each time the CPU writes the data to the Transmission Buffer. When all data is output, INTES0 <ITX0C> will be set to generate the INTTX0 interrupt.

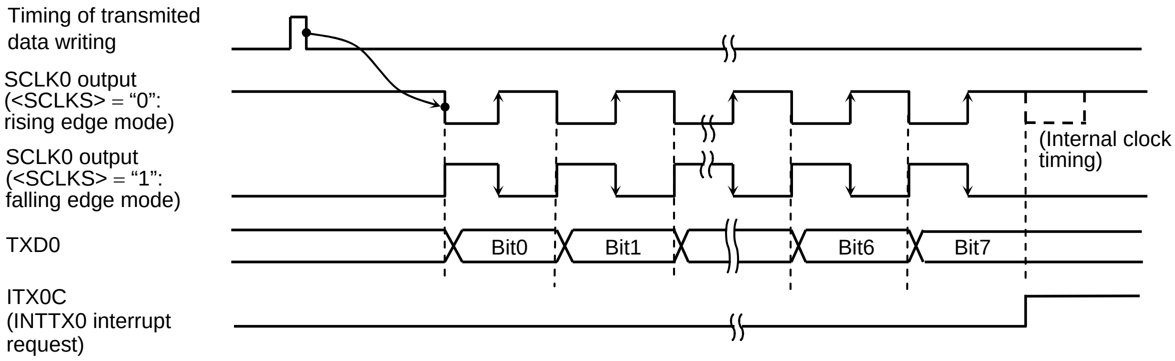


Figure 3.15.20 Transmitting Operation in I/O Interface Mode (SCLK0 Output Mode)

In SCLK Input Mode, 8-bit data is output on the TXD0 pin when the SCLK0 input becomes active after the data has been written to the Transmission Buffer by the CPU.

When all data is output, INTES0 <ITX0C> will be set to generate INTTX0 interrupt.

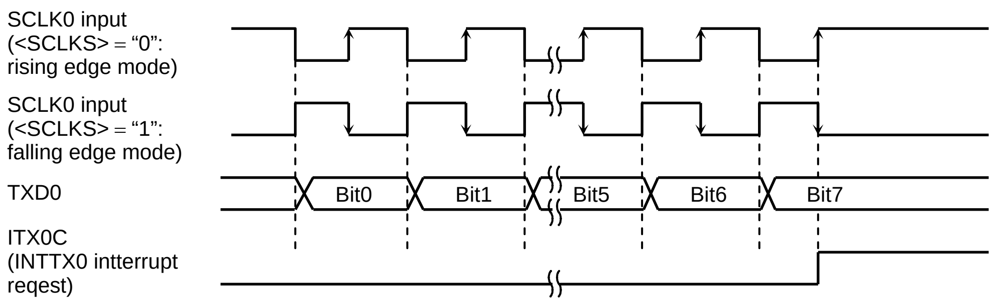


Figure 3.15.21 Transmitting Operation in I/O Interface Mode (SCLK0 Input Mode)

b. Receiving

In SCLK Output Mode the synchronous clock is output on the SCLK0 pin and the data is shifted to Receiving Buffer 1. This is initiated when the Receive Interrupt flag INTES0<IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is transferred to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0<IRX0C> is set to "1" again, causing an INTRX0 interrupt to be generated.
Setting SC0MOD0<RXE> to "1" initiates SCLK0 output.

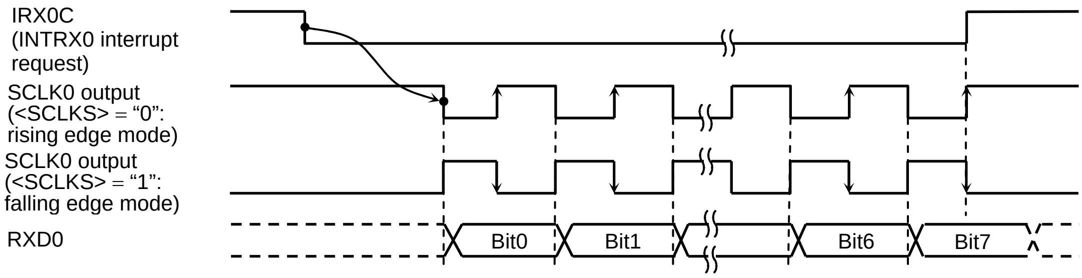


Figure 3.15.22 Receiving operation in I/O Interface Mode (SCLK0 Output Mode)

In SCLK Input Mode the data is shifted to Receiving Buffer 1 when the SCLK input goes active. The SCLK input goes active when the Receive Interrupt flag INTES0 <IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is shifted to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0 <IRX0C> is set to "1" again, causing an INTRX0 interrupt to be generated.

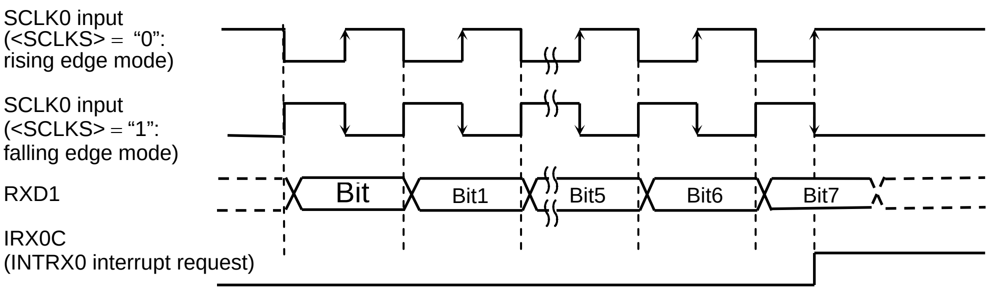


Figure 3.15.23 Receiving Operation in I/O interface Mode (SCLK0 Input Mode)

Note: The system must be put in the receive-enable state (SC0MOD0<RXE> = "1") before data can be received.

c. Transmission and Receiving (Full Duplex Mode)

When Full Duplex Mode is used, set the Receive Interrupt Level to 0, and only set the interrupt level (from 1 to 6) of the transmit interrupt. Ensure that the program which transmits the interrupt reads the receiving buffer before setting the next transmit data.

The following is an example of this:

Example: Channel 0, SCLK output
Baud rate = 9600 bps
f_{SYS} = 2.4576 MHz

Main routine

		7	6	5	4	3	2	1	0	
INTES0		X	0	0	1	X	0	0	0	Set the INTTX0 level to 1. Set the INTRX0 level to 0.
P9CR		X	X	X	X	X	1	0	1	Set P90, P91 and P92 to function as the TXD0,
P9FC		–	–	X	X	X	1	X	1	RXD0 and SCLK0 pins respectively.
SC0MOD0		–	–	–	–	0	0	–	–	Select I/O interface mode.
SC0MOD1		–	1	X	X	X	X	X	X	Select full duplex mode.
SC0CR		–	–	–	–	–	–	0	0	SCLK0 output mode, select rising edge
BR0CR		0	0	0	1	1	0	0	0	Baud rate = 9600 bps.
SC0MOD0		–	–	1	–	–	–	–	–	Enable receiving.
SC0BUF		*	*	*	*	*	*	*	*	Set the transmit data and start.

INTTX0 interrupt routine

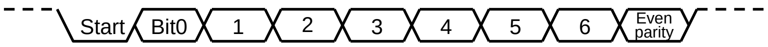
A_{CC}	←	SC0BUF								Read the receiving buffer.
SC0BUF		*	*	*	*	*	*	*	*	Set the next transmit data.

X: Don't care, –: No change

(2) Mode 1 (7-bit UART Mode)

7-Bit UART Mode is selected by setting the Serial Channel Mode Register SC0MOD0<SM1:0> field to “01”.
In this mode a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the Serial Channel Control Register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to “1” (enabled).

Setting example: When transmitting data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 2400 bps at f_{SYS} = 19.6608 MHz)

		7	6	5	4	3	2	1	0	
P9CR	←	X	X	X	X	X	–	–	1	Set P90 to function as the TXD0 pin.
P9FC	←	–	–	X	X	X	–	X	1	
SC0MOD0	←	X	0	–	X	0	1	0	1	Select 7-bit UART mode.
SC0CR	←	–	1	1	–	–	–	–	–	Add even parity.
BR0CR	←	0	0	1	0	1	0	0	0	Set the transfer rate to 2400 bps.
INTES0	←	X	1	0	0	X	0	0	0	Enable the INTTX0 interrupt and set it to interrupt level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

X: Don't care, –: No change

(3) Mode 2 (8-Bit UART Mode)

8-Bit UART Mode is selected by setting SC0MOD0<SM1:0> to “10”. In this mode a parity bit can be added (use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to “1” (enabled).

Setting example: When receiving data of the following format, the control registers should be set as described below

← Transmission direction (Transmission rate: 9600 bps at f_{SYS} = 19.6608 MHz)

Main routine

		7	6	5	4	3	2	1	0	
P9CR	←	X	X	X	X	X	–	0	–	Set P91 to function as the RXD0 pin.
P9FC	←	–	–	X	X	X	–	X	–	
SC0MOD0	←	–	–	1	–	1	0	0	1	Enable receiving in 8-bit UART mode.
SC0CR	←	–	0	1	–	–	–	–	–	Add odd parity.
BR0CR	←	0	0	0	1	1	0	0	0	Set the transfer rate to 9600 bps.
INTES0	←	X	1	0	0	X	0	0	0	Enable the INTTX0 interrupt and set it to interrupt level 4.

Interrupt routine

A_{CC}	←	SC0CR AND 00011100	Check for errors
if $A_{CC} \neq 0$ then ERROR			
A_{CC}	←	SC0BUF	Read the received data

X: Don't care, –: No change

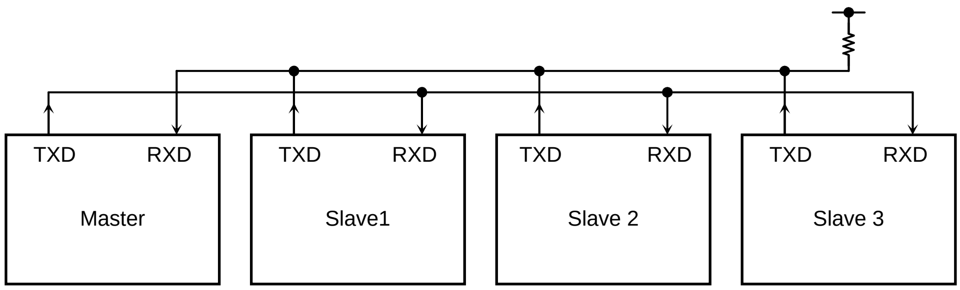
(4) Mode 3 (9-Bit UART Mode)

9-Bit UART Mode is selected by setting SC0MOD0<SM1:0> to “11”. In this mode a parity bit cannot be added.

In the case of transmission the MSB (9th bit) is written to SC0MOD0<TB8>. In the case of receiving it is stored in SC0CR<RB8>. When the buffer is written or read, <TB8> or <RB8> is read or written first, before the rest of the SC0BUF data.

Wake-up function

In 9-Bit UART Mode, the wake-up function for slave controllers is enabled by setting SC0MOD0<WU> to “1”. The interrupt INTRX0 can only be generated when<RB8> = “1”.

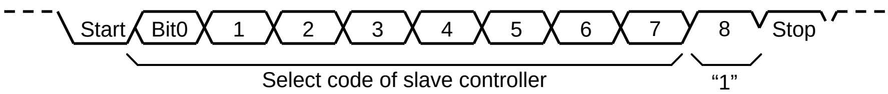


Note: The TXD pin of each slave controller must be in Open-Drain Output Mode.

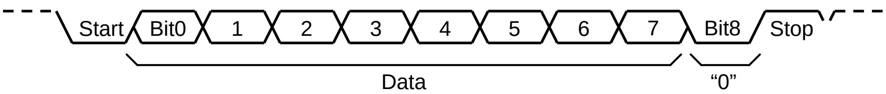
Figure 3.15.24 Serial Link using Wake-up function

Protocol

1. Select 9-Bit UART Mode on the master and slave controllers.
2. Set the SC0MOD0<WU> bit on each slave controller to "1" to enable data receiving.
3. The master controller transmits data one frame at a time. Each frame includes an 8-bit select code which identifies a slave controller. The MSB (bit 8) of the data (<TB8>) is set to "1".

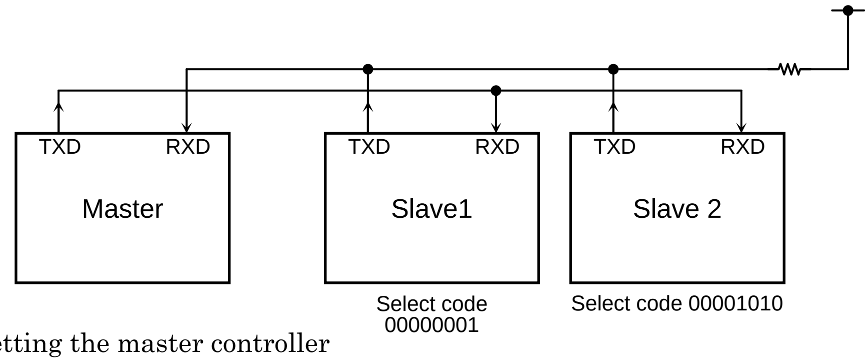


4. Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its <WU> bit to "0".
5. The master controller transmits data to the specified slave controller (the controller whose SC0MOD0<WU> bit has been cleared to "0"). The MSB (bit 8) of the data (<TB8>) is cleared to "0".



6. The other slave controllers (whose <WU> bits remain at 1) ignore the received data because their MSBs (bit 8 or <RB8>) are set to "0", disabling INTRX0 interrupts.
   The slave controller whose <WU> bit = "0" can also transmit to the master controller. In this way it can signal the master controller that the data transmission from the master controller has been completed.

Setting example: To link two slave controllers serially with the master controller using the internal clock f_{IO} as the transfer clock.



Setting the master controller

Main routine

P9CR	←	X X X X X − 0 1	Set P90 and P91 to function as the TXD0 and RXD0 pins respectively.
P9FC	←	− − X X X − X 1	
INTES0	←	X 1 0 0 X 1 0 1	Enable the INTTX0 interrupt and set it to Interrupt Level 4. Enable the INTRX0 interrupt and set it to Interrupt Level 5.
SC0MOD0	←	1 0 1 0 1 1 1 0	Set f_{IO} as the transmission clock for 9-Bit UART Mode.
SC0BUF	←	0 0 0 0 0 0 0 1	Set the select code for slave controller 1.

Interrupt routine (INTTX0)

SC0MOD0	←	0 − − − − − − −	Set TB8 to “0”.
SC0BUF	←	* * * * * * * *	Set data for transmission.

Setting the slave controller

Main routine

P9CR	←	X X X X X − 0 1	Select P91 and P90 to function as the RXD0 and TXD0 pins respectively (open-drain output).
P9FC	←	− − X X X − X 1	
P9FC2	←	X X X X X X X 1	
INTES0	←	X 1 0 0 X 1 0 0	Enable INTRX0 and INTTX0.
SC0MOD0	←	0 0 1 1 1 1 1 0	Set <WU> to “1” in 9-Bit UART Transmission Mode using f_{IO} as the transfer clock.

Interrupt routine (INTRX0)

Acc ← SC0BUF
if Acc =Select code
Then SC0MOD0 ← − − − 0 − − − − Clear <WU> to “0”.

3.15.5 Support for IrDA

SIO0 and SIO1 include support for the IrDA 1.0 infrared data communication specification.

Figure 3.15.25 shows the block diagram.

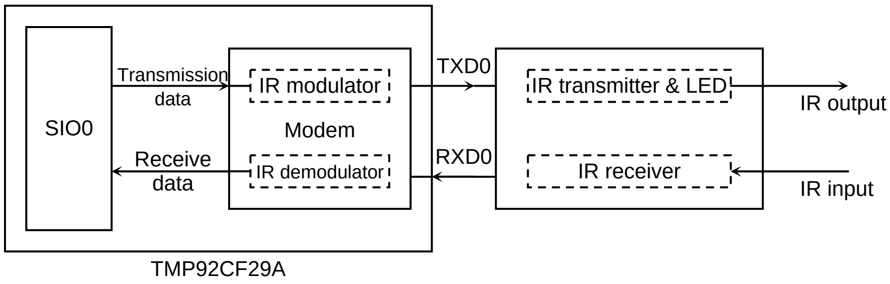


Figure 3.15.25 Block Diagram

(1) Modulation of the transmission data

When the transmit data is "0", the modem outputs 1 to TXD0 pin with either 3/16 or 1/16 times for width of baud-rate. The pulse width is selected by the SIR0CR<PLSEL>. When the transmit data is "1", the modem outputs "0".

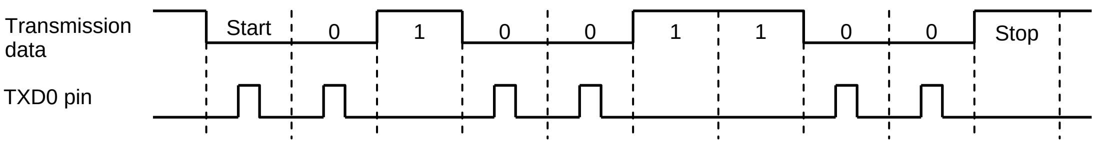


Figure 3.15.26 Transmission example

(2) Modulation of the receive data

When the receive data has an effective pulse width of pulse "1", the modem outputs "0" to SIO0. Otherwise the modem outputs "1" to SIO0. The effective pulse width is selected by SIR0CR<SIR0WD3:0>.

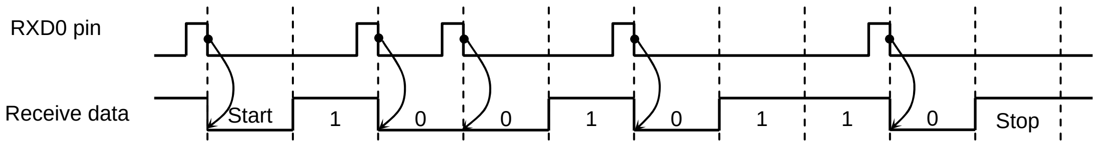


Figure 3.15.27 Receiving example

(3) Data format

The data format is fixed as follows:

- Data length: 8-bit
- Parity bits: none
- Stop bits: 1bit

(4) SFR

Figure 3.15.28 shows the control register SIR0CR. Set SIR0CR data while SIO0 is stopped. The following example describes how to set this register:

1)	SIO setting	; Set the SIO to UART Mode.
	↓	
2)	LD (SIR0CR), 07H	; Set the receive data pulse width to 16×.
3)	LD (SIR0CR), 37H	; TXEN, RXEN Enable the Transmission and receiving.
	↓	
4)	Start transmission and receiving for SIO0	; The modem operates as follows: • SIO0 starts transmitting. • IR receiver starts receiving.

(5) Notes

1. Baud rate for IrDA

   When IrDA is operated, set 01 to SC0MOD0<SC1:0> to generate baud-rate.

   Setting other than the above (TA0TRG, f_{IO} and SCLK0-input) cannot be used.

2. The pulse width for transmission

   The IrDA 1.0 specification is defined in Table 3.15.4.

Table 3.15.4 Baud rate and pulse width specifications

Baud Rate	Modulation	Rate Tolerance (% of rate)	Pulse Width (minimum)	Pulse Width (typical)	Pulse width (maximum)
2.4 kbps	RZI	±0.87	1.41 μs	78.13 μs	88.55 μs
9.6 kbps	RZI	±0.87	1.41 μs	19.53 μs	22.13 μs
19.2 kbps	RZI	±0.87	1.41 μs	9.77 μs	11.07 μs
38.4 kbps	RZI	±0.87	1.41 μs	4.88 μs	5.96 μs
57.6 kbps	RZI	±0.87	1.41 μs	3.26 μs	4.34 μs
115.2 kbps	RZI	±0.87	1.41 μs	1.63 μs	2.23 μs

The infra-red pulse width is specified either baud rate T× 3/16 or 1.6 μs (1.6 μs is equal to 3/16 pulse width when baud rate is 115.2 kbps).

The TMP92CF29A has a function which can select the pulse width of Transmission as either 3/16 or 1/16. However, 1/16 pulse width can only be selected when the baud rate is equal to or less than 38.4 kbps.
For the same reason, the + (16 – k)/16 division functions in the baud rate generator of SIO0 cannot be used to generate a 115.2 kbps baud rate. The + (16-K)/16 division function cannot be used also when the baud rate is 38.4 kbps and the pulse width is 1/16.

Table 3.15.5 Baud rate and pulse width for (16 – K) / 16 division function

Pulse Width	Baud Rate					
	115.2 Kbps	57.6 Kbps	38.4 Kbps	19.2 Kbps	9.6 Kbps	2.4 Kbps
T × 3/16	× (Note)	○	○	○	○	○
T × 1/16	–	–	×	○	○	○

○: (16 – K)/16 division function can be used.
×: (16 – K)/16 division function cannot be used.
–: Cannot be set to 1/16 pulse width

Note: (16 – K)/16 division function can be used under special conditions.

SIR0CR (1207H)

	7	6	5	4	3	2	1	0
Bit symbol	PLSEL	RXSEL	TXEN	RXEN	SIR0WD3	SIR0WD2	SIR0WD1	SIR0WD0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0: "H" pulse 1: "L" pulse	Transmit 0: disable 1: enable	Receive 0: disable 1: enable	Select receive pulse width Set effective pulse width to equal to more than 2x × (value + 1) + 100ns Can be set : 1 to 14 Can not be set : 0, 15			

Select receive pulse width

Formula: Effective pulse width ≥ 2x × (value + 1) + 100ns

x = 1/fSYS

0000	Cannot be set
0001	Equal or more than 4x + 100ns
to	
1110	Equal or more than 30x + 100ns
1111	Can not be set

Receive (recovery) operation

0	Disable receiving operation (Received data is ignored)
1	Enabled receiving operation

Transmit (modulation) operation

0	Disabled transmission operation (Input from SIO is ignored)
1	Enabled transmission operation

Select transmit pulse width

0	3/16 pulse width
1	1/16 pulse width

Note: If a pulse width complying with the IrDA1.0 standard (1.6 μs min.) can be guaranteed with a low baud rate, setting this bit to "1" shortens the duration of infrared ray activation, resulting in reduced power dissipation.

Figure 3.15.28 IrDA Control Register (for SIO0)

SIR1CR (120FH)

	7	6	5	4	3	2	1	0
Bit symbol	PLSEL	RXSEL	TXEN	RXEN	SIR1WD3	SIR1WD2	SIR1WD1	SIR1WD0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0: "H" pulse 1: "L" pulse	Transmit 0: disable 1: enable	Receive 0: disable 1: enable	Select receive pulse width Set effective pulse width to equal to more than 2x × (value + 1) + 100ns Can be set : 1 to 14 Can not be set : 0, 15			

Select receive pulse width

Formula: Effective pulse width $\geq$ 2x $\times$ (value $+$ 1) $+$ 100ns

x =1/fSYS

0000	Cannot be set
0001	Equal or more than 4x + 100ns
to	
1110	Equal or more than 30x + 100ns
1111	Can not be set

Receive (recovery) operation

0	Disable receiving operation (Received data is ignored)
1	Enabled receiving operation

Transmit (modulation) operation

0	Disabled transmission operation (Input from SIO is ignored)
1	Enabled transmission operation

Select transmit pulse width

0	3/16 pulse width
1	1/16 pulse width

Note: If a pulse width complying with the IrDA1.0 standard (1.6 μs min.) can be guaranteed with a low baud rate, setting this bit to "1" shortens the duration of infrared ray activation, resulting in reduced power dissipation.

Figure 3.15.29 IrDA Control Register (for SIO1)

3.16 Serial Bus Interface (SBI)

The TMP92CF29A has a 1-channel serial bus interface which an I2C bus mode. This circuit supports only I2C bus mode (Multi master).

The serial bus interface is connected to an external device through PV6 (SDA) and PV7 (SCL) in the I2C bus mode.
Each pin is specified as follows.

	PVFC2<PV7F2, PV6F2>	PVCR<PV7C, PV6C>	PVFC<PV7F, PV6F>
I2C bus mode	11	11	11

3.16.1 Configuration

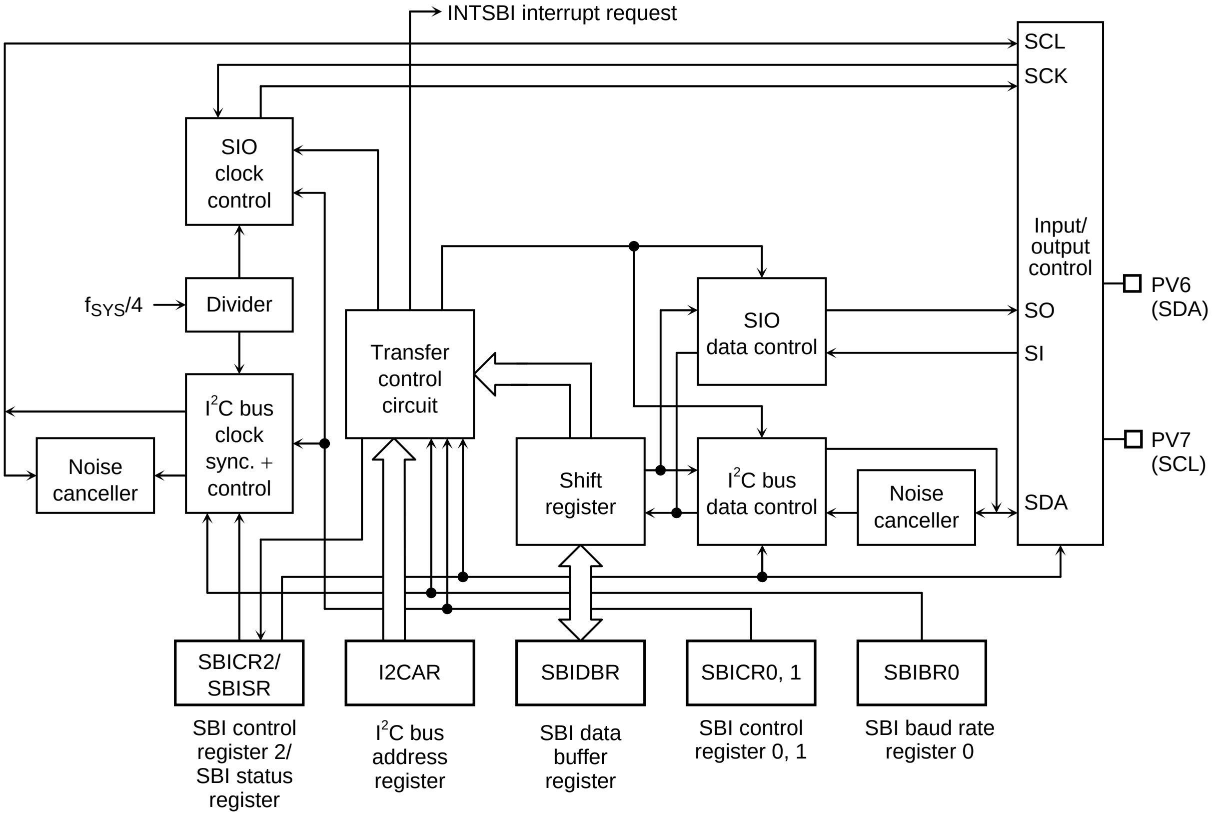


Figure 3.16.1 Serial bus interface (SBI)

3.16.2 Serial Bus Interface (SBI) Control

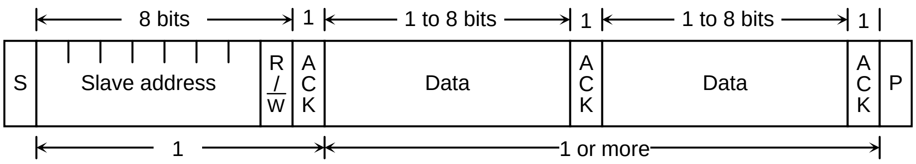
The following registers are used to control the serial bus interface and monitor the operation status.

- Serial bus interface control register 0 (SBICR0)
- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)
- Serial bus interface baud rate register 0 (SBIBR0)

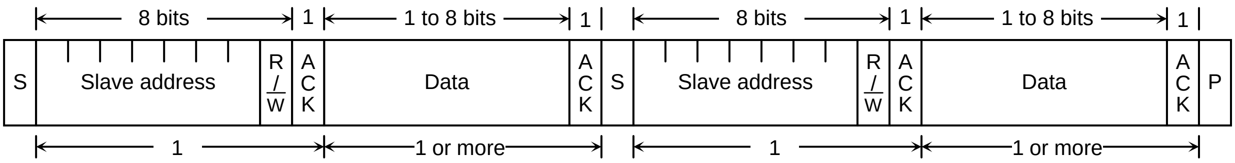
3.16.3 The Data Formats in the I²C Bus Mode

The data formats in the I²C bus mode is shown below.

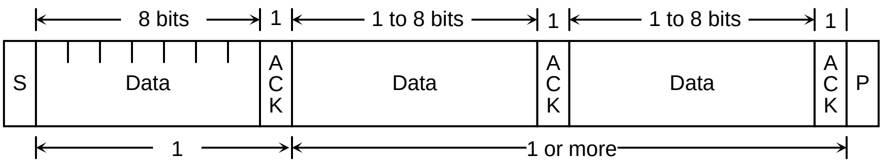
(a) Addressing format



(b) Addressing format (with restart)



(c) Free data format (data transferred from master device to slave device)



S: Start condition

R/$\overline{W}$: Direction bit

ACK: Acknowledge bit

P: Stop condition

Figure 3.16.2 Data format in the I²C bus mode

3.16.4 I²C Bus Mode Control Register

The following registers are used to control and monitor the operation status when using the serial bus interface (SBI) in the I²C bus mode.

Serial Bus Interface Control Register 0

		7	6	5	4	3	2	1	0
SBICR0 (1247H)	Bit symbol	SBIEN	–	–	–	–	–	–	–
	Read/Write	R/W	R						
	Reset State	0	0	0	0	0	0	0	0
A read-modify-write operation cannot be performed	Function	SBI operation 0 : disable 1 : enable	Always read "0".						

<SBIEN> : When using SBI, <SBIEN> should be set "1" (SBI operation enable) before setting each register of SBI module.

Figure 3.16.3 Registers for the I²C bus mode

Serial Bus Interface Control Register 1

SBICR1 (1240H)
A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	BC2	BC1	BC0	ACK	−	SCK2	SCK1	SCK0/ SWRMON
Read/Write	R/W			R/W	R	R/W		R/W
Reset State	0	0	0	0	1	0	0	0/1(Note2)
Function	Number of transferred bits (Note 1)			Acknowledge mode specification 0: Not generate 1:Generate	Always read as “1”.	Internal serial clock selection and software reset monitor		

Internal serial clock selection <SCK2:0> at write
f_{SYS}=80MHz (Output to SCL pin), Clock gear = fc/1

000	n = 4	− (Note3)	System Clock: f_{SYS} (=80MHz)
001	n = 5	− (Note3)	Clock Gear : fc/1
010	n = 6	− (Note3)	$fscl = \frac{f_{SYS}/4}{2^n + 36}$ [Hz]
011	n = 7	− (Note3)	
100	n = 8	68 kHz	
101	n = 9	36 kHz	
110	n = 10	19 kHz	
111	(Reserved)	(Reserved)	

Software reset state monitor <SWRMON> at read

0	During software reset
1	(Initial Data)

Acknowledge mode specification

0	Not generate clock pulse for acknowledge signal
1	Generate clock pulse for acknowledge signal

Number of bits transferred

<BC2:0>	<ACK> = 0		<ACK> = 1	
	Number of clock pulses	Bits	Number of clock pulses	Bits
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Note1: For the frequency of the SCL line clock, see 3.15.5 (3) Serial clock.

Note2: The initial data of SCK0 is “0”, the initialdata of SWRMON is “1” if SBI operation is enable (SBICR0<SBIEN> = “1”). If SBI operation is disable (SBICR0<SBIEN> = “0”), the initialdata of SWRMON is “0”.

Note3: This I^2C bus circuit does not support Fast-mode, it supports the Standard mode only. Although the I^2C bus circuit itself allows the setting of a baud rate over 100kbps, the compliance with the I^2C specification is not guaranteed in that case.

Figure 3.16.4 Registers for the I^2C bus mode

Serial Bus Interface Control Register 2

SBICR2 (1243H)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
Read/Write	W				W (Note 1)		W (Note 1)	
Reset State	0	0	0	1	0	0	0	0
Function	Master/Slave selection 0:Slave 1:Master	Transmitter /Receiver selection 0:Receiver 1:Transmitter	Start/Stop condition Generation 0:Generate stop condition 1:Generate start condition	Cancel INTSBI interrupt request 0:Don't care 1:Cancel interrupt request	Serial bus interface operating mode selection (Note 2) 00: Port mode 01: Reserved 10: I²C Bus mode 11: Reserved		Software reset generate write "10" and "01", then an internal reset signal is generated.	

Serial bus interface operating mode selection (Note2)

00	Port Mode (Serial Bus Interface output disabled)
01	Reserved
10	I²C Bus Mode
11	Reserved

Note 1: Reading this register functions as SBISR register.

Note 2: Switch a mode to port mode after confirming that the bus is free.

Switch a mode between I²C bus mode and port mode after confirming that input signals via port are high-level.

Figure 3.16.5 Registers for the I²C bus mode

Table 3.16.1Resolution of base clock

@f_{SYS} = 80MHz

Clock Gear <GEAR1:0>	Base Clock Resolution
000(fc)	$f_{SYS}/2^2$ (50ns)
001(fc/2)	$f_{SYS}/2^3$ (0.1μs)
010(fc/4)	$f_{SYS}/2^4$ (0.2μs)
011(fc/8)	$f_{SYS}/2^5$ (0.4μs)
100(fc/16)	$f_{SYS}/2^6$ (0.8μs)

Serial Bus Interface Status Register

SBISR (1243H)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
Read/Write	R							
Reset State	0	0	0	1	0	0	0	0
Function	Master/ Slave status monitor 0:Slave 1:Master	Transmitter/ Receiver status monitor 0:Receiver 1:Tranmitter	I^2C bus status monitor 0:Free 1:Busy	INTSBI interrupt request monitor 0: Interrupt requested 1: Interrupt canceled	Arbitration lost detection monitor 0: – 1: Detected	Slave address match detection monitor 0:Undetected 1: Detected	GENERAL CALL detection monitor 0:Undetected 1: Detected	Last received bit monitor 0: 0 1: 1

Last received bit monitor

0	Last received bit was 0
1	Last received bit was 1

GENERAL CALL detection monitor

0	Undetected
1	GENERAL CALL detected

Slave address match detection monitor

0	Slave address don't match or Undetected
1	Slave address match or GENERAL CALL detected

Arbitration lost detection monitor

0	–
1	Arbitration lost

Note1: Writing in this register functions as SBICR2.

Note2: The initialdata SBISR<PIN> is "1" if SBI operation is enable (SBICR0<SBIEN>="1"). If SBI operation is disable (SBICR0<SBIEN>="0"), the initialdata of SBISR<PIN> is "0".

Figure 3.16.6 Registers for the I^2C bus mode

Serial Bus Interface Baud Rate Register 0

SBIBR0 (1244H) A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI	–	–	–	–	–	–
Read/Write	W	R/W	R					R/W
Reset State	0	0	1	1	1	1	1	0
Function	Always read “0”	IDLE2 0: Stop 1: Run	Always read as “1”					Always write “0”.

Operation during IDLE 2 mode

0	Stop
1	Operation

Serial Bus Interface Data Buffer Register

SBIDBR (1241H) A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (received)/W (transfer)							
Reset State	Undefined							

Note1: When writing transmitted data, start from the MSB (bit 7).Receiving data is placed from LSB(bit0).

Note2: SBIDBR can’t be read the written data because of it has buffer for writing and buffer for reading individually.Therefore Read modify write instruction (e.g.“BIT” instruction) is prohibitted.

I^2C Bus Address Register

I2CAR (1242H) A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Slave address selection for when device is operating as slave device							Address recognition mode specification

Address recognition mode specification

0	Slave address recognition
1	Non slave address recognition

Figure 3.16.7 Registers for the I^2C bus mode

3.16.5 Control in I^2C Bus Mode

(1) Acknowledge Mode Specification

When slave address is matched or detecting GENERAL CALL, and set the SBICR1<ACK> to “1”, TMP92CF29A operates in the acknowledge mode. The TMP92CF29A generates an additional clock pulse for an Acknowledge signal when operating in Master Mode. In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the Low in order to generate the acknowledge signal.

Clear the <ACK> to “0” for operation in the Non-Acknowledge Mode; The TMP92CF29A does not generate a clock pulse for the Acknowledge signal when operating in the Master Mode.

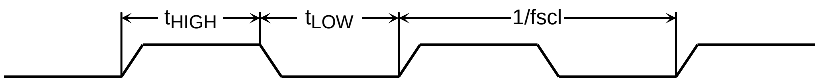
(2) Number of transfer bits

The SBICR1<BC2:0> is used to select a number of bits for next transmitting and receiving data.
Since the <BC2:0> is cleared to 000 as a start condition, a slave address and direction bit transmission are executed in 8 bits. Other than these, the <BC2:0> retains a specified value.

(3) Serial clock

a. Clock source

The SBICR1 <SCK2:0> is used to select a maximum transfer frequency outputted on the SCL pin in Master Mode. Set a communication baud rates that meets the I^2C bus specification, such as the shortest pulse width of t_{LOW}, based on the equations shown below.



$t_{LOW} = (2^{n-1} + 29)/(f_{SYS}/4)$

$t_{HIGH} = (2^{n-1} + 6)/(f_{SYS}/4)$

$$fscl = 1/(t_{LOW} + t_{HIGH}) = \frac{f_{SYS}/4}{2^n + 36}$$

SBICR1<SCK2:0>	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

Figure 3.16.8 Clock source

b. Clock synchronization

In the I²C bus mode, in order to wired-AND a bus, a master device which pulls down a clock line to low-level, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse. The master device with a high-level clock pulse needs to detect the situation and implement the following procedure.

The TMP92CF29A has a clock synchronization function for normal data transfer even when more than one master exists on the bus.

The example explains the clock synchronization procedures when two masters simultaneously exist on a bus.

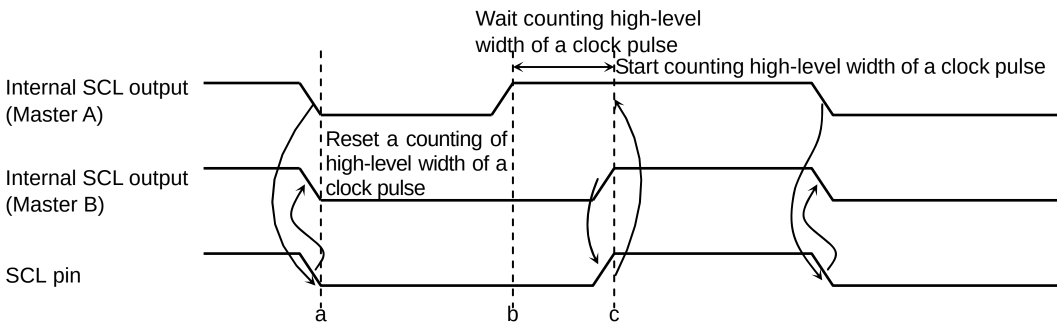


Figure 3.16.9 Clock synchronization

As Master A pulls down the internal SCL output to the Low level at point “a”, the SCL line of the bus becomes the Low-level. After detecting this situation, Master B resets a counter of High-level width of an own clock pulse and sets the internal SCL output to the Low-level.

Master A finishes counting Low-level width of an own clock pulse at point “b” and sets the internal SCL output to the High-level. Since Master B holds the SCL line of the bus at the Low-level, Master A wait for counting high-level width of an own clock pulse. After Master B finishes counting low-level width of an own clock pulse at point “c” and Master A detects the SCL line of the bus at the High-level, and starts counting High-level of an own clock pulse. The clock pulse on the bus is determined by the master device with the shortest High-level width and the master device with the longest Low-level width from among those master devices connected to the bus.

(4) Slave address and address recognition mode specification

When the TMP92CF29A is used as a slave device, set the slave address <SA6:0> and <ALS> to the I2CAR. Clear the <ALS> to “0” for the address recognition mode.

(5) Master/Slave selection

Set the SBICR2<MST> to “1” for operating the TMP92CF29A as a master device. Clear the SBICR2<MST> to “0” for operation as a slave device. The <MST> is cleared to “0” by the hardware after a stop condition on the bus is detected or arbitration is lost.

(6) Transmitter/Receiver selection

Set the SBICR2<TRX> to "1" for operating the TMP92CF29A as a transmitter. Clear the <TRX> to "0" for operation as a receiver.

In Slave Mode,

- Data with an addressing format is transferred
- A slave address with the same value that an I2CAR
- A GENERAL CALL is received (all 8-bit data are "0" after a start condition)

The <TRX> is set to "1" by the hardware if the direction bit (R/$\overline{W}$) sent from the master device is "1", and is cleared to "0" by the hardware if the bit is "0".

In the Master Mode, after an Acknowledge signal is returned from the slave device, the <TRX> is cleared to "0" by the hardware if a transmitted direction bit is "1", and is set to "1" by the hardware if it is "0". When an Acknowledge signal is not returned, the current condition is maintained.
The <TRX> is cleared to "0" by the hardware after a stop condition on the I²C bus is detected or arbitration is lost.

(7) Start/Stop condition generation

When the SBISR<BB> is "0", slave address and direction bit which are set to SBIDBR are output on a bus after generating a start condition by writing "1" to the SBICR2 <MST, TRX, BB, PIN>. It is necessary to set transmitted data to the data buffer register (SBIDBR) and set "1" to <ACK> beforehand.

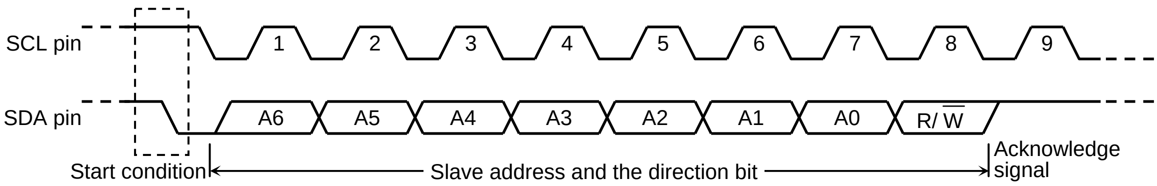
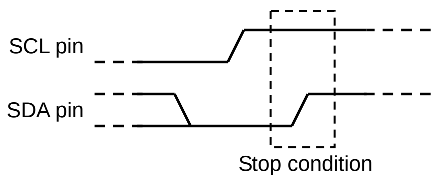


Figure 3.16.10 Start condition generation and slave address generation

When the <BB> is "1", a sequence of generating a stop condition is started by writing "1" to the <MST, TRX, PIN>, and "0" to the <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until a stop condition is generated on a bus.

Figure 3.16.11 Stop condition generation



The state of the bus can be ascertained by reading the contents of SBISR<BB>. SBISR<BB> will be set to 1 if a start condition has been detected on the bus, and will be cleared to 0 if a stop condition has been detected.

(8) Interrupt service requests and interrupt cancellation

When a serial bus interface interrupt request (INTSBI) occurs, the SBICR2 <PIN> is cleared to “0”. During the time that the SBICR2<PIN> is “0”, the SCL line is pulled down to the Low level.
The <PIN> is cleared to “0” when a 1-word of data is transmitted or received. Either writing/reading data to/from SBIDBR sets the <PIN> to “1”.
The time from the <PIN> being set to “1” until the SCL line is released takes tLOW.
In the address recognition mode (<ALS> = “0”), <PIN> is cleared to “0” when the received slave address is the same as the value set at the I2CAR or when a GENERAL CALL is received (all 8-bit data are “0” after a start condition). Although SBICR2<PIN> can be set to “1” by the program, the <PIN> is not clear it to “0” when it is written “0”.

(9) Serial bus interface operation mode selection

SBICR2<SBIM1:0> is used to specify the serial bus interface operation mode. Set SBICR2< SBIM1:0> to “10” when the device is to be used in I²C Bus Mode after confirming pin condition of serial bus interface to “H”.
Switch a mode to port after confirming a bus is free.

(10) Arbitration lost detection monitor

Since more than one master device can exist simultaneously on the bus in I²C Bus Mode, a bus arbitration procedure has been implemented in order to guarantee the integrity of transferred data.

In case set start condition bit with bus is busy, start condition is not output on SCL and SDA pin, but arbitration lost is generated.
Data on the SDA line is used for I²C bus arbitration.
The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on the bus. Master A and Master B output the same data until point “a”. After Master A outputs “L” and Master B, “H”, the SDA line of the bus is wire-AND and the SDA line is pulled down to the Low-level by Master A. When the SCL line of the bus is pulled up at point b, the slave device reads the data on the SDA line, that is, data in Master A. A data transmitted from Master B becomes invalid. The state in Master B is called “ARBITRATION LOST”. Master B device which loses arbitration releases the internal SDA output in order not to affect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

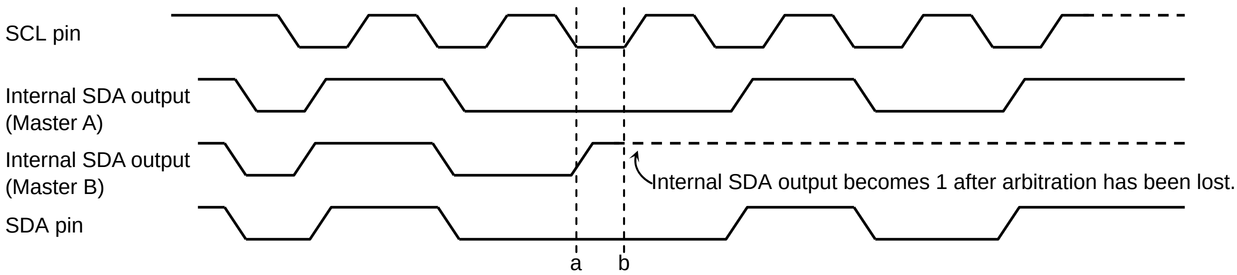


Figure 3.16.12 Arbitration lost

The TMP92CF29A compares the levels on the bus's SDA line with those of the internal SDA output on the rising edge of the SCL line. If the levels do not match, arbitration is lost and SBISR<AL> is set to "1".
When SBISR<AL> is set to "1", SBISR<MST, TRX> are cleared to "00" and the mode is switched to Slave Receiver Mode. Thus, clock output is stopped in data transfer after setting <AL>="1".

SBISR<AL> is cleared to "0" when data is written to or read from SBIDBR or when data is written to SBICR2.

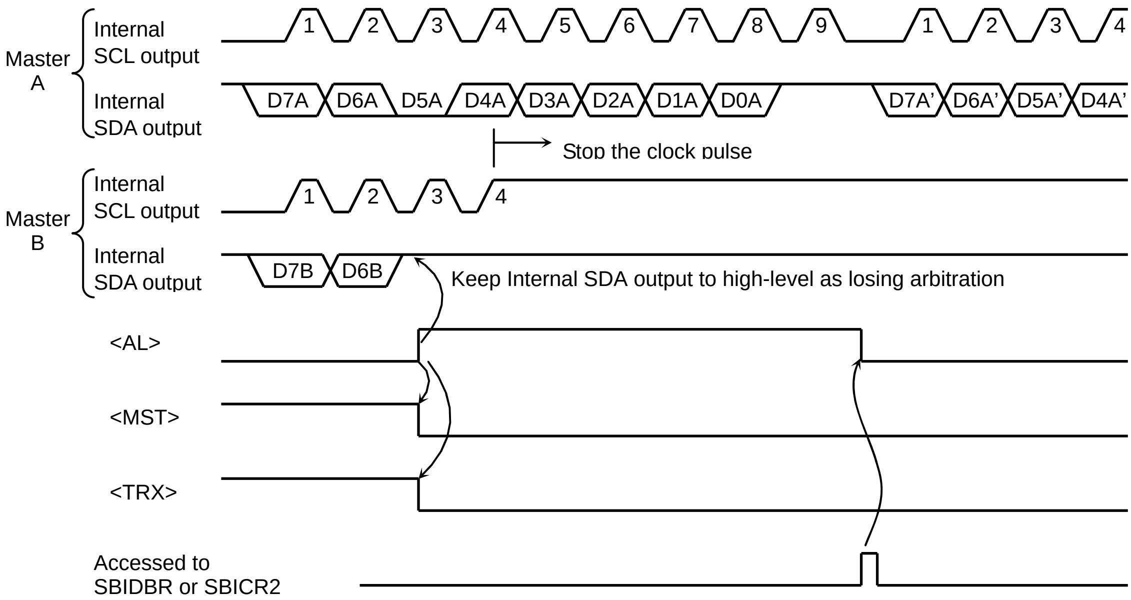


Figure 3.16.13 Example of when TMP92CF29A is a master device B

(D7A = D7B, D6A = D6B)

(11) Slave address match detection monitor

SBISR<AAS> is set to "1" in Slave Mode, in Address Recognition Mode (i.e. when I2CAR<ALS> = "0"), when a GENERAL CALL is received, or when a slave address matches the value set in I2CAR. When I2CAR<ALS> = "1", SBISR<AAS> is set to "1" after the first word of data has been received. SBISR<AAS> is cleared to "0" when data is written to or read from the data buffer register SBIDBR.

(12) GENERAL CALL detection monitor

SBISR<AD0> is set to "1" in Slave Mode, when a GENERAL CALL is received (all 8-bit received data is "0", after a start condition). SBISR<AD0> is cleared to "0" when a start condition or stop condition is detected on the bus.

(13) Last received bit monitor

The SDA line value stored at the rising edge of the SCL line is set to the SBISR<LRB>. In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the SBISR<LRB>.

(14) Software Reset function

The software Reset function is used to initialize the SBI circuit, when SBI is rocked by external noises, etc.
An internal Reset signal pulse can be generated by setting SBICR2<SWRST1:0> to "10" and "01". This initializes the SBI circuit internally. All command registers and status registers are initialized as well.
SBICR1<SWRMON>is automatically set to "1" after the SBI circuit has been initialized.

Note: If the software reset is executied , operation selection is reset, and its mode is set to port mode from I^2C mode.

(15) Serial Bus Interface Data Buffer Register (SBIDBR)

The received data can be read and transferred data can be written by reading or writing the SBIDBR.
In the master mode, after the start condition is generated the slave address and the direction bit are set in this register.

(16) I^2CBUS Address Register (I2CAR)

I2CAR<SA6:0> is used to set the slave address when the TMP92CF29A functions as a slave device.
The slave address output from the master device is recognized by setting the I2CAR<ALS> to "0". The data format is the addressing format. When the slave address is not recognized at the <ALS> = "1", the data format is the free data format.

(17) Setting register for IDLE2 mode operation (SBIBR0)

SBIBR0<I2SBI> is the register setting operation/stop during IDLE2-mode. Therefore, setting <I2SBI> is necessary before the HALT instruction is executed.

3.16.6 Data Transfer in I^2C Bus Mode

(1) Device initialization

Set the SBICR1<ACK, SCK2:0>, Set SBIBR1 to “1” and clear bits 7 to 5 and 3 in the SBICR1 to “0”.
Set a slave address <SA6:0> and the <ALS> (<ALS> = “0” when an addressing format) to the I2CAR.
For specifying the default setting to a slave receiver mode, clear “0” to the <MST, TRX, BB> and set “1” to the <PIN>, “10” to the <SBIM1:0>.

```
           7 6 5 4 3 2 1 0
SBICR1  ←  0 0 0 X 0 X X X      Set acknowledge and SCL clock.
I2CAR   ←  X X X X X X X X      Set slave address and address recognition mode.
SBICR2  ←  0 0 0 1 1 0 0 0      Set to slave receiver mode.
Note: X: Don’t care
```

(2) Start condition and slave address generation

a. Master Mode

In the Master Mode, the start condition and the slave address are generated as follows.
Check a bus free status (when <BB> = “0”).
Set the SBICR1<ACK> to “1” (Acknowledge Mode) and specify a slave address and a direction bit to be transmitted to the SBIDBR.
When SBICR2<BB> = “0”, the start condition are generated by writing “1111” to SBICR2<MST, TRX, BB, PIN>. Subsequently to the start condition, nine clocks are output from the SCL pin. While eight clocks are output, the slave address and the direction bit which are set to the SBIDBR. At the 9th clock, the SDA line is released and the acknowledge signal is received from the slave device.
An INTSBI interrupt request occurs at the falling edge of the 9th clock. The <PIN> is cleared to “0”. In the Master Mode, the SCL pin is pulled down to the Low-level while <PIN> is “0”. When an interrupt request occurs, the <TRX> is changed according to the direction bit only when an acknowledge signal is returned from the slave device.

Setting in main routine

```
             7 6 5 4 3 2 1 0
 ┌→ Reg.     ← SBISR
 │  Reg.     ← Reg. e 0x20
 │  if Reg.  ≠ 0x00                 Wait until bus is free.
 └─ Then
    SBICR1   ← X X X 1 X X X X      Set to acknowledgement mode.
    SBIDBR1  ← X X X X X X X X      Set slave address and direction bit.
    SBICR2   ← 1 1 1 1 1 0 0 0      Generate start condition.
```

```
In INTSBI interrupt routine
        INTCLR ← 0X2a      Clear the interrupt request
        Process
        End of interrupt
```

b. Slave Mode

In the Slave Mode, the start condition and the slave address are received. After the start condition is received from the master device, while eight clocks are output from the SCL pin, the slave address and the direction bit that are output from the master device are received.
When a GENERAL CALL or the same address as the slave address set in I2CAR is received, the SDA line is pulled down to the Low-level at the 9th clock, and the acknowledge signal is output.
An INTSBI interrupt request occurs on the falling edge of the 9th clock. The <PIN> is cleared to "0". In Slave Mode the SCL line is pulled down to the Low-level while the <PIN> = "0".

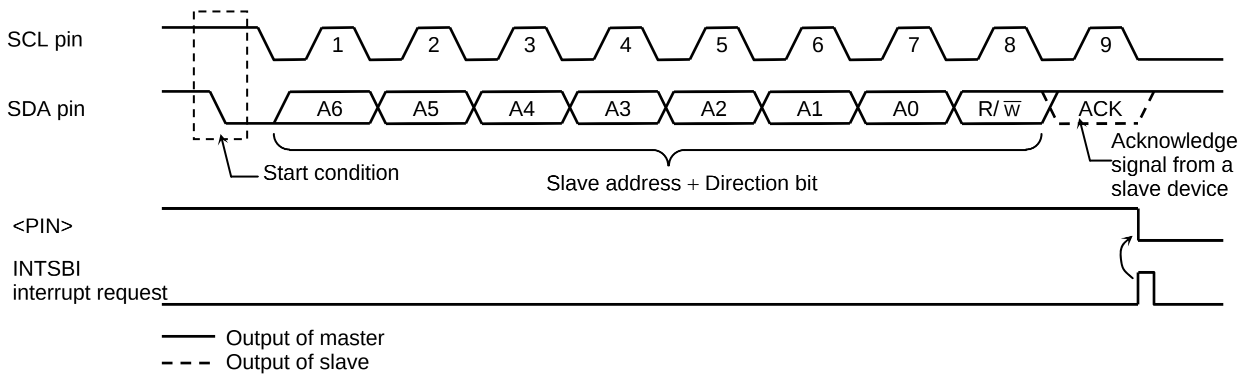


Figure 3.16.14 Start condition generation and slave address transfer

(3) 1-word Data Transfer

Check the <MST> by the INTSBI interrupt process after the 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. If <MST> = "1" (Master Mode)

Check the <TRX> and determine whether the mode is a transmitter or receiver.

When the <TRX> = "1" (Transmitter mode)

Check the <LRB>. When <LRB> is "1", a receiver does not request data. Implement the process to generate a stop condition (Refer to 3.15.6 (4)) and terminate data transfer.

When the <LRB> is "0", the receiver is requests new data. When the next transmitted data is 8 bits, write the transmitted data to SBIDBR. When the next transmitted data is other than 8 bits, set the <BC2:0> <ACK> and write the transmitted data to SBIDBR. After written the data, <PIN> becomes "1", a serial clock pulse is generated for transferring a new 1-word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, an INTSBI interrupt request occurs. The <PIN> becomes "0" and the SCL line is pulled down to the Low-level. If the data to be transferred is more than one word in length, repeat the procedure from the <LRB> checking above.

```
INTSBI interrupt
if MST = 0
Then shift to the process when slave mode
if TRX = 0
Then shift to the process when receiver mode.
if LRB = 0
Then shift to the process that generates stop condition.
                7 6 5 4 3 2 1 0
   SBICR1   ←  X X X X X X X X        Set the bit number of transmit and ACK.
   SBIDBR   ←  X X X X X X X X        Write the transmit data.
   End of interrupt
   Note: X: Don't care
```

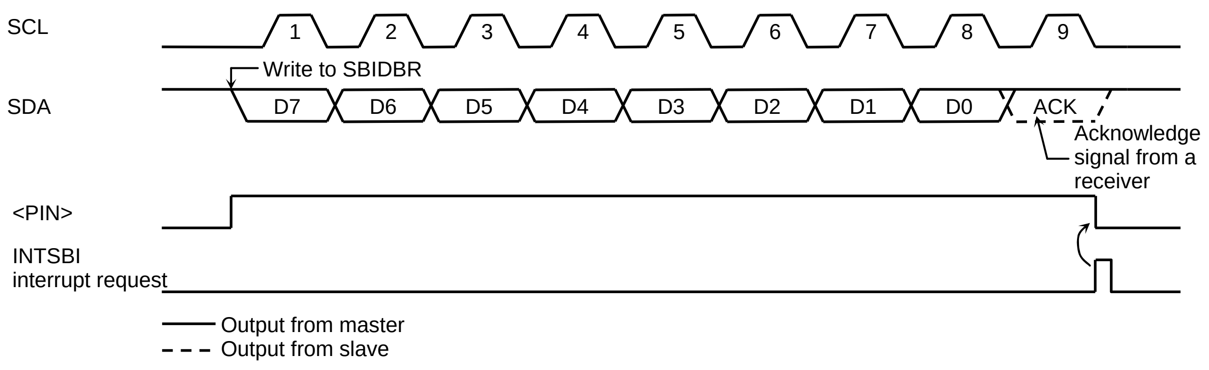


Figure 3.16.15 Example in which <BC2:0> = "000" and <ACK> = "1" in transmitter mode

When the <TRX> is “0” (Receiver mode)

When the next transmitted data is other than 8 bits, set <BC2:0> <ACK> and read the received data from SBIDBR to release the SCL line (data which is read immediately after a slave address is sent is undefined). After the data is read, <PIN> becomes “1”.

Serial clock pulse for transferring new 1 word of data is defined SCL and outputs “L” level from SDA pin with acknowledge timing.

An INTSBI interrupt request then occurs and the <PIN> becomes “0”, Then the TMP92CF29A pulls down the SCL pin to the Low-level. The TMP92CF29A outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

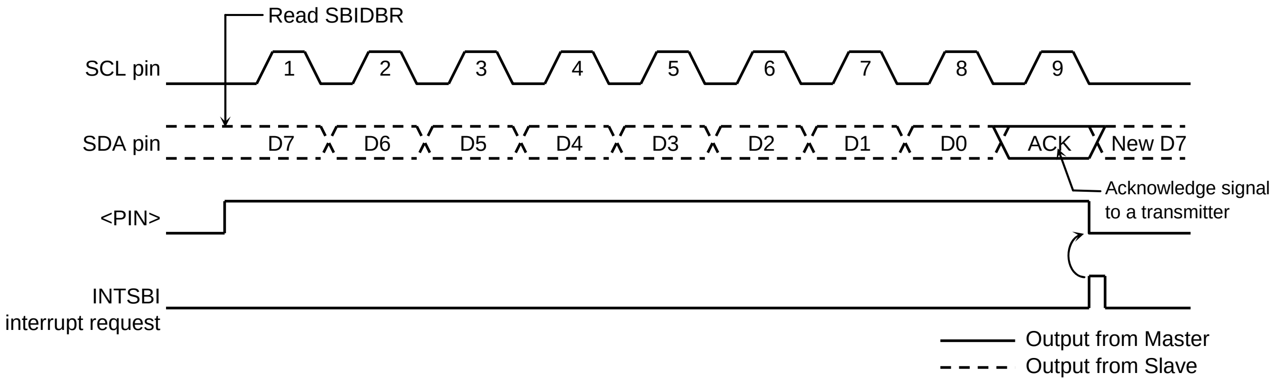


Figure 3.16.16 Example of when <BC2:0> = “000”, <ACK> = “1” in receiver mode

In order to terminate the transmission of data to a transmitter, clear <ACK> to “0” before reading data which is 1-word before the last data to be received. The last data word does not generate a clock pulse as the Acknowledge signal. After the data has been transmitted and an interrupt request has been generated, set <BC2:0> to “001” and read the data. The TMP92CF29A generates a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line on the bus remains High. The transmitter interprets the High signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.
After the one data bit has been received and an interrupt request been generated, the TMP92CF29A generates a stop condition (see Section 3.16.6 (4) Stop condition generation) and terminates data transfer.

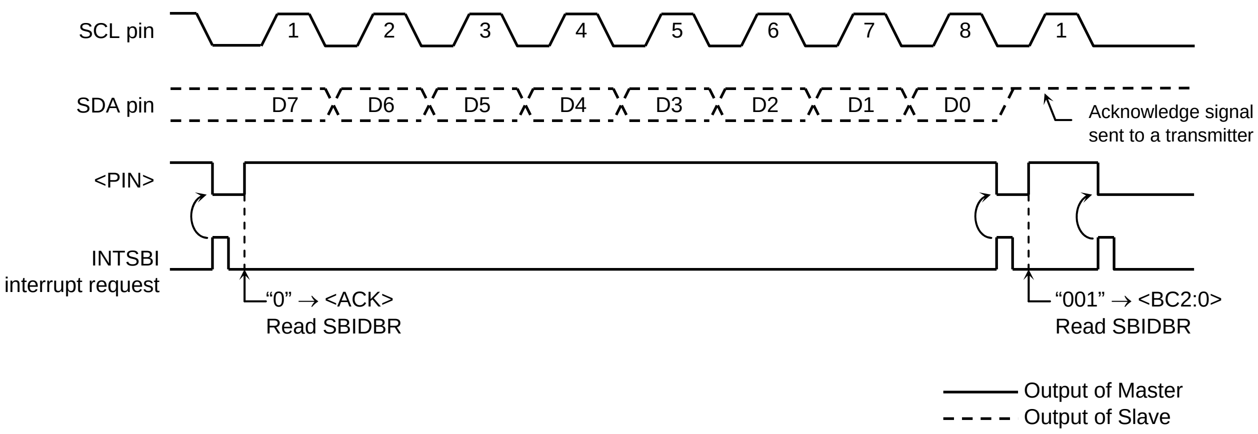


Figure 3.16.17 Termination of data transfer in master receiver mode

Example: In case receive data N times

INTSBI interrupt (After transmitting data)

		7 6 5 4 3 2 1 0	
SBICR1	←	X X X X X X X X	Set the bit number of receive data and ACK.
Reg.	←	SBIDBR	Load the dummy data.
End of interrupt			

INTSBI interrupt (Receive data of 1st to (N−2) th)

		7 6 5 4 3 2 1 0	
Reg.	←	SBIDBR	Load the data of 1st to (N−2)th.
End of interrupt			

INTSBI interrupt ((N−1) th Receive data)

		7 6 5 4 3 2 1 0	
SBICR1	←	X X X 0 0 X X X	Not generate acknowledge signal
Reg.	←	SBIDBR	Load the data of (N−1)th
End of interrupt			

INTSBI interrupt (Nth Receive data)

		7 6 5 4 3 2 1 0	
SBICR1	←	0 0 1 0 0 X X X	Generate the clock for 1bit transmit
Reg.	←	SBIDBR	Receive the data of Nth.
End of interrupt			

INTSBI interrupt (After receiving data)

The process of generating stop condition	Finish the transmit of data
End of interrupt	

Note: X: Don’t care

b. If <MST> = 0 (Slave Mode)

In the slave mode the TMP92CF29A operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBI interrupt request occurs when the TMP92CF29A receives a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is complete, or after matching received address. In the master mode, the TMP92CF29A operates in a slave mode if it losing arbitration. An INTSBI interrupt request occurs when a word data transfer terminates after losing arbitration. When an INTSBI interrupt request occurs the <PIN> is cleared to “0” and the SCL pin is pulled down to the Low-level. Either reading/writing from/to the SBIDBR or setting the <PIN> to “1” will release the SCL pin after taking tLOW time.
Check the SBISR<AL>, <TRX>, <AAS>, and <AD0> and implements processes according to conditions listed in the next table.

```
  Example: In case matching slave address in slave receive mode, direction bit is “1”.
  INTSBI interrupt
  if TRX = 0
  Then shift to other process
  if AL = 1
  Then shift to other process
  if AAS = 0
  Then shift to other process
                   7 6 5 4 3 2 1 0
SBICR1     ←  X X X 1 X X X X        Set the bit number of transmit.
SBIDBR     ←  X X X X X X X X        Set the data of transmit.
Note: X: Don’t care
```

Table 3.16.2 Operation in the slave mode

<TRX>	<AL>	<AAS>	<AD0>	Conditions	Process
1	1	1	0	The TMP92CF29A loses arbitration when transmitting a slave address and receives a slave address for which the value of the direction bit sent from another master is “1”.	Set the number of bits a word in <BC2:0> and write the transmitted data to SBIDBR
	0	1	0	In Salve Receiver Mode, the TMP92CF29A receives a slave address for which the value of the direction bit sent from the master is “1”.	
		0	0	In Salve Transmitter Mode, a single word of is transmitted.	Check the <LRB> setting. If <LRB> is set to “1”, set <PIN> to “1” since the receiver win no request the data which follows. Then, clear <TRX> to “0” to release the bus. If <LRB> is cleared to “0”, set <BC2:0> to the number of bits in a word and write the transmitted data to SBIDBR since the receiver requests next data.
0	1	1	1/0	The TMP92CF29A loses arbitration when transmitting a slave address and receives a slave address or GENERAL CALL for which the value of the direction bit sent from another master is “0”.	Read the SBIDBR for setting the <PIN> to “1” (reading dummy data) or set the <PIN> to “1”.
		0	0	The TMP92CF29A loses arbitration when transmitting a slave address or data and terminates word data transfer.	
	0	1	1/0	In Slave Receiver Mode, the TMP92CF29A receives a slave address or GENERAL CALL for which the value of the direction bit sent from the master is “0”.	
		0	1/0	In Slave Receiver Mode, the TMP92CF29A terminates receiving word data.	Set <BC2:0> to the number of bits in a word and read the received data from SBIDBR.

(4) Stop condition generation

When SBISR<BB> = “1”, the sequence for generating a stop condition start by writing “1” to SBICR2<MST, TRX, PIN> and “0” to SBICR2<BB>. Do not modify the contents of SBICR2<MST, TRX, PIN, BB> until a stop condition has been generated on the bus. When the bus’s SCL line has been pulled Low by another device, the TMP92CF29A generates a stop condition when the other device has released the SCL line and SDA pin rising.

		7	6	5	4	3	2	1	0	
SBICR2	←	1	1	0	1	1	0	0	0	Generate stop condition.

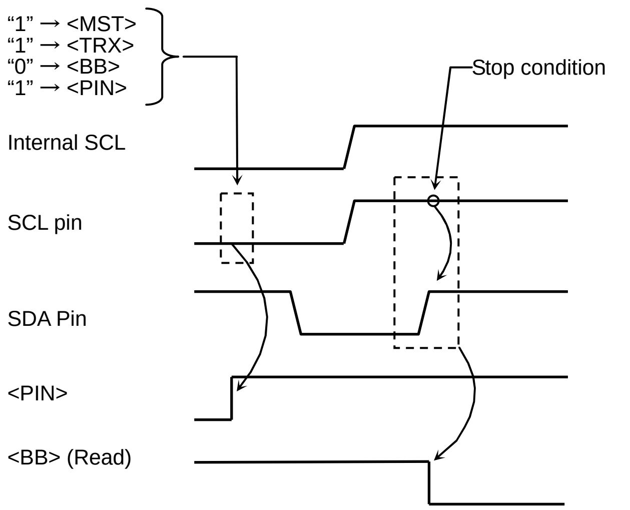


Figure 3.16.18 Stop condition generation (Single master)

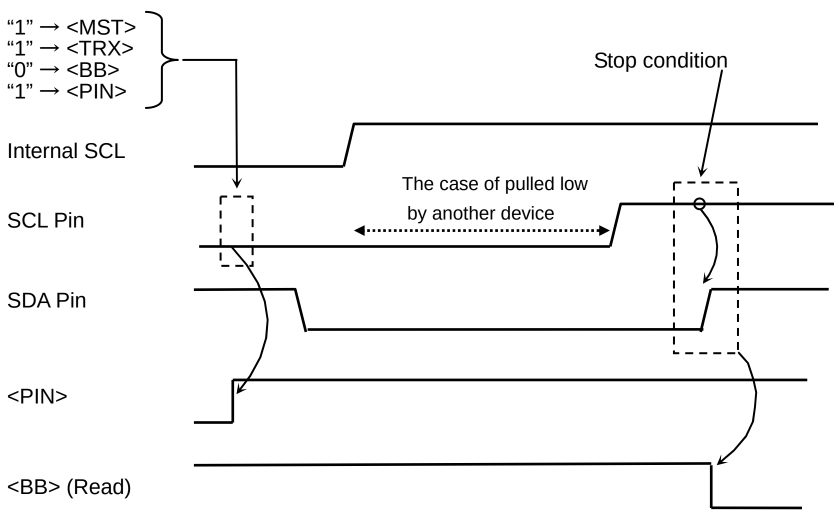


Figure 3.16.19 Stop condition generation (Multi master)

(5) Restart

Restart is used during data transfer between a master device and a slave device to change the data transfer direction.

The following description explains how to restart when the TMP92CF29A is in Master Mode.

Clear SBICR2<MST, TRX, and BB> to “0” and set SBICR2<PIN> to “1” to release the bus. The SDA line remains High and the SCL pin is released. Since a stop condition has not been generated on the bus, other devices assume the bus to be in busy state.

And confirm SCL pin, that SCL pin is released and become bus-free state by SBISR<BB> = “0” or signal level “1” of SCL pin in port mode. Check the <LRB> until it becomes 1 to check that the SCL line on a bus is not pulled down to the low-level by other devices. After confirming that the bus remains in a free state, generate a start condition using the procedure described in (2).

In order to satisfy the set-up time requirements when restarting, take at least 4.7 μs of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

```
                7 6 5 4 3 2 1 0
┌→ SBICR2  ←   0 0 0 1 1 0 0 0     Release the bus
└─ if SBISR<BB> ≠ 0                Check if SCL pin is released.
┌→ Then
└─ if SBISR<LRB> ≠ 1               Check if SCL pin of other device is “L” level.
   Then
   4.7 μs Wait
   SBICR1  ←   X X X 1 X X X X     Set acknowledgement mode.
   SBIDBR  ←   X X X X X X X X     Set the slave address and direction bit.
   SBICR2  ←   1 1 1 1 1 0 0 0     Generate start condition.
```

Note: X: Don’t care

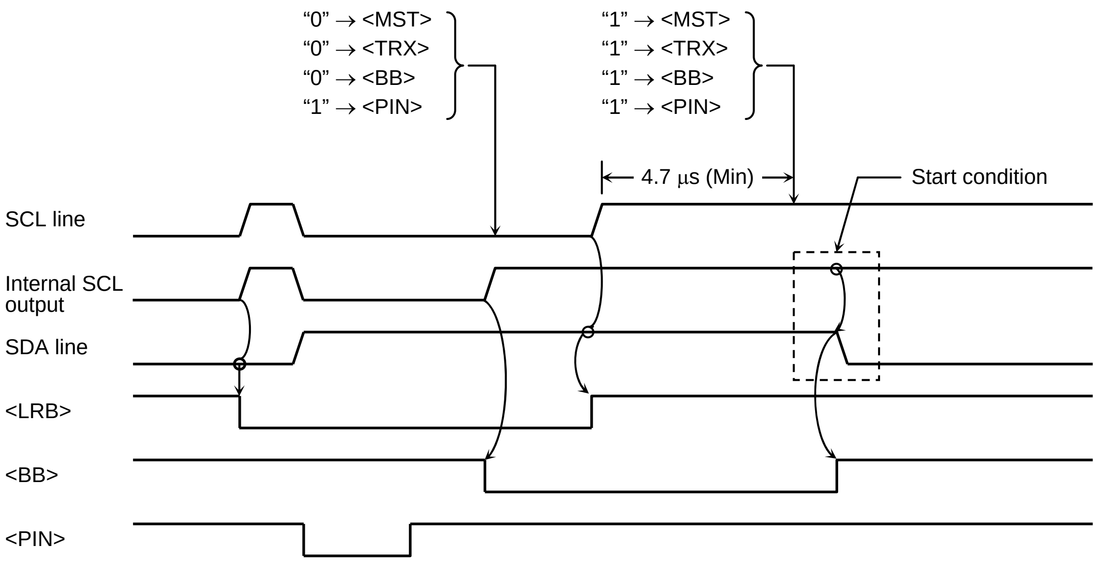


Figure 3.16.20 Timing chart for generate restart

Note: Don’t write <MST> = “0”, when <MST> = “0” condition. (Cannot be restarted)

3.17 USB Controller

3.17.1 Outline

This USB controller (UDC) is designed to support a variety of serial links in the construction of a USB system.

The outline is as follows:

(1) Compliant with USB rev1.1

(2) Full-speed: 12 Mbps (low-speed (1.5 Mbps) not supported)

(3) Auto bus enumeration with 384-byte descriptor RAM

(4) Supports 3 kinds of transfer type: Control, interrupt and bulk

- Endpoint 0: Control 64 bytes × 1-FIFO
- Endpoint 1: BULK (out) 64 bytes × 2-FIFOs
- Endpoint 2: BULK (in) 64 bytes × 2-FIFOs
- Endpoint 3: Interrupt (in) 8 bytes × 1-FIFO

(5) Built-in DPLL which generates sampling clock for receive data

(6) Detecting and generating SOP, EOP, RESUME, RESET and TIMEOUT

(7) Encoding and decoding NRZI data

(8) Inserting and discarding stuffed bit

(9) Detecting and checking CRC

(10) Generating and decoding packet ID

(11) Built-in power management function

(12) Dual packet mode supported

Note1:The TMP92CF29A does not include the pull-up resister necessary for D+pin. An external pull-up resistor plus software support is required.

Note2:There are some differences between our specifications and USB 1.1. Refer to check “3.17.11 Notice and Restrictions”.

3.17.1.1 System Configuration

The USB controller (UDC) consists of the following 3 blocks.

1. 900/H1 CPU I/F (details given in Section 3.17.2, below).
2. UDC core block (DPLL, SIE, IFM and PWM), request controller, descriptor RAM and 4 endpoint FIFO (details given in Section 3.17.3, below).
3. USB transceiver

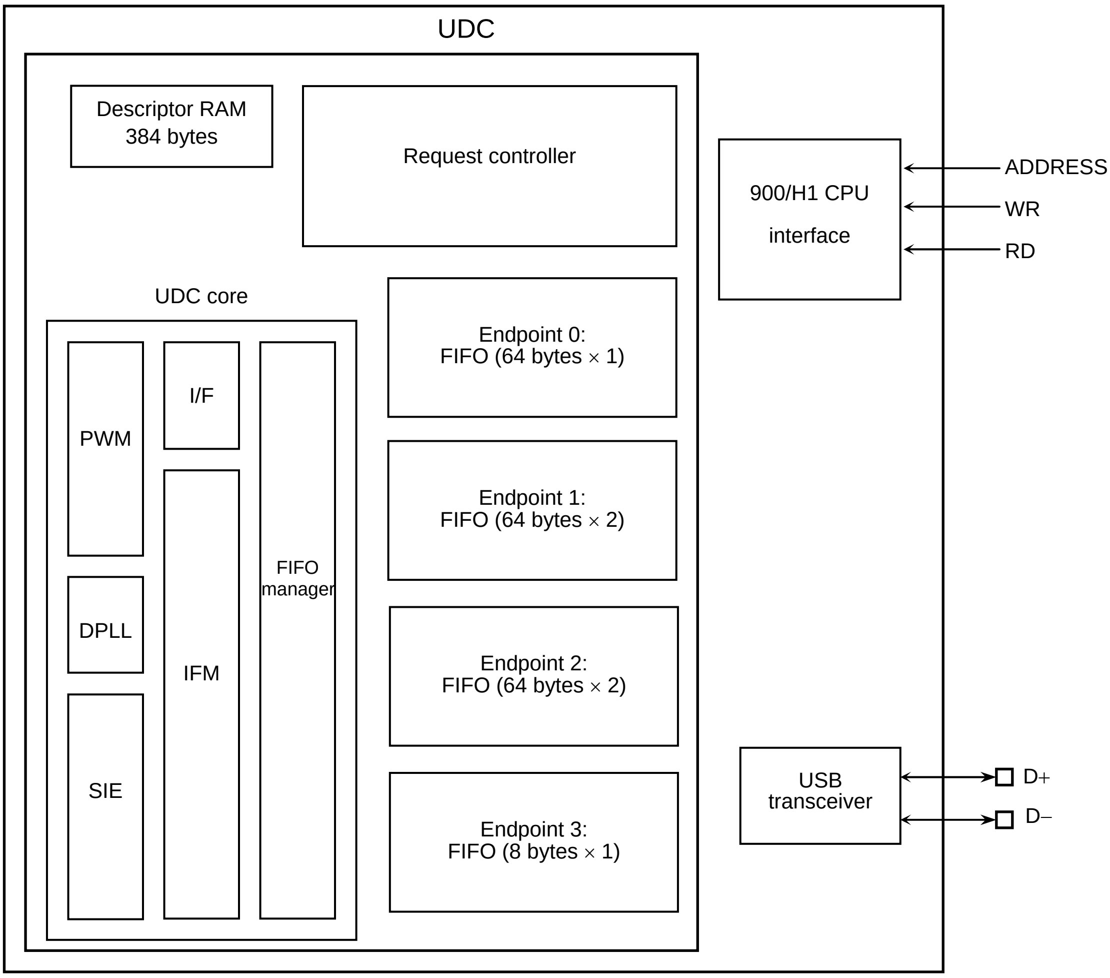
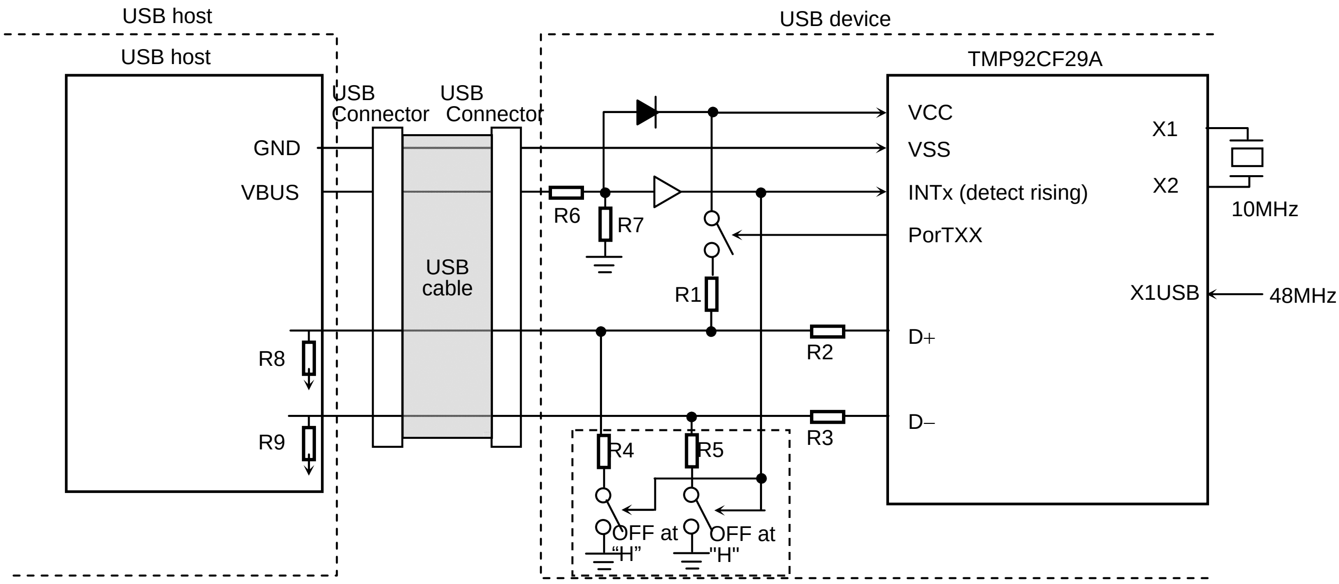


Figure 3.17.1 UDC Block Diagram

3.17.1.2 Example



The above setting is required If when using the TMP92CF29A's USB controller.

1) Pull-up of D^+ pin

- In the USB standard, in Full Speed connection, the D^+ pin must be set to pull-up. The ON/OFF control of this pull-up must be by S/W.

  Recommended value: R1=1.5kΩ

2) Add cascade resistor of D^+, D^-signal

- In the USB standard, for a D+ or D^- signal, a cascade resistor must be added to each signal. Recommended value : R2=27Ω, R3=27Ω

3) Flow current provision of the Connector connection and D^+ pin, D^- pin

- For the D^+ and D^- pin of the TMP92CF29A, the level must be fixed for flow current provision when not in use (when not connected to host). In this case, the connector detection signal is used to control the pull-down resistor which determines the level.

  Recommended value: R4=10kΩ, R5=10kΩ

- The example shows use of the connector detection method by using VBUS (5V voltage).

Note: Where waveform rise is solw, buffering of wabeform is recommended.

Recommended value: R6=60kΩ, R7=100kΩ

(VBUS current consumption when suspended is <500μA)

4) Connection of 10MHz oscillator to X1,X2, or input 48MHz clock to X1USB

- When using USB with a combination of 10MHz external oscillator and internal PLL, the number of external hub stages which can be used is restricted by the accuracy of the internal (Max 3 stages).
- If 5 stages connection is required for external hub, it is required that input 48MHz clock from X1USB pin (Restriction ≤±2500ppm.)

5) HOST side pull-down resistor

- In the USB standard, set pull-down D^+ pin and D^- signal at USB_HOST side.

  Recommended value: R8=15kΩ, R9=15kΩ

Note: The above connections and resistor values, etc, are given as examples only. Operation is not guaranteed. Please confirm the latest USB standar specifications and operations on your system.

3.17.2 900/H1 CPU I/F

The 900/H1 CPU I/F is a bridge between the 900/H1 CPU and the UDC. Its main functions are as follow.

- INTUSB (interrupt from UDC) generation
- A bridge for SFR
- USB clock control (48 MHz)

3.17.2.1 SFRs

The 900/H1 CPU I/F incorporates the following SFRs to control the UDC and USB transceiver.

- USB control
  USBCR1 (USB control register 1)
- USB interrupt control
  USBINTFR1 (USB interrupt flag register 1)
  USBINTFR2 (USB interrupt flag register 2)
  USBINTFR3 (USB interrupt flag register 3)
  USBINTFR4 (USB interrupt flag register 4)
  USBINTMR1 (USB interrupt mask register 1)
  USBINTMR2 (USB interrupt mask register 2)
  USBINTMR3 (USB interrupt mask register 3)
  USBINTMR4 (USB interrupt mask register 4)

Table 3.17.1 900/H1 CPU I/F SFR

Address	Read/Write	SFR Symbol
07F0H	R/W	USBINTFR1
07F1H	R/W	USBINTFR2
07F2H	R/W	USBINTFR3
07F3H	R/W	USBINTFR4
07F4H	R/W	USBINTMR1
07F5H	R/W	USBINTMR2
07F6H	R/W	USBINTMR3
07F7H	R/W	USBINTMR4
07F8H	R/W	USBCR1

3.17.2.2 USBCR1 Register

This register is used to set USB clock enables, transceiver enable etc.

		7	6	5	4	3	2	1	0
USBCR1	bit Symbol	TRNS_USE	WAKEUP					SPEED	USBCLKE
(07F8H)	Read/Write	R/W	R/W					R/W	R/W
	Reset State	0	0					1	0
	Function								

- TRNS_USE (Bit7)

  0: Disable USB transceiver
  1: Enable USB transceiver

  Always set to “1” on the application using USB.

- WAKEUP (Bit6)

  0: –
  1: Start remote-wakeup function

  When the remote-wakeup function is needed, first check Current_Config<REMOTE WAKEUP>.

  If <REMOTE WAKEUP> = “1” (meaning SUSPEND-status), write “1”, and “0” to <WAKEUP>. This will initiate the remote-wakeup function.

  If <REMOTE WAKEUP> = “0” or EP0, 1, 2, 3_STATUS<SUSPEND> = “0”, do not write “1” to <WAKEUP>.

- SPEED (Bit1)

  1: Full speed (12 MHz)
  0: Reserved

  This bit selects USB speed.

  Always set to “1”.

- USBCLKE (Bit0)

  0: Disable USB clock
  1: Enable USB clock

  This bit controls supply of USB clock.

  The USB clock (“f_{USB}”: 48MHz) is generated by an internal PLL. When the USB is started, write “1” to <USBCLKE> after confirming PLL lock up is terminated.

  Also, write “0” to <USBCLKE> before stopping the PLL.

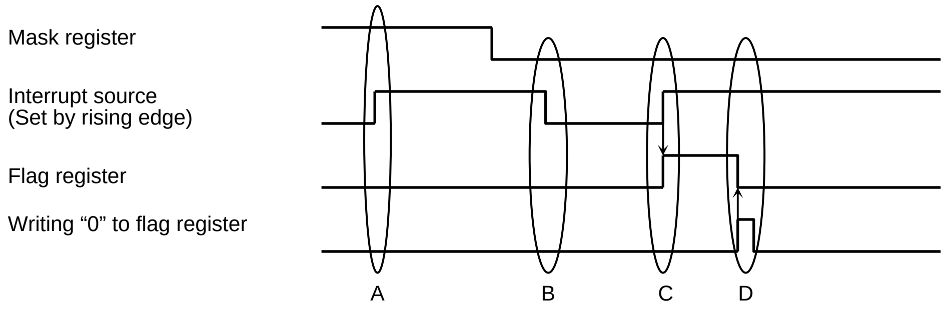
3.17.2.3 USBINTFRn, MRn Register

These SFRs control the INTUSB (only one interrupt to CPU) using the 23 interrupt sources output by the UDC.

The USBINTMRn are mask registers and the USBINTFRn are flag registers. In the INTUSB routine, execute operations according to generated interrupt source after checking USBINTFRn.

The common specification for all MASK and FLAG registers is shown below.

(Common specifications for all mask and flag registers.)



A: The flag register is not set because mask register = “1”.

B: The flag register is not set because interrupt souce changes “1” → “0”.

C: The flag register is set because mask register = “0” and interrupt souce changes “0” → “1”.

D: The flag register is reset to “0” by writing “0” to flag register.

Note 1: The “INTUSB generated number” and “bit number which is set to flag register” are not always equal. In the INTUSB interrupt routine, clear FLAG register (USBINTFRn) after checking it. The interrupt request flag, which occurrs between the INTUSB interrupt routine and flag register (USBINTFRn) read, is kept in the interrupt controller.
Therefore, after returning from the interrupt routine, the CPU jumps to INTUSB interrupt routine again. Software support is required to avoid ending in an error routine when none of the bits in the flag register (USBINTFRn) is set to “1”.

Note 2: Disable INTUSB (write 00H to INTEUSB register) before writing to USBINTMRn or USBINTFRn.

USBINTFR1 (07F0H) Prohibit to read-modify-write

	7	6	5	4	3	2	1	0
bit Symbol	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
Reset State	0	0	0	0	0	0		
Function	When read 0: Not generate interrupt 1: Generate interrupt			When write 0: Clear flag 1: –				

Note: The above interrupts can release Halt state from IDLE2 and IDLE1 mode. (STOP mode cannot be released)

*Those 6 interrupts of all 24 INTUSB sources can release Halt state from IDLE1 mode. Therefore, a low power dissipation system can be built. However, the method of use is limited as below.

Shift to IDLE1 mode :

Execute Halt instruction when the INT_SUS or INT_CLKSTOP flag is “1” (SUSPEND state)

Release from IDLE1 mode :

Release Halt state by INT_RESUME or INT_CLKON request (request of release SUSPEND)

Release Halt state by INT_URST_STR or INT_URST_request (request of RESET)

- INT_URST_STR (Bit7)

  This is the flag register for INT_URST_STR (“USB reset” start - interrupt).

  This is set to “1” when the UDC started to receive a “USB reset” signal from a USB-host.

  An application program has to initialize the whole UDC with this interrupt.

- INT_URST_END (Bit6)

  This is the flag register for INT_URST_END (“USB reset” end - interrupt).

  This is set to “1” when the UDC receives a “USB reset end” signal from a USB-host.

- INT_SUS (Bit5)

  This is the flag register for INT_SUS (suspend - interrupt).

  This is set to “1” when the USB changes to “suspend status”.

- INT_RESUME (Bit4)

  This is the flag register for INT_RESUME (resume - interrupt).

  This is set to “1” when the USB changes to “resume status”.

- INT_CLKSTOP (Bit3)

  This is the flag register for INT_CLKSTOP (enables stopping of the clock supply - interrupt).

  This is set to “1” after the USB changes to “suspend status”. Set USBCR1<USBCLKE> to “0” to stop the clock after detecting this interrupt if needed.

- INT_CLKON (Bit2)

  This is the flag register for INT_CLKON (enabled starting clock supply - interrupt).

  This is set to “1” after changing to “resume status” or when the UDC started to receive a “USB reset” signal from a USB-host. In case the clock has be stopped, set USBCR1<USBCLKE> to “1” to start the clock after detecting this interrupt if needed.

		7	6	5	4	3	2	1	0
USBINTFR2	bit Symbol	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
(07F1H)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Prohibit to read -modify -write	Reset State	0	0	0	0	0	0	0	0
	Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

		7	6	5	4	3	2	1	0
USBINTFR3	bit Symbol	EP3_FULL_A	EP3_Empty_A	EP3_FULL_B	EP3_Empty_B				
(07F2H)	Read/Write	R/W	R/W	R/W	R/W				
	Reset State	0	0	0	0				
Prohibit to read -modify -write	Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

- EPx_FULL_A/B:

  (When transmitting)
  This is set to "1" when CPU full write data to FIFO_A/B.
  (When receiving)
  This is set to "1" when UDC full receive data to FIFO_A/B.

- EPx_Empty_A/B:

  (When transmitting)
  This is set to "1" when FIFO become empty after transmission.
  (When receiving)
  This is set to "1" when FIFO becomes empty after CPU reads all data from FIFO.

Note: The EPx_FULL_A/B and EPx_Empty_A/B flags are not status flags. Therefore, check DATASET register to determine if the FIFO-status is needed.

		7	6	5	4	3	2	1	0
USBINTFR4 (07F3H) Prohibit to read -modify -write	bit Symbol	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	Reset State	0	0	0	0	0	0	0	
	Function	When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

Note: The above interrupt can release Halt state from IDLE2 mode. (IDLE1 and STOP mode cannot be released.)

- INT_SETUP (Bit7)

  This is the flag register for INT_SETUP (setup - interrupt).

  This is set to “1” when the UDC receives a request that S/W (software) control is needed from USB host.

  Using S/W (INT_SETUP routine), first read 8-byte device requests from the UDC and execute operation according to each request.

- INT_EP0 (Bit6)

  This is the flag register for INT_EP0 (received data of the data phase for Control transfer type - interrupt).

  This is set to “1” when the UDC receives data of the data phase for Control transfer type. If this interrupt occurs during Control write transfer, data reading from FIFO is needed. If this interrupt occurs during Control read transfer, transmission data writing to FIFO is needed.

  In some cases, the host may not assert “ACK” of the last packet in the data stage. In this case, this interrupt cannot be generated. Therefore, ignore this interrupt if it occurs after the last packet data has been written in the data stage because the transmission data number is specified by the host, or it depends on the capacity of the device.

- INT_STAS (Bit5)

  This is the flag register for INT_STAS (status stage end - interrupt).

  This is set to “1” when the status stage ends.

  If this interrupt is generated, it means that request ended normally.

  If this interrupt is not generated and INT_SETUP is generated, EP0_STATUS <STAGE_ERR> is set to “1”, and it means that request did not end normally.

- INT_STASN (Bit4)

  This is the flag register for INT_STASN (change host status stage - interrupt).

  This is set to "1" when the USB host changes to status stage at the Control read transfer. This interrupt is needed if data length is less than wLength (specified by the host).

- INT_EPxN (Bit3, 2, 1)

  This is the flag register for INT_EPxN (NAK acknowledge to the USB host - interrupt).

  This is set to "1" when the Endpoint1, 2 and 3 transmit NAK.

USBINTMR1
(07F4H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		
Reset State	1	1	1	1	1	1		
Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- MSK_URST_STR (Bit7)

  This is the mask register for USBINTFR1<INT_URST_STR>.
- MSK_URST_END (Bit6)

  This is the mask register for USBINTFR1<INT_URST_END>.
- MSK_SUS (Bit5)

  This is the mask register for USBINTFR1<INT_SUS>.
- MSK_RESUME (Bit4)

  This is the mask register for USBINTFR1<INT_RESUME>.
- MSK_CLKSTOP (Bit3)

  This is the mask register for USBINTFR1<INT_CLKSTOP>.
- MSK_CLKON (Bit2)

  This is the mask register for USBINTFR1<INT_CLKON>.

USBINTMR2 (07F5H)		7	6	5	4	3	2	1	0
	bit Symbol	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	1	1	1	1	1	1	1	1
	Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- EP1/2_MSK_FA/FB/EA/EB

  This is the mask register for USBINTFR2<EPx_FULL_A/B> or <EPx_Empty_A/B>.

USBINTMR3 (07F6H)		7	6	5	4	3	2	1	0
	bit Symbol	EP3_MSK_FA	EP3_MSK_EA						
	Read/Write	R/W	R/W						
	Reset State	1	1						
	Function	When read 0: not masked 1: masked When write 0: Clear flag 1: –							

- EP3_MSK_FA/FB/EA/EB:

  This is the mask register for USBINTFR3<EP3_FULL_A> or <EP3_Empty_A>.

USBINTMR4 (07F7H)

	7	6	5	4	3	2	1	0
bit Symbol	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset State	1	1	1	1	1	1	1	
Function	When read 0: Be not masked 1: Be masked When write 0: Clear flag 1: –							

- MSK_SETUP (Bit7)

  This is the mask register for USBINTFR4<INT_SETUP>.
- MSK_EP0 (Bit6)

  This is the mask register for USBINTFR4<INT_EP0>.
- MSK_STAS (Bit5)
- This is the mask register for USBINTFR4<INT_STAS>.
- MSK_STASN (Bit4)

  This is the mask register for USBINTFR4<INT_STASN>.
- MSK_EP1N (Bit3)

  This is the mask register for USBINTFR4<INT_EP1N>.
- MSK_EP2N (Bit2)

  This is the mask register for USBINTFR4<INT_EP2N>.
- MSK_EP3N (Bit1)

  This is the mask register for USBINTFR4<INT_EP3N>.

3.17.3 UDC CORE

3.17.3.1 SFRs

The UDC CORE has the following SFRs to control the UDC and USB transceiver.

a) FIFO

Endpoint 0 to 3 FIFO register

b) Device request

bmRequestType	register	bRequest	register
wValue_L	register	wValue_H	register
wIndex_L	register	wIndex_H	register
wLength_L	register	wLength_H	register

c) Status

Current_Config	register	USB_STATE	register
StandardRequest	register	Request	register
EPx_STATUS	register		

d) Setup

EPx_BCS	register	EPx_SINGLE	register
Standard Request Mode	register	Request Mode	register
Descriptor RAM	register	PortStatus	register

e) Control

EPx_MODE	register	EOP	register
COMMAND	register	INT_ Control	register
Setup Received	register	USBREADY	register

f) Others

ADDRESS	register	DATASET	register
EPx_SIZE_L_A	register	EPx_SIZE_H_A	register
EPx_SIZE_L_B	register	EPx_SIZE_H_B	register
FRAME_L	register	FRAME_H	register
USBBUFF TEST	register		

Table 3.17.2 UDC CORE SFRs (1/3)

Address	Read/Write	SFR Symbol
0500H	R/W	Descriptor RAM0
0501H	R/W	Descriptor RAM1
0502H	R/W	Descriptor RAM2
0503H	R/W	Descriptor RAM3
⋮	⋮	⋮
067DH	R/W	Descriptor RAM381
067EH	R/W	Descriptor RAM382
067FH	R/W	Descriptor RAM383
0780H	R/W	ENDPOINT0
0781H	R/W	ENDPOINT1
0782H	R/W	ENDPOINT2
0783H	R/W	ENDPOINT3
*0784H	R/W	ENDPOINT4
*0785H	R/W	ENDPOINT5
*0786H	R/W	ENDPOINT6
*0787H	R/W	ENDPOINT7
0788H	–	Reserved
0789H	R/W	EP1_MODE
078AH	R/W	EP2_MODE
078BH	R/W	EP3_MODE
*078CH	R/W	EP4_MODE
*078DH	R/W	EP5_MODE
*078EH	R/W	EP6_MODE
*078FH	R/W	EP7_MODE
0790H	R	EP0_STATUS
0791H	R	EP1_STATUS
0792H	R	EP2_STATUS
0793H	R	EP3_STATUS
*0794H	R	EP4_STATUS
*0795H	R	EP5_STATUS
*0796H	R	EP6_STATUS
*0797H	R	EP7_STATUS
0798H	R	EP0_SIZE_L_A
0799H	R	EP1_SIZE_L_A
079AH	R	EP2_SIZE_L_A
079BH	R	EP3_SIZE_L_A
*079CH	R	EP4_SIZE_L_A
*079DH	R	EP5_SIZE_L_A
*079EH	R	EP6_SIZE_L_A
*079FH	R	EP7_SIZE_L_A
07A1H	R	EP1_SIZE_L_B
07A2H	R	EP2_SIZE_L_B
07A3H	R	EP3_SIZE_L_B
*07A4H	R	EP4_SIZE_L_B
*07A5H	R	EP5_SIZE_L_B
*07A6H	R	EP6_SIZE_L_B
*07A7H	R	EP7_SIZE_L_B
07A8H	–	Reserved

Note: "*" is not used in the TMP92CF29A.

Table 3.17.3 UDC CORE SFRs (2/3)

Address	Read/Write	SFR Symbol
07A9H	R	EP1_SIZE_H_A
07AAH	R	EP2_SIZE_H_A
07ABH	R	EP3_SIZE_H_A
*07ACH	R	EP4_SIZE_H_A
*07ADH	R	EP5_SIZE_H_A
*07AEH	R	EP6_SIZE_H_A
*07AFH	R	EP7_SIZE_H_A
07B1H	R	EP1_SIZE_H_B
07B2H	R	EP2_SIZE_H_B
07B3H	R	EP3_SIZE_H_B
*07B4H	R	EP4_SIZE_H_B
*07B5H	R	EP5_SIZE_H_B
*07B6H	R	EP6_SIZE_H_B
*07B7H	R	EP7_SIZE_H_B
07C0H	R	bmRequestType
07C1H	R	bRequest
07C2H	R	wValue_L
07C3H	R	wValue_H
07C4H	R	wIndex_L
07C5H	R	wIndex_H
07C6H	R	wLength_L
07C7H	R	wLength_H
07C8H	W	Setup Received
07C9H	R	Current_Config
07CAH	R	Standard Request
07CBH	R	Request
07CCH	R	DATASET1
07CDH	R	DATASET2
07CEH	R	USB_STATE
07CFH	W	EOP
07D0H	W	COMMAND
07D1H	R/W	EPx_SINGLE1
*07D2H	R/W	EPx_SINGLE2
07D3H	R/W	EPx_BCS1
*07D4H	R/W	EPx_BCS2
07D5H	–	Reserved
07D6H	R/W	INT_Control
07D7H	–	Reserved
07D8H	R/W	Standard Request Mode
07D9H	R/W	Request Mode
07DAH	–	Reserved
07DBH	–	Reserved
07DCH	–	Reserved
07DDH	–	Reserved
07DEH	W	ID_CONTROL
07DFH	R	ID_STATE

Note: “*” is not used in the TMP92CF29A.

Table 3.17.4 UDC CORE SFRs (3/3)

Address	Read/Write	SFR Symbol
07E0H	R/W	Port_Status
07E1H	R	FRAME_L
07E2H	R	FRAME_H
07E3H	R	ADDRESS
07E4H	–	Reserved
07E5H	–	Reserved
07E6H	R/W	USBREADY
07E7H	–	Reserved
07E8H	W	Set Descriptor STALL

Note: “*” is not used in the TMP92CF29A.

3.17.3.2 EPx_FIFO Register (x: 0 to 3)

This register is prepared for each endpoint independently.

This is the window register from or to FIFO RAM.

In the auto bus enumeration, the request controller in UDC sets the mode, which is defined by the endpoint descriptor for each endpoint automatically. By this means, each endpoint is automatically set to each voluntary direction.

		7	6	5	4	3	2	1	0
Endpoint0 (0780H)	bit Symbol	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

		7	6	5	4	3	2	1	0
Endpoint1 (0781H)	bit Symbol	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

		7	6	5	4	3	2	1	0
Endpoint2 (0782H)	bit Symbol	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

		7	6	5	4	3	2	1	0
Endpoint3 (0783H)	bit Symbol	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Note: Read or write to these window registers using 1-byte load instructions only, since each register has only a 1-byte address. Do not use load instructions of 2 bytes or 4 bytes.

The device request that is received from the USB host is stored in the following 8-byte registers:

bmRequestType, bRequest, wValue_L, wValue_H, wIndex_L, wIndex_H, wLength_L and wLength_H. These are updated whenever a new SETUP token is received from the host.

When the UDC receives without error, INT_SETUP interrupt is asserted, meaning the new device request has been received.

There is also request which is operated automatically by the UDC, depending on the request received.

In that case, the UDC does not assert the INT_SETUP interrupt. Any request which the UDC is currently operating can be checked by reading STANDARD_REQUEST_FLAG and REQUEST_FLAG.

3.17.3.3 bmRequestType Register

This register shows the bmRequestType field of the device request.

		7	6	5	4	3	2	1	0
bmRequestType (07C0H)	bit Symbol	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	0	0	0	0

DIRECTION (Bit7)
0: from host to device
1: from device to host

REQ_TYPE [1:0] (Bit6 to bit5)
00: Standard
01: Class
10: Vendor
11: (Reserved)

RECIPIENT [4:0] (Bit4 to bit0)
00000: Device
00001: Interface
00010: Endpoint
00011: etc.
Others: (Reserved)

3.17.3.4 bRequest Register

This register shows the bRequest field of the device request.

		7	6	5	4	3	2	1	0
bRequest (07C1H)	bit Symbol	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	0	0	0	0

(Standard)
00000000: GET_STATUS
00000001: CLEAR_FEATURE
00000010: Reserved
00000011: SET_FEATURE
00000100: Reserved
00000101: SET_ADDRESS
00000110: GET_DESCRIPTOR
00000111: SET_DESCRIPTOR
00001000: GET_CONFIGURATION
00001001: SET_CONFIGURATION
00001010: GET_INTERFACE
00001011: SET_INTERFACE
00001100: SYNCH_FRAME

(Printer class)
00000000: GET_DEVICE_ID
00000001: GET_PORT_STATUS
00000010: SOFT_RESET

3.17.3.5 wValue Register

There are 2 registers; the wValue_L register and wValue_H register. wValue_L shows the lower-byte of the wValue field of the device request, and wValue_H register shows the upper byte.

wValue_L (07C2H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wValue_H (07C3H)

	7	6	5	4	3	2	1	0
bit Symbol	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.17.3.6 wIndex Register

There are 2 registers, the wIndex_L register and wIndex_H register. the wIndex_L register shows the lower byte of the wIndex field of the device request, and wIndex_H register shows the upper byte.

These are usually used to transfer index or offset.

wIndex_L (07C4H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wIndex_H (07C5H)

	7	6	5	4	3	2	1	0
bit Symbol	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.17.3.7 wLength Register

There are 2 registers, the wLength_L register and wLength_H register. The wLength_L register shows the lower-byte of the wLength field of the device request and wLength_H register shows the upper byte.

In the case of data phase, these registers show the byte number to transfer.

wLength_L (07C6H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

wLength_H (07C7H)

	7	6	5	4	3	2	1	0
bit Symbol	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

3.17.3.8 Setup Received Register

This register informs the UDC that an application program has recognized the INT_SETUP interrupt.

		7	6	5	4	3	2	1	0
SetupReceived	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
(07C8H)	Read/Write	W	W	W	W	W	W	W	W
	Reset State	0	0	0	0	0	0	0	0

If this register is accessed by an application program, the UDC disables access to the EP0's FIFO RAM because the UDC recognizes the device request has been received.

This is to protect data stored in the EP0 in the time between the completion of the previous device request and the recognition by the application program of the INT_SETUP interrupt relating to a new request f.

Therefore, write "00H" to this register when the device request in INT_SETUP routine is recognized.

Note : A recovery time of 2clock at 12MHz is needed after writing to this register in order to access EP0_FIFO.

3.17.3.9 Current_Config Register

This register shows the present value that is set by SET_CONFIGURATION and SET_INTERFACE.

		7	6	5	4	3	2	1	0
Current_Config	bit Symbol	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
(07C9H)	Read/Write	R		R	R	R	R	R	R
	Reset State	0		0	0	0	0	0	0

CONFIG[1:0] (Bit1 to bit0)

00: UNCONFIGURED	Set to UNCONFIGURED by the host.
01: CONFIGURED1	Set to CONFIGURED 1 by the host.
10: CONFIGURED2	Set to CONFIGURED 2 by the host.

INTERFACE[1:0] (Bit3 to bit2)

00: INTERFACE0	Set to INTERFACE 0 by the host.
01: INTERFACE1	Set to INTERFACE 1 by the host.
10: INTERFACE2	Set to INTERFACE 2 by the host.

ALTERNATE[1:0] (Bit5 to bit4)

00: ALTERNATE0	Set to ALTERNATE 0 by the host.
01: ALTERNATE1	Set to ALTERNATE 1 by the host.
10: ALTERNATE2	Set to ALTERNATE 2 by the host.

REMOTE WAKEUP (Bit7)

0: Disable	Disabled remote wakeup by the host.
1: Enable	Enabled remote wakeup by the host.

Note1: CONFIG, INTERFACE and ALTERNATE each support 3 kinds (0,1 and 2).

Note2: If each request is controlled by S/W, this register is not set.

3.17.3.10 Standard Request Register

This register shows the standard request currently being executed.

Any bit which is set to "1" shows a request currently being executed.

Standard Request (07CAH)		7	6	5	4	3	2	1	0
	bit Symbol	S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	0	0	0	0

S_INTERFACE (Bit 7) : SET_INTERFACE
G_INTERFACE (Bit 6) : GET_INTERFACE
S_CONFIG (Bit 5) : SET_CONFIGRATION
G_CONFIG (Bit 4) : GET_CONFIGRATION
G_DESCRIPT (Bit 3) : GET_DESCRIPTOR
S_FEATURE (Bit 2) : SET_FEATURE
C_FEATURE (Bit 1) : CLEAR_FEATURE
G_STATUS (Bit 0) : GET_STATUS

3.17.3.11 Request Register

This register shows the device request currently being executed.

Any bit which is set to "1" shows a request currently being executed.

Request (07CBH)		7	6	5	4	3	2	1	0
	bit Symbol		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
	Read/Write		R	R	R	R	R	R	R
	Reset State		0	0	0	0	0	0	0

SOFT_RESET (Bit 6) : SOFT_RESET
G_PORT_STS (Bit 5) : GET_PORT_STATUS
G_DEVICE_ID (Bit 4) : GET_DEVICE_ID
VENDOR (Bit 3) : Vendor class request
CLASS (Bit 2) : Class request
ExSTANDARD (Bit 1) : Auto Bus Enumeration not supported (SET_DESCRIPTOR, SYNCH_FRAME)
STANDARD (Bit 0) : Standard request

3.17.3.12 DATASET Register

This register shows whether FIFO contains data or not.

The application program can access this register to check whether FIFO contains data or not.

In the receiving status, when valid data transfer from the USB host has finished, the bit which corresponds to the corresponding endpoint is set to “1” and an interrupt generated. And, when the application reads the 1-packet data, this bit is cleared to “0”. In transmit status, when it has completed the 1-packet data transfer to FIFO, this bit is set to “1”. And when valid data is transferred to the USB host, this bit is cleared to “0” and an interrupt generated.

DATASET1 (07CCH)

	7	6	5	4	3	2	1	0
bit Symbol	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
Read/Write	R	R	R	R	R	R		R
Reset State	0	0	0	0	0	0		0

DATASET2 (07CDH)

	7	6	5	4	3	2	1	0
bit Symbol	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

Note: DATASET1<EP3_DSET_B>, DATASET2 registers are not used in the TMP92CF29A.

- Single packet mode
  (DATASET1: Bit0, bit2, bit4 and bit6 DATASET2: Bit0, bit2, bit4 and bit6)

These bits show whether FIFO of the corresponding endpoint has data or not.

In receive mode endpoint, if the corresponding endpoint bit is “1”, FIFO contains data to be read. Access EPx_SIZE register, determine the size of the data that should be read, and read data of this size. When this bit is “0”, there is no data to be read.

In transmit mode endpoint, if the corresponding endpoint bit is “0”, the CPU can transfer data under the FIFO payload. If this bit is “1”, because FIFO has transfer data waiting, transfer data to FIFO from UDC after the corresponding bit has been cleared to “0”. When a short-packet is transferred, access EOP register after writing transmission data to the corresponding endpoint.

- Dual packet mode
  (DATASET1: Bit3, bit5 and bit7 DATASET2: Bit1, bit3 bit5 and bit7)

These bits become effective in the dual packet mode. FIFO has 2-packets in this mode.

Each packet (packet-A and packet-B) has its own DATASET-bit.

Unlike as in the case above, in isochronous transfer, this shows the packet that can access the current frame. In this case, whether bit A or B is set to “1”, it is renewed according to the shifting frame.

Note1: In receive mode, if the endpoint bits corresponding to packet-A or paclet-B are "1", read the required packet-number data after checking EPx_SIZE<PKT_ACTIVE>.

Note2: In transmit mode, if both A and B bits are not "1", this means there is space in FIFO. So, write data of payload or less to FIFO. If the transmission is short-packet, write "0" to EOP<EPn_EOPB> after writing data to the FIFO. The maximum size that can be written to A or B packet is the same as the maximum payload size. If both A and B bits are "0", continuous writing of double maximum payload size is available.

Note3: In dual packet transmit mode, if both A and B packet are empty and EOP<EPn_EOPB> is written "0", the NULL-data is set to FIFO. In single mode, the NULL-data is also set to FIFO if the above operation is executed when packet-A contains no data.

Note4:No data is set in this register when NULL-packet (0Length-packet) is received.

3.17.3.13 EPx_STATUS Register (x: 0 to 7)

These registers are status registers for each endpoint. The <SUSPEND> is common to all endpoints.

			7	6	5	4	3	2	1	0
EP0_STATUS (0790H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP1_STATUS (0791H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP2_STATUS (0792H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP3_STATUS (0793H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP4_STATUS (0794H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP5_STATUS (0795H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP6_STATUS (0796H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0
EP7_STATUS (0797H)	bit Symbol			TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
	Read/Write			R	R	R	R	R	R	R
	Reset State			0	0	1	1	1	0	0

Note: EP4, 5, 6 and 7_STATUS registers are not used in the TMP92CF29A.

TOGGLE Bit (Bit6) — This bit shows status of toggle sequence bit.

0: TOGGLE Bit0
1: TOGGLE Bit1

SUSPEND (Bit5) — This bit shows status of UDC power management. In the SUSPEND status, access to UDC is limited. For details, refer to 3.17.9.

0: RESUME
1: SUSPEND

STATUS [2:0] (Bit4 to bit2)

These bits show status of UDC endpoint.

The status shows whether transfer is possible or not, and the result of the transfer. These depend on transfer type.

(For the Isochronous transfer type, refer to 3.17.9.)

000: READY	Receiving:	Device can be received. In endpoints 1 to 7, this register is initialized to “READY” by setting transfer type at SET_CONFIGURATION. In endpoint 0, this register is initialized to “READY” by detecting USB reset from the host. This is initialized to “READY” by terminating the status stage without error.
	Transmitting:	Basically, the same as with “Receiving”. But in transmitting, when data for transmission is set to FIFO and answer to token from host and transfer data to host collect and received ACK, status register does not change, and it remains “READY”. In this case, EPx_Empty_A or EPx_Empty_B interrupt terminates the transfer correctly.
001: DATAIN		UDC set to DATAIN and generates EPx_FULL_A or EPx_FULL_B interrupt when data is received from the host without error.
010: FULL		Refer to 3.17.8 (2) Details for the STATUS register.
011: TX_ERR		After transfer of data to IN token from host, UDC sets TX-ER to status register when “ACK” is not received from host. In this case, an interrupt is not generated. The hosts re-try IN token transfer.
100: RX_ERR		UDC sets RX_ERR to status register without transmitting “ACK” to host when an error (such as a CRC-error) is detected in data of received token. In this case, an interrupt is not generated. The hosts re-try and IN token transfer. In case of toggle error with normal data, UDC returns ACK and set RX_ERR of STATUS register.
101: BUSY		This status is used only for the control transfer type and it is set when a status-stage token is received from the host after a terminated data-stage. When status-stage can be finished, terminates correctly and returns to READY. This is not used in the Bulk and interrupts transfer type.
110: STALL		This status shows that the corresponding endpoint is in STALL status. In this status, STALL-handshake returns, except for SETUP-token. The control endpoint returns to READY from stall condition when SETUP-token is received. Other endpoints return to READY when initialization command of FIFO is received.
111: INVALID		This status shows that the corresponding endpoint is in UNCONFIGURED status. In this status, the UDC has no effect when a token is received from the host. On reset, all endpoints are set to INVALID status. Only endpoint 0 returns to READY on receiving USB-reset. Corresponding endpoints return to READY by according to configuration.

FIFO_DISABLE (Bit1)

0: FIFO enabled
1: FIFO disabled

This bit symbol shows FIFO status except for EP0.

If the FIFO is set to disabled, the UDC transmits NAK handshake for all transfers. Disabled or enabled status is set the COMMAND register. This bit is cleared to “0” when transfer type is changed.

STAGE_ERROR (Bit0)

0: SUCCESS
1: ERROR

This bit symbol shows that the status stage has not been terminated correctly. ERROR is set when a status stage is not terminated correctly and a new SETUP token is received.

When this bit is “1”, this bit is cleared to “0” by read EP0_STATUS register. This bit is not cleared even if normal control transfer or other transfer is executed after. To clear, read this bit. When software transaction is finished and UDC writes EOP register, UDC shifts to status register and waits termination of status stage. In this case, if software is needed to confirm that the status stage has been terminated correctly, when a new request flag is received, it is possible to confirm whether or not the last request has been terminated correctly. It can also be confirmed, when a new request flag is asserted, whether or not the last request has been cancelled before completion.

3.17.3.14 EPx_SIZE Register (x: 0 to 7)

These registers have the following functions.

a) In receive mode, showing the 1-packet data number which has been received correctly.

b) In the transmit mode, showing payload size. Showing length value when short packet is transferred.
It is not necessary to read this register when it is transmitting.

c) Showing dual packet mode and currently effective packet.

Each endpoint has an H (High)-register that shows upper bit 9 to bit7 of data size and an L (Low) register which shows lower bit 6 to bit0 and control bit of FIFO.

Each H/L register also has 2-set for dual-packet mode.

Register		7	6	5	4	3	2	1	0
EP0_SIZE_L_A (0798H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP1_SIZE_L_A (0799H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_L_A (079AH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_L_A (079BH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_L_A (079CH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_L_A (079DH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_L_A (079EH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_L_A (079FH)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	1	0	0	0	1	0	0	0

Note: EP4,5,6,7_SIZE_L_A registers are not used in the TMP92CF29A.

		7	6	5	4	3	2	1	0
EP1_SIZE_L_B (07A1H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_L_B (07A2H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_L_B (07A3H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_L_B (07A4H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_L_B (07A5H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_L_B (07A6H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_L_B (07A7H)	bit Symbol	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
	Read/Write	R	R	R	R	R	R	R	R
	Reset State	0	0	0	0	1	0	0	0

Note EP3,4,5,6,7_SIZE_L_B registers are not used in the TMP92CF29A.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_A (07A9H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_H_A (07AAH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_H_A (07ABH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_H_A (07ACH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_H_A (07ADH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_H_A (07AEH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_H_A (07AFH)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

Note EP4,5,6,7_SIZE_H_A registers are not used in the TMP92CF29A.

		7	6	5	4	3	2	1	0
EP1_SIZE_H_B (07B1H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP2_SIZE_H_B (07B2H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP3_SIZE_H_B (07B3H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP4_SIZE_H_B (07B4H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP5_SIZE_H_B (07B4H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP6_SIZE_H_B (07B6H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0
		7	6	5	4	3	2	1	0
EP7_SIZE_H_B (07B7H)	bit Symbol						DATASIZE9	DATASIZE8	DATASIZE7
	Read/Write						R	R	R
	Reset State						0	0	0

Note EP3,4,5,6,7_SIZE_H_B registers are not used in the TMP92CF29A.

DATASIZE[9:7] (H register: Bit2 to bit0)

DATASIZE[6:0] (L register: Bit6 to bit0)

In receiving, the data number of the 1 packet received from the host is shown. This is renewed when data from the host is received with no error.

By setting EPx_MODE register, these bits are initialized to MAX pay load size in bulk/interrupt transfer, and "0" in isochronous transfer.

PKT_ACTIVE (L register: Bit7)

1: OUT_ENABLE

0: OUT_DISABLE

When dual-packet mode is selected, this bit show the packet that can be accessed. In this case, the UDC accesses packets that divide FIFO (Packet A and Packet B) mutually. When FIFO in UDC is accessed by CPU, refer to this bit. If receiving endpoint, start reading from that packet that this bit is "1". In single-packet mode, this bit has no effect because packet-A is always used.

3.17.3.15 FRAME Register

This register shows the frame number which is issued with SOF token from the host and is used for Isochronous transfer type.

Each HIGH and LOW register shows upper and lower bits.

FRAME_L (07E1H)

	7	6	5	4	3	2	1	0
bit Symbol	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
Read/Write	R	R	R	R	R	R	R	R
Reset State	0	0	0	0	0	0	0	0

FRAME_H (07E2H)

	7	6	5	4	3	2	1	0
bit Symbol	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
Read/Write	R	R	R	R		R	R	R
Reset State	0	0	0	0		0	1	0

T[10:7] (H register: Bit7 to bit4)
T[6:0] (L register: Bit6 to bit0)

These bits are renewed when SOF-token is received. They also shows the frame-number.

CREATE (H register: Bit2)
0: DISABLE
1: ENABLE

These bits show whether the function that generates SOF automatically from the UDC is enabled or not. This is used in case of error in receiving SOF token.

This function is set by accessing COMMAND register.

On reset, this bit is initialized to "0".

FRAME STS[1:0]
(H register: Bit1 and bit0)
0: BEFORE
1: VALID
2: LOST

These bits show the status whether a frame number that is shown in the FRAME register is correct or not. At the LOST status, a correct frame number is undefined.

If this register is "VALID", the number that is shown to the FRAME register is correct.

If this register is "BEFORE", during SOF auto generation, BEFORE condition shows it from USB host controller inside that from SOF generation time to reception of SOF token. Correct frame-number value is the value that is selected from FRAME register value.

3.17.3.16 ADDRESS Register

This register shows the device address which is specified by the host in bus enumeration.

By reading this register, the present address can be confirmed.

ADDRESS (07E3H)

	7	6	5	4	3	2	1	0
bit Symbol		A6	A5	A4	A3	A2	A1	A0
Read/Write		R	R	R	R	R	R	R
Reset State		0	0	0	0	0	0	0

ADDRESS [6:0] (Bit6 to bit0)

The UDC compares this registers and address in all packet ID, and UDC judges whether it is an effective transaction or not.

This is initialized to "00H" by USB reset.

3.17.3.17 EOP Register

This register is used when a control transfer type dataphase terminates or when a short packet is transmitting bulk-IN or interrupt-IN.

EOP (07CFH)		7	6	5	4	3	2	1	0
	bit Symbol	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
	Read/Write	W	W	W	W	W	W	W	W
	Reset State	1	1	1	1	1	1	1	1

Note1: EOP<EP7_EOPB, EP6_EOPB, EP5_EOPB, EP4_EOPB> registers are not used in the TMP92CF29A.

Note2: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

In a control transfer type dataphase, write “0” to <EP0_EOPB> when all transmission data is written to the FIFO, or read all receiving data from the FIFO. The UDC terminates its status stage on this signal.

When a short packet is transmitted by using bulk-IN or interrupt-IN endpoint, use this to terminate writing of transmission data. In this case, write “0” to <EP0_EOPB> of writing endpoint. Write “1” to other bits.

3.17.3.18 Port Status Register

This register is used when a request of printer class request is received.

In the case of a GET_PORT_STATUS request, the UDC operates automatically using this data.

Port Status (07E0H)		7	6	5	4	3	2	1	0
	bit Symbol	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
	Read/Write	W	W	W	W	W	W	W	W
	Reset State	0	0	0	1	1	0	0	0

Note: The TMP92CF29A doed not use this register since not support printer-class.

The data should be written before receiving request.

Write “0” to the <Reserved> bit of this register. This register is initialized to “18H” on reset.

3.17.3.19 Standard Request Mode Register

This register sets the answer for Standard Request either answering automatically in hardware, or by control through software. Each bit represents a kind of request.

When the relevant bit in this register is set to “0”, the answer is executed automatically by hardware. When the relevant bit in this register is set to “1”, the answer is controlled by software. If a request is received during hardware control, the interrupt signal (INT_SETUP, INT_EP0, INT_STAS, INT_STAN) is set to disable. If a request is received during software control, the interrupt signal is asserted, and it is controlled by software.

Standard Request Mode (07D8H)		7	6	5	4	3	2	1	0
	bit Symbol	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	0	0	0	0	0	0	0	0

S_Intetface	(Bit 7) : SET_INTERFACE
G_Interface	(Bit 6) : GET_INTERFACE
S_Config	(Bit 5) : SET_CONFIGRATION
G_Config	(Bit 4) : GET_CONFIGRATION
G_Descript	(Bit 3) : GET_DESCRIPTOR
S_Feature	(Bit 2) : SET_FEATURE
C_Feature	(Bit 1) : CLEAR_FEATURE
G_Status	(Bit 0) : GET_STATUS

3.17.3.20 Request Mode Register

This register sets the answer for Class Request either automatically in hardware or by control through software. Each bit represents a kind of request.

When relevant bit in this register is set to "0", the answer is executed automatically by hardware. When relevant bit in this register is set to "1", the answer is controlled by software. If request is received during hardware control, interrupt signal (INT_SETUP, INT_EP0, INT_STAS, INT_STATUSN) is set to disable. If a request is received during software control, the interrupt signal is asserted, and it is controlled by software.

Request Mode (07D9H)		7	6	5	4	3	2	1	0
	bit Symbol		Soft_Reset	G_Port_Sts	G_DeviceId				
	Read/Write		R/W	R/W	R/W				
	Reset State		0	0	0				

Note: the TMP92CF29A doed not use this register since it does not support printer-class.

-	(Bit 7)	: Reserved
Soft_Reset	(Bit 6)	: SOFT_RESET
G_Port_Sts	(Bit 5)	: GET_PORT_STATUS
G_Config	(Bit 4)	: GET_DEVICE_ID
-	(Bit 3 to 0)	: Reserved

Note1: SET_ADDRESS request is supported only by auto-answer .

Note2: SET_DESCRIPTOR and SYNCH_FRAME are controlled only by software .

Note3: Vendor Request and Class Request (Printer Class and so on) are controlled only by software.

Note4: INT_SETUP, EP0, STAS and STASN interrupts assert only when it is software-control.

3.17.3.21 COMMAND Register

This register sets COMMAND at each endpoint. This register can be set to select of endpoint in bit6 to bit4 and kind of COMMAND in bit3 to bit0.

COMMAND for endpoint that is supported is ignored.

			7	6	5	4	3	2	1	0
COMMAND (07D0H)	bit Symbol			EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
	Read/Write			W	W	W	W	W	W	W
	Reset State			0	0	0	0	0	0	0

Note: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

EP [2:0] (Bit6 to bit4)

000: Select endpoint 0

001: Select endpoint 1

010: Select endpoint 2

011: Select endpoint 3

COMMAND [3:0] (Bit3 to bit0)

0000: Reserved

0001: Reserved

0010: SET_DATA0 This COMMAND clear toggle sequence bit of corresponding endpoint (EP0 to EP3).

If this COMMAND is input, it sets toggle sequence bit of the corresponding endpoint to “0”. Data toggle for transfer is renewed automatically by UDC. However, this COMMAND execution is required if setting toggle sequence bit of endpoint to “0”.If control transfer type and Isochronous transfer type, execution of this COMMAND is not required because of hardware control.

0011: RESET This COMMAND resets the corresponding endpoint (EP0 to EP3).

If this COMMAND is input, the corresponding endpoint is initialized. CLEAR_FEATURE request stalls endpoint. When this stall is cleared, execute this COMMAND. (This command does not affect transfer mode.)

This command initializes the following.

・Clear toggle sequence bit of corresponding endpoint.

・Clear STALL of corresponding endpoint.

・Set to FIFO_ENABLE condition.

・Clear the data in FIFO

0100: STALL This COMMAND sets corresponding endpoint to STALL (EP0 to EP3).

If STALL handshake must be return as answer for device request, execute this command.

0101: INVALID This COMMAND sets condition to prohibition of use corresponding endpoint (EP1 to EP3).

If UDC detects USB_RESET signal from USB host, it sets all endpoints (except endpoint 0) to prohibition using it automatically. If Config and Interface are changed by device request, set endpoint that is not used to prohibit use.

0110: CREATE_SOF This COMMAND sets quasi-SOF generation function to enable (EP0).

Default is set to disable, it must be used for Isochronous transfer.

0111: FIFO_DISABLE This COMMAND sets FIFO of corresponding endpoint to disable (EP1 to EP3).

If this command is set from external, all of transfers except for toggle error for corresponding endpoint return NAK. When it is set externally while receiving packet, this becomes valid from next token. This command does not affect the packet that is transferring.

1000: FIFO_ENABLE — This COMMAND sets FIFO of corresponding endpoint to enable (EP1 to EP3).

If FIFO is set to disable by FIFO_DISABLE COMMAND, this command is used for release of disable condition. If set while receiving packet, this becomes valid from next token. If USB_RESET is detected from host and RESET COMMAND execute and transfer mode is set by using SET_CONFIG and SET_INTERFACE request, the corresponding endpoint enters FIFO_ENABLE condition.

1001: INIT_DESCRIPTOR — This COMMAND is used if descriptor RAM is rewritten during system operation (EP0).

If UDC detects USB_RESET from host controller, it reads content of descriptor RAM automatically, and it performs relevant settings.

If descriptor RAM is changed during system operation, it must read setting again. Therefore, execute this command. When connected to USB host, this function starts reading automatically. Therefore, in this case, it is not necessary to execute this command.

1010: FIFO_CLEAR — This COMMAND initializes FIFO of corresponding endpoint (EP1 to EP3).

However, EPx_STATUS<TOGGLE> is not initialized.

If resetting by software, execute this COMMAND.

This command Initializes the following item.

・Clear STALL of relevant endpoint.

・Set to FIFO_ENABLE condition.

・Clear the data in FIFO

1011: STAL_CLEAR — This COMMAND clear STALL of corresponding endpoint (EP1 to EP3).

If clearing only STALL of endpoint, execute this COMMAND.

3.17.3.22 INT_Control Register

INT_STASN interrupt is disabled and enabled by the value that is written to this register.

This is initialized to disable by external reset. When setup packet is received, it becomes disabled.

INT_Control (07D6H)		7	6	5	4	3	2	1	0
	bit Symbol								Status_nak
	Read/Write								R/W
	Reset State								0

In control read transfer, if the host terminates a dataphase with small data length (smaller than the data length that is specified by the host as wLength), the device side and stage management cannot be synchronized. Therefore, INT_STASN interrupt signals this shift to status stage. If needed, set to "1" after receiving setup packet.

STATUS_NAK (Bit0)

0: INT_STATSN interrupt disable
1: INT_STATSN interrupt enable

3.17.3.23 USB STATE Register

This register shows the current device state for connection with USB host.

USB STATE (07CEH)		7	6	5	4	3	2	1	0
	bit Symbol						Configured	Addressed	Default
	Read/Write						R/W	R	R
	Reset State						0	0	1

Note: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

Inside the UDC, the answer for each Device Request is managed by referring to these bits (Configured, Addressed and Default). If transaction for SET_CONFIG request is executed by using software, write the present state to this register. If host appointconfig is 0, this becomes Unconfigured, and it is necessary to return to Addressed state. Therefore, if host appoint config is 0, write "0" to bit2.

When Configured bit (Bit2) is written "0", Addressed bit (bit 1) is set automatically by hardware. When host appoint config value that supported by device, device must execute mode setting for each endpoint by using the value that is appointed by endpoint-descriptor in the config-descriptor. After finish mode setting, set Configured bit (Bit2) to "1" before accessing EOP register. When this bit is set to "1", Addressed bit (Bit1) is set to "0" automatically.

Bit2 to bit0

000: Default
010: Addressed
100: Configured

3.17.3.24 EPx_MODE Register (x: 1 to 3)

This register sets transfer mode of endpoint (EP1 to EP3).

If SET_CONFIG and SET_INTERFACE processing is set to software control, this control must use appointed config or interface. Access this register to set mode.

		7	6	5	4	3	2	1	0
EP1_MODE (0789H)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP2_MODE (078AH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0
		7	6	5	4	3	2	1	0
EP3_MODE (078BH)	bit Symbol			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
	Read/Write			R/W	R/W	R/W	R/W	R/W	R/W
	Reset State			0	0	0	0	0	0

There is a limitation to the timing that can be written.

If transaction for SET_CONFIG and SET_INTERFACE processing is set to software control, after INT_SETUP interrupt is received, finish writing before accessing EOP register. This register prohibits writing when it is other timing, and it is ignored.

Note1: When writing to this register, a recovery time of 5clocks at 12MHz is needed. After writing this register, insert dummy instruction of 420 ns or longer.

Note2: When writing to this register, endpoint is initialized same as RESET of COMMAND register.

DIRECTION (Bit0)

0: OUT Direction from host to device
1: IN Direction from device to host

MODE [1:0] (Bit2 and bit1)

00: Control transfer type
01: Isochronous transfer type
10: Bulk transfer type or interrupt transfer type
11: Interrupt (No toggle)

PAYLOAD [2:0] (Bit3, bit4 and bit5)

000: 8 bytes
001: 16 bytes
010: 32 bytes
011: 64 bytes
0100:128 bytes
0101:256 bytes
0110:512 bytes
0111:1023 bytes (Note1, 2)

Note1: Max packet size of Isochronous transfer type is 1023 bytes.

Note2: If wMaxPacketSize of descriptor has been set to other than 8, 16, ..., 1023, Payload more than descriptor value is set by auto-answer of Set_Configration and Set_Interface.

Others (Bit6 and bit7) Reserved

3.17.3.25 EPx_SINGLE Register

This register sets mode of FIFO in each endpoint (SINGLE/DUAL).

		7	6	5	4	3	2	1	0
EPx_SINGLE1 (07D1H)	bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
	Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
	Reset State	0	0	0		0	0	0	

Note: Endpoint 3 support only SINGLE mode in the TMP92CF29A.

Bit number
- 0: No use
- 1: EP1_SINGLE
- 2: EP2_SINGLE
- 3: EP3_SINGLE
- 4: No use
- 5: EP1_SELECT
- 6: EP2_SELECT
- 7: EP3_SELECT

When EPx_SELECT bit is "1", EPx_SINGLE bit becomes valid in the following content.

0: DUAL mode 1: SINGLE mode

If setting content of EPx_SINGLE bit to valid, set EPx_SELECT bit to "1".

0: Invalid 1: Valid

3.17.3.26 EPx_BCS Register

This register sets mode of access to FIFO in each endpoint.

		7	6	5	4	3	2	1	0
EPx_BCS1 (07D3H)	bit Symbol	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
	Read/Write	R/W	R/W	R/W		R/W	R/W	R/W	
	Reset State	0	0	0		0	0	0	

Bit number
- 0: No use
- 1: EP1_BCS
- 2: EP2_BCS
- 3: EP3_BCS
- 4: No use
- 5: EP1_SELECT
- 6: EP2_SELECT
- 7: EP3_SELECT

Always write "1" to EPx_BCS bit regardless of whether endpoint is used or not.

0: Reserved 1: CPU access

If setting content of EPx_BCS bit to valid, set EPx_SELECT bit to "1".

0: Invalid 1: Valid

3.17.3.27 USBREADY Register

This register informs finishing writing data to descriptor RAM on UDC.

After assigned data to descriptor RAM, write “0” to bit0.

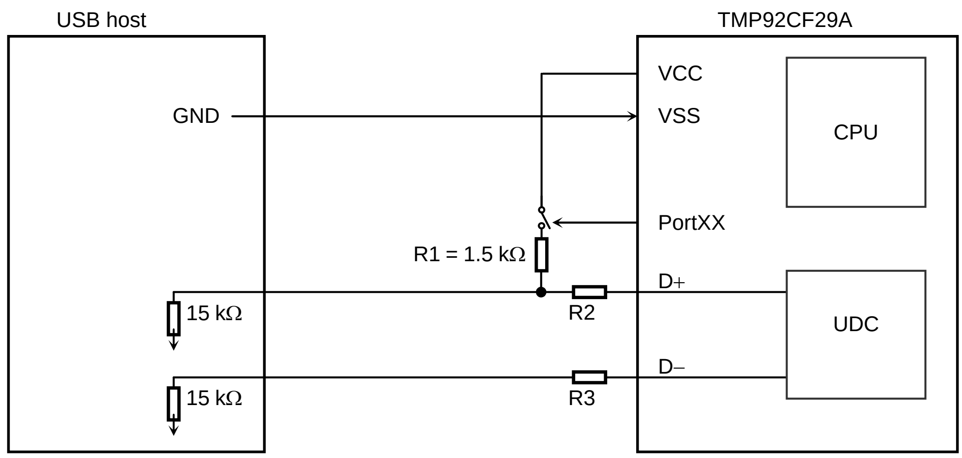
		7	6	5	4	3	2	1	0
USBREADY (07E6H)	bit Symbol								USBREADY
	Read/Write								R/W
	Reset State								0

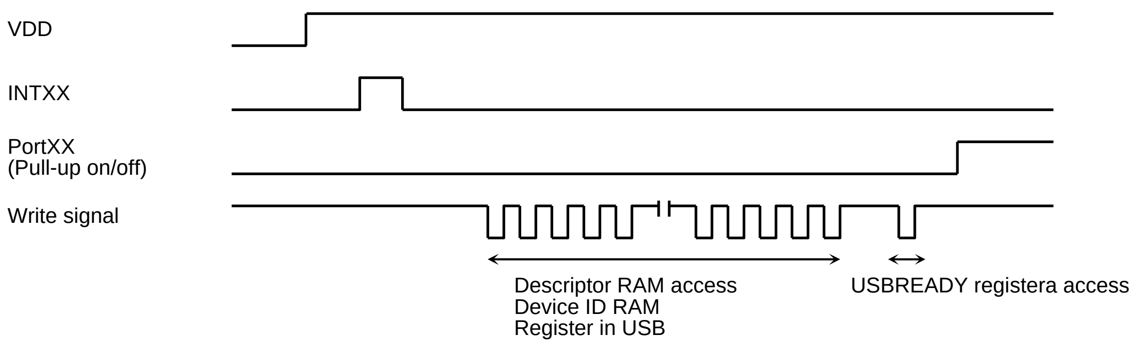
USBREADY (Bit0)

0: Writing to descriptor RAM has finished.

1: Writing to descriptor RAM is enabled.

(However, writing to descriptor RAM is prohibited when connected to host.)





Detect level of VDD signal from USB cable, and execute initialize sequence. In this case, UDC disable detecting USB_RESET signal until USBREADY register is written “0” after release of USB_RESET.

If the pull-up resistor on D+ signal is controlled by control signal, when pull-up resistor is connected to host in OFF condition, this condition is equivalent condition with USB_RESET signal by pull-down resistor on the host side. Therefore UDC is not detected in USB_RESET until “0” is written to USBREADY register

Note1: External pull-up resistor and control switch are needed with the TMP92CF29A.

Note2: The above setting is an example for when communication. A specific circuit is required to prevent cullent flow at connector detection , no-use, and no connection.

3.17.3.28 Set Descriptor STALL Register

This register sets whether returns STALL automatically in data stage or status stage for Set Descriptor Request.

Set Descriptor STALL (07E8H)		7	6	5	4	3	2	1	0
	bit Symbol								S_D_STALL
	Read/Write								W
	Reset State								0

Bit0: S_D_STALL

0: Software control (Default)

1: Automatically STALL

3.17.3.29 Descriptor RAM Register

This register is used for store descriptor to RAM. The size of the descriptor is 384 bytes. However, when storing descriptor, write according to descriptor RAM structure sample.

Descriptor RAM (0500H) ~ (067FH)		7	6	5	4	3	2	1	0
	bit Symbol	D7	D6	D5	D4	D3	D2	D1	D0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset State	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

Read/Write timing is only possible before detection of USB_RESET or during processing of SET_DESCRIPTOR request.

SET_DESCRIPTOR request processes from INT_SETUP assert until access of EOP register.

If there is rewriting request of descriptor in SET_DESCRIPTOR, process the request in the following sequence.

1) Read every packet of the descriptor that is transferred by SET_DESCRIPTOR requests every packet.
2) When reading descriptor number of last packet finished, write all descriptors to RAM for descriptor.
3) When writing is completed, execute INIT_DESCRIPTOR of COMMAND register.
4) When all the process is completed, access EOP register, and finish status stage.
5) When INT_STAS is received, it shows normal finish of status stage.

If USB_RESET is detected, it starts reading automatically. Therefore, when it connect to the host, executing INIT_DESCRIPTOR command is not necessary.

3.17.4 Descriptor RAM

This area stores the descriptor that is defined in USB. Device, Config, Interface, Endpoint and String descriptor must set to RAM using the following format.

Device descriptor	18 bytes
Config 1 descriptor (Interfaces, endpoints)	255 bytes or less
Config 2 descriptor (Interfaces, ENDPOINT)	255 bytes or less
String0 length	1 byte
String1 length	1 byte
String2 length	1 byte
String3 length	1 byte
String0 descriptor	63 bytes or less
String1 descriptor	63 bytes or less
String2 descriptor	63 bytes or less
String3 descriptor	63 bytes or less

Note 1: If String Descriptor is supported, set StringxLength area to size0. No support String Dedcriptor is returned STALL.

Note 2: Config Descriptior refers to descriptor sample.

Note 3: Sequencer in UDC determines Config number, Interface number and Endpoint number. Therefore, if supporting Endpoint number is small, assign address according to priority.

Note 4: This function become effective only in case of store descriptor as RAM.

Note 5: RAM size is total 384 bytes.

Note 6: Possible timing in RD/WR of descriptor RAM is only before detection of USB_RESET and processing of SET_DESCRIPTOR request. (Prohibit access other than this timing.)
Writing must finish before connection to USB host and processing of SET_DESCRIPTOR request.
SET_DESCRIPTOR request processes from INT_SETUP assert until access of EOP register.

Note 7: The class descriptor, and the vender descriptors, etc except a standard descriptor cannnot be supported by an auto bus enumeration.

Descriptor RAM setting example:

Address	Data	Description	Description
Device Descriptor			
500H	12H	bLength	
501H	01H	bDescriptorType	Device Descriptor
502H	00H	bcdUSB (L)	USB Spec 1.00
503H	01H	bcdUSB (H)	IFC’s specify own
504H	00H	bDeviceClass	
505H	00H	bDeviceSubClass	
506H	00H	bDeviceProtocol	
507H	08H	bMaxPacketSize0	
508H	6CH	bVendor (L)	Toshiba
509H	04H	bVendor (H)	
50AH	01H	IdProduct (L)	
50BH	10H	IdProduct (H)	
50CH	00H	bcdDevice (L)	Release 1.00
50DH	01H	bcdDevice (H)	
50EH	00H	bManufacture	
50FH	00H	IProduct	
510H	00H	bSerialNumber	
511H	01H	bNumConfiguration	
Config1 Descriptor			
512H	09H	BLength	
513H	02H	bDescriptorType	Config Descriptor
514H	4EH	wtotalLength (L)	78 bytes
515H	00H	wtotalLength (H)	
516H	01H	bNumInterfaces	
517H	01H	bConfigurationValue	
518H	00H	iConfiguration	
519H	A0H	bmAttributes	Bus-powered-remote wakeup
51AH	31H	MaxPower	98 mA
Interface0 Descriptor AlternateSetting0			
51BH	09H	bLength	
51CH	04H	bDescriptorType	Interface Descriptor
51DH	00H	bInterfaceNumber	
51EH	00H	bAlternateSetting	AlternateSetting0
51FH	01H	bNumEndpoint	
520H	07H	bInterfaceClass	
521H	01H	bInterfaceSubClass	
522H	01H	bInterfaceProtocol	
523H	00H	iInterface	
Endpoint1 Descriptor			
524H	07H	bLength	
525H	05H	bDescriptorType	Endpoint Descriptor
526H	01H	bEndpointAddress	OUT
527H	02H	bmAttributes	BULK
528H	40H	wMaxPacketSize (L)	64 bytes
529H	00H	wMaxPacketSize (H)	
52AH	00H	bInterval	

Address	Data	Description	Description
Interface0 Descriptor AlternateSetting1			
52BH	09H	bLength	
52CH	04H	bDescriptorType	Interface Descriptor
52DH	00H	bInterfaceNumber	
52EH	01H	bAlternateSetting	AlternateSetting1
52FH	02H	bNumEndpoints	
530H	07H	bInterfaceClass	
531H	01H	bInterfaceSubClass	
532H	02H	bInterfaceProtocol	
533H	00H	iInterface	
Endoint1 Descriptor			
534H	07H	bLength	
535H	05H	bDescriptorType	Endpoint Descriptor
536H	01H	bEndpointAddress	OUT
537H	02H	bmAttributes	BULK
538H	40H	wMaxPacketSize (L)	64 bytes
539H	00H	wMaxPacketSize (H)	
53AH	00H	bInterval	
Endpoint2 Descriptor			
53BH	07H	bLength	
53CH	05H	bDescriptorType	Endpoint Descriptor
53DH	82H	bEndpointAddress	IN
53EH	02H	bmAttributes	BULK
53FH	40H	wMaxPacketSize (L)	64 bytes
540H	00H	wMaxPacketSize (H)	
541H	00H	bInterval	
Interface0 Descriptor AlternateSetting2			
542H	09H	bLength	
543H	04H	bDescriptorType	Interface Descriptor
544H	00H	bInterfaceNumber	
545H	02H	bAlternateSetting	AlternateSetting2
546H	03H	bNumEndpoints	
547H	FFH	bInterfaceClass	
548H	00H	bInterfaceSubClass	
549H	FFH	bInterfaceProtocol	
54AH	00H	iInterface	
Endpoint1 Descriptor			
54BH	07H	bLength	
54CH	05H	bDescriptorType	Endpoint Descriptor
54DH	01H	bEndpointAddress	OUT
54EH	02H	bmAttributes	BULK
54FH	40H	wMaxPacketSize (L)	64 bytes
550H	00H	wMaxPacketSize (H)	
551H	00H	bInterval	
Endpoint2 Descriptor			
552H	07H	bLength	
553H	05H	bDescriptorType	Endpoint Descriptor
554H	82H	bEndpointAddress	IN
555H	02H	bmAttributes	BULK
556H	40H	wMaxPacketSize (L)	64 bytes
557H	00H	wMaxPacketSize (H)	
558H	00H	bInterval	

Address	DATA	Description	Description
Endpoint3 Descriptor			
559H	07H	bLength	
55AH	05H	bDescriptorType	Endpoint Descriptor
55BH	83H	bEndpointAddress	IN
55CH	03H	bmAttributes	Interrupt
55DH	08H	wMaxPacketSize (L)	8 bytes
55EH	00H	wMaxPacketSize (H)	
55FH	01H	bInterval	1 ms
String Descriptor Length Setup Area			
560H	04H	bLength	Length of String Descriptor0
561H	10H	bLength	Length of String Descriptor1
562H	00H	bLength	Length of String Descriptor2
563H	00H	bLength	Length of String Descriptor3
String Descriptor0			
564H	04H	bLength	
565H	03H	bDescriptorType	String Descriptor
566H	09H	bString	Language ID 0x0409
567H	04H	bString	
String Descriptor1			
568H	10H	bLength	
569H	03H	bDescriptorType	String Descriptor
56AH	00H	bString	(Toshiba)
56BH	54H	bString	T
56CH	00H	bString	
56DH	6FH	bString	o
56EH	00H	bString	
56FH	73H	bString	s
570H	00H	bString	
571H	68H	bString	h
572H	00H	bString	
573H	69H	bString	i
574H	00H	bString	
575H	62H	bString	b
576H	00H	bString	
577H	61H	bString	a
String Descriptor2			
String Descriptor3			

3.17.5 Device Request

3.17.5.1 Standard request

UDC support automatically answer in standard request.

(1) GET_STATUS Request

This request automatically returns to status that is determined by receive side.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B 10000001B 10000010B	GET_STATUS	0	0 Interface endpoint	2	Device, interface or endpoint status

Request to device returns according to priority of little endian as follows.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote wakeup	Self power
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- Remote wakeup: Reinstates current remote wakeup setting. This bit is set or reset by SET_FEATURE or CLEAR_FEATURE request. Default is “0”.
- Self power: Reinstates current power supply setting. This bit return Self or Bus Power according to value that is set to bmAttributes field in Config descriptor.

Request to interface returns 00H of 2 bytes.

Request to endpoint returns according to priority of little endian as follows.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	HALT
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

- HALT: Returns to halt status of selected endpoint.

(2) CLEAR_FEATURE request

This request clears or disables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	CLEAR_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device
  Feature selector: 1 Present remote wakeup setting is disabled.
  Feature selector: except 1 STALL state
- Reception side interface
  STALL state
- Reception side end point
  Feature selector: 0 Halt of relevant endpoint is cleared.
  Note: When cleared HALT state, following is set.
  ･Initialize FIFO
  ･Clear the toggle sequence bit
  ･Clear STALL state
  Feature selector: except 0 STALL state

Note: Stalls if request is to non-existent endpoint.

(3) SET_FEATURE request

This request sets or enables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B 00000001B 00000010B	SET_ FEATURE	Feature selector	0 Interface endpoint	0	None

- Reception side device
  Feature selector: 1 Present remote wakeup setting is disabled.
  Feature selector: except 1 STALL state
- Reception side interface
  STALL state
- Reception side end point
  Feature selector: 0 Halt of relevant endpoint
  Feature selector: except 0 STALL state

Note: Stalls if request is to non-existent endpoint.

(4) SET_ADDRESS request

This request sets the device address. Answer subsequent requests using this device address.

Answer requests using the current device address until the status stage of this request is terminated normally.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ADDRESS	Device Address	0	0	None

(5) GET_DESCRIPTOR request

This request returns appointed descriptor.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ DESCRIPTOR	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

- Device Device transmits device descriptor that is stored in descriptor RAM.
- Config Config transmits config descriptor that is stored in descriptor RAM.

  At this point, it transmits not only config descriptor but also interface and endpoint descriptor.
- String String transmits string descriptor of index that is specified by lower byte of wValue field.

Note: Decriptor of short data length in wLength and descriptor length is automatically transmitted by answer of Get_Descriptor.

(6) SET_DESCRIPTOR request

This request sets or enables the relevant function.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ Descriptor	Descriptor type and Descriptor index	0 or Language ID	Descriptor length	Descriptor

Automatic answer of this request is not supported.

According to INT_SETUP interrupt, if the receiving requested has been identified as a SET_DESCRIPTOR request, take back data after confirming EP0_DSET_A bit of DATASET register is “1”. When completed, access EOP register, and write “0” to EP0_EOPB bit, so, status stage is finished. The process is the same for a vendor request.

Please refer to vendor request section.

(7) GET_CONFIGURATION request

This request returns configuration value of present device.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000000B	GET_ CONFIG	0	0	1	Configuration value

If it is not configured, it returns “0”. Otherwise, it returns the configuration value.

(8) SET_CONFIGURATION request

This request sets device configuration.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000000B	SET_ CONFIG	Configuration value	0	0	None

The configuration value is that specified using lower byte of wValue field.

When this value is “0”, it is not configured.

(9) GET_INTERFACE request

This request returns AlternateSetting value that is set by specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000001B	GET_ INTERFACE	0	Interface	1	Alternate setting

If there is no specified interface, it enters to STALL state.

(10) SET_INTERFACE request

This request selects AlternateSetting in specified interface.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
00000001B	SET_ INTERFACE	Alternate setting	Interface	0	None

If there is no specified interface, it enters STALL state.

(11) SYNCH_FRAME request

This request transmits synchronous frame of endpoint.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
10000010B	SYNCH_FRAME	0	Endpoint	2	Frame No.

Automatic answer of this request is not supported.

According to INT_SETUP interrupt, if request received has been identified as a SYNCH_FRAME request, write 2byte data in Frame No after confirming EP0_DSET_A bit of DATASET register is “0”. When completed, access EOP register, and write “0” to EP0_EOPB bit, so, status stage is completed. This can be used only where the endpoint supports isochronous transfer type and supports this request. The process is the same for a vendor request.

Please refer to vendor request section.

3.17.5.2 Printer Class Request

UDC does not support "Automatic answer" of printer class request.

Processing of Class requests is the same as for vendor requests when answering INT_SETUP interrupt.

3.17.5.3 Vendor request (Class request)

UDC does not support "Automatic answer" of Vendor requests.

According to INT_SETUP interrupt, access the register in which the device request is stored, and identify the request. If this request is a Vendor request, control the UDC externally, and process the Vendor request.

Below is an explanation for the case where data phase is transmitting (Control read), and for the case where data phase is receiving (Control write).

(a) Control Read request

bmRequestType	bRequest	wValue	wIndex	wLength	Data
110000xxB	Vendor specific	Vendor specific	Vendor specific	Vendor specific (Expire 0)	Vendor data

When INT_SETUP is received, identify contents of request by bmRequestType, bRequest, wValue, wIndex and wLength registers and process each request. According to application, access Setup_Received register after request has been identified.UDC must also be informed that INT_SETUP interrupt has been recognized.

After transmitting data prepared in application, access DATASET register, and confirm EP0_DSET_A bit is "0". After confirming, write data FIFO of endpoint 0. If transmitting data is more than payload, write data after it confirming whether EP0_DSET_A bit in DATASET register is "0". (INT_ENDPOINT0 interrupt can be used.) If writing all data is finished, write "0" to EP0 bit of EOP register. When UDC receives this, the status stage finish automatically.

INT_STATUS interrupt is asserted when UDC finishes status stage normally. If finishing status stage normally is recognized by external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, a new SETUP token maybe received. In this case, when INT_SETUP interrupt signal is asserted, "1" is set to STAGE_ERROR bit of EP0_STATUS register Informing externally that the status stage cannot be finished normally.

The dataphase may have finished on a data number that is shorter than the value showed to wLength by protocol of control read transfer type in USB. If the application program is configured using only the wLength value, processing cannot be carried out when the host shifts status stage without arriving at the expected data number. At this point, shifting to status stage can be confirmed by using INT_STATUSNAK interrupt signal. (However, releasing mask of STATUS_NAK bit by using interrupt control register is needed.) In Vendor Request, this problem will not occur because the receiving buffer size is set to host controller by driver (In every host, data (data that is transmitted from device by payload of 8 bytes) may be taken to be short packet until confirmation of payload size on device side. Therefore, exercise care if controlling standard requests by software.)

(b) Control write/request

There is no dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor specific	Vendor specific	Vendor specific	0	None

When INT_SETUP is received, identify contents of request by bmRequestType, bRequest, wValue, wIndex, wLength registers and process each request. According to application, access Setup_Received register after request has been identified. UDC must also be informed that the INT_SETUP interrupt has been recognized. If application processing is finished, write "0" to EP0 bit of EOP register. When UDC receives this, the status stage finish automatically.

There is dataphase

bmRequestType	bRequest	wValue	wIndex	wLength	Data
010000xxB	Vendor specific	Vendor specific	Vendor specific	Vendor specific (Except for 0)	Vendor data

When INT_SETUP is received, identify contents of device request by bmRequestType, bRequest, wValue, wIndex, wLength registers and process each request. According to application, access Setup_Received register after request has been identified. UDC must also be informed that the INT_SETUP interrupt has been recognized.

After receiving data prepared in application, access DATASET register, and confirm EP0_DSET is "1". After confirming, read data FIFO of endpoint 0. If receiving data is more than payload, write data after it confirming whether the EP0_DSET_A bit in DATASET register is "1". (INT_ENDPOINT0 interrupt can be used.) If reading all data is finished, write "0" to EP0 bit of EOP register. When UDC receives this, the status stage finishes automatically.

INT_STATUS interrupt is asserted when UDC finishes status stage normally. If finishing status stage normally is recognized by external application, manage this stage by using this interrupt signal. If status stage cannot be finished normally and during status stage, a new SETUP token may be received. In this case, when INT_SETUP interrupt signal is asserted, "1" is set to STAGE_ERROR bit of EP0_STATUS register informing externally that the status stage cannot be finished normally.

Below is control flow in UDC as seen from application.

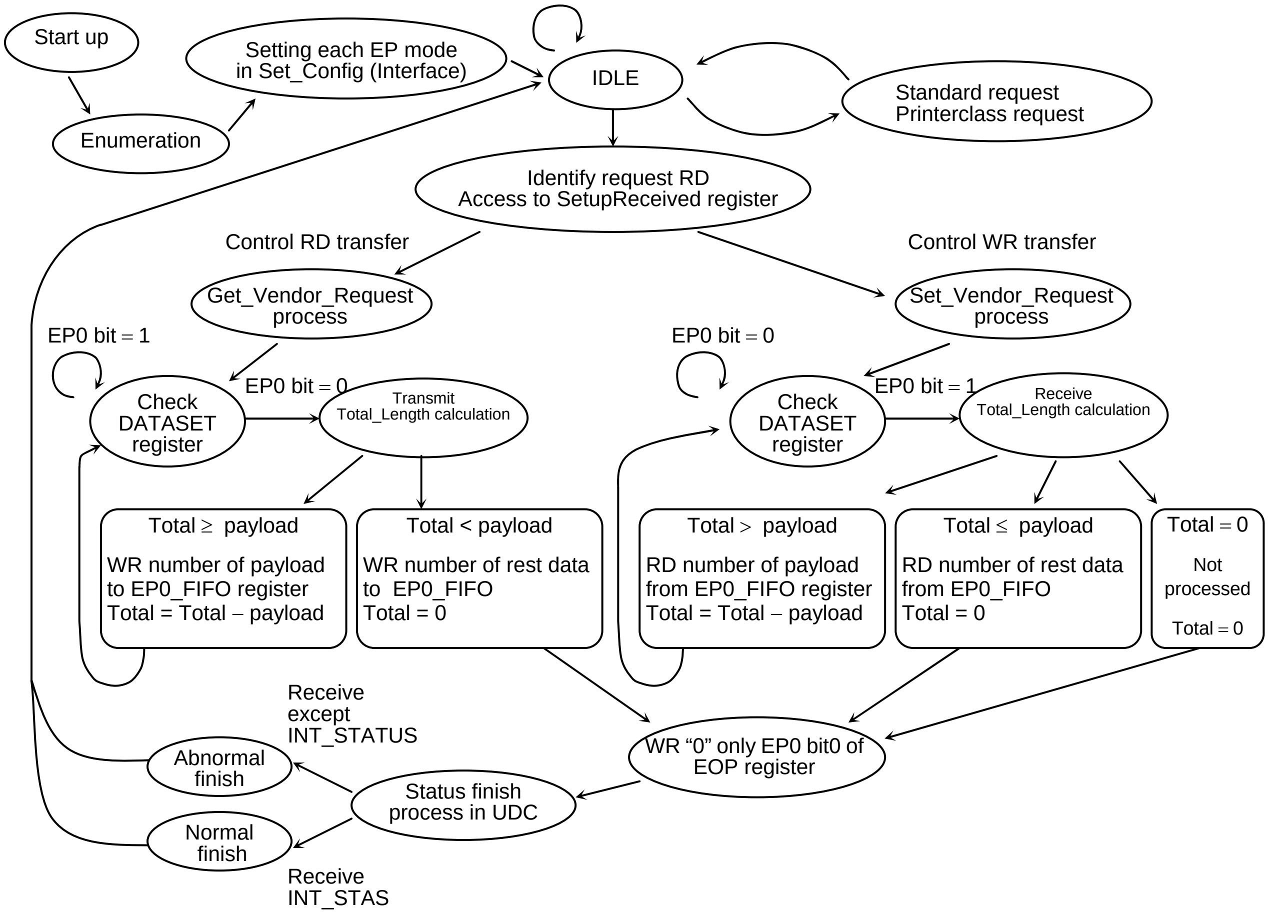


Figure 3.17.2 Control Flow in UDC as seen from Application

Note : This chart does not cover special cases in this flow such as overlap receive SETUP packet.
Please refer to 3.17.6 (2) (c) Control transfer type.

3.17.6 Transfer mode and Protocol Transaction

The UDC performs the following automatically in hardware;

- Receive packet
- Determine address endpoint transfer mode
- Error process
- Confirm toggle bit CRC of data receiving packet
- Generate toggle bit CRC of data transmitting packet, etc
- Handshake answer

(1) Protocol outline

Format of USB packet is shown below. This is processed during transmission and receiving by hardware into the UDC.

- SYNC field

  This field always comes first in each packet, and input data and internal CLK is synchronized in the UDC.

- Packet identification field (PID)

  This field follows SYNC field in every USB packet. The UDC distinguishes the PID type and determines the transfer type by decoding this code.

- Address field

  The UDC uses this field to confirm whether or not this function was specified by the host. The UDC compares the address with that set to the ADDRESS register. If the address accords with it, the UDC continues the process. If the address does not accord, the UDC ignores this token.

- Endpoint field

  If sub-channels of more than two is needed in fields of 4 bits, it decides the function. The UDC can support a maximum of seven endpoints, excluding the control endpoint. Tokens for endpoints that are not permitted are ignored.

- Frame number field

  A field of 11 bits is added by the host at each frame. This field follows the SOF token that is transmitted first in each frame, and the frame number is specified. The UDC reads the content of this field when the SOF token is received, and sets the frame number to the FRAME register.

- Data field

  This field is data of unit bytes in 0 to 1023. When receiving it, the UDC transfers only part of this data to FIFO, and after CRC is confirmed, an interrupt signal is asserted and the UDC informs FIFO that data transfer is completed. When transmitting, following IN token, FIFO data is transferred. Finally, data CRC field is attached.

- CRC function

  5 bits CRC is attached to the token, and15 bits CRC to the data. The UDC automatically compares the CRC of the received data with the attached CRC. When transmitting, CRC is generated automatically and is transmitted. This function may be compared by various transfer modes.

(2) Transfer mode

UDC supports FULL speed transfer mode.

- FULL speed device
  - Control transfer type
  - Interrupt transfer type
  - Bulk transfer type
  - Isochronous transfer type

The following is an explanation of UDC operation in each transfer mode.

The explanation is of data flow up until FIFO.

(a) Bulk transfer type

Bulk transfer type warrants transferring no error between host and function by using detect error and retry. Basically, 3 phases are used - token, data and handshake. However, with flow control and a STALL condition, data phase is changed to hand shake phase, and it become to 2 phases. The UDC holds status of each endpoint, and flow control is controlled in hardware. Each endpoint condition can be confirmed using EPx_STATUS register.

(a-1) Transmission bulk mode

Below is the transaction format for bulk transfer during transmitting.

- Token: IN
- Data: DATA0/DATA1, NAK, STALL
- Handshake: ACK

Control flow

Below is the control-flow when the UDC receive an IN token.

1. The token packet is received and the address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the IN token. If it does not correspond, the state returns to IDLE.
2. Condition of EPx_STATUS register is confirmed.
   - INVALID condition: State returns to IDLE.
   - STALL condition: Stall handshake is returned and state returns to IDLE.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, NAK handshake is returned, and state returns to IDLE.

   If data number of 1 packet is prepared to FIFO, it shifts to 3.
3. Data packet is generated.

   Data packet generated by using toggle bit register in UDC.

   Next, data is transferred from FIFO of internal UDC to SIE, and data packet is generated. At this point, the confirms transferred data number is confirmed. And if there is more than the maximum payload size of each endpoint, bit stuff error is generated, transfer is finished and STATUS becomes STALL.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.
5. When ACK handshake from host is received,
   - Clear FIFO.
   - Clear DATASET register.
   - Renew toggle bit, and prepare for next.
   - Set STATUS to READY.

UDC finishes normally. FIFO can receive the next data.

If a time out occurs without receiving ACK from host,

- Set STATUS to TX_ERR.
- Return FIFO address pointer.

Execute above setting. And wait next retry keeping FIFO data.

This flow is shown in Figure 3.17.3.

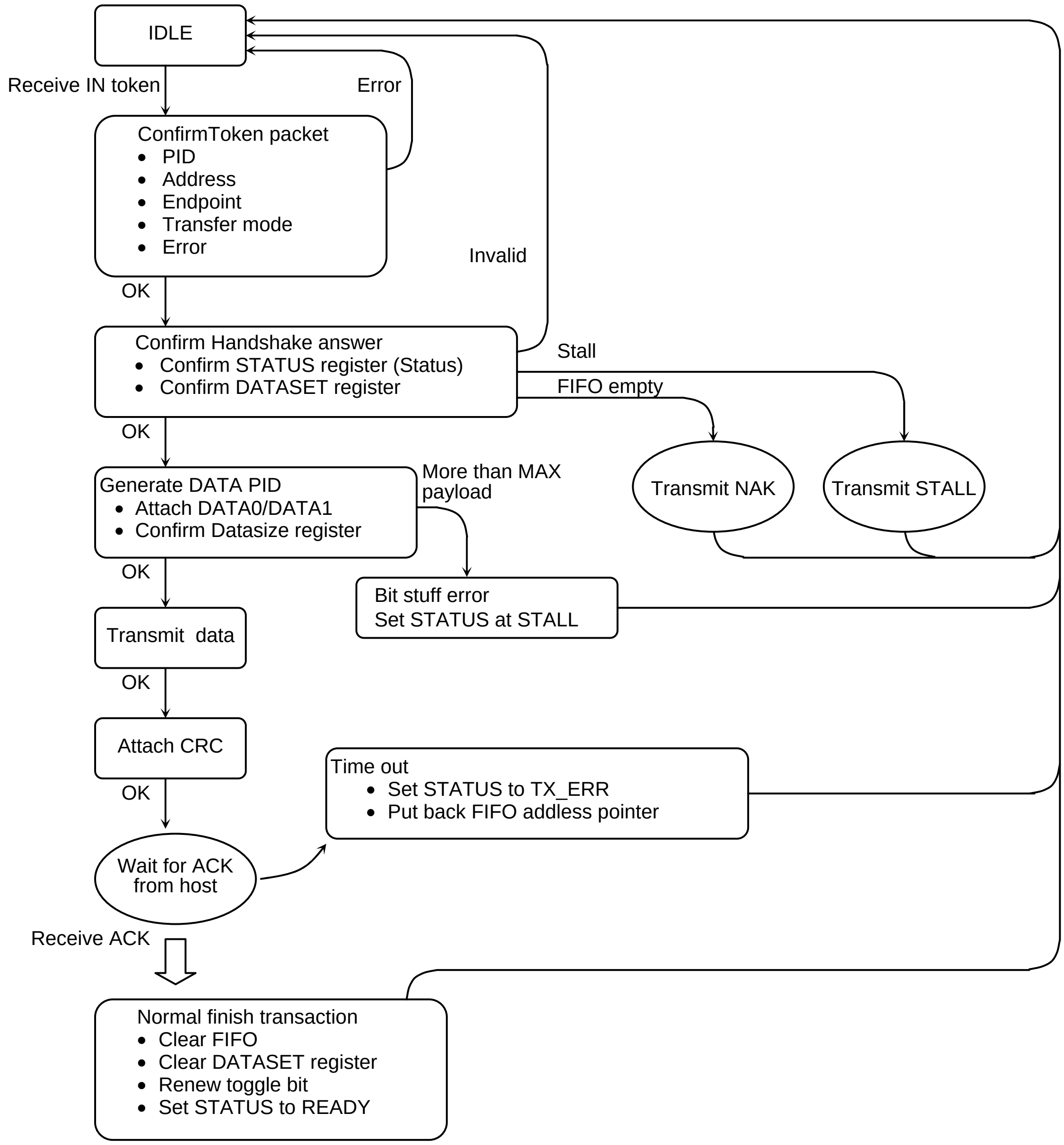


Figure 3.17.3 Control Flow in UDC (Bulk transfer type (transmission)/Interrupt transfer type (transmission))

(a-2) Receiving bulk mode

Below is the transaction format for receiving bulk transfer type.

- Token: OUT
- Data: DATA0/DATA1
- Handshake: ACK, NAK, STALL

Control flow

Below is the control-flow when the UDC receive an IN token.

1. The token packet is received and the address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the OUT token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State returns to IDLE.
   - STALL condition: When dataphase finishes, stall handshake is returned, the state returns to IDLE, and data is canceled.

   FIFO condition is confirmed, if data number of 1 packet is not prepared, present transferred data is canceled, NAK handshake is returned after dataphase, and the state returns to IDLE.
3. Data packet is received.
   Data is transferred from SIE of internal UDC to FIFO. At this point, it confirms transferred data number and if there is more than the maximum payload size of each endpoint, STATUS becomes to STALL and the state returns to IDLE. ACK handshake does not return.
4. After last data is transferred, the counted CRC is compared with the transferred CRC. If they do not correspond, STATUS is set to RX_ERR and the state returns to IDLE. At this point ACK is not returned.
   After retry, when next data is received normally, STATUS changes to DATIN.
   If the data toggle does not correspond, it is judged not to have taken ACK in the last loading the current loading is regarded as a retry of the last loading and data is canceled. Set STATUS as RX_ERR, return to host and return to IDLE. FIFO address pointer returns and the next data can be received.
5. If CRC is compared with toggle and it finishes normally, ACK handshake is returned.
   Below is the process in the UDC.
   - Set transfer data number to DATASIZE register.
   - Set DATASET register.
   - Renew toggle bit, and prepare for next.
   - Set STATUS to READY.

UDC finishes normally.

This flow is shown in Figure 3.17.4.

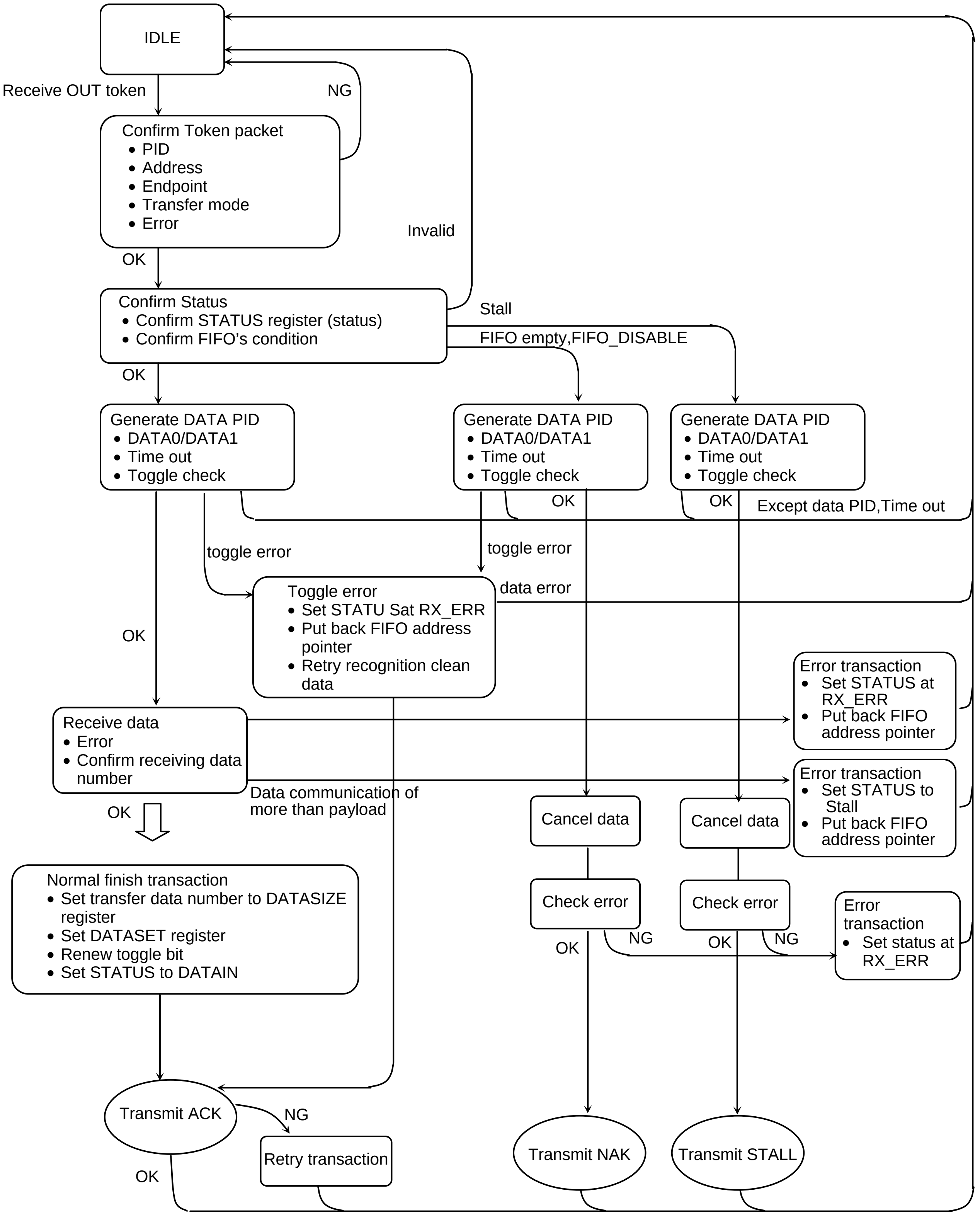


Figure 3.17.4 Control Flow in UDC (Bulk transfer type (Receiving))

(b) Interrupt transfer type

Interrupt transfer type uses the same transaction format as transmission bulk transfer.

For transmission using toggle bit, hardware setting and answer in the UDC are the same as for transmission bulk transfer. Interrupt transfer can be transferred without using toggle bit. In this case, if ACK handshake from host is not received, toggle bit is renewed, and finish is normal. The UDC clears FIFO for next transfer.

(b-1) Interrupt transmitting mode (Toggle mode)

UDC operation is same as in bulk transmission mode. Please refer to section (a).

(b-2) Interrupt transmission mode (Not toggle mode)

This is basically the same as bulk transmission mode. However, if ACK handshake from host is not received, transaction is different.

When ACK handshake from host is received after transmission of data packet

- Clear FIFO.
- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to READY.

UDC finishes normally by above transaction. FIFO can receive next data.

If a time out occurs without receiving ACK from host,

- Clear FIFO.
- Clear DATASET register.
- Renew toggle bit and prepare for next.
- Set STATUS to TX_ERR.

Execute above setting. This setting is the same except for STATUS changes.

(c) Control transfer type

Control transfer type is configured in the three stages below.

- Setup stage
- Data stage
- Status stage

Data stage is sometimes skipped. Each stage is configured in one or several transactions. The UDC executes each transaction while managing three stages in hardware. Control transfer has the 3 types given below depending on whether there is data stage or not, and on direction.

- Control read transfer type
- Control write transfer type
- Control write transfer type (No data stage)

The 3 transfer sequences are shown in Figure 3.17.6, Figure 3.17.7 and Figure 3.17.8.

The UDC automatically answers standard requests in hardware. Class request and vendor request must have an intervening CPU controlling the UDC.

Below is the control flow in the UDC and the control flow in the intervening CPU.

(c-1) Setup stage

Setup stage is the same as transmission bulk transaction except that token ID becomes SETUP.

However, control flow in the UDC is different.

- Token: SETUP
- Data: DATA 0
- Handshake: ACK

Control flow

Below is the control flow in the UDC when SETUP token is received.

1. SETUP token packet is received and address, endpoint number and error are confirmed. It also checks whether the relevant endpoint is in control transfer mode.
2. STATUS register state is confirmed.

   State return to IDLE only if it is INVALID state.

   In bulk transfer mode, receiving data is enabled by STATUS registers value and FIFO condition. However, in SETUP stage, STATUS is returned to READY and accessing from the CPU to FIFO is always prohibited and internal FIFO of endpoint 0 is cleared. It also prepares for following dataphase.

   If the CPU accesses Setup Received registers in the UDC, it recognizes as Device request as received, and accessing from the CPU to EP0 is enabled.

   This function is for receiving a new request when the current device request has not finished normally.

3. Data packet is received.

   Device request of 8 bytes from SIE in UDC is transferred to the request register below.

   - bmRequestType register
   - bmRequest register
   - wValue register
   - wIndex register
   - wLength register

4. After last data is transferred, counted CRC is compared with transferred CRC. If they do not correspond, STATUS is set to RX_ERR and the state returns to IDLE. At this point it does not return ACK, and host retries.

5. If CRC corresponds with toggle and it finishes normally, ACK handshake is returned to host. The process in the UDC is shown below.

   - Receiving device request is judged whether software control or hardware control. If the request needs control in software, INT_SETUP interrupt is asserted. If hardware is used, INT_SETUP interrupt is not asserted.
   - According to stage control flow, prepare for next stage.
   - Set STATUS to DATAIN.
   - Set toggle bit to “1”.

The Setup stage is completed by the above.

This flow is shown in Figure 3.17.2.

8-byte data that is transferred by this SETUP stage is device request.

The CPU must process corresponding to device request.

The UDC detects the following contents only from data of 8 bytes, and it manages stage in hardware.

- Whether there is data stage or not
- Data stage direction

These are used to determine control read transfer type, control write transfer type, and control write transfer type (no data phase).

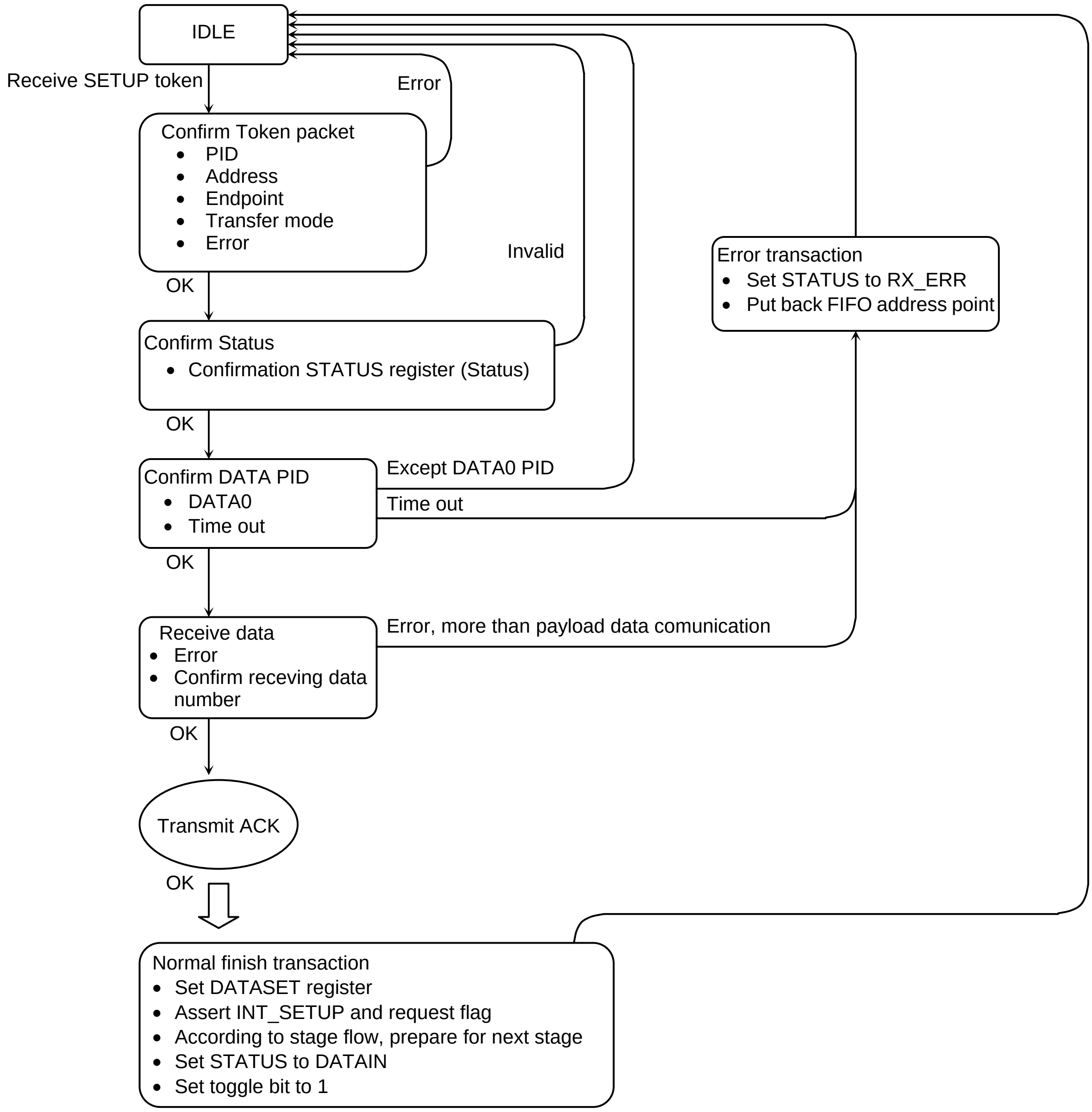


Figure 3.17.5 Control Flow in UDC (Setup stage)

(c-2) Data stage

Data stage is configured by one or several transactions based on toggle sequence.

The transaction is the same as for format transmission or receiving bulk transaction except for the following differences;

- Toggle bit starts from "1" by SETUP stage.
- It determines whether right or not by comparing IN and OUT token with direction bit of device request. If a token of the opposite direction is received, it is recognized as status stage.
- INT_ENDPOINT0 interrupt is asserted.

(c-3) Status stage

Status stage is configured 0-data-length packet with DATA1's PID and handshake IN or OUT token. It uses a transaction in the opposite direction to the preceding stage.

The combination is given below.

- Control read transfer type: OUT
- Control write transfer type: IN
- Control write transfer type (not dataphase): IN

UDC processes status stage base of control flow in control transfer type. At this point, CPU must write "0" to EP0 bit of EOP register in last transaction for status stage to finish normally.

Details of status stage are given below.

(c-3-1) IN status stage

IN status stage transaction format is given below.

- Token: IN
- Data: DATA1 (0 data length), NAK, STALL
- Handshake: ACK

Control flow

The transaction flow of IN status stage in UDC is given below.

1. Token packet is received and address, endpoint number and error are confirmed. If it does not correspond, the state returns to IDLE. If status stage is enabled based on stage control flow in the UDC, advance to next stage.
2. STATUS register state is confirmed.
   - INVALID condition: State returns to IDLE.
   - STALL condition: Stall handshake is returned and state returns to IDLE.

   Confirmation of whether EOP register is accessed or not is carried out externally. If it is not accessing, NAK handshake is returned to continue control transfer and state returns to IDLE.
3. If EOP register is access is confirmed, 0-data-length data packet and CRC are transmitted.

4. If ACK handshake from host is received,
   - Set STATU to READY.
   - Assert INT_STATUS interrupt.

It finishes normally by the above transaction.

If a time out occurs without receiving ACK from host,

- Set STATUS register to TX_ERR and state returns to IDLE and wait for restring status stage.

At this point, if new SETUP stage is started without status stage finishing normally, the UDC sets error to STATUS register.

(c-3-2) OUT status stage

The transaction format for OUT status stage is given below.

- Token: OUT
- Data: DATA1 (0 data length)
- Handshake: ACK, NAK, STALL

Control flow

The transaction flow for OUT status stage in the UDC is given below.

1. Token packet is received and address, endpoint number and error are confirmed. If they do not correspond, the state returns to IDLE. If status stage is enabled base on stage control flow in the UDC, advance to next stage.
2. STATUS register state is confirmed.
   - INVALID condition: State returns to IDLE.
   - STALL condition: Data is cleared, stall handshake is returned, and state returns to IDLE.

   Whether EOP register is accessed or not is confirmed externally. If it is not accessed, NAK handshake is returned to continue control transfer and state returns to IDLE.
3. If EOP register is access is confirmed, 0-data-length data packet and CRC are received.
4. If there is no error in data, ACK handshake is transmitted to host.
   - Set STATUS to READY.
   - Assert INT_STATUS interrupt.

It finishes normally by the above transaction.

If there is an error in data, ACK handshake is not returned.

- Set RX_ERR to STATUS register and return to IDLE. It waits to retry status stage.

At this point, if new SETUP stage is started without status stage finishing normally, the UDC sets error to STATUS register. For sequence of this protocol, refer to section supplement.

(c-4) Stage management

The UDC manages each stage of control transfer by hardware.

Each stage is changed by receiving token from USB host, or CPU accesses register. Each stage in control transfer type has to process combination software. UDC detects the following contents from 8-byte data in SETUP stage. The stage is managed by determining control transfer type.

- Whether there is data stage or not
- Data stage direction

Based on these it is determines to be either control read transfer type control write transfer type, or control write transfer type (No data stage).

Various conditions for changing stage in control transfer are given below.

If receiving token for next stage from host before switching to next stage from state of internal UDC, NAK handshake is returned and BUSY is informed to USB host. In all control transfer types, if SETUP token is received from host current transaction is stopped, and it switches to SETUP stage in the UDC. The CPU receives new INT_SETUP even if it is processing previous control transfer.

Stage change condition of control read transfer type

1. Receive SETUP token from host
   - Start setup stage in UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
   - Change data stage in the UDC.
2. Receive IN token from host
   - The CPU receives a request from the request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register to inform the UDC that INT_SETUP interrupt has been recognized.
   - According to Device request, monitor EP0 bit of DATASET register, and write data to FIFO.
   - If the UDC is set data of payload to FIFO or CPU set short packet transfer in EOP register, EP0 bit of DATASET register is set.
   - The UDC transfers data that is set to FIFO to host by IN token interrupts.
   - When the CPU finishes transaction, it writes “0” to EP0 bit of EOP register.
   - Change status stage in the UDC.
3. Receive OUT token from host.
   - Return ACK to OUT token, and change state to IDLE in the UDC.
   - Assert INT_STATUS interrupt externally.

These changing conditions are shown in Figure 3.17.6.

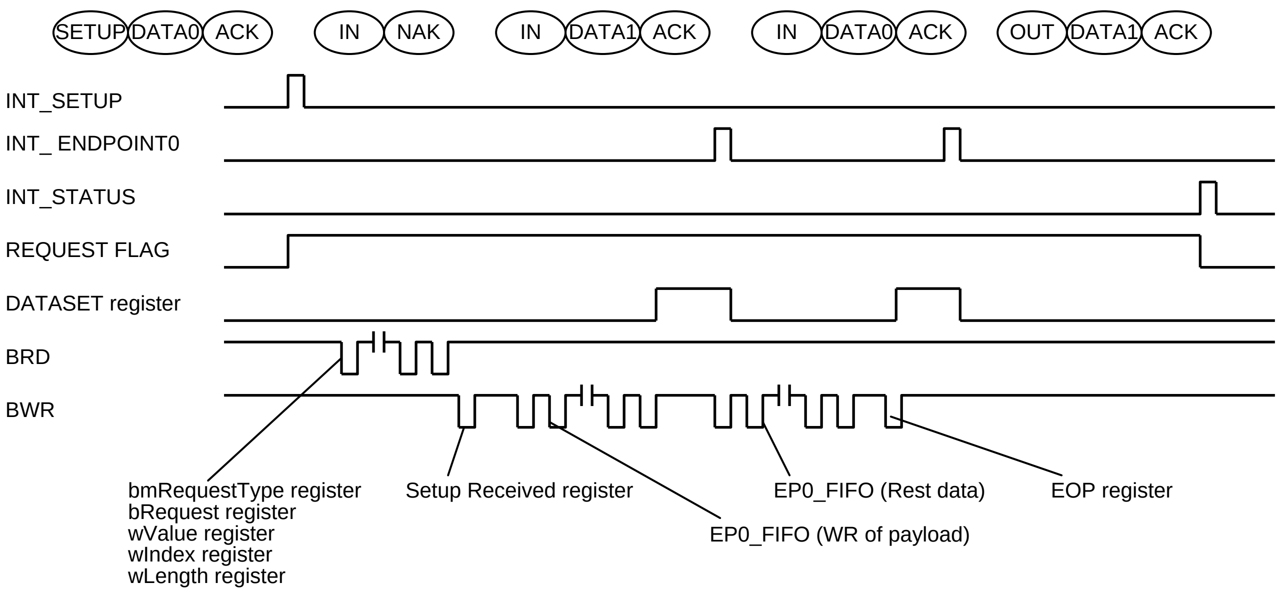


Figure 3.17.6 The Control Flow in UDC (Control Read Transfer Type)

Stage change condition of control write transfer type

1. Receive SETUP token from host.
    - Start setup stage in the UDC.
    - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
    - Change data stage in the UDC.
2. Receive OUT token from host.
    - CPU receives a request from the request register every INT_SETUP interrupt.
    - Judge request and access Setup Received register for inform the UDC that INT_SETUP interrupt has been recognized.
    - Receive dataphase data normally, and set EP0 bit of DATASET register.
    - The CPU receives data in FIFO by setting DATASET.
    - The CPU processes receiving data by device request.
    - When the CPU finishes transaction, it writes "0" to EP0 bit of EOP register.
    - Change status stage in the UDC.
3. Receive IN token from host.
    - Return data packet of 0 data to IN token, and change state to IDLE in the UDC.
    - Assert INT_STATUS interrupt externally when ACK for 0 data packet is received.

These changing conditions are shown in Figure 3.17.7.

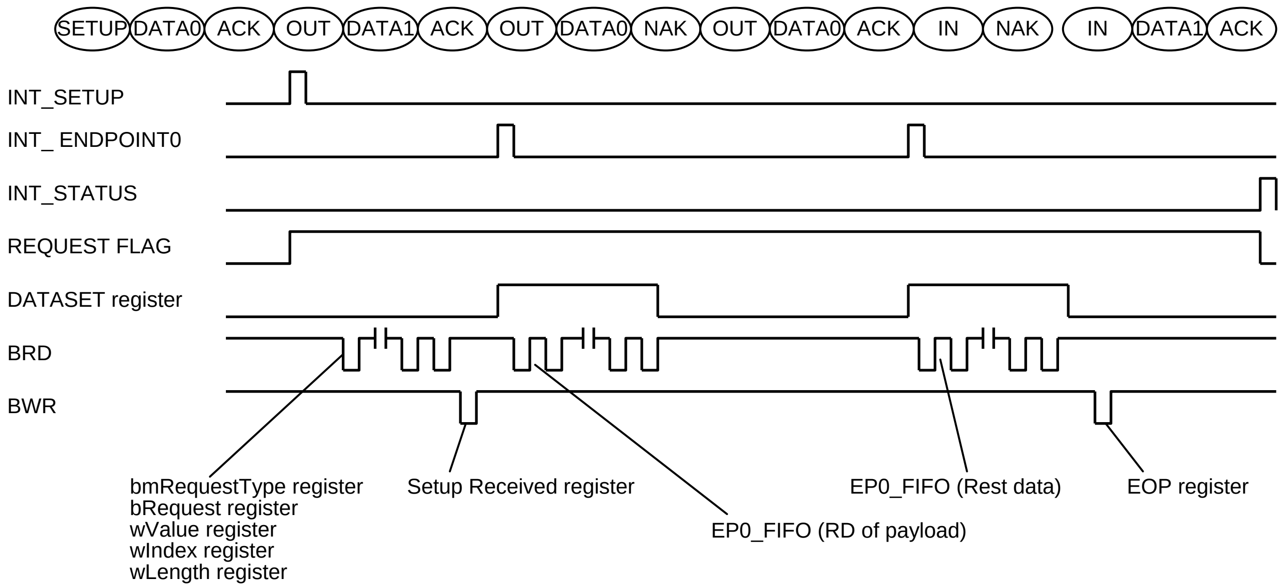


Figure 3.17.7 The Control Flow in UDC (Control Write Transfer Type)

In control read transfer type, transaction number of data stage does not always correspond with the data number specified by the device request. The CPU can therefore process using INT_STATUSNAK interrupt. However, when class and vendor request is used, wLength value corresponds to data transfer number in data phase. With this setting, using this interrupt is not need. Data stage data can be confirmed by accessing DATASIZE register.

Stage change condition of control write (no data stage) transfer type

1. Receive SETUP token from host
   - Start setup stage in the UDC.
   - Receive data in request normally and judge. And assert INT_SETUP interrupt externally.
   - Change data stage in the UDC.
2. Receive IN token from host
   - CPU receives a request from the request register every INT_SETUP interrupt.
   - Judge request and access Setup Received register to inform the UDC that INT_SETUP interrupt has been recognized.
   - The CPU processes receiving data by device request.
   - When the CPU finishes transaction, it writes "0" to EP0 bit of EOP register.
   - Change status stage in the UDC.
   - Return data packet of 0 data to IN token, and change state to IDLE in the UDC.
   - Assert INT_STATUS interrupt externally when ACK for 0 data packet is received.

These change condition is Figure 3.17.8.

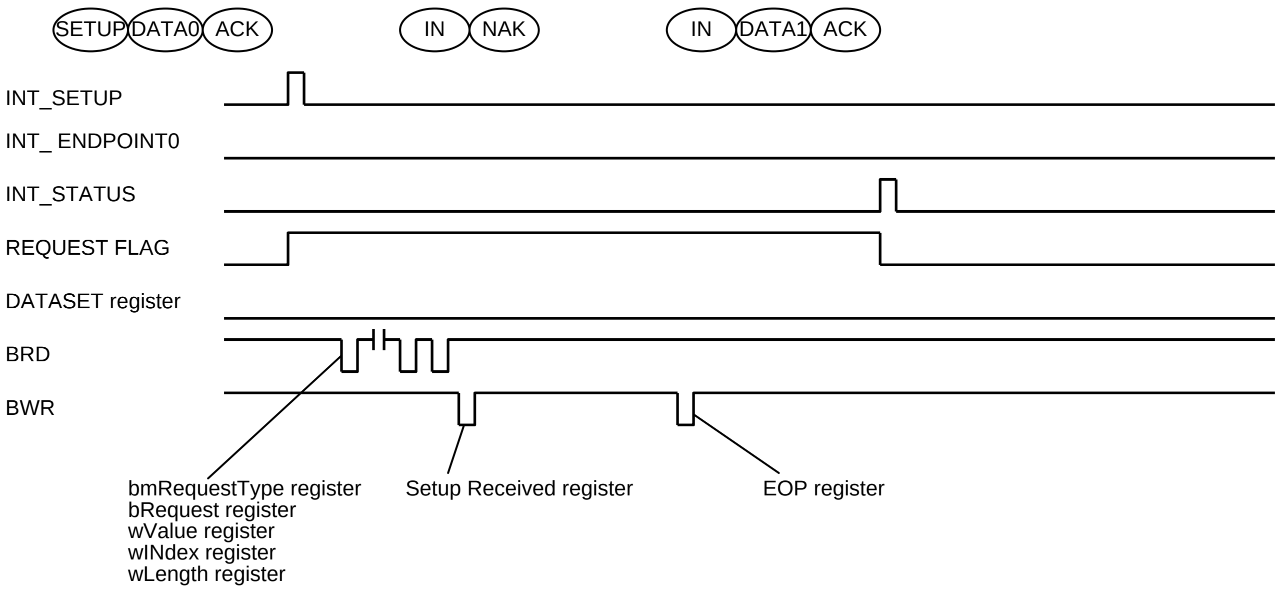


Figure 3.17.8 The Control Flow in UDC (Control Write Transfer Type not Dataphase)

(d) Isochronous transfer type

Isochronous transfer type is guaranteed transfer by data number that is limited to each frame.

However, this transfer does not retry when an error occurs. Therefore, Isochronous transfer type transfer only 2 phases (token, data) and it does not use handshake phase. And data PID for data phase is always DATA0 because of this transaction does not support toggle sequence. Therefore, UDC does not confirm when data PID is in receiving mode.

Isochronous transfer type processes data every frame. Therefore, all transaction for completed transfer use receiving SOF token. The UDC uses FIFO that is divided into two in Isochronous transfer type.

(d-1) Isochronous transmission mode

The transaction format for Isochronous transfer type format in transmitting is given below.

- Token : IN
- Data : DATA0

Control flow

Isochronous transfer type is frame management. And data that is written to FIFO in endpoint is transmitted by IN token in the next frame.

Below are two conditions in FIFO of Isochronous transmission mode transferring.

X. FIFO for storing data that transmits to host in present frame
(DATASET register bit = 1)

Y. FIFO for storing data for transmitting host in next frame
(DATASET register bit = 0)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. The flow below is explained as X Condition (packet A), Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Control flow in the UDC when receiving IN token is shown below.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the IN token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
   INVALID condition: State returns to IDLE.
3. Data packet is generated.
   Data packet is generated. At this point, data PID is always attached to DATA0. Next, data is transferred from FIFO (X condition) of packet A in UDC to SIE and DATA packet is generated.
4. CRC bit (counted transfer data of FIFO from first to last) is attached to last.

5. Below is transaction when SOF token is received from host.
   - Change the packet A's FIFO from X Condition to Y Condition and clear data.
   - Change the packet B from Y Condition to X Condition.
   - Set frame number to frame register.
   - Assert SOF and inform externally that frame is incremented.
   - DATASET register clears packet A bit and it sets packet B bit arrangement loading in present frame.
   - Set STATUS to READY.

The UDC finishes normally by above transaction.

Packet A's FIFO can be received with next data.

In renewed frame, Packet A's FIFO interchanges with packet B's FIFO, and transaction uses same flow.

If SOF token is not received by error and so on, this data is lost because frame is not renewed. There is no problem in receiving PID if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and exact frame number is unknown. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

Note : EPx_DATASETA,B change at 3 clocks of 12MHz after receiving SOF. Write data to FIFO after EPx_DATASETA,B are changing.

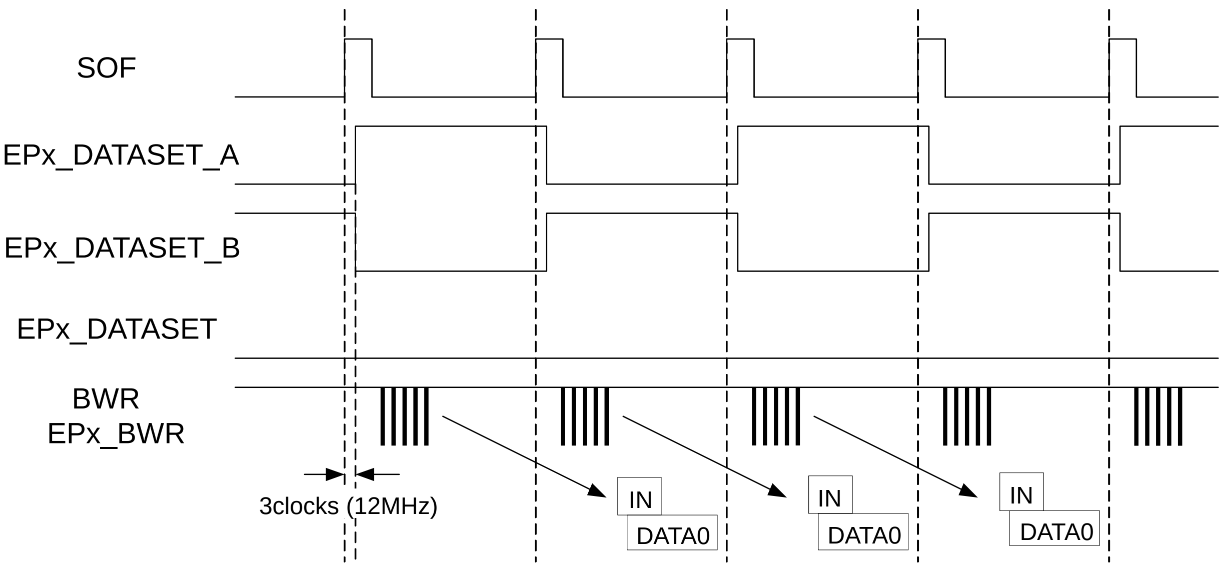


Figure 3.17.9 Isochronous transfer Mode

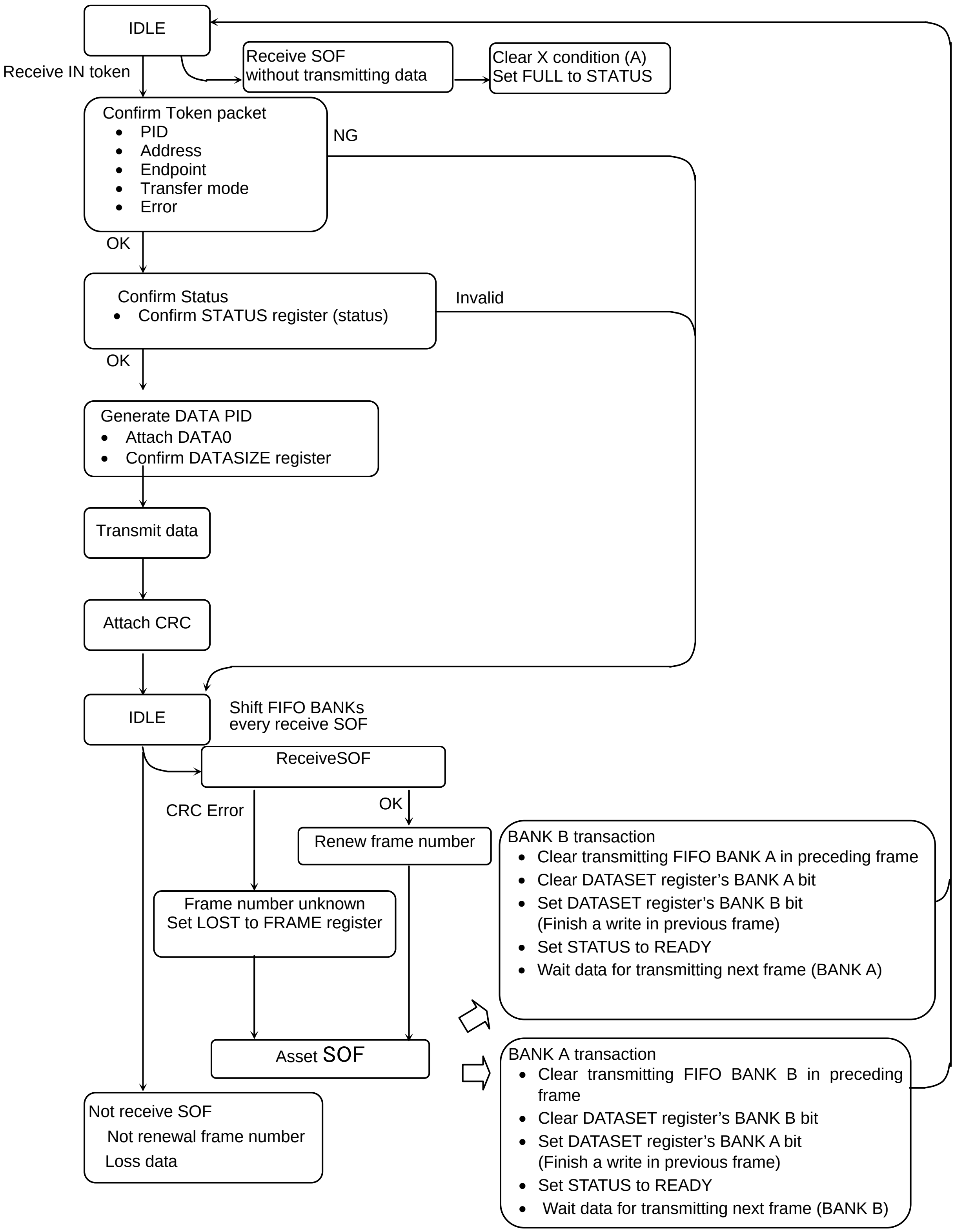


Figure 3.17.10 Control Flow in UDC (Isochronous transfer type (Transmission))

(d-2) Isochronous receiving mode

Transaction format for Isochronous transfer type in receiving is given below.

- Token :OUT
- Data : DATA0

Control flow

Isochronous transfer type is frame management. And data that is written to FIFO by OUT token is received to the CPU in the next frame.

Below are two conditions in FIFO of Isochronous receiving mode transferring

X. FIFO for storing data received from host in present frame
(DATASET register bit = 0)

Y. FIFO for storing data for transmitting host in previous frame
(DATASET register bit = 1)

FIFO that is divided into two (packet A and packet B) conditions is whether X condition or Y condition. The flow below explains X Condition (packet A) and Y Condition (packet B) in present frame.

X and Y conditions change one after the other by receiving SOF.

Below is control flow in the UDC when receiving OUT token.

The whole transaction is processed by hardware.

1. Token packet is received and address endpoint number error is confirmed, and it checks whether the relevant endpoint transfer mode corresponds with the OUT token. If it does not correspond, the state returns to IDLE.
2. Condition of status register is confirmed.
   - INVALID condition: State return to IDLE.
3. Data packet is received.
   Data is transferred from SIE into the UDC to packet A's FIFO (X Condition).
4. After last data has been transferred, and counted CRC is compared with transferred CRC. When transfer is finished, the result is reflected to STATUS. However, data is stored FIFO, data number that packet A is received is set to DATASIZE register of packet A.
5. The transaction when SOF token from host is received is given below.
   - Change packet A's FIFO from X Condition to Y Condition.
   - Change packet B from Y Condition to X Condition, and clear data. Prepare for next transfer.
   - Set frame number to frame register.
   - Assert SOF and inform externally that frame is incremented.
   - DATASET register set packet A bit and clear packet B bit arrangement loading in present frame.
   - If CRC comparison result agrees it, DATAIN is set to STATUS. If result does not agree, RX_ERR is set to STATUS.

The UDC finishes normally by the above transaction.

The CPU takes back packet A's data.

In renewed frame, Packet A's FIFO interchanges with packet B's FIFO, and the transaction uses the same flow.

If SOF token is not received by error and so on, this data is lost because the frame is not renewed. There is no problem in receiving PID and if frame data is received with CRC error, USB sets LOST to STATUS on FRAME register, and exact frame number is unknown. However, in this case, SOF is asserted and FIFO condition is renewed. If SOF token is received without transmit and transfer Isochronous in frame, UDC clears FIFO (X Condition) and sets STATUS to FULL.

These are shown in Figure 3.17.12.

Note : EPx_DATASET changes at 2 clocks of 12MHz after receiving SOF. Read data from FIFO after EPx_DATASET is rising.

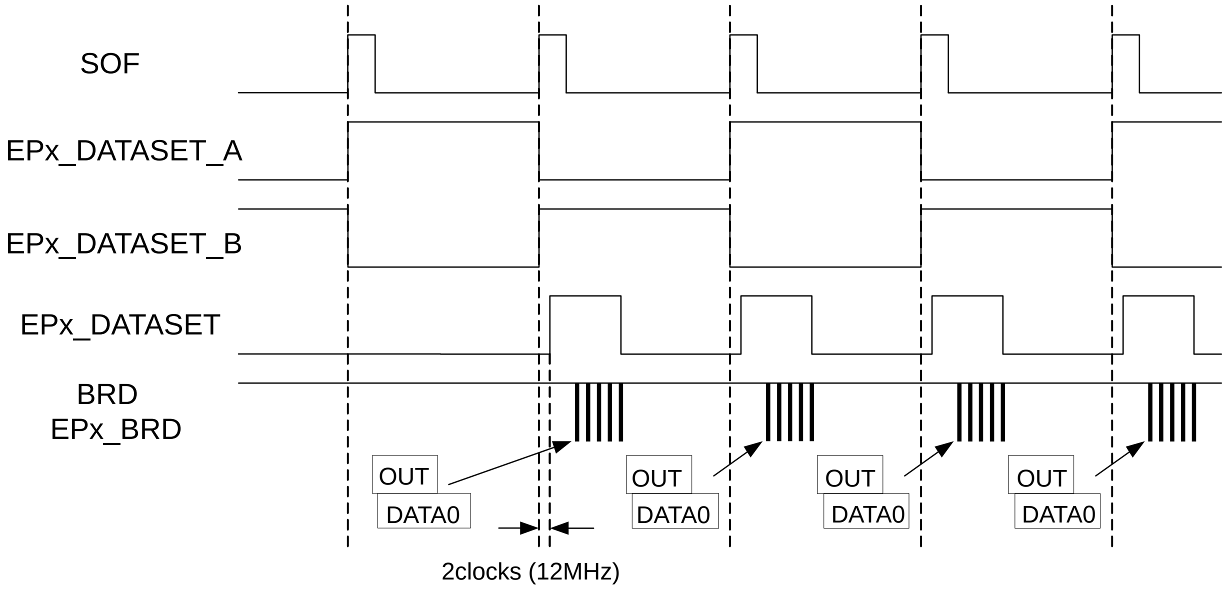


Figure 3.17.11 Isochronous Receiving mode

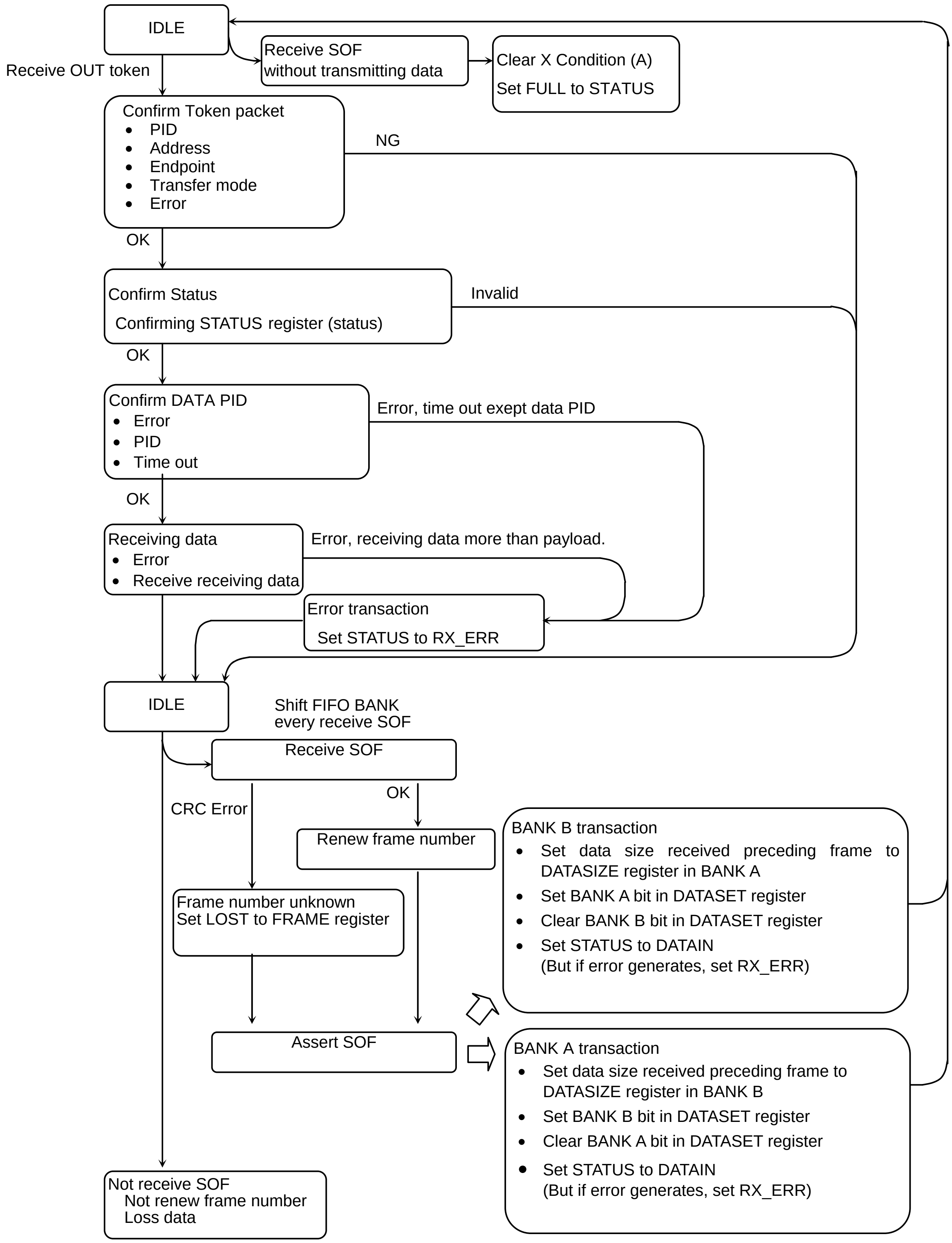


Figure 3.17.12 Control Flow in UDC (Isochronous transfer type (Receiving))

3.17.7 Bus Interface and Access to FIFO

(1) CPU bus interface

The UDC prepares two types of FIFO access, single packet and dual packet. In single packet mode, FIFO capacity that is implemented by hardware is used as large FIFO. In dual packet mode, FIFO capacity is divided into two and used as two FIFOs. It is also used as an independent FIFO. Even if the UDC is transmitting and receiving to USB host, it can be used as an efficient bus by possible load to FIFO.

But control transfer type receives only single packet mode.

Epx_SINGLE signal in dual packet mode must be fixed to "0". If this signal is fixed to "0", FIFO register runs in single mode.

Sample: Where endpoint 1 is used to dual packet of payload 64 bytes.

EP1_FIFO size	:	Prepare 128 bytes
EP1_SINGLE signal	:	Hold 0
EP1 Descriptor setting		
Direction	:	Optional
Max payload size	:	64 bytes
Transfer mode	:	Optional

(a) Single packet mode

This is data sequence of single packet mode when CPU bus interface is used. Figure 3.17.13 is receiving sequence. Figure 3.17.14 is transmitting sequence. This chapter focuses on access to FIFO. For Data sequence with USB host refer to chapter 5.

Endpoint 0 cannot be changed to exclusive single packet mode. Endpoints 1 to 3 can be changed between single packet and dual packet by setting Epx_SINGLE register. Do not change packet when transferring.

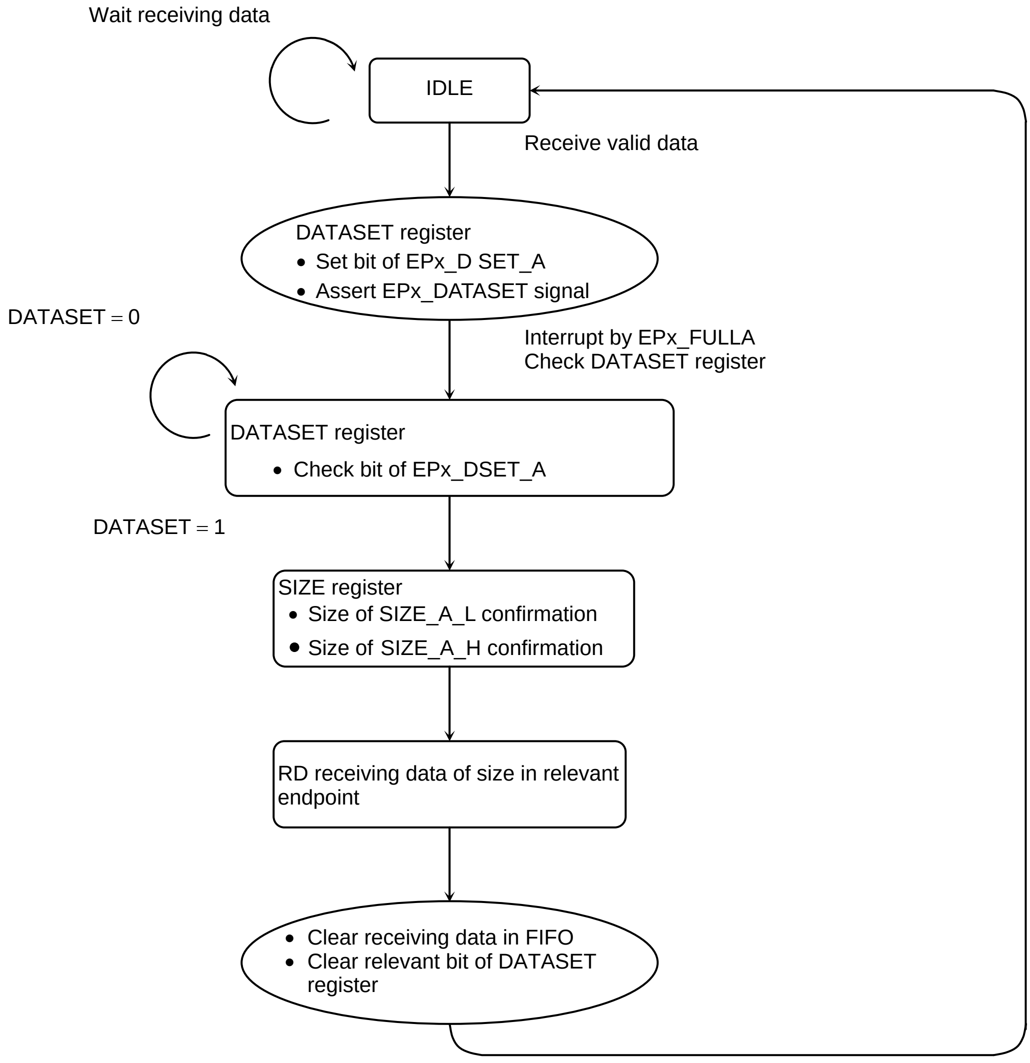
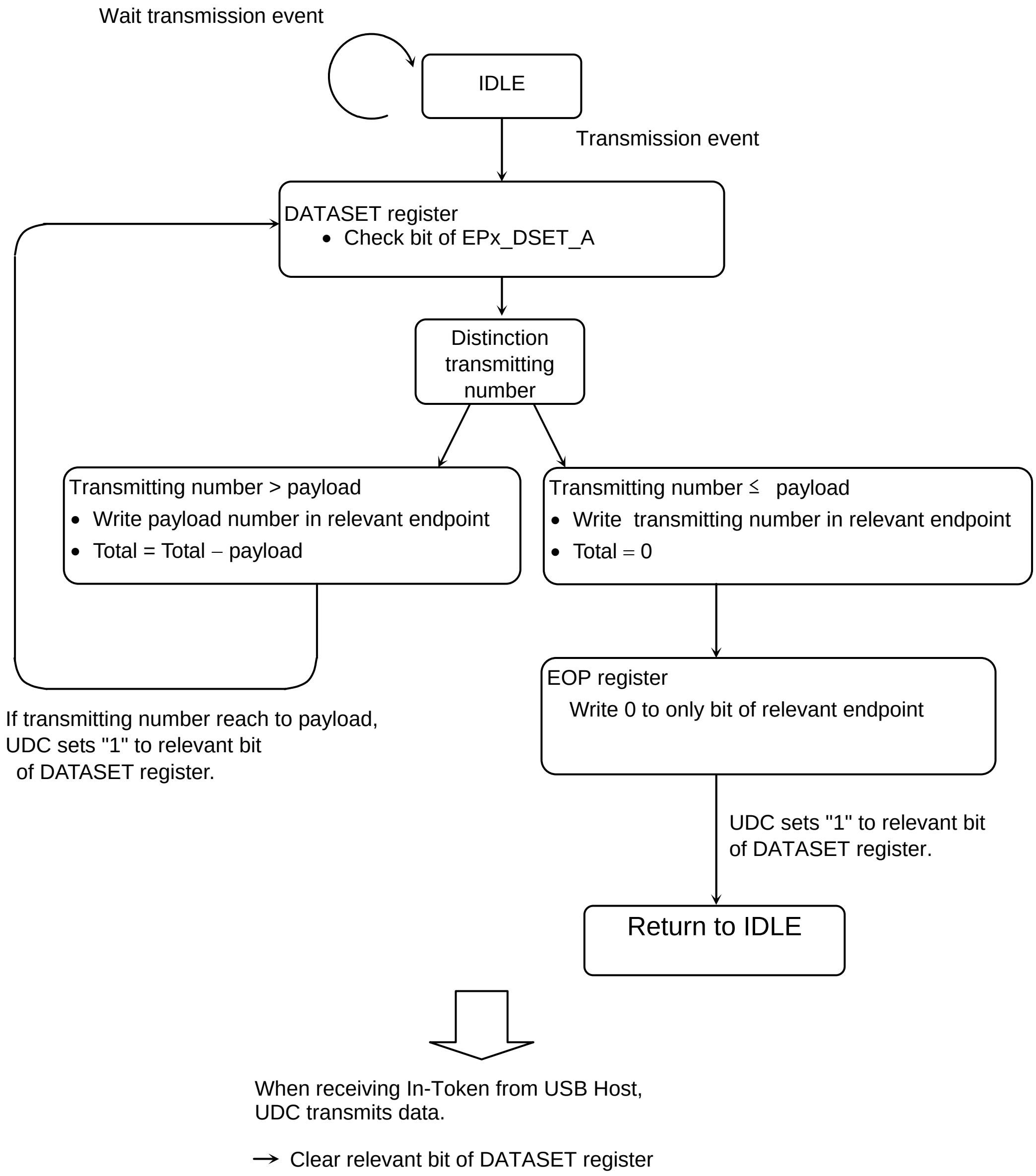


Figure 3.17.13 Receiving Sequence in Single Packet Mode

Below is the transmitting sequence in single packet mode.



- Accessing to EOP register is needed in transmitting short packet.
- Acessing endpoint0 is used for showing closing control transfer.Therefore,always access to endpoint 0 in closing control transfer whether short packet or not.

Figure 3.17.14 Transmitting Sequence in Single Packet Mode

(b) Dual packet mode

In dual packet mode, FIFO is divided into A and B packet, and is controlled according to priority in hardware. It can be performed at once, transmitting and receiving data to USB host and exchanges to external of UDC.

When it reads out data from FIFO for receiving, confirm condition of two packets, and consider the order of priority. If it has received data to two packets, the UDC outputs from first receiving data by FIFO that can be accessed are common in two packets. EPx_SIZE register is prepared for both packet A and packet B. First, the CPU must recognize the data number of first receiving packet by PKT_ACTIVE bit. If PKT_ACTIVE bit has been set to 1, that packet is received first. Packet A and packet B set data turn about always.

This is shown below.

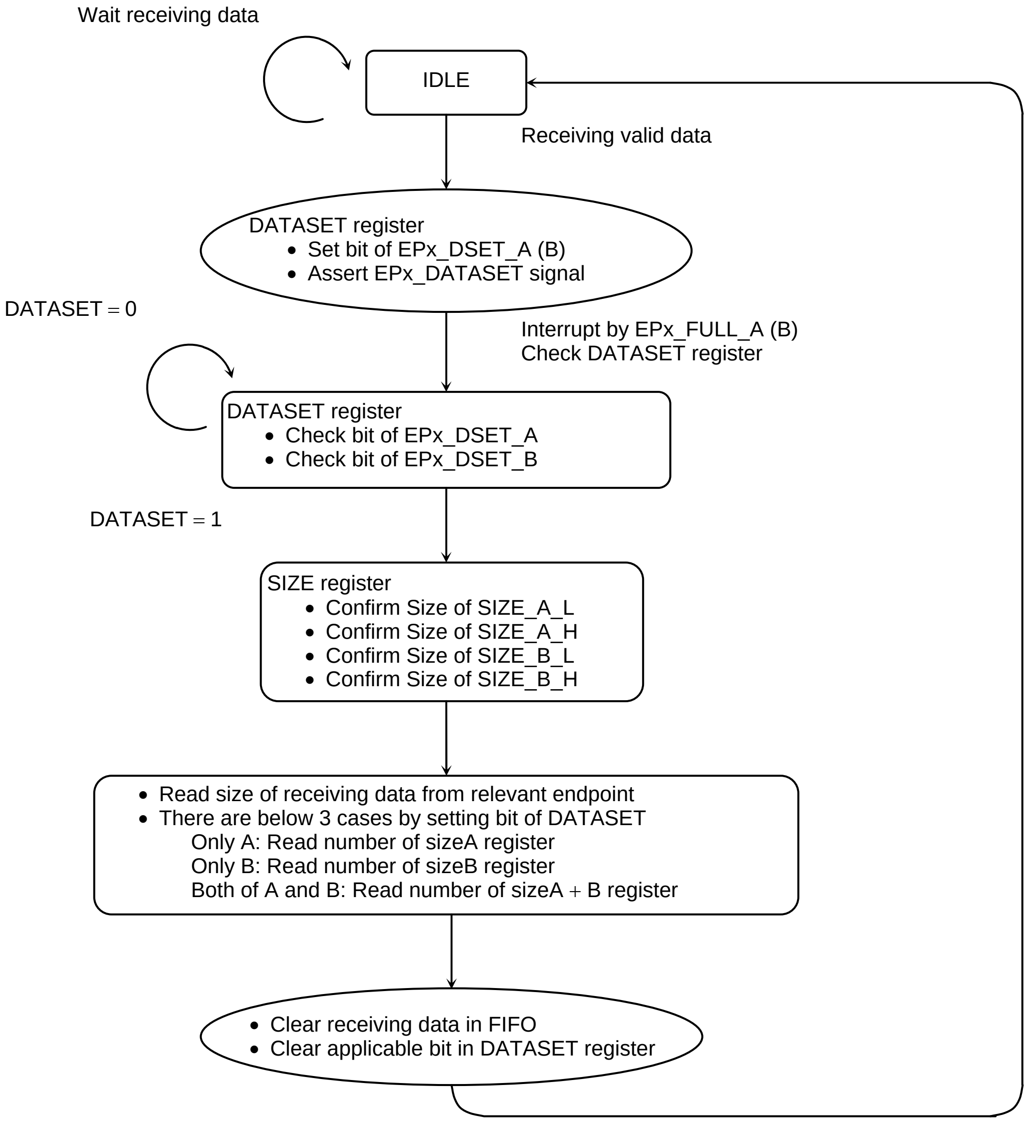


Figure 3.17.15 Receiving Sequence in Dual Packet Mode

Data can be set to available FIFO when transmitting regardless of packet A or B.
Below is the Transmitting Sequence in Dual Packet Mode.

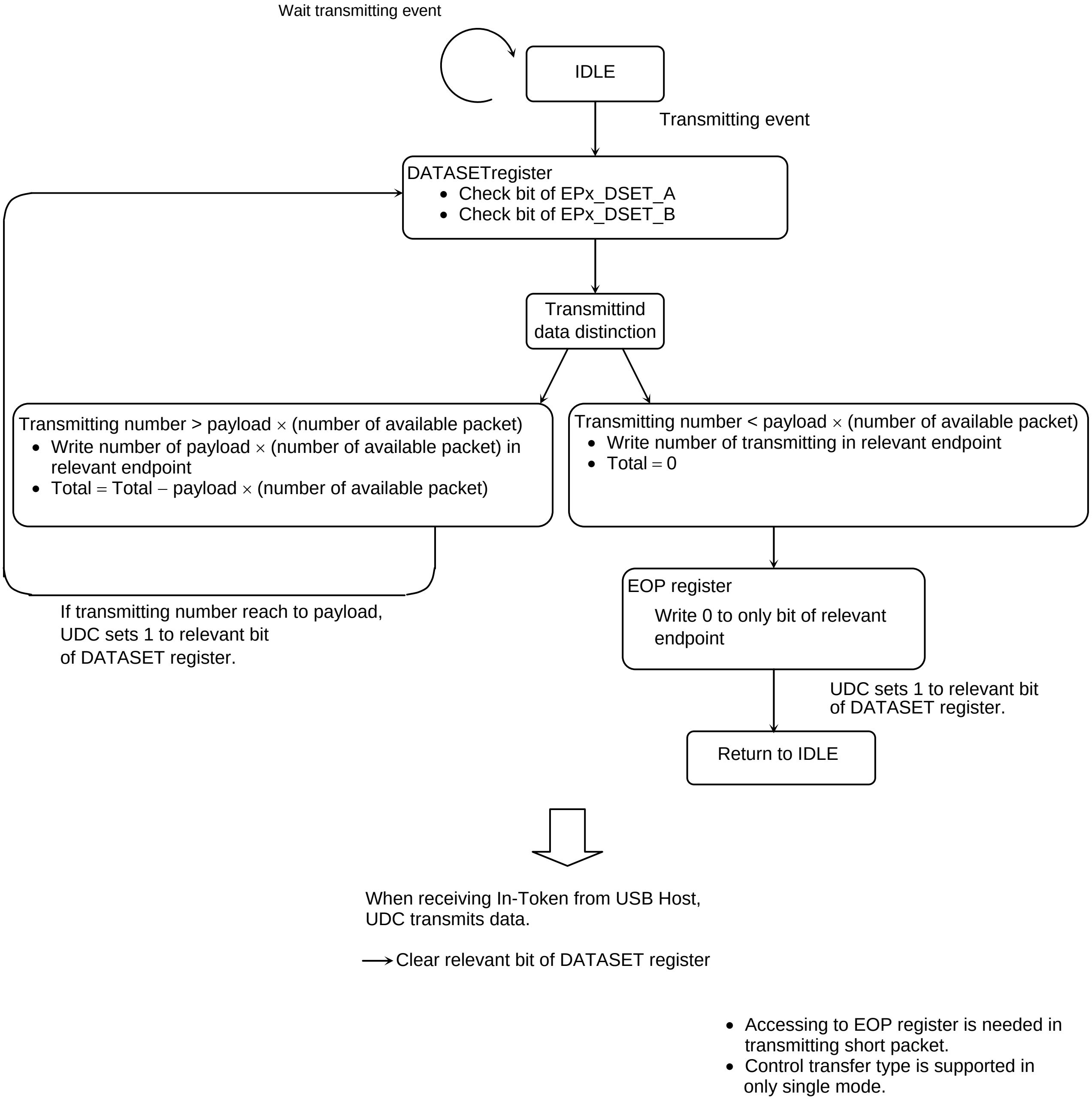


Figure 3.17.16 Transmitting Sequence in Dual Packet Mode

(c) Issuance of NULL packet

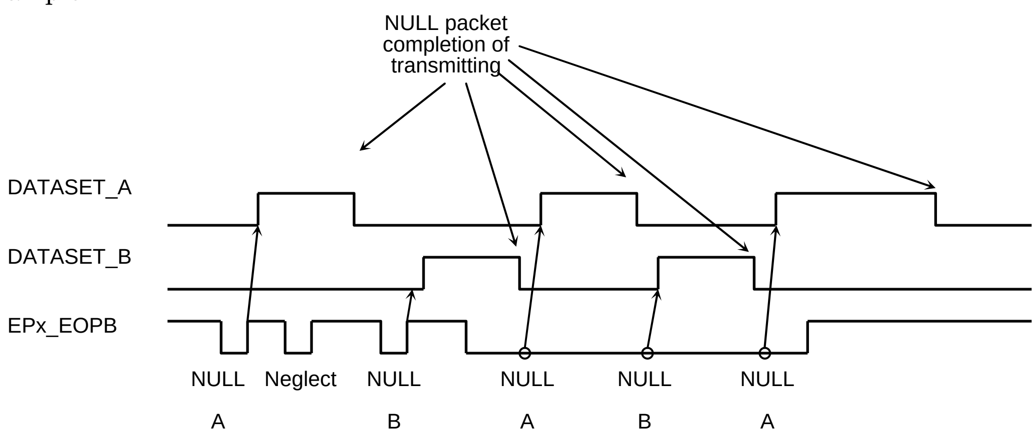
If transmitting NULL packet, by input L pulse from EPx_EOPB signal, data of 0 length is set to FIFO, and NULL packet can be transferred to IN token.

But if NULL data is set to FIFO, it is valid only in the case whole SET signal is L level condition (where FIFO is empty). If it answer to receiving IN token by using NULL packet in a certain period, it is answered by keeping EPx_EOPB signal to L level.

However, if mode is dual packet mode, EPx_DATASET signal assert L level for showing space of data. Therefore, data condition (whether either has data or not) cannot be confirmed externally.

Note: NULL packet can also be set by accessing EOP register.

Example:



(2) Interrupt control

Interrupt signal is prepared. This function use adept system.

For detail refer to 3.17.2 900/H1 CPU I/F.

3.17.8 USB Device answer

The USB controller (UDC) sets various register and initialization in the UDC in detecting of hardware reset, detecting of USB bus reset, and enumeration answer.

Each condition is explained below.

(1) bus reset detect condition.

When the UDC detects a bus reset on the USB signal line, it initializes internal register, and it prepares enumeration operation from USB host. After detecting a USB reset, the UDC sets ENDPOINT0 to control transfer type 8-byte payload and default address for using default pipe. Any endpoint other than this is prohibited.

Register name		Initial value
ENDPOINT STATUS	EP0	00H
	Except for EP0	1CH

(2) Detail of STATUS register

Status register that has been prepared for each endpoint shows the condition of each endpoint in the UDC.

Each condition affects the various USB transfers. Refer to chapter 5 for the changing conditions for each transfer type.

EPx_STATUS register value is 0 to 3, and its shows conditions are shown. 0 to 4 are the results of various transfers. It can be confirmed previous result that is transferred to endpoint by confirming from external of UDC.

0 READY
1 DATAIN
2 FULL
3 TX_ERR
4 RX_ERR

These conditions mean that the endpoint is operating normally. The meaning that is showed is different for each transfer mode. Therefore, please refer to each transfer mode column below.

ISO transfer mode

Below is the transfer condition for the previous frame. Receiving SOF renews this.

	OUT (RX)	IN (TX)
Initial	READY	READY
Not transfer	READY	FULL
Finish normally	DATAIN	READY
Detect an error	RXERR	TXERR

Transfer modes other than ISO transfer

This is the result of the previous transfer. When transfer is finished, this is renewed.

	OUT, SETUP	IN
Initial	READY	READY
Transfer finish normally	DATAIN	READY
Status stage finish	READY	READY
Transfer error	RXERR	TXERR

"Initial" is that renew RESET, USB reset, Current_Config register. In detect error, it does not generate EPx_DATASET except in toggle transfer mode and Isochronous transfer mode of interrupt.

5 to 7 in shows the status register means that the endpoint is in special condition.

5 BUSY — BUSY is generated only at endpoint of control transfer. If UDC transfer in control writes transfer, when CPU has not finished enumeration transaction, and if it receives ID of status stage from USB host, BUSY is set. STATUS is BUSY until CPU finishes enumeration transaction and EP0 bit of EOP register is written 0 in UDC. If CPU enumeration transaction finishes and EP0 bit of EOP register is written 0 and status stage from USB host finishes normally, it displays READY.

6 STALL — STALL shows that endpoint is in STALL condition.

This condition is generated if it violates protocol or error in bus enumeration. To return endpoint to normal transfer condition, USB device request is needed. This request returns to normal condition. But control endpoint returns to normal condition by receiving SETUP token. And it becomes to SETUP stage.

7 INVALID — This condition shows condition that endpoint cannot be used. UDC sets condition that isn't designated in ENDPOINT to INVALID condition, and it ignores all tokens for this endpoint. In initializing, this condition is always generated. When UDC detects hardware reset, it sets all endpoints to INVALID condition. Next, if USB reset is received, endpoint 0 only is renewed to READY. Other endpoints that are defined on disruptor are renewed if SET_CONFIG request finishes normally.

3.17.9 Power Management

USB controller (UDC) can be switched from optional resume condition (turn on the power supply condition) to suspend (Suspension) condition, and it can be returned from suspend condition to turn on the power supply condition.

This function can be set to low electricity consumption by operating CLK supplying for UDC.

(1) Switch to suspend condition

The USB host can set the USB device to suspend condition by maintaining IDLE state. The UDC switches to suspend condition by the following process.

- UDC switches to suspend condition if it detects IDLE state of more than 3 ms (about 3.07ms) on USB signal. At this point, UDC sets SUSPEND bit of STATUS register to “1”.
- UDC renews USBINTFR1<INT_SUS> and <INT_CLKSTOP> from “0” to “1” if it detects IDLE state of more than 5 ms (about 5.46ms) on USB signal. Afterward reset USBCR1<USBCLKE> to “0” to stop USB clock.
- In this condition, all register values into the UDC are kept. However, external access is not possible except for reading of STATUS register, Current_Config register, and USBINTFR1, USBINTFR2, USBINTMR1, USBINTMR2 and USBCR1.

(2) Return from suspend condition by host resume

When activity of bus on USB signal is restored by resume condition output from USB host, the UDC releases SUSPEND condition, and it resets SUSPEND bit of STATUS register to “0”. The system is thereby resumed. The resume condition output from the host is maintained for at least 20 ms. Therefore effective protocol occurs on USB signal line after this time has elapsed.

(3) Return from suspend condition by remote wakeup

Remote wakeup is system for prompt resume from suspended USB device to USB host. Some applications do not support remote wakeup. Remote wakeup is also limited using from USB host by bus enumeration.

UDC remote wakeup function can be used when it is permitted.

Setting remote wakeup by bus can be confirmed by bit7 of Current_Config register. When this bit is “1”, remote wakeup can be used. Remote wakeup is not disabled by this bit. Therefore, if this bit shows disabled, remote wakeup must not be set. If it fill the conditions, output resume condition output to USB host by writing USBCR1<WAKEUP> from “1” to “0” of UDC in suspend condition. And it prompts resume from UDC to host. After UDC changes to suspend condition, WAKEUP input is ignored for 2 ms. Therefore, remote wakeup becomes effective when USBINTFR1<INT_SUS> is set to “1”.

(4) Low power consumption by control of CLK input signal

When the UDC switches to suspend condition, it stops CLK and switches to low power consumption condition. But as system, this function enables low power consumption by stopping source of CLK. CLK that is supplied to the UDC can be controlled by using USBINTFR1<INT_SUS>, <INT_CLKSTOP> and USBCR1<USBCLKE>.

If UDC switches to suspend condition, USBINTFR1<INT_SUS> is set to “1”, and <INT_CLKSTOP> is set to “1”. After confirmation, stop CLK supply (USBCLK) by setting “0” to USBCR1<USBCLKE>. If SUSPEND condition is released by resuming from host, supply normal CLK to UDC within 3 ms.

When remote wakeup is used, it is necessary to supply a stable CLK to the UDC before use. When doubler circuit is used as generation source, the above control is needed.

- Return from suspend condition by USB reset (by INT_CLKON interrupt)

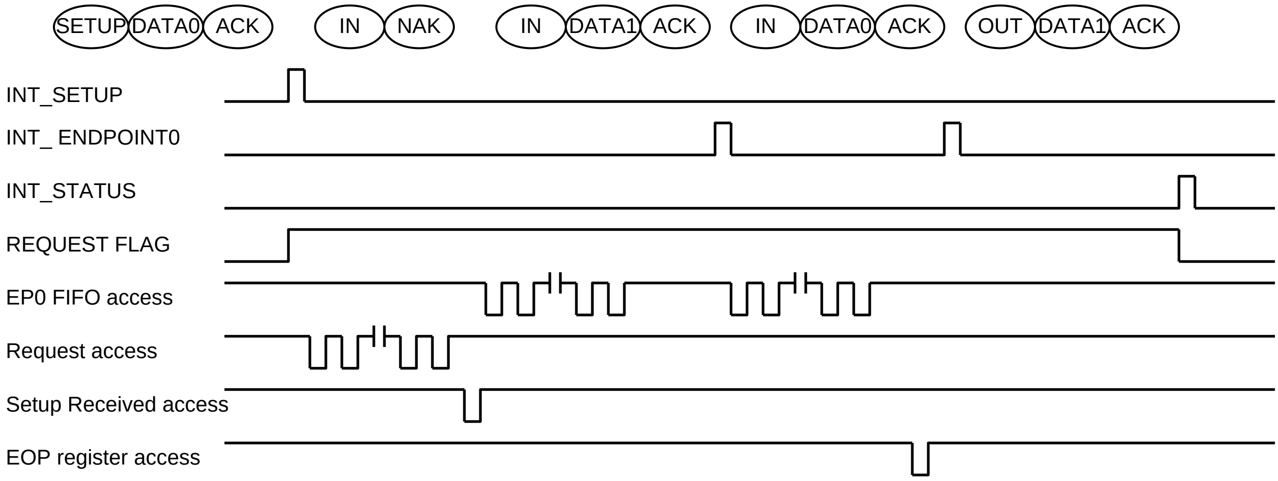
  When UDC stops CLK in suspend condition, UDC can not detect USB reset and control CLK in suspend condition as above mentioned.

  In case CLK is stopped in suspend condition, UDC can detect USB reset and return from suspend condition by supplying CLK (USBCR1<USBCLKE>=1) after detecting INT_CLKON interrupt.

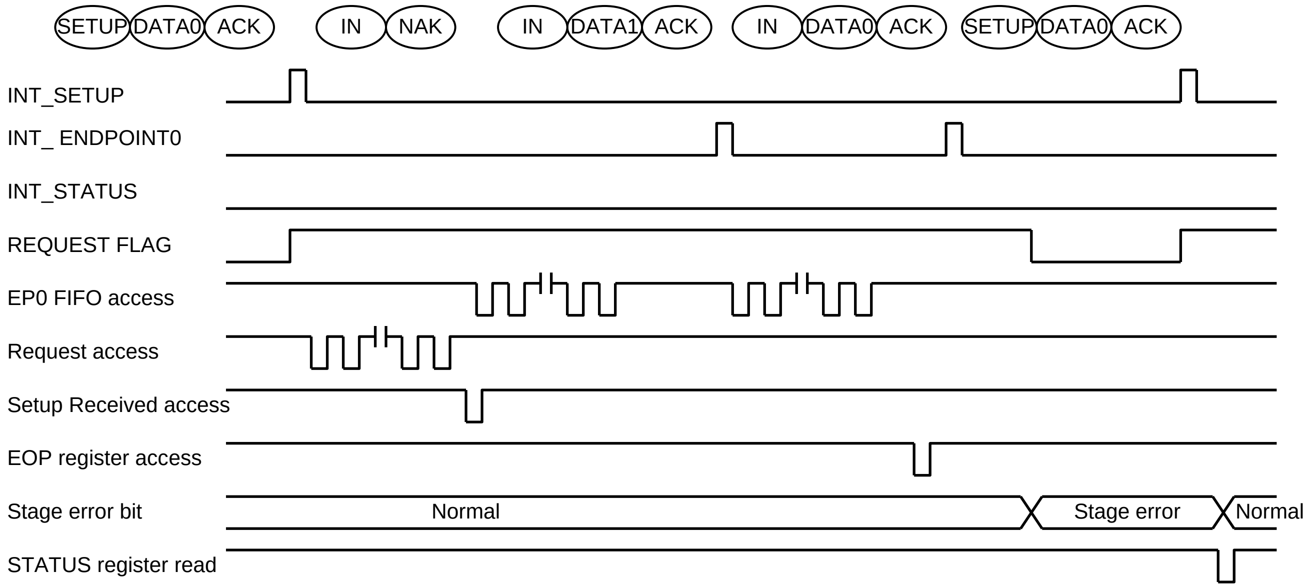
3.17.10 Supplement

(1) External access flow to USB communication

a) Normal movement



b) Stage error



(2) Register Initial value

Register Name	Initial Value OUTSIDE Reset	Initial Value USB_RESET
bmRequestType	0x00	0x00
bRequest	0x00	0x00
wValue_L	0x00	0x00
wValue_H	0x00	0x00
wIndex_L	0x00	0x00
wIndex_H	0x00	0x00
wLength_L	0x00	0x00
wLength_H	0x00	0x00
Current_Config	0x00	0x00
Standard request	0x00	0x00
Request	0x00	0x00
DATASET	0x00	0x00
Port Status	0x18	Hold
Standard request mode	0x00	Hold
Request mode	0x00	Hold

Register Name	Initial Value OUTSIDE Reset	Initial Value USB_RESET
INT control	0x00	0x00
USBBUFF_TEST	0x00	Hold
USB state	0x01	0x01
EPx_MODE	0x00	0x00
EPx_STATUS	0x1C	0x1C
EPx_SIZE_L_A	0x88	0x88
EPx_SIZE _L_B	0x08	0x08
EPx_SIZE_H_A	0x00	0x00
EPx_SIZE_H_B	0x00	0x00
FRAME_L	0x00	0x00
FRAME_H	0x02	0x02
ADRESS	0x00	0x00
EPx_SINGLE	0x00	Hold
EPx_BCS	0x00	Hold
ID_STATE	0x01	0x00

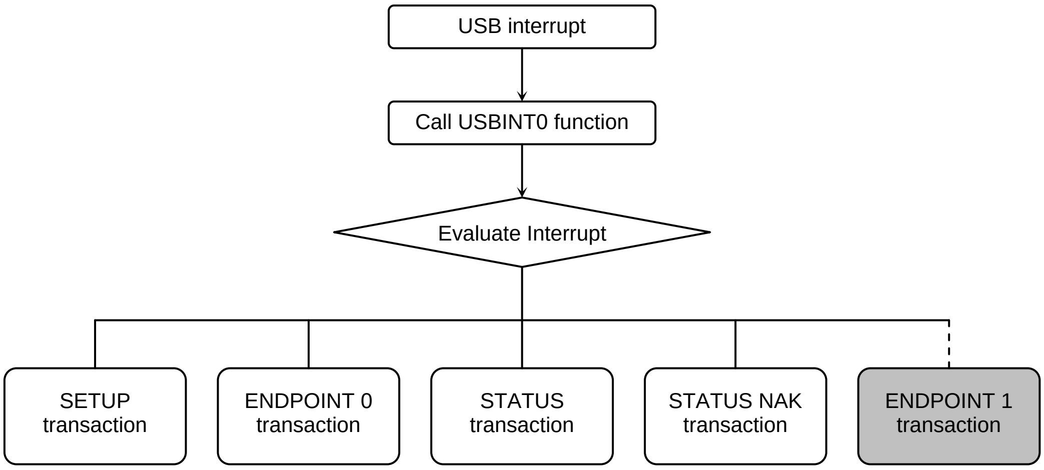
Note 1: The above initial value is the value that is initialized by external reset, USB_RESET. This value may differ from that displayed depending on conditions.
Please refer to register configure in chapter 2.

Note 2: EP0_STATUS register is initialized to 0x00 after USB_RESET is received.

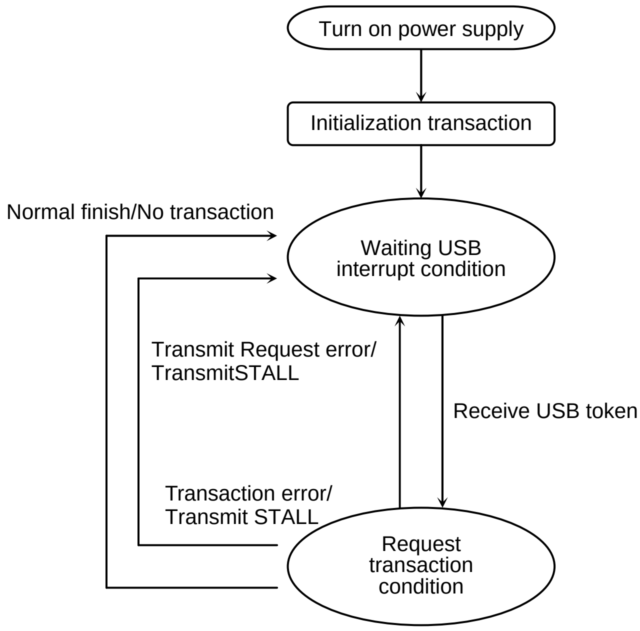
Note 3: Initial value of ID_STATE register is initialized by external reset, BRESET. When USB_RESET signal is received from host, it isinitialized to 0x00.

(3) USB control flow chart

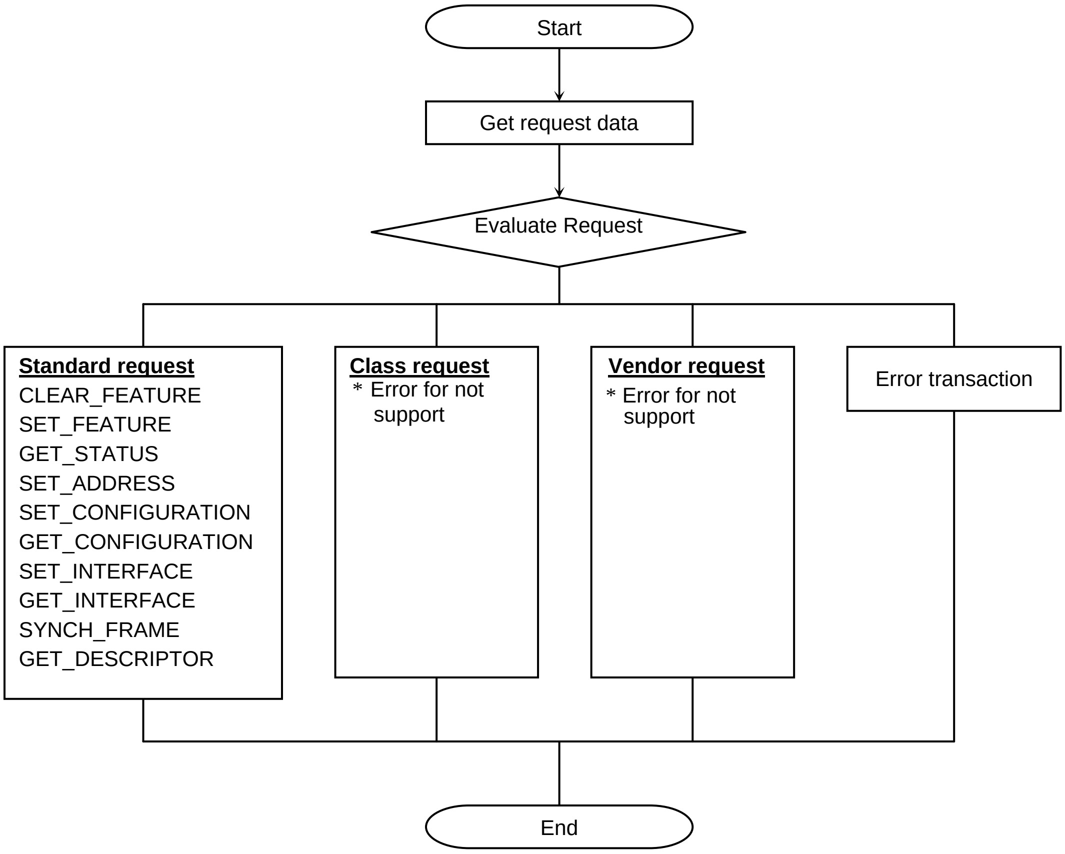
(a) Transaction for standard request (Outline flowchart (Example))



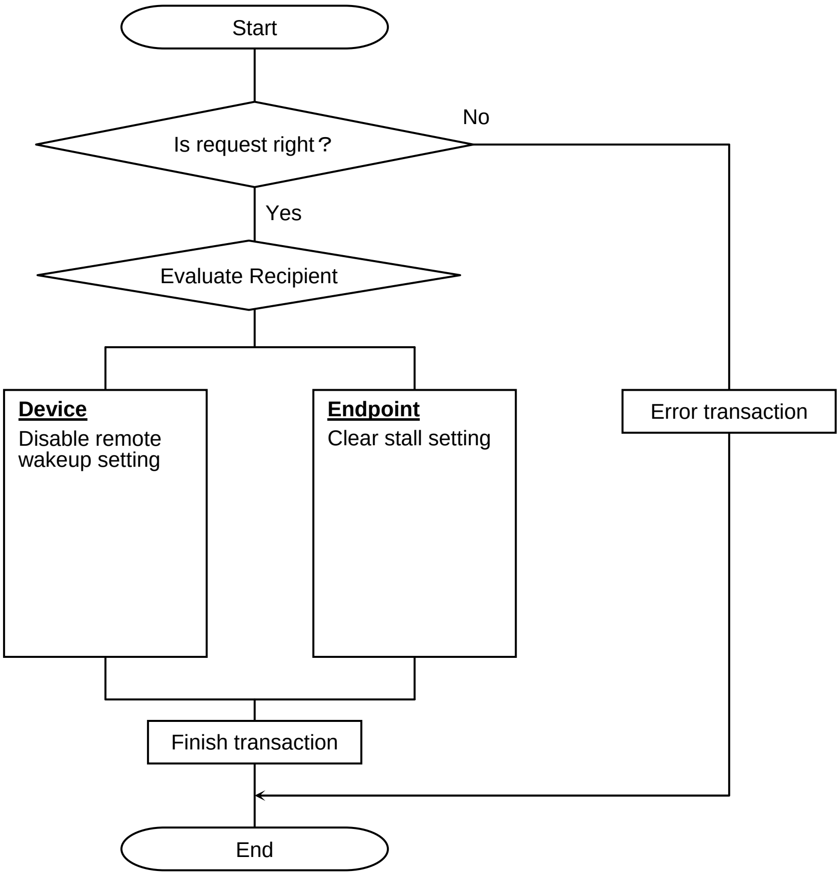
(b) Condition change



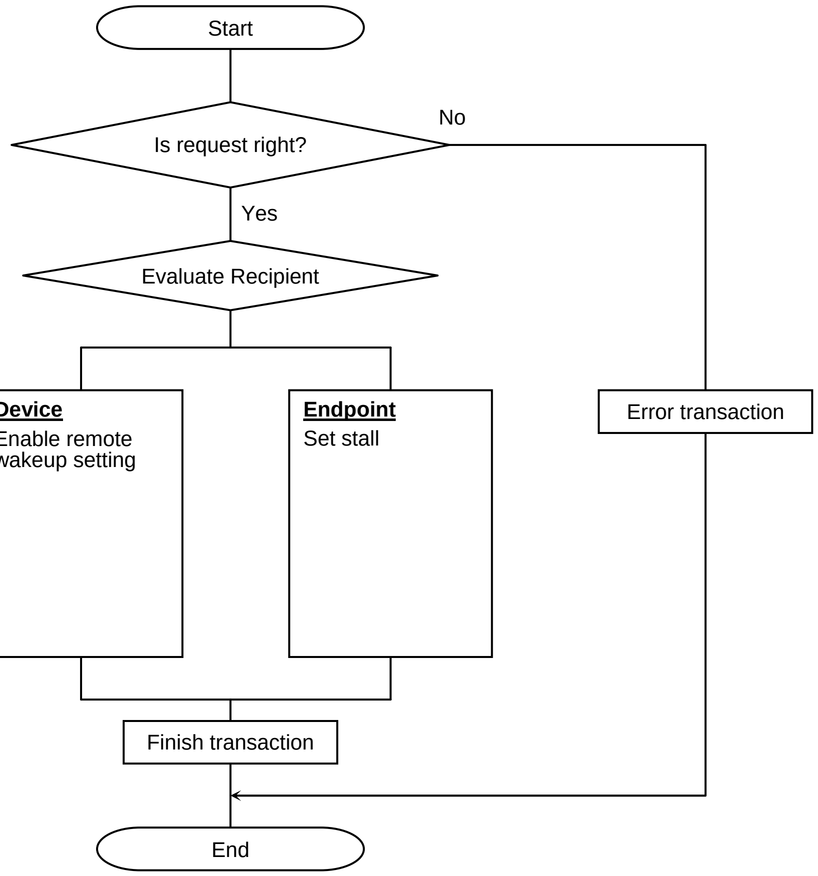
(c) Device request and evaluation of various requests



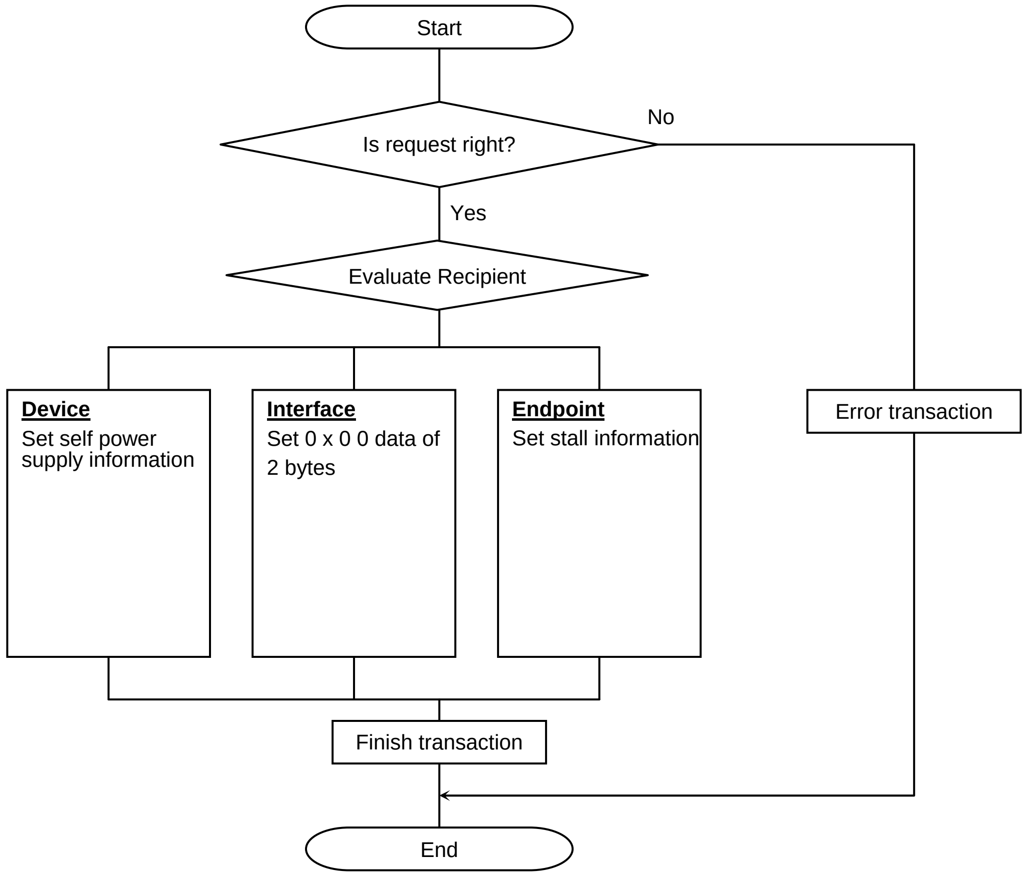
(c-1) CLEAR_FEATURE request transaction



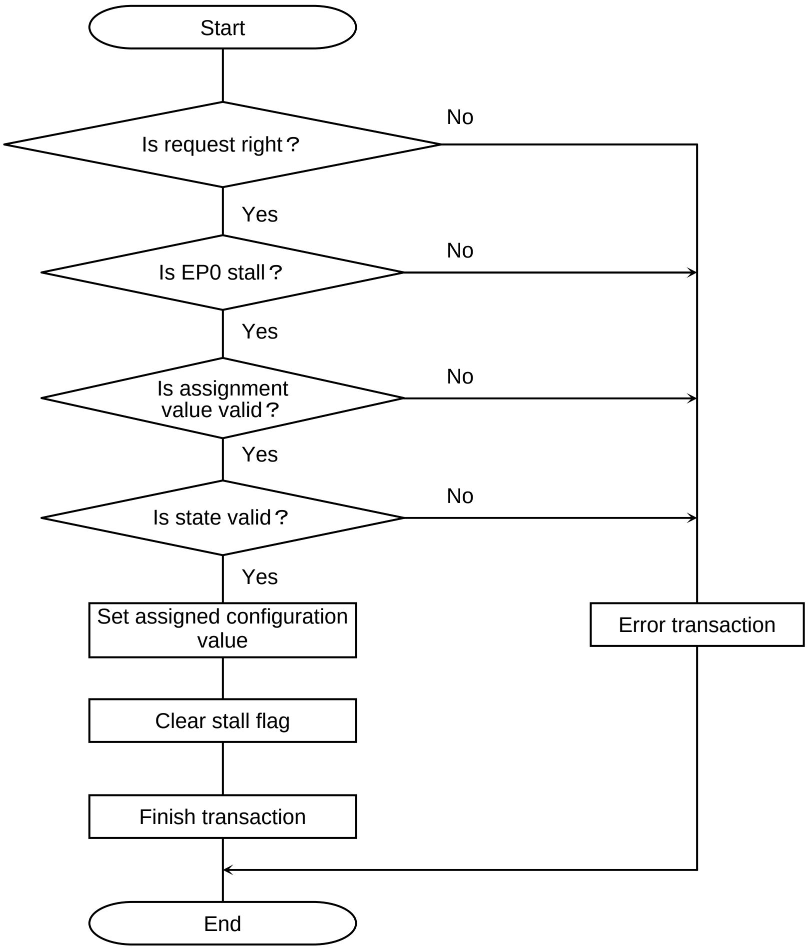
(c-2) SET_FEATURE request transaction



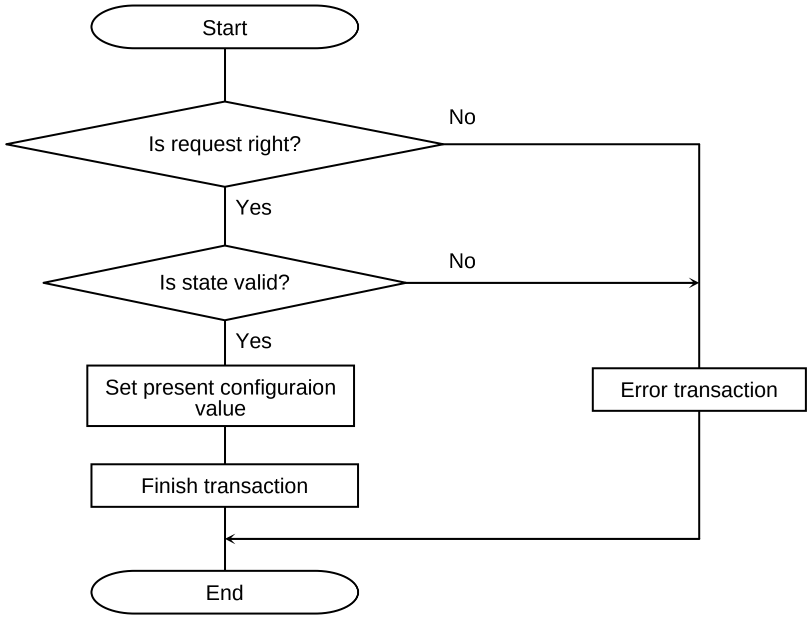
(c-3) GET_STATUS request transaction



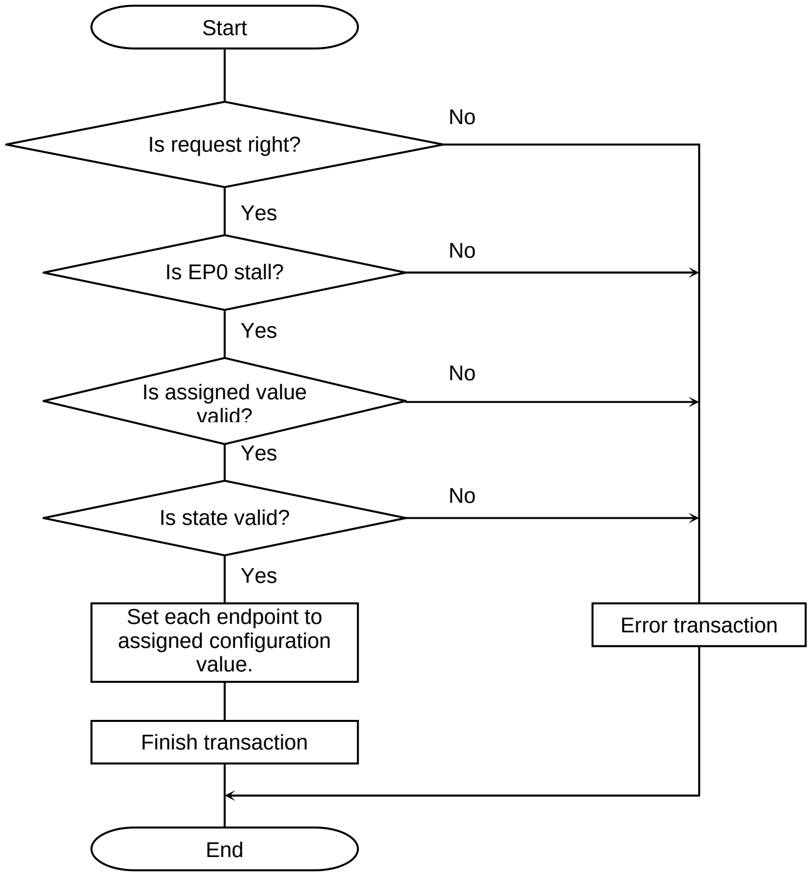
(c-4) SET_CONFIGRATION request transaction



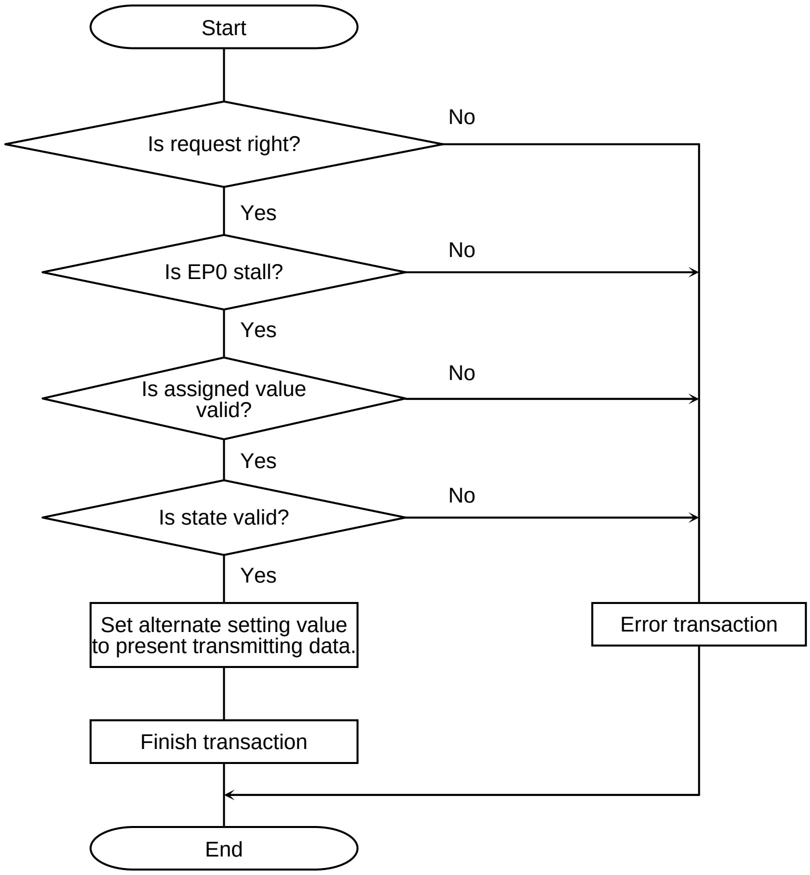
(c-5) GET_CONFIGRATION request transaction



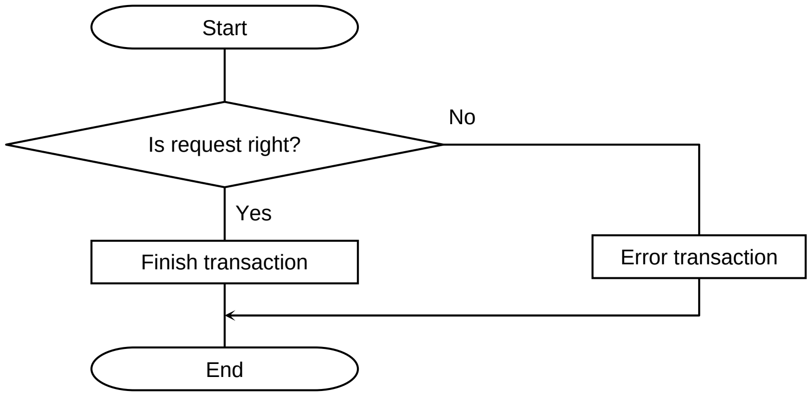
(c-6) SET_INTERFACE request transaction



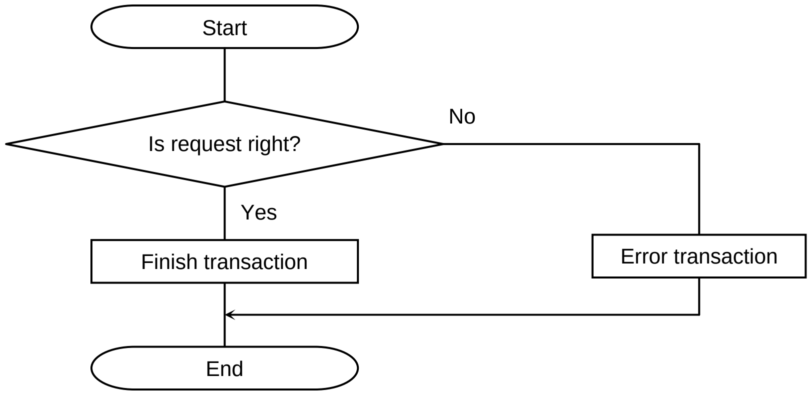
(c-7) SYNCH_FRAME request transaction



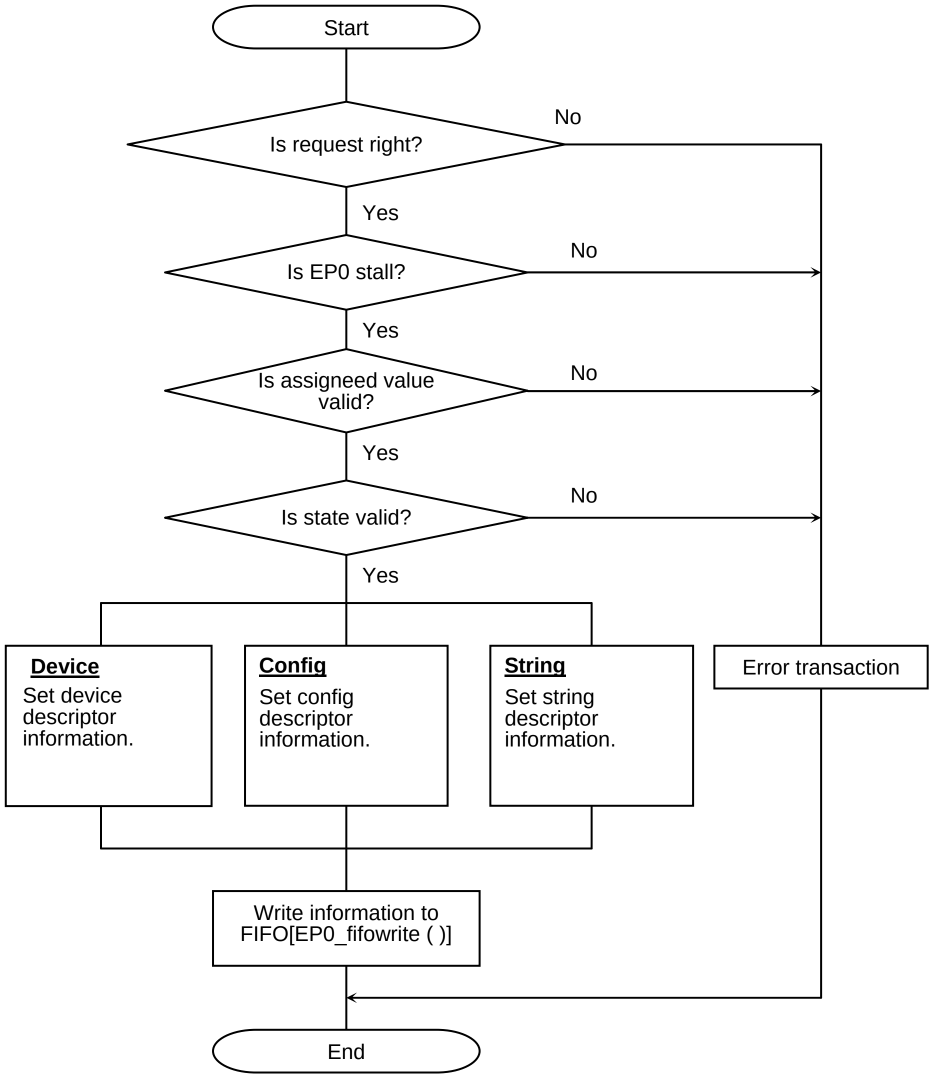
(c-8) SYNCH_FRAME request transaction



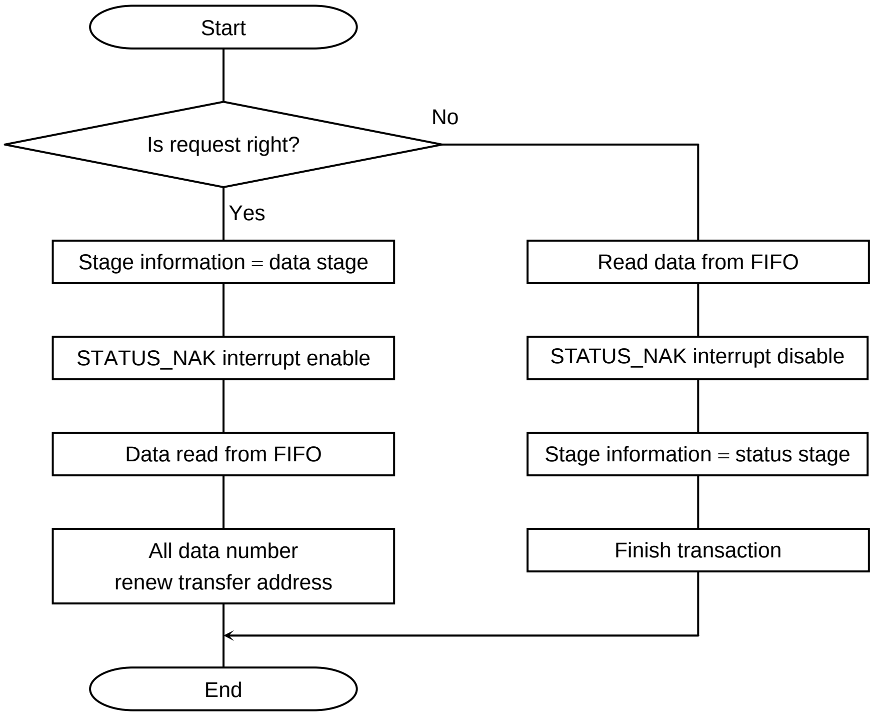
(c-9) SET_DESCRIPTOR request transaction



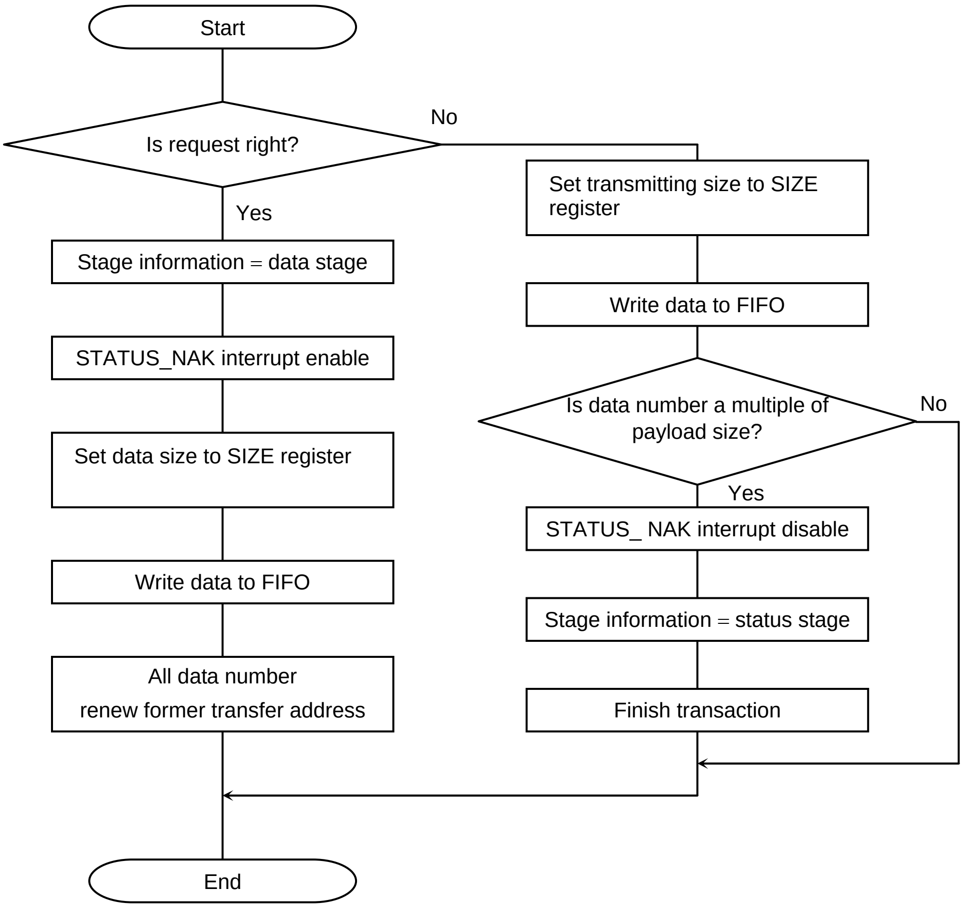
(c-10)GET_DESCRIPTOR request transaction



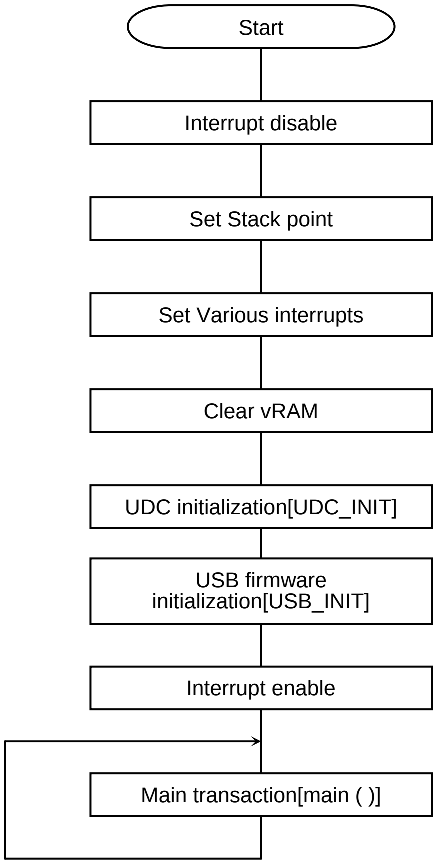
(c-11) Data read transaction to FIFO by EP0



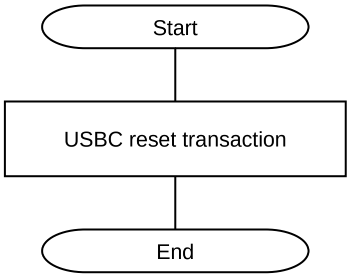
(c-12)Data write transaction to FIFO by EP0



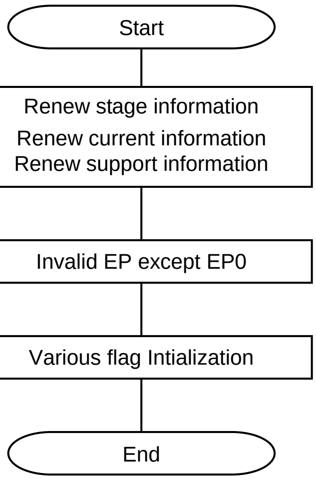
(c-13)Initial setting transaction of microcontroller



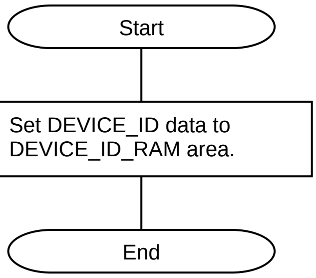
(c-14)Initial setting transaction of UDC



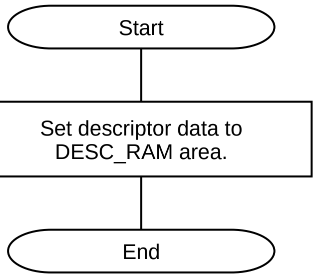
(c-15)Initial transaction of USB number changing firmware



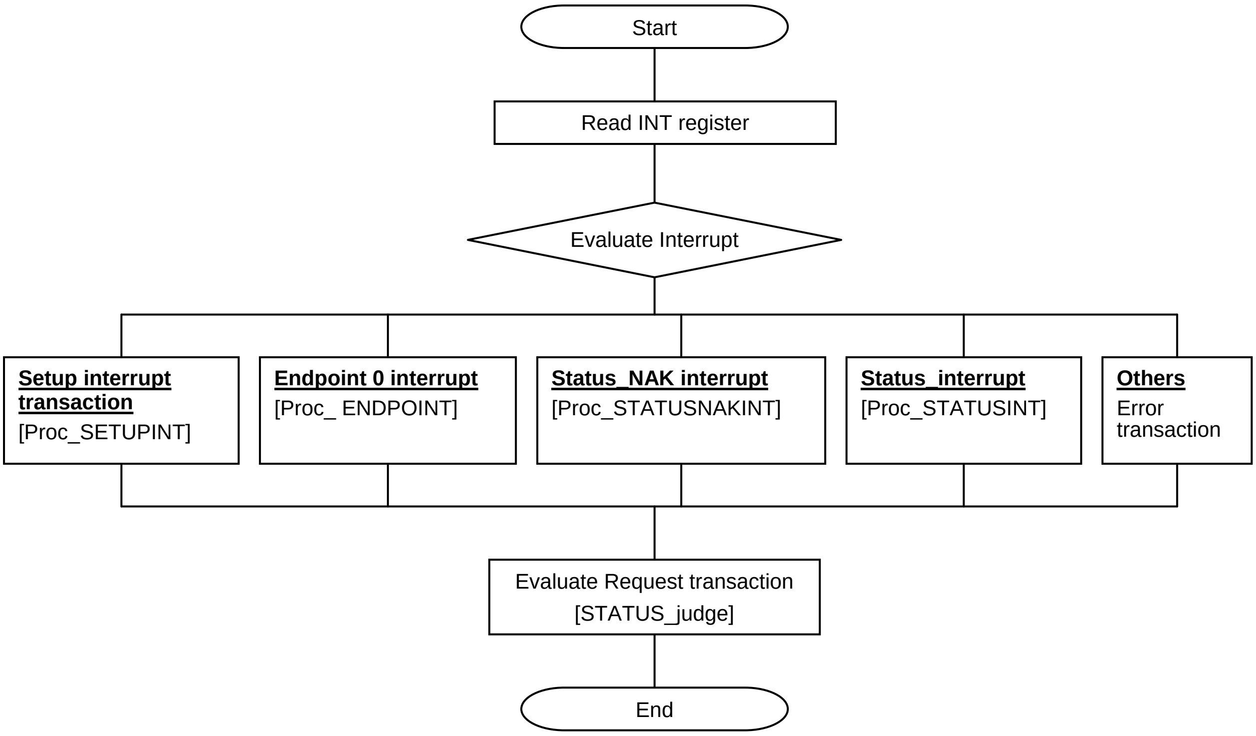
(c-16)Set DEVICE_ID data to DEVICE_ID of UDC



(c-17)Descriptor data set transaction



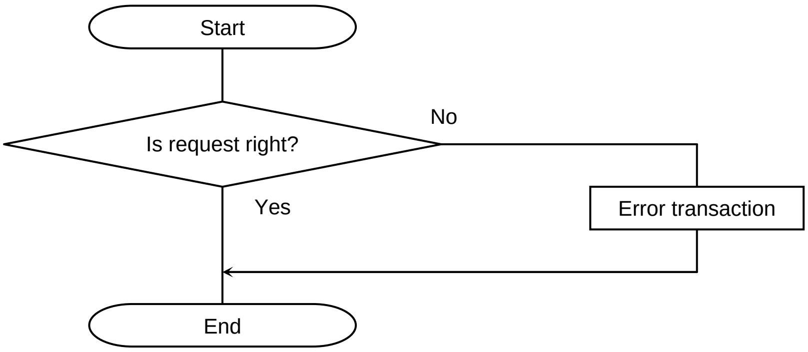
(c-18)USB interrupt transaction



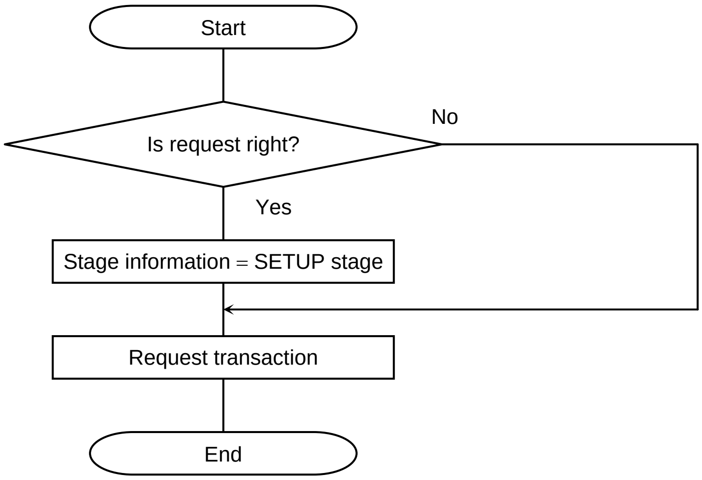
(c-19)Dummy function for not using maskable interrupts.

- Transaction performs nothing, therefore outline flow is skipped.

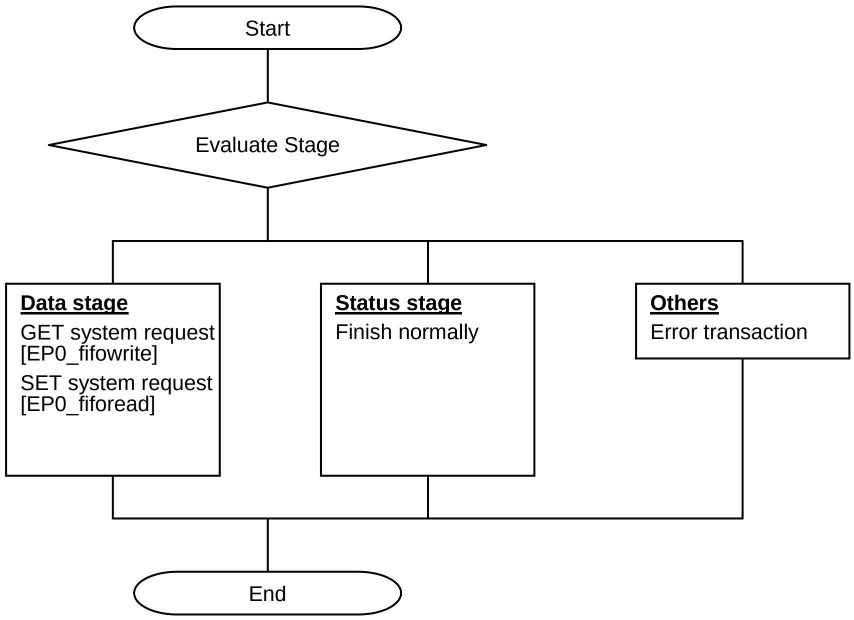
(c-20)Request evaluation transaction. If transaction result is error, it initiates STALL command.



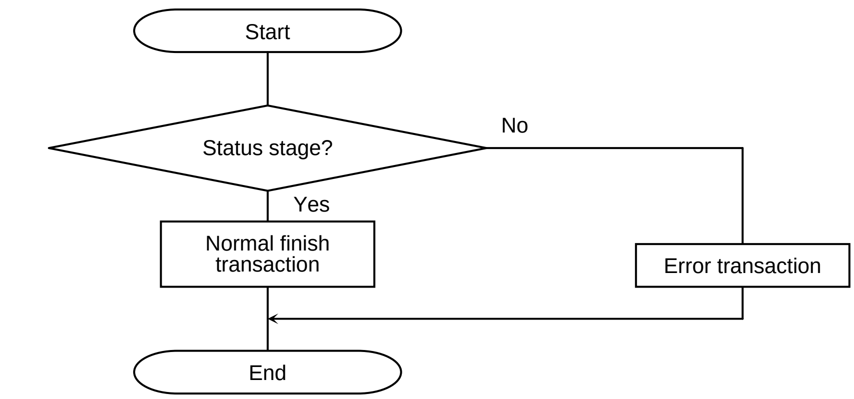
(c-21)SETUP stage transaction



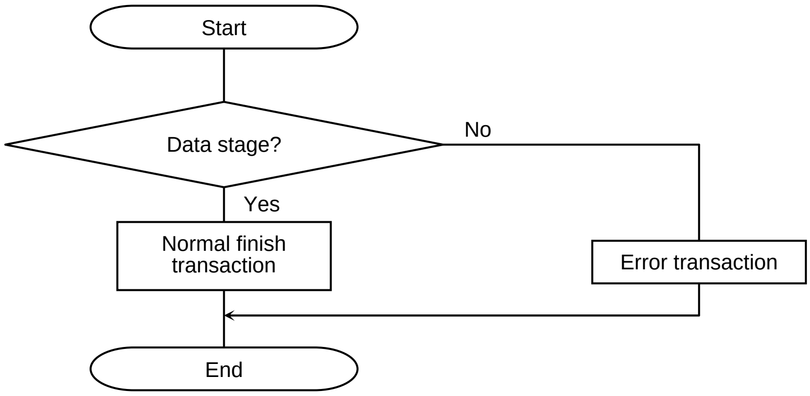
(c-22)Perform endpoint 0 transaction except in SETUP stage.



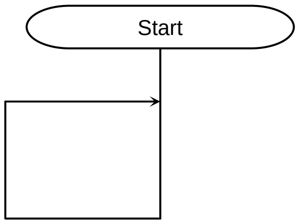
(c-23)Status stage interrupt transaction



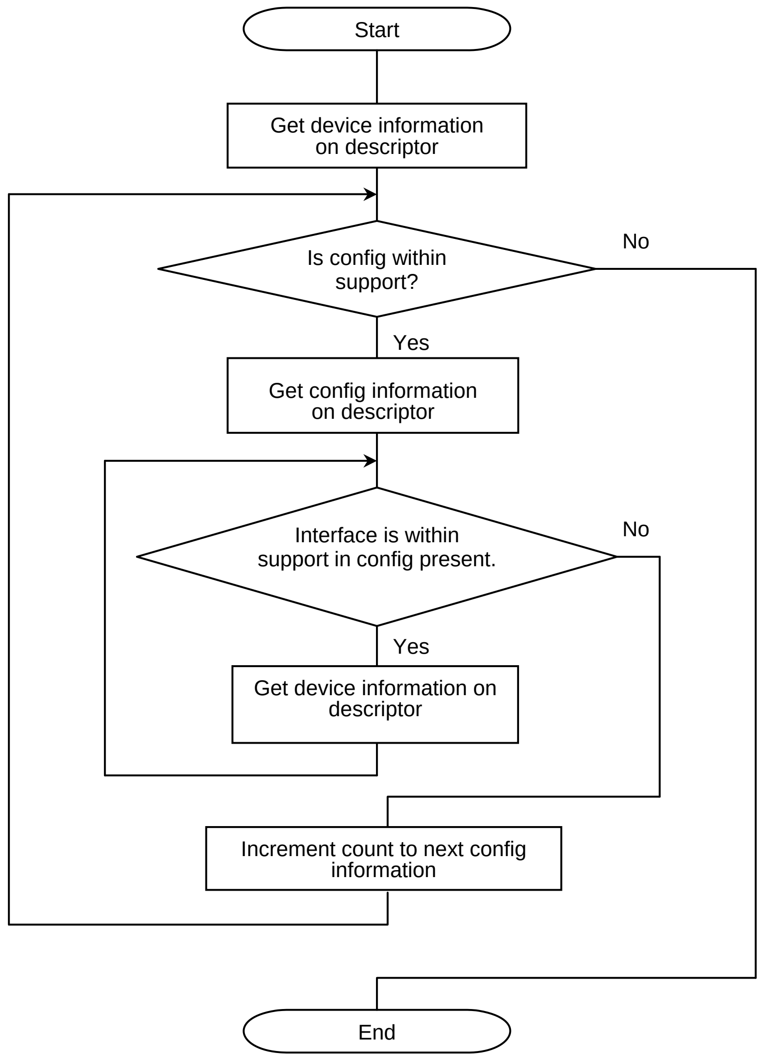
(c-24)STATUS_NAK interrupt transaction



(c-25)This transaction is a non-transaction for USB interrupts.



(c-26)Getting descriptor information (related to standard request)



3.17.11 Notice and Restrictions

1. When using the USB device controller in the TMP92CF29A, a crystal oscillator is recommended (USB standard ≤ 10 MHz±2500ppm). In this case, a maximum of 3 stages of external hub can be due to the precision of this USB device controller and the internal clock. If USB compliance (USB logo) is needed, the 5 stages connection is needed for external hub. And it is needed that input 48MHz clock from X1USB pin (USB standard ≤ ±2500ppm.)

2. Precaution for using the USB dual packet mode in the TMP92CF29A

   In the dual packet mode, each FIFO is divided into two independent packets (A and B) to be controlled alternately by hardware.

   When reading data from a receive FIFO, it is necessary to check the state of the two packets to determine which packet should be processed first. At this time, the following precaution is required.

   The EPx_SIZE register that indicates the presence of valid data is provided separately for packets A and B. The CPU is required to check the respective PKT_ACTIVE bits to determine which packet was accessed first and then to know the number of data in this packet. The packet with its PKT_ACTIVE bit set to “1” is the packet which was received first.

   In determining whether only packet A is active, only packet B is active, or both packets A and B are active, if the respective PKT_ACTIVE bits are read sequentially, the state of each bit may change between each read. If this happens, the packets may not be processed in proper order.

   Therefore, the PKT_ACTIVE bit information in the EPx_SIZE register should be captured and saved in another location such as RAM by using an interrupt request. Then, use this saved information to perform branch processing.

3.18 SPI Controller (SPIC)

The SPIC is a Serial Peripheral Interface Controller that supports only master mode.

It can be connected to the SD card, MMC (Multi Media Card) etc. in SPI mode.

Its features are summarized as follows:

1) On-chip 32-byte FIFOs for both transmission and reception

2) Generates the CRC-7 and CRC-16 values for transmission and reception

3) Baud Rate: 20 Mbps (max)

4) Can be connected to multiple SD cards and the MMC. (Since there is only on chip select signal preassigned as $\overline{\text{SPCS}}$, use other output ports to allow for more than two connections.)

This device has 1 channel SPI circuit. It shared with PR0~PR3 pins. However, it is possible also that it assign SPI function to PC4 ~PC6 pins.

5) Operates as the general synchronous SIO

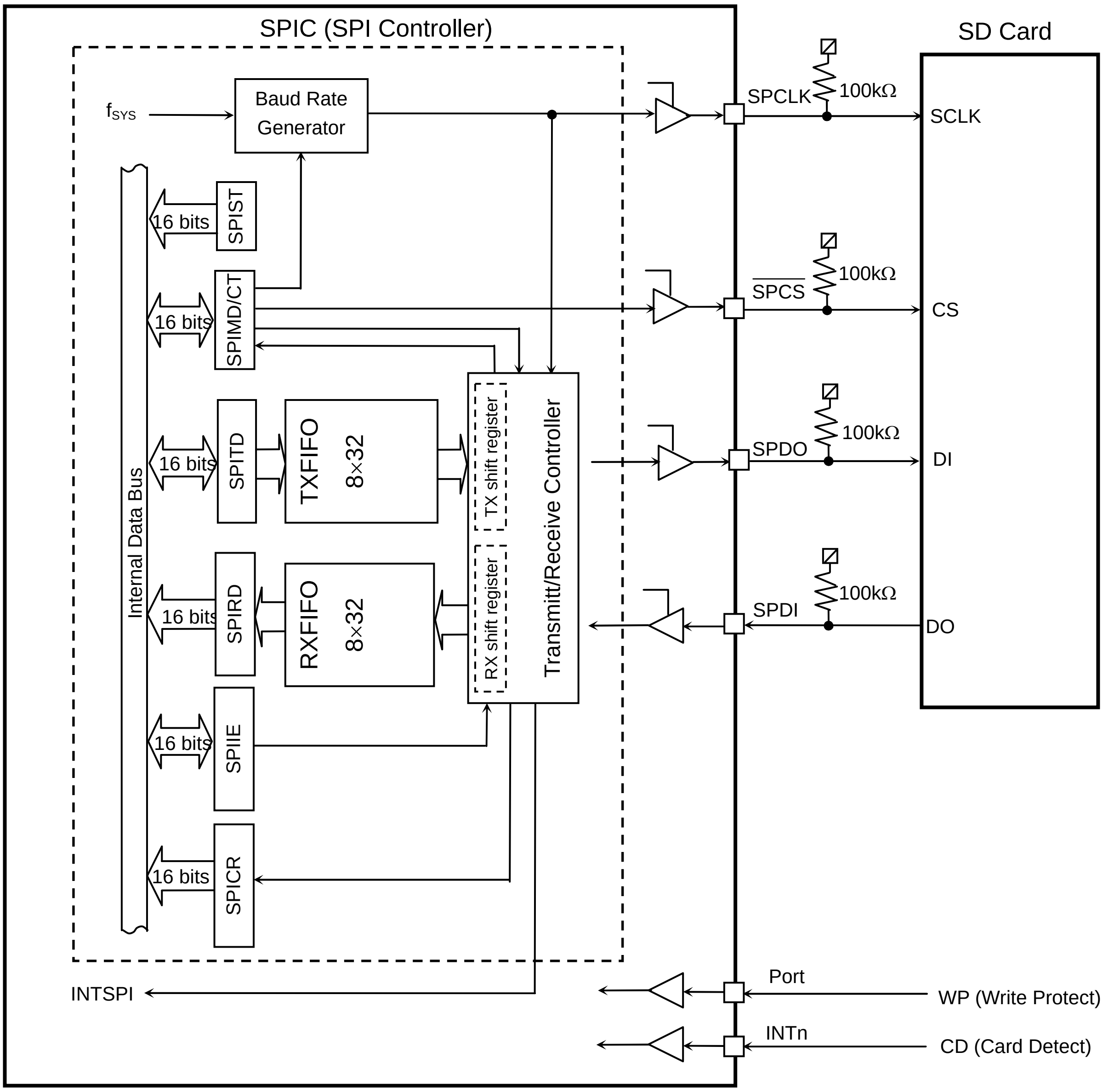
Selects the followings: MSB/LSB-first, 8/16-bit data length, rising/falling edge

6) Two types of interrupts: INTSPITX (Transmit interrupt), INTSPIRX (receive interrupt)

Select Read/Mask for interrupts: RFUL, TEMP, REND and TEND

3.18.1 Block Diagram

Figure 3.18.1 shows a block diagram of the SPIC and its connections with a SD card.



Note 1: The SPCLK, $\overline{\text{SPCS}}$, SPDO and SPDI pins are configured as input ports (Ports PR3, PR2, PR1 and PR0) upon reset.
Thus, these pins require pull-up resisters to fix their voltage levels. The pull-up resistor values should be adjusted under real-world conditions.

Note 2: Any one of general inputs and interrupt should be used as the WP (Write Protect) and CD (Card Detect) inputs, respectively.

Figure 3.18.1 Block Diagram and Connection Example

3.18.2 Special Function Registers (SFRs)

This section describes the SFRs of the SPIC. These are connected to the CPU with 16 bit data buses.

(1) SPIMD (SPI Mode Select register)

The SPIMD register specifies the operating mode, clock operation, etc.

SPIMD Register

SPIMD (0820H) A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit Symbol	SWRST	XEN				CLKSEL2	CLKSEL1	CLKSEL0
Read/Write	W	R/W				R/W		
Reset State	0	0				1	0	0
Function	Software Reset 0: Don't care 1: Reset	SYSCK 0: Disable 1: Enable				Select Baud Rate(Note1) 000: Reserved 001: $f_{SYS}/2$ 010: $f_{SYS}/3$ 011: $f_{SYS}/4$	100: $f_{SYS}/8$ 101: $f_{SYS}/16$ 110: $f_{SYS}/64$ 111: $f_{SYS}/256$	

(0821H)

	15	14	13	12	11	10	9	8
Bit Symbol	LOOPBACK	MSB1ST	DOSTAT		TCPOL	RCPOL	TDINV	RDINV
Read/Write	R/W				R/W			
Reset State	0	1	1		0	0	0	0
Function	LOOPBACK Test Mode 0:Disbale 1:Enable	Start Bit for Transmission / Reception 0: LSB 1: MSB	SPDO Pin State When Not Transmitting 0: Fixed to "0" 1:Fixed to "1"		Synchronization Clock Edge Select for Transmission 0: Falling edge 1: Rising edge	Synchronization Clock Edge Select for Reception 0: fall 1: rise	Data Inversion for Transmission 0: Disable 1: Enable	Data Inversion for Reception 0: Disable 1: Enable

Note: The SD card of the TMP92CF29A supports a baud rate of up to 20 Mbps in SPI mode. The baud rate should be adjusted with the operating frequency of the CPU (f_{SYS}) so that it does not exceed 20 MHz.

Figure 3.18.2 SPIMD Register

(a) LOOPBACK

Setting the XEN and LOOPBACK bits to 1 enables the internal SPDO output to be internally connected to the SPDI input. This setup can be used for testing.

Also, a clock signal is generated from the SPCLK pin, regardless of whether data transmission or receptionis in progress.

Data transmission or reception must not be performed while changing the state of this bit.

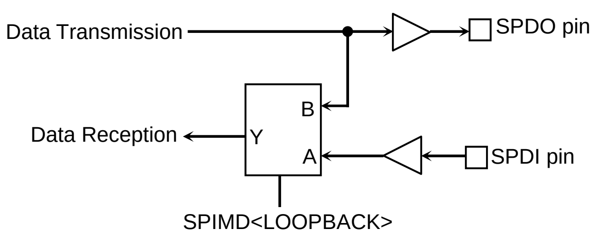


Figure 3.18.3 LOOPBACK Bit Configuration

(b) MSB1ST

This bit specifies whether to transmit/receive byte with the MSB first or with the LSB first. Data transmission or reception must not be performed while changing the state of this bit.

(c) DOSTAT

This bit specifies the status of the SPDO pin of when data transmission is not performed (i.e., after completing data transmission or during data reception). Data transmission or reception must not be performed while changing the state of this bit.

(d) TCPOL

This bit specifies the polarity of the active edge of the synchronization clock for data transmission.

The XEN bit should be cleared to “0” for changing the state of this bit. At the same time, RCPOL should also be cleared to “0”.

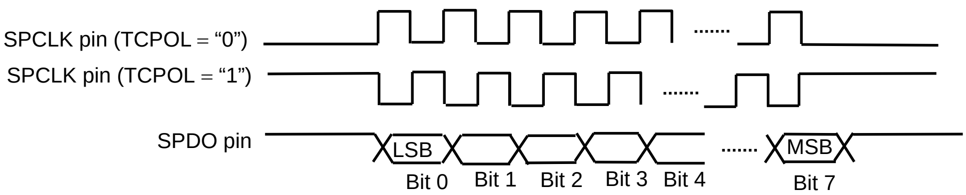


Figure 3.18.4 Timing Diagram of Data Transmissions Controlled by the TCPOL Bit

(e) RCPOL

This bit specifies the polarity of the active edge of the synchronization clock for data reception.

The SPIMD<XEN> bit should be cleared to “0” for changing the state of this bit. TCPOL should also be cleared to “0”.

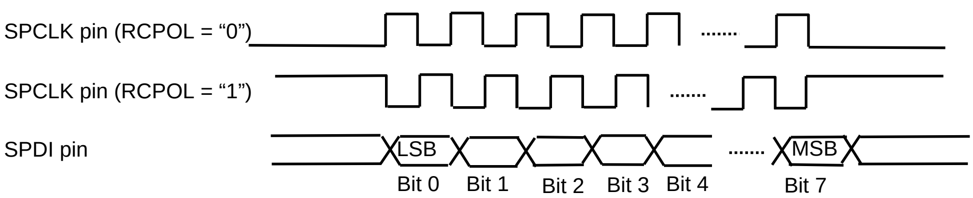


Figure 3.18.5 Timing Diagram of Data Receptions Controlled by the TCPOL Bit

(f) TDINV

This bit specifies whether to logically invert the data transmitted from the SPDO pin or not. Data transmission or reception must not be performed while changing the state of this bit.

(g) RDINV

This bit specifies whether to logically invert the data received from the SPDI pin or not. Data transmission or reception must not be performed while changing the state of this bit.

(h) SWRST

This bit is used to performs a software reset of the read and write pointers for data transmission and reception. Stop the data transmission after writing a "0" to the SPICT<TXE> bit where XEN = "1". Then, write a "1" to the SWRST bit to initialize the read and write pointers of transmit and receive FIFO buffers.

Writing a "0" to the SPICT<TXE> bit stops data transmission after transmitting the UNIT data that is currently being transmitted. Then, writing a "1" to the SWRST bit invalidate the data in the transmit FIFO buffer. Therefore, the data is not output even if the data transmission is restarted after performing a software reset. Do not write a "1" to the SWRST bit in the middle of data transmission.

In case of performing data reception, the received data contained in the receive FIFO buffer becomes invalid.

However, when performing Sequential-mode data reception, data reception continues even if the data in the receive FIFO buffer becomes invalid. Therefore, stop data reception by writing a "0" to the SPICT<RXE> bit after receiving the data that is currently being received. Then, (after confirming there is no UNIT data currently being received, or) the receive operation can be stopped completely by writing a "1" to the SWRST bit after checking no UNIT data in receiving (namely after REND interrupt or the time to receive 1UNIT).

Do not write a "1" to the SWRST bit during a data reception. Software reset can be performed in a single-shot operation, which is to write a "1" to the SWRST bit (it is not required to write a "0" to the SWRST bit). Simultaneous writing of 1s to the XEN and SWRST bits is also supported.

(i) XEN

This bit enables or disables the internal clock signal. Always set this bit to "1" when using the SPI controller.

(j) CLKSEL2:0

This bit selects the baud rate. The baud rate is generated using the system clock f_{SYS} and is programmable as shown below according to the system clock settings.

Data transmission or reception must not be performed while changing the state of these bits

Note: The SD card of the TMP92CF29A supports a baud rate of up to 20 Mbps. This field should be programmed so that SPCLK signal does not exceed 20 MHz When setting the baud rates, select less than 20 Mbps according to the operation speed of CPU (f_{SYS}).

Table 3.18.1 Example of Baud Rate

<CLKSEL2:0>	Baud Rate [Mbps]	
	f_{SYS} = 60 MHz	f_{SYS} = 80 MHz
f_{SYS}/2	–	–
f_{SYS} /3	20	–
f_{SYS} /4	15	20
f_{SYS} /8	7.5	10
f_{SYS} /16	3.75	5
f_{SYS} /64	0.9375	1.25
f_{SYS} /256	0.234375	0.3125

(2) SPI Control Register (SPICT)

The SPICT register specifies data length, CRC, etc.

SPICT Register

		7	6	5	4	3	2	1	0
SPICT (0822H)	Bit Symbol	CEN	SPCS_B	UNIT16	TXMOD	TXE	FDPXE	RXMOD	RXE
	Read/Write	R/W							
	Reset State	0	1	0	0	0	0	0	0
	Function	Communicati-on Control 0: Disable 1: Enable	$\overline{SPCS}$ Pin Control 0: Set to "0" 1: Set to "1"	Data Length Select 0: 8 bits 1: 16 bits	Transmit Mode Select 0: UNIT 1: Sequential	Transmission Enable 0: Disable 1: Enable	Alignment Enable in Fullduplex mode 0: Disable 1: Enable	Receive Mode Select 0: UNIT 1: Sequential	Receive Enable 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(0823H)	Bit Symbol	CRC16_7_B	CRCRX_TX_B	CRCRESET_B					
	Read/Write	R/W							
	Reset State	0	0	0					
	Function	CRC Select 0: CRC7 1: CRC16	CRC Data 0: Transmit 1: Receive	CRC Calculation Register Control 0: Reset 1:Reset Release					

Figure 3.18.6 SPICT Register

(a) CRC16_7_B

This bit selects the CRC calculation algorithm from the CRC7 and CRC16.

(b) CRCRX_TX_B

This bit selects the data to be sent to the CRC generator. When CRCRX_TX_B = "0", the CRC calculation is performed on the transmit data. Otherwise, it is performed on the received data.

(c) CRCRESET_B

This bit is used to initialize the CRC calculation register.

This section describes how to calculate the CRC16 of the transmit data and to append the calculated CRC value at the end of the transmit data. Figure 3.18.7 below illustrates the flow chart of the CRC calculation procedures.

(1) Program the SPICT<CRC16_7_B> bit to select the CRC algorithm from CRC7 and CRC16. Then, also program the CRCRX_TX_B bit to specify the data on which the CRC calculation is performed.

(2) To reset the SPICR register, write a “0” to the CRCRESET_B bit and then write a “1” to the same bit.

(3) Load the SPITD register with the transmit data, and wait until transmission of all data is completed.

(4) Read the SPICR register and obtain the result of the CRC calculation.

(5) Transmit the CRC obtained in step (4) in the same way as step (3).

The CRC calculation on the receive data can be performed in the same procedures.

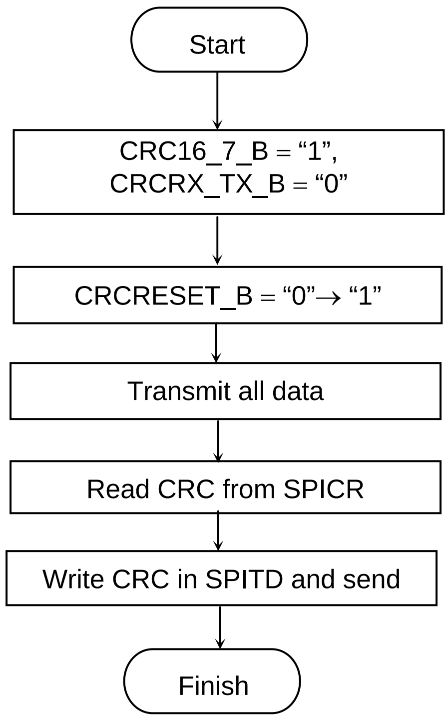


Figure 3.18.7 Flow Chart of the CRC Calculation Procedures

(d) CEN

This bit enables or disables the pins for the SD card and MMC connections.

When the card is not inserted or when it is not powered on, a shoot through current might flow in the SPDI pin, for it enters the floating state. Also, currents may unintentionally flow into the card from the $\overline{\text{SPCS}}$, SPCLK and SPDO pins when they generate a logic 1. This bit can be used to avoid these problems.

If write <CEN> to "0" with PRCR and PRFC selecting $\overline{\text{SPCS}}$, SPCLK, SPDO and SPDI signal, SPDI pin is prohibited to input (avoiding penetrated current) and $\overline{\text{SPCS}}$, SPCLK, SPDO pin become high impedance.

When writing a "1" to the CEN bit, ensure that a card is properly inserted and powered on, as well as that the clock signal is supplied to the SPIC (SPIMD<XEN> = "1").

(e) SPCS_B

This bit specified the logic state of the $\overline{\text{SPCS}}$ output.

(f) UNIT16

This bit selects the data length for transmission and reception. The data length is hereafter referred to as the UNIT. Data transmission or reception must not be performed while changing the state of this bit

(g) FDPXE

This bit should be set to "1" when performing the full-duplex communication. This bit specifies whether to align the transmit and receive data on the UNIT-size boundaries.

Data transmission or reception must not be performed while changing the state of this bit.

(h) TXMOD

This bit selects the data transmission mode from UNIT and Sequential modes. During transmission, it is prohibited to change the transmission mode from Sequential to UNIT, or vice versa.

For UNIT-mode transmission, the transmit FIFO buffer is disabled. The TEMP interrupt is generated when the data is loaded from the transmit data register (SPITD) to the transmit shift register.

For sequential-mode transmission, the 32-byte FIFO is enabled. The TEMP interrupt is generated when the empty space of the FIFO becomes 16 bytes or 32 bytes.

(i) TXE

This bit enables or disables data transmission. Data transmission is started when this bit set to "1" after loading the transmit data into the transmit FIFO, or when loading the transmit data to the transmit FIFO when this bit is already set to "1". The state of this bit can be changed even during data transmission. If this bit is cleared to 0 during a data transmission, the transmission is stopped after completing the transmission of the UNIT data currently being transmitted.

Important Note:

When in UNIT mode (TXMOD = “0”), the following restriction is imposed on the system operation.

When the SPICT<TEX> bit is set to “1”, the state of any bits must not be changed until the data transmission is completed.

Sample Program 1:

```
        LD       (SPITDx), A    ; Load the transmit data
        DI                      ; Disable the interrupt
        SET 3,   (SPICT)        ; Start transmission by setting the TXE bit to “1”

Wait:   BIT 1,   (SPIST)        ;  Wait for the completion of the transmission
        JPZ,     Wait
        RES 3,   (SPICT)        ; Disable the transmission by clearing the TXE bit to “0”
        EI                      ; Enable the interrupt
```

Sample Program 2 (Recommend):

Check the transmission end flag. (SPIST<TEND> = “1”)

```
LD       (SPITDx), A    ; Load “A” the transmit data
DI                      ; Disable the interrupt
SET 3,   (SPICT)        ; Start transmission be setting the TXE bit to “1”
RES 3,   (SPICT)        ; Disable the transmission by clearing the TXE bit to “0”
EI                      ; Enable the interrupt
```

(j) RXMOD

This bit selects the data reception mode from UNIT and Sequential modes. During reception, it is prohibited to change the reception mode from Sequential to UNIT, or vice versa.

For UNIT-mode reception, the receive FIFO buffer is disabled and the RFUL interrupt is generated when the received data is loaded from the receive shift register to the receive data register (SPIRD).

For sequential-mode reception, the 32-byte receive FIFO is enabled and the RFUL interrupt is generated when the size of received data stored in the receive FIFO reaches 16 or 32 bytes.

(k) RXE

In the UNIT–mode reception, writing a "1" to this bit enables the reception of only one UNIT-size data.

When reading the receive data register (SPIRD) while this bit is kept enabled, one more UNIT data is additionally received.

In Sequential mode, writing a "1" to this bit enables the sequential data reception until the 32-byte FIFO buffer becomes full. The state of this bit can be changed even during the data reception. If this bit is cleared to "0" during a data reception, the reception is stopped after completing the reception of the UNIT data currently being received.

[Data Transmission/Reception Modes]

This SPI Controller supports six operating modes as listed below.

These are specified by the FDPXE, RXMOD, RXE, TXMOD, TXE bits.

Table 3.18.2 Data Transmission Reception Modes

Operating Mode	Bit Settings					Description
	<FDPXE>	<TXMOD>	<TXE>	<RXMOD>	<RXE>	
(1) UNIT transmission	0	0	1	x	x	Transmit the SPITD data per UNIT
(2) Sequential transmission	0	1	1	x	x	Transmit the FIFO data sequentially
(3) UNIT reception	0	x	x	0	1	Receive only one UNIT-size data
(4) Sequential reception	0	x	x	1	1	Automatically receive data if FIFO buffer has any empty space
(5) UNIT transmission and reception	1	0	1	0	1	Transmit/receive one UNIT-size data with the addresses of transmit/receive data aligned on UNIT-size boundaries
(6)Sequential transmission and reception	1	1	1	1	1	Transmit/receive data sequentially with the addresses of transmit/receive data aligned on UNIT-size boundaries

x: Don't care

Differences Between the UNIT-mode and Sequential-mode transmissions

The UNIT mode for the data transmission can be selected by writing a “0” to the SPICT<TXMOD> bit.

The transmit FIFO buffer is disabled in UNIT mode. The UNIT-mode transmission starts when the UNIT-size data is loaded into the SPITD register where SPICT<TXE> = “1”, or when the SPICT<TXE> is set to “1” after loading one UNIT-size data into the SPITD register. During the data transmission, it is prohibited to change the transmission mode from Sequential to UNIT, or vice versa.

In the UNIT-mode transmission, the TEMP interrupt is generated when the transmit data is loaded from the transmit data register (SPITD) to the transmit shift register. Also, the TEND interrupt is generated upon completion of the transmission of the last UNIT data.

Important Note:

When in UNIT mode (TXMOD = “0”), the following restriction is imposed on the system operation.

When the SPICT<TEX> bit is set to “1”, the state of any bits must not be changed until the data transmission is completed.

Sample Program 1:

```
        LD      (SPITDx), A     ; Load the transmit data
        DI                      ; Disable the interrupt
        SET 3,  (SPICT)         ; Start transmission by setting the TXE bit to “1”

Wait:   BIT 1,  (SPIST)         ;  Wait for the completion of the transmission
        JPZ,    Wait
        RES 3,  (SPICT)         ; Disable the transmission by clearing the TXE bit to “0”
        EI                      ; Enable the interrupt
```

Sample Program 2 (Recommend):

Check the transmission end flag. (SPIST<TEND> = “1”)

```
        LD      (SPITDx), A     ; Load “A” the transmit data
        DI                      ; Disable the interrupt
        SET 3,  (SPICT)         ; Start transmission be setting the TXE bit to “1”
        RES 3,  (SPICT)         ; Disable the transmission by clearing the TXE bit to “0”
        EI                      ; Enable the interrupt
```

The Sequential mode for the data transmission can be selected by writing a “1” to the SPICT<TXMOD> bit. The 32-byte FIFO is enabled in Sequential mode.

In this mode, the data writes to the transmit FIFO must be performed in 16-byte units. Otherwise, the TEMP interrupt is not properly generated.

In the Sequential-mode transmission, transmit data written into the SPITD is loaded sequentially when SPICT<TXE> = “1”. The transmission in this mode can also be started by setting the SPICT<TXE> bit to “1” after writing the transmit data into the transmit FIFO. The transmit data is transmitted in the same order as they were written into the FIFO.

This mode of transmission keeps transmitting data as long as the transmit data exists. Therefore, the Sequential-mode transmission continues as long as the transmit FIFO (32 bytes) has any valid data. During the data transmission, it is prohibited to change the transmission mode from Sequential to UNIT, or vice versa.

The state of the SPICT<TXE> bit can be changed even during the data transmission. Writing a "0" to the SPICT<TXE> bit during a transmission stops the transmission after completing the transmission of the UNIT data currently being transmitted.

The TEMP interrupt is generated when the empty space size of the FIFO becomes 16 or 32 bytes. The TEND interrupt is generated upon completion of the transmission of the last UNIT data.

Differences Between the UNIT-mode and Sequential-mode Receptions

The UNIT-mode reception receives only one UNIT-size data. The UNIT mode for the data reception can be selected by writing a "0" to the SPICT<RXMOD> bit.

The receive FIFO is disabled in UNIT mode. Writing a "1" to the SPICT<RXE> bit initiates a receive operation of one UNIT data. Then, the transmission is terminated after storing the received data into the receive data register (SPIRD). To perform one-UNIT data reception, read the SPIRD register after writing a "0" to the SPICT<RXE> bit. If the SPIRD register is read again when the SPICT<RXE> bit is set to "1", one-UNIT data is additionally received. During the data reception, it is prohibited to change the reception mode from Sequential to UNIT, or vice versa.

In this mode, the RFUL and REND interrupts are generated when the receive data is loaded into the SPIRD register from the receive shift register.

The Sequential-mode reception automatically receives the data as long as the receive FIFO has any empty space. The Sequential mode is selected by writing a "1" to the SPICT<RXMOD> bit. The 32-byte receive FIFO is disabled in this mode. In this reception mode, the data reads from the receive FIFO must be performed in 16-byteunits. Otherwise, the RFUL interrupt is not properly generated.

Received data is stored into the receive FIFO by writing a "1" to the SPICT<RXE> bit.

This mode of reception keeps receiving the next data automatically unless the data receive FIFO becomes full (32 bytes). Therefore, the reception continues sequentially without stopping at every UNIT-sized reception. During the data reception, it is prohibited to change the reception mode from Sequential to UNIT, or vice versa.

Writing a "0" to the SPICT<RXE> bit during a reception stops the data reception after completing the reception of the UNIT data currently being received.

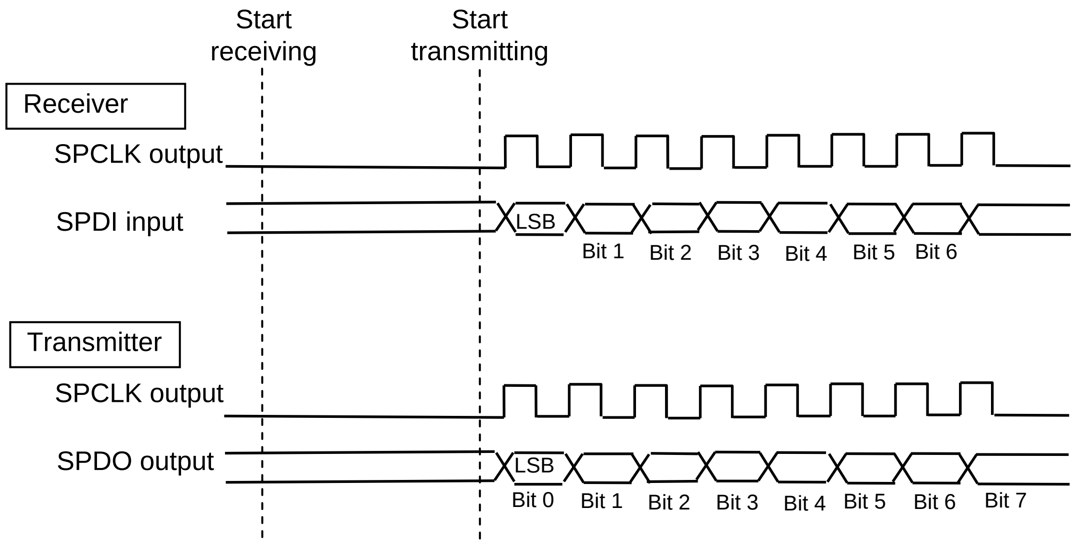
The RFUL interrupt is generated when the size of data stored into the FIFO reaches 16 or 32 bytes. The REND interrupt is generated when the 32-byte receive FIFO becomes full.

Transmit and Receive Operation

When performing a data transmission and reception simultaneously, the FDPXE bit must be set to "1".

Write a "1" to the SPICT<RXE> bit after writing a "1" to the FDPXE bit to put the receiver into standby mode for the UNIT-mode reception. Writing a "1" to the SPICT<RXE> bit after writing a "1" to the <FDPXE> bit does not immediately initiate the receive operation. This is because the data to be transmitted at the same time has not been prepared. Transmit and receive operation is started only after the transmit data is written into the SPITD register where SPICT<TXE> = "1".

The figure below shows the operations of the receiver and transmitter for the simultaneous transmit and receive operation. :



Note: If the data transmission and reception are not performed simultaneously, data communication should be performed with the FDPXE bit cleared to "0".

Figure 3.18.8 Transmit and Receive Operation

(3) Interrupts

The SPIC generates two types of interrupt requests to the Interrupt Controller (INTC), which are the transmit interrupt (INTSPITX) and receive interrupt (INTSPIRX) requests. Also, the SPIC has four types of interrupts; two for transmission and two for reception.

(a) Transmit interrupts

TEMP (Transmit FIFO Empty interrupt) and TEND (Transmit End interrupt)

As for the TEMP interrupt, the timing of the interrupt generation differs depending on the transmission mode, which is UNIT or Sequential.

In the Sequencial-mode transmission, the data writes to the transmit FIFO must be performed in 16–byte units. Otherwise, the TEMP interrupt is not properly generated.

UNIT–mode transmission

Since the transmit FIFO is disabled in this mode, the TEMP interrupt is generated when the data written in the transmit data register (SPITD) is loaded into the transmit shift register.

The TEND interrupt is generated when the transmission of the last UNIT data is completed with the FIFO being empty (i.e., after the falling edge of the last bit clock where SPIMD<TCPOL> = “0”).

Sequential–mode transmission

The TEMP interrupt is generated by the following two conditions: One is when the empty space size of the transmit FIFO reaches 16 bytes, and the other is when it reaches 32 bytes.

The TEND interrupt is generated when the transmission of the last UNIT data is completed with the FIFO being empty (i.e., after the falling edge of the last bit clock where SPIMD<TCPOL> = “0”).

(b) Receive interrupts

RFUL (Receive FIFO interrupt) and REND (Receive End interrupt).

As for the RFUL interrupt, the timing of the interrupt generation differs depending on the reception mode; which is UNIT or Sequential.

In the Sequencial-mode transmission, the data reads from the receive FIFO must be performed in 16-byte units. Otherwise, the RFUL interrupt is not properly generated.

UNIT-mode reception

Since the receive FIFO is disabled in this mode, the RFUL interrupt is generated at the same timing as the REND interrupt is generated.

The RFUL and REND interrupts are generated when the data is loaded from the receive shift register into the receive data register (SPIRD).

Sequential-mode reception

The RFUL interrupt is generated by the following two conditions: One is when the size of data stored into the receive FIFO reaches 16 bytes, and the other is when it reaches 32 bytes.

The REND interrupt is generated when the 32-byte receive FIFO becomes full.

(3-1) SPI Status Register (SPIST)

The SPIST register contains three bits that indicates the status of data communication.

SPIST Register

		7	6	5	4	3	2	1	0
SPIST (0824H)	Bit Symbol					TEMP		TEND	REND
	Read/Write					R		R	
	Reset State					1		1	0
	Function					Transmit FIFO Status 0: No empty space 1: Has an empty space		Transmission Status 0: Transmission in progress or having transmit data 1: Transmission ended	Reception Status 0: Reception in progress or not having receive data 1: Reception Ended or FIFO full
		15	14	13	12	11	10	9	8
(0825H)	Bit Symbol								
	Read/Write								
	Reset State								
	Function								

Figure 3.18.9 SPIST Register

(a) TEMP

For UNIT-mode transmission, this bit is cleared to "0" when the transmit register (SPITD) contains valid data; otherwise, it is set to "1".

For Sequential-mode transmission, this bit is set to "1" when the transmit FIFO buffer contains no valid data.

(b) TEND

This bit is cleared to "0" when the SPITD register or the transmit FIFO contains valid transmit data, and also when the transmission is in progress. This bit is set to "1" after completing the data transmission where the SPITD register and the transmit FIFO contain no valid data.

(c) REND

For UNIT-mode reception, this bit is set to "1" when completing the data reception and valid data is stored into the receive data register (if there is any valid data). This bit is cleared to "0" when the receive register (SPIRD) contains no valid data, or when the reception is in progress.

For Sequential-mode reception, this bit is set to "1" when the 32-byte receive FIFO is full with the valid data after completing the reception of the last data. This bit is cleared to "0" when there is still an empty space of one byte or more in the FIFO.

The RFUL flag does not exist because its function is exactly the same as the REND flag.

(3-2) SPI Interrupt Enable Register (SPIE)

The SPIIE register enables or disables the generation of four types of interrupts.

SPIIE Register

		7	6	5	4	3	2	1	0
SPIIE (082CH)	Bit Symbol					TEMPIE	RFULIE	TENDIE	RENDIE
	Read/Write					R/W			
	Reset State					0	0	0	0
	Function					TEMP interrupt 0:Disable 1:Enable	RFUL interrupt 0:Disable 1:Enable	TEND interrupt 0:Disable 1:Enable	REND interrupt 0:Disable 1:Enable
		15	14	13	12	11	10	9	8
(082DH)	Bit Symbol								
	Read/Write								
	Reset State								
	Function								

Figure 3.18.10 SPIIE Register

(a) TEMPIE

This bit enables or disables the TEMP interrupt.

(b) RFULIE

This bit enables or disables the RFUL interrupt.

(c) TENDIE

This bit enables or disables the TEND interrupt.

(d) RENDIE

This bit enables disables the REND interrupt.

Note: The SPIC supports four types of interrupts; two transmit interrupts (TEMP,and TEND, both of which causes the generation of the INTSPITX interrupt request) and two receive interrupts (RFUL and REND, both of which causes the generation of the INTSPIRX interrupt request). However, for the proper operation, select either one of the TEMP and TEND interrupts and also select either one of the RFUL and REND interrupts. (Simultaneous use of the TEMP and TEND interrupts is prohibited, as well as the simultaneous usage of the RFUL and REND interruptsy.)

(4) SPI CRC Register (SPICR)

The SPICR register contains the CRC calculation result for transmit/receive data.

SPICR Register

		7	6	5	4	3	2	1	0
SPICR (0826H)	Bit Symbol	CRCD7	CRCD6	CRCD5	CRCD4	CRCD3	CRCD2	CRCD1	CRCD0
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	CRC result bits [7:0]							
		15	14	13	12	11	10	9	8
(0827H)	Bit Symbol	CRCD15	CRCD14	CRCD13	CRCD12	CRCD11	CRCD10	CRCD9	CRCD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	CRC result bits [15:8]							

Figure 3.18.11 SPICR Register

(a) CRCD15:0

The CRC result which is calculated according to the settings of the CRC16_7_b, CRCRX_TX_B and CRCRESET_B bits in the SPICT register are loaded into this register. When using the CRC16 algorithm, all the bits participate in the CRC generation. When using the CRC7 algorithm, only the lower seven bits participates in the CRC generation. The following describes the steps required to calculate the CRC16 for the transmit data.

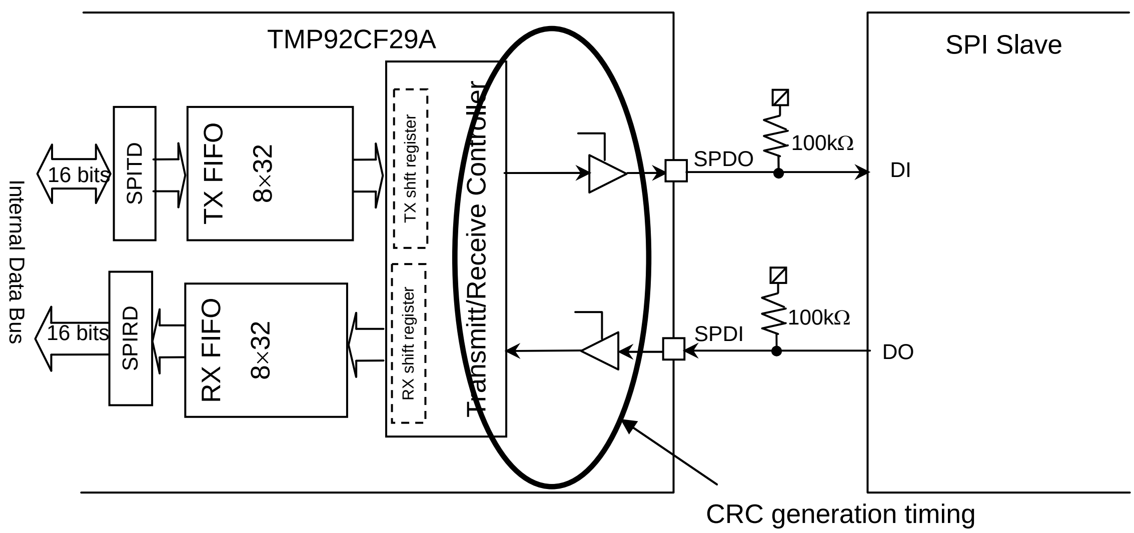
First, initialize the CRC calculation register by writing a "1" to the CRCRESET_B bit after programming three bits as follows: CRC16_7_b = "1", CRCRX_TX_B = "0", and CRCRESET_B = "0".

Then, by writing the transmit data into the SPITD register, complete the transmission of all bits, for which the CRC should be calculated.

The SPIST<TEND> bit should be checked to confirm whether the reception is completed.

By reading the SPICR register after the transmission is completed, the CRC16 for the transmit data can be obtained.

Note: The CRC is generated upon data input and output of the TMP92CF29A as illustrated below. The timing of the CRC comparison should be fully considered when performing Sequential-mode transmit and receive operation using the FIFOs.



(5) SPI Transmit Data Register (SPITD)

The SPITD0 and SPITD1 registers are used for writing the transmit data.

SPITD0 Register

		7	6	5	4	3	2	1	0
SPITD0	Bit Symbol	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
(0830H)	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transmit data bits [7:0]							
		15	14	13	12	11	10	9	8
(0831H)	Bit Symbol	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transmit data bits [15:8]							

SPITD1 Register

		7	6	5	4	3	2	1	0
SPITD1	Bit Symbol	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
(0832H)	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transmit data bits [7:0]							
		15	14	13	12	11	10	9	8
(0833H)	Bit Symbol	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Transmit data bits [15:8]							

Figure 3.18.12 SPITD Register

This register is used for writing the transmit data. When this register is read, the last-written data is read out. This register is overwritten if the next data is written with the transmit FIFO being full.

Since the transmit data registers can contain data of up to four bytes, it can support write operations that are performed by using four-byte instructions, such as the parallel operation of the SPI and DMA.

When writing the data, the transmit data at the address 830 must always be the first to be written.

There are several restrictions of the data writing methods (i.e., instructions to be used). For more details, please refer to the following table.

Transmit Data Write Size	Instruction Example	UNIT-mode Transmission (FIFO Disabled)		Sequential-mode Transmission (FIFO Enabled)	
		1-byte transmission unit16 = 0	2-byte transmission unit16 = 1	1-byte transmission unit16 = 0	2-byte transmission unit16 = 1
1-byte write	ld (0x830),a	○	●	Prohibited	●
2-byte write	ld (0x830),wa	●	○	○	○
4-byte write	ld (0x830),xwa	●	●	○	○

○: All data that are written by the CPU are transmitted.

●: Invalid data are also transmitted along with the data written by the CPU.

(6) SPI Receive Data Register (SPIRD)

The SPIRD0 and SPIRD1 registers are used for reading the received data.

SPIRD0 Register

		7	6	5	4	3	2	1	0
SPIRD0	Bit Symbol	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
(0834H)	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Receive data bits [7:0]							
		15	14	13	12	11	10	9	8
(0835H)	Bit Symbol	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Receive data bits [15:8]							

SPIRD1 Register

		7	6	5	4	3	2	1	0
SPIRD1	Bit Symbol	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
(0836H)	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Receive data bits [7:0]							
		15	14	13	12	11	10	9	8
(0837H)	Bit Symbol	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Receive data bits [15:8]							

Figure 3.18.13 SPIRD Register

This register is used for reading the received data. Please check the state of the RFUL or REND bit before starting a read operation.

Since the receive data registers can contain data of up to four bytes, it can support read operations that are performed by using four-byte instructions, such as the parallel operation of the SPI and DMA.

When reading the data, the receive data at the address 834 should be the first to be read. (There are some exceptions.)

There are several restrictions of the data reading methods (i.e., instructions to be used). For mode details, please refer to the following table.

Receive Data Read Size	Instruction Example	UNIT-mode Reception (FIFO Disabled)		Sequential-mode Reception (FIFO Enabled)	
		1-byte reception unit16 = 0	2-byte reception unit16 = 1	1-byte reception unit16 = 0	2-byte reception unit16 = 1
1-byte read	ld a,(0x834)	○	○	Prohibited	Prohibited
	ld a,(0x835)	●	○	Prohibited	Prohibited
2-byte read	ld wa,(0x834)	♦*1	○	○	○
4-byte read	ld xwa,(0x834)	♦*2	♦*3	○	○

○: Only the valid data are read when the CPU is reading.

♦: Valid data + invalid data are read when the CPU is reading. Invalid data must be deleted later.

●: Only the invalid data are read when the CPU is reading.

*1: Address 834 = Valid data, address 835 = Invalid data,

*2: Address 834 = Valid data, address 835 = Invalid data, address 836 = Invalid data, address 837 = Invalid data

*3: Address 834 = Valid data, address 835 = Valid data, address 836 = Invalid data, address 837 = Invalid data

3.18.3 Notes on the Operations Using the FIFO Buffers

Things to be noted when using the SPIC are as follows:

1) Transmission

The transmit FIFO buffer is overwritten if the new data is written with the transmit FIFO buffer being full. Also, since the FIFO write pointer does not point to the correct write position, interrupts and transmissions are not properly executed. Therefore, the number of writes should be controlled by using software.

In the Sequential-mode transmission, the data writes to the transmit FIFO must be performed in 16-byte units. Otherwise, the TEMP interrupt is not properly generated.

Note: For data transmission in units of other than 16 bytes, UNIT mode must be selected.

2) Reception

If a read operation is performed when the receive FIFO is empty, undefined data is read. Also, since the FIFO read pointer does not point to the correct read position, interrupts and receptions are not properly executed. Therefore, the number of reads should be controlled by using software.

In the Sequential-mode reception, the data reads from the receive FIFO must be performed in 16–byte units. Otherwise, the RFUL interrupt is not properly generated.

Note: For data reception in units of other than 16 bytes, UNIT mode must be selected.

3) CRC

The CRC is generated upon transmission and reception to/from the SPI slave device. (Refer to the section on the SPICRC register fro more details.) The timing of the CRC comparison should be fully considered when performing Sequential-mode transmit and receive operation using the FIFOs.

Example: Sequential-mode reception

1. Start Sequential-mode reception
2. finish valid data receive (FIFO_Full)
3. Stop data reception
4. Read valid data from the FIFO to a temporary buffer (internal RAM, etc.)
5. Read CRC1 from the CRC generator in the SPIC
6. Start CRC2 reception (upon UNIT-mode reception from the SD-CARD)
7. Compare CRC1 and CRC2

Note: The steps 2 to 4 of the above sequence can be used DMAC. However, to perform the CRC comparison, the receive operation must be stopped once as described in step 3. Otherwise, the CRC1 value obtained from the internal CRC generator unintentionally contains CRC2 as well as the valid data, which leads to an incorrect CRC comparison.

3.19 I^2S (Inter-IC Sound)

The TMP92CF29A incorporates serial output circuitry that is compliant with the I^2S format. This function enables the TMP92CF29A to be used for digital audio systems by connecting an LSI for audio output such as a DA converter.

The I^2S unit has the following features:

Table 3.19.1 I^2S Operation Features

Item	Description
Number of Channels	1 channel
Format	I^2S-format compliant Right-justified and left-justified formats supported Stereo / monaural Master transmission only
Pins used	1. I2S0CKO (clock output) 2. I2S0DO (output) 3. I2S0WS (Word Select output)
WS frequency	Refer to “Setting the transfer clock generator and Word Select signal”.
Data transfer rate	
Transmission buffer	64 bytes × 2
Direction of data	MSB-first or LSB-first selectable
Data length	8 bits or 16 bits
Clock edge	Rising edge or falling edge
Interrupt	INTI2S0 (64-byte FIFO empty interrupt)

3.19.1 Block Diagram

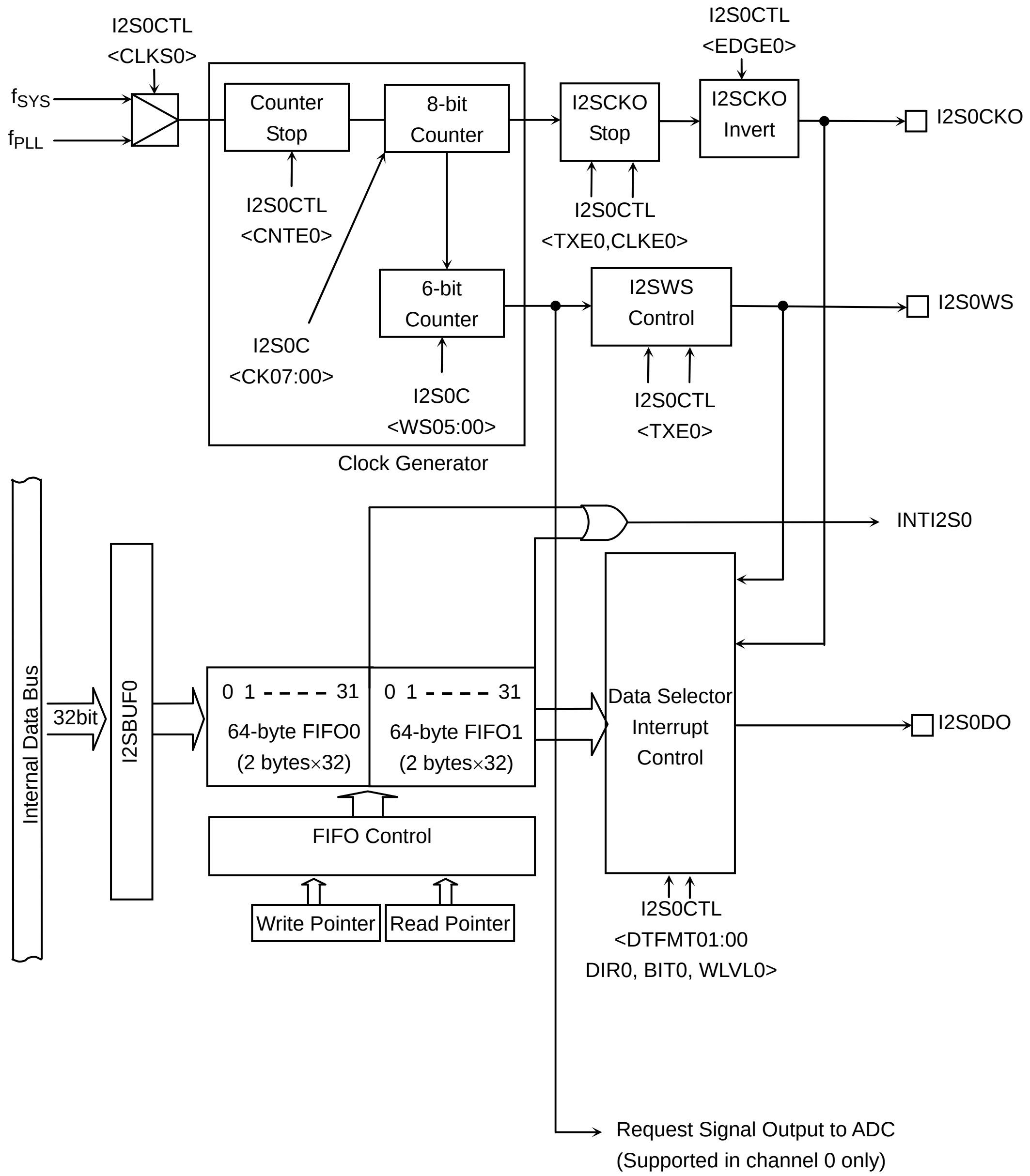


Figure 3.19.1 I^2S Block Diagram

3.19.2 SFRs

The I²S unit is provided with the following registers. These registers are connected to the CPU via a 32-bit data bus. The transmission buffers I2S0BUF must be accessed using 4-byte load instructions.

I^2S0 Control Register

		7	6	5	4	3	2	1	0
I2S0CTL (1808H)	bit Symbol	TXE0	*CNTE0		DIR0	BIT0	DTFMT01	DTFMT00	SYSCKE0
	Read/Write	R/W			R/W				
	Reset State	0	0		0	0	0	0	0
	Function	Transmission 0: Stop 1: Start	Counter control 0: Clear 1: Start		Transmission start bit 0:MSB 1:LSB	Bit length 0: 8 bits 1: 16 bits	Output format 00: I^2S 10: Right 01: Left 11: Reserved		System clock 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(1809H)	bit Symbol	CLKS0			FSEL0	TEMP0	WLVL0	EDGE0	CLKE0
	Read/Write	R/W			R/W	R	R/W		
	Reset State	0			0	1	0	0	0
	Function	Source clock 0: f_{SYS} 1: f_{PLL}			Stereo /monaural 0: Stereo 1: Monaural	Transmission FIFO state 0: Data 1: No data	WS level 0: Low left 1: High left	Data output clock edge 0: Falling 1: Rising	Clock operation (after transmis-sion) 0: Enable 1: Disable

I^2S0 Divider Value Setting Register

		7	6	5	4	3	2	1	0
I2S0C (180AH)	bit Symbol	CK07	CK06	CK05	CK04	CK03	CK02	CK01	CK00
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Divider value for CK signal (8-bit counter)							
		15	14	13	12	11	10	9	8
(180BH)	Bit symbol			WS05	WS04	WS03	WS02	WS01	WS00
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Divider value for WS signal (6-bit counter)					

I^2S0 Buffer Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2S0BUF (1800H)	bit Symbol	B015	B014	B013	B012	B011	B010	B009	B008	B007	B006	B005	B004	B003	B002	B001	B000
	Read/Write	W															
A read-modify-write operation cannot be performed	Reset State	Undefined															
	Function	Transmission buffer register (FIFO)															
		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	bit Symbol	B031	B030	B09	B028	B027	B026	B025	B024	B023	B022	B021	B020	B019	B018	B017	B016
	Read/Write	W															
	Reset State	Undefined															
	Function	Transmission buffer register (FIFO)															

Figure 3.19.2 I^2S Channel 0 Control Registers

(a) <SYSCKE0>

This bit controls to connect source clock to I^2S circuit.

In case of this circuit is operated, it must enable: <SYSCKE0>= "1". And except operating, for reduce the power consumption, we recommends to disable: <SYSCKE0>= "0".

(b) <DTFMT01:00>

This bit controls data format: I^2S, right justify and left justify.

It is not possible to change data format during data transmission. Before changing the data format, set <SYSCKE0>= "1", <CNTE0>="0" and <TXE0>= "0".

(c) <BIT0>

This bit controls data length: 8/16 bits.

It is not possible to change data length during data transmission. Before changing the data format, set <SYSCKE0>= "1", <CNTE0>= "0" and <TXE0>= "0".

(d) <DIR0>

This bit controls direction: LSB_Fast or MSB_Fast

It is not possible to change data direction during data transmission. Before changing the data format, set <SYSCKE0>= "1", <CNTE0>="0" and <TXE0>="0".

(e) <CNTE0>

This bit controls clock generator counter: Clear/Start.

When this circuit is used, always set to the start condition.

Clock generator counter will clear by <TXE0>="0" and <CNTE0>="0", However, Clock generator counter will not clear by <TXE0>="0" and <CNTE0>="1"

(f) <TXE0>

This bit controls data transmission and FIFO buffer clear: Trans/Stop and Clear Transmission is stopped by <TXE0>="0", started by <TXE0>="1".

Output FIFO buffer is cleared by <TXE0>="0".

(g) <CLKE0>

This bit controls CLK out period.

<CLKE0>="0": always out I2S0CKO clock, <CLKE0>="1": I2S0CKO clock out during effective data out period.

Note: In case of I^2S format, firstly I2S0WS signal change and after 1clock period, effective data out. If set to <CLKE0>= "1" with I^2S format, 1 clock pulse after I2S0WS don't out. It is not possible <CLKE0>="0" setting with I^2S format.

(h) <EDGE0>

This bit controls relation of phase between I2S0CKO and data.

<EDGE0>="0": the data is changed in the falling of clock, and the data is latched in the rising edge of clock.

<EDGE0>="1": the data is changed in the rising of clock, and the data is latched the falling edge of clock.

It is not possible to change phase during data transmission. Before changing the data format, set <SYSCKE0>="1", <CNTE0>="0" and <TXE0>="0".

(i) <WLVL0>

This bit controls phase of Word Select signal: I2S0WS

I2S0WS signal always out "1" level first. The order of data output changes by <WLVL0>. Refer the "FIFO buffer and data format" in details.

It is not possible to change phase of Word Select signal during data transmission. Before changing the data format, set <SYSCKE0>= "1", <CNTE0>= "0" and <TXE0>="0".

(j) <TEMP0>

This bit is empty flag of output FIFO buffer.

<TEMP0>="1": FIFO buffer is empty, <TEMP0>="0": remain data in FIFO buffer.

This bit is read only. FIFO buffer is cleared by <TXE0>="0"

(k) <FSEL0>

This bit controls sound mode: Stereo / Monaural

<FSEL0>="0": Stereo, <FSEL0>="1": Monaural. Refer the chapter of "Data format" in details.

It is not possible to change sound mode during data transmission. Before changing the data format, set <SYSCKE0>="1", <CNTE0>="0" and <TXE0>="0".

(l) <CLKS0>

This bit controls source clock to I^2S circuit: f_{SYS} / f_{PLL}.

<CLKS0>="0": f_{SYS} is supplied, <CLKS0>="1": f_{PLL} is supplied.

In case of using f_{PLL}, before set f_{PLL} clock, please take care set-up time: Lock-Up time. In details, refer the chapter of PLL, please.

(m) <CK07:00>

These bits are set counter value of clock generator. [I2S0CK]

It is not possible to change these counter value during data transmission. Before changing the counter value, set <SYSCKE0>="1", <CNTE0>="0" and <TXE0>="0".

(n) <WS05:00>

These bits are set counter value of clock generator. [I2S0WS]

It is not possible to change these counter value during data transmission. Before changing the counter value, set <SYSCKE0>="1", <CNTE0>="0" and <TXE0>="0".

3.19.3 Description of Operation

(1) Settings the transfer clock generator and Word Select signal

In the I²S unit, the clock frequencies for the I2S0CKO and I2S0WS signals are generated using the system clock (f_{SYS}) as a source clock. The system clock is divided by a prescaler and a dedicated clock generator to set the transfer clock and sampling frequency.

The counters are started by setting I2S0CTL<CNTE0> to "1" and are stopped and cleared by setting <CNTE0> to "0".

A) Clock generator

- 8-bit counter

  This is an 8-bit counter that generates the I2S0CKO signal by dividing the clock selected by I2S0CTL<CLKS0>.

- 6-bit counter

  This is a 6-bit counter that generates the I2S0WS signal by dividing the I2S0CKO signal.

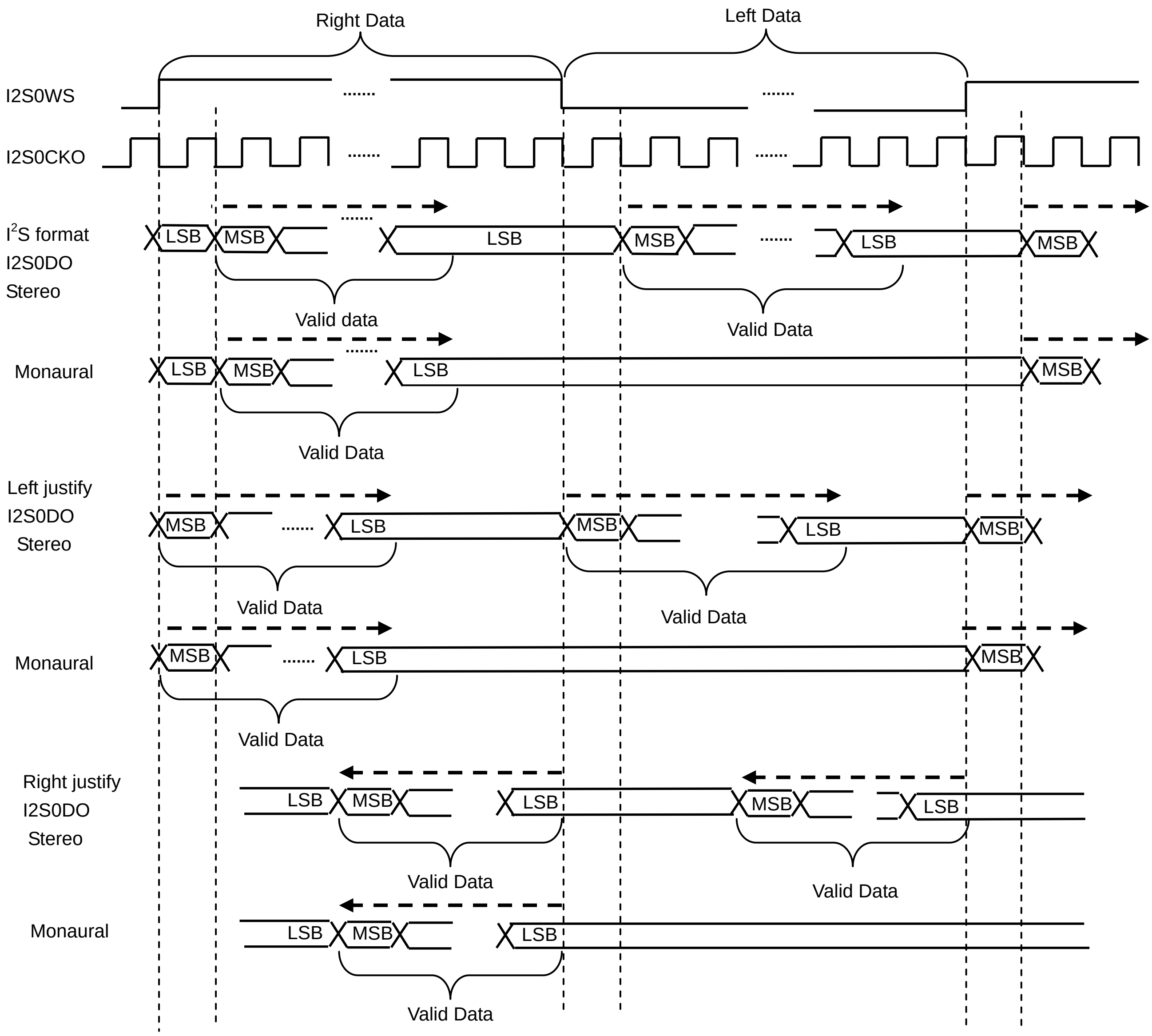
B) Word Select

- Word Select signal (I2S0WS)

  The I2S0WS signal is used to distinguish the position of valid data and whether left data or right data is being transmitted in the I²S format. This signal is clocked out in synchronization with the data transfer clock. In only channel 0, this signal can be used as an AD conversion trigger signal for the ADC. How valid data is to be output in relation to the WS signal can be specified as I²S format, left-justified, or right-justified. In only channel 0, an interrupt request can be output to the ADC on the rising edge of the WS signal. (This is controlled by the ADC's control register.)

(2) Data format

This circuit support I²S format, left justify and right justify format by setting I2S0CTL<DTFMT01:00> register. And support stereo and monaural both, controlled by I2S0CTL<FSEL0> register.



Note: When Monaural is set, Right data and Left data output the same-signal. When Stereo is set, the data of FIFO buffer is renewed every one-word. However, when Monaural is set, it is renewed to the next data after the same-data is transmitted two times.

When Monaural is set, it output the same-data to Right and Left, without the same-data written to the FIFO buffer.

The Monaural function of TMP92CZ26A/CF26A and TMP92CF29A is different.

The Monaural function of TMP92CZ26A/CF26A is one of channels only.

Figure3.19.3 Output Format

(3) Setting example for the clock generator (8-bit counter/6-bit counter)

The clock generator generates the reference clock for setting the data transfer speed and sampling frequency.

		7	6	5	4	3	2	1	0
I2S0C	bit Symbol	CK07	CK06	CK05	CK04	CK03	CK02	CK01	CK00
(180AH)	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Divider value for CK signal (8-bit counter)							
		15	14	13	12	11	10	9	8
	Bit symbol			WS05	WS04	WS03	WS02	WS01	WS00
(180BH)	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Divider value for WS signal (6-bit counter)					

- Setting the transfer clock I2S0CKO

The transfer clock is generated by dividing the clock selected by I2S0CTL <CLKS0>. An 8-bit counter is provided to divide the source clock by 3 to 256. (The divider value cannot be set to 1 or 2.)

The transfer clock must not exceed 10 MHz. Make sure that the transfer clock is set to within 10 MHz by an appropriate combination of source clock frequency and divider value.

8-bit counter set value	Divider value
00000000	256
00000001	1
11111111	255

When f_{SYS} = 60 MHz and I2S0C<CK07:00> = 150, the data transfer speed is set as follows:

I2S0CKO = f_{SYS}/150

= 60 [MHz]/150 = 400 [kbps]

Note: It is recommended that the value to be set in I2S0C<CK07:00> be an even number. Although it is possible to set an odd number, the clock duty of the CK signal does not become 50%. Setting an odd number causes the High width of the I2S0CKO signal to become longer by one f_{SYS} or f_{PLL} pulse than the Low width. (When <EDGE0> = 0, the Low width becomes longer than the High width.)

- Setting the sampling frequency WS

  The sampling frequency is set by dividing the transfer clock (CK) described above. A 6-bit counter is provided to divide the transfer clock by 16 to 64. (The divider value cannot be set to 1 to 15.)

6-bit counter set value	Divider value
000000	64
000001	1
111111	63

When f_{SYS} = 60 MHz, I2S0C<CK07:00> = 150, and I2SnC<WS05:00> = 50, the sampling frequency is set as follows:

I2S0CKO = f_{SYS} / 150 / 50

= 60 [MHz] / 150 / 50 = 8 [kHz]

Based on the above, the transfer clock is set to 400 kbps, and the sampling frequency is set to 8 kHz in this example.

Note 1: The value to be set in I2S0C<WS05:00> must be 16 or larger (18 or larger for I^2S transfer) when the data length is 8 bits and 32 or larger (34 or larger for I^2S transfer) when the data length is 16 bits.

Note 2: It is recommended that the value to be set in I2S0C<WS05:00> be an even number. Although it is possible to set an odd number, the clock duty of the WS signal does not become 50%. Setting an odd number causes the High width of the WS signal to become longer by one I2S0CKO pulse than the Low width.

- Special function

  As a special function available only in channel 0, the rising edge of the WS signal can be used as an AD conversion start trigger for the AD converter in this LSI. Setting I2S0CTL<SYSKE0>=1 and I2S0CTL<CNTE0>=1 enables the WS signal to be sent to the AD converter. This can be done regardless of the setting of I2S0CTL<TXE0>.

  For details about AD conversion using the WS signal, refer to the chapter on the AD converter.

(4) FIFO buffer and data format

The I²S unit is provided with a 128-byte FIFO buffer (32-bit wide × 32-entry). The data written to the 4 bytes (32 bits) of the I2S0BUF register is written to this FIFO buffer. This FIFO must be written in units of 4 bytes. It is also necessary to consider the output order and to distinguish between right data and left data.

To write data to the I2S0BUF register, be sure to use a 4-byte load instruction. If a 1-byte load instruction is used, invalid data will be transmitted. In case of using 1-byte or 2-byte transmission instruction, FIFO buffer isn't renewed and transmission isn't started.

And window addresses are 1800H (channel 0) and 1810H (channel1).

Write Data Size	Example instruction	8-bit width	16-bit width
1-byte access	ld (0x1800),a	Not allowed	Not allowed
2-byte access	ld (0x1800),wa	Not allowed	Not allowed
4-byte access	ld (0x1800),xwa	OK	OK

Also note that data must be written in units of 64 bytes using the following sequence:

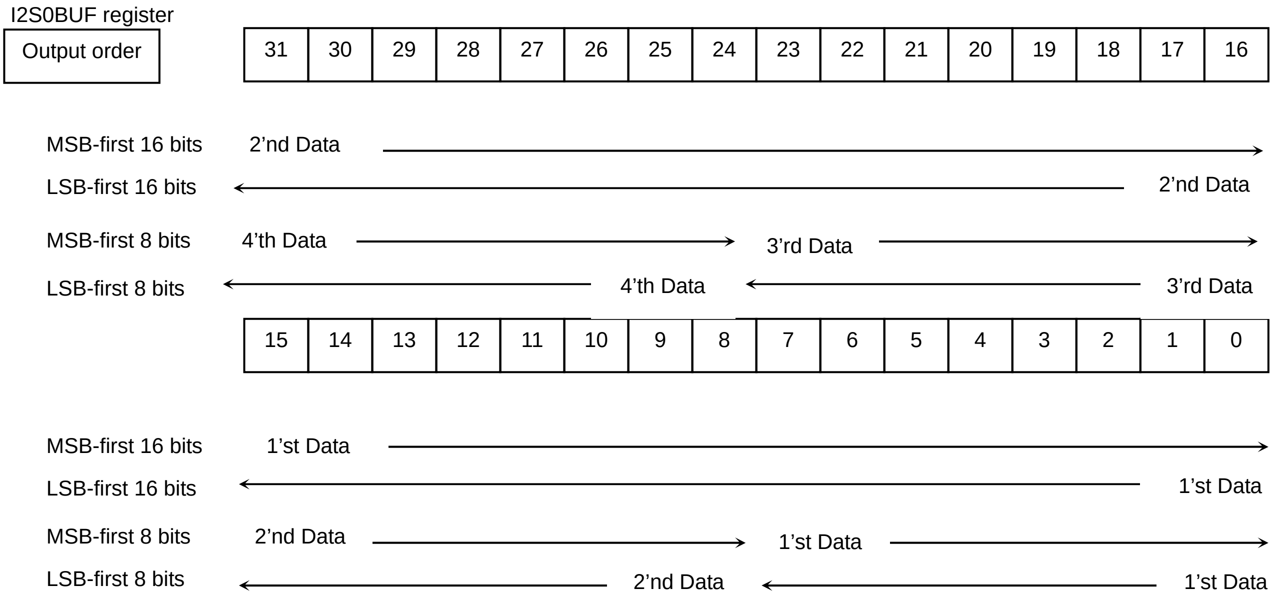
4-byte load instruction × 16 times = 64-byte data write

If data is not written in units of 64 bytes, interrupts cannot be generated at the normal timing.

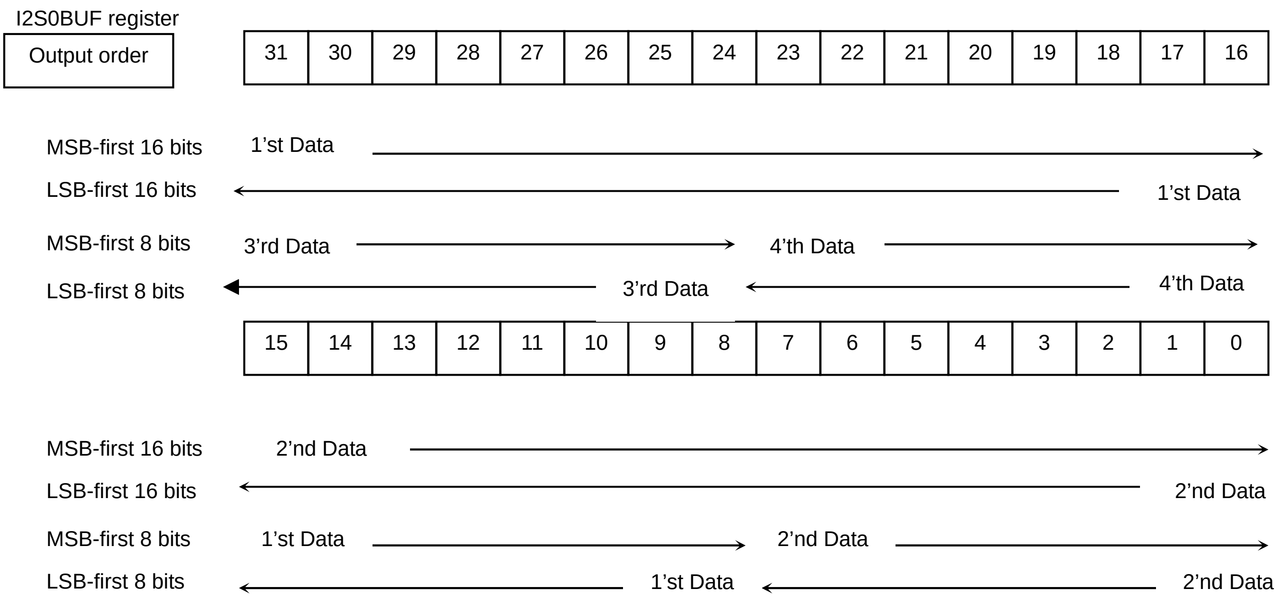
The I2S0CTL<TEMP0> flag is set to "1" when the FIFO buffer for each channel contains no valid data. If there is even one byte of valid data in the FIFO, the flag is cleared to "0". (The <TEMP0> flag is set to "1" as soon as the last valid data in the FIFO is sent to the transmission shift register.)

The following shows how written data is output under various conditions.

When I2S0CTL<WLVL0> = “0”



When I2S0CTL<WLVL0> = “1”

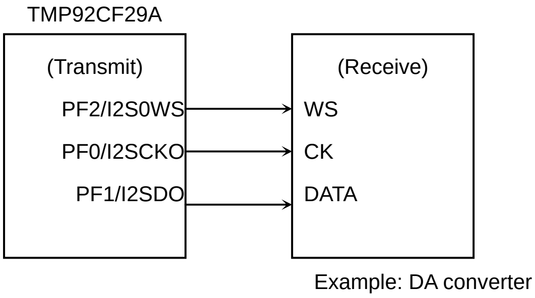


Note: In case of using monaural setting, and change right / left: I2S0CTL<WLVL0>, data output order change off 1'st data and 2'nd data. when Monaural is set, it is renewed to the next data after the same-data is transmitted two times.

3.19.4 Detailed Description of Operation

(1) Connection example

Figure3.19.4 shows an example of connections between the TMP92CF29A and an external LSI (DA converter) using channel 0.



Note: After reset, PF0 to PF2 are placed in a high-impedance state. Connect each pin with a pull-up or pull-down resistor as necessary.

Figure3.19.4 Connection Example between the TMP92CF29A and an External LSI

(2) Operation procedure

The I²S unit incorporates a 128-byte FIFO buffer that is divided into two 64-byte units. Whenever each 64-byte buffer space becomes empty, an INTI2S0 interrupt is generated. The next data to be transmitted should be written to the FIFO in the interrupt routine.

Example settings and timing diagram are shown below.

(Example settings) I2S0WS = 8kHz, I2S0CKO = 400kHz, data transmission on the rising edge (at f_{SYS} = 60 MHz)

(Main routine)

	7	6	5	4	3	2	1	0	
INTEI2S01	X	–	–	–	X	0	0	1	Set interrupt level.
PFCR	X	X	–	–	–	–	–	–	Set pins: PF0 (I2S0CKO), PF1 (I2S0DO), PF2 (I2S0WS)
PFFC	–	X	–	–	–	1	1	1	
I2S0C	1	0	0	1	0	1	1	0	Divider value N=150
	X	X	1	1	0	0	1	0	Divider value K=50
I2S0CTL	0	0	X	0	1	0	0	1	Set transmit mode (I2S mode, MSB-first, 16-bit).
	0	X	X	X	X	0	0	0	Falling edge, WS=0 Left, clock stop.
I2S0BUF	*	*	*	*	*	*	*	*	Write left and right data to FIFO (4 bytes × 32 = 128 bytes).
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
I2S0CTL	1	1	X	0	1	0	0	1	Start transmission.
	0	X	X	0	X	0	0	0	

(INTI2S Interrupt Routine)

	7	6	5	4	3	2	1	0	
I2S0BUF	*	*	*	*	*	*	*	*	Write left and right data to FIFO (4 bytes × 16 = 64 bytes).
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	
	*	*	*	*	*	*	*	*	

X: Don't care, –: No change

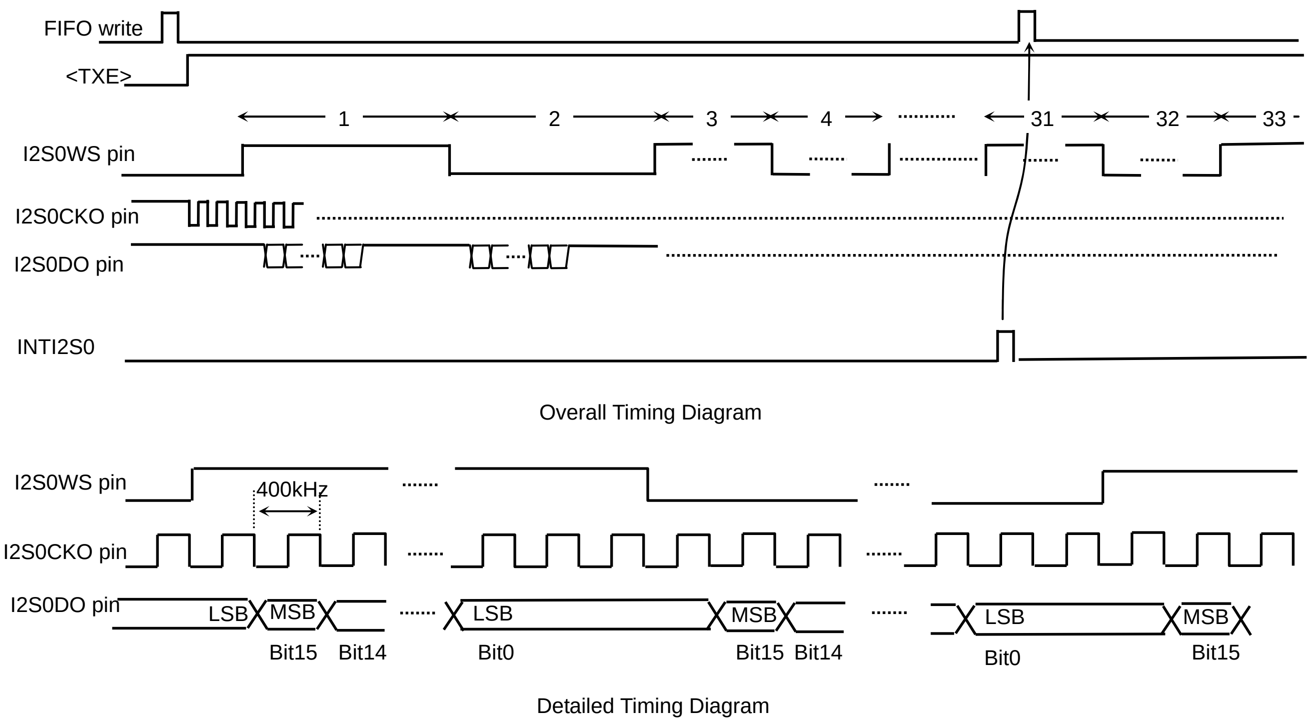


Figure3.19.5 Timing Diagrams (I²S FMT/Stereo/16bit/MSB first)

(3) Considerations for using the I^{2}S unit

1) INTI2S0 generation timing

Every 4bytes data trance from FIFO buffer to shift register per one time.

An INTI2S0 interrupt is generated under two conditions. One is when there are 64 bytes of empty space in the FIFO (after 61- 64th byte has been transferred to the shift register). The other is when the FIFO becomes completely empty (after 125-128th byte has been transferred to the shift register). Therefore, INTI2S0 indicates that there are 64 bytes or 128 bytes of empty space in the FIFO, enabling the next data to be written.

The FIFO must be written in units of 64 bytes. Since the FIFO can contain 128 bytes of data, I^{2}S output can be performed continuously as long as there are 64 bytes of data in the FIFO. It is also possible to check the FIFO state by using the I2S0CTL<TEMP0> flag.

2) I2S0CTL<TXE0>

Transmission is started by setting I2S0CTL <TXE0> to “1”. Once <TXE0> is set to “1”, transmission is continued automatically as long as the FIFO contains the data to be transmitted. While <TXE0> is set to “1” (transmission in progress), the other bits in the I2S0CTL register must not be changed.

To stop transmission, make sure that the FIFO is empty by checking the I2S0CTL<TEMP0> flag. Then, after waiting for two periods of the I2S0WS signal (after all the data has been transmitted), set <TXE0> to “0”. In case monaural setting, make sure that the FIFO is empty by checking the I2S0CTL<TEMP0> flag. Then, after waiting for four periods of the I2SWS signal (after all the data has been transmitted), set <TXE0> to “0”.

If <TXE0> is set to “0” while data is being transmitted, the transmission is stopped immediately. At the same time, the read and write pointers of the FIFO, the data in the output shift register and the clock generator are all cleared. (However, when I2S0CTL<CNTE0>=1, the clock generator is not cleared. To clear the clock generator, I2S0CTL<CNTE0> must be set to “0”). Therefore, if transmission is stopped and then resumed, no data will be output.

The WS signal stops at Low level and the CK signal stops at Low level when the rising edge is selected and at High level when the falling edge is selected.

3) I2S0CTL<CNTE0>

I2S0CTL<CNTE0> is used to control the clock generator (8-bit counter, 6-bit counter) for generating the I2S0CKO and I2S0WS signals.

Setting I2S0CTL<CNTE0> to “1” starts the counters, and setting this bit to “0” stops the counters. Normally, I^{2}S data transmission is executed by setting both I2S0CTL<TXE0> and <CNTE0> to “1”. When transmission is stopped by setting I2S0CTL<TXE0> to “0” with I2S0CTL<CNTE0>= “1”, the clock generator is not cleared. To clear the clock generator, I2S0CTL<CNTE0> must be set to “0”.

4) FIFO buffer

The I²S unit is provided with a 128-byte FIFO. Although it is not necessary to use all 128 bytes in the FIFO, data should basically be written in units of 64 bytes using an INTI2S0 interrupt as a trigger. If data is written to the FIFO without waiting for an INTI2S0 interrupt or in units other than 64 bytes, interrupts cannot be generated properly.

If the last set of data, for which an interrupt is not needed, contains less than 64 bytes, set I2S0CTL<TXE0> to “0” to stop the transmission after writing the data, then checking that the <TEMP0> flag is set to “1”, and waiting for two I2S0WS periods (i.e., after all the data has been transmitted). In case monaural setting, make sure that the FIFO is empty by checking the I2S0CTL<TEMP0> flag. Then, after waiting for four periods of the I2S0WS signal (after all the data has been transmitted), set <TXE0> to “0”.

5) I2S0BUF

When writing data to the I2S0BUF register, be sure to use long-word data load instructions. Word data load or byte data load instructions cannot be used.

Examples)

```
ld    (I2S0BUF), xwa;  OK
ld    (I2S0BUF), wa;   NG
ld    (I2S0BUF), a;    NG
```

6) Share with HALT instruction

I²S circuit is not operated at IDLE1/STOP modes. Therefore, maybe PLL clock that operate at IDLE1 mode affects to this circuits. If mode is shifted to HALT mode, set it after I²S circuit is stopped.

When the CPU is shifted to the HALT mode after transmission is stopped, the time to stop completely is necessary before execution of HALT instruction.

It’s time is NOP×10.

Example:

```
ld    (I2S0CTL), 0x00   ; Stop transmission
NOP×10
HALT
```

3.20 LCD Controller (LCDC)

The TMP92CF29A incorporates an LCD controller (LCDC) for controlling an LCD driver LSI (LCD module). This LCDC supports monochrome, grayscale, from 256-color to 65536-color and display sizes from 64 × 64 to 640 × 480 dots. The supported LCD driver (LCD module) types are STN (Super Twisted Nematic) and digital RGB input TFT (Thin Film Transistor).

- STN support

  With LCD drivers supporting STN, an 8-bit data interface is used to realize 2-monochrome, 4-graysale, 16-grayscale, 64-grayscale, 256-color, 4096-color, 65536-color display.

  After required settings such as the operation mode, display RAM start address, and LCD size (common, segment) are made in the I/O registers, the start register is set to enable the LCDC. The LCDC outputs a bus request to the CPU, reads data from the display RAM, converts the data as necessary, and writes it to a dedicated FIFO buffer.

- TFT support

  With LCD drivers supporting digital RGB input TFT, an 8- to 16-bit data interface is used to realize 4096-color, 65536-color display. The data transfer method is the same as in the case of STN.

  The LCDC controls LCD display operations using 8-bit RGB (R3:G3:B2), 12-bit RGB (R4:G4:B4), 16-bit RGB (R5:G6:B5) display data, the shift clock LCP0 for capturing data, the frame signal LFR, the data load signal LLOAD, and the LDIV signal for indicating the inversion of data output. The LDIV signal can be used effectively in reducing noise and power consumption.

  The LCDC also has horizontal synchronization signal LHSYNC and vertical synchronization signal LVSYNC for controlling gate drivers, and three programmable OE pins for supporting various signals of the TFT driver to be used.

  TFT also support Monochrome TFT panel as a special function.

3.20.1 LCDC Features according to LCD Driver Type

Table 3.20.1 LCDC Features according to LCD Driver Type

(This table assumes the connection with a TOSHIBA-made LCD driver.)

LCD Driver		Shift Register Type	
		TFT	STN
Display colors		256/4096/65536 colors	Monochrome, 4/16/64 grayscale levels 256/4096/65536 colors
Number of pixels that can be displayed		**For 4096 colors or less** Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 240, 320, 480, 640	**For Monochrome/grayscale/4096 colors or less** Rows (Commons): 64, 96, 120, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320, 480, 640
		For 4096 colors or less Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320, 480, 640	
		For 65536 colors or less Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320, 480	**For 65536 colors or less** Rows (Commons): 64, 96, 128, 160, 200, 240, 320, 480 Columns (Segments): 64, 128, 160, 240, 320
Data rotation function		Horizontal flip, vertical flip, horizontal and vertical flip, 90-degree rotation (supported for QVGA size, 65536 colors only)	
PIP function support		A sub window can be inserted.	
Source data bus width (SRAM, SDRAM)		16 bits (32 bits: internal RAM) , not support external 32-bit	16 bits (32 bits: internal RAM), not support external 32-bit
Destination data bus width (LCD driver)		8 to 16 bits	8 bits
Maximum transfer rate (VRAM read)		1-clk / 4byte at internal RAM	
External Pins	LCD driver data bus: LD15 to LD0 pins	To be connected to LCD driver data bus. ・ 8-bit mode: LD7 to LD0 ・TFT mode: LD15 to LD0	
	LCP0 pin	Data shift clock for TFT source driver	Shift clock pulse output pin 0. To be connected to column driver's CP pin. The LCD driver latches the data bus value on the falling edge of this pin.
	LHSYNC pin	Vertical shift clock for TFT gate driver	Latch pulse output pin. To be connected to the LCD driver's LP pin. The display data in the LCD driver's output line register is updated on the rising edge of this pin.
	LLOAD pin	Enable signal for TFT source driver to load data to TFT panel	N/A
	LGOE0 to LGOE2 pins	Adjustment signal for TFT gate driver's gate control signal	N/A
	LFR pin	LCD alternate signal output pin. To be connected to column/row driver's FR pin.	LCD alternate signal output pin. To be connected to column/row driver's FR pin.
	LVSYNC pin	This signal indicates the start of shift clock capture by TFT gate driver.	Frequency that sets LCD refresh rate
	LDIV pin	This signal indicates the inversion of data. To be connected to TFT source driver having the data inversion function.	N/A

3.20.2 SFRs

LCDMODE0 Register

LCDMODE0 (0280H)

	7	6	5	4	3	2	1	0
bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
Read/Write	R/W							
Reset State	0	0	1	1	0	0	0	0
Function	Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= "0" 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= "1" 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		Mode selection 0000: Reserved 1000: STN (64K-color) 0001: SR (mono) 1001: Reserved 0010: SR (4-gray) 1010: TFT (256-color) 0011: Reserved 1011: TFT (4096-color) 0100: SR (16-gray) 1100: TFT (64K-color) 0101: SR (64-gray) 1101: Reserved 0110: STN (256-color) 1110 : TFT (mono) 0111:STN (4096-color) 1111: Reserved			

Note: When SDRAM is used as the LCDC's display RAM, it can only be accessed by "burst 1-clock access".

LCDMODE1 Register

LCDMODE1 (0281H)

	7	6	5	4	3	2	1	0
bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
Read/Write	R/W						W	
Reset State	0	0	0	0	0	0	0	0
Function	Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Horizontal & vertical flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNCREAR Edge	LD bus Trance Speed 0: normal 1: 1/3

Note: <LDINV>= "1" inverts all output data on the LD bus. However, the LDIV signal that indicates the inversion of output data by auto bus inversion remains unchanged.

LCD Size Setting Register

LCDSIZE (0284H)

	7	6	5	4	3	2	1	0
bit Symbol	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	Common setting 0000: Reserved 1000: 320 0001: 64 1001: 480 0010: 96 1010: Reserved 0011: 120 1011: Reserved 0100: 128 1100: Reserved 0101: 160 1101: Reserved 0110: 200 1110: Reserved 0111: 240 1111: Reserved				Segment setting 0000: Reserved 1000: Reserved 0001: 64 1001: Reserved 0010: 128 1010: Reserved 0011: 160 1011: Reserved 0100: 240 1100: Reserved 0101: 320 1101: Reserved 0110: 480 1110: Reserved 0111: 640 1111: Reserved			

Note1: Although the TMP92CF29A contains 144Kbytes of RAM that can be used as display RAM, it may not be enough depending on display size and color mode.

Note2: When TFT monochrome mode is used, 64, 128 and 160 segment setting cannot use.

LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W			R/W		R/W		
Reset State	0	0	0	0		0	0	0
Function	PIP function 0:Disable 1:Enable	Segment data 0: Normal 1: Always output "0"	Frame divide setting 0: Disable 1: Enable	Always write "0"		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0 (Note) 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

LCD Control 1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
Read/Write	R/W						R/W	
Reset State	1	0	1	0			0	0
Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

LCD Control 2 Register

LCDCTL2 (0287H)

	7	6	5	4	3	2	1	0
bit Symbol	LGOE2P	LGOE1P	LGOE0P					
Read/Write	R/W							
Reset State	0	0	0					
Function	LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					

Divide FRM 0 Register

LCDDVM0 (0283H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 3-0)				LHSYNC DVM (bits 3-0)			

Divide FRM 1 Register

LCDDVM1 (0288H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 7-4)				LHSYNC DVM (bit 7-4)			

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP (028AH)	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7–0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

LCD LVSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDVSP (028CH)	bit Symbol	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LVSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028DH)	bit Symbol							LVP9	LVP8
	Read/Write							W	
	Reset State							0	0
	Function							LVSYNC period (bits 9-8)	

LCD LVSYNC Pre Pulse Register

		7	6	5	4	3	2	1	0
LCDPRVSP (028EH)	bit Symbol		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
	Read/Write		W						
	Reset State		0	0	0	0	0	0	0
	Function		Front dummy LVSYNC (bits 6-0)						

LHSYNC Delay Register

LCDHSDLY (028FH)

	7	6	5	4	3	2	1	0
bit Symbol		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		LHSYNC delay (bits 6-0)						

LLOAD Delay Register

LCDLDDLY (0290H)

	7	6	5	4	3	2	1	0
bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
Read/Write	R/W	W						
Reset State	0	0	0	0	0	0	0	0
Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

LGOE0 Delay Register

LCDO0DLY (0291H)

	7	6	5	4	3	2	1	0
bit Symbol		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE0 delay (bits 6-0)						

LGOE1 Delay Register

LCDO1DLY (0292H)

	7	6	5	4	3	2	1	0
bit Symbol		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE1 delay (bits 6-0)						

LGOE2 Delay Register

LCDO2DLY (0293H)

	7	6	5	4	3	2	1	0
bit Symbol		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE2 delay (bits 6-0)						

LHSYNC width Register

LCDHSW (0294H)		7	6	5	4	3	2	1	0
	bit Symbol	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC width (bits 7-0)							

LLOAD width Register

LCDLDW (0295H)		7	6	5	4	3	2	1	0
	bit Symbol	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LLOAD width (bits 7-0)							

LGOE0 width Register

LCDHO0W (0296H)		7	6	5	4	3	2	1	0
	bit Symbol	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE0 width (bits 7-0)							

LGOE1 width Register

LCDHO1W (0297H)		7	6	5	4	3	2	1	0
	bit Symbol	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE1 width (bits 7-0)							

LGOE2 width Register

LCDHO2W (0298H)		7	6	5	4	3	2	1	0
	bit Symbol	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 7-0)							

signal width Bit8,9 Register

LCDHWB8 (0299H)		7	6	5	4	3	2	1	0
	bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

LCD Main Area Start Address Register

		7	6	5	4	3	2	1	0
LSAML (02A0H)	bit Symbol	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	
	Function	LCD main area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSAMM (02A1H)	bit Symbol	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMSA9	LMSA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD main area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSAMH (02A2H)	bit Symbol	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
	Read/Write	R/W							
	Reset State	0	1	0	0	0	0	0	0
	Function	LCD main area start address (A23-A16)							

Note: When assigned internal RAM as VRAM, A1 signal cannot be used. Every 4bytes setting is needed.

LCD Sub Area Start Address Register

		7	6	5	4	3	2	1	0
LSASL (02A4H)	bit Symbol	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	
	Function	LCD sub area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSASM (02A5H)	bit Symbol	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSASH (02A6H)	bit Symbol	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
	Read/Write	R/W							
	Reset State	0	1	0	0	0	0	0	0
	Function	LCD sub area start address (A23-A16)							

Note: When assigned internal RAM as VRAM, A1 signal cannot be used. Every 4bytes setting is needed.

LCD Sub Area HOT Point Register (X-dir)

		7	6	5	4	3	2	1	0
LSAHX (02A8H)	bit Symbol	SAHX7	SAHX6	SAHX5	SAHX4	SAHX3	SAHX2	SAHX1	SAHX0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02A9H)	bit Symbol							SAHX9	SAHX8
	Read/Write							R/W	
	Reset State							0	0
	Function							LCD sub area HOT point (9-8)	

LCD Sub Area HOT Point Register (Y-dir)

		7	6	5	4	3	2	1	0
LSAHY (02AAH)	bit Symbol	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02ABH)	bit Symbol								SAHY8
	Read/Write								R/W
	Reset State								0
	Function								LCD sub area HOT point (8)

LCD Sub Area Display Segment Size Register

		7	6	5	4	3	2	1	0
LSASS (02ACH)	bit Symbol	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area segment size (7-0)							
		7	6	5	4	3	2	1	0
(02ADH)	bit Symbol							SAS9	SAS8
	Read/Write							R/W	
	Reset State							0	0
	Function							LCD sub area segment size (9-8)	

LCD Sub Area Display Common Size Register

		7	6	5	4	3	2	1	0
LSACS (02AEH)	bit Symbol	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area common size (7-0)							
		7	6	5	4	3	2	1	0
(02AFH)	bit Symbol								SAC8
	Read/Write								R/W
	Reset State								0
	Function								LCD sub area common size (8)

3.20.3 Description of Operation

3.20.3.1 Outline

After the required settings such as the operation mode, display data memory address, color mode, and LCD size are specified, the start register is set to start the LCDC operation.

The LCDC issues a bus request to the CPU. When the bus is granted, the LCDC reads data of the display size from the display RAM, stores the data in the FIFO buffer in the LCDC, and then returns the bus to the CPU.

The display data in the FIFO buffer is transferred to the LCD driver via a dedicated bus (LD pin). At this time, control pins (such as LCP0) that are connected to the LCD driver also output specified waveforms in synchronization with the transfer of display data.

Note: While display RAM data is being read, the CPU operation is halted by the internal BUSREQ signal. Therefore, the CPU stop time must be taken into account in programming.

External SDRAM, SRAM, or internal RAM (144 Kbytes) can be used as the display RAM. Since the internal RAM allows very fast accesses (32-bit bus, 2-1-1-1 read/write), it enables data transfer to the LCD driver (DMA operation) with the minimum CPU stop time. Using the internal RAM also greatly reduces power consumption during LCD display.

3.20.3.2 Display Memory Mapping

Since the number of bits needed to display one pixel varies even for the same display size depending on the selected color mode, the required display RAM size also varies with each color mode. (The color mode can be selected from a range of monochrome to 65536 colors.)

In monochrome mode, one pixel of display data corresponds to one bit of display RAM data. Likewise, the number of display RAM data used for displaying one pixel in each color mode is as follows:

4-grayscale	1 pixel = 2 bits
16-grayscale	1 pixel = 4 bits
64-grayscale	1 pixel = 6 bits
STN 256-color	1 pixel = 8 bits
STN 4096-color	1 pixel = 12 bits
STN 65536-color	1 pixel = 16 bits

For example, a 320-segment × 240-common display in 4-grayscale mode requires 19200 bytes of display RAM space (320 × 240 × 2 = 152600 bits = 19200 bytes).

For details, refer to “Memory Map Image and Data Output in Each Display Mode” later in this chapter.

3.20.3.3 Restriction of Display Memory

This LCD controller is supported for display RAM as internal RAM, external SRAM and external SDRAM. However in case of using SDRAM for display RAM, there is one restriction as follows.

Condition & Restrictions

a) Use for SDRAM as VRAM of LCD controller

and

b) Use DMAC operation

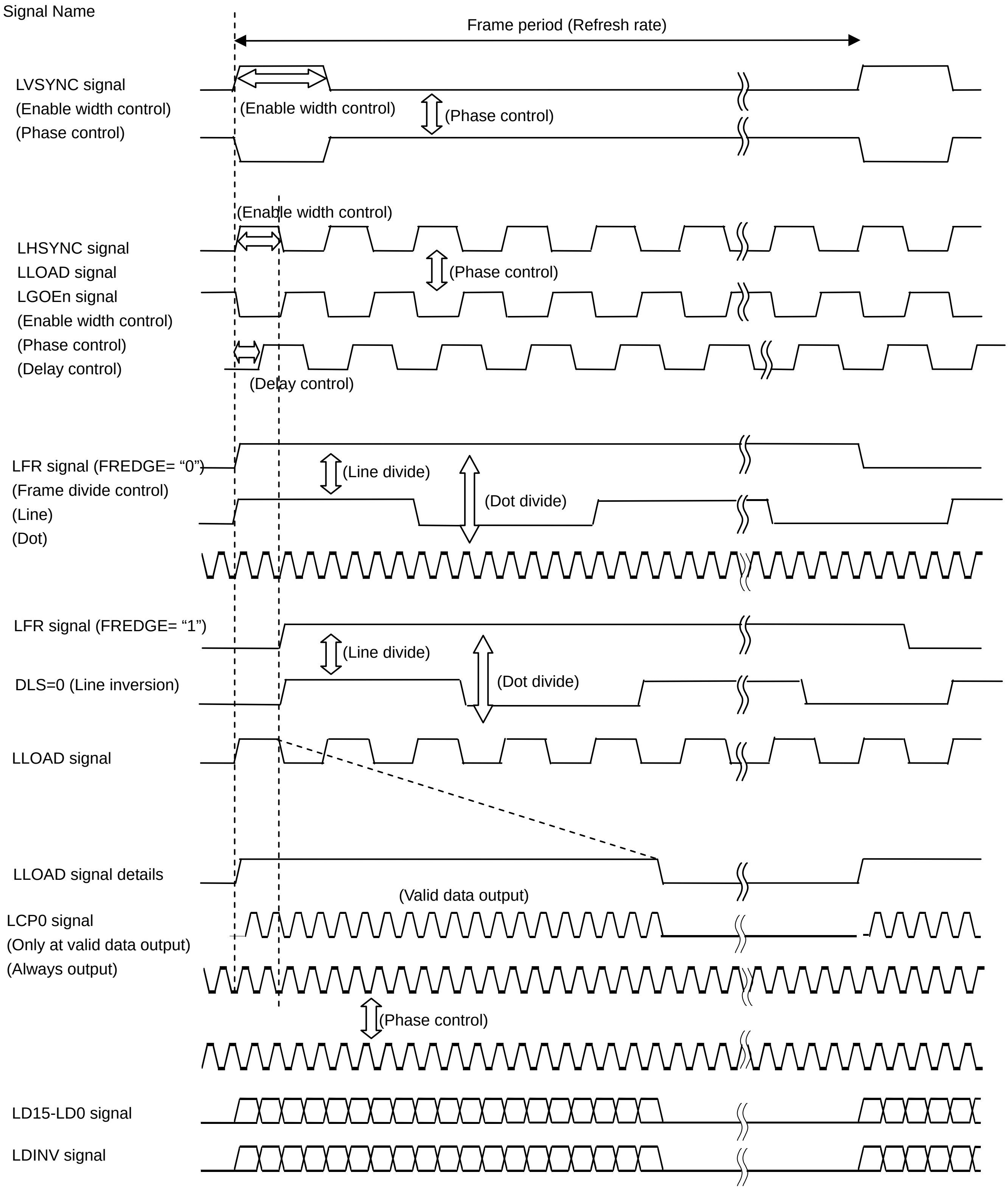
In case of above condition, Need to set SDACR<SPRE>= “1”.

Please refer the chapter of SDRAM controller about SDRAM specification in detail.

3.20.3.4 Basic Operation

The following diagram shows the basic timings of the waveforms generated by the LCDC and adjustable elements. The adjustable elements for each signal include enable time, phase, and delay time.

The signals used and their connections and settings vary with the LCD driver type (STN/TFT) and specifications to be used.



3.20.3.5 Reference Clock LCP0

LCP0 is used as the reference clock for all the signals in the LCDC.

This section explains how to set the frequency (period) of the LCP0 signal.

The LCP0 clock speed (LD bus transfer speed) is determined by selecting TFT or STN and setting LCDMODE0<SCPW1:0> and LCDMODE1<SWPW2>. The clock speed should be selected to meet the characteristics of the LCD driver to be used.

The LCP0 period can be selected from four types: f_{SYS}/2, f_{SYS}/4, f_{SYS}/8, f_{SYS}/16, f_{SYS}/24 and f_{SYS}/48.

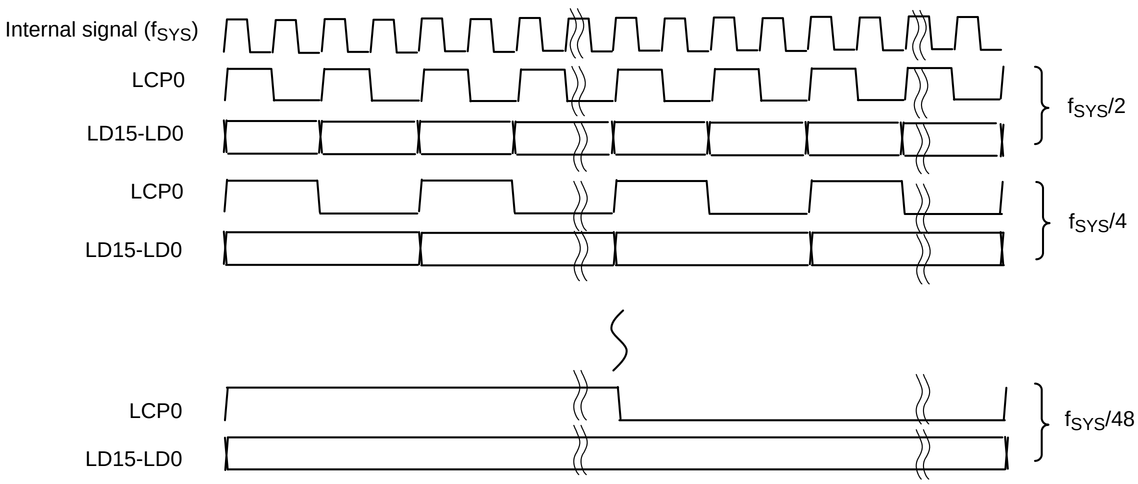


Figure 3.20.1 LCP Frequency Selection

Minimum speed

The LCP0 period needs to be short enough to prevent the next line signal from overlapping the current line signal.

The transfer speed of display data must be set to suit the refresh rate; otherwise data cannot be transferred properly. Set the data transfer speed so that each transfer completes within the LHSYNC period.

STN monochrome/grayscale	:	Segment size / 8 × LCP0 [s: period] < LHSYNC [s: period] STN color
STN color	:	Segment size × 3 / 8 LCP0 [s: period] < LHSYNC [s: period]
TFT	:	Segment size × LCP0 [s: period] < LHSYNC [s: period]

Maximum speed

If the LCP0 period is too short, the data to be transferred to the LCD driver cannot be prepared in time, causing wrong data to be transferred. The maximum transfer speed is limited by the operation mode and display RAM type (bus width, wait condition, and so on). If the data rotation function is used, the transfer speed must be slower.

LCP0 Setting Range Table

Conditions : f_{SYS} = 60MHz
Display size : (color) up to 320 × 320
Display size : (monochrome/grayscale) up to 640 × 480

Note: This table shows the range of LCP0 settings that can be made under the conditions shown above. If the CPU clock speed, display size, or refresh rate is changed, the LCP0 range also changes.

Display RAM / Display Mode	Internal RAM	SDRAM	External SRAM (0 waits)	External SRAM (N waits)
STN monochrome Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 tof_{SYS}/16 (up to 2 waits) f_{SYS}/8 to f_{SYS}/16 (up to 6 waits) f_{SYS}/16 (up to 14 waits)
STN 4-grayscale Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/8 (up to 2 waits) f_{SYS}/8 (up to 6 waits)
STN 16-grayscale Refresh cycle = 140 Hz	f_{SYS}/2 to f_{SYS}/8	f_{SYS}/2 to f_{SYS}/8	f_{SYS}/4 to f_{SYS}/8	f_{SYS}/8 to f_{SYS}/16 (up to 2 waits) f_{SYS}/16 (up to 6 waits)
STN 64-grayscale Refresh cycle = 200 Hz	f_{SYS}/4	f_{SYS}/4	f_{SYS}/4	f_{SYS}/4 (up to 1 wait)
STN 256-color Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16	f_{SYS}/8 to f_{SYS}/16 (up to 2 waits) f_{SYS}/16 (up to 6 waits)
STN 4K-color Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16 (up to 2 waits) f_{SYS}/8 to f_{SYS}/16 (up to 6 waits) f_{SYS}/16 (up to 14 waits)
STN 64K-color Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16	f_{SYS}/8 to f_{SYS}/16	f_{SYS}/16 (up to 3 waits)
STN 64K-color Refresh Cycle = 70 Hz + rotation operation	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to $f_{SYS/}$16	f_{SYS}/8 to f_{SYS}/16	f_{SYS}/16 (up to 3 waits)
TFT 4K-color Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 To f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16 (up to 2 waits) f_{SYS}/8 to f_{SYS}/16 (up to 6 waits) f_{SYS}/16 (up to 14 waits)
TFT 64K-color Refresh cycle = 70 Hz	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16 (up to 2 waits) f_{SYS}/8 to f_{SYS}/16 (up to 6 waits) f_{SYS}/16 (up to 14 waits)
TFT 64K-color + rotation operation	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/2 to f_{SYS}/16	f_{SYS}/4 to f_{SYS}/16 (up to 2 waits) f_{SYS}/8 to f_{SYS}/16 (up to 6 waits) f_{SYS}/16 (up to 14 waits)

Example 1: When f_{SYS} = 10 MHz, STN mode, LCDMODE0<SCPW1:0> = “01”

Internal reference clock LCP0 = f_{SYS} / 8 = 10 MHz / 8 = 1.25 [MHz]

LCP0 period = 1 / 1.25 [MHz] = 0.8 [μs]

Example 2: when f_{SYS} = 60 MHz, TFT mode, LCDMODE0<SCPW1:0> = “11”

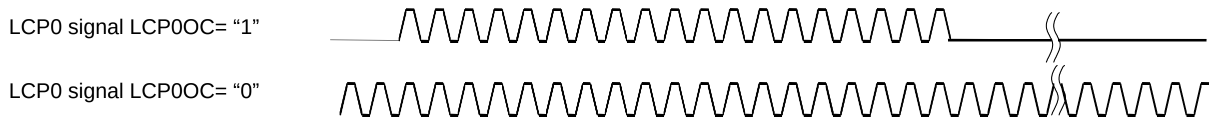
Internal reference clock LCP0 = f_{SYS} / 16 = 60 MHz / 16 = 3.75 [MHz]

LCP0 period = 1 / 3.75 [MHz] = 266 [ns]

LCDMODE0 Register

		7	6	5	4	3	2	1	0
LCDMODE0 (0280H)	bit Symbol	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
	Read/Write	R/W							
	Reset State	0	0	1	1	0	0	0	0
	Function	Display RAM 00: Internal RAM(32-bit) 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= “0” 00: 2-clk 01: 4-clk 10: 8-clk 11: 16-clk SCPW2= “1” 00: 6-clk 01: 12-clk 10: 24-clk 11: 48-clk		Mode selection 0000: Reserved 0001: SR (mono) 0010: SR (4-gray) 0011: Reserved 0100: SR (16-gray) 0101: SR (64-gray) 0110: STN (256-color) 0111:STN (4096-color) 1000: STN (64K-color) 1001: Reserved 1010: TFT (256-color) 1011: TFT (4096-color) 1100: TFT (64K-color) 1101: Reserved 1110 : TFT (mono) 1111: Reserved			

LCDCTL0 <LCP0OC> is used to control the output timing of the LCP0 signal. When <LCP0OC>= “0”, the LCP0 signal is always output. When <LCP0OC>= “1”, the LCP0 signal is output only when valid data is output.

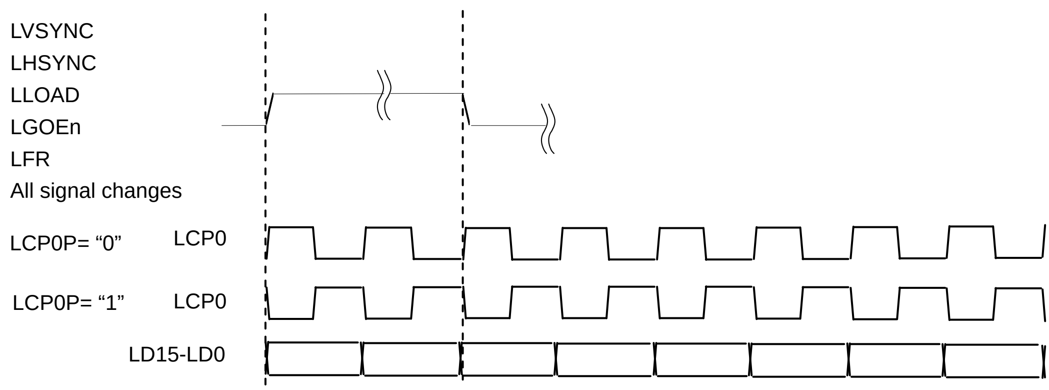


LCD Control 0 Register

LCDCTL0 (0285H)		7	6	5	4	3	2	1	0
	bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
	Read/Write	R/W			R/W		R/W		
	Reset State	0	0	0	0		0	0	0
	Function	PIP function 0:Disable 1:Enable	Segment data 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0(Note 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

The phase of the LCP0 signal can be inverted by the setting of LCDCTL1<LCP0P>.



LCD Control 1 Register

LCDCTL1 (0286H)		7	6	5	4	3	2	1	0
	bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
	Read/Write	R/W						R/W	
	Reset State	1	0	1	0			0	0
	Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

3.20.3.6 Refresh Rate

The period of the horizontal synchronization signal LHSYNC is defined as the product of the value set in LCDHSP<LH15:0> and the LCP0 clock period.

The value to be set in LCDHSP<LH15:0> is obtained as follows:

<u>TFT</u>

Segment size + number of dummy clocks (*)

<u>STN</u>

Monochrome/grayscale : (Segment size / 8) + number of dummy clocks (*)

Color : (Segment size × 3 / 8) + number of dummy clocks (*)

LHSYNC [s: period] = LCP0 [s: period] × (<LH15:0> + 1)

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
(028AH)	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7–0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

The period of the vertical synchronization signal LVSYNC is defined as the product of the value set in LCDVSP<LV9:0> and the LHSYNC period.

The value to be set in LCDVSP<LV9:0> is obtained as follows:

TFT

Common size + number of dummy clocks (*)

STN

Common size + number of dummy clocks (*)

(A minimum of one dummy clock must be inserted in the back porch.)

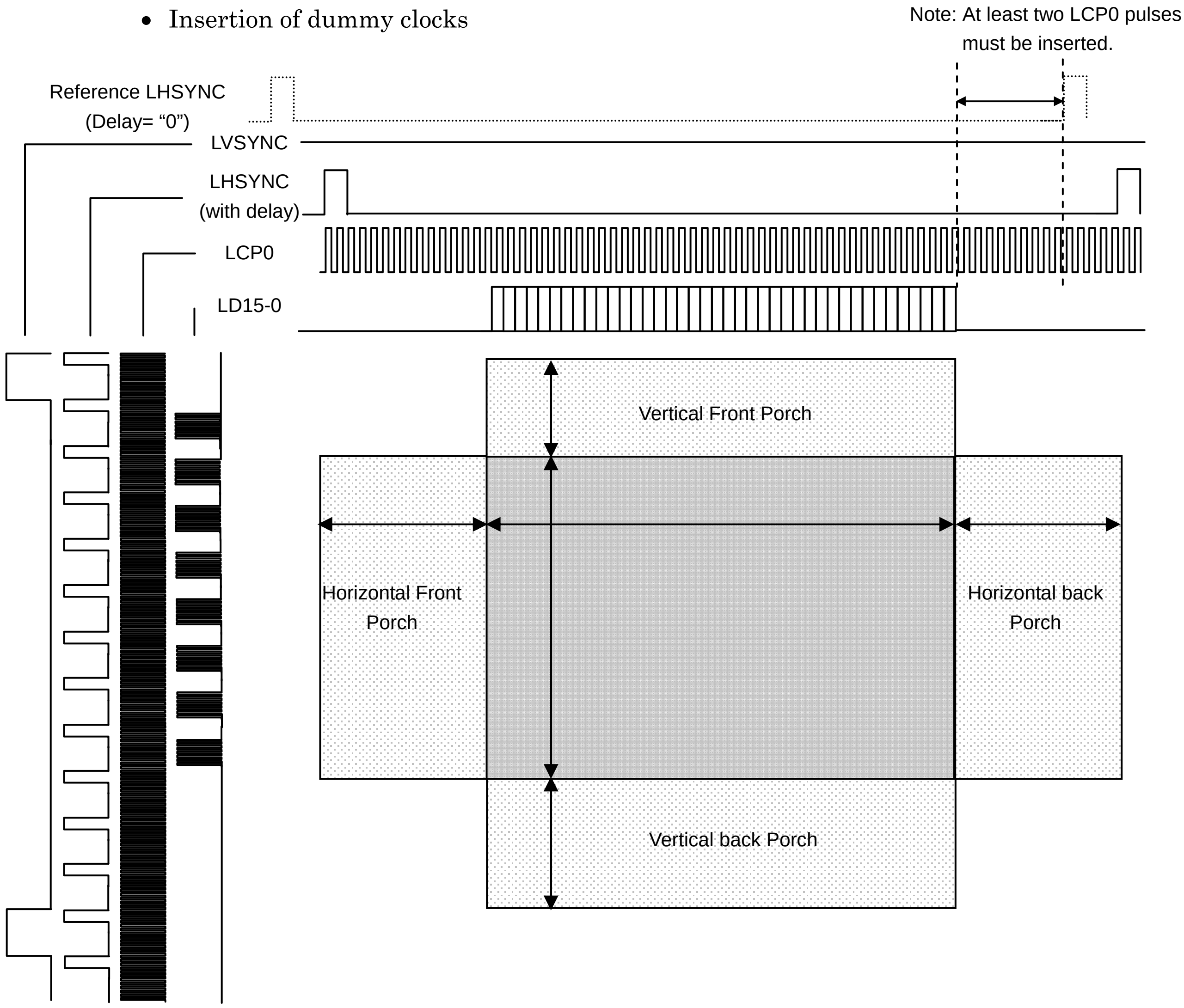
LVSYNC [s: period] = LHSYNC [s: period] × (<LV9:0> + 1)

= LCP0 [s: period] × (<LH15:0> + 1) × (<LV9:0> + 1)

LCD LVSYNC Pulse Register

<table>
<tr><td></td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>LCDVSP</td><td>bit Symbol</td><td>LVP7</td><td>LVP6</td><td>LVP5</td><td>LVP4</td><td>LVP3</td><td>LVP2</td><td>LVP1</td><td>LVP0</td></tr>
<tr><td>(028CH)</td><td>Read/Write</td><td colspan="8">W</td></tr>
<tr><td></td><td>Reset State</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td></td><td>Function</td><td colspan="8">LVSYNC period (bits 7-0)</td></tr>
<tr><td></td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>(028DH)</td><td>bit Symbol</td><td></td><td></td><td></td><td></td><td></td><td></td><td>LVP9</td><td>LVP8</td></tr>
<tr><td></td><td>Read/Write</td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">W</td></tr>
<tr><td></td><td>Reset State</td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td></tr>
<tr><td></td><td>Function</td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">LVSYNC period
(bits 9-8)</td></tr>
</table>

- Insertion of dummy clocks



The above is a conceptual diagram showing the data (LD15-0), shift clock (LCP0), horizontal synchronization signal (LHSYNC), and vertical synchronization signal (LVSYNC) on the LCD panel.

The front porch and back porch as shown above should be taken into consideration in setting LCDHSP<LH15:0> and LCDVSP<LV9:0> explained earlier.

Note 1: The horizontal back porch must be set so that “data transfer” plus “LCP0 × 2 clocks” are completed within one period of the reference clock LHSYNC (with 0 delay), as defined by the following equation:
Delay time (LLOAD) + number of data transfer times + 2 < LHSYNC (LCP0 pulse count)

Note 2: The vertical back porch must have a minimum of one dummy clock.

(*) TFT driver

The recommended number of dummy clocks is specified by each TFT driver (or LCD module). Refer to the specifications of the TFT driver (LCD module) to be used.

(*) STN driver

For an STN driver, the refresh rate can be set accurately by adjusting the value of the horizontal back porch. If the desired refresh rate cannot be obtained by the horizontal back porch, it can be further adjusted by the vertical back porch. For details, refer to the setting example to be described later in this section.

- Setting method

The front dummy LHSYNC (vertical front porch) not accompanied by valid data in the total of LHSYNC period in the LVSYNC period is defined by the value set in LCDPRVSP<PLV6:0>.

Front dummy LHSYNC (vertical front porch) = <PLV6:0>

The back dummy LHSYNC (vertical back porch) is defined as follows:

(<LVP9:0>+1) – (valid LHSYNC: common size) – (front dummy LHSYNC: <PLV6:0>)

The vertical back porch must have a minimum of one dummy clock.

The front dummy LCP0 (horizontal front porch) not accompanied by valid data in the total number of LCP0 clocks in the LHSYNC period is defined by the value set in LCDLDDLY<LDD6:0>.

Front dummy LCP0 (horizontal front porch) = <LDD6:0>

The back dummy LCP0 (horizontal back porch) is defined as follows:

(<LH15:0> + 1) – (Valid LCP0: segment size) – (Front dummy LCP0: <LDD6:0>)

Note 1: The back dummy LCP0 (horizontal back porch) must have a minimum of two LCP0 clocks.

Note 2: The delay time that is set in LCDLDDLY<LDD6:0> is counted based on LHSYNC (with 0 delay).

LLOAD Delay Register

		7	6	5	4	3	2	1	0
LCDLDDLY (0290H)	bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
	Read/Write	R/W	W						
	Reset State	0	0	0	0	0	0	0	0
	Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

Example 1) Setting the refresh rate to 200 Hz under the following conditions:

f_{SYS} = 30 MHz, STN mode, 320-segment × 240-common, 4096-color display,
LCDMODE0<SCPW1:0> = “00”

Internal reference clock LCP0 = f_{SYS} / 4 = 30 [MHz] / 4 = 7.5 [MHz]
Therefore, LCP0 period = 1 / 7.5 [MHz] = 0.133 [μs]

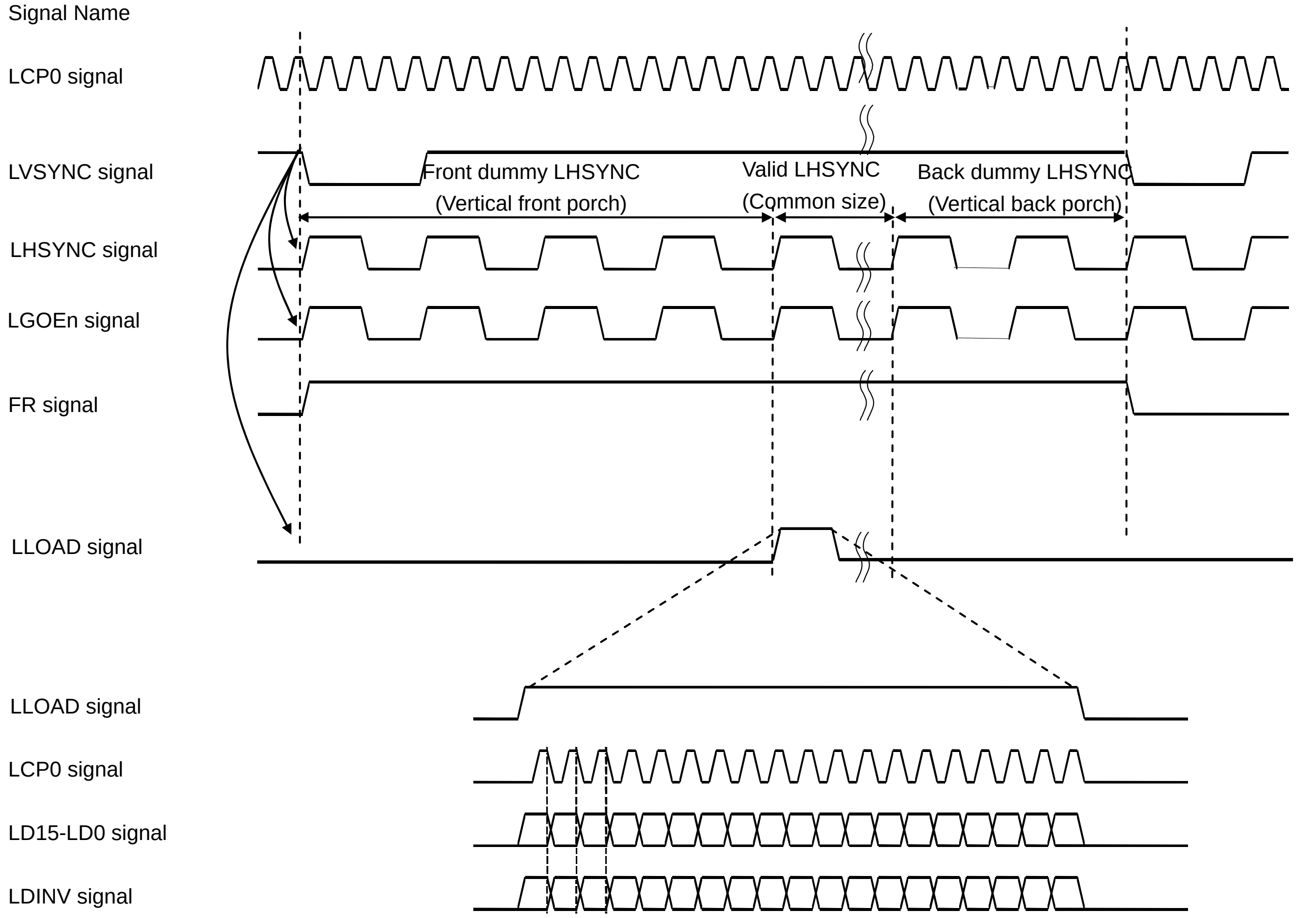
Condition 1: Refresh rate = 200 Hz, Refresh cycle = 5 [ms]
Condition 2: LH = <LH15:0> ≥ (320×3/8) − 1 = 119
Condition 3: LV = <LVP9:0> ≥ 240 − 1

When <LVP9:0> = 239 (minimum value):

LVSYNC [s: period] = LHSYNC [S: period] × ((LV9:0) + 1)
= LCP0 [S: period] × ((LH15:0) + 1) × ((LV9:0) + 1)

5 [ms] = (1 / 7.5 [MHz]) × (LH + 1) × 240
LH + 1 = $(5 \times 10^{-3}) \times (7.5 \times 10^{6}) / 240$
= 156.25

3.20.3.7 Signal Settings



The above diagram shows the typical timings of the signals controlled by the LCDC. This section explains how to control each of these signals.

(1) LVSYNC Signal

The period of the vertical synchronization signal LVSYNC indicates the time for each screen update (refresh rate). The LVSYNC period is defined as an integral multiple of the period of the horizontal synchronization signal LHSYNC.

The LVSYNC period is calculated as the product of the value set in LCDVSP<LV 9:0> and the LHSYNC period. The value to be set in LCDVSP<LV9:0> should be "common size + number of dummy clocks" or larger for TFT and STN.

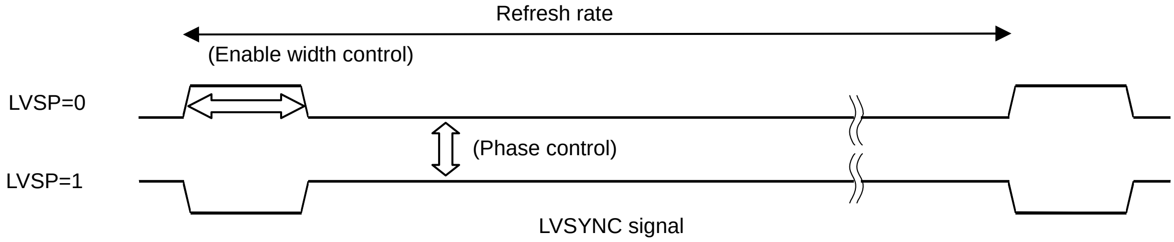
LVSYNC [s: period] = LHSYNC [s: period] × (<LVP9:0> + 1)
= LCP0 [s: period] × (<LH15:0> + 1) × (<LVP9:0> + 1)

LCD LVSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDVSP (028CH)	bit Symbol	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LVSYNC period (bits 7-0)							
		7	6	5	4	3	2	1	0
(028DH)	bit Symbol							LVP9	LVP8
	Read/Write							W	
	Reset State							0	0
	Function							LVSYNC period (bits 9-8)	

The enable width of the LVSYNC signal can be specified as 1 clock, 2 clocks, or 3 clocks of LHSYNC in LCDCTL1<LVSW1:0>.

The phase of the LVSYNC signal can be inverted by the setting of LCDCTL1 <LVSP>.



LCD Control 1 Register

		7	6	5	4	3	2	1	0
LCDCTL1 (0286H)	bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
	Read/Write	R/W						R/W	
	Reset State	1	0	1	0			0	0
	Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

(2) LHSYNC Signal

The period of the horizontal synchronization signal LHSYNC corresponds to one line of display. The LHSYNC period is defined as an integral multiple of the reference clock signal LCP0.

The LHSYNC period is defined as the product of the value set in LCDHSP<LH15:0> and the LCP0 clock period. The value to be set in LCDHSP<LH15:0> should be "segment size + number of dummy clocks" or larger for TFT. In the case of STN, the minimum value of LCDHSP<LH15:0> is:

Monochrome/grayscale : (Segment size / 8) + number of dummy clocks

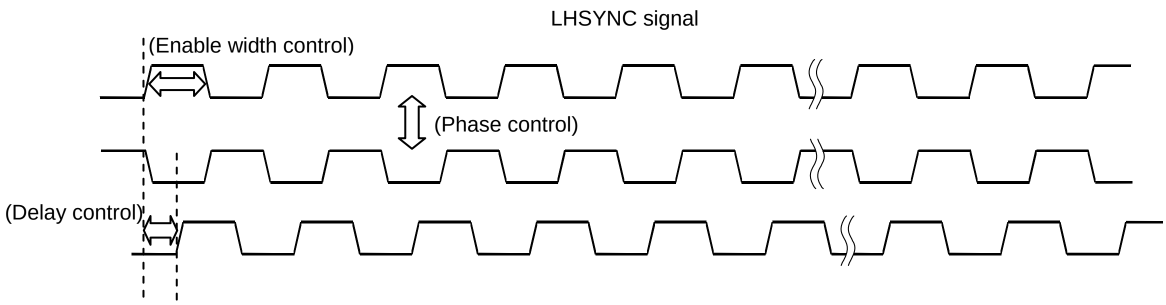
Color : (Segment size × 3 / 8) + number of dummy clocks

LHSYNC [s: period] = LCP0 [s: period] × (<LH15:0> + 1)

LCD LHSYNC Pulse Register

		7	6	5	4	3	2	1	0
LCDHSP	bit Symbol	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
(028AH)	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 7–0)							
		7	6	5	4	3	2	1	0
(028BH)	bit Symbol	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC period (bits 15-8)							

The enable width of the LHSYNC signal can be specified by LCDHSW<HSW9:0>. It is also possible to set the delay time for the LVSYNC signal in units of LCP0 pulses.

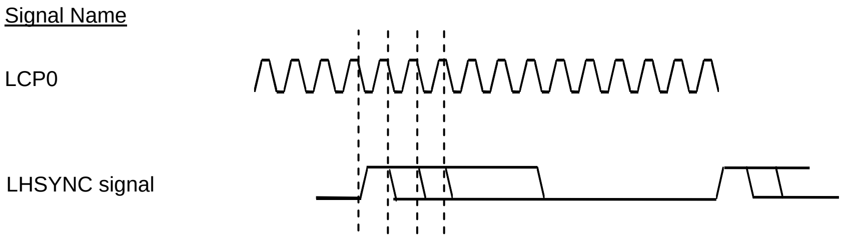


The enable width of the LHSYNC signal is set using LCDHSW<HSW8:0>. It can be specified in a range of 1 to 512 pulses of the LCP0 clock.

The enable width is represented by the following equation:

Enable width = <HSW8:0> + 1

Thus, when LCDHSW<HSW8:0> is set to “0”, the enable width is set as one pulse of the LCP0 clock.



High width setting
LCP0 clock = 1, 2, 3 … 512 pulses

LHSYNC width Register

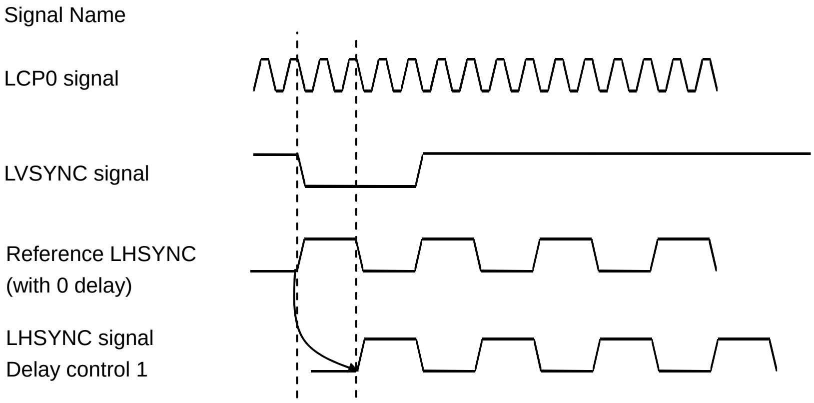
		7	6	5	4	3	2	1	0
LCDHSW (0294H)	bit Symbol	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LHSYNC width (bits 7-0)							

Signal width Bit8,9 Register

		7	6	5	4	3	2	1	0
LCDHWB8 (0299H)	bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

As shown in the diagram below, delay time of 0 to 127 pulses of the LCP0 clock can be inserted in the LHSYNC signal.

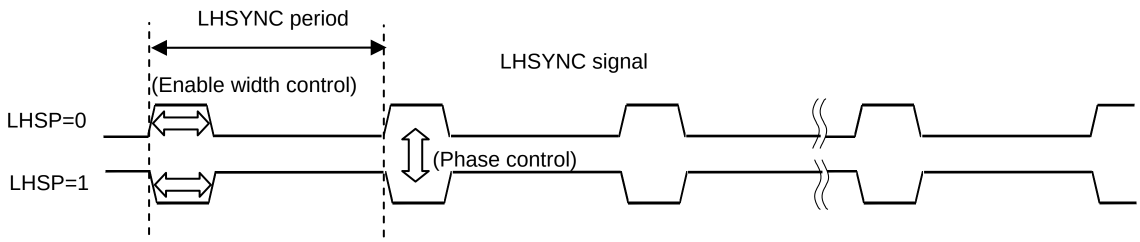
Delay time = <HSD6:0>



LHSYNC Delay Register

		7	6	5	4	3	2	1	0
LCDHSDLY (028FH)	bit Symbol		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
	Read/Write		W						
	Reset State		0	0	0	0	0	0	0
	Function		LHSYNC delay (bits 6-0)						

The phase of the LHSYNC signal can be inverted by the setting of LCDCTL1 <LVSP>.

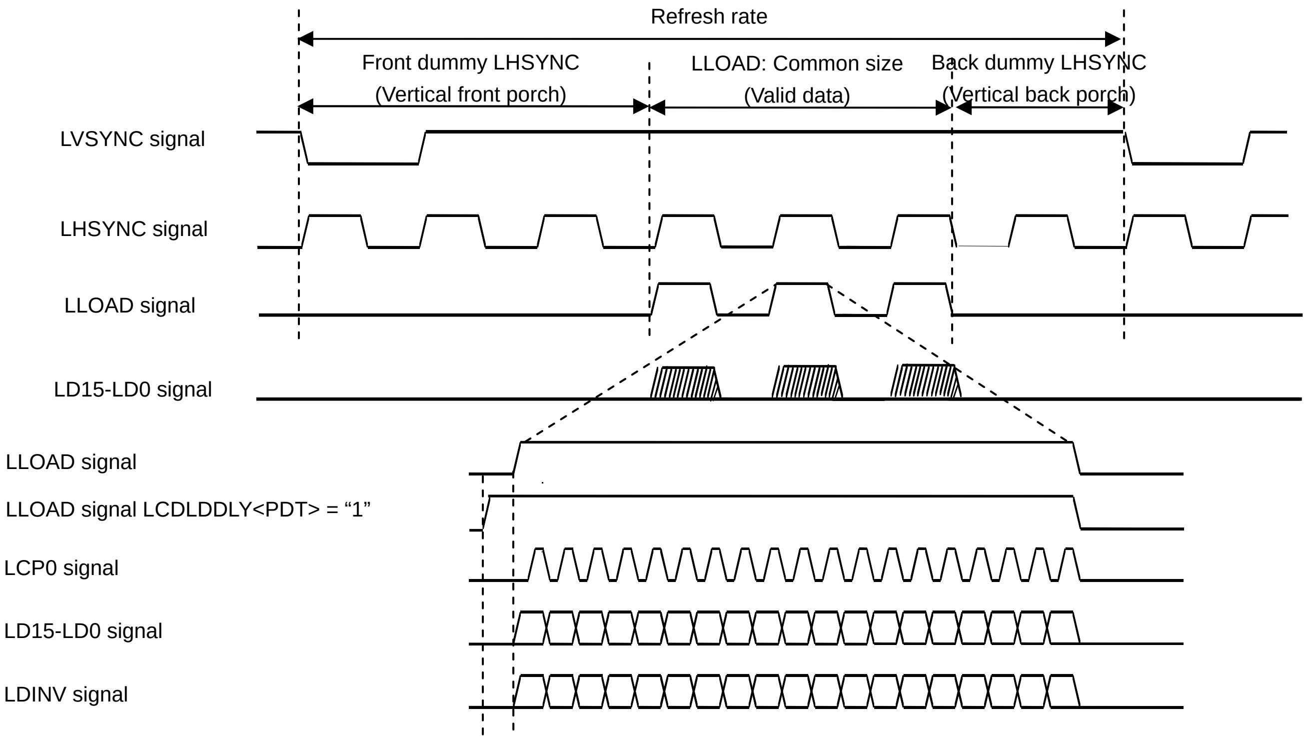


LCD Control 1 Register

		7	6	5	4	3	2	1	0
LCDCTL1 (0286H)	bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
	Read/Write	R/W						R/W	
	Reset State	1	0	1	0			0	0
	Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

(3) LLOAD Signal

The LLOAD signal is used to control the timing for the LCD driver to receive display data. The period of the LLOAD signal synchronizes to one line of display. It is defined as an integral multiple of the reference clock LCP0.



The LHSYNC signal and LLOAD signal differs in that the LHSYNC signal is output all the time whereas the LLOAD signal is output only at valid data lines (commons).

Display data is output in synchronization with the LLOAD signal. Therefore, if a delay is inserted in the LLOAD signal through the LCDLDDLY register, data output is also delayed.

Also note that when LCDLDDLY<PDT>=1, data is output one LCP0 clock later than the LLOAD signal.

LCDLDDLY<PDT>= "0": Data is output in synchronization with the LLOAD signal.

LCDLDDLY<PDT>= "1": Data is output one LCP0 clock later than the LLOAD signal.

The delay time for the LLOAD signal is controlled based on LCDLDDLY<PDT>= "1". Therefore, even if the delay time is set to "0" with LCDLDDLY<PDT>= "0", the LLOAD signal is output with a delay of one LCP0 clock. Be careful about this point.

The number of pulses in the front dummy LHSYNC (vertical front porch) is specified by LCDPRVSP<PLV6:0>. This delay time can be set in a range of 0 to 127 pulses of the LCP0 clock.

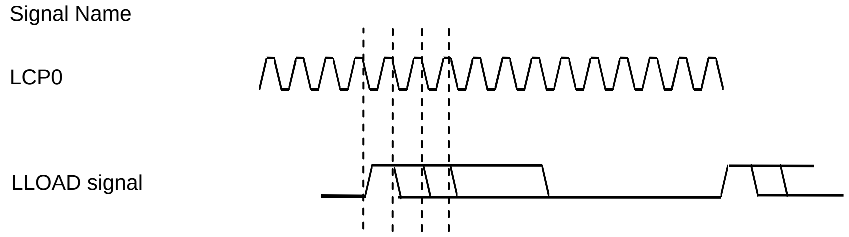
Front dummy LHSYNC = <PLV6:0>

LCD LVSYNC Pre Pulse Register

LCDPRVSP (028EH)		7	6	5	4	3	2	1	0
	bit Symbol		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
	Read/Write		W						
	Reset State		0	0	0	0	0	0	0
	Function		Front dummy LVSYNC (bits 6-0)						

The back dummy LHSYNC (vertical back porch) is defined as follows:

(<LVP9:0> + 1) – (valid LHSYNC: common size) – (front dummy LHSYNC: <PLV6:0>)



High width setting
LCP0 clock = 1, 2, 3 … 1023 pulses (<PDT>=0) / 1024 pulses (<PDT>=1)

Note: The vertical back porch must be set to “1” or longer in all the cases (STN/TFT).

The enable width of the LLOAD signal is determined depending on the LCDCTL0<LCP0OC> setting, as shown below.

LCDCTL0<LCP0OC> = “0” : Output at setting value in (LCDDLW) <LDW9:0>

LCDCTL0<LCP0OC> = “1” : Output at valid data

LCD Control 0 Register

LCDCTL0 (0285H)		7	6	5	4	3	2	1	0
	bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
	Read/Write	R/W			R/W		R/W		
	Reset State	0	0	0	0		0	0	0
	Function	PIP function 0:Disable 1:Enable	Segment data 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0(Note 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

The enable width of the LLOAD signal is specified using LCDLDW<LDW9:0>. It can be set in a range of 0 to 1024 pulses of the LCP0 clock.

The actual enable width is determined depending on the LCDLDDLY<PDT> setting, as shown below.

Enable width = <LDW9:0> + 1 (when <PDT> = "1", <LDW9:0>="0" is prohibited)

Enable width = <LDW9:0> (when <PDT> = "0")

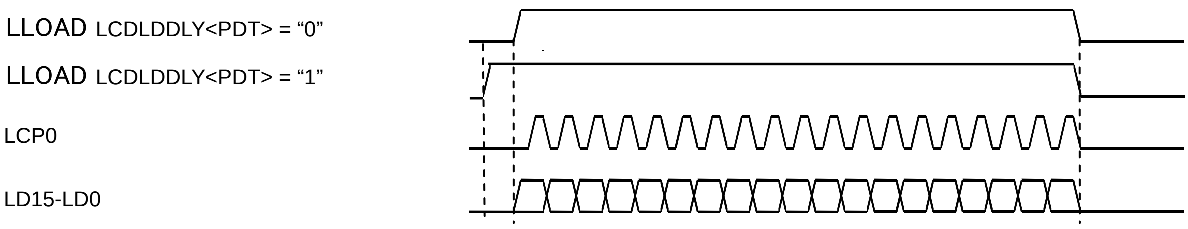
LLOAD width Register

LCDLDW (0295H)		7	6	5	4	3	2	1	0
	bit Symbol	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LLOAD width (bits 7-0)							

Signal width Bit8,9 Register

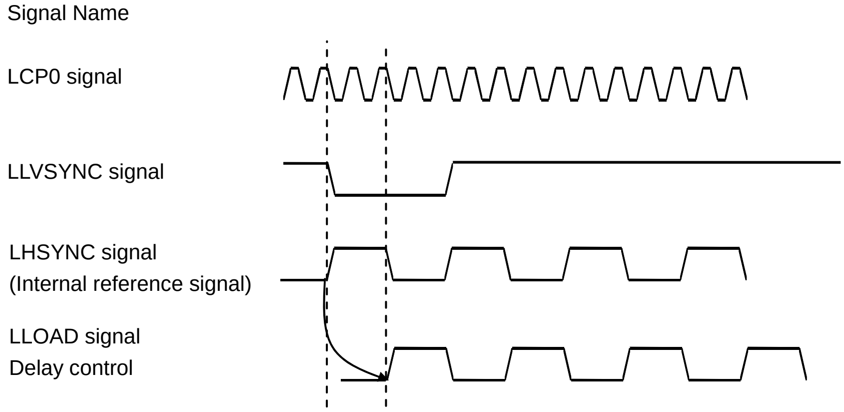
LCDHWB8 (0299H)		7	6	5	4	3	2	1	0
	bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

When LCDCTL0<LCP0OC>= "1", the enable width of the LLOAD signal is shown below.



As shown in the diagram below, delay time of 0 to 127 pulses of the LCP0 clock can be inserted in the LLOAD signal.

Delay time = <LDD6:0>



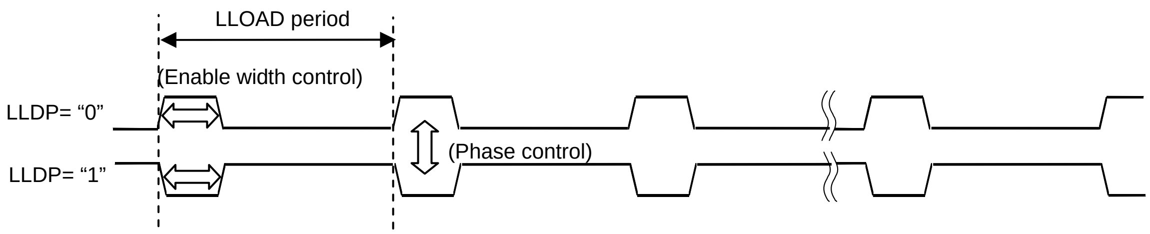
Note: The delay time for the LLOAD signal is controlled based on LCDLDDLY<PDT>= "1". Therefore, even if the delay time is set to "0" with LCDLDDLY<PDT>= "0", the LLOAD signal is output with a delay of one LCP0 clock. Be careful about this point.

LLOAD Delay Register

LCDLDDLY (0290H)

	7	6	5	4	3	2	1	0
bit Symbol	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
Read/Write	R/W	W						
Reset State	0	0	0	0	0	0	0	0
Function	Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

The phase of the LLOAD signal can be inverted by the setting of LCDCTL1 <LLDP>.



LLOAD signal

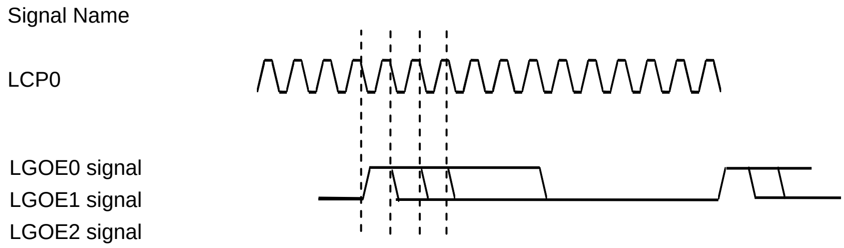
LCD Control 1 Register

LCDCTL1 (0286H)

	7	6	5	4	3	2	1	0
bit Symbol	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
Read/Write	R/W						R/W	
Reset State	1	0	1	0			0	0
Function	LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	

(4) LGOE0 to LGOE2 Signals

The LCDC has three signals (LGOE0 to LGOE2) that can be controlled like the LHSYNC signal. For these signals, the enable width, delay time, and phase timing can be adjusted as shown below.



High width setting
LGOE0: LCP0 clock = 1, 2, 3 … 512 pulses
LGOE1: LCP0 clock = 1, 2, 3 … 1024 pulses
LGOE2: LCP0 clock = 1, 2, 3 … 1024 pulses

LGOE0 width Register

		7	6	5	4	3	2	1	0
LCDHO0W (0296H)	bit Symbol	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE0 width (bits 7-0)							

LGOE1 width Register

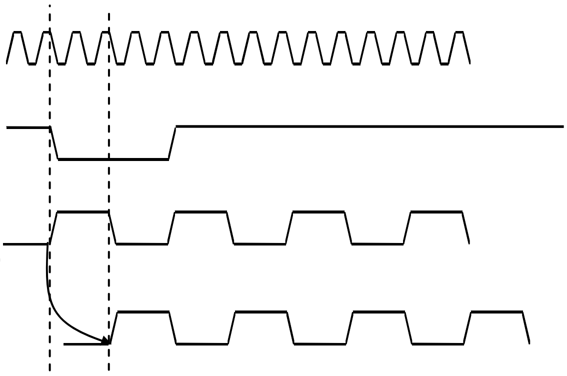
		7	6	5	4	3	2	1	0
LCDHO1W (0297H)	bit Symbol	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE1 width (bits 7-0)							

LGOE2 width Register

		7	6	5	4	3	2	1	0
LCDHO2W (0298H)	bit Symbol	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 7-0)							

Signal width Bit8,9 Register

		7	6	5	4	3	2	1	0
LCDHWB8 (0299H)	bit Symbol	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
	Read/Write	W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)



LGOE0 Delay Register

LCDO0DLY (0291H)

	7	6	5	4	3	2	1	0
bit Symbol		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE0 delay (bits 6-0)						

LGOE1 Delay Register

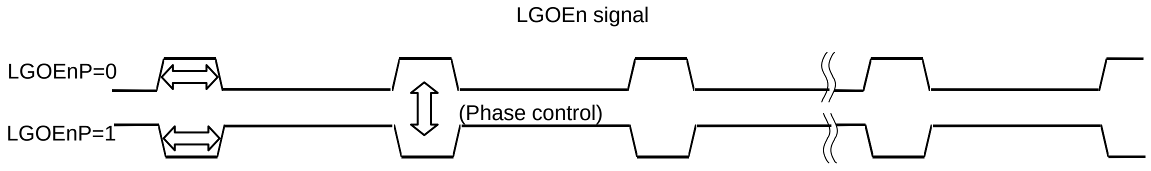
LCDO1DLY (0292H)

	7	6	5	4	3	2	1	0
bit Symbol		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE1 delay (bits 6-0)						

LGOE2 Delay Register

LCDO2DLY (0293H)

	7	6	5	4	3	2	1	0
bit Symbol		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
Read/Write		W						
Reset State		0	0	0	0	0	0	0
Function		OE2 delay (bits 6-0)						



LCD Control 2 Register

		7	6	5	4	3	2	1	0
LCDCTL2 (0287H)	bit Symbol	LGOE2P	LGOE1P	LGOE0P					
	Read/Write	R/W							
	Reset State	0	0	0					
	Function	LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					

(5) LFR Signal

The LFR (frame) signal is used to control the direction of bias the LCD driver applies on liquid crystal cells. With small screens in monochrome mode, the polarity of the LFR signal is normally inverted in synchronization with each screen display. With large screens or when grayscale or color mode is used, the polarity is inverted at shorter intervals to adjust the display quality.

When LCDCTL0<FRMON>="1" and LCDCTL0<DLS> = "0", the LFR signal is inverted at intervals of "LHSYNC × N" (LHSYNC: internal reference signal with 0 delays). The "N" value is specified in LCDDVM0<FML3:0> and LCDDVM1<FML7:4>.

When <DLS>= "0" and <FREDGE>= "0", LFR signal synchronous with front edge of LHSYNC signal, and when <DLS>="0" and <FREDGE>=1, LFR signal synchronous with rear edge of LHSYNC signal.

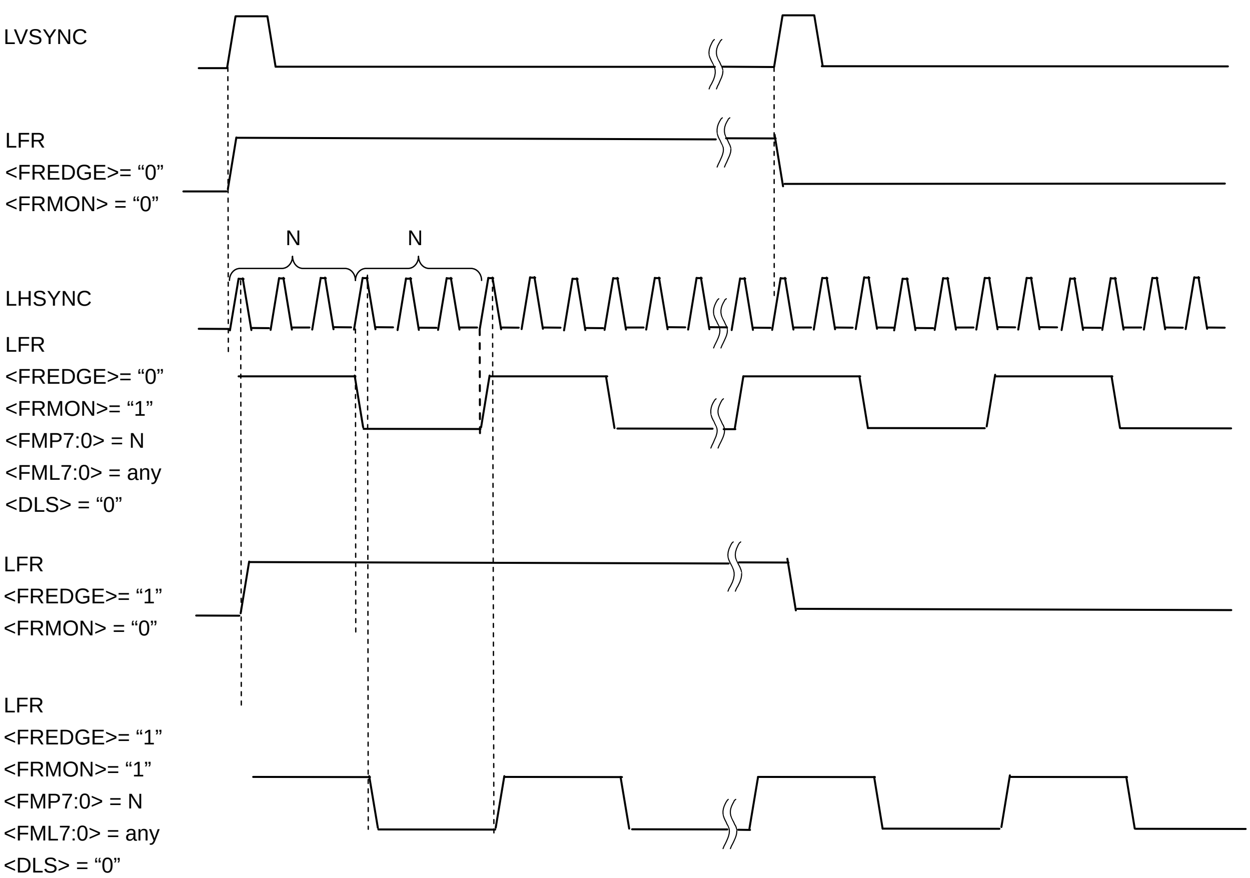
When LCDCTL0<FRMON> is set to "0" to disable the frame divide function, the LFR signal is inverted in synchronization with the LVSYNC period.

Enabling this function does not affect the waveform and timing of the LVSYNC signal. (The refresh rate is not changed.)

Note1: The effect of this function varies with the characteristics of the LCD driver and LCD panel to be used.

Note2: LFR signal delaies synchronous with LHSYNC signal.

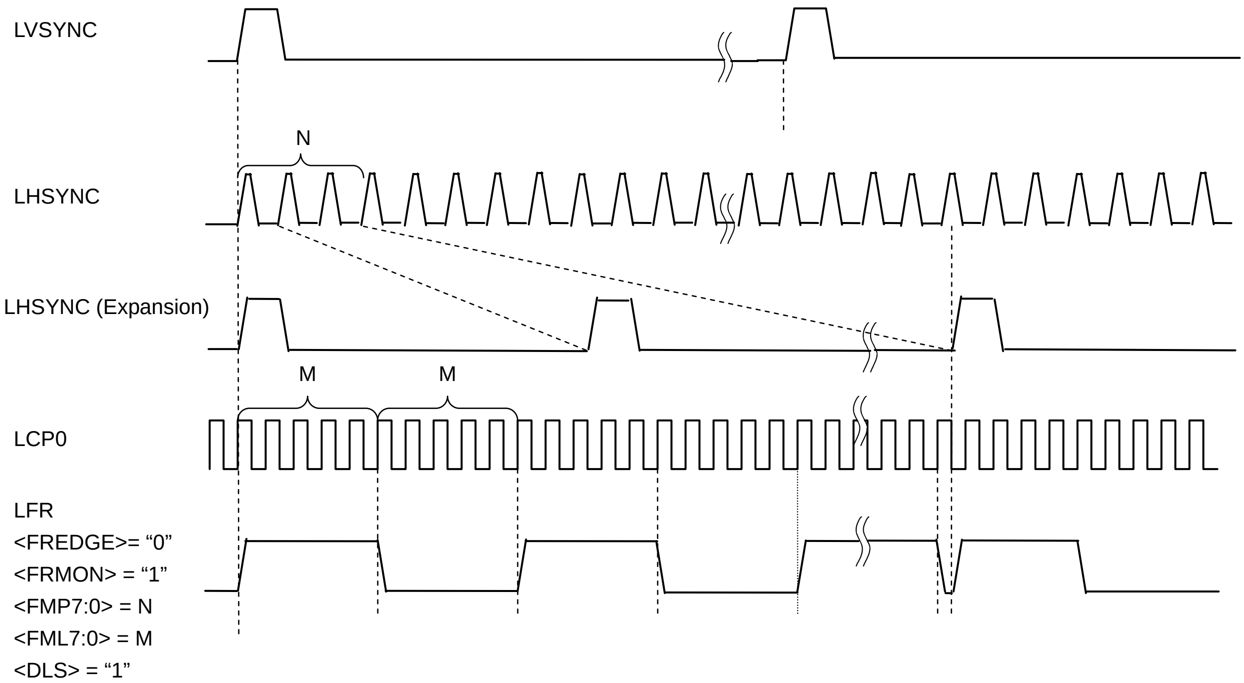
Generally, setting a prime number (3, 5, 7, 11, 13 and so on) as the "N" value produces better results.



When LCDCTL0<FRMON>= “1” and LCDCTL0<DLS>= “1”, frame output is inverted at intervals set in LCDDVM0<FML3:0> and the LFR signal is inverted at intervals of “LCP0 × M”. The “M” value is specified in LCDDVM0<FMP7:4>.

When <DLS>= “1” LFR signal synchronous with front edge of LHSYNC signal.

So, prohibit to set <FREDGE>= “1”, always need to set <FREDGE>= “0”.



Note: prohibit to set <FREDGE>= “1”, always need to set <FREDGE>= “0”.

LCD Control 0 Register

LCDCTL0 (0285H)

	7	6	5	4	3	2	1	0
bit Symbol	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
Read/Write	R/W			R/W		R/W		
Reset State	0	0	0	0		0	0	0
Function	PIP function 0:Disable 1:Enable	Segment data 0: Normal 1: Always output “0”	Frame divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0(Note 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

Note: When select STN mode, LCP0 is output at valid data only regardless of the setting of <LCP0OC> bit.

Divide FRM 0 Register

LCDDVM0 (0283H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 3-0)				LHSYNC DVM (bits 3-0)			

Divide FRM 1 Register

LCDDVM1 (0288H)

	7	6	5	4	3	2	1	0
bit Symbol	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
Read/Write	R/W							
Reset State	0	0	0	0	0	0	0	0
Function	LCP0 DVM (bits 7-4)				LHSYNC DVM (bit 7-4)			

(6) LD Bus

The data to be transferred to the LCD driver is output via a dedicated bus (LD15 to LD0). The output format can be selected according to the input method of the LCD driver to be used.

The LCDC reads data of the size corresponding to the specified LCD size from the display RAM and transfers it to the external LCD driver via the data bus pin dedicated to the LCD. Thus, the LCDC automatically issues a bus request to the CPU (to stop CPU operation) when it needs to read data from the display RAM. The bus occupancy rate of the LCDC varies depending on the display mode and the speed at which data is read from the display RAM.

Display RAM	Bus Width	Valid Data Read Time (f_{SYS} clocks/bytes)	Valid Data Read Time t_{LRD}(ns/bytes) at f_{SYS} = 60 MHz
External SRAM	16-bit	(2 + number of waits) / 2	16.6
Internal RAM	32-bit	**1/4	**4.16
External SDRAM	16-bit	*1/2	*8.33

Note: When SDRAM is used, additional 9 clocks are needed as overhead time for reading each common (line) data. When internal RAM is used, additional 1 clock is needed as overhead time for reading each common (line) data. Additional 1 clock of overhead time is also needed when a change of blocks occur in the internal RAM even if the common (line) remains the same.

The time the CPU stops operating while data for one common (line) is being transferred is defined as t_{STOP}, which is represented by the following equation:

$$t_{STOP} = (SegNum \times K / 8) \times t_{LRD}$$

SegNum : Number of display segments

K : Number of bits needed for displaying one pixel

Monochrome display	K=1
4-grayscale display	K=2
16-grayscale display	K=4
256-color display	K=8
4096-color display	K=12
65536-color display	K=16

Note: When SDRAM is used, overhead time is added as follows:

t_{STOP} [s] = (SegNum × K / 8) × t_{LRD} + ((1 / f_{SYS}) × 8)

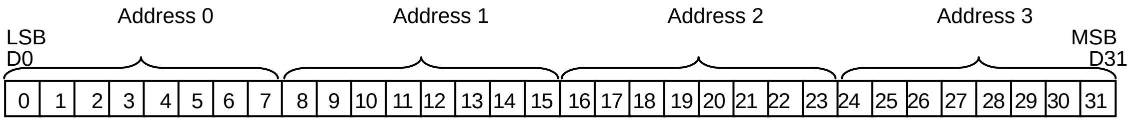
The bus occupancy rate indicates the proportion of the one common (line) update time t_{LP} occupied by t_{STOP} and is calculated by the following equation:

CPU bus occupancy rate = t_{STOP} [s] / LHSYNC [s: period]

- Memory Map Image and Data Output in Each Display Mode

<u>STN monochrome (1-pixel display data = 1-bit memory data)</u>

Display Memory



LD Bus Output

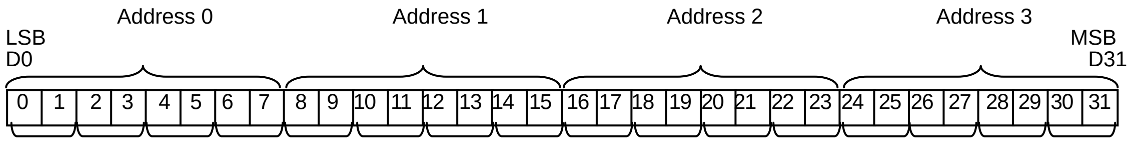
8-bit type

LD0 0 → 8 ⋯
LD1 1 → 9 ⋯
LD2 2 → 10 ⋯
LD3 3 → 11 ⋯
LD4 4 → 12 ⋯
LD5 5 → 13 ⋯
LD6 6 → 14 ⋯
LD7 7 → 15 ⋯

Note: When setting 240 segment, 256 segment size of data is required.

<u>STN 4-grayscale (1-pixel display data = 2-bit memory data)</u>

Display Memory



LD Bus Output

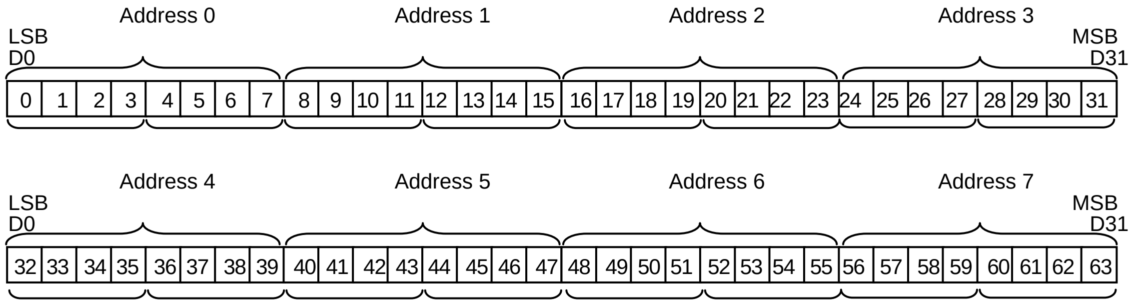
8-bit type

LD0 1 - 0 → 17-16 ⋯
LD1 3 - 2 → 19-18 ⋯
LD2 5 - 4 → 21-20 ⋯
LD3 7- 6 → 23-22 ⋯
LD4 9- 8 → 25-24 ⋯
LD5 11-10 → 27-26 ⋯
LD6 13-12 → 29-28 ⋯
LD7 15-14 → 31-30 ⋯

Figure 3.20.2 Memory Map Image and Data Output in STN Monochrome/4-Grayscale Mode

STN 16-grayscale (1-pixel display data = 4-bit memory data)

Display Memory



LD Bus Output

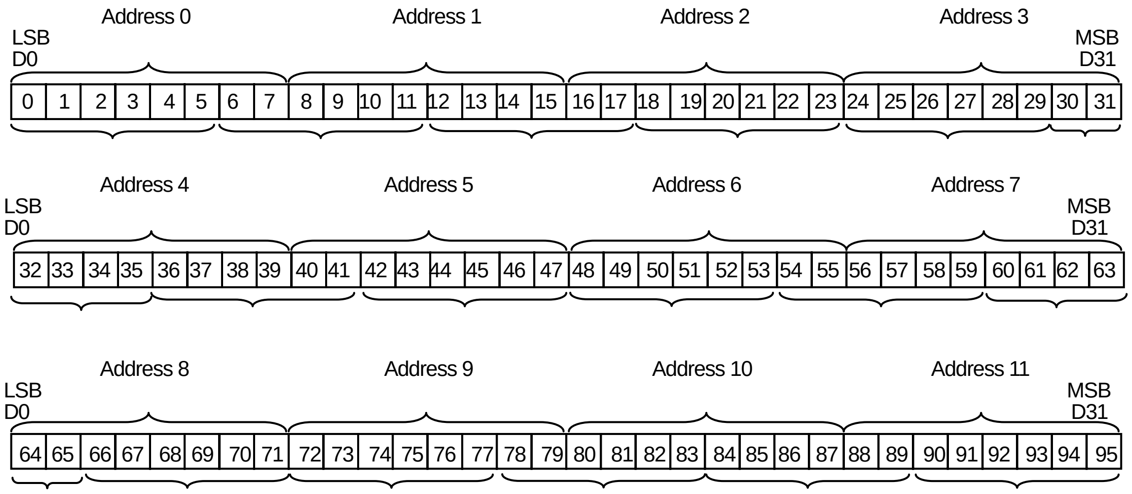
8-bit type

LD0	3-0	→ 35-32 ⋯
LD1	7-4	→ 39-36 ⋯
LD2	11-8	→ 43-40 ⋯
LD3	15-12	→ 47-44 ⋯
LD4	19-16	→ 51-48 ⋯
LD5	23-20	→ 55-52 ⋯
LD6	27-24	→ 59-56⋯
LD7	31-28	→ 63-60 ⋯

Figure 3.20.3 Memory Map Image and Data Output in STN 8-/16-Grayscale Mode

STN 64-grayscale (1-pixel display data = 6-bit memory data)

Display Memory



LD Bus Output

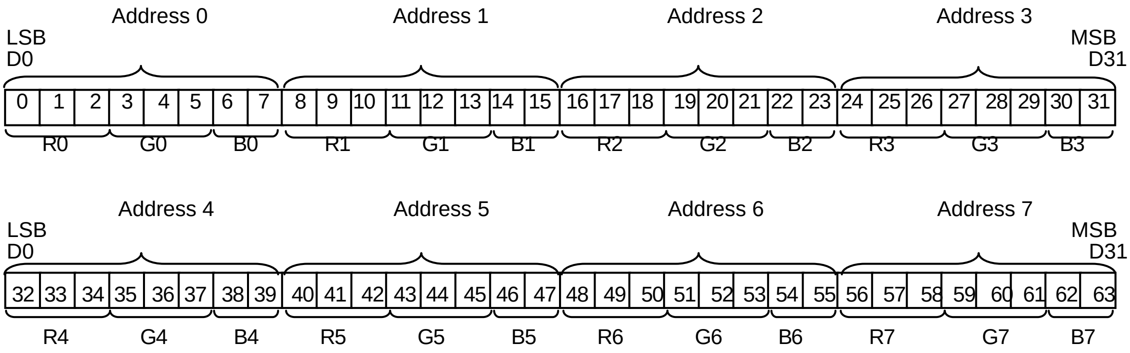
8-bit type

LD0	5-0	→	53-48
LD1	11-6	→	59-54
LD2	17-12	→	65-60
LD3	23-18	→	71-66
LD4	29-24	→	77-72
LD5	35-30	→	83-78
LD6	41-36	→	89-84
LD7	47-42	→	95-90

Figure 3.20.4 Memory Map Image and Data Output in STN 64-Grayscale Mode

STN 256-color (1-pixel display data = 8-bit memory data (R: 3 bits, G: 3 bits, B: 2 bits))

Display Memory



LD Bus Output

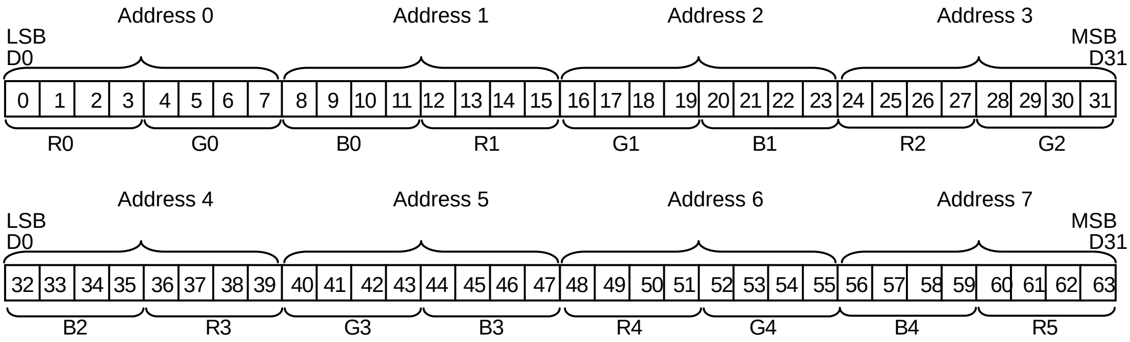
8-bit type

LD0	2-0(R0)	→ 23-22(B2) ⋯
LD1	5-3(G0)	→ 26-24(R3) ⋯
LD2	7-6(B0)	→ 29-27(G3) ⋯
LD3	10-8(R1)	→ 31-30(B3) ⋯
LD4	13-11(G1)	→ 34-32(R4) ⋯
LD5	15-14(B1)	→ 37-35(G4) ⋯
LD6	18-16(R2)	→ 39-38(B4) ⋯
LD7	21-19(G2)	→ 42-40(R5) ⋯

Figure 3.20.5 Memory Map Image and Data Output in STN 256-Color Mode

STN 4096-color (12 bpp: R: 4 bits, G: 4 bits, B: 4 bits)

Display Memory



LD Bus Output

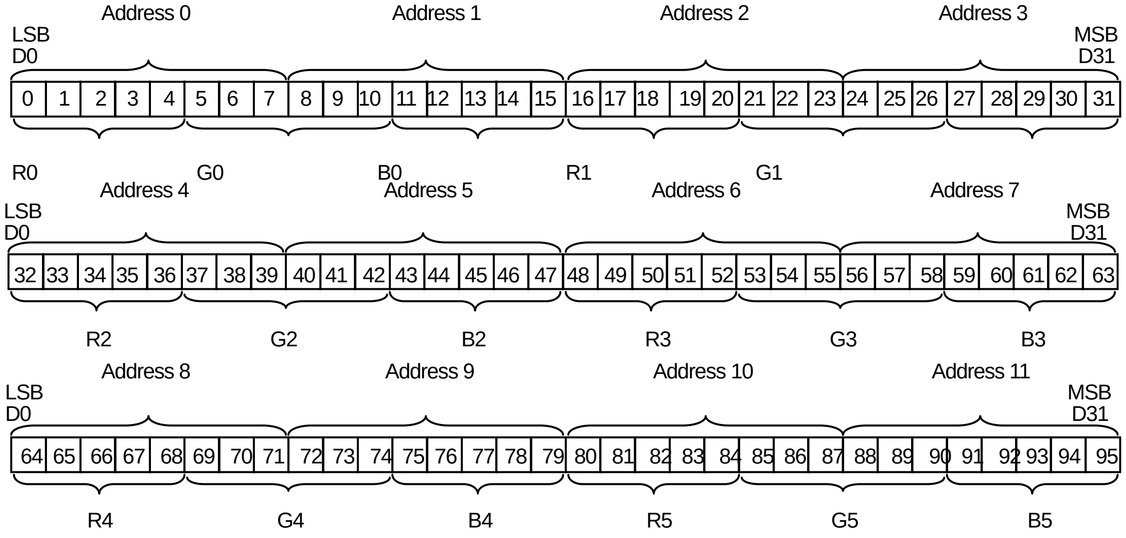
8-bit type

LD0	3-0(R0)	→ 35-32(B2)···
LD1	7-4(G0)	→ 39-36(R3)···
LD2	11-8(B0)	→ 43-40(G3)···
LD3	15-12(R1)	→ 47-44(B3)···
LD4	19-16(G1)	→ 51-48(R4)···
LD5	23-20(B1)	→ 55-52(G4)···
LD6	27-24(R2)	→ 59-56(B4)···
LD7	31-28(G2)	→ 63-60(R5)···

Figure 3.20.6 Memory Map Image and Data Output in STN 4096-Color Mode

STN 65536-color (16bpp: R: 5 bits, G: 6 bits, B: 5 bits)

Display Memory



LD Bus Output

8-bit type

LD0 4-0(R0) → 47-43(B2)···
LD1 10-5(G0) → 52-48(R3)···
LD2 15-11(B0) → 58-53(G3)···
LD3 20-16(R1) → 63-59(B3)···
LD4 26-21(G1) → 68-64(R4)···
LD5 31-27(B1) → 74-69(G4)···
LD6 36-32(R2) → 79-75(B4)···
LD7 42-37(G2) → 84-80(R5)···

Figure 3.20.7 Memory Map Image and Data Output in STN 65536-Color Mode

TFT Monochrome (1-pixel display data = 1-bit memory data)

Display Memory

LSB D0 … MSB D31

Address 0								Address 1								Address 2								Address 3							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

3-bit (TFT)

LD0	0	→	3	⋯
LD1	1	→	4	⋯
LD2	2	→	5	⋯

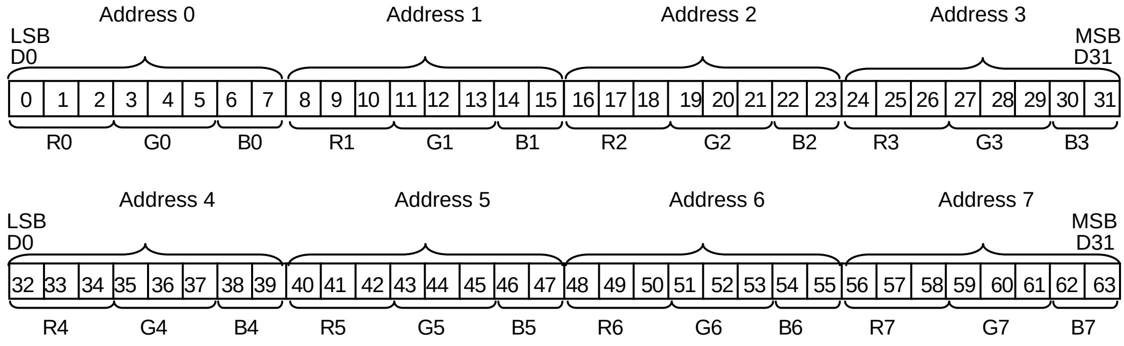
Note: When 240 segment is set, the data of 256 segments is necessary.

This mode is effective for Monochrome TFT display in TMD (Toshiba Matsushita Display Technology).

Figure 3.20.8 Memory Map Image and Data Output in TFT Monochrome Mode

TFT 256-color (1-pixel display data = 8-bit memory data (R: 3 bits, G: 3 bits, B: 2 bits)

Display Memory



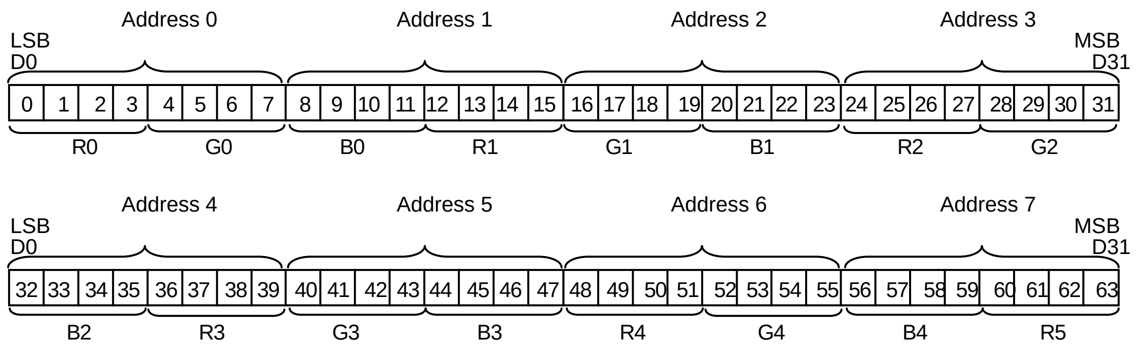
8-bit (TFT)

LD0	0(R0)	→	8(R1)	⋯
LD1	1(R0)	→	9(R1)	⋯
LD2	2(R0)	→	10(R1)	⋯
LD3	3(G0)	→	11(G1)	⋯
LD4	4(G0)	→	12(G1)	⋯
LD5	5(G0)	→	13(G1)	⋯
LD6	6(B0)	→	14(B1)	⋯
LD7	7(B0)	→	15(B1)	⋯

Figure 3.20.9 Memory Map Image and Data Output in TFT 256-Color Mode

TFT 4096-color (1-pixel display data = 12-bit memory data (R: 4 bits, G: 4 bits, B: 4 bits)

Display Memory



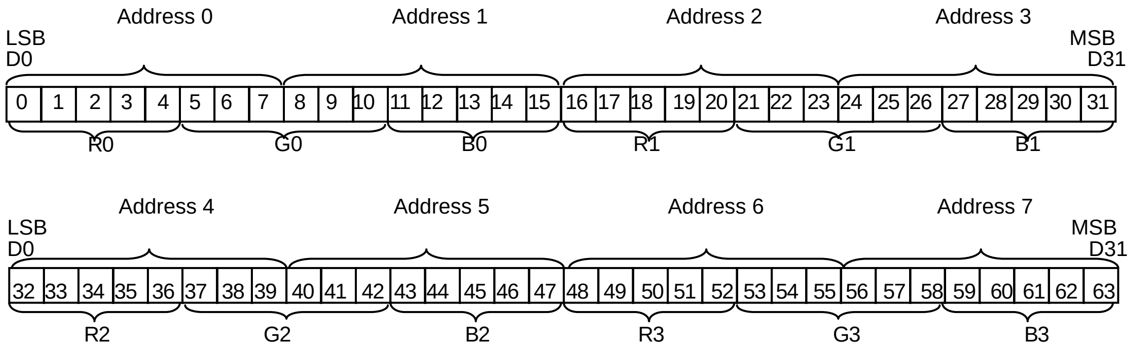
12-bit TFT

LD0	0(R0)	→ 12(R1) ⋯
LD1	1(R0)	→ 13(R1) ⋯
LD2	2(R0)	→ 14(R1) ⋯
LD3	3(R0)	→ 15(R1) ⋯
LD4	4(G0)	→ 16(G1) ⋯
LD5	5(G0)	→ 17(G1) ⋯
LD6	6(G0)	→ 18(G1) ⋯
LD7	7(G0)	→ 19(G1) ⋯
LD8	8(B0)	→ 20(B1) ⋯
LD9	9(B0)	→ 21(B1) ⋯
LD10	10(B0)	→ 22(B1) ⋯
LD11	11(B0)	→ 23(B1) ⋯

Figure 3.20.10 Memory Map Image and Data Output in TFT 4096-Color Mode

TFT 65536-color (16 bpp: R: 5 bits, G: 6 bits, B: 5 bits)

Display Memory



16-bit TFT

LD0	0(R0)	→ 16(R1) ⋯
LD1	1(R0)	→ 17(R1) ⋯
LD2	2(R0)	→ 18(R1) ⋯
LD3	3(R0)	→ 19(R1) ⋯
LD4	4(R0)	→ 20(R1) ⋯
LD5	5(G0)	→ 21(G1) ⋯
LD6	6(G0)	→ 22(G1) ⋯
LD7	7(G0)	→ 23(G1) ⋯
LD8	8(G0)	→ 24(G1) ⋯
LD9	9(G0)	→ 25(G1) ⋯
LD10	10(G0)	→ 26(G1) ⋯
LD11	11(B0)	→ 27(B1) ⋯
LD12	12(B0)	→ 28(B1) ⋯
LD13	13(B0)	→ 29(B1) ⋯
LD14	14(B0)	→ 31(B1) ⋯
LD15	15(B0)	→ 32(B1) ⋯

Figure 3.20.11 Memory Map Image and Data Output in TFT 65536-Color Mode

(7) LDIV Signal

The <LDINV> and <AUTOINV> bits of the LCDMODE1 register are used to control the LDIV signal as well as data output. The LDIV signal indicates the inversion of all the LD bus signals.

When LCDMODE1<LDINV>= “1”, all display data is forcefully inverted and the LDIV signal is also driven high. When LCDMODE1<AUTOINV>= “1”, the data that has just been transferred and the data to be transferred next are compared. If there are more changed bits than unchanged bits (for example, 7 or more bits are changed when using a 12-bit bus, and 5 or more bits are changed when using a 8-bit bus), the data is inverted and the LDIV signal is also driven high. This function can be used with TFT source drivers having the data inversion function to reduce radiated noise and power consumption due to high-speed data inversion.

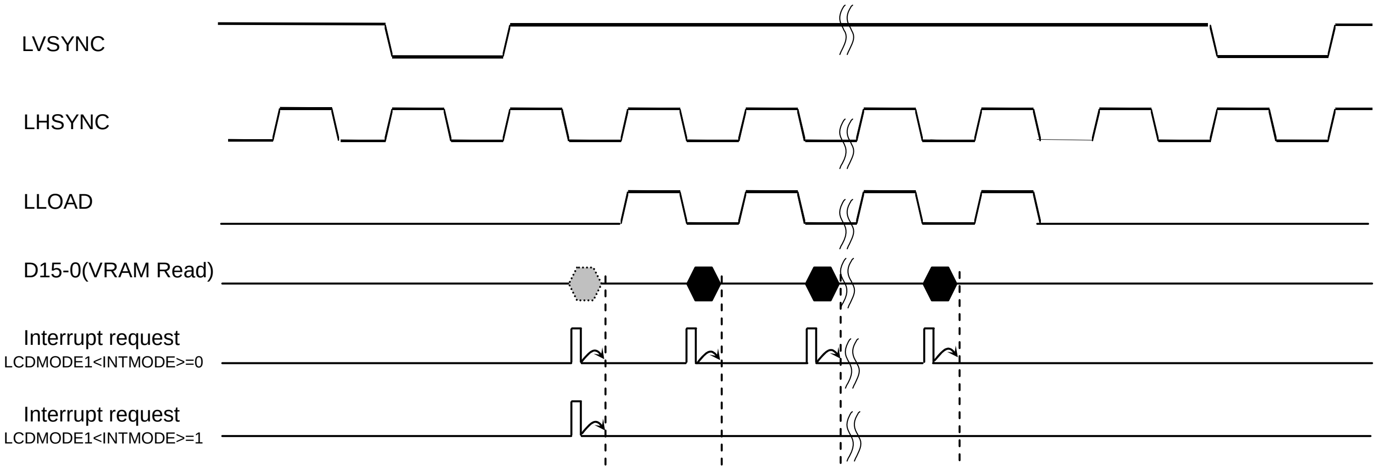
If <LDINV> and <AUTOINV> are both set to “1” at the same time, <LDINV> is given priority and <AUTOINV> is disabled.

3.20.4 Interrupt Function

The LCDC has two types of interrupts.

One is generated synchronous with the LLOAD signal and the other is generated synchronous with the LLOAD signal that is output immediately after the LVSYNC signal.

LCDMODE1<INTMODE> is used to switch between these two types of interrupts.



When LCDMODE1<INTMODE>= “0”, an interrupt request is generated at the start of each VRAM read before the LLOAD generates (once in each LLOAD period).

When LCDMODE1<INTMODE>= “1”, an interrupt request is generated at the start of VRAM read before the first LLOAD generates (once in each LVSYNC period).

Note: The interrupt request generates when reading the data from VRAM at once. Since reading from VRAM is executed by DMA with bus request to the CPU, DMA operation is given priority. Thus CPU accepts interrupt immediately after reading the data from VRAM.

LCDMODE1 Register

		7	6	5	4	3	2	1	0
LCDMODE1 (0281H)	bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
	Read/Write	R/W						W	
	Reset State	0	0	0	0	0	0	0	0
	Function	Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Horizontal & vertical flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNCREAR Edge	LD bus Trance Speed 0: normal 1: 1/3

Note: The LCDMODE1<INTMODE> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to “0” to stop the LCDC operation before changing the interrupt setting.

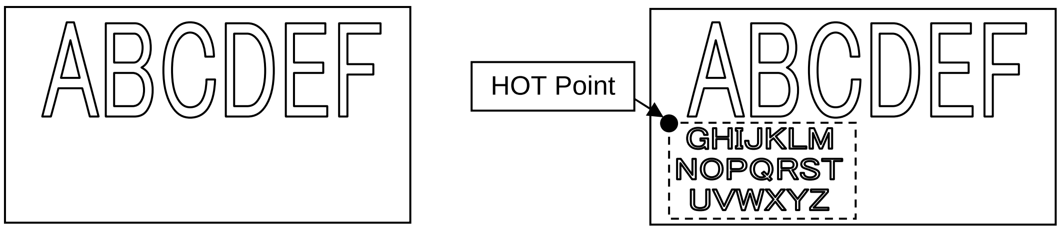
3.20.5 Special Functions

3.20.5.1 PIP (Picture in Picture) Function

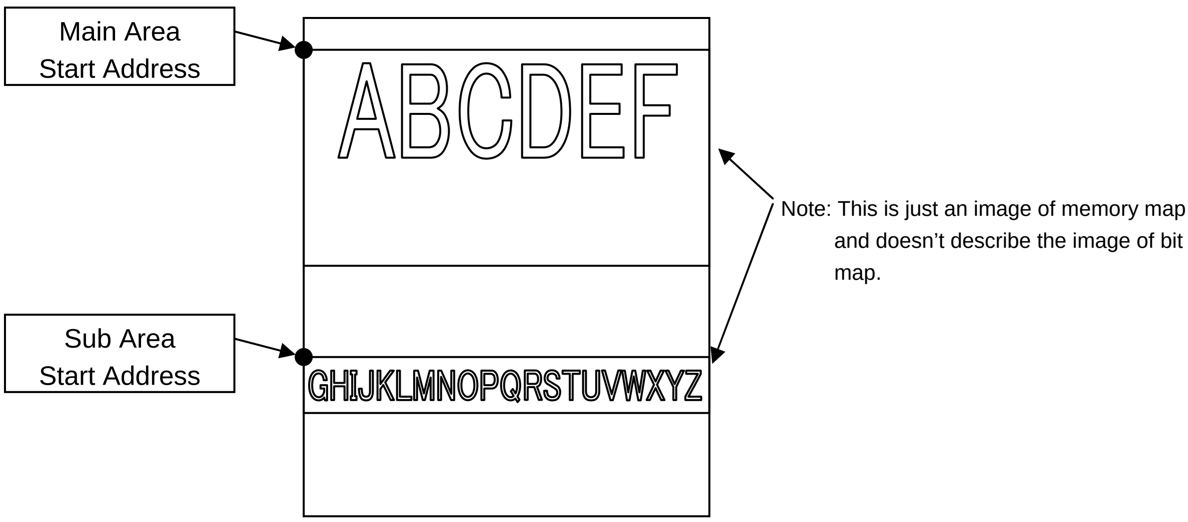
The TMP92CF29A includes a PIP (Picture in Picture) function that allows a different screen to be displayed over the screen currently being displayed on the LCD.

The PIP function manages the address space of display memory by dividing it into “main screen” and “sub screen”. For the main screen, the display size and start address are specified as in the case of the normal screen display. For the sub screen, the display size and start address are also specified for determining the position and size of the sub screen.

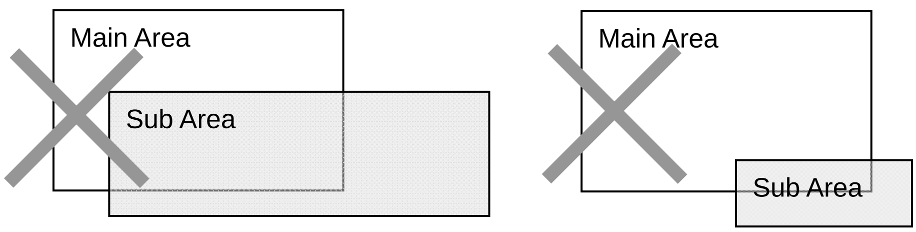
When the HOT point (upper-left corner) and segment/common size are set for the sub screen and the PIP function is enabled by setting LCDCTL0 <PIPE> to “1”, the sub screen is displayed over the main screen.



LCD Panel (PIP OFF)   LCD Panel (PIP ON)



VRAM Memory Map



Case1: Main Area size＜Sub Area size   Case2: Sub Area exceed Main Area

Note: Always set Sub Area within Main Area. The size that is bigger than the Main Area can not be set to the Sub Area, and the Sub area setting that lap Main Area.

The table below shows the HOT point locations that can be specified.

	*VRAM Access	HOT_Point(X_dir)	HOT_Point(Y_dir)
Monochrome display	16bit	In units of 16 dots	In units of 1 line
	32bit	In units of 32 dots	
4-grayscale display	16bit	In units of 8 dots	
	32bit	In units of 16 dots	
16-grayscale display	16bit	In units of 4 dots	
	32bit	In units of 8 dots	
64-grayscale display	16bit	In units of 8 dots	
	32bit	In units of 16 dots	
256-color display	16bit	In units of 2 dots	
	32bit	In units of 4 dots	
4K-color display	16bit	In units of 4 dots	
	32bit	In units of 8 dots	
64K-color display	16bit	In units of 1 dots	
	32bit	In units of 2 dots	

Note 1: The “VRAM Access” colomn shows the bus size for accessing the display RAM. When external RAM is used, the bus size depends on the bit width of the external RAM to be used. When the internal RAM is used VRAM is always accessed via a 32-bit bus.

Note 2: The same RAM must be used for both the main and sub areas.

The table below shows the HOT point segment and common sizes that can be specified.

	*VRAM Access	Segment size		Common size
		Minimum size	units	
Monochrome display	16bit	32 dots	In units of 16 dots	In units of 1 line
	32bit	64 dots	In units of 32 dots	
4-grayscale display	16bit	16 dots	In units of 8 dots	
	32bit	32 dots	In units of 16 dots	
16-grayscale display	16bit	8 dots	In units of 4 dots	
	32bit	16 dots	In units of 8 dots	
64-grayscale display	16bit	16 dots	In units of 8 dots	
	32bit	32 dots	In units of 16 dots	
256-color display	16bit	4 dots	In units of 2 dots	
	32bit	8 dots	In units of 4 dots	
4K-color display	16bit	8 dots	In units of 4 dots	
	32bit	16 dots	In units of 8 dots	
64K-color display	16bit	2 dots	In units of 1 dots	
	32bit	4 dots	In units of 2 dots	

LCD Main Area Start Address Register

		7	6	5	4	3	2	1	0
LSAML (02A0H)	bit Symbol	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	
	Function	LCD main area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSAMM (02A1H)	bit Symbol	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMSA9	LMSA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD main area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSAMH (02A2H)	bit Symbol	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
	Read/Write	R/W							
	Reset State	0	1	0	0	0	0	0	0
	Function	LCD main area start address (A23-A16)							

LCD Sub Area Start Address Register

		7	6	5	4	3	2	1	0
LSASL (02A4H)	bit Symbol	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	
	Function	LCD sub area start address (A7-A1)							
		7	6	5	4	3	2	1	0
LSASM (02A5H)	bit Symbol	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area start address (A15-A8)							
		7	6	5	4	3	2	1	0
LSASH (02A6H)	bit Symbol	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
	Read/Write	R/W							
	Reset State	0	1	0	0	0	0	0	0
	Function	LCD sub area start address (A23-A16)							

LCD Sub Area HOT Point Register (X-dir)

		7	6	5	4	3	2	1	0
LSAHX (02A8H)	bit Symbol	SAHX7	SAHX6	SAHX5	SAHX4	SAHX3	SAHX2	SAHX1	SAHX0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02A9H)	bit Symbol							SAHX9	SAHX8
	Read/Write							R/W	
	Reset State							0	0
	Function							LCD sub area HOT point (9-8)	

LCD Sub Area HOT Point Register (Y-dir)

		7	6	5	4	3	2	1	0
LSAHY (02AAH)	bit Symbol	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area HOT point (7-0)							
		7	6	5	4	3	2	1	0
(02ABH)	bit Symbol								SAHY8
	Read/Write								R/W
	Reset State								0
	Function								LCD sub area HOT point (8)

Note: The HOT point should be set in units of the specified number of dots, which is determined by the display color mode and display RAM access data bus width.

LCD Sub Area Display Segment Size Register

		7	6	5	4	3	2	1	0
LSASS (02ACH)	bit Symbol	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area segment size (7-0)							
		7	6	5	4	3	2	1	0
(02ADH)	bit Symbol							SAS9	SAS8
	Read/Write							R/W	
	Reset State							0	0
	Function							LCD sub area segment size (9-8)	

Note: The segment size should be set in units of the specified number of dots, which is determined by the display color mode and display RAM access data bus width.

LCD Sub Area Display Common Size Register

		7	6	5	4	3	2	1	0
LSACS	bit Symbol	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
(02AEH)	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	LCD sub area common size (7-0)							
		7	6	5	4	3	2	1	0
(02AFH)	bit Symbol								SAC8
	Read/Write								R/W
	Reset State								0
	Function	LCD sub area common size (8)							

Note: The common size should be set in units of 1 line.

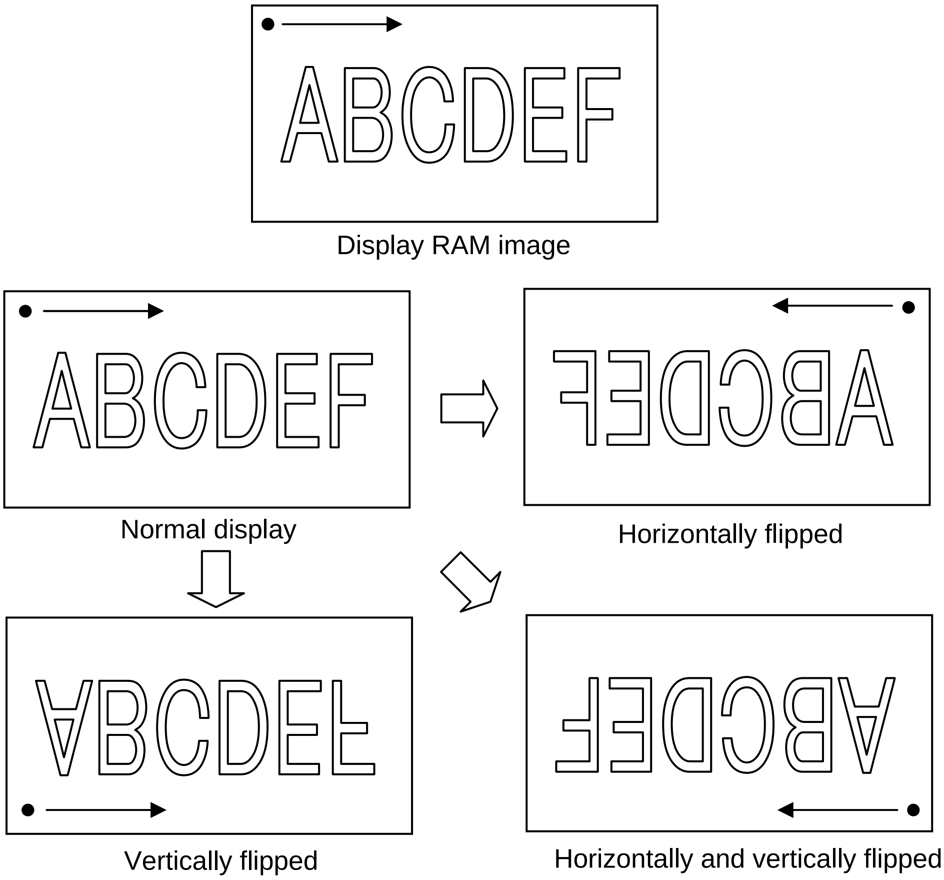
3.20.5.2 Display Data Rotation Function

When display RAM data is output to the LCD driver (LCDD), the data output direction can be automatically rotated by hardware to meet the specifications of the LCDD (or LCD module) to be used.

Table 3.20.2 Operation Conditions

Item	Vertical/Horizontal Flip Function	90-Degree Rotation Function
Display size	320 × 240	320×240 → 240 × 320
Color mode	64K colors (16 bpp)	64K colors (16 bpp)
Supported LCDD	TFT, STN	TFT, STN
Display RAM	Internal RAM, external SRAM	Internal RAM, external SRAM

1. Horizontal and Vertical Flip Function



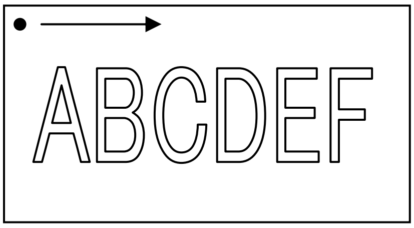
The display RAM image shown above uses the data scan method for the normal display screen so that data is read from the display RAM and written to the LCDD from left to right and top to bottom.

The data on the LCD screen appears as "horizontally flipped" if data is read from the display RAM from left to right and top to bottom and written to the LCDD from right to left and top to bottom.

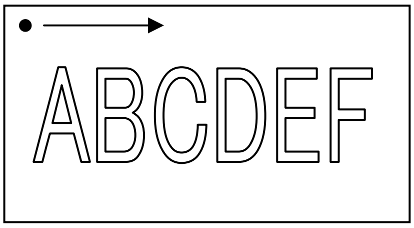
Likewise, the data on the LCD screen appears as "vertically flipped" if data is written to the LCDD from left to right and bottom to top, or as "horizontally and vertically flipped" if the data is written to the LCDD from right to left and bottom to top.

The horizontal and vertical flip function enables the output of display data to meet the specifications of each LCDD without the need to rearrange the display RAM data. In other words, the screen display can be flipped horizontally and vertically without the need to rewrite the display RAM data.

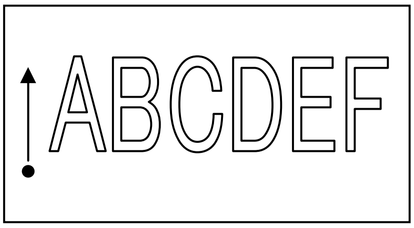
2. 90-Degree Rotation Function



Display RAM Image (QVGA 320×240)



QVGA (320×240)



Portrait-type QVGA (240×320)
(when this function is used)

The display RAM image above shows typical data of QVGA size (320 segments × 240 commons: landscape type). If the LCDD to be used is of landscape type, the data can be written to the LCDD without any problem.

If the LCDD to be used is of portrait type (240 segments × 320 commons), the data cannot be displayed properly.

This function enables the orientation of each display image to be rotated 90 degrees without the need to change the display RAM data.

3. Setting Method

The <LDC2:0> bits in the LCDMODE1 register are used to set the display data rotation function.

LCDMODE1 Register

		7	6	5	4	3	2	1	0
LCDMODE1 (0281H)	bit Symbol	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
	Read/Write	R/W						W	
	Reset State	0	0	0	0	0	0	0	0
	Function	Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 100: 90-degree 001: Horizontal flip 101: Reserved 010: Vertical flip 110: Reserved 011: Horizontal & vertical flip 111: Reserved			LD bus inversion 0: Normal 1: Invert	Auto bus inversion 0: Disable 1: Enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	LFR edge 0: LHSYNC Front Edge 1:LHSYNCREAR Edge	LD bus Trance Speed 0: normal 1: 1/3

Note: The <LDC2:0> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to “0” to stop the LCDC operation before changing <LDC2:0>.

When the horizontal and vertical flip function or 90-degree rotation function is used, the display RAM start address of main/sub area should be set differently from when in normal mode, as shown in the table below.

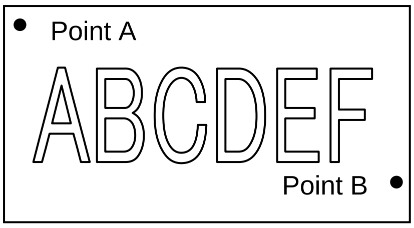
Mode	Setting Point	Display RAM Start Address Setting Example
Normal	Point A	00000h
90-degree rotation	Point B	257FEh
Horizontal flip	Point A	00000h
Vertical flip	Point B	257FEh
Horizontal and vertical flip	Point B	257FEh

How to calculate the point B address:

$(320\times240\times16/8) - 2 = 153600 - 2$

$= 153598$ [decimal]

$= 257FE$ [hex]



Display RAM Image (QVGA 320 × 240)

3.20.6 Considerations for Using the LCDC

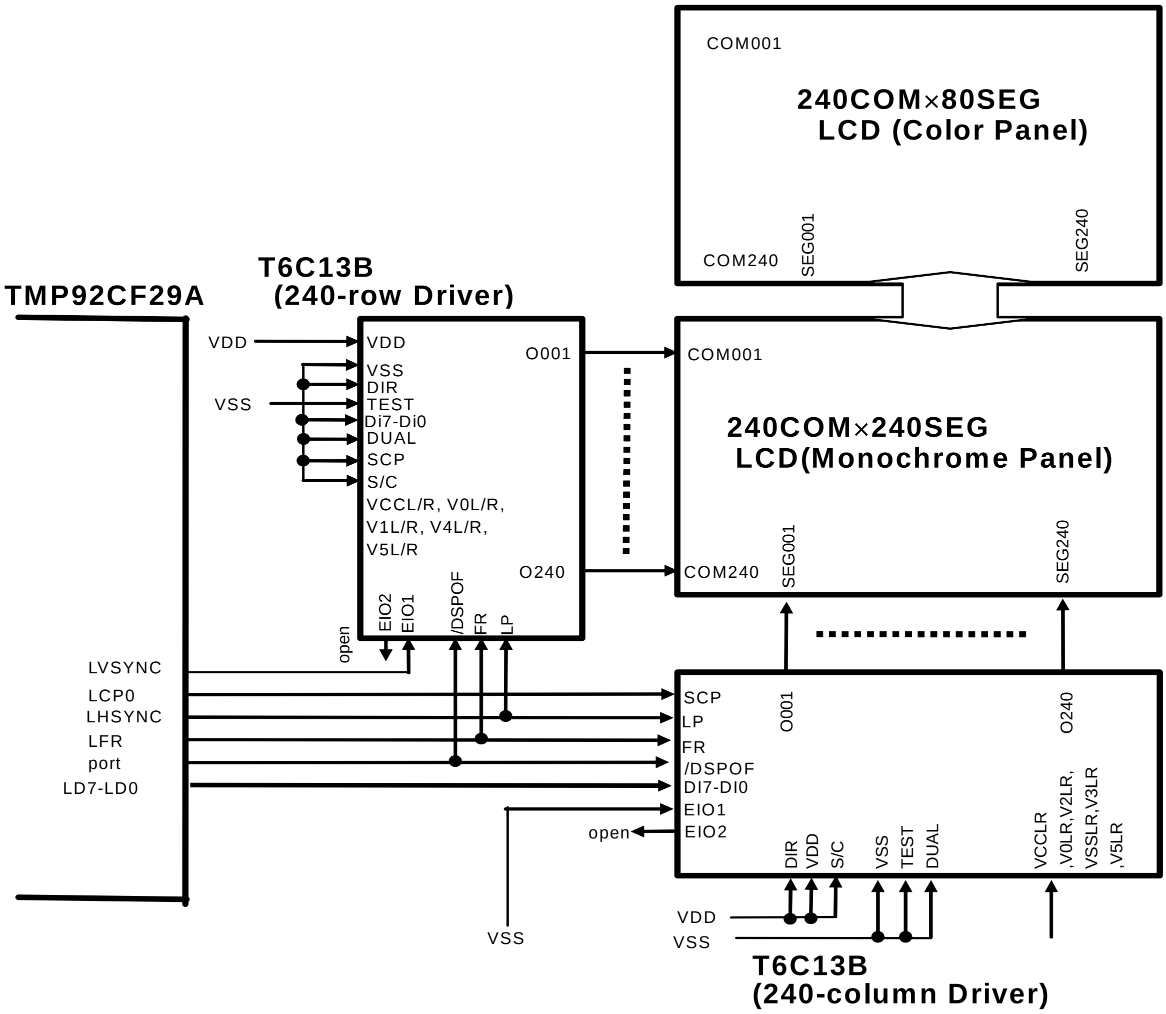
1. If the operation mode is changed while the LCDC is operating, a maximum of one frame may not be displayed properly. Although this degree of disturbance does not normally pose any problem (e.g. no response on LCD, display not visible to human eyes), the actual operation largely depends on the conditions such as the LCD driver, LCD panel, and frame frequency to be used. It is therefore recommended that operation checks be performed under the actual conditions.

2. The LCDMODE1<LDC2:0> setting must not be changed while the LCDC is operating. Be sure to set LCDCTL0<START> to "0" to stop the LCDC operation before changing <LDC2:0>.

3. The LCDC obtains the bus from the CPU when it has some operation to perform. Since the TMP92CF29A includes other units that act as bus masters such as HDMA and SDRAMC, it is necessary to estimate the bus occupancy rate of each bus master in advance. For details, see the chapter on HDMA.

3.20.7 Setting Example

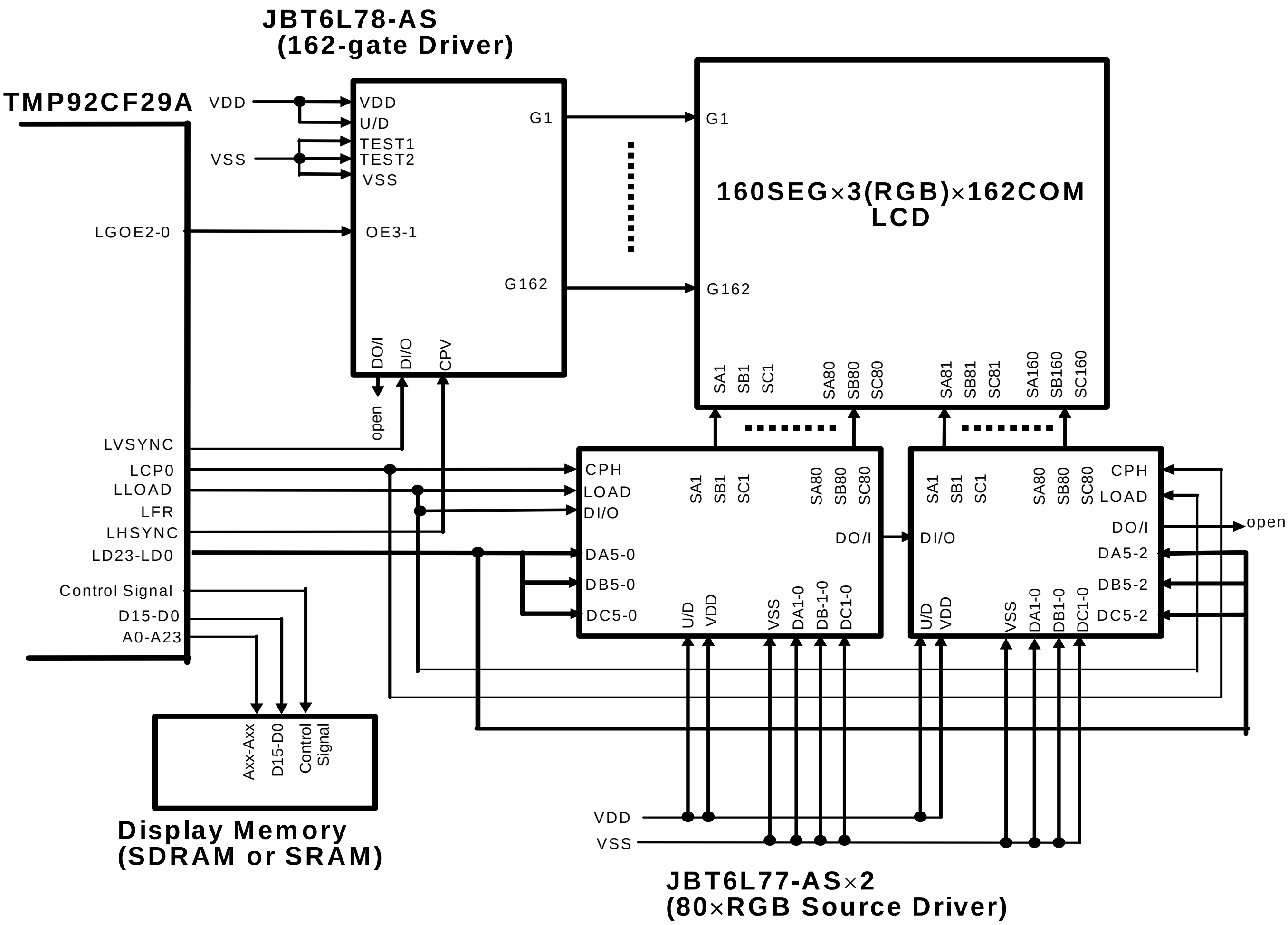
- STN



Note: The LCD drive power for LCD display mut be supplied from an external circuit.

Figure 3.20.12 STN-Type LCD Driver Connection Example

- TFT



Note: The LCD drive power for LCD display mut be supplied from an external circuit.

Figure 3.20.13 TFT-Type LCD Driver Connection Example

3.21 Touch Screen Interface (TSI)

An interface for 4-terminal resistor network touch-screen is built in.

The TSI easily supports two procedures: touch detection and X/Y position measurement.

Each procedure is performed by setting the TSI control register (TSICR0 and TSICR1) and using an internal AD converter.

3.21.1 Touch-Screen Interface Module Internal/External Connection

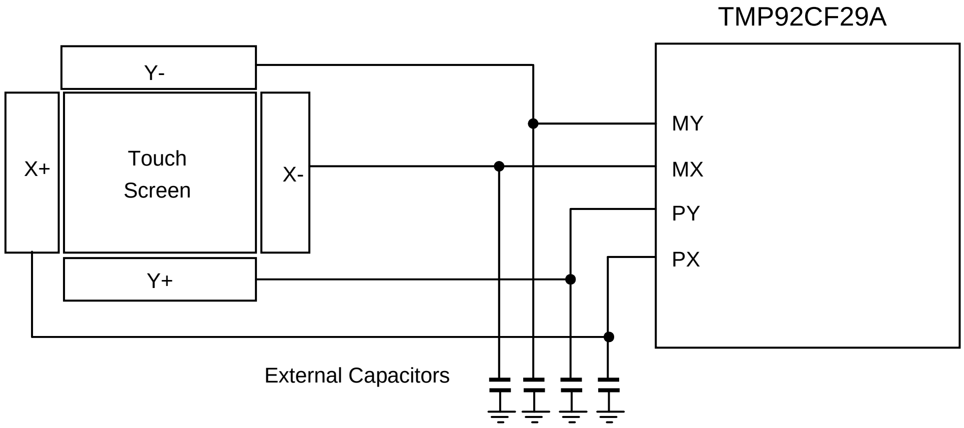


Figure 3.21.1External connection of TSI

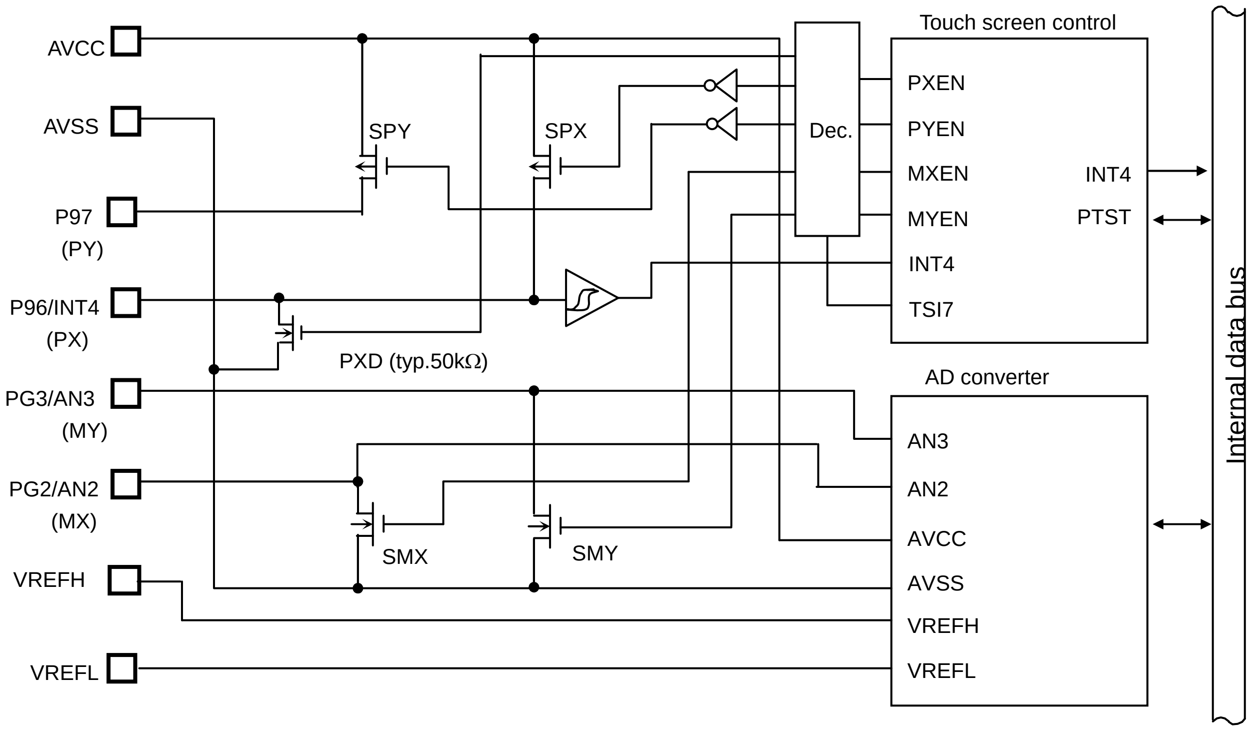


Figure 3.21.2 Internal block diagram of TSI

3.21.2 Touch Screen Interface (TSI) Control Register

TSI control register

		7	6	5	4	3	2	1	0
TSICR0	bit Symbol	TSI7	INGE	PTST	TWIEN	PYEN	PXEN	MYEN	MXEN
(01F0H)	Read/Write	R/W		R	R/W				
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Disable 1: Enable	Input gate control of Port 96,97 0: Enable 1: Disable	Detection condition 0: no touch 1: touch	INT4 interrupt control 0: Disable 1: Enable	SPY 0 : OFF 1 : ON	SPX 0 : OFF 1 : ON	SMY 0 : OFF 1 : ON	SMX 0 : OFF 1 : ON

PXD (internal pull-down resistor) ON/OFF setting

<TSI7> \ <PXEN>	0	1
0	OFF	OFF
1	ON	OFF

Debounce time setting register

		7	6	5	4	3	2	1	0
TSICR1	bit Symbol	DBC7	DB1024	DB256	DB64	DB8	DB4	DB2	DB1
(01F1H)	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	0: Disable 1: Enable	1024	256	64	8	4	2	1
			Debounce time is set by the formula “(N*64-16) / f_{SYS}”. “N” is the number of bits between bit6 and bit0 which are set to “1”. Note3:						

Note1: Since the CPU clock is used for the debounce circuit, the debounce circuit does not operate and also no interrupts that bypass the debounce circuit are generated during IDLE1and STOP mode, or the PCM state. During IDLE1 or STOP mode, set this circuit to disable (Write “0” in TSICR1<DBC7>) before entering the HALT stateIf debounce time is set to “0”, the signal is captured into the inside after a count of 6 system clocks (f_{SYS}) from the point when this circuit is set to disable.

Note2: To avoid a flow-through current to the normal C-MOS input gate when converting analog input data by using the AD converter, TSICR0<INGE> can be controlled. If the intermediate voltage is input, cut the input signal to the C-MOS logic (P96,P97) by setting this bit. TSICR0<PTST> is to confirm the initial pen-touch. Note that, when the input to the C-MOS logic is blocked by TSICR0<INGE>, this bit is always “1”.

Note3: For example:

TSICR1=95H →N = 64 + 4 + 1 = 69, if set to (TSICR1) = 95H

3.21.3 Touch detection procedure

The touch detection procedure includes the procedure starting from when the pen is touched onto the touch screen and until the pen-touch is detected.

Touching the screen generates the interrupt (INT4) and terminates this procedure. After an X/Y position measuring procedure is terminated, return to this procedure to wait for the next touch.

When waiting for a touch with no contact, set only the SPY switch to ON and set all other three switches (SMY, SPX, SMX) to OFF. At this time, the pull-down resistor built in the P96/INT4/PX pin is set ON.

In this state, because the internal X- and Y-direction resistors in the touch screen are not connected, the P96/INT4/PX pin is set to Low by the internal pull-down resistor (PXD), generating no INT4 interrupt.

When a next pen-touch is given, the X- and Y-direction internal resistors in the touch screen are connected, which sets the P96/INT4/PX pin to High and generates an INT4 interrupt.

To avoid generating more than one INT4 interrupt by one pen-touch, the debounce circuit as shown below is provided. Setting debounce time in the TSICR1 register ignores pulses whose time equals to or is below the set time.

The debounce circuit detects a rising of signal to count up a set debounce counter time and then captures the signal into the inside after counting. When the signal turns to “L” during counting, the counter is cleared, starting to wait for a rising edge again.

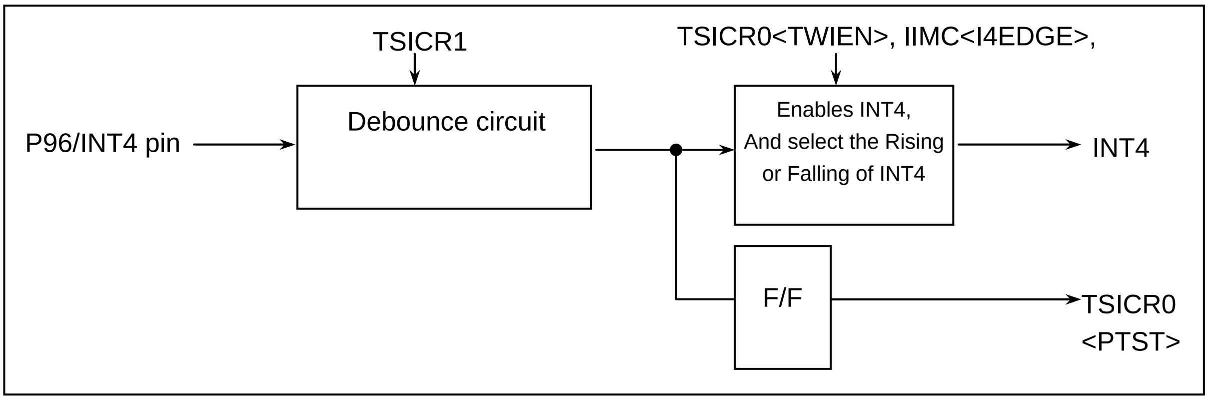


Figure 3.21.3 Block diagram of debounce circuit

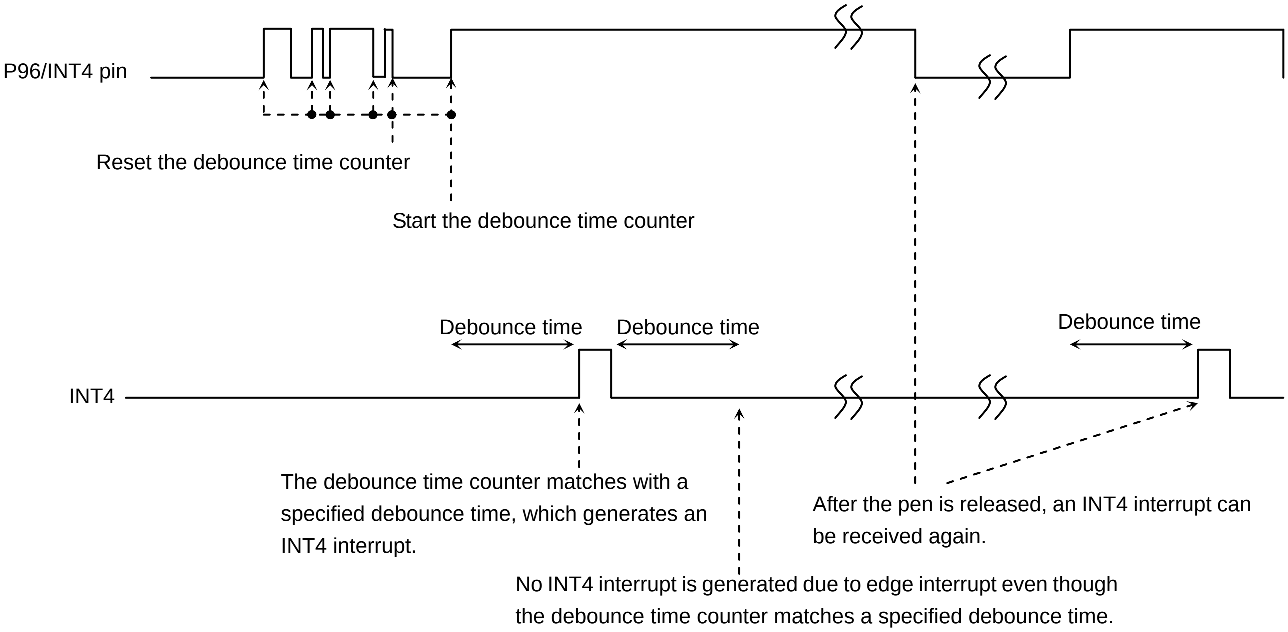


Figure 3.21.4 Timing diagram of debounce circuit

3.21.4 X/Y position measuring procedure

During the routine of pen-touch and INT4 interrupt generation, execute a pen position measuring following the procedure below:

<X position coordinate measurement>

Make the SPX and SMX switches ON, and the SPY and SMY switches OFF.

With this setting, an analog-voltage that shows the X position will be input to the PG3/MY/AN3 pin.

The X-position coordinate can be measured by converting this voltage to digital code using the AD converter.

<Y position coordinate measurement>

Make the SPY and SMY-switches ON, and the SPX and SMX switches OFF.

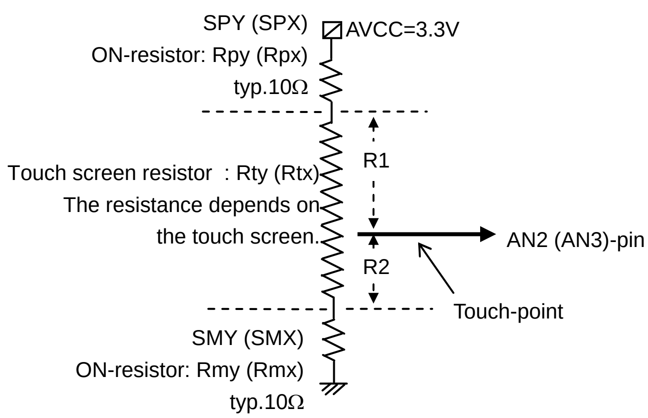
With this setting, an analog voltage that shows the Y position will be input to the PG2/MX/AN2 pin.

The Y position can be measured by converting this voltage to digital code using the AD converter.

The above analog voltage which is input to AN3 and AN2 pins during the X and Y position measurement above can be determined with the ratio between the ON resistance value of the switch in the TMP92CF29A and the resistance value in the touch screen as shown in Figure 3.21.5.

Therefore, even when touching an end area on the touch screen, the analog input voltage will be neither 3.3V nor 0.0V.

Note that the rate of each resistance varies. Remember to take this into consideration during designing. It is also recommended that an average taken from several AD conversions performed if required be adopted as the final correct value.



[Analog input voltage to theAN2 and AN3 pins: Formula to calculate E1]

E1 = ((R2+Rmy) / (Rpy+Rty+Rmy)) × AVCC [V]

Ex.) Where AVCC=3.3V, Rpy=Rmy=10Ω, R1=400Ω and R2=100Ω
E1 = ((100+10) / (10+400+100+10) × 3.3
= 0.698V

Note1: An X-coordinate position can be calculated in the same way though above formula is for Y-coordinate position.
Note2: Rty = R1+R2.

Figure 3.21.5 Calculation analog voltage

3.21.5 Flow chart for TSI

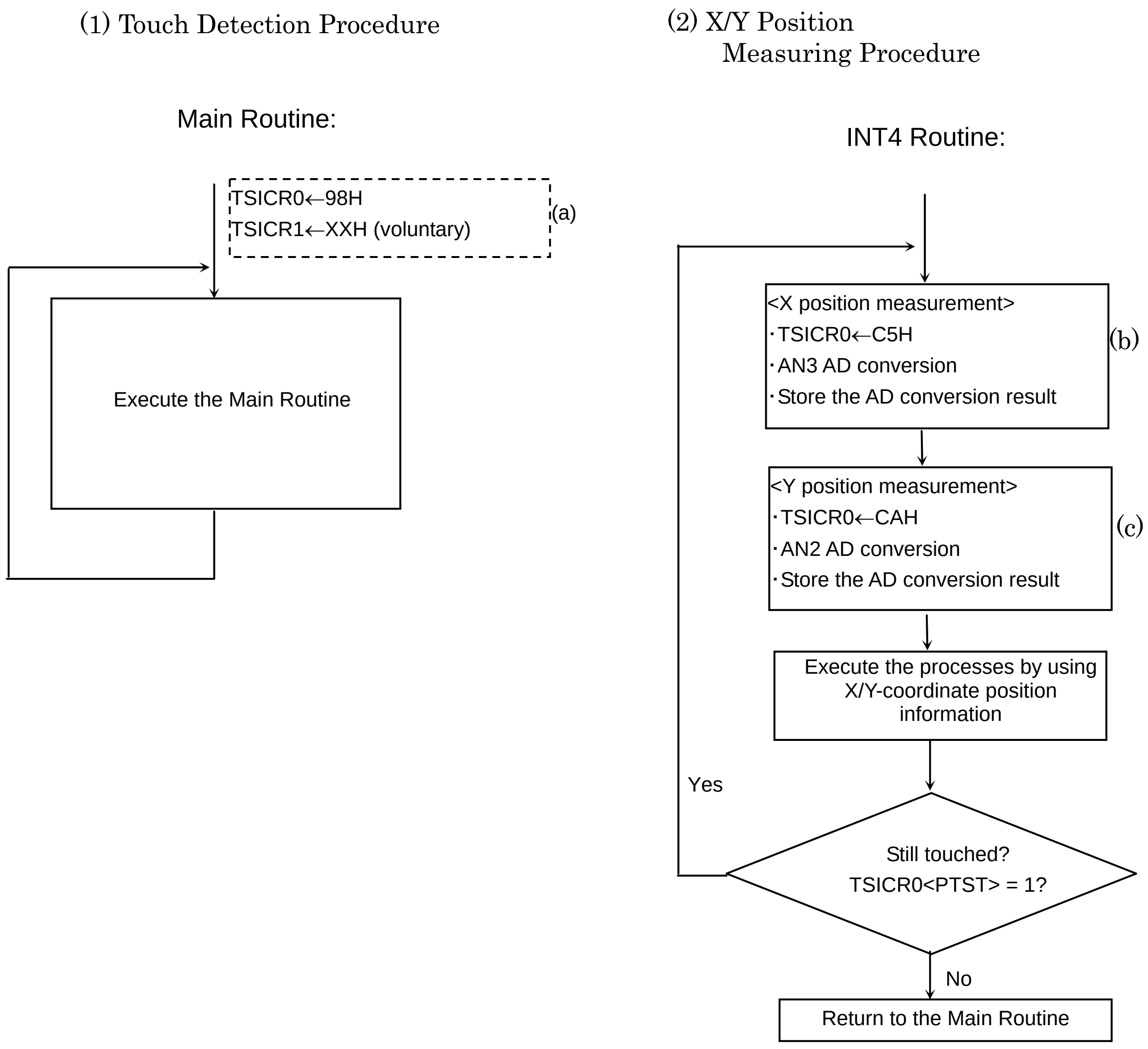
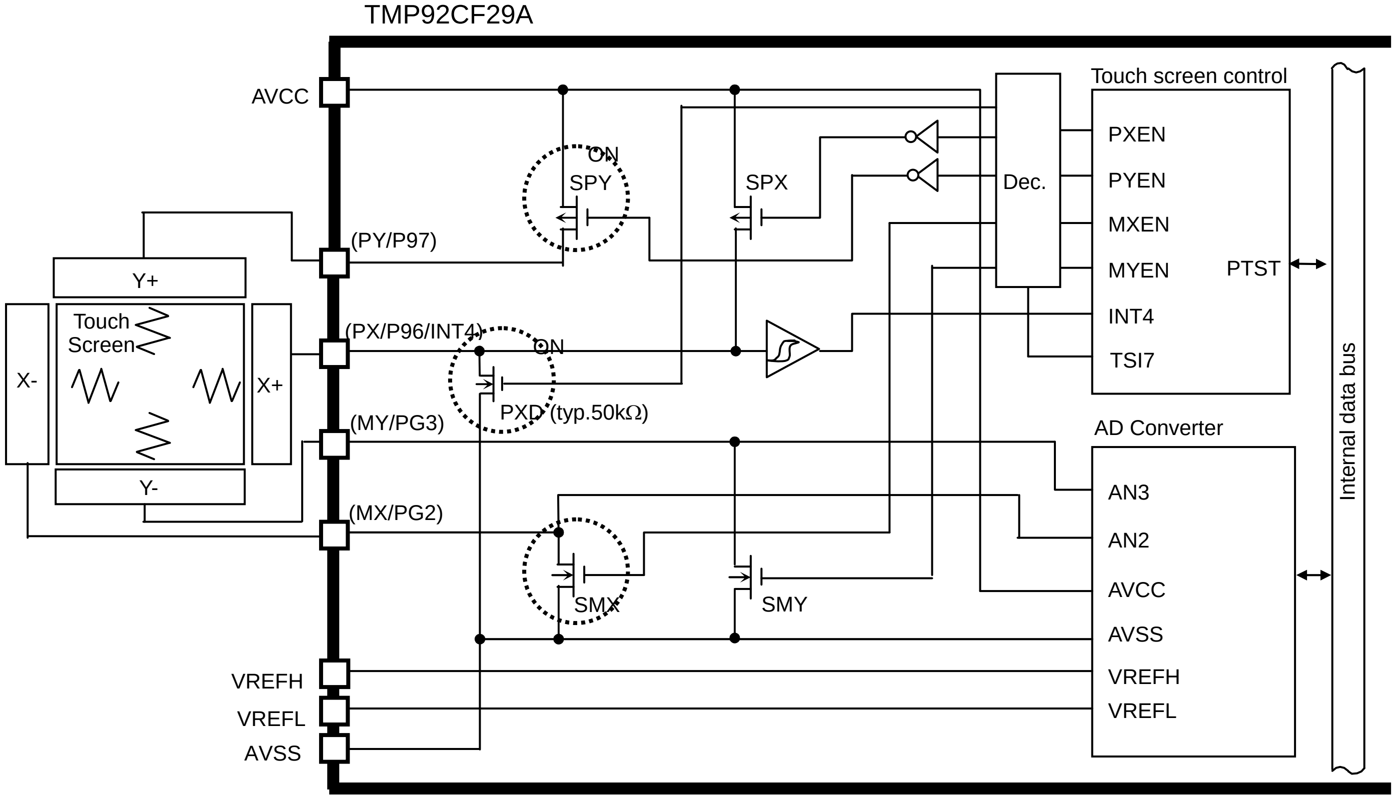


Figure 3.21.6 Flow chart for TSI

The following pages explain each circuit condition (a), (b) and (c) in the flow chart above:

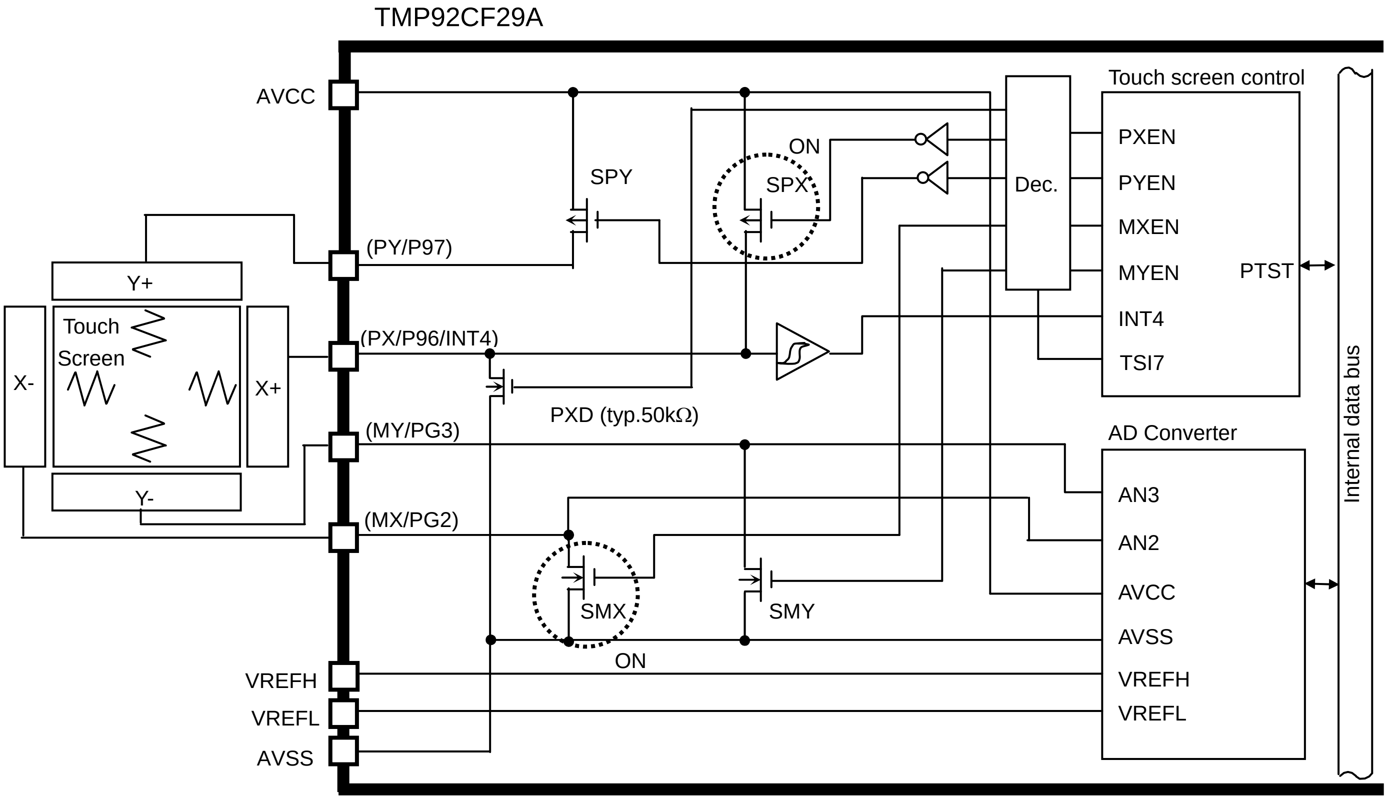
(a) Main routine (condition of waiting INT4 interrupt)

(p9fc)<P96F>, <P97F>= "1"	:	Set P96 to int4/PX, set P97 to PY
(inte34)	:	Set interrupt level of INT4
(tsicr0)=98h	:	Pull-down resistor on, SPY on, Interrupt-set<TWIEN>
ei	:	Enable interrupt



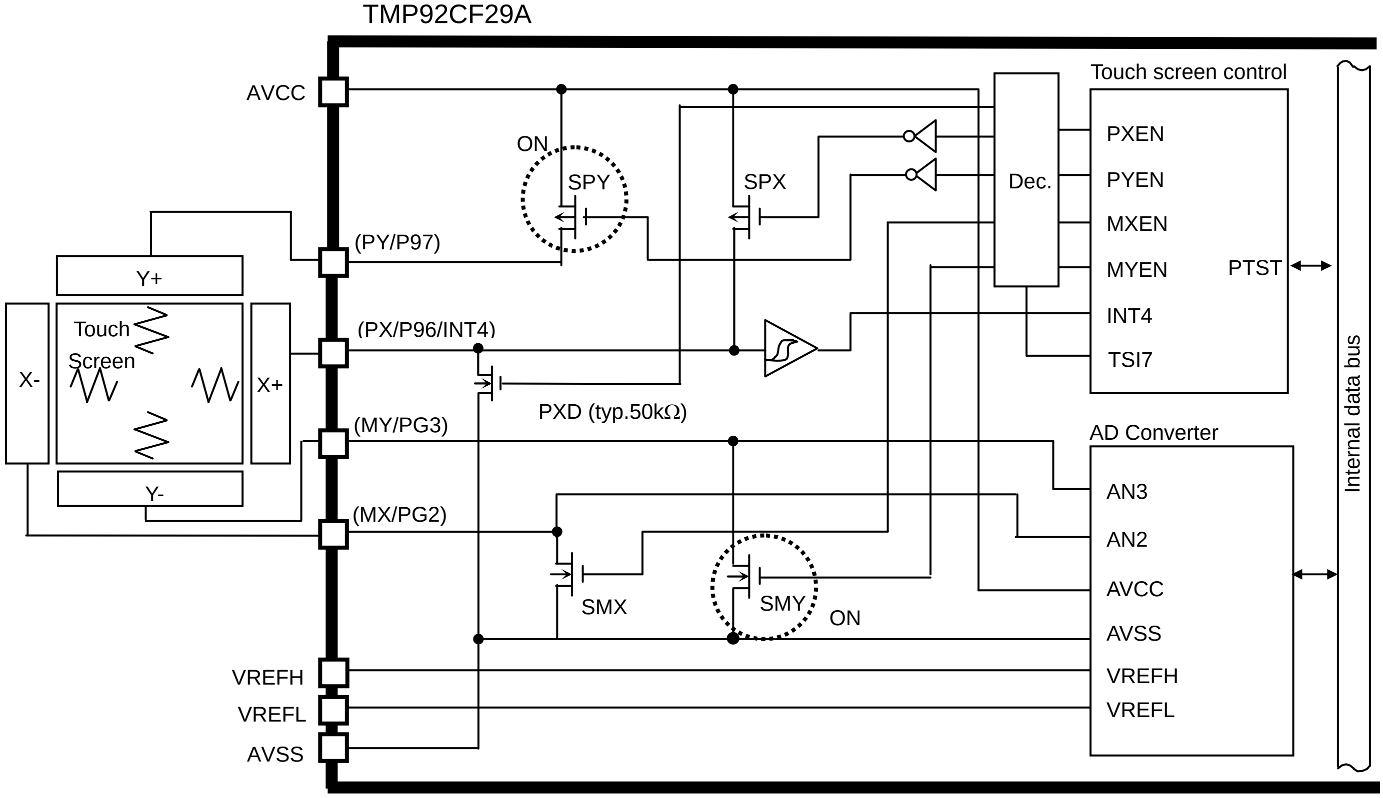
(b) INT4 routine: X-position coordinate measurement (AD conversion start)

(tsicr0)=c5h	:	Set SMX, SPX to ON. Set the input gate of P97, P96 to OFF.
(admod1)=b0h	:	Set to AN3.
(admod0)=08h	:	Start AD conversion.



(c) INT4 routine: Y-position coordinate measurement (AD conversion start)

(tsicr0)=cah	:	Set SMX, SPX to ON. Set the input gate of P97, P96 to OFF.
(admod1)=a0h	:	Set to AN2.
(admod0)=08h	:	Start AD conversion.



3.21.6 Use Cautions

1. Debounce circuit

The CPU system clock is used in debounce circuit. Therefore, when no clock is supplied to the CPU (during IDLE1 and STOP modes, or PCM state), the debounce circuit does not operate. Because of this, interrupts bypassing the debounce circuit are not generated either.

When using a startup that uses the TSI starting from the state during IDLE1 and STOP modes, or the PCM state, set the debounce circuit to disable before entering the HALT or PCM state. (TSICR1<DBC7>= “0”)

2. Port setting

When an intermediate voltage of 0 V to AVcc is converted using the AD converter, the intermediate voltage is also applied to the normal C-MOS input gates (P96 and P97) due to the circuit structure.

Take measures against the flow-through current to Port 96 and 97 by using TSICR0<INGE>. At this time (TSICR0<INGE>= “1”). Note that blocking the input to the C-MOS logics sets “1” at all times in TSICR0<PTST> that confirms a first pen-touch.

3.22 Real time clock (RTC)

3.22.1 Function description for RTC

1) Clock function (hour, minute, second)
2) Calendar function (month and day, day of the week, and leap year)
3) 24 or 12-hour (AM/PM) clock function
4) +/- 30 second adjustment function (by software)
5) Alarm function (Alarm output)
6) Alarm interrupt generate

3.22.2 Block diagram

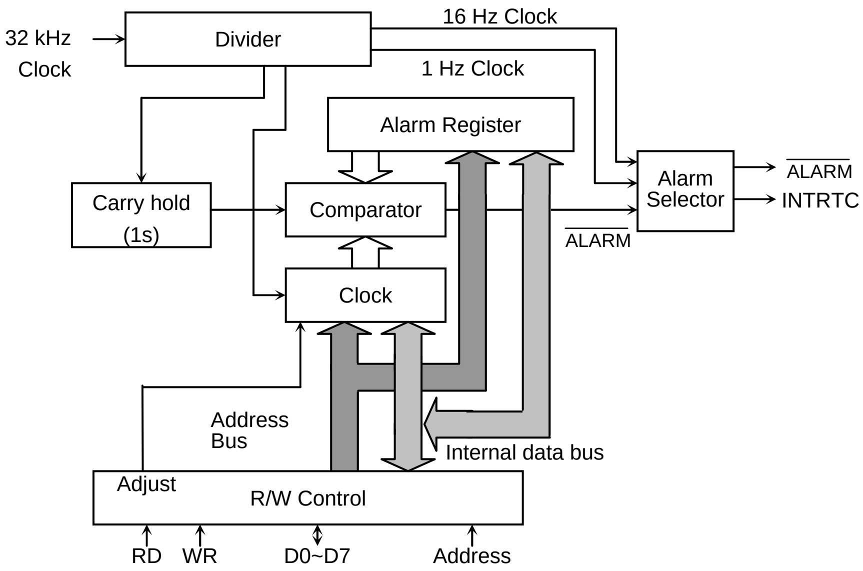


Figure 3.22.1 RTC block diagram

Note 1: Western calendar year column:

This product uses only the final two digits of the year. Therefore, the year following 99 is 00 years. In use, please take into account the first two digits when handling years in the western calendar.

Note 2: Leap year:

A leap year is divisible by 4, but the exception is any leap year which is divisible by 100; this is not considered a leap year. However, any year which is divisible by 400, is a leap year. This product does not take into account the above exceptions . Since this product accounts only for leap years divisible by 4, please adjust the system for any problems.

3.22.3 Control registers

Table 3.22.1 PAGE 0 (Clock function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	1320H		40 sec	20 sec	10 sec	8 sec	4 sec	2 sec	1 sec	Second column	R/W
MINR	1321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H				Oct.	Aug.	Apr.	Feb.	Jan.	Month column	R/W
YEARR	1326H	Year 80	Year 40	Year 20	Year 10	Year 8	Year 4	Year 2	Year 1	Year column (Lower two columns)	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write "0"				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, DATER, MONTHR, YEARR of PAGE0, the current state is read.

Table 3.22.2 PAGE1 (Alarm function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	1320H										R/W
MINR	1321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	1322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	1323H						W2	W1	W0	Day of the week column	R/W
DATER	1324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	1325H								24/12	24-hour clock mode	R/W
YEARR	1326H							LEAP1	LEAP0	Leap-year mode	R/W
PAGER	1327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	1328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write "0"				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, DATER, MONTHR, YEARR of PAGE1, the current state is read.

3.22.4 Detailed explanation of control register

RTC is not initialized by system reset. Therefore, all registers must be initialized at the beginning of the program.

(1) Second column register (for PAGE0 only)

SECR (1320H)

	7	6	5	4	3	2	1	0
Bit symbol		SE6	SE5	SE4	SE3	SE2	SE1	SE0
Read/Write		R/W						
Reset State		Undefined						
Function	"0" is read.	40 sec. column	20 sec. column	10 sec. column	8 sec. column	4 sec. column	2 sec. column	1 sec. column

0	0	0	0	0	0	0	0 sec
0	0	0	0	0	0	1	1 sec
0	0	0	0	0	1	0	2 sec
0	0	0	0	0	1	1	3 sec
0	0	0	0	1	0	0	4 sec
0	0	0	0	1	0	1	5 sec
0	0	0	0	1	1	0	6 sec
0	0	0	0	1	1	1	7 sec
0	0	0	1	0	0	0	8 sec
0	0	0	1	0	0	1	9 sec
0	0	1	0	0	0	0	10 sec
:							
0	0	1	1	0	0	1	19 sec
0	1	0	0	0	0	0	20 sec
:							
0	1	0	1	0	0	1	29 sec
0	1	1	0	0	0	0	30 sec
:							
0	1	1	1	0	0	1	39 sec
1	0	0	0	0	0	0	40 sec
:							
1	0	0	1	0	0	1	49 sec
1	0	1	0	0	0	0	50 sec
:							
1	0	1	1	0	0	1	59 sec

Note: Do not set data other than as shown above.

(2) Minute column register (for PAGE0/1)

MINR (1321H)

	7	6	5	4	3	2	1	0
Bit symbol		MI6	MI5	MI4	MI3	MI2	MI1	MI0
Read/Write		R/W						
Reset State		Undefined						
Function	"0" is read.	40 min, column	20 min, column	10 min, column	8 min, column	4 min, column	2 min, column	1 min, column

0	0	0	0	0	0	0	0 min
0	0	0	0	0	0	1	1 min
0	0	0	0	0	1	0	2 min
0	0	0	0	0	1	1	3 min
0	0	0	0	1	0	0	4 min
0	0	0	0	1	0	1	5 min
0	0	0	0	1	1	0	6 min
0	0	0	0	1	1	1	7 min
0	0	0	1	0	0	0	8 min
0	0	0	1	0	0	1	9 min
0	0	1	0	0	0	0	10 min
:							
0	0	1	1	0	0	1	19 min
0	1	0	0	0	0	0	20 min
:							
0	1	0	1	0	0	1	29 min
0	1	1	0	0	0	0	30 min
:							
0	1	1	1	0	0	1	39 min
1	0	0	0	0	0	0	40 min
:							
1	0	0	1	0	0	1	49 min
1	0	1	0	0	0	0	50 min
:							
1	0	1	1	0	0	1	59 min

Note: Do not set data other than as shown above.

(3) Hour column register (for PAGE0/1)

1. In case of 24-hour clock mode (MONTHR<MO0>= "1")

HOURR (1322H)		7	6	5	4	3	2	1	0
	Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
	Read/Write			R/W					
	Reset State			Undefined					
	Function	"0" is read.		20 hour column	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o'clock
0	0	0	0	0	1	1 o'clock
0	0	0	0	1	0	2 o'clock
:						
0	0	1	0	0	0	8 o'clock
0	0	1	0	0	1	9 o'clock
0	1	0	0	0	0	10 o'clock
:						
0	1	1	0	0	1	19 o'clock
1	0	0	0	0	0	20 o'clock
:						
1	0	0	0	1	1	23 o'clock

Note: Do not set data other than as shown above.

2. In case of 12-hour clock mode (MONTHR<MO0>= "0")

HOURR (1322H)		7	6	5	4	3	2	1	0
	Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
	Read/Write			R/W					
	Reset State			Undefined					
	Function	"0" is read.		PM/AM	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o'clock (AM)
0	0	0	0	0	1	1 o'clock
0	0	0	0	1	0	2 o'clock
:						
0	0	1	0	0	1	9 o'clock
0	1	0	0	0	0	10 o'clock
0	1	0	0	0	1	11 o'clock
1	0	0	0	0	0	0 o'clock (PM)
1	0	0	0	0	1	1 o'clock

Note: Do not set data other than as shown above.

(4) Day of the week column register (for PAGE0/1)

DAYR (1323H)		7	6	5	4	3	2	1	0
	Bit symbol						WE2	WE1	WE0
	Read/Write						R/W		
	Reset State						Undefined		
	Function	"0" is read.					W2	W1	W0

0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday

Note: Do not set data other than as shown above.

(5) Day column register (PAGE0/1)

DATER (1324H)		7	6	5	4	3	2	1	0
	Bit symbol			DA5	DA4	DA3	DA2	DA1	DA0
	Read/Write			R/W					
	Reset State			Undefined					
	Function	"0" is read.		Day 20	Day 10	Day 8	Day 4	Day 2	Day 1

0	0	0	0	0	1	1st day
0	0	0	0	1	0	2nd day
0	0	0	0	1	1	3rd day
0	0	0	1	0	0	4th day
			:			
0	0	1	0	0	1	9th day
0	1	0	0	0	0	10th day
0	1	0	0	0	1	11th day
			:			
0	1	1	0	0	1	19th day
1	0	0	0	0	0	20th day
			:			
1	0	1	0	0	1	29th day
1	1	0	0	0	0	30th day
1	1	0	0	0	1	31st day

Note1: Do not set data other than as shown above.

Note2: Do not set for non-existent days (e.g.: 30^{th} Feb)

(6) Month column register (for PAGE0 only)

MONTHR (1325H)		7	6	5	4	3	2	1	0
	Bit symbol				MO4	MO4	MO2	MO1	MO0
	Read/Write				R/W				
	Reset State				Undefined				
	Function	"0" is read.			10 months	8 months	4 months	2 months	1 month

0	0	0	0	1	January
0	0	0	1	0	February
0	0	0	1	1	March
0	0	1	0	0	April
0	0	1	0	1	May
0	0	1	1	0	June
0	0	1	1	1	July
0	1	0	0	0	August
0	1	0	0	1	September
1	0	0	0	0	October
1	0	0	0	1	November
1	0	0	1	0	December

Note: Do not set data other than as shown above.

(7) Select 24-hour clock or 12-hour clock (for PAGE1 only)

MONTHR (1325H)		7	6	5	4	3	2	1	0
	Bit symbol								MO0
	Read/Write								R/W
	Reset State								Undefined
	Function	"0" is read.							1: 24-hour 0: 12-hour

(8) Year column register (for PAGE0 only)

YEARR (1326H)

	7	6	5	4	3	2	1	0
Bit symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
Read/Write	R/W							
Reset State	Undefined							
Function	80 Years	40 Years	20 Years	10 Years	8 Years	4 Years	2 Years	1 Year

0	0	0	0	0	0	0	0	00 years
0	0	0	0	0	0	0	1	01 years
0	0	0	0	0	0	1	0	02 years
0	0	0	0	0	0	1	1	03 years
0	0	0	0	0	1	0	0	04 years
0	0	0	0	0	1	0	1	05 years
				:				
1	0	0	1	1	0	0	1	99 years

Note: Do not set data other than as shown above.

(9) Leap-year register (for PAGE1 only)

YEARR (1326H)

	7	6	5	4	3	2	1	0
Bit symbol							LEAP1	LEAP0
Read/Write							R/W	
Reset State							Undefined	
Function	"0" is read.						00: leap-year 01: one year after leap-year 10: two years after leap-year 11: three years after leap-year	

0	0	Current year is a leap-year
0	1	Current year is the year following a leap year
1	0	Current year is two years after a leap year
1	1	Current year is three years after a leap year

(10)PAGE register (for PAGE0/1)

		7	6	5	4	3	2	1	0
PAGER (1327H)	Bit symbol	INTENA			ADJUST	ENATMR	ENAALM		PAGE
	Read/Write	R/W			W	R/W			R/W
	Reset State	0			Undefined	Undefined			Undefined
A Read-modify- write operation cannot be performed	Function	Interrupt 0: Disable 1: Enable	"0" is read.		0: Don't care 1: Adjust	Clock 0: Disable 1: Enable	ALARM 0: Disable 1: Enable	"0" is read.	PAGE selection

Note: Please keep the setting order below of <ENATMR>, <ENAAML> and <INTENA>. Set difference time for Clock/Alarm setting and interrupt setting.

Example: Clock setting/Alarm setting

```
ld    (pager), 0ch    :    Clock, Alarm enable

ld    (pager), 8ch    :    Interrupt enable
```

PAGE	0	Select Page0
	1	Select Page1

ADJUST	0	Don't care
	1	Adjust sec. counter. When this bit is set to "1" the sec. counter becomes to "0" when the value of the sec. counter is 0-29. When the value of the sec. counter is 30-59, the min. counter is carried and sec. counter becomes "0". Output Adjust signal during 1 cycle of f_{SYS}. After being adjusted once, Adjust is released automatically. (PAGE0 only)

(11) Reset register (for PAGE0/1)

		7	6	5	4	3	2	1	0
RESTR (1328H)	Bit symbol	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	–	–	–	–
	Read/Write	W							
	Reset State	Undefined							
A Read-modify-write operation cannot be performed	Function	1Hz 0: Enable 1: Disable	16Hz 0: Enable 1: Disable	1:Clock reset	1:Alarm reset	Always write "0"			

RSTALM	0	Unused
	1	Reset alarm register

RSTTMR	0	Unused
	1	Reset Counter

<DIS1HZ>	<DIS16HZ>	PAGER<ENAALM>	Interrupt source signal
1	1	1	Alarm
0	1	0	1Hz
1	0	0	16Hz
Others			Output "0"

3.22.5 Operational description

(1) Reading clock data

1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can read correctly if reading data after 1Hz interrupt occurred.

2. Using two times reading

There is a possibility of incorrect clock data reading when the internal counter carries over. To ensure correct data reading, please read twice, as follows:

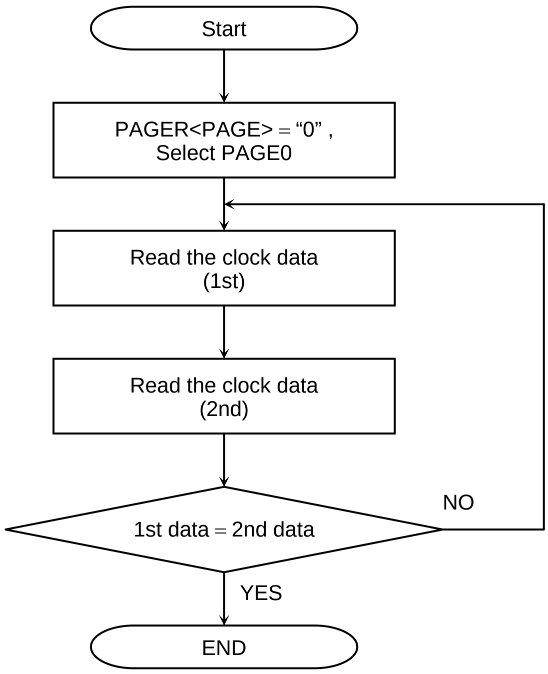


Figure 3.22.2 Flowchart of clock data read

(2) Writing clock data

When a carry over occurs during a write operation, the data cannot be written correctly. Please use the following method to ensure data is written correctly.

1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can write correctly if writing data after 1Hz interrupt occurred.

2. Resets counter

There are 15-stage counter inside the RTC, which generate a 1Hz clock from 32,768 kHz. The data is written after reset this counter.

However, if clearing the counter, it is counted up only first writing at half of the setting time, first writing only. Therefore, if setting the clock counter correctly, after clearing the counter, set the 1Hz-interrupt to enable. And set the time after the first interrupt (occurs at 0.5s) is occurred.

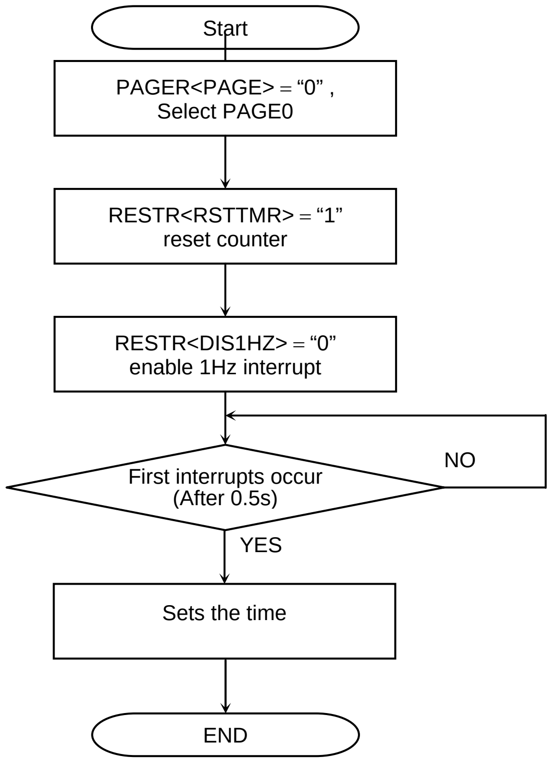


Figure 3.22.3 Flowchart of clock data read

3. Disabling the clock

A clock carry over is prohibited when “0” is written to PAGER<ENATMR> in order to prevent malfunction caused by the Carry hold circuit. While the clock is prohibited, the Carry hold circuit holds a one sec. carry signal from a divider. When the clock becomes enabled, the carry signal is output to the clock, the time is revised and operation continues. However, the clock is delayed when clock-disabled state continues for one second or more. Note that at this time system power is down while the clock is disabled. In this case the clock is stopped and clock is delayed.

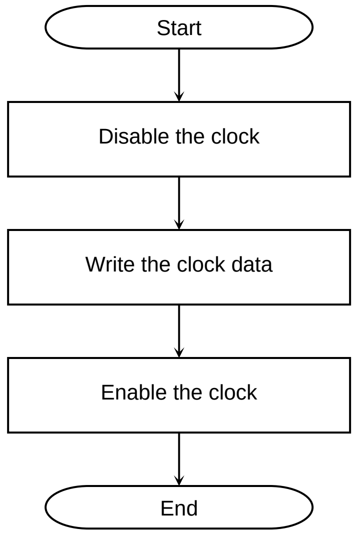


Figure 3.22.4 Flowchart of Clock disable

3.22.6 Explanation of the interrupt signal and alarm signal

The alarm function used by setting the PAGE1 register and outputting either of the following three signals from $\overline{\text{ALARM}}$ pin by writing "1" to PAGER<PAGE>. INTRTC outputs a 1-shot pulse when the falling edge is detected. RTC is not initialized by RESET. Therefore, when the clock or alarm function is used, clear interrupt request flag in INTC (interrupt controller).

(1) When the alarm register and the clock correspond, output "0".
(2) 1Hz Output clock.
(3) 16Hz Output clock.

(1) When the alarm register and the clock correspond, output "0"

When PAGER<ENAALM>= "1", and the value of PAGE0 clock corresponds with PAGE1 alarm register output "0" to $\overline{\text{ALARM}}$ pin and generate INTRTC.

The methods for using the alarm are as follows:

Initialization of alarm is done by writing in "1" to RESTR<RSTALM>. All alarm settings become Don't care. In this case, the alarm always corresponds with value of the clock, and if PAGER<ENAALM> is "1", INTRTC interrupt request is generated.

Setting alarm min., alarm hour, alarm date and alarm day is done by writing data to the relevant PAGE1 register.

When all setting contents correspond, RTC generates an INTRTC interrupt, if PAGER<INTENA><ENAALM> is "1". However, contents which have not been set up (don't care state) are always considered to correspond.

Contents which have already been set up, cannot be returned independently to the Don't care state. In this case, the alarm must be initialized and alarm register reset.

The following is an example program for outputting an alarm from $\overline{\text{ALARM}}$ -pin at noon (PM12:00) every day.

```
  LD   (PAGER), 09H    ;  Alarm disable, setting PAGE1
  LD   (RESTR), D0H    ;  Alarm initialize
  LD   (DAYR), 01H     ;  W0
  LD   (DATER),01H        1 day
  LD   (HOURR), 12H    ;  Setting 12 o'clock
  LD   (MINR), 00H     ;  Setting 00 min
                       ;  Set up time 31 μs (Note)
  LD   (PAGER), 0CH    ;  Alarm enable
( LD   (PAGER), 8CH    ;  Interrupt enable )
```

When the CPU is operating at high frequency oscillation, it may take a maximum of one clock at 32 kHz (about 30μs) for the time register setting to become valid. In the above example, it is necessary to set 31μs of set up time between setting the time register and enabling the alarm register.

Note: This set up time is unnecessary when you use only internal interruption.

(2) With 1Hz output clock

RTC outputs a clock of 1Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “0”, <DIS16HZ>= “1”. RTC also generates an INTRTC interrupt on the falling edge of the clock.

(3) With 16Hz output clock

RTC outputs a clock of 16Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “1”, <DIS16HZ>= “0”. RTC also generates INTRTC an interrupt on the falling edge of the clock.

3.23 Melody / Alarm generator (MLD)

The TMP92CF29A contains a melody function and alarm function, both of which are output from the MLDALM pin. Five kind of fixed cycle interrupt are generated by using a 15bit counter for use as the alarm generator.

The features are as follows.

1) Melody generator

The Melody function generates signals of any frequency (4Hz- 5461Hz) based on a low-speed clock (32.768 kHz) and outputs the signals from the MLDALM pin.

The melody tone can easily be heard by connecting an external loud speaker.

2) Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency (4096Hz) determined by the low-speed clock (32.768 kHz). This waveform can be inverted by setting a value to a register.

The alarm tone can easily be heard by connecting an external loud speaker.

Five kinds of fixed cycle interrupts are generated (1Hz, 2Hz, 64Hz, 512Hz, 8192Hz) by using a counter that is used for the alarm generator.

3.23.1 Block Diagram

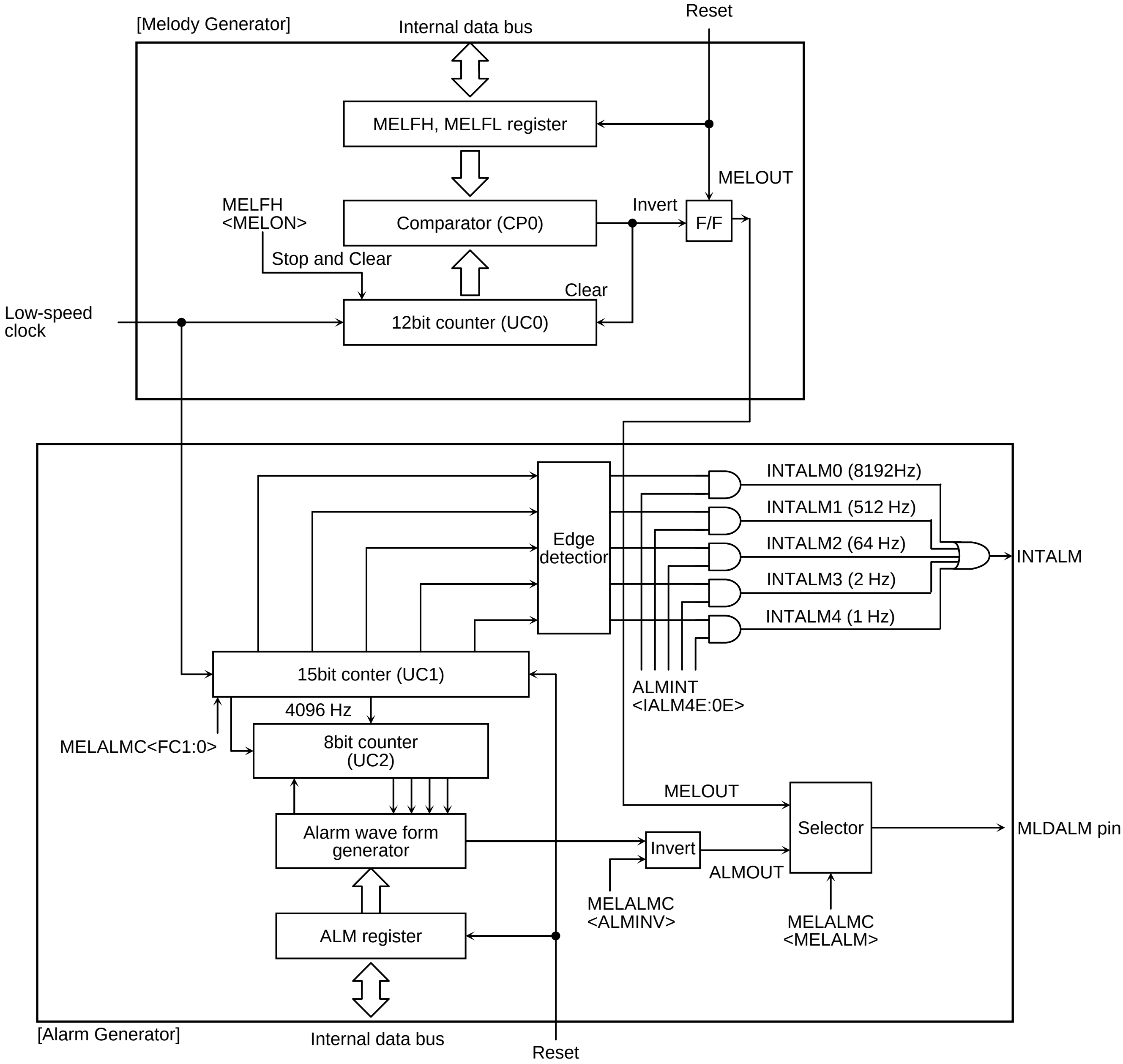


Figure 3.23.1MLD Block Diagram

3.23.2 Control registers

ALM register

ALM (1330H)		7	6	5	4	3	2	1	0
	bit Symbol	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Setting alarm pattern							

MELALMC register

MELALMC (1331H)		7	6	5	4	3	2	1	0
	bit Symbol	FC1	FC0	ALMINV	–	–	–	–	MELALM
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Free-run counter control 00: Hold 01: Restart 10: Clear & Stop 11: Clear & Start		Alarm Waveform invert 1:Invert	Always write “0”				Select Output Waveform 0: Alarm 1: Melody

Note1: MELALMC<FC1> is always read “0”.

Note2: When setting MELALMC register except <FC1:0> while the free-run counter is running, <FC1:0> is kept “01”.

MELFL register

MELFL (1332H)		7	6	5	4	3	2	1	0
	bit Symbol	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Setting melody frequency (lower 8bit)							

MELFH register

MELFH (1333H)		7	6	5	4	3	2	1	0
	bit Symbol	MELON				ML11	ML10	ML9	ML8
	Read/Write	R/W				R/W			
	Reset State	0				0	0	0	0
	Function	Control melody counter 0: Stop & Clear 1: Start				Setting melody frequency(upper 4bit)			

ALMINT register

ALMINT (1334H)		7	6	5	4	3	2	1	0
	bit Symbol			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
	Read/Write			R/W					
	Reset State			0	0	0	0	0	0
	Function			Always write “0”	1:INTALM4 (1Hz) enable	1:INTALM3 (2Hz) enable	1:INTALM2 (64Hz) enable	1:INTALM1 (512Hz) enable	1:INTALM0 (8192Hz) enable

Note: INTALM0 to INTALM4 prohibit that set to enable at same time. If setting to enable, set only 1.

3.23.3 Operational Description

3.23.3.1 Melody generator

The Melody function generates signals of any frequency (4Hz-5461Hz) based on a low-speed clock (32.768kHz) and outputs the signals from the MLDALM pin.

The melody tone can easily be heard by connecting an external loud speaker.

(Operation)

MELALMC<MELALM> must first be set as 1 in order to select the melody waveform to be output from MLDALM. The melody output frequency must then be set to 12-bit register MELFH, MELFL.

The following are examples of settings and calculations of melody output frequency.

(Formula for calculating melody waveform frequency)

@fs = 32.768 [kHz]

Melody output waveform $f_{MLD}[Hz] = 32768/(2 \times N + 4)$

Setting value for melody $N = (16384/f_{MLD}) - 2$

(Note: N = 1~4095 (001H~FFFH), 0 is not acceptable)

(Example program)

When outputting an “A” musical note (440Hz)

```
LD    (MELALMC), −−XXXXX1B    ;  Select melody waveform
LD    (MELFL), 23H            ;  N = 16384/440 − 2 = 35.2 = 023H
LD    (MELFH), 80H            ;  Start to generate waveform
```

(Refer: Basic musical scale setting table)

Scale	Frequency [Hz]	Register Value: N
C	264	03CH
D	297	035H
E	330	030H
F	352	02DH
G	396	027H
A	440	023H
B	495	01FH
C	528	01DH

3.23.3.2 Alarm generator

The Alarm function generates eight kinds of alarm waveform having a modulation frequency of 4096Hz determined by the low-speed clock (32.768 kHz). This waveform is reversible by setting a value to a register.

The alarm tone can easily be heard by connecting an external loud speaker.

Five kind of fixed cycle (interrupts can be generated 1Hz, 2Hz, 64Hz, 512Hz, 8192Hz) by using a counter which is used for the alarm generator.

(Operation)

MELALMC<MELALM> must first be set as 0 in order to select the alarm waveform to be output from MLDALMC. The "10" must be set on the MELALMC <FC1:0> register, and clear internal counter. Alarm pattern must then be set on the 8-bit register of ALM. If it is inverted output-data, set <ALMINV> as invert.

Then set the MELAMC<FC1:0> to "11" to start the free-run counter.

To stop the alarm output, write "00H" to the ALM register.

The following are examples of program, setting value of alarm pattern and waveform of each setting value.

(Setting value of alarm pattern)

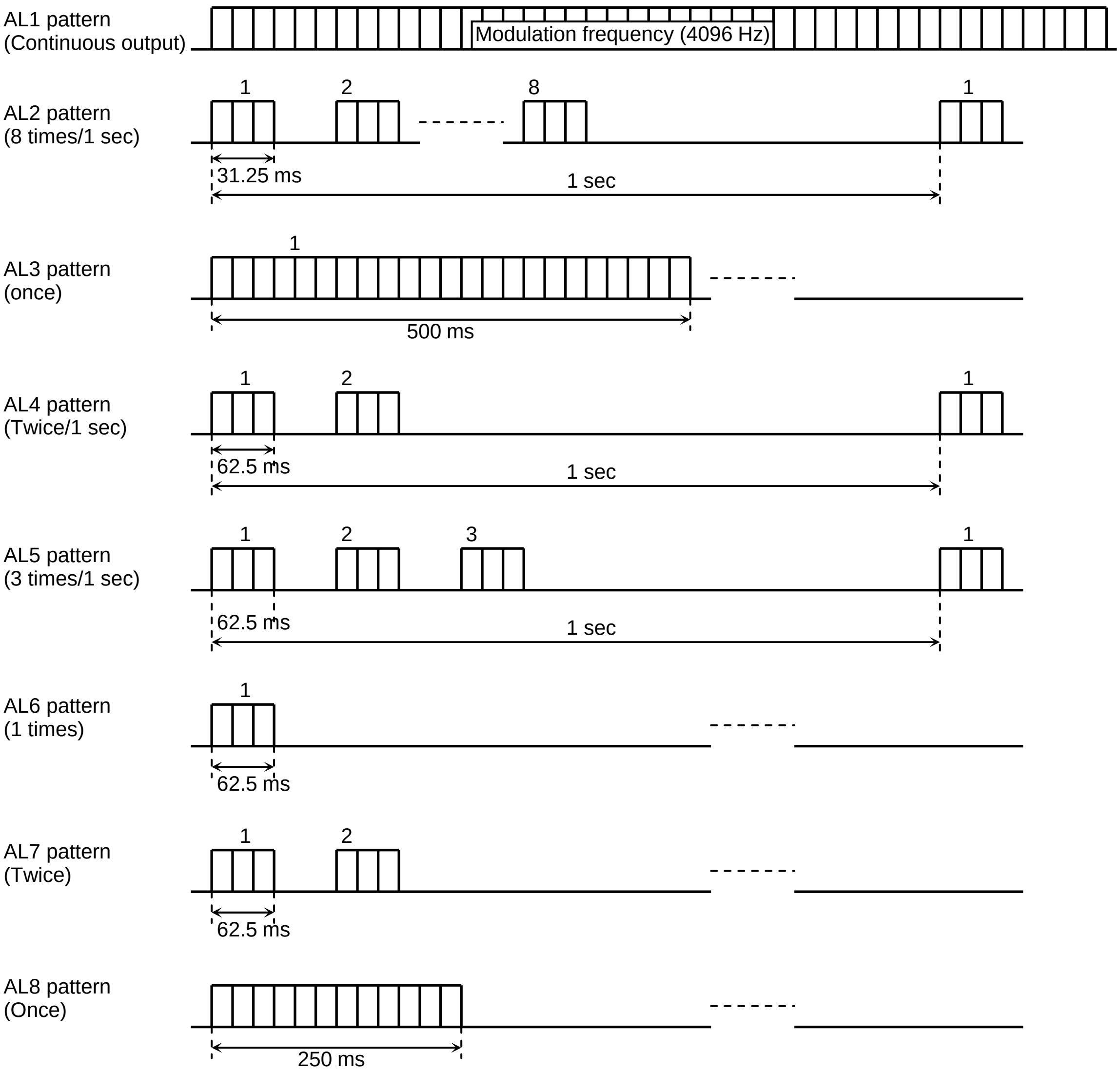
Setting value for ALM register	Alarm waveform
00H	"0" fixed
01H	AL1 pattern
02H	AL2 pattern
04H	AL3 pattern
08H	AL4 pattern
10H	AL5 pattern
20H	AL6pattern
40H	AL7 pattern
80H	AL8 pattern
Other	Undefined (Do not set)

(Example program)

When outputting AL2 pattern (31.25ms/8 times/1sec)

```
LD    (MELALMC), 80H    ;  Clear counter, set output alarm waveform
LD    (ALM), 02H        ;  Set AL2 pattern
LD    (MELALMC), C0H    ;  Free-run counter start
```

Example: Waveform of alarm pattern for each setting value: not inverted)



3.24 Analog-Digital Converter (ADC)

A 10-bit serial conversion analog/digital converter (AD converter) having six channels of analog input is built in.

Figure 3.24.1 shows the block diagram of the AD converter.

The 6-analog input channels (AN0-AN5) can be used as general-purpose inputs.

> Note1: To reduce the power supply current by IDLE2, IDLE1, STOP or PCM mode, the standby state may be maintained with the internal comparator still being enabled, depending on the timing. Check that the AD converter operation is in a stop before executing HALT instruction. In IDLE2 mode it operates only the case of ADMOD0<I2AD>= "0".
>
> Note2: Setting ADMOD1<DACON> = "0" while the AD converter is in a stop can reduce current consumption.

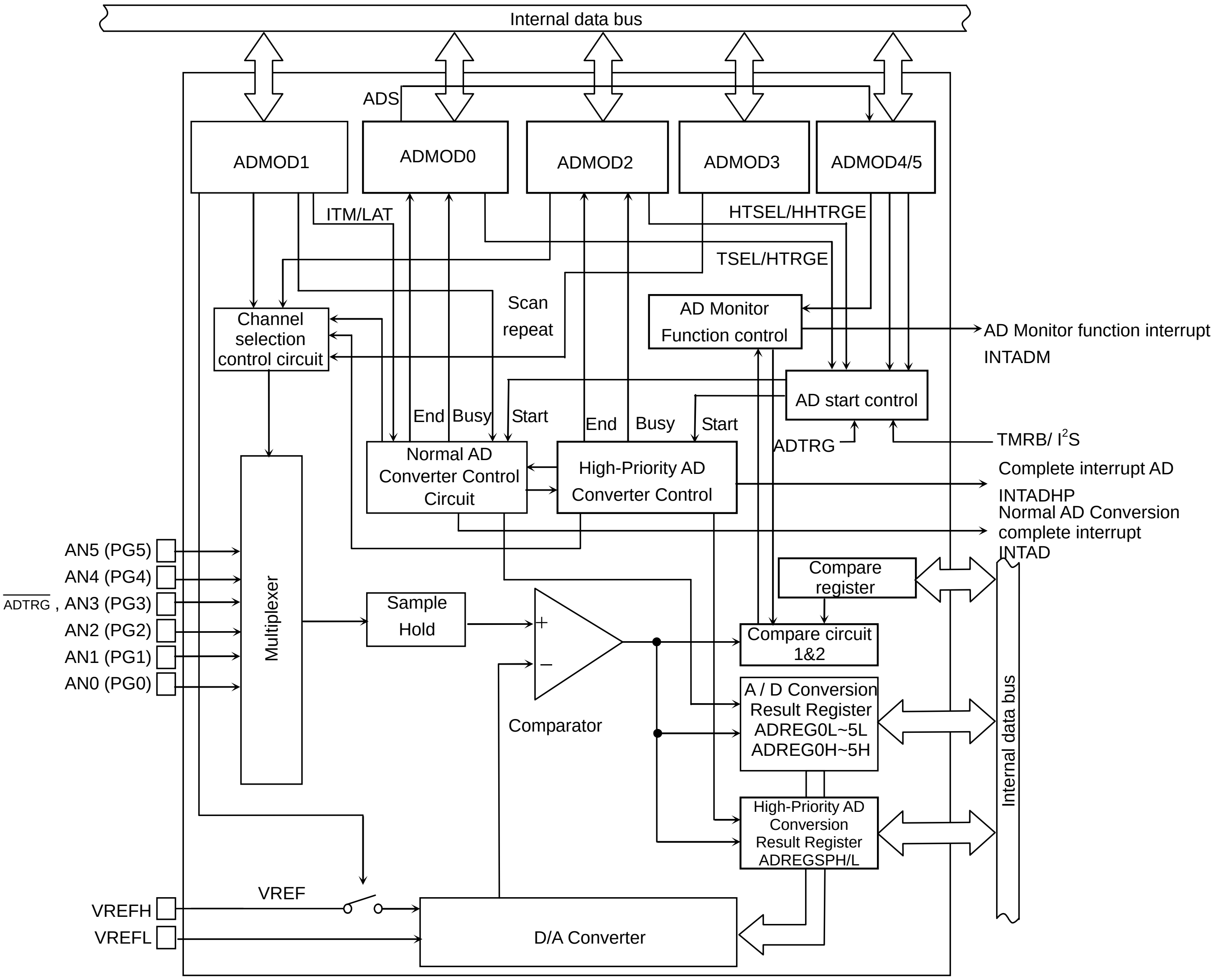


Figure 3.24.1 ADC Block Diagram

3.24.1 Control register

The AD converter is controlled by the AD mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3, ADMOD4 and ADMOD5). AD conversion results are stored in the six registers of AD conversion result higher-order/lower-order registers ADREG0H/L to ADREG5H/L. Top-priority conversion results are stored in ADREGSPH/L.

Figure 3.24.2 to Figure 3.24.11 show the registers available in the AD converter.

AD Mode Control Register 0 (Normal conversion control)

ADMOD0 (12B8H)

	7	6	5	4	3	2	1	0
bit Symbol	EOS	BUSY		I2AD	ADS	HTRGE	TSEL1	TSEL0
Read/Write	R			R/W				
Reset State	0	0		0	0	0	0	0
Function	Normal AD conversion end flag 0:During conversion sequence or before starting 1:Complete conversion sequence	Normal AD conversion BUSY Flag 0:Stop conversion 1:During conversion		AD conversion when IDLE2 mode 0: Stop 1: Operate	Start Normal AD conversion 0: Don't Care 1:Start AD conversion Always read as"0".	Normal AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB00 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: Reserved	

Figure 3.24.2 AD Conversion Registers

AD Mode Control Register 1 (Normal conversion control)

ADMOD1 (12B9H)		7	6	5	4	3	2	1	0
	bit Symbol	DACON	ADCH2	ADCH1	ADCH0	LAT	ITM	REPEAT	SCAN
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	DAC and VREF application control	Analog input channel select			Latency 0: No Wait 1:Start after reading conversion result store Register of last channel	Interrupt specification when conversion channel-fix repeat mode	Repeat mode specification 0:Single conversion 1:Repeat conversion	Scan mode specification 0: Channel-fix mode 1: Channel scan mode

Specify AD conversion interrupt for Channel Fixed Repeat Conversion mode

	Channel Fixed Repeat Conversion Mode <SCAN> = "0", <REPEAT> = "1"
0	Generates interrupt every conversion
1	Generated interrupt every fourth conversion

Next SCAN start timing control for the channel-scan repeat mode

	Channel Scan Repeat mode (<SCAN> = "1", <REPEAT> = "1")
0	No Wait
1	Start after read last of conversion result store Register

Analog input channel select

<ADCH2:0> \ <SCAN>	0: Channel -fix	1: Channel-scan
000	AN0	AN0
001	AN1	AN0→AN1
010	AN2	AN0→AN1→AN2
011	AN3(note)	AN0→AN1→AN2→AN3 (note)
100	AN4	AN0→AN1→AN2→AN3→AN4 (note)
101	AN5	AN0→AN1→AN2→AN3→AN4→AN5 (note)
110	Reserved	
111	Reserved	

DAC & VREF application control

0	DAC & VREF off (Set before into STOP mode)
1	DAC & VREF on (Set to "1" before starting conversion)

Figure 3.24.3 AD Converter Related Register

AD Mode Control Register 2 (Top-priority conversion control)

		7	6	5	4	3	2	1	0
ADMOD2 (12BAH)	bit Symbol	HEOS	HBUSY			HADS	HHTRGE	HTSEL1	HTSEL0
	Read/Write	R				R/W			
	Reset State	0	0			0	0	0	0
	Function	Top-priority AD conversion sequence FLAG 0: During conversion sequence or before starting 1: Complete conversion sequence	Top-priority AD conversion BUSY Flag 0:Stop conversion 1:During conversion			Start Top-priority AD conversion 0: Don't Care 1: Start AD conversion Always read as"0".	Top-priority AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB10 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: I^2S Sampling Counter Output	

AD Mode Control Register 3 (Top-priority conversion control)

		7	6	5	4	3	2	1	0
ADMOD3 (12BBH)	bit Symbol	–	HADCH2	HADCH1	HADCH0				–
	Read/Write	R/W							R/W
	Reset State	0	0	0	0				0
	Function	Always write "0".	Top-priority analog input channel select						Always write "0".

Analog input channel select

<HADCH2:0>	Analog input channel when High-priority conversion
000	AN0
001	AN1
010	AN2
011	AN3(Note)
100	AN4
101	AN5
110	Reserved
111	Reserved

Note: When using PG3 pin as $\overline{\text{ADTRG}}$, it cannot be set.

Figure 3.24.4 AD Conversion Registers

AD Mode Control Register 4 (AD Monitor function control)

ADMOD4 (12BCH)

		7	6	5	4	3	2	1	0
ADMOD4 (12BCH)	bit Symbol	CMEN1	CMEN0	CMP1C	CMP0C	IRQEN1	IRQEN0	CMPINT1	CMPINT0
	Read/Write	R/W						R	
	Reset State	0	0	0	0	0	0	0	0
	Function	AD Monitor function1 0: Disable 1: Enable	AD Monitor function0 0: Disable 1: Enable	Generation condition of AD monitor function interrupt 1 0: less than 1: Greater than or Equal	Generation condition of AD monitor function interrupt 0 0: less than 1: Greater than or Equal	AD monitor function interrupt 1 0: Disable 1: Enable (Note)	AD monitor function interrupt 0 0: Disable 1: Enable (Note)	Status of AD monitor function interrupt 1 0: No generation 1: Generation	Status of AD monitor function interrupt 0 0: No generation 1: Generation

Note: When AD monitor function interrupts generate, it is cleared automatically and it is set to disable condition.

AD Mode Control Register 5 (AD Monitor function control)

		7	6	5	4	3	2	1	0
ADMOD5 (12BDH)	bit Symbol		CM1CH2	CM1CH1	CM1CH0		CM0CH2	CM0CH1	CM0CH0
	Read/Write		R/W				R/W		
	Reset State		0	0	0		0	0	0
	Function		Select analog channel for AD monitor function 1 000: AN0 100: AN4 001: AN1 101: AN5 010: AN2 110: Reserved 011: AN3 111: Reserved				Select analog channel for AD monitor function 0 000: AN0 100: AN4 001: AN1 101: AN5 010: AN2 110: Reserved 011: AN3 111: Reserved		

Note1: When converting AD in hard ware trigger by setting <HHTRGE> and <HTRGE>to “1”, set PGFC<PG3F> to “1” (as ADTRG) in case of external TRG before enabling it. When using an INTTBx0 of 16-bit timer, first set the <TSEL1:0> or <HTSEL1:0> bit to “00” when the timer is not operating. Then, set the <HHTRGE> and <HTRGE> to “1” and enable trigger operation. Finally, operate the timer so that AD conversion will be initiated at constant intervals.

Note 2: When disabling an external trigger ($\overline{\text{ADTRG}}$) for AD conversion, first clear the <HHTRGE> or <HTRGE> bit to “0”, and clear the PGFC<PG3F> to “0”, thus configuring port G as a general-purpose port.

Note 3: When starting AD by using external trigger (ADTRG), it can be started after enabling (<HHTRGE> = “1” or <HTRGE> = “1”) and 3 clock at f_{SYS} was executed. AD is not started when before that time.

Note 4: When chaging compare register value of AD Monitor function, change it after setting AD Monitor function to disable(ADMOD4<CMEN1:0> = “0”).

Figure 3.24.5 AD Conversion Registers

AD Conversion Result Register 0 Low

		7	6	5	4	3	2	1	0
ADREG0L (12A0H)	bit Symbol	ADR01	ADR00					OVR0	ADR0RF
	Read/Write	R						R	
	Reset State	0	0					0	0
	Function	Store Lower 2 bits of AN0 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 0 High

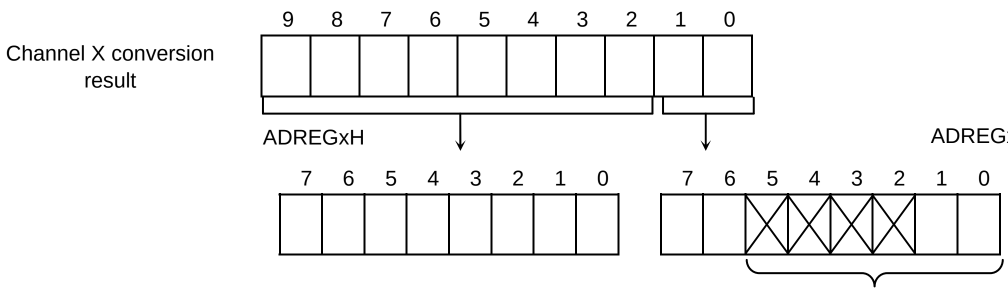
		7	6	5	4	3	2	1	0
ADREG0H (12A1H)	bit Symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of AN0 AD conversion result							

AD Conversion Result Register 1 Low

		7	6	5	4	3	2	1	0
ADREG1L (12A2H)	bit Symbol	ADR11	ADR10					OVR1	ADR1RF
	Read/Write	R						R	
	Reset State	0	0					0	0
	Function	Store Lower 2 bits of AN1 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 1 High

		7	6	5	4	3	2	1	0
ADREG1H (12A3H)	bit Symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of AN1 AD conversion result							



- Bits 5 ~ 2 are always read as "0".
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to "1". When Lower register (ADRECxL) is read, this bit is cleared to "0".
- Bit 1 is the Overrun flag <OVRx>. This bit is set to "1" if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to "0" by reading Flag.

Figure 3.24.6 AD Conversion Registers

AD Conversion Result Register 2 Low

ADREG2L (12A4H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR21	ADR20					OVR2	ADR2RF
	Read/Write	R						R	
	Reset State	0	0					0	0
	Function	Store Lower 2 bits of AN2 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 1 High

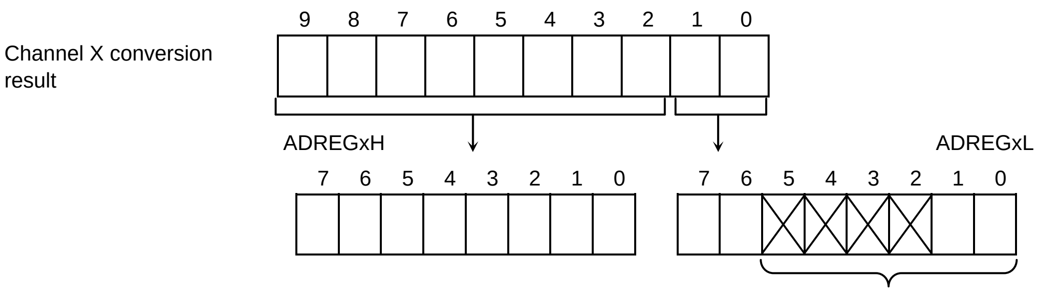
ADREG2H (12A5H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of AN2 AD conversion result							

AD Conversion Result Register 3 Low

ADREG3L (12A6H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR31	ADR30					OVR3	ADR3RF
	Read/Write	R						R	
	Reset State	0	0					0	0
	Function	Store Lower 2 bits of AN3 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 3 High

ADREG3H (12A7H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of AN3 AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.24.7 AD Conversion Registers

AD Conversion Result Register 4 Low

ADREG4L (12A8H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR41	ADR40					OVR4	ADR4RF
Read/Write	R						R	
Reset State	0	0					0	0
Function	Store Lower 2 bits of AN4 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 4 High

ADREG4H (12A9H)

	7	6	5	4	3	2	1	0
bit Symbol	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
Read/Write	R							
Reset State	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN4 AD conversion result							

AD Conversion Result Register 5 Low

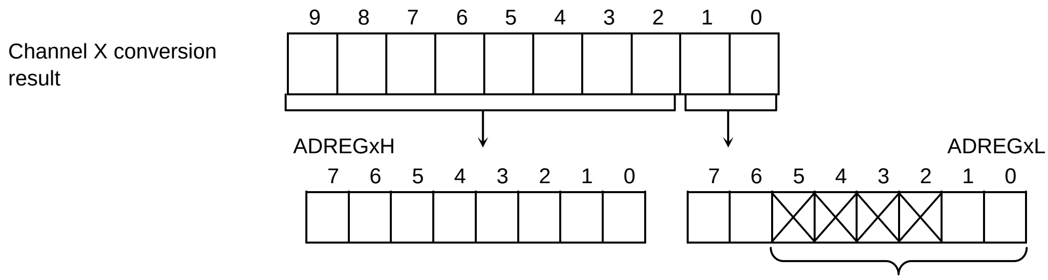
ADREG5L (12AAH)

	7	6	5	4	3	2	1	0
bit Symbol	ADR51	ADR50					OVR5	ADR5RF
Read/Write	R						R	
Reset State	0	0					0	0
Function	Store Lower 2 bits of AN5 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

AD Conversion Result Register 5 High

ADREG5H (12ABH)

	7	6	5	4	3	2	1	0
bit Symbol	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
Read/Write	R							
Reset State	0	0	0	0	0	0	0	0
Function	Store Upper 8 bits of AN5 AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

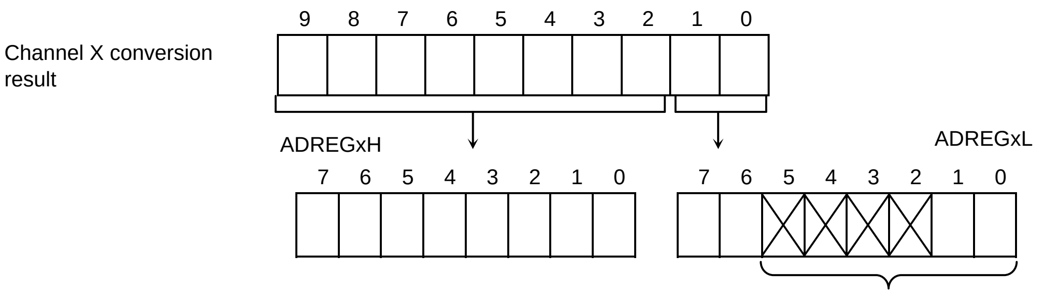
Figure 3.24.8 AD Conversion Registers

Top-priority AD Conversion Result Register SP Low

		7	6	5	4	3	2	1	0
ADREGSPL (12B0H)	bit Symbol	ADRSP1	ADRSP0					OVSRP	ADRSPRF
	Read/Write	R						R	
	Reset State	0	0					0	0
	Function	Store Lower 2 bits of an AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored

Top-priority AD Conversion Result Register SP High

		7	6	5	4	3	2	1	0
ADREGSPH (12B1H)	bit Symbol	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
	Read/Write	R							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of an AD conversion result							



- Bits 5 ~ 2 are always read as “0”.
- Bit 0 is the AD conversion result store flag <ADRxRF>. When AD conversion result is stored, the flag is set to “1”. When Lower register (ADRECxL) is read, this bit is cleared to “0”.
- Bit 1 is the Overrun flag <OVRx>. This bit is set to “1” if a next conversion result is written to the ADREGxH/L before both the ADREGxH and ADREGxL are read. This bit is cleared to “0” by reading Flag.

Figure 3.24.9 AD Conversion Registers

AD Conversion Result Compare Criterion Register 0 Low

ADCM0REGL (12B4H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR21	ADR20						
	Read/Write	R/W							
	Reset State	0	0						
	Function	Store Lower 2 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 0 High

ADCM0REGH (12B5H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 1 Low

ADCM1REGL (12B6H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR21	ADR20						
	Read/Write	R/W							
	Reset State	0	0						
	Function	Store Lower 2 bits of an AD conversion result compare criterion							

AD Conversion Result Compare Criterion Register 1 High

ADCM1REGH (12B7H)		7	6	5	4	3	2	1	0
	bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R/W							
	Reset State	0	0	0	0	0	0	0	0
	Function	Store Upper 8 bits of an AD conversion result compare criterion							

Note: Disable the AD monitor function (ADMOD4<CMEN1:0> = “0”) before attempting to set or modify the value of these registers.

Figure 3.24.10 AD Conversion Registers

AD Conversion Clock Setting Register

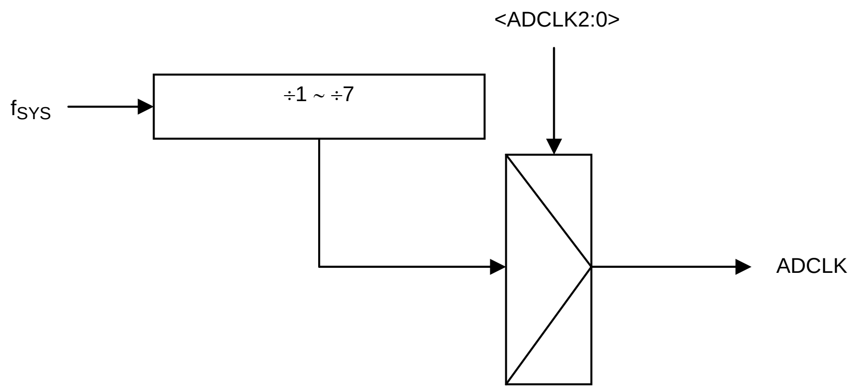
ADCCLK (12BFH)

	7	6	5	4	3	2	1	0
bit Symbol					–	ADCLK2	ADCLK1	ADCLK0
Read/Write					R/W			
Reset State					0	0	0	0
Function					Always write "0"	Select clock for AD conversion 000: Reserved 100: f_{IO}/4 001: f_{IO}/1 101: f_{IO}/5 010: f_{IO}/2 110: f_{IO}/6 011: f_{IO}/3 111: f_{IO}/7		

Note1: AD conversion is executed at the clock frequency selected in the above register. To assure conversion accuracy, however, the conversion clock frequency must not exceed 12MHz.

Note2: Don 't change the clock frequency while AD conversion is in progress.

Figure 3.24.11 AD Conversion Registers



f_{IO}(f_{SYS}/2)	<ADCLK2:0>	ADCLK	AD conversion speed
40MHz	100(f_{IO}/4)	10.0MHZ	12 μsec
	101(f_{IO}/5)	8MHZ	15 μsec
30MHz	011(f_{IO}/3)	10.0MHZ	12 μsec
	100(f_{IO}/4)	7.5MHZ	16 μsec

AD conversion speed can be calculated by following.

Conversion speed = 120 × (1/ADCLK)

3.24.2 Operation

3.24.2.1 Analog Reference Voltages

Apply the analog reference voltage's “H” level side to the VREFH pin and the “L” level side to the VREFL pin.

3.24.2.2 Selecting Analog Input Channels

Selecting an analog input channel depends on the operation mode of the AC converter.

(1) For normal AD conversion

When using an analog input channel in fix mode, select one channel from the AN0 to AN5 pins by setting (ADMOD1<SCAN> = “0”) ADMOD1<ADCH2:0>.

When using an analog input channel in scan mode, select one scan mode from the six scan modes by setting (ADMOD1<SCAN> = “1”) ADMOD1 <ADCH2:0>.

(2) For top-priority AD conversion

Select one channel from the analog input pins AN0 to AN5 by setting ADMOD3<HADCH2:0>.

After reset, ADMOD1<SCAN> is initialized to “0” and ADMOD1<ADCH2:0> to “000”. Since these settings are used for channel selection, the channel fixed input with the AN0 pin will be selected. Pins not used as analog input channels can be used as normal ports.

3.24.2.3 Starting an AD Conversion

The AD conversion has the two types of normal AD conversion and top-priority AD conversion.

Normal AD conversion can be started up by setting ADMOD0<ADS> to “1.” Top-priority AD conversion can be started up by software by setting ADMOD2<HADS> to “1.”

For normal AD conversion, one operation mode is selected from the four types of operation modes specified by ADMOD1<REPEAT, SCAN>. The operation mode for top-priority AD conversion is only single conversion by channel-fix mode.

The ADC supports two types of AD conversion: normal AD conversion and Top-priority AD conversion. The ADC initiates a normal AD conversion by software when the ADMOD0<ADS> is set to “1”. It initiates a Top-priority AD conversion by software when the ADMOD2<HADS> is set to “1”. For a normal AD conversion, ADMOD1<REPEAT, SCAN> select one of four conversion modes. For a Top-priority AD conversion, the ADC only supports Fixed-Channel Single Conversion mode.

The ADMOD0<TSEL1:0> and ADMOD2<HTSEL1:0> enable a hardware trigger for a normal and Top-priority AD conversion, respectively. When these bits are set to “10”, a normal or Top-priority AD conversion is triggered by a falling edge applied to $\overline{\text{ADTRG}}$ pin. When ADMOD0<TSEL1:0> is set to “00”, a normal AD conversion is triggered by INTTB00 of 16-Bit Timer interrupt. When ADMOD2<HTSEL1:0> is set to “00”, a Top-priority AD conversion is triggered by INTTB10 of 16-Bit Timer interrupt. If this bit is “11”, it is triggered by I^{2}S sampling block. Even when a hardware trigger is enabled, software starting can be used.

Note: If changing HTSEL at HHTRGE is “ON”, maybe unexpected interrupts occurs. If changing HTSEL, once set HHTRGE to “OFF”.

When normal AD conversion is started, the AD conversion BUSY flag (ADMOD0<BUSY>) that shows the state for AD being converted is set to “1.”

When top-priority AD conversion is started, the AD conversion BUSY flag (ADMOD2<HBUSY>) that shows the state for AD being converted is set to “1.”

In addition, when top-priority conversion is started during normal AD conversion, ADMOD0<BUSY> is kept to “1.”

<HEOS> and <EOS> are set to "1" after conversion is completed. This flag is cleared to "0" only when read.

During a normal AD conversion, writing a "1" to ADMOD0<ADS> causes the ADC to abort any ongoing conversion immediately, and restart.

During a normal AD conversion, if normal AD conversion starting is enabled by hard ware trigger, normal AD conversion is restarted when start condition from hard ware trigger is satisfied. When restart is set, normal AD conversion is aborted immediately.

During a normal AD conversion, if a Top-priority AD conversion starts (writing a "1" to ADMOD2<HADS> or a hard ware trigger occurs), the ADC aborts any ongoing conversion immediately, and then start a Top-priority AD conversion for the channel specified by ADMOD3<HADCH2:0>. Upon the completion of the Top-priority conversion, the ADC stores the conversion result to ADREGSPH/L, and then resumes the suspended normal conversion with that channel.

Note: It cannot overlap with three or more AD conversions.
Prohibition example 1: In FIRST normal AD conversion
→ (Before finished FIRST normal AD conversion) Started SECOND normal AD conversion
→ (Before finished SECOND normal AD conversion) Started THIRD normal AD conversion
Prohibition example 2: In FIRST normal AD conversion
→ (Before finished FIRST normal AD conversion) Started SECOND normal AD conversion
→ (Before finished SECOND normal AD conversion) Started THIRD high-priority AD conversion

3.24.2.4 AD Conversion Modes and AD Conversion-End Interrupts

For AD conversion, the following four operation modes are provided: For normal AD conversion, selection is available by setting ADMOD1<REPEAT and SCAN>. As for top-priority AD conversion, only single conversion mode by channel-fix mode is available.

a. Channel-fix single conversion mode

b. Channel-scan single conversion mode

c. Channel-fix repeat conversion mode

d. Channel-scan repeat conversion mode

(1) Normal AD conversion

To select operation modes, use ADMOD1<REPEAT, SCAN>. After AD conversion is started, ADMOD0<BUSY> is set to "1." When a specified AD conversion ends, the Normal AD conversion end interrupt (INTAD) is generated, which sets "1" in ADMOD0<EOS> is set "1", that shows the end of the AD conversion sequence.

a. Channel-fix single conversion mode

Setting ADMOD1<REPEAT, SCAN> to "00" selects the channel-fix single conversion mode.

This mode performs a conversion only one time at one channel selected. After conversion ends, ADMOD0<EOS> is set to "1," generating Normal AD conversion End an INTAD interrupt request. <EOS> is cleared to "0" only by being read.

b. Channel-scan single conversion mode

Setting ADMOD1<REPEAT, SCAN> to "01" selects the channel-scan single conversion mode.

This mode performs a conversion only one time at each scan channel selected. After scan conversion ends, ADMOD0<EOS> is set to "1," generating Normal AD conversion End interrupt request. <EOS> is cleared to "0" only by being read.

c. Channel-fix repeat conversion mode

Setting ADMOD1<REPEAT, SCAN> to "10" selects the channel-fix repeat conversion mode.

This mode performs a conversion at one channel selected repeatedly. After conversion ends, ADMOD0<EOS> is set to "1." The timing of Normal AD conversion End INTAD interrupt request generation can be selected by setting ADMOD1 <ITM>. The timing of <EOS> being set is also liked to the interrupt timing.

ADMOD0<EOS> is cleared to "0" only by being read.

Setting <ITM> to "0" generates an interrupt request each time an AD conversion ends. In this case, conversion results are always stored into the storage register of ADREGxH/L. At the point of storage, <EOS> is set to 1.

Setting <ITM> to "1" generates an interrupt request each time four AD conversions end. In this case, conversion results are stored into the storage registers of ADREG0H/L to ADREG3H/L one after another. After stored into ADREG3, <EOS> is set to "1," restarting storage from ADREG0. ADMOD0<EOS> is set to "1" after a forth conversion result is stored. <EOS> is cleared to "0" only by being read.

d. Channel-scan repeat conversion mode

Setting ADMOD1<REPEAT, SCAN> to “11” selects the channel-scan repeat conversion mode.

This mode performs a conversion at selected scan channels repeatedly. Each time after the conversion at a final channel ends, ADMOD0<EOS> is set to “1,” generating Normal AD conversion End interrupt request. <EOS> is cleared to “0” only by being read.

To stop the repeat conversion mode (mode of c and d) operation, write “0” in ADMOD1<REPEAT>. At the point when a scan conversion being executed ends, the repeat conversion mode ends.

Shift to a standby mode (IDLE2 Mode with ADMOD0<I2AD> = “0”, IDLE1 Mode or STOP Mode) immediately stops operation of the AD converter even if AD conversion is still in progress. Therefore, ADC may consume current even if operation is stopped, depending on stop condition of ADC that switches to standby mode. For avoiding this problem, Stop ADC before switching to standby mode.

(2) Top-priority AD conversion

The operation mode is only single conversion by channel-fix mode. The settings in ADMOD1<REPEAT, SCAN> are not involved.

When startup conditions are established, a conversion at a channel specified by ADMOD3<HADCH2:0> is performed only one time. When conversion ends, the top-priority AD conversion end interrupt (INTADHP) is generated, which sets “1” in ADMOD2<HEOS>. The HEOS flag is cleared to “0” only by being read.

Table 3.24.1 Interrupt Generation Timing and Flag Setting in Each AD Conversion Mode

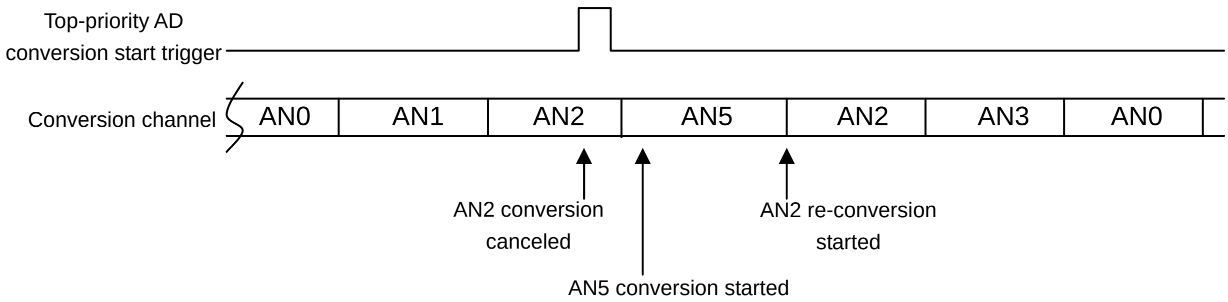
Conversion mode	Interrupt Generation Timing	EOS set timing (Note)	ADMOD1		
			ITM	REPEAT	SCAN
Channel-fix Single conversion	After conversion end	After conversion end	–	0	0
Channel-fix Repeat conversion	Per one conversion	Each time after one conversion ends	0	1	0
	Per four conversions	Each time after four conversions end	1		
Channel-scan Single conversion	After scan conversion end	After scan conversion end	–	0	1
Channel-scan Repeat conversion	Each time after one scan conversion ends	Each time after one scan conversion ends	–	1	1

Note: EOS is cleared to “0” only by reading this bit.

3.24.2.5 Top-Priority Conversion Mode

The ADC can perform a Top-priority AD conversion while it is performing a normal AD conversion sequence. A Top-priority AD conversion can be started at software by setting the ADMOD2<HADS> to "1". It is also triggered by a hardware trigger if so enabled using ADMOD2<HTSEL1:0>. If a Top-priority AD conversion is triggered during a normal AD conversion, the ADC aborts any ongoing conversion immediately, and then begins a single Top-priority AD conversion for the channel specified with the ADMOD3<HADC2:0>. Upon the completion of the Top-priority AD conversion, the ADC stores the results of the conversion in the ADREGSPH/L, generates the Top-priority AD conversion interrupt (INTADHP), and then resumes the suspended normal conversion with that channel. While a Top-priority conversion is being performed, a trigger for another Top-priority conversion is ignored.

Example: When AN5 top-priority AD conversion is started up with ADMOD3<HADCH2:0> = "101" during repeat scan conversion at channels AN0 to AN3 with ADMOD1<REPEAT, SCAN> = "11" and ADMOD1<ADCH2:0> = "011"



3.24.2.6 AD Monitor Function

Setting ADMOD4<CMEN1:0> to "1" enables the AD monitoring function.

The value of Result storage register that is appointed by ADMOD5 is compared with the value of AD conversion result register (H/L), ADMOD4<CMP1C:0C> can select greater or smaller of comparison format. As register ADMOD4<IRQEN1:0> is Enable,

This comparison operation is performed each time when a result is stored in the corresponding conversion result storage register. When conditions are met, the interrupt is generated. Be careful that the storage registers assigned for the AD monitoring function are usually not ready by software, which means that the overrun flag <OVRx> is always set and the conversion result storage flag <ADRxRF> is also set.

If each of them is assigned to separate channels, the monitoring of greater or smaller is possible in the two analog channels. In addition, if assigned to the same channels, the monitoring with the voltage range set is possible.

3.24.2.7 AD Conversion Time

One AD conversion takes 120 clocks including sampling clocks. The AD conversion clock is selected from 1/1 to 1/7 f_{IO} by ADCLK <ADCLK2:0>. To meet the guaranteed accuracy, the AD conversion clock needs to be set to 12 MHz or less; or equivalently 10 μs or more of AD conversion time.

3.24.2.8 Storing and Read of AD Conversion Results

AD conversion results are stored in the AD conversion result higher-order/lower-order registers (ADREG0H/L~ ADRG5H/L) for the normal AD conversion (ADREG0H/L to ADREG5H/L are read-only registers)

In the channel-fix repeat conversion mode, AD conversion results are stored into ADREG0H/L to ADREG3H/L one after another. In other modes, the conversion results of channels AN0, AN1, AN2, AN3, AN4, and AN5 are each stored into ADREG0H/L, ADREG1H/L, ADREG2H/L, ADREG3H/L, ADREG4H/L, and ADREG5H/L.

Table 3.24.2 shows the correspondence between analog input channels and AD conversion result registers.

Table 3.24.2 Correspondence between analog input channels and AD conversion result registers

<table>
<tr><th rowspan="2">Analog input channel (Port G)</th><th colspan="2">AD Conversion result registers</th></tr>
<tr><th>Other conversion modes than shown in the right</th><th>Channel-fix repeat conversion mode (per 4 times)</th></tr>
<tr><td>AN0</td><td>ADREG0H/L</td><td rowspan="6">ADREG0H/L ←
↓
ADREG1H/L
↓
ADREG2H/L
↓
ADREG3H/L ─┘</td></tr>
<tr><td>AN1</td><td>ADREG1H/L</td></tr>
<tr><td>AN2</td><td>ADREG2H/L</td></tr>
<tr><td>AN3</td><td>ADREG3H/L</td></tr>
<tr><td>AN4</td><td>ADREG4H/L</td></tr>
<tr><td>AN5</td><td>ADREG5H/L</td></tr>
</table>

Note: In order to detect overruns without omission, read the conversion result storage register's higher-order bits first, and than read the lower-order bits next. As this result, receiving the result of OVRn = "0" and ADRnRF = "1" for overruns existing in the lower-order bits means that a correct conversion result has been obtained.

3.24.2.9 Data Polling

To process AD conversion results by using data polling without using interrupts, perform a polling on ADMOD0<EOS>. After confirming that ADMOD0<EOS> is set to "1," read the AD conversion storage register.

Setting example:

1. Convert the analog input voltage on the AN3 pin and write the result to memory address 2800H using the AD interrupt(INTAD) processing routine.

Main routine

		7	6	5	4	3	2	1	0	
INTEAD	←	1	1	0	0	–	–	–	–	Enable INTAD and set it to interrupt level 4.
ADMOD1	←	1	1	0	0	0	0	1	1	Set pin AN3 to be the analog input channel.
ADMOD0	←	X	X	0	0	0	0	0	1	Start conversion in channel-fix single conversion mode.

Interrupt routine processing example

WA	←	ADREG3	Read value of ADREG3L and ADREG3H into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2800H)	←	WA	Write contents of WA to memory address 2800H.

2. This example repeatedly converts the analog input voltages on the three pins AN0, AN1 and AN2, using channel-scan repeat conversion mode.

INTEAD	←	1	0	0	0	–	–	–	–	Disable INTAD.
ADMOD1	←	1	1	0	0	0	0	1	0	Set pins AN0 to AN2 to be the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in channel-scan repeat conversion mode.

3. Convert the analog input voltage on the AN2 pin as a Top-priority AD conversion, and write the result to memory address 2A00H using the Top-priority AD interrupt (INTADHP) processing routine.

Main routine

INTEAD	←	1	1	0	1	–	–	–	–	Enable INTADHP and set it to interrupt level 6.
ADMOD1	←	1	0	0	0	0	0	0	0	DAC On.
ADMOD3	←	0	0	1	0	0	0	0	0	Set pin AN2 to be the analog input channel.
ADMOD2	←	0	0	0	0	1	0	0	0	Start a Top-priority AD conversion by software.

Interrupt routine processing example

WA	←	ADREGSP	Read value of ADREGSPL and ADREGSPH into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2A00H)	←	WA	Write contents of WA to memory address 2A00H.

4. Convert the analog input voltage on the AN4 pin as a normal AD conversion of a channel-fix single conversion mode. And then if its conversion result is greater or equal than the value of (ADCM0REGL/H), write the result to memory address 2C00H using the AD monitor function interrupt (INTADM) processing routine.

Main routine

INTEAD	←	–	–	–	–	1	0	1	1	Enable INTAD and set it to interrupt level 3.
ADMOD5	←	0	0	0	0	1	0	0	0	Set the analog input channel AN4 for AD monitor function 0.
ADMOD4	←	0	0	1	0	0	0	0	0	Enable the AD monitor function0 and AD monitor function interrupt 0. Set “a conversion result ≥ AD conversion result compare criterion register” for generation condition of monitor function interrupt 0.
ADMOD1	←	1	0	1	0	0	0	0	0	Set pin AN4 to be the analog input channel.
ADMOD0	←	0	0	0	0	1	0	0	0	Start a normal AD conversion by software.

Interrupt routine processing example

WA	←	ADREG4	Read value of ADREG4L and ADREG4H into 16-bits general-purpose register WA.
WA	←	> > 6	Shift contents read into WA six times to right and zero fill upper bits.
(2C00H)	←	WA	Write contents of WA to memory address 2C00H.

X : Don't care, –: No change

3.25 Watchdog Timer (Runaway detection timer)

The TMP92CF29A contains a watchdog timer of runaway detecting.

The watchdog timer (WDT) is used to return the CPU to the normal state when it detects that the CPU has started to malfunction (runaway) due to causes such as noise. When the watchdog timer detects a malfunction, it generates a non-maskable interrupt INTWD to notify the CPU of the malfunction.

Connecting the watchdog timer output to the reset pin internally forces a reset.

(The level of external $\overline{\text{RESET}}$ pin is not changed.)

3.25.1 Configuration

Figure 3.25.1 is a block diagram of the watchdog timer (WDT).

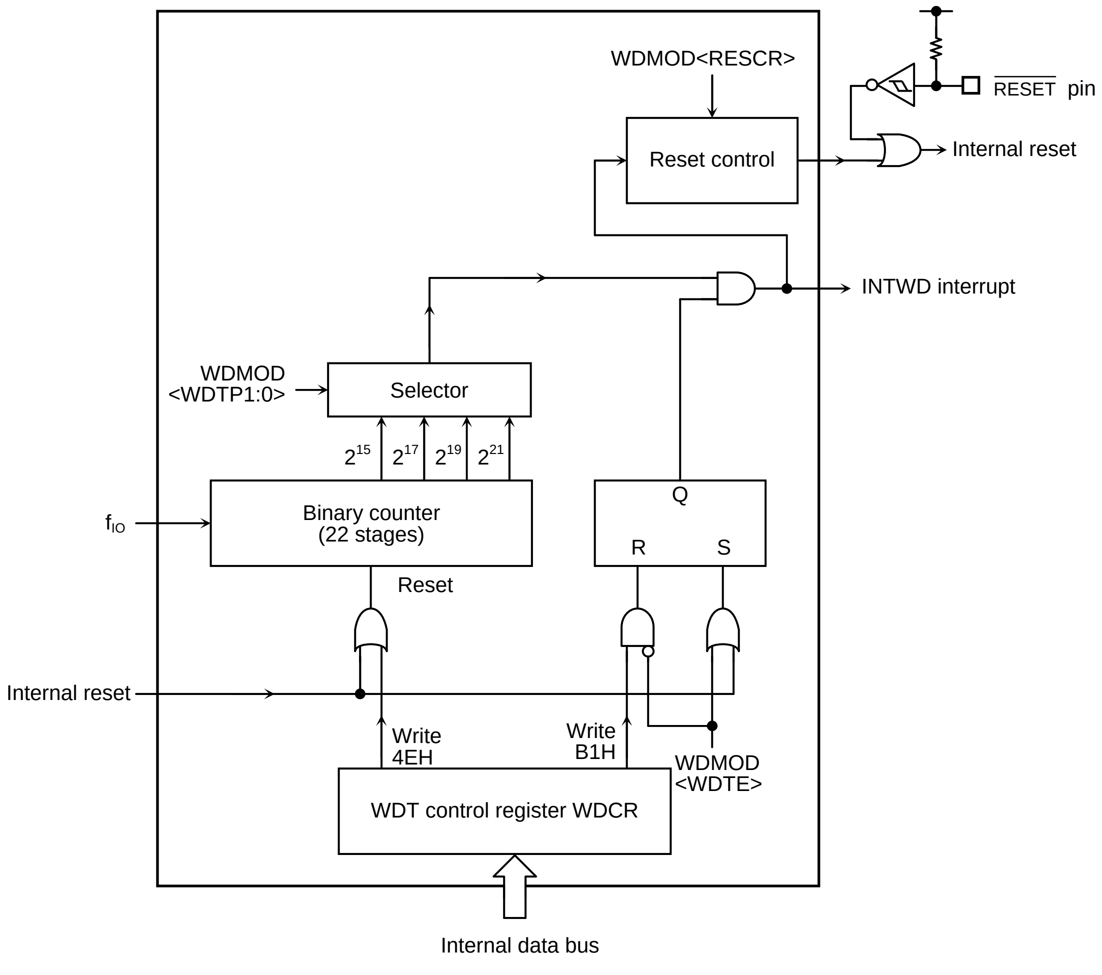


Figure 3.25.1 Block Diagram of Watchdog Timer

Note: Care must be exercised in the overall design of the apparatus since the watchdog timer may fail to function correctly due to external noise, etc.

3.25.2 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1:0> has elapsed. The watchdog timer must be cleared "0" in software before an INTWD interrupt will be generated. If the CPU malfunctions (e.g., if runaway occurs) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (runaway) due to the INTWD interrupt and in this case it is possible to return to the CPU to normal operation by means of an anti-malfunction program.

The watchdog timer begins operating immediately on release of the watchdog timer reset.

The watchdog timer is halted in IDLE1 or STOP mode. The watchdog timer counter continues counting during bus release (when $\overline{\text{BUSAK}}$ goes low).

When the device is in IDLE2 mode, the operation of WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 mode.

The watchdog timer consists of a 22-stage binary counter which uses the clock (f_{IO}) as the input clock. The binary counter can output $2^{15}/f_{IO}$, $2^{17}/f_{IO}$, $2^{19}/f_{IO}$ and $2^{21}/f_{IO}$.

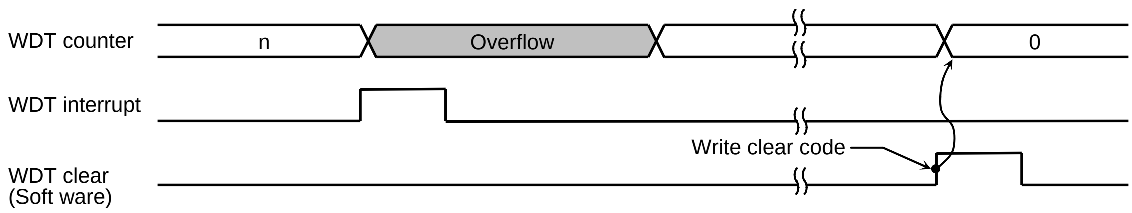


Figure 3.25.2 Normal Mode

The runaway detection result can also be connected to the reset pin internally.

In this case, the reset time will be 32 clocks (102.4 μs at f_{OSCH} = 10 MHz) as shown in Figure 3.25.3. After a reset, the clock f_{IO} is divided f_{SYS} by two, where f_{SYS} is generated by dividing the high-speed oscillator clock (f_{OSCH}) by sixteen through the clock gear function.

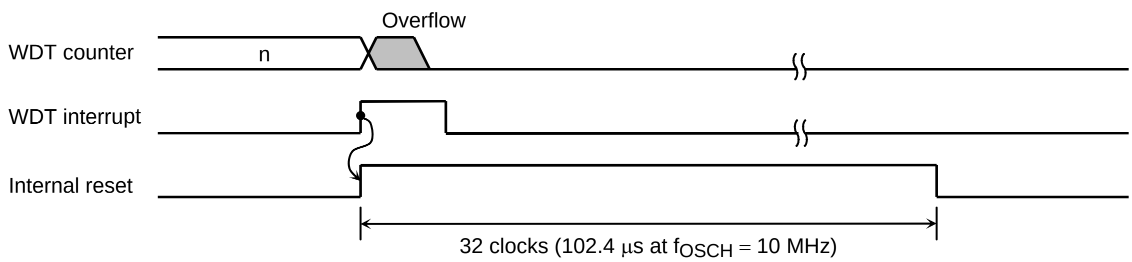


Figure 3.25.3 Reset Mode

3.25.3 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

(1) Watchdog timer mode registers (WDMOD)

1. Setting the detection time for the watchdog timer in <WDTP1:0>

   This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway.

   On a reset this register is initialized to WDMOD<WDTP1:0> = “00”.

   The detection time for WDT is $2^{15}/f_{IO}$ [s]. (The number of system clocks is approximately 65,536.)

2. Watchdog timer enable/disable control register <WDTE>

   At reset, the WDMOD<WDTE> is initialized to “1”, enabling the watchdog timer.

   To disable the watchdog timer, it is necessary to clear this bit to “0” and to write the disable code (B1H) to the watchdog timer control register (WDCR). This makes it difficult for the watchdog timer to be disabled by runaway.

   However, it is possible to return the watchdog timer from the disabled state to the enabled state merely by setting <WDTE> to “1”.

3. Watchdog timer out reset connection <RESCR>

   This register is used to connect the output of the watchdog timer with the RESET terminal internally. Since WDMOD<RESCR> is initialized to “0” at reset, a reset by the watchdog timer will not be performed.

(2) Watchdog timer control register (WDCR)

This register is used to disable and clear the binary counter for the watchdog timer.

- Disable control

  The watchdog timer can be disabled by clearing WDMOD<WDTE> to “0” and then writing the disable code (B1H) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).
WDMOD	←	0	−	−	X	X	−	−	0	Clear WDMOD <WDTE> to “0”.
WDCR	←	1	0	1	1	0	0	0	1	Write the disable code (B1H).

- Enable control

  Set WDMOD<WDTE> to “1”.

- Watchdog timer clear control

  To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).

Note1: If the disable control is used, set the disable code (B1H) to WDCR after write the clear code (4EH) once.
(Please refer to setting example.)

Note2: If the watchdog timer setting is changed, change setting after setting to disable condition once.

WDMOD (1300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
Read/Write	R/W					R/W		
Reset State	1	0	0			0	0	0
Function	WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$				IDLE2 0: Stop 1: Operate	1: Internally connects WDT out to the reset pin	Always write "0"

Watchdog timer out control

0	–
1	Connects WDT out to a reset

IDLE2 control

0	Stop
1	Operation

Watchdog timer detection time

00	$2^{15}/f_{IO}$ (Approximately 819.2 μs at f_{IO} = 40 MHz)
01	$2^{17}/f_{IO}$ (Approximately 3.276 ms at f_{IO} = 40 MHz)
10	$2^{19}/f_{IO}$ (Approximately 13.107 ms at f_{IO} = 40 MHz)
11	$2^{21}/f_{IO}$ (Approximately 52.428 ms at f_{IO} = 40 MHz)

Watchdog timer enable/disable control

0	Disabled
1	Enabled

Figure 3.25.4 Watchdog Timer Mode Register

WDCR (1301H)

A read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
Reset State	–							
Function	B1H: WDT disable code 4EH: WDT clear code							

WDT disable/clear control

B1H	Disable code
4EH	Clear code
Others	Don't care

Figure 3.25.5 Watchdog Timer Control Register

3.26 Power Management Circuitry (PMC)

The TMP92CF29A incorporates the power management circuitry (PMC) for managing standby current to minimize the leakage current in deep to sub-quarter-micron technology. The TMP92CF29A is provided with the following six power supply rails.

・Analog power supply	:	AVCC & AVSS (for AD converter)
・3−V-A power supply for digital I/Os	:	DVCC3A & DVSSCOM (for general pins)
・1.5-V-A internal power supply for the digital logic	:	DVCC1A & DVSSCOM (for general circuits)
・1.5-V-B internal power supply for the digital logic	:	DVCC1B & DVSSCOM (for RTC and PMC)
・1.5-V-C power supply for oscillator	:	DVCC1C & DVSS1C (for high-frequency oscillator and PLL)

Each power supply rail is independent of one another (VSS is partially shared).

Among the six power supply rails, those that are supplied in Power Cut mode are the ones for external pins (DVCC-3A), AD converter (AVCC) and RTC and backup RAM (DVCC-1B). After entering this mode, internal signals that communicate with the circuit blocks powered by DVCC1A and DVCC1C are cut off so that no shoot-through current is generated in the circuitry when the power is removed from those blocks.

- DVCC-3A

  This 3-V power supply rail provides power for external pins preventing them from entering a floating state, for turning on/off the external power supplies, and for signaling the wake-up interrupt for exiting the standby state.

- AVCC

  This 3-V power supply rail provides power for the touch panel interface, and for signaling the Wake-up interrupt for exiting the standby state.

- DVCC-1B

  This 1.5-V power supply rail provides power to the RTC, 16 Kbytes of RAM and the PMC.

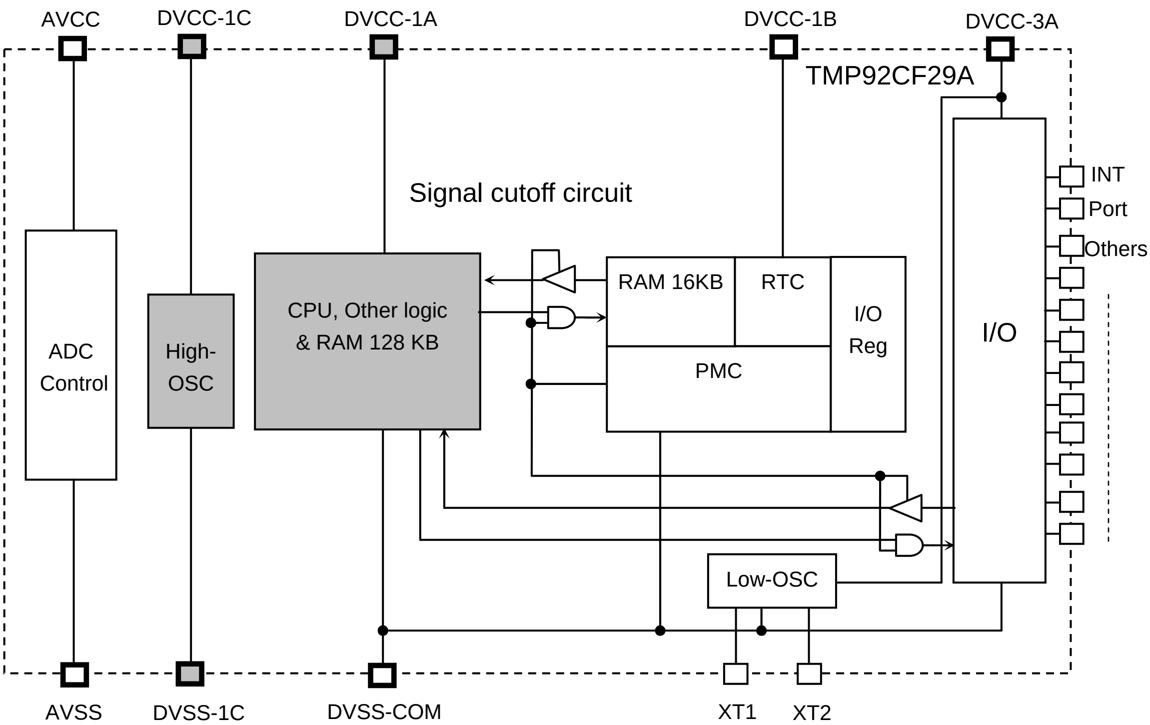


Figure 3.26.1 Power Supply System

3.26.1 Special Function Register (SFR)

		7	6	5	4	3	2	1	0
PMCCTL (02F0H)	Bit Symbol	PCM_ON					−	WUTM1	WUTM0
	Read/Write	R/W					W	R/W	
	System Reset State	0					0	0	0
	Hot Reset State	Data retained					−	Data retained	Data retained
	Function	Power Cut Mode 0: Disable 1: Enable					Must be written as "0" Always read as 0	Warm-up Time 00: 2^9 (15.625 ms) 01: 2^{10} (31.25 ms) 10: 2^{11} (62.5 ms) 11: 2^{12} (125 ms)	

Note1: About 77 μs after a wake-up interrupt has been requested, the external PWE terminal changes from low to high. At this point, the warm-up counter starts counting up the time period specified by the WUTM1 and WUTM0 bits. Then, about 92 μs later, the internal reset signal is negated. The time required for the power supply voltage to stabilize varies depending on the power supply response and the board conditions. This characteristic should be considered in specifying the warm-up time.

Note 2: This register should usually be set in the initial status (all bits are "0"). Writing should be made immediately before the power-cut mode is assumed. Reset the values of all registers to the initial status (all bits are "0") immediately after the power-cut mode. For details, refer to the flow of transition to the power cut status described later.

The operations depending on the setting of the PCM_ON bit are shown below.

	PCM_ON = 1	PCM_ON = 0
External interrupt input	No interrupt HOT_RESET signal asserted	Interrupt
Operation after reset	−	Startup depending on the settings of the AM1 and AM0 pins
Operation after hot reset	Startup from the boot-ROM regardless of the settings of the AM1 and AM0 pins and a program flow jumps to the specified address in the on-chip RAM area.	−
Warm-up counter	A change in the PWE pin level is used as a trigger to start counting the low-frequency clock. Then HOT_RESET signal negated.	−

3.26.2 Detailed Description of Mode Transitions

This section explains the procedures for entering and exiting the Power Cut mode.

- Entering the Power Cut Mode

When entering the Power Cut mode, the CPU needs to be executing in the on-chip RAM.

The low-frequency clock (XT) must be enabled.

It is also necessary to disable interrupts, and to stop DMA operations, WDT and AD converter. Then, configure the output pins to function as ports through the Pn, PnCR and PnDR registers. At this time, the PM7 pin should be configured as the PWE input pin. Also, the internal RTC pin and the external interrupt pins that are used for waking up from the Power Cut mode should be configured as interrupt inputs and enabled.

The interrupt inputs should be configured as rising-edge triggered, if configurable. When the INT4 pin is used as the TSI input, the debounce circuit should be disabled.

The wake-up program must be prewritten to the on-chip RAM area at addresses from 46000H to 49FFFH.

(Including the initial setting of the WDT and other registers, all the required settings for waking up should be predefined in this wake-up program.)

Finally, stop the PLL if it is operated, and specify the warm-up time for waking up from the Power Cut mode (the time period required for the power supply voltage and the high-frequency clock to stabilize) by the PMCCTL<WUTM1:0> bits. Power Cut mode is then entered by writing a 1 to the PMCCTL<PCM_ON> bit.

At this time, the RESET (HOT_RESET) signal is asserted to all the circuits excluding the external I/O and PMC.

Note: As soon as the PMCCTL<PCM_ON> bit is set to “1”, the power management signal (PWE) changes from “1” to “0” and external power supplies are turned off.

1. Configurations Required for Entering the Power Cut Mode

(1) Writing the boot program that is executed after the warm-up time has elapsed (46000Hto 49FFFH)

Only bit 7 of the PMCCTL register is checked whether it is "1" or "0" in the boot-ROM program. All codes required for initializing registers including WDT must be written in the fixed RAM area (46000Hto 49FFFH).

(2) Controlling the low-frequency clock (XT)

Entering or exiting the Power Cut mode is performed using the low-frequency clock. Thus, the low-frequency clock (XT) must always be enabled.

2. Mode Transition Sequence

(1) Program execution jumps to the on-chip RAM area.

Before entering the Power Cut mode, all the sources that might disturb the mode transition must be disabled.

a. Disable the Watch Dog Timer

b. Disable the A/D converter

c. Disable all the DMA functions of the system

- Disable the LCDC
- Disable the auto-refresh function of SDRAM (switching to the self refresh mode)
- Disable the HDMA function

(2) Configure the required port settings (through the Pn, PnCR, PnFC and PnDR registers)

All the external interrupt inputs usable for wake-up signaling must be configured as rising-edge triggered.

When the INT4 pin is used as the TSI input, the debounce circuit should be disabled.

(3) Disable interrupts (DI)

(4) Stop the PLL operation

Program the high-frequency clock frequency f_{SYS} to be f_{OSCH} and stop the PLL operation.

(5) Setup the warm-up time: PMCCTL<WUTM1:0>

About 77 μs after a wake-up interrupt has been requested, the external PWE terminal changes from low to high. At this point, the warm-up counter starts counting up the time period specified by the WUTM1 and WUTM0 bits. Then, about 92 μs later, the internal reset signal is negated. The time required for the power supply voltage to stabilize varies depending on the power supply response and the board conditions. This characteristic should be considered in specifying the warm-up time.

(Warm-up time can be selected from 15.625 ms, 31.25 ms, 62.5 ms and 125 ms.)

(6) Transition to the Power Cut mode (PMCCTL<PCM_ON> = “1”)

* You can set both the warm-up time specificatotion bits, PMCCTL<WUTM1:0>, and the Power Cut mode enable bit, PMCCTL<PCM_ON>, simultaneously.

(7) Insert a dummy instruction for waiting for the mode transition time to PCM (recommended to use 20 NOP instructions)

* Any writing access to the PMCCTL register, including the warm-up time configuration, is only allowed upon entering the PCM and immediately after exiting the PCM. The warm-up time must not be preprogrammed. (The PMCCTL register must be written as 00H at timings other than the above.)

- Exiting the Power Cut Mode

The Power Cut mode can be exited by the assertion of external interrupt or the internal reset. (It is prohibited to exit the reset state when DVCC1A is off. A reset signal must be asserted after supplying power to DVCC1A and waiting for its voltage to fully stabilize.) The interrupts that can be used to exit the Power Cut mode are the RTC interrupt, INT0 to INT7 (TSI interrupts) and INTKEY interrupts.

Table 3.26.1 Interrupts Used for Waking Up from the PCM

Interrupt Source	Symbol	Remarks
RTC	INTRTC	
External	INT0	Only configurable as rising-edge triggered
	INT1	Only configurable as rising-edge triggered
	INT2	Only configurable as rising-edge triggered
	INT3	Only configurable as rising-edge triggered
	INT4	When used as TSI, the debounce circuit should be disabled. Only configurable as rising-edge triggered
	INT5	Only configurable as rising-edge triggered
	INT6	Only configurable as rising-edge triggered
	INT7	Only configurable as rising-edge triggered
Key	INTKEY	KI0 to KI8 Only configurable as falling-edge triggered

When an interrupt request is accepted, the power management signal (PWE) changes from 0 to 1 allowing for the power to be supplied to each block, from which power has been removed. After the warm-up time specified by the PMCCTL<WUTM1:WUTM0> bits has elapsed, HOT_RESET is automatically negated and the CPU boots from the on-chip boot ROM regardless of the external AM pin state. All external ports retain the state of before entering the Power Cut mode except for the PnDR pin, which is also negated upon negation of HOT_RESET.

* Output pin: Hi-Z state → Set to “1” or “0”
* Input gates of input pins: OFF → ON

The PMCCTL <PCM_ON> bit in the PMC is first checked in the on-chip boot-ROM program. If this bit is set to “1”, a program execution jumps to address 46000H in the on-chip RAM before initializing any registers. The <PCM_ON> bit in the PMC is cleared to “0” by software. At the same time, ensure that the warm-up time is reset to the initial value. (The PMCCTL<WUTM1:0> bits must be written as 00H.)

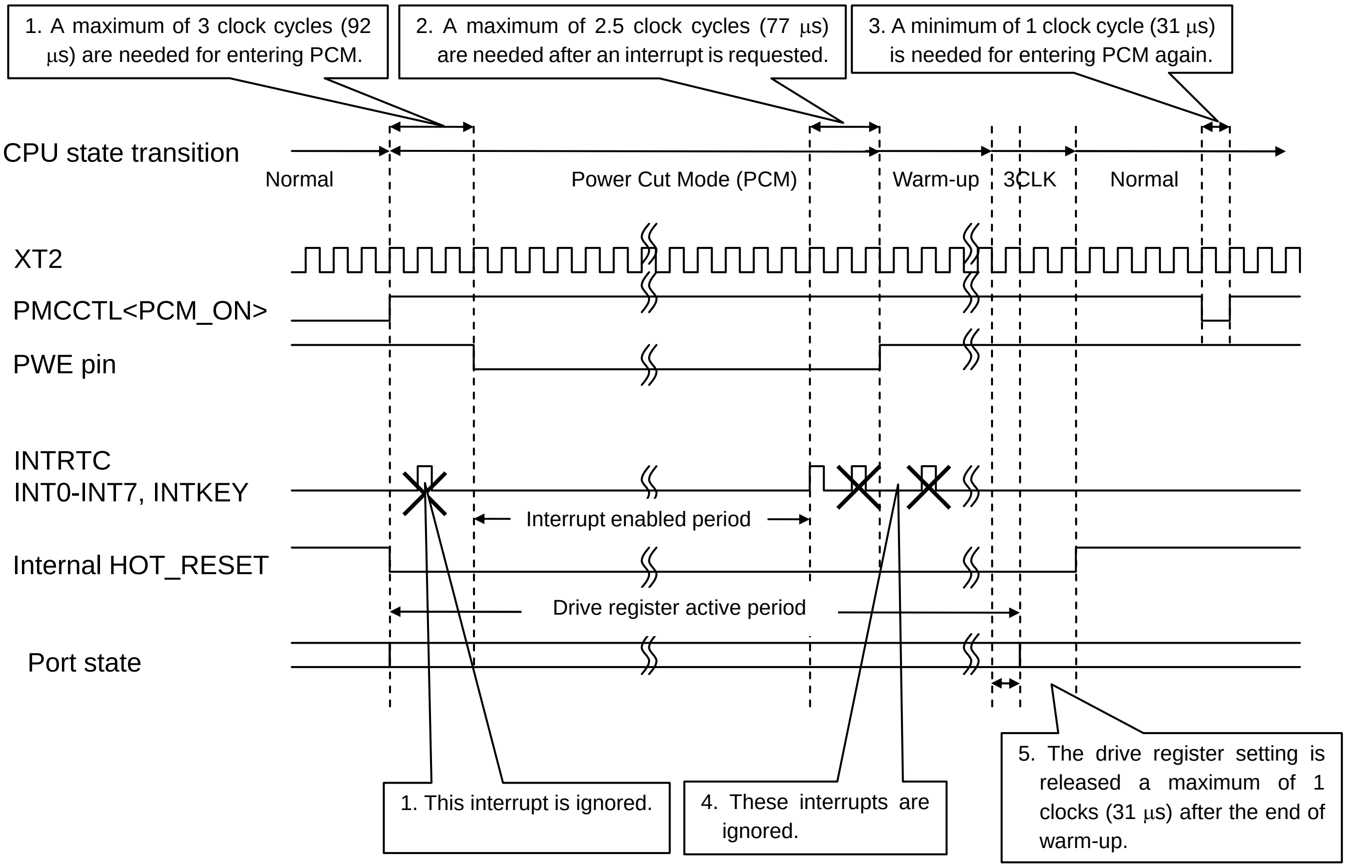
Note 1: The signals that are serviced as interrupt signals in normal mode can be used as Wake-up signals to exit the Power Cut Mode.

Note 2: Once the PMCCTL<PCM_ON> bit is set to “1”, it remains in this state. To re-enter the Power Cut mode, it is necessary to clear this bit to “0” once and then set it to 1 again. At this time, it is required to wait for at least 31 μs after clearing the PCM_ON bit to “0”.

Note 3: Please not that some settings must be configured by software,for the Power Cut mode is exited using the boot ROM.

BROMCR (016CH)		7	6	5	4	3	2	1	0
	Bit Symbol						CSDIS	ROMLESS	VACE
	Read/Write						R/W		
	Reset State						1	0	1
	Function						NAND Flash Area CS Output 0: Enable 1: Disable	Boot-ROM 0: Use 1: Bypass	Vector Address Translation 0: Disable 1: Enable

3.26.3 Detailed Descriptions and Timing Considerations



Internal HOT_RESET assert to dead circuit only. (DVCC1A &DVCC1C circuit)

1. When PMCCTL<PCM_ON> = “1”, mode transition from normal mode to the Power Cut mode takes a maximum of three low-frequency clock cycles (about 92 μs). During this period, the external wake-up requests are ignored.
2. A maximum of 2.5low-frequency clock cycles (about 77 μs) is required for the PWE pin to change from “0” to “1” after the wake-up interrupt is received.
3. After exiting the Power Cut mode, the PMCCTL<PCM_ON> bit is cleared to “0” by soft ware to return to normal mode. To enter the Power Cut mode again, the PMCCTL<PCM_ON> bit should be once cleared to “0” and set to “1” again. In this case, the PMCCTL<PCM _ON> bit should be fixed at “0” for a minimum of one low-frequency clock cycle (about 31 μs). Otherwise, the PCM may not be entered by changing its state from “1” to “0” and to “1” again.
4. The wake-up triggers asserted during the wake-up operation from the PCM are ignored.
5. When a maximum of one low-frequency clock cycle (about 31 μs) has elapsed after the warm-up counter is expired, the DRV setting of every port is switched to the normal setting. Then, two low-frequency clock cycles (about 62 μs) later, the internal reset signal (Hot_Reset) is negated.

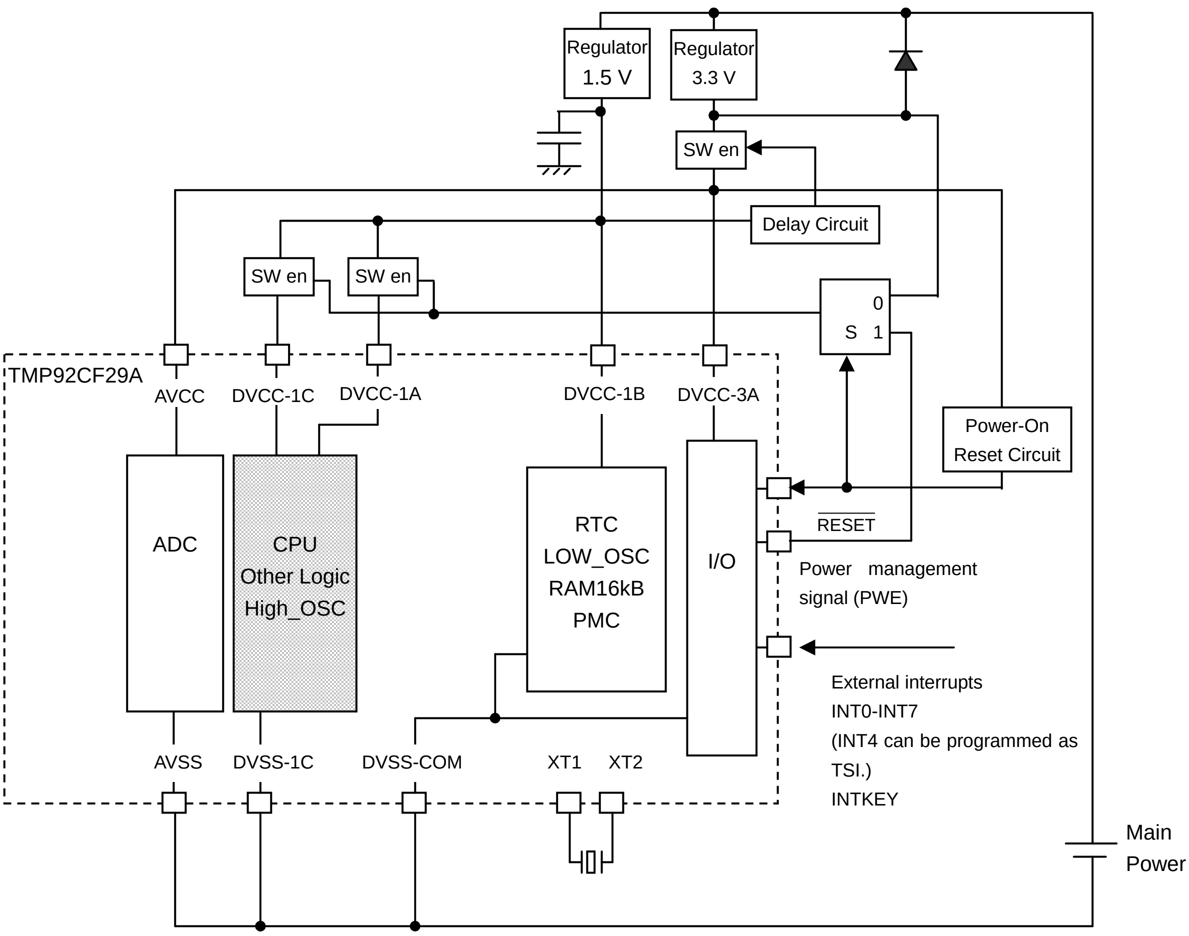


Figure 3.26.2 Application Circuit Examples of the PMC

Figure 3.26.2 shows the examples of the PMC application circuit.

In normal mode, the power management pin (PWE) goes high, which allows the power to be supplied to all the blocks in the TMP92CF29A.

In the Power Cut mode, the PWE pin goes low, which allows the power to be removed from the on-chip circuit blocks excluding the CPU, part of on-chip RAM, AD converter and RTC. This leads to a reduction of the leakage current. In the Power Cut mode, power is supplied only to the followings: I/O (including the AD pins), TSI circuit, 16 Kbytes of on-chip RAM, low-frequency oscillation circuit, RTC and PMC.

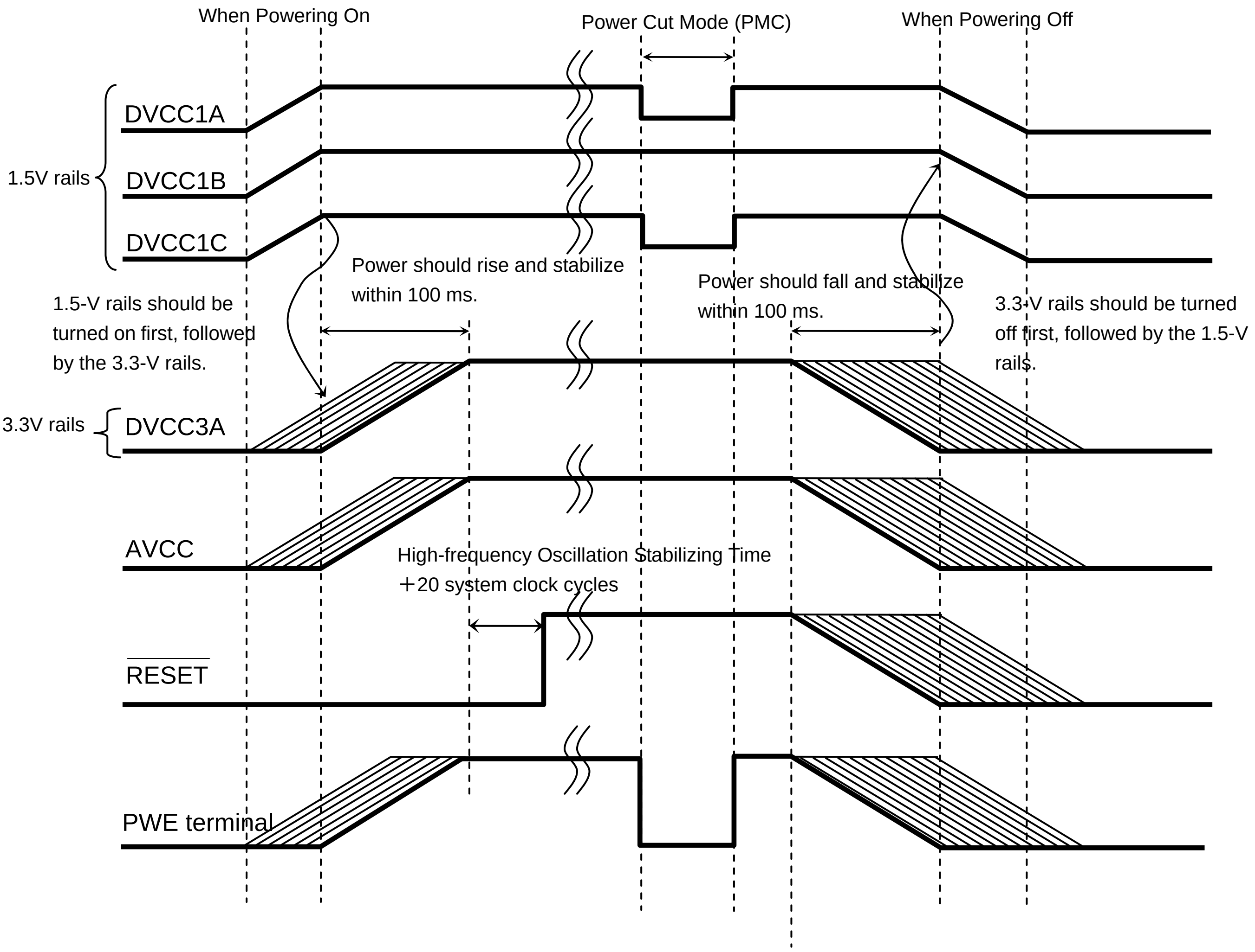
3.26.4 Notes on Power-On/Off Sequences

- Power On/Off Sequences (Initial Power ON/Complete Power OFF)

  As shown below, in the initial power-on sequence, power must be supplied to the on-chip circuit blocks first and then to the external circuit blocks. Also, in the complete power-off sequence, power must be removed from the external circuit blocks and then from the on-chip circuit blocks.

  Power-on
  (DVCC1A, DVCC1B, DVCC1C) → (DVCC3A, AVCC)

  Power-off
  (AVCC, DVCC3A) → (DVCC1C, DVCC1B, DVCC1A)



Note1: Although it is possible to turn on or off the 1.5-V and 3.3-V power supply rails simultaneously, it may cause external pins to temporarily become unstable. Therefore, if there is any possibility that this would affect peripheral devices connected with the TMP92CF29A, external power supplies should be turned on or off while the internal power supplies are stable, as indicated by the heavy lines in the diagram above.

Note2: In the power-on sequence, the 3.3-V power supply rails must not be turned on before the ones of 1.5-V. In the power-off sequence, the 3.3-V power supply rails must not be turned off after the ones of 1.5-V.

3.26.5 Programming Example

Example 1: Mode transition to the PCM

Condition: Wake-up trigger = INT4 (TSI)

```
        org     002000h

        ld      (syscr0),40h        ;  Enable the low-frequency clock
        ldw     (wdmod),0b100h      ;  Disable the WDT
        ldw     (admod0),0000h      ; ⎫ Disable the AD converter
        ldw     (admod2),0000h      ; ⎬
        ldw     (admod4),0000h      ; ⎭
        ld      (lcdctl0),00h       ;  Disable DMA operation
        ld      (pmfc),80h          ;  Program the PM7 port as PWE

        ld      (p9fc),40h          ; ⎫ Enable INT4 and program the interrupt
                                    ; ⎭ level
        ld      (inte34),50h        ;
        ld      (tsicr1),00h        ;  Disable the debounce circuit

        ld      (pllcr0), 00h       ;  Change the CPU clock from PLL to fOSCH
        ld      (pllcr1), 00h       ;  Stop the PLL circuit
        ld      (pmcctl),00h        ;  Program the warm-up time
        di                          ;
        ld      (pmcctl),80h        ;  Enable the PCM_ON bit
                                       (Enters the Power Cut mode)
                                       * Before you program the PMCCTL register
                                       at this point, the PMCCTL register must
                                       remain in the reset state: 00H.
        nop×20                         Wait until PCM is entered

; After Wake-up
          org   046000h

          ld    (pmcctl),00h        ;  Disable the PCM_ON bit
                                       * At the same time, the warm-up time must
                                       be set to default. (The PMCCTL register
                                       must be written as 00H.)
```

Example 2: Mode transition to the PCM
Condition: SDRAM= Self-refresh mode

```
              ld        (syscr0),40h      ;  Enable the low-frequency clock
              ldw       (wdmod),0b100h    ;  Disable the WDT
              ldw       (admod0),0000h    ; ┐ Disable the AD converter
              ldw       (admod2),0000h    ; │
              ldw       (admod4),0000h    ; ┘
              ld        (lcdctl0),00h     ;  Disable the LCDC
              ld        (pmcctl),00h      ;  Program the warm-up time
              ld        (inte0),55h       ;  Enable INT0 and program the interrupt level to 5
              ei        5                 ;
              dl        0,0               ;
              ld        (pccr),00h        ; ┐ Program PC0-PC3 as INT0-INT3
              ld        (pcfc),01h        ; ┘
                                          ;
;((( Entry Self Refresh mode )))          ;
               res       ld               ;  Disable the Self Refresh auto exit function
               ld        (sdcmm),02h      ;  Select the All Bank Precharge command
ABP:                                      ;
               ld        a,(sdcmm)        ;
               cp        a,00h            ;
               jr        nz,ABP           ;  Perform polling until the All Bank Precharge
                                             command is finished

               ld        (sdcmm),05h      ;  Select the Self Refresh Entry command
               nop×10                     ;  Note: Execute at least 10 bytes of NOP or other
                                                   instructions.

               ld        (pj),7fh         ;  Clear the PJ7 bit
               ld        (pjfc),1fh       ;  Configure <PJ7> as Port function
               ld        (pjdr),80h       ;  Configure the PJDR register

;((( Entry PMC mode )))
               di
;--- PLL off setting -----
               ld        (pllcr0),00h     ;  Program the clock signal as: fSYS=fOSCH
               ld        (pllcr1),00h     ;  Stop the PLL circuit

               ld        (pmcctl),80h     ;  Enable PCM  condition
                                             (Start PCM mode)
                         nop×20           ;  Wait until PCM is entered

; After Wake-up
               org       046000h          ;

               ld        (pmcctl),00h     ;  Disable the PCM_ON bit
                                             Note: At the same time, the warm-up time must
                                             be set to default as well. (The PMCCTL register
                                             must be written as 00H)
```

Note: SDRAMC is initialized by hot reset upon a wake-up.

The SDCKE pin output is initialized to “1” by initializing the SDRAMC. Therefore, SDRAM exits from self-refresh mode. Auto-refresh function of the SDRAMC register is disabled at same time. Therefore, SDRAM data might be lost.

However, though the SDRAMC is initialized by hot reset, port configurations are not initialized by Hot reset. Thus, SDRAM can retain its contents.

To keep SDRAM data, program the PJ7 pin as the SDCKE pin and drive it low before entering the PMC mode. The output level of the PJ7 pin while in PMC mode is determined by the PJ and PJDR register settings. Please program the PJ7 pinto be driven low while in PMC mode in the same manner as shown above.

3.27 Multiply and Accumulate Calculation Unit (MAC)

The TMP92CF29A includes a multiply-accumulate unit (MAC) capable of 32-bit × 32-bit + 64-bit arithmetic operations at high speed. The MAC has the following features:

- One-cycle execution for all MAC operations (excluding register access time)
- Three operation modes : 1) 64-bit + 32-bit × 32-bit
2) 64-bit − 32-bit × 32-bit
3) 32-bit × 32-bit − 64-bit
- Support for signed/unsigned operations
- Support for integer operations only

3.27.1 Registers

The MAC in the TMP92CF29A has one control register and three data registers. These registers are connected to the CPU via a 32-bit bus and can be accessed in one system clock (f_{SYS}).

3.27.1.1 Control Register

The control register is used to control the operation of the MAC.

MAC Control Register

		7	6	5	4	3	2	1	0
MACCR (1BFCH)	bit Symbol	MOVF	MOPST	MSTTG2	MSTTG1	MSTTG0	MSGMD	MOPMD1	MOPMD0
	Read/Write	R/W	W	R/W					
	Reset State	0	0	0	0	0	0	0	0
A read-modify-write operation cannot be performed	Function	Overflow flag 0: No overflow 1: Overflow occurred	Calculation soft start 0:Don't care 1:Start calculation	Calculation start trigger 000: Write to MACMA<7:0> 001: Write to MACMB<7:0> 010: Write to MACMOR<7:0> 011: Write to MACMOR<39:32> 1xx: Write of "1" to <MOPST>			Sign mode 0: Unsigned 1: Signed	Calculation mode 00: 64 + 32×32 01: 64 − 32×32 10: 32×32 − 64 11: Reserved	

Note 1: <MOPST> is write-only and it is read as "0".

Note 2: Writing "1xx" to <MSTTG2:0> and writing "1" to <MOPST> can be executed in the same write cycle.

Note 3: <MOVF> is fixed two system clocks (f_{SYS}) after calculation is started.

3.27.1.2 Data Registers

The data registers are arranged as shown below.

	Data Registers							
	Bits<63:56>	Bits<55:48>	Bits<47:40>	Bits<39:32>	Bits<31:24>	Bits<23:16>	Bits<15:8>	Bits<7:0>
Multiplier A Register					(1BE3H)	(1BE2H)	(1BE1H)	MACMA (1BE0H)
Multiplier B Register					(1BE7H)	(1BE6H)	(1BE5H)	MACMB (1BE4H)
MAC Register	(1BEFH)	(1BEEH)	(1BEDH)	MACORH (1BECH)	(1BEBH)	(1BEAH)	(1BE9H)	MACORL (1BE8H)

Note 1: After reset, all the registers are cleared to “0”.

Note 2: Read-modify-write instructions can be used on all the registers.

Note 3: All the registers can be accessed in long word, word, or byte units. (In case of using “sign mode”, it can be accessed in long word only)

Note 4: When MACCR<MSTTG2:0> is set to “0”, “001”, “010” or “011” and the registers are written in word or byte units, the <7:0> bits of each register must be written last.

Note 5: The MACORL register is fixed one system clock (f_{SYS}) after calculation is started, and the MACORH register is fixed two system clocks (f_{SYS}) after calculation is started. Therefore, to read the MACOR register immediately after calculation, be sure to read the MACORL register first.

Note 6: In case of using “sign mode”, MACCR<MSGMD> = “1”, it must need to write to MACMA and MACMB register with longword (32bit).

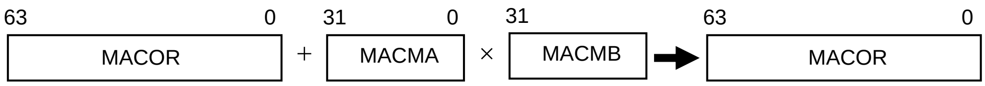
3.27.2 Description of Operation

(1) Calculation mode

The MAC has the following three types of calculation mode. The calculation mode to be used is specified in MACCR<MOPMD1:0>. MACCR<MSGMD> is used to select unsigned or signed mode. The operation of each calculation mode is explained below.

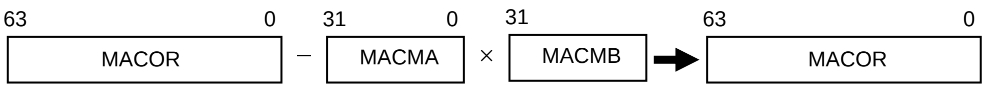
(a) 64 + 32 × 32 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the result is added to the contents of the MACOR register. Then, the result is stored back in the MACOR register.



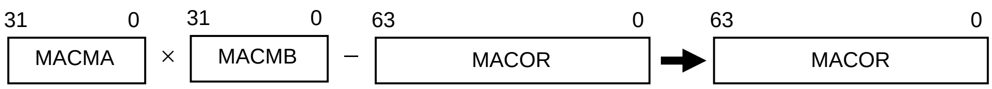
(b) 64 − 32 × 32 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the result is subtracted from the contents of the MACOR register. Then, the result is stored back in the MACOR register.



(c) 32 × 32 − 64 mode

In this mode, the contents of the MACMA register and the MACMB register are multiplied and the contents of the MACOR register are subtracted from the result. Then, the result is stored back in the MACOR register.



(d) Sign mode

Both multiply-accumulate and multiply-subtract operations can be executed in unsigned or signed mode.

In signed mode, the MACMA, MACMB, and MACOR registers become signed registers, and the most significant bit is treated as the sign bit and the data set in each register is treated as a two's complement value. Table 3.27.1 shows the range of values that can be represented in each sign mode.

Table 3.27.1 Data Range in Unsigned/Signed Mode

	MACMA, MACMB Registers	MACOR Register
Unsigned	$0 \sim 2^{32}-1$	$0 \sim 2^{64}-1$
Signed	$-2^{31} \sim +2^{31}-1$	$-2^{63} \sim +2^{63}-1$

Use signed mode when the values to be set in the MACMA and MACMB registers are signed (two's complement) data. Even in unsigned mode it is possible to set signed (two's complement) data in the MACOR register to perform additions and subtractions in signed mode.

In case of using "sign mode", MACCR<MSGMD> = "1", it must need to write to MACMA and MACMB register with longword (32bit).

(2) Calculation start trigger

As a trigger to start calculation, writing to the MACMA, MACMB or MACOR register or soft start (MACCR<MOPST>= "1") can be selected in MACCR<MSTTG2:0>.

(3) Overflow flag

When an overflow occurs in the calculation result (see Table 3.27.2), MACCR<MOVF> is set to "1". Once an overflow occurs, MACCR<MOVF> is held at "1" regardless of subsequent calculation results. Since the overflow flag is not automatically cleared by a read operation, it is necessary to write "0" to clear this flag.

Table 3.27.2 Overflow Definitions

Sign Mode	Calculation Result (MACOR register value)	MACCR<MOVF>
Signed	$MACOR > 2^{64}-1$	1
	$0 \le MACOR \le 2^{64}-1$	0
	$MACOR < 0$	1
Unsigned	$MACOR > 2^{63}-1$	1
	$-2^{63} \le MACOR \le 2^{63}-1$	0
	$MACOR < -2^{63}$	1

3.27.3 Operation Examples

(1) Unsigned multiply-accumulate operation

The following shows a setting example for calculating "33333333 + 11111111 × 22222222":

```
ld    (MACCR), 0x08        ; Unsigned multiply-accumulate mode
                           Start calculation by write to MACMB.
ld    xde, 0x00000000
ld    xhl, 0x33333333
ld    xix, 0x11111111
ld    xiy, 0x22222222
ld    (MACORL), xhl        ; Write 33333333 to MACORL.
ld    (MACORH), xde        ; Clear MACORH.
ld    (MACMA), xix         ; Write 11111111 to MACMA.
ld    (MACMB), xiy         ; Write 22222222 to MACMB.        ← Calculation start
ld    xhl, (MACORL)        ; Read lower result 0x41FDB975.
bit   7, (MACCR)           ; Check over-flow error
jp    nz, ERROR            ; Go to error routine, if there is over-flow error
ld    xde, (MACORH)        ; Read upper result 0x02468ACF.
```

(2) Signed multiply-subtract operation

The following shows a setting example for calculating "33333333 − 11111111 × −22222222":

```
ld    (MACCR), 0x25        ; Signed multiply-subtract mode
                           Start calculation by write of "1" to <MOPST>.
ld    xde, 0x00000000
ld    xhl, 0x33333333
ld    xix, 0x11111111
ld    xiy, 0xDDDDDDDE      ; −22222222
ld    (MACORL), xhl        ; Write 33333333 to MACORL.
ld    (MACORH), xde        ; Clear MACORH.
ld    (MACMA), xix         ; Write 11111111 to MACMA.
ld    (MACMB), xiy         ; Write −22222222 to MACMB.
set   5, (MACCR)           ;                                 ← Calculation start
ld    xhl, (MACORL)        ; Read lower result 0x41FDB975.
bit   7, (MACCR)           ; Check over-flow error
jp    nz, ERROR            ; Go to error routine, if there is over-flow error
ld    xde, (MACORH)        ; Read upper result 0x02468ACF.
```

(3) Unsigned multiply-accumulate operation (two multiply-accumulate operations)

The following shows a setting example for calculating "(33333333 + 11111111 × 22222222) + (11111111 × 44444444)":

```
ld    (MACCR), 0x08        ; Unsigned multiply-accumulate mode
                           Start calculation by write to MACMB.
ld    xde, 0x00000000
ld    xhl, 0x33333333
ld    xix, 0x11111111
ld    xiy, 0x22222222
ld    xiz, 0x44444444
ld    (MACORL), xhl        ; Write 33333333 to MACORL.
ld    (MACORH), xde        ; Clear MACORH.
ld    (MACMA), xix         ; Write 11111111 to MACMA.
ld    (MACMB), xiy         ; Write 22222222 to MACMB.        ← Calculation start
ld    (MACMB), xiz         ; Write 44444444 to MACMB.        ← Calculation start
ld    xhl, (MACORL)        ; Read lower result 0x5F92C5F9.
bit   7, (MACCR)           ; Check over-flow error
jp    nz, ERROR            ; Go to error routine, if there is over-flow error
ld    xde, (MACORH)        ; Read upper result 0x06D3A06D.
```

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Symbol	Contents	Rating	Unit
DVCC3A	Power Supply Voltage	-0.3 to 3.9	V
DVCC1A DVCC1B DVCC1C		-0.3 to 3.0	
AVCC		-0.3 to 3.9	
V_{IN}	Input Voltage	-0.3 to DVCC3A+0.3 (Note1) -0.3 to AVCC + 0.3 (Note2)	V
I_{OL}	Output Current (1pin)	15	mA
I_{OH}	Output Current (1pin)	-15	mA
ΣI_{OL}	Output Current (total)	80	mA
ΣI_{OH}	Output Current (total)	-50	mA
P_D	Power Dissipation (Ta = 85°C)	600	mW
T_{SOLDER}	Soldering Temperature (10s)	260	°C
T_{STG}	Storage Temperature	-65 to 150	°C
T_{OPR}	Operation Temperature	-0 to 70	°C

Note1: If setting it, don’t exceed the Maximum Ratings of DVCC3A.

Note2: In PG0 to PG5, P96,P97,VREFH,VREFL maximum ratings for AVCC is applied.

Note3: The absolute maximum ratings are rated values that must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products that include this device, ensure that no absolute maximum rating value will ever be exceeded.

Solderability

Test parameter	Test condition	Note
Solderability	Use of Sn-37Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5Cu lead-free solder bath Solder bath temperature = 245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	

4.2 DC Electrical Characteristics

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
DVCC3A	General I/O Power Supply Voltage (DVCC = AVCC) (DVSSCOM = AVSS = 0V)	3.0	3.3	3.6	V	X1= 6 to 10MHz CPU CLK (80MHz)	XT1=30 to 34kHz
DVCC1A	Internal Power A	1.4	1.5	1.6	V		
DVCC1B	Internal Power B						
DVCC1C	High CLK oscillator and PLL Power						
V_{IL0}	Input Low Voltage for D0 to D7 P10 to P17 (D8 to 15), P60 to P67 P71 to P76, P90 PC4 to PC7, PF0 to PF5 PG0 to PG5, PJ5 to PJ6 PN0 to PN7, PR0 to PR3, PT0 to PT7, PX5	-0.3	–	0.3×DVCC3A	V	3.0 ≤ DVCC3A ≤ 3.6	
V_{IL1}	Input Low Voltage for PV6 to PV7		–	0.3×DVCC3A		3.0 ≤ DVCC3A ≤ 3.6	
V_{IL2}	Input Low Voltage for P91 to P92, P96 to P97, PA0 to PA7 PC0 to PC3, PP3 to PP5, $\overline{\text{RESET}}$		–	0.25×DVCC3A		3.0 ≤ DVCC3A ≤ 3.6	
V_{IL3}	Input Low Voltage for AM0 to AM1		–	0.1×DVCC3A		3.0 ≤ DVCC3A ≤ 3.6	
V_{IL4}	Input Low Voltage for X1		–	0.1×DVCC1C		1.4 ≤ DVCC1C ≤ 1.6	
V_{IL5}	Input Low Voltage for XT1		–	0.15 ×DVCC3A		3.0 ≤ DVCC3A ≤ 3.6	

Note: Above power supply range is premised that all power supply of same system is equal.
(DVCC1A = DVCC1B = DVCC1C or DVCC3A = AVCC)

Symbol	Parameter	Min	Typ.	Max	Unit	Condition
V_{IH0}	Input High Voltage for D0 to D7 P10 to P17 (D8 to 15), P60 to P67 P71 to P76, P90 PC4 to PC7, PF0 to PF5 PG0 to PG5, PJ5 to PJ6 PN0 to PN7, PR0 to PR3 PT0 to PT7, PU0 to PU7, PX5	0.7 × DVCC3A	–	DVCC3A + 0.3	V	3.0 ≤ DVCC3A ≤ 3.6
V_{IH1}	Input High Voltage for PV6 to PV7	0.7 × DVCC3A	–	DVCC3A + 0.3		3.0 ≤ DVCC3A ≤ 3.6
V_{IH2}	Input High Voltage for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PP3 to PP5, $\overline{\text{RESET}}$	0.75 × DVCC3A	–	DVCC3A + 0.3		3.0 ≤ DVCC3A ≤ 3.6
V_{IH3}	Input High Voltage for AM0 to AM1	0.9 ×DVCC3A	–	DVCC3A + 0.3		3.0 ≤ DVCC3A ≤ 3.6
V_{IH4}	Input High Voltage for X1	0.9 ×DVCC1C	–	DVCC1C + 0.3		1.4 ≤ DVCC1C ≤ 1.6
V_{IH5}	Input High Voltage for XT1	0.85 × DVCC3A	–	DVCC3A + 0.3		3.0 ≤ DVCC3A ≤ 3.6

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
V_{OL1}	Output Low Voltage1 P90 to P92, PC0 to PC3, PC7 PF0 to PF5, PK1 to PK7 PM1 to PM2, PM7 PN0 to PN7, PP3 to PP6 PV6 to PV7, PX5	–	–	0.4	V	I_{OL} = 0.5mA, 3.0 ≤ DVCC3A	
V_{OL2}	Output Low Voltage2 Except V_{OL1} output pin					I_{OL} = 2mA, 3.0 ≤ DVCC3A	
V_{OH1}	Output High Voltage1 P90 to P92, PC0 to PC3, PC7 PF0 to PF7, PK1 to PK7 PM1 to PM2, PM7 PN0 to PN7, PP3 to PP6 PV6 to PV7, PX5	2.4	–	–		I_{OH} = -0.5mA, 3.0 ≤ DVCC3A	
V_{OH2}	Output High Voltage2 Except V_{OL1} output pin					I_{OH} = -2mA, 3.0 ≤ DVCC3A	
I_{Mon}	Internal resistor (ON) MX, MY pins	–	–	30	Ω	V_{OL} = 0.2V	V_{CC} = 3.0 to 3.6 V
I_{Mon}	Internal resistor (ON) PX, PY pins	–	–	30		V_{OH} = V_{CC} –0.2V	
I_{LI}	Input Leakage Current	–	0.02	±5	μA	0.0 ≤ Vin ≤ DVCC3A	
I_{LO}	Output Leakage Current	–	0.05	±10	μA	0.2 ≤ Vin ≤ DVCC3A-0.2V	
R_{RST}	Pull Up/Down Resistor for $\overline{RESET}$, PA0 to PA7, P96	30	50	70	kΩ		
C_{IO}	Pin Capacitance	–	–	10	pF	fc = 1MHz	
V_{TH}	Schmitt Width for P91 to P92, P96 to P97, PA0 to PA7, PC0 to PC3, PP3 to PP5, $\overline{RESET}$	0.6	0.8	1.0	V	3.0 ≤ DVCC3A ≤ 3.6	

Note: Typical values are value that when Ta = 25°C and V_{CC} = 3.3 V unless otherwise noted.

Symbol	Parameter	Min	Typ.	Max	Unit	Condition	
I_{CC}	NORMAL (Note2)	–	15	30	mA	PLL_ON f_{SYS}=80MHz	DVCC3A = 3.6V
			37	52			DVCC1A,1B,1C = 1.6V
	IDLE2	–	0.5	1			DVCC3A = 3.6V
			20	35			DVCC1A,1B,1C = 1.6V
	NORMAL (Note2)	–	12	23		PLL_ON f_{SYS}=60MHz	DVCC3A = 3.6V
			28	39			DVCC1A,1B,1C = 1.6V
	IDLE2	–	0.4	0.8			DVCC3A = 3.6V
			15	26			DVCC1A,1B,1C = 1.6V
	IDLE1	–	12	45	μA	PLL_OFF f_{SYS} =10MHz	DVCC3A = 3.6V
			200	3200			DVCC1A,1B,1C = 1.6V
	Power Cut Mode (With PMC function)	–	6	35	μA	Ta ≤ 70°C	DVCC3A = 3.6V AVCC = 3.6V
				30		Ta ≤ 50°C	
			2	50		Ta ≤ 70°C	DVCC1A = 0V DVCC1B = 1.6V DVCC1C =0V XT = 32kHz X = OFF
				35		Ta ≤ 50°C	
	STOP	–	6	35		Ta ≤ 70°C	DVCC3A = 3.6V AVCC = 3.6V
				30		Ta ≤ 50°C	
			200	800		Ta ≤ 70°C	DVCC1A = 1.6V DVCC1B = 1.6V DVCC1C = 1.6V XT = OFF X = OFF
				600		Ta ≤ 50°C	

Note1: Typical values are value that when Ta = 25°C and V_{CC} = 3.3 V unless otherwise noted.

Note2: I_{CC} measurement conditions (NORMAL, SLOW):

All functions are operational; output pins except bus pin are open, and input pins are fixed. Bus pin C_L= 50pF (Access toexternal memory at 8-wait setting)

Note3: Above I_{CC} measurement value is a data when 16 bit external bus starting.

4.3 AC Characteristics

The Following all AC regulation is the measurement result in following condition, if unless otherwise noted.

AC measuring condition

- Clock of top column in above table shows system clock frequency, and "T" shows system clock period [ns].
- Output level: High = 0.7 × DVCC3A, Low = 0.3 × DVCC3A
- Input level: High = 0.9 × DVCC3A, Low = 0.1 × DVCC3A

Note: In table, "Variable" shows the regulation at DVCC3A=3.0V~3.6V, DVCC1A=DVCC1B=DVCC1C=1.4~1.6V.

4.3.1 Basic Bus Cycle

Read cycle

No.	Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
			Min	Max			
1	OSC period (X1/X2)	t_{OSC}	100	166.6	–	–	ns
2	System clock period (= T)	t_{CYC}	12.5	2666	12.5	16.6	
3	SDCLK low width	t_{CL}	0.5T – 3		3.25	5.3	
4	SDCLK high width	t_{CH}	0.5T – 3		3.25	5.3	
5-1	A0 ~ A23 valid → D0 ~ D15 input at 0 waits	t_{AD}		2.0T – 18.0	7	15.3	
5-2	A0 ~ A23 valid → D0 ~ D15 input at 4 waits/6 waits	t_{AD4}		6.0T – 18.0	57	82	
		t_{AD6}		8.0T – 18.0	82	115	
6-1	$\overline{RD}$ falling → D0 ~ D15 input at 0 waits	t_{RD}		1.5T – 18.0	0.75	7	
6-2	$\overline{RD}$ falling → D0 ~ D15 input at 4 waits/6waits	t_{RD4}		5.5T – 18.0	50.75	73.6	
		t_{RD6}		7.5T – 18.0	75.75	106.5	
7-1	$\overline{RD}$ low width at 0 waits	t_{RR}	1.5T – 10		8.75	14.9	
7-2	$\overline{RD}$ low width at 4 waits/6waits	t_{RR4}	5.5T – 10		58.75	81.3	
		t_{RR6}	7.5T – 10		83.75	114.5	
8	A0 ~ A23 valid → $\overline{RD}$ falling	t_{AR}	0.5T – 5		1.25	3.3	
9	$\overline{RD}$ falling → SDCLK rising	t_{RK}	0.5T – 5		1.25	3.3	
10	A0 ~ A23 valid → D0 ~ D15 hold	t_{HA}	0		0	0	
11	$\overline{RD}$ rising → D0 ~ D15 hold	t_{HR}	0		0	0	
12	$\overline{WAIT}$ setup time	t_{TK}	20		20	20	
13	$\overline{WAIT}$ hold time	t_{KT}	2		2	2	
14-1	Data byte control access time for SRAM at 0 waits	t_{SBA}		1.5T – 18.0	0.75	7	
14-2	Data byte control access time for SRAM at 4 waits/6waits	t_{SBA4}		5.5T – 18.0	50.75	73.6	
		t_{SBA6}		7.5T – 18.0	75.75	107.0	
15	$\overline{RD}$ high width	t_{RRH}	0.5T – 5		1.25	3.3	

AC measuring condition

- Data_bus, Address_bus, various function control signal capacitance C_L = 50 pF

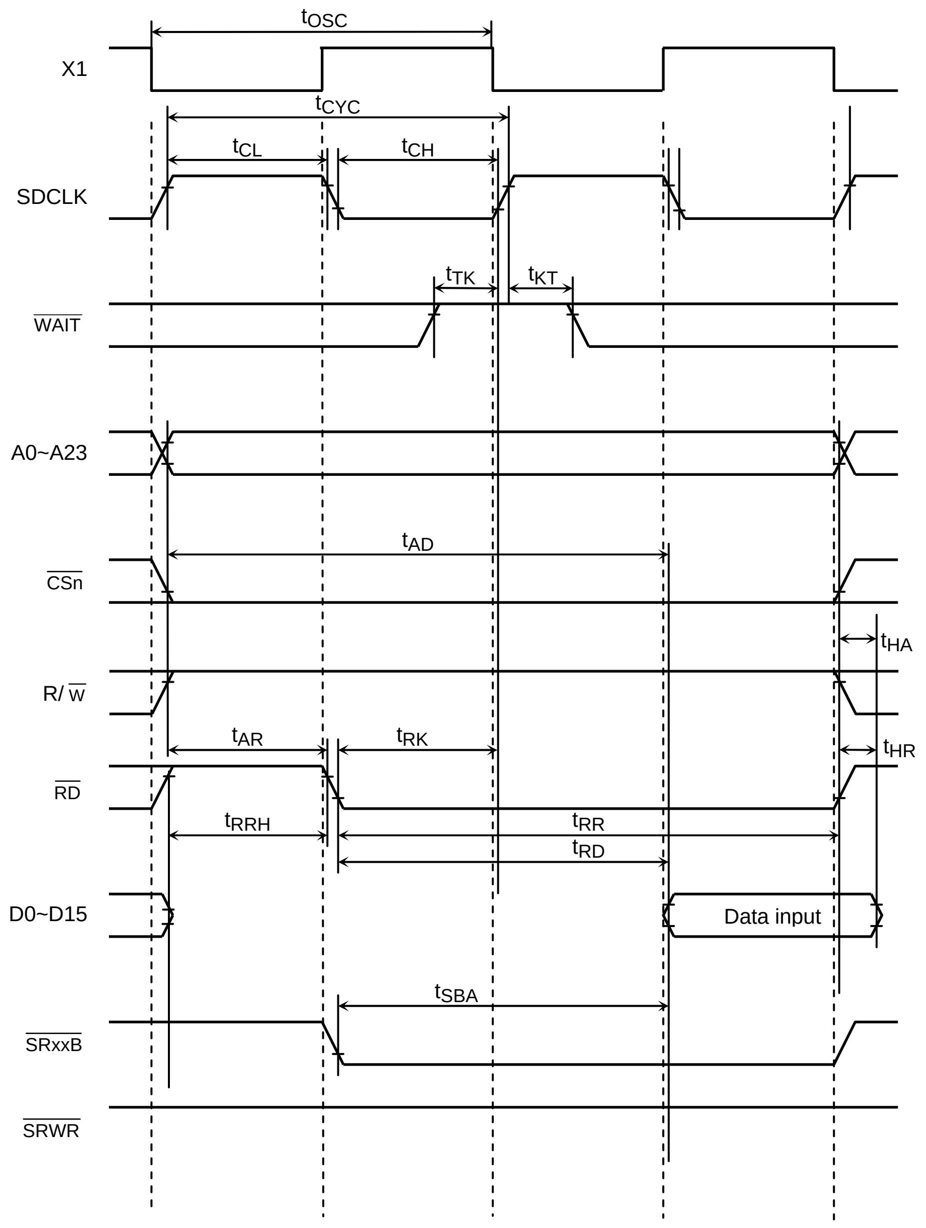
Write cycle

No.	Parameter	Symbol	Variable		80MHz	60MHz	Unit
			Min	Max			
16-1	D0 ~ D15 valid → $\overline{WR}$ xx rising at 0 waits	t_{DW}	1.0T – 6.0		6.5	10.6	ns
16-2	D0 ~ D15 valid → $\overline{WR}$ xx rising at 2 waits/4 waits	t_{DW2}	3.0T – 6.0		31.5	43.8	
		t_{DW4}	5.0T – 6.0		56.5	77.0	
17-1	$\overline{WR}$ xx low width at 0 waits	t_{WW}	1.0T – 4.0		8.5	12.6	
17-2	$\overline{WR}$ xx low width at 2 waits/4 waits	t_{WW2}	3.0T – 4.0		33.5	45.8	
		t_{WW4}	5.0T – 4.0		58.5	79.0	
18	A0 ~ A23 valid → $\overline{WR}$ falling	t_{AW}	0.5T – 5.0		1.25	3.3	
19	$\overline{WR}$ xx falling → SDCLK rising	t_{WK}	0.5T – 5.0		1.25	3.3	
20	$\overline{WR}$ xx rising → A0 ~ A23 hold	t_{WA}	0.5T – 5.0		1.25	3.3	
21	$\overline{WR}$ xx rising → D0 ~ D15 hold	t_{WD}	0.5T – 5.0		1.25	3.3	
22	$\overline{RD}$ rising → D0 ~ D15 output	t_{RDO}	0.5T – 1.0		5.25	7.3	
23-1	Write width for SRAM at 0 waits	t_{SWP}	1.0T – 4.0		8.5	12.6	
23-2	Write width for SRAM at 2 waits/4 waits	t_{SWP2}	3.0T – 4.0		33.5	45.8	
		t_{SWP4}	5.0T – 4.0		58.5	79.0	
24-1	Data byte control ~ end of write for SRAM at 0 waits	t_{SBW}	1.0T – 4.0		8.5	12.6	
24-2	Data byte control ~ end of write for SRAM at 2 waits/4 waits	t_{SBW2}	3.0T – 4.0		33.5	45.8	
		t_{SBW4}	5.0T – 4.0		58.5	79.0	
25	Address setup time for SRAM	t_{SAS}	0.5T – 5.0		1.25	3.3	
26	Write recovery time for SRAM	t_{SWR}	0.5T – 5.0		1.25	3.3	
27-1	Data setup time for SRAM at 0 waits	t_{SDS}	1.0T – 6.0		6.5	10.6	
27-2	Data setup time for SRAM at 2 waits/4 waits	t_{SDS2}	3.0T – 6.0		31.5	43.8	
		t_{SDS4}	5.0T – 6.0		56.5	77.0	
28	Data hold time for SRAM	t_{SDH}	0.5T – 5.0		1.25	3.3	

AC measuring condition

- Data_bus, Address_bus, various function control signal capacitance $C_L = 50$ pF

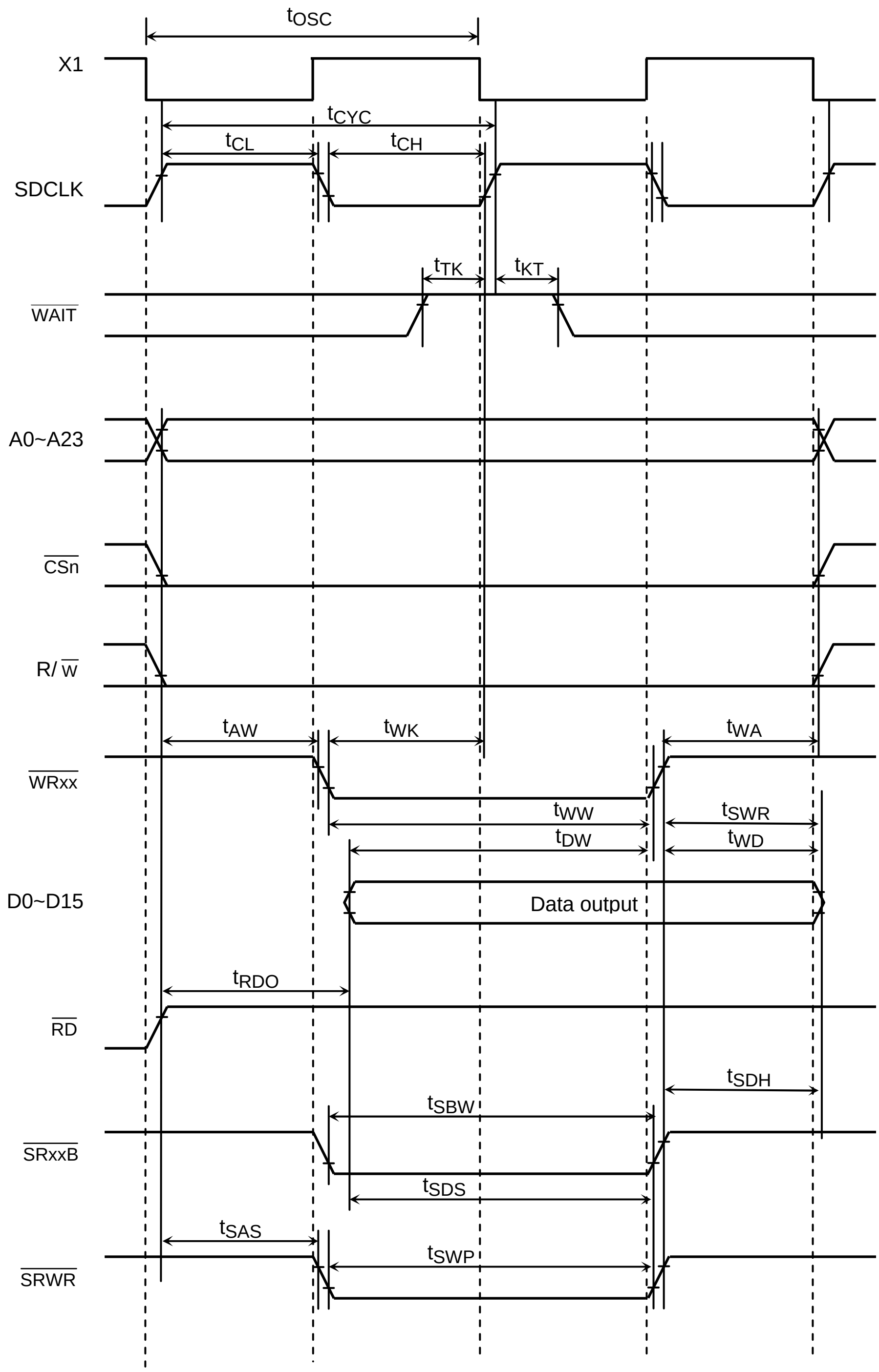
(1) Read cycle (0 waits)



Note1: The phase relation between X1 input signal and the other signals is undefined.

Note2: The above timing chart show an example of basic bus timing. The CSn, R/W, RD, WRxx, SRxxB, SRWR pins timing can be adjusted by memory controller timing adjust function.

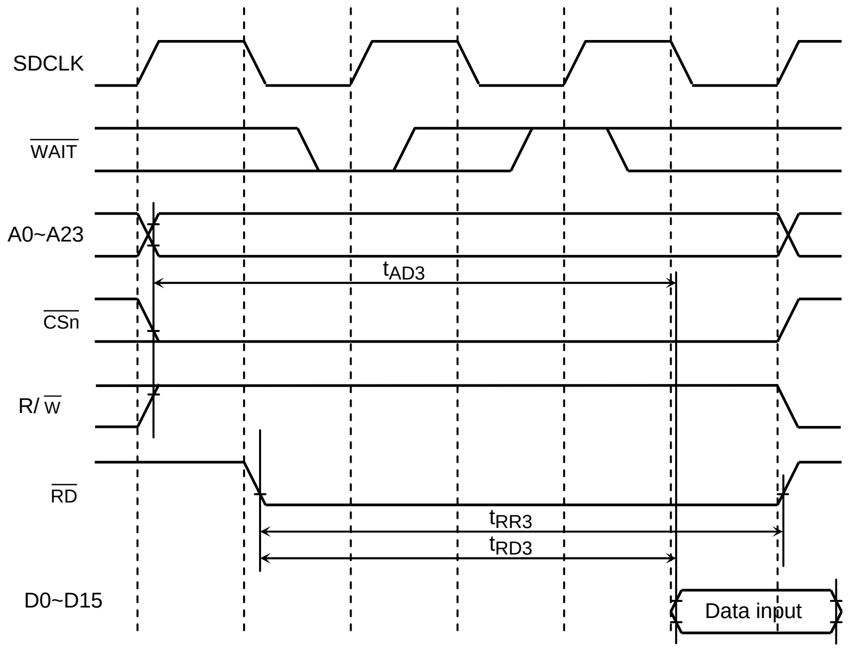
(2) Write cycle (0 waits)



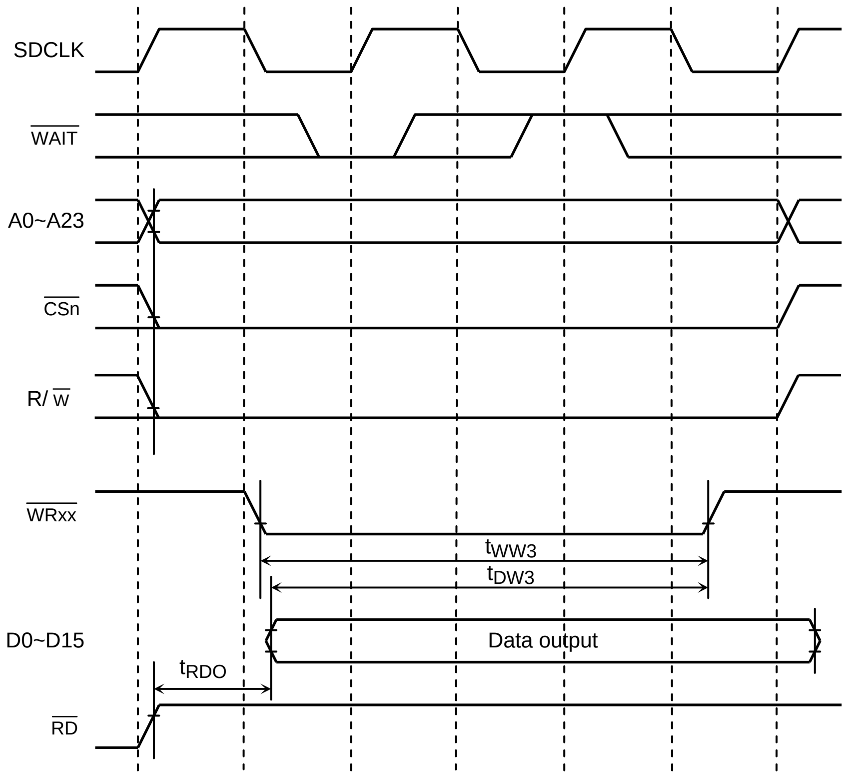
Note1: The phase relation between X1 input signal and the other signals is undefined.

Note2: The above timing chart show an example of basic bus timing. The $\overline{CSn}$, R/$\overline{W}$, $\overline{RD}$, $\overline{WRxx}$, $\overline{SRxxB}$, $\overline{SRWR}$ pins timing can be adjusted by memory controller timing adjust function.

(3) Read cycle (1 wait)



(4) Write cycle (1 wait)



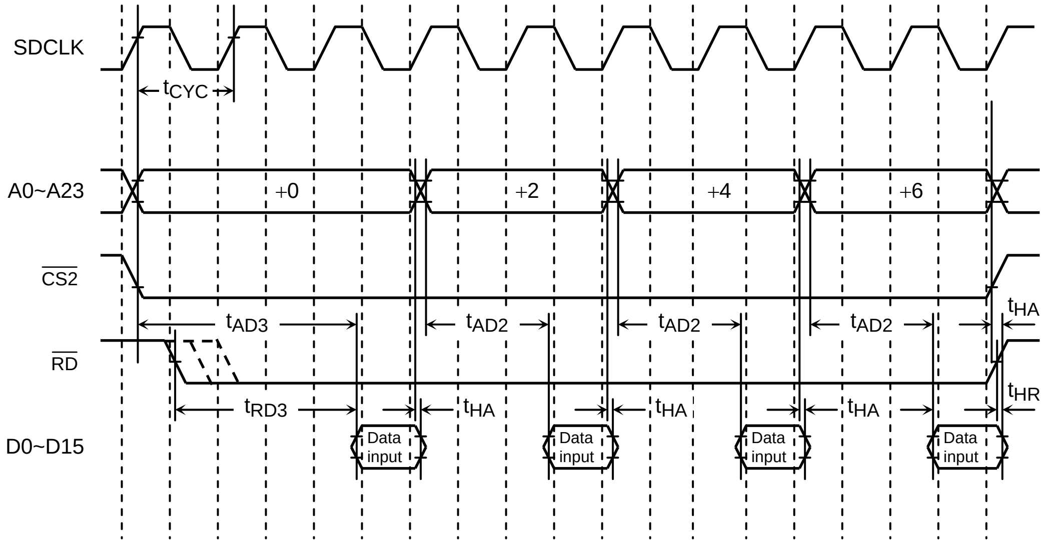
4.3.2 Page ROM Read Cycle

(1) 3-2-2-2 mode

No.	Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
			Min	Max			
1	System clock period (= T)	t_{CYC}	12.5	2666	12.5	16.6	ns
2	A0, A1 → D0 ~ D15 input	t_{AD2}		2.0T − 18	7	15.2	
3	A2 ~ A23 → D0 ~ D15 input	t_{AD3}		3.0T − 18	19.5	31.8	
4	$\overline{RD}$ falling → D0 ~ D15 input	t_{RD3}		2.5T − 18	13	24	
5	A0 ~ A23 Invalid → D0 ~ D15 hold	t_{HA}	0		0	0	
6	$\overline{RD}$ rising → D0 ~ D15 hold	t_{HR}	0		0	0	

AC measuring condition

Note: The (a), (b) and (c) of "Symbol" in above table depend on the falling timing of $\overline{RD}$ pin. The falling timing of $\overline{RD}$ pin is set by MEMCR0<RDTMG1:0> in memory controller. If MEMCR0<RDTMG1:0> is set to "00", it correspond with (a) in above table, and "01" is (b), "10" is (c).



Page Mode Access Timing (when using a 8-byte page size example)

4.3.3 SDRAM controller AC Characteristics

No.	Parameter		Symbol	Variable		80 MHz	60 MHz	Unit
				Min	Max			
1	Ref/Active to ref/active command period	<STRC[2:0]>="000"	t_{RC}	T		12.5	16.6	ns
		<STRC[2:0]> ="110"		7T		87.5	116.2	
2	Active to precharge command period	<STRC[2:0]> ="000"	t_{RAS}	2T(Note1)		25.0	33.2	
		<STRC[2:0]> ="110"		7T		87.5	116.2	
3	Active to read/write command delay time	<STRCD>= "0"	t_{RCD}	T		12.5	16.6	
		<STRCD>= "1"		2T		25.0	33.2	
4	Precharge to active command period	<STRP>= "0"	t_{RP}	T		12.5	16.6	
		<STRP>= "1"		2T		25.0	33.2	
5	Active to active command period	<STRC[2:0]> ="000"	t_{RRD}	3T(Note2)		37.5	49.8	
		<STRC[2:0]> ="110"		7T		87.5	116.2	
6	Write recovery time	<STWR>= "0"	t_{WR}	T		12.5	16.6	
		<STWR>= "1"		2T		25.0	33.2	
7	CLK cycle time		t_{CK}	T		12.5	16.6	
8	CLK high level width		t_{CH}	0.5T − 3		3.25	5.3	
9	CLK low level width		t_{CL}	0.5T − 3		3.25	5.3	
10-1	Access time from CLK(CL* =2) <SRDS>= "0"(Read data shift OFF)		t_{AC}		T – 16	- 3.5	0.6	
10-2	Access time from CLK(CL* =2) <SRDS>= "1"(Read data shift ON)		t_{AC}		T – 6.5	6	10.1	
11	Data hold time from internal read		t_{HR}	0		0	0	
12	Data-in set-up time	1Word/Single	t_{DS}	0.5T – 4		2.25	4.3	
		Burst	t_{DS}	0.5T – 4		2.25	4.3	
13	Data-in hold time	1Word/Single	t_{DH}	T – 10		2.5	6.6	
		Burst	t_{DH}	0.5T – 4		2.25	4.3	
14	Address set-up time		t_{AS}	0.5T – 4		2.25	4.3	
15	Address hold time		t_{AH}	0.5T – 4		2.25	4.3	
16	CKE set-up time		t_{CKS}	0.5T – 3		3.25	5.3	
17	Command set-up time		t_{CMS}	0.5T – 3		3.25	5.3	
18	Command hold time		t_{CMH}	0.5T – 4		2.25	4.3	
19	Mode register set cycle time		t_{RSC}	T		12.5	16.6	

*CL: CAS latency

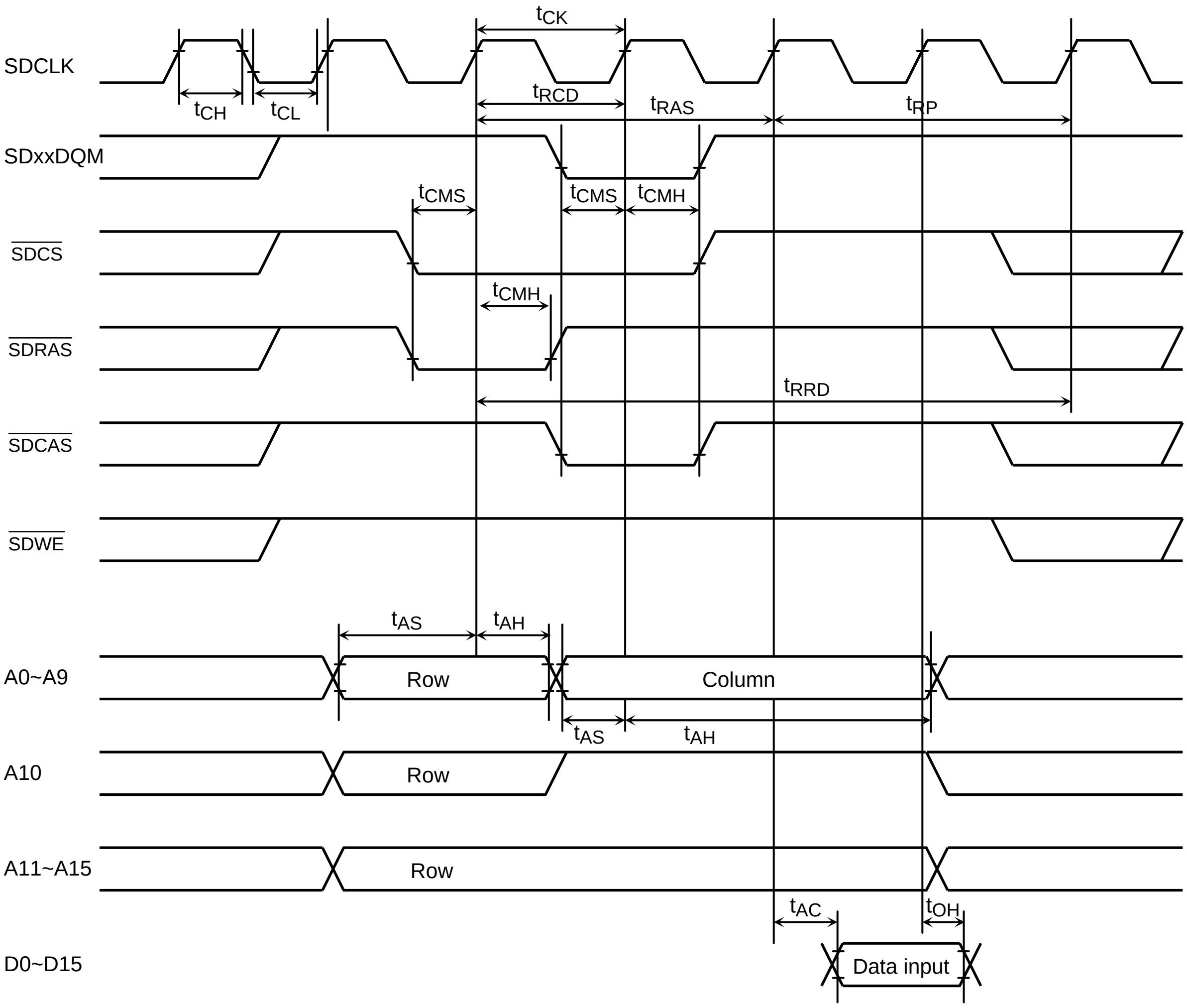
<u>AC measuring condition</u>

SDCLK pin C_L = 30 pF, Other pins C_L = 50 pF

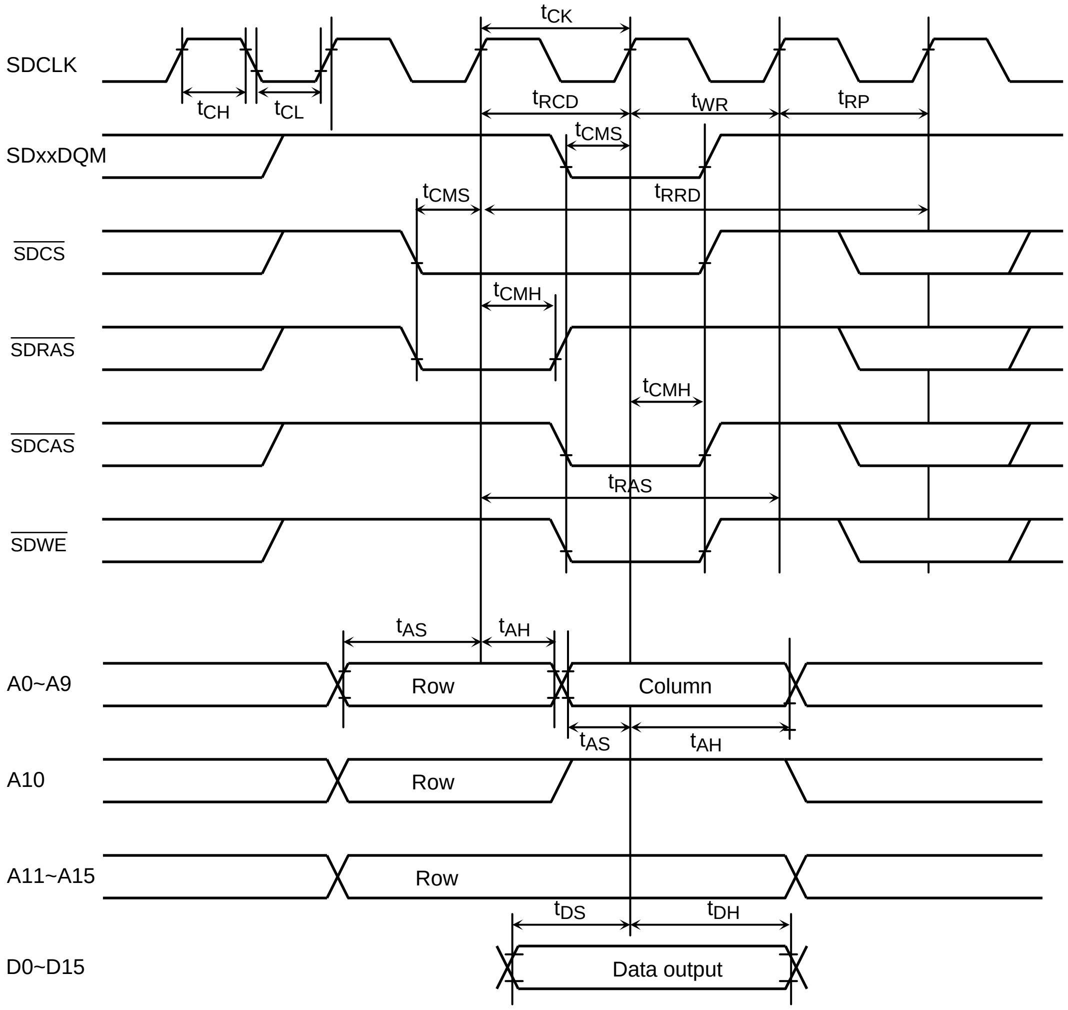
Note1: The Minimum cyclye of "Active to pre-charge command period" is 2T (2 clocks) because the cycle of "READ/WRITE + PRECHARGE" occur by SDCISR<STRC2:0>="000", "001" and "010". If other settigs the above setiing, the clolck is value of "Register setting velue +1". (ex. if "010" setting, the clock is 3clocks.)

Note 2: The Minimum cyclye of "Active to active command period" is 3T (3 clocks) because the clycle of "READ/WRITE + PRECHARGE + ACTIVE" occur by SDCISR<STRC2:0>="000", "001" and "010". If other settigs the above setiing, the clolck is value of "Register setting velue +1". (ex. if "011" setting, the clock is 4 clocks.)

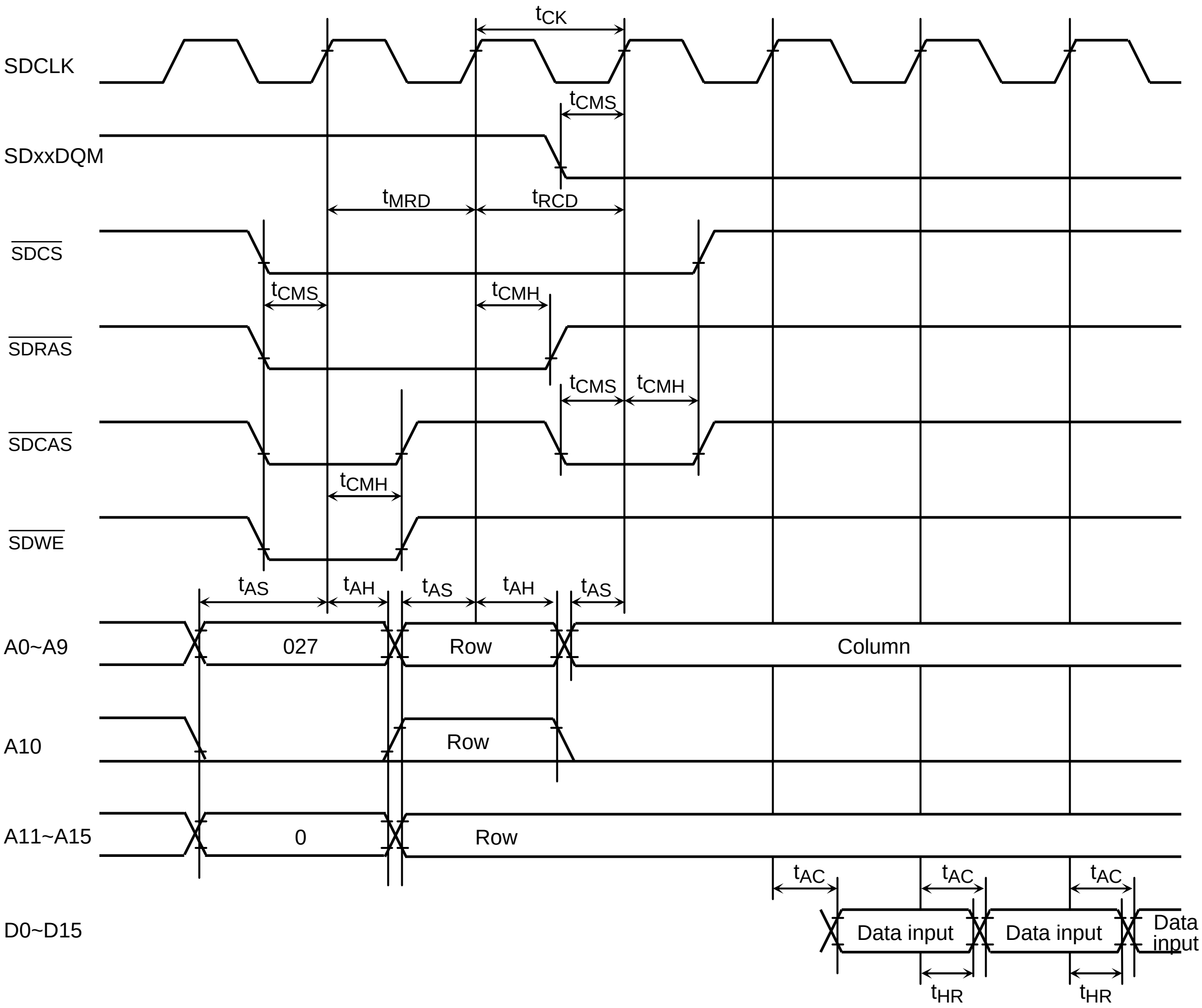
(1) SDRAM read timing (1Word length read mode, <SPRE>= "1")



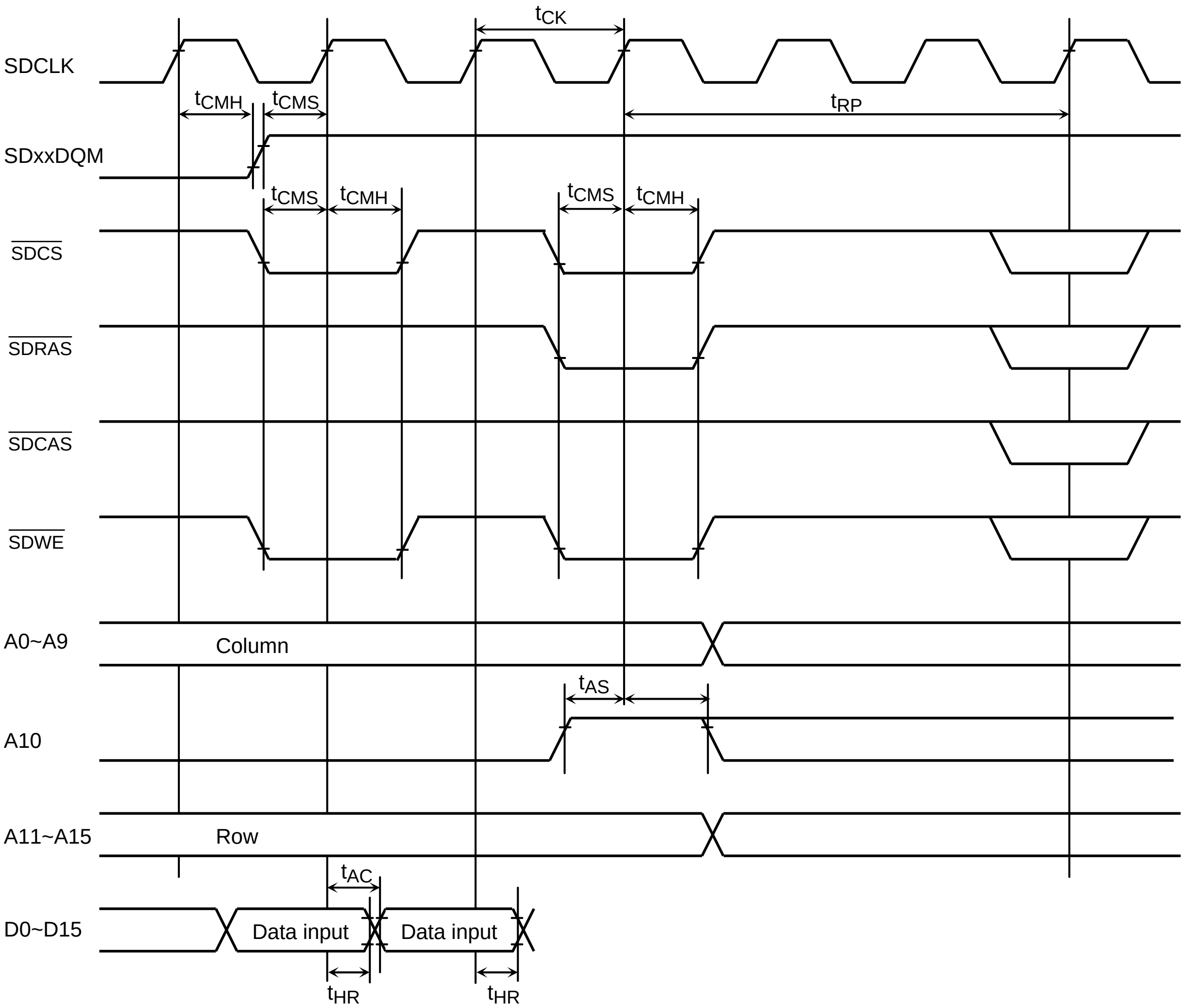
(2) SDRAM write timing (Single write mode, <SPRE>= “1”)



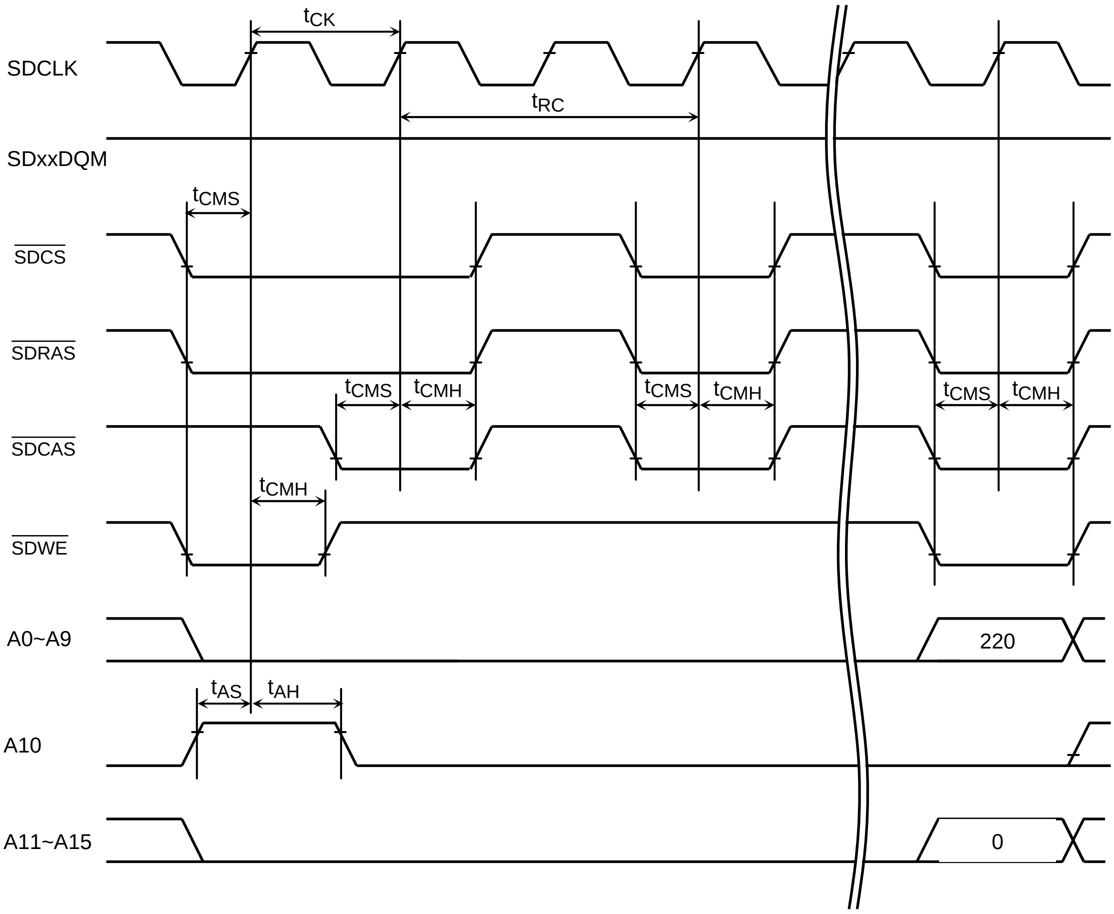
(3) SDRAM burst read timing (Start burst cycle)



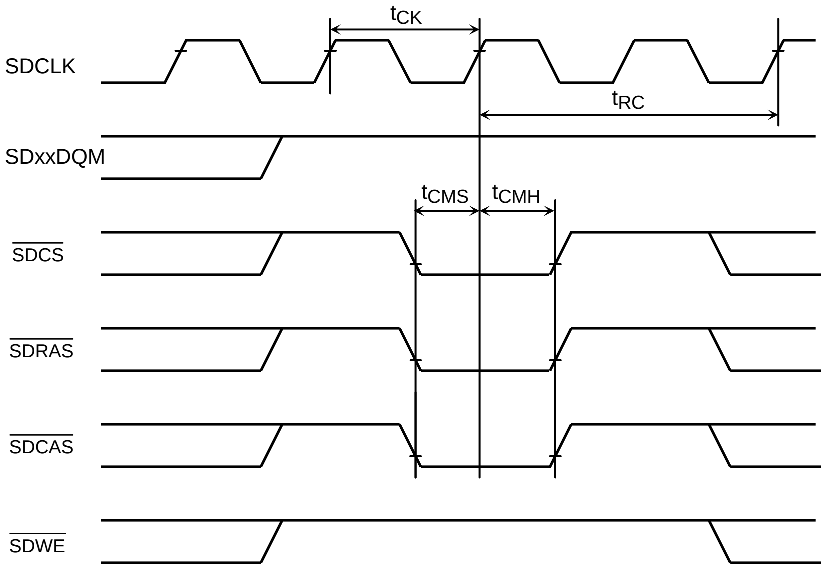
(4) SDRAM burst read timing (End burst timing)



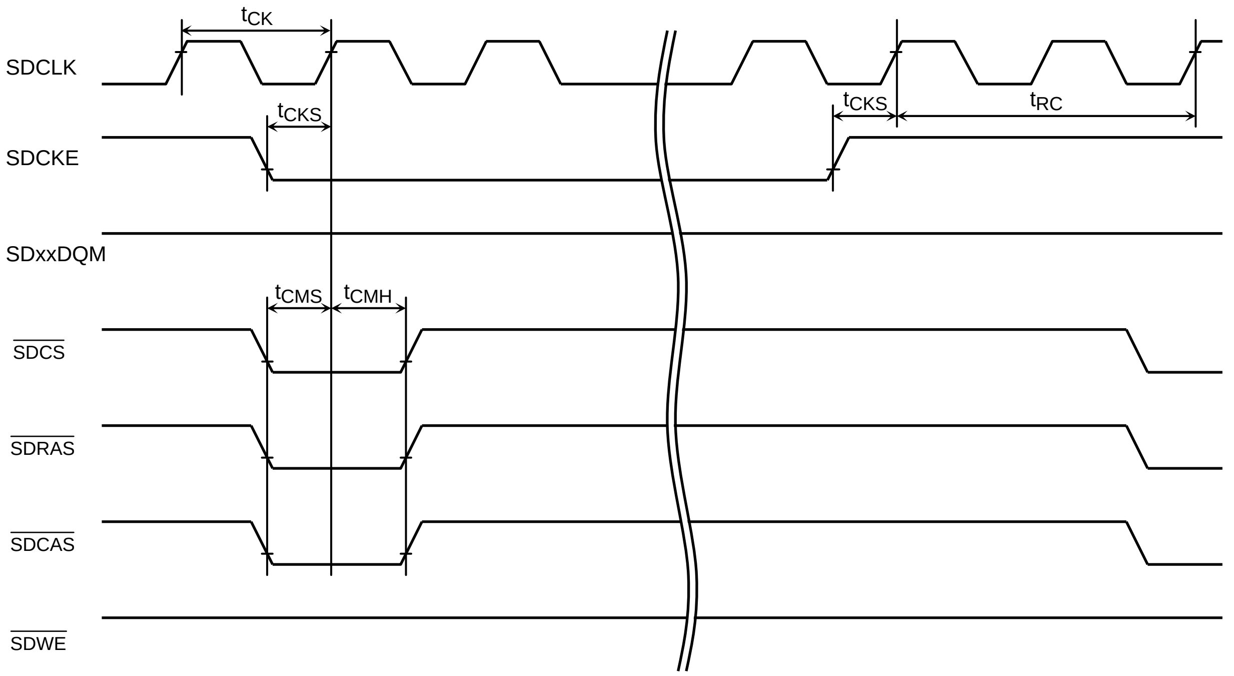
(5) SDRAM initializes timing



(6) SDRAM refreshes timing



(7) SDRAM self refresh timing



4.3.4 NAND Flash Controller AC Characteristics

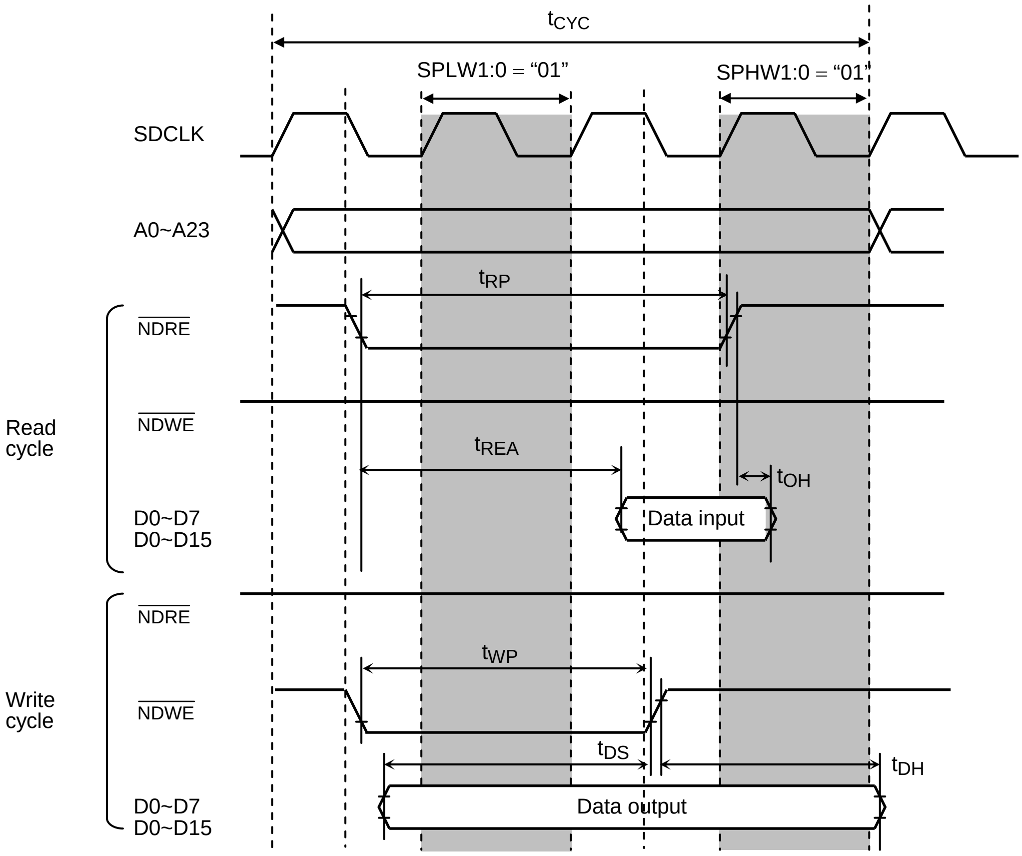
No.	Symbol	Parameter	Variable Min	Variable Max	80 MHz (n=3) (m=3)	60 MHz (n=3) (m=3)	Unit
1	t_{NC}	Access cycle	(2 + n + m) T		100	132	ns
2	t_{RP}	$\overline{\text{NDRE}}$ low level width	(1.5 + n) T − 12		45	63	
3	t_{REA}	$\overline{\text{NDRE}}$ data access time		(1.5 + n) T − 15	41	60	
4	t_{OH}	Read data hold time	0		0	0	
5	t_{WP}	$\overline{\text{NDWE}}$ low level width	(1.0 + n) T − 20		30	47	
6	t_{DS}	Write data setup time	(1.0 + n) T − 20		30	47	
7	t_{DH}	Write data hold time	(0.5 + m) T − 2		42	56	

AC measuring condition

Note1: The "n" in "Variable" means wait-number which is set to NDFMCR0<SPLW1:0>, and "m" means number which is set to NDFMCR0<SPHW1:0>.

Example: If NDFMCR0<SPLW1:0> is set to "01", n = 1, t_{RP} = (1.5 + n) T − 12 = 2.5T − 12

Note2: In above variable, the setting that result is minus can not use.



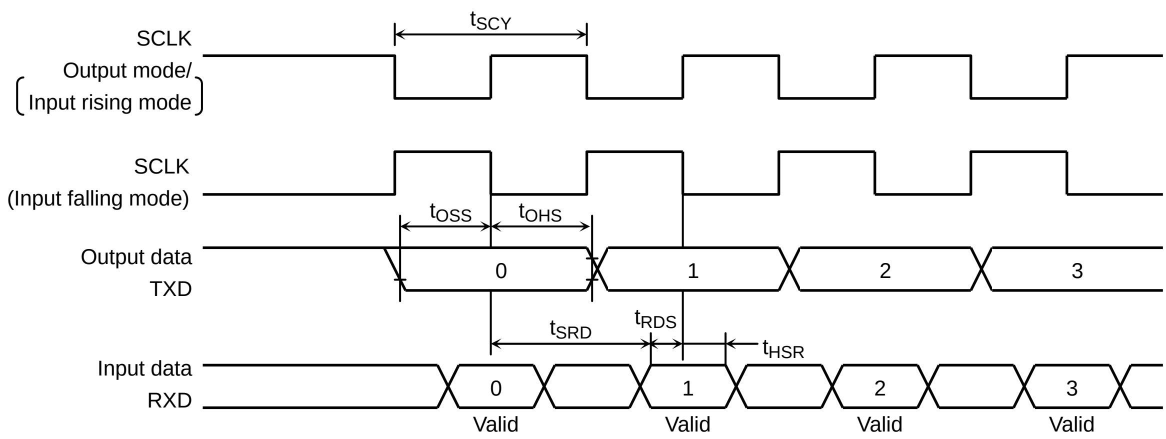
4.3.5 Serial channel timing

(1) SCLK input mode (I/O interface mode)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
SCLK cycle	t_{SCY}	16T		200	266	ns
Output data → SCLK rising/ falling	t_{OSS}	$t_{SCY}/2 - 4T - 30$		20	36.4	
SCLK rising/ falling → Output data hold	t_{OHS}	$t_{SCY}/2 + 2T$ -20		105	146	
SCLK rising/ falling → Input data hold	t_{HSR}	2T + 10		35	43	
SCLK rising/ falling → Input data valid	t_{SRD}		$t_{SCY} - 20$	180	246	
Input data valid → SCLK rising/ falling	t_{RDS}	20		20	20	

(2) SCLK output mode (I/O interface mode)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
SCLK cycle (Programmable)	t_{SCY}	16T	8192T	200	266	ns
Output data → SCLK rising/ falling	t_{OSS}	$t_{SCY}/2 - 40$		60	93	
SCLK rising/ falling → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		60	93	
SCLK rising/ falling → Input data hold	t_{HSR}	0		0	0	
SCLK rising/ falling → Input data valid	t_{SRD}		$t_{SCY} - 1T - 50$	137.5	199	
Input data valid → SCLK rising/ falling	t_{RDS}	1T + 50		62.5	66	



4.3.6 Timer input pulse (TA0IN, TA2IN, TB0IN0, TB1IN0)

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
Clock cycle	t_{VCK}	8T+100		200	234	ns
Low level pulse width	t_{VCKL}	4T + 40		90	107	
High level pulse width	t_{VCKH}	4T + 40		90	107	

4.3.7 Interrupt Operation

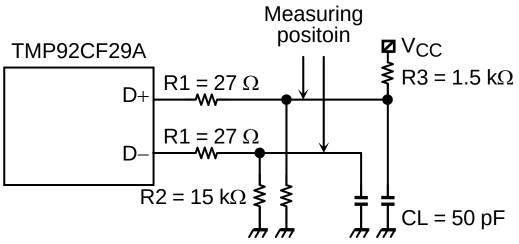
Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
INT0~INT7 low width	t_{INTAL}	2T + 40		65	74	ns
INT0~INT7 high width	t_{INTAH}	2T + 40		65	74	

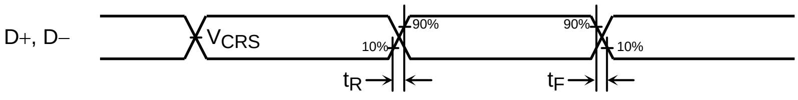
4.3.8 USB Timing (Full-speed)

DVCCA = 3.3 ± 0.3 V/f_{USB} = 48 MHz

Parameter	Symbol	Min	Max	Unit
D+, D– rising time	t_R	4	20	ns
D+, D– falling time	t_F	4	20	
Output signal crossover voltage	V_{CRS}	1.3	2.0	V

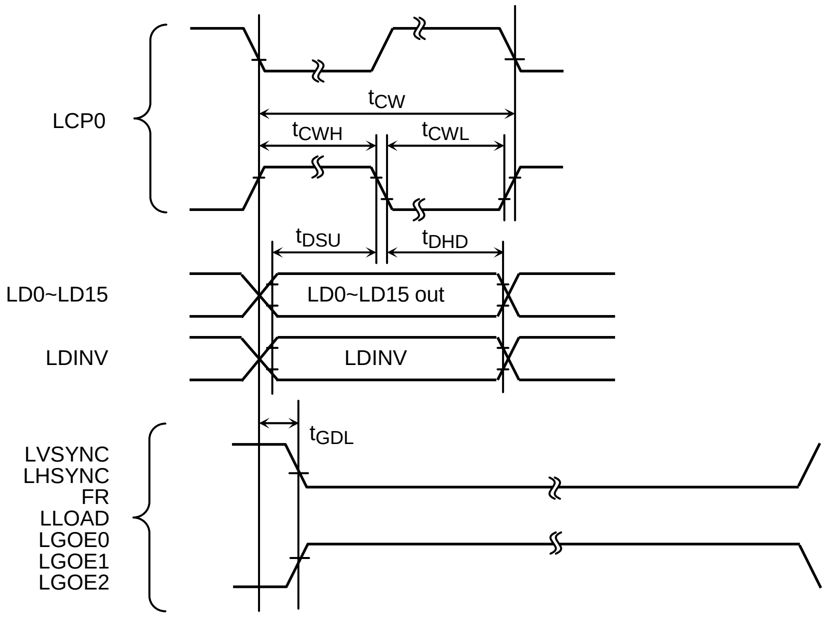
AC measuring condition





4.3.9 LCD Controller

Parameter	Symbol	Variable		80 MHz (n=0)	60 MHz (n=0)	Unit
		Min	Max			
LCP0 clock period	t_{CW}	2T(n+1)		25	33.3	ns
LCP0 high width (Include phase inversion)	t_{CWH}	T(n+1) − 5		7.5	11.6	
LCP0 low width (Include phase inversion)	t_{CWL}	T(n+1) − 5		7.5	11.6	
Data valid →LCP0 falling (Include phase inversion)	t_{DSU}	T(n+1) − 7.5		5	9.1	
LCP0 falling →Data hold (Include phase inversion)	t_{DHD}	T(n+1) − 7.5		5	9.1	
Signal delay from LCP0 basic changing point (Include phase inversion)	t_{GDL}	-15	15	±15	±15	



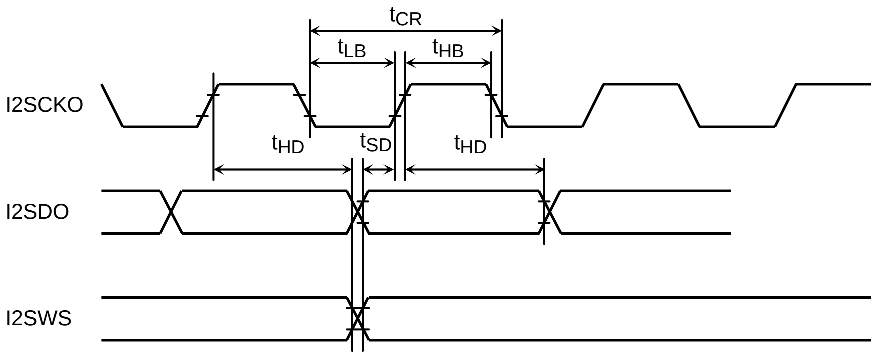
AC measuring condition

- C_L = 50 pF (LCP0 only C_L = 30 pF)

Note: The “n” in “Variable” show value that is set to LCDMODE0<SCPW1:0>.
Example: If LCDMODE0<SCPW1:0> = “01”, n=1, t_{RWP} = 2T(n+1) = 2T

4.3.10 I^2S Timing

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
I2SCKO clock period	t_{CR}	t_{IC}		100	100	ns
I2SCKO high width	t_{HB}	0.5 t_{CR} − 15		35	35	
I2SCKO low width	t_{LB}	0.5 t_{CR} − 15		35	35	
I2SDO, I2SWS setup time	t_{SD}	0.5 t_{CR} − 15		35	35	
I2SDO, I2SWS hold time	t_{HD}	0.5 t_{CR} − 8		42	42	



Note: The Maximum operation frequency of I2SCKO in I^2S circuit is 10MHz. Don't set I2SCKO to value more than 10MHz.

AC measuring condition

- I2SCKO, I2SDO and I2SWS pins $C_L = 30$ pF

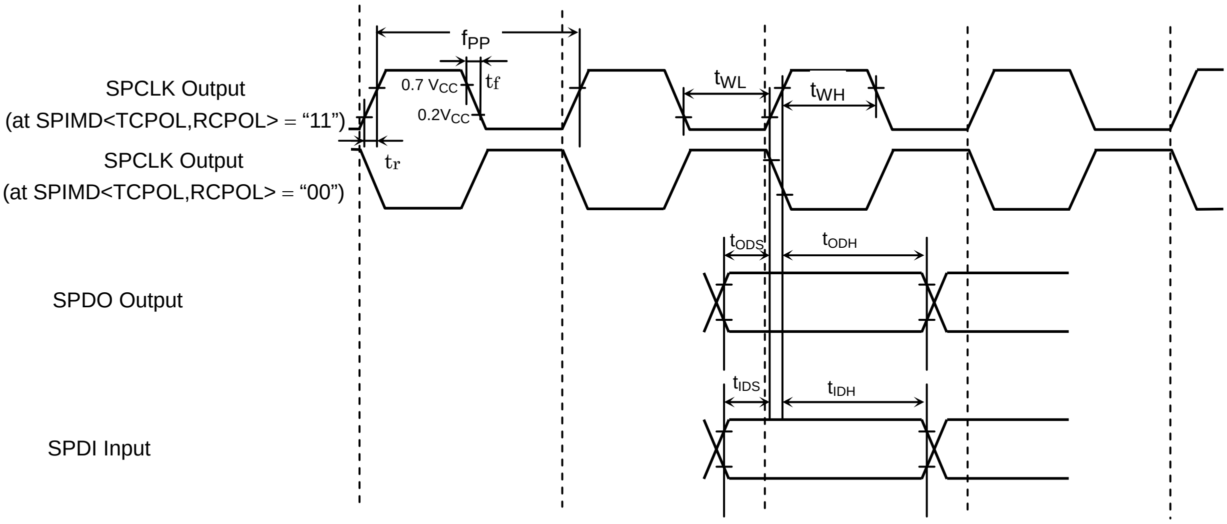
4.3.11 SPI Controller

Parameter	Symbol	Variable		80 MHz	60 MHz	Unit
		Min	Max			
SPCLK frequency (= 1/S)	f_{PP}		20	20	15	MHz
SPCLK rising time	t_r		6	6	6	ns
SPCLK falling time	t_f		6	6	6	
SPCLK low width	t_{WL}	0.5S − 6		19	28	
SPCLK high width	t_{WH}	0.5S − 6		19	28	
Output data valid → SPCLK rising/falling	t_{ODS}	0.5S − 18		7	15	
SPCLK rising/ falling → Output data hold	t_{ODH}	0.5S − 10		15	23.4	
Input data valid → SPCLK rising/ falling	t_{IDS}	5		5	5	
SPCLK rising/ falling → Input data valid	t_{IDH}	5		5	5	

AC measuring condition

•Clock of top column in above table shows system clock frequency, and "S" in "Variable" show SPCLK clock cycle [ns].

• $C_L = 25$ pF



4.4 AD Conversion Characteristics

<table>
<tr><th>Parameter</th><th>Symbol</th><th>Condition</th><th>Min</th><th>Typ.</th><th>Max</th><th>Unit</th></tr>
<tr><td>Analog reference voltage (+)</td><td>VREFH</td><td></td><td>AVCC − 0.2</td><td>AVCC</td><td>AVCC</td><td rowspan="5">V</td></tr>
<tr><td>Analog reference voltage (−)</td><td>VREFL</td><td></td><td>DVSS</td><td>DVSS</td><td>DVSS + 0.2</td></tr>
<tr><td>AD converter power supply voltage</td><td>AVCC</td><td></td><td>DVCC3A</td><td>DVCC3A</td><td>DVCC3A</td></tr>
<tr><td>AD converter ground</td><td>AVSS</td><td></td><td>DVSS</td><td>DVSS</td><td>DVSS</td></tr>
<tr><td>Analog input voltage</td><td>AVIN</td><td></td><td>VREFL</td><td></td><td>VREFH</td></tr>
<tr><td rowspan="2">Analog current for analog reference voltage</td><td>IREFON</td><td><VREFON> = “1”</td><td></td><td>0.38</td><td>0.45</td><td>mA</td></tr>
<tr><td>IREFOFF</td><td><VREFON> = “0”</td><td></td><td>1</td><td>5</td><td>μA</td></tr>
<tr><td>Total error
(Quantize error of ±0.5 LSB is included)</td><td>E_T</td><td>Conversion speed at 12μs</td><td></td><td>±2.0</td><td>±4.0</td><td>LSB</td></tr>
</table>

Note1: 1 LSB = (VREFH−VREFL)/1024[V]

Note2: Minimum frequency for operation

Minimum clock for AD converter operate is 3MHz. (Clock frequency that is seleted by Clock gear ≥ f_{SYS} = 3MHz)

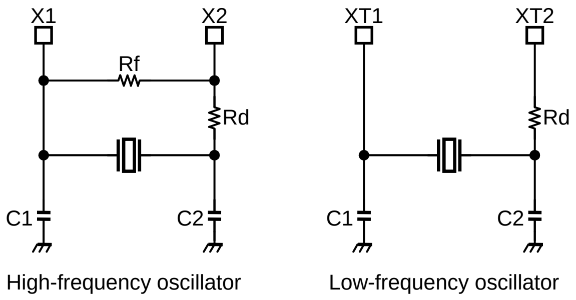
Note3: The power supply current from AVCC pin is included in the power supply current of V_{CC} pin (I_{CC}).

4.5 Recommended Oscillation Circuit

The TMP92CF29A has been evaluated by the oscillator vender below. Use this information when selecting external parts.

Note: The total load value of the oscillator is the sum of external loads (C1 and C2) and the floating load of the actual assembled board. There is a possibility of operating error when using C1 and C2 values in the table below. When designing the board, design the minimum length pattern around the oscillator. We also recommend that oscillator evaluation be carried out using the actual board.

(1) Connection example



(2) Recommended ceramic oscillator

TMP92CF29A recommends the high-frequency oscillator by Murata Manufacturing Co., Ltd.

Please refer to the following URL;
http://www.murata.com/

5. Table of Special function registers (SFRs)

The SFRs include the I/O ports and peripheral control registers allocated to the 8-Kbyte address space from 000000H to 001FF0H.

(1) I/O Port
(2) Interrupt control
(3) Memory controller
(4) TSI(Touch screen I/F)
(5) SDRAM controller
(6) LCD controller
(7) PMC
(8) USB controller
(9) SPI controller
(10) MMU
(11) NAND-Flash controller
(12) DMA controller
(13) Clock gear, PLL
(14) 8-bit timer
(15) 16-bit timer
(16) SIO
(17) SBI
(18) AD converter
(19) Watchdog timer
(20) RTC(Real time clock)
(21) MLD(Melody/alarm generator)
(22) I²S
(23) MAC

Table layout

Symbol	Name	Address	7	6			1	0	
									Bit Symbol
									Read/Write
									Initial value System Reset State
									Initial value HOT Reset State
									Remarks

Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these register.

Example: When setting bit0 only of the register PxCR, the instruction "SET 0, (PxCR)" cannot be used. The LD (transfer) instruction must be used to write all eight bits.

Read/Write

R/W:	Both read and write are possible.
R:	Only read is possible.
W:	Only write is possible.
W*:	Both read and write are possible (when this bit is read as1)
Prohibit RMW:	Read modify write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read modify write instructions.)
R/W*:	Read modify write is prohibited when controlling the pull-up resistor.

Table 5.1 I/O Register Address Map

[1] Port (1/2)

Address	Name
0000H	
1H	
2H	
3H	
4H	P1
5H	
6H	P1CR
7H	P1FC
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0010H	P4
1H	
2H	
3H	P4FC
4H	P5
5H	
6H	
7H	P5FC
8H	P6
9H	
AH	P6CR
BH	P6FC
CH	P7
DH	
EH	P7CR
FH	P7FC

Address	Name
0020H	P8
1H	P8FC2
2H	
3H	P8FC
4H	P9
5H	P9FC2
6H	P9CR
7H	P9FC
8H	PA
9H	
AH	
BH	PAFC
CH	
DH	
EH	
FH	

Address	Name
0030H	PC
1H	PCFC2
2H	PCCR
3H	PCFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	PF
DH	
EH	PFCR
FH	PFFC

Address	Name
0040H	PG
1H	
2H	
3H	PGFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	PJ
DH	
EH	PJCR
FH	PJFC

Address	Name
0050H	PK
1H	
2H	
3H	PKFC
4H	PL
5H	Reserved
6H	PLCR
7H	PLFC
8H	PM
9H	
AH	
BH	PMFC
CH	PN
DH	
EH	PNCR
FH	PNFC

Address	Name
0060H	PP
1H	PPFC2
2H	PPCR
3H	PPFC
4H	PR
5H	
6H	PRCR
7H	PRFC
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0070H	Reserved
1H	Reserved
2H	Reserved
3H	Reserved
4H	Reserved
5H	Reserved
6H	Reserved
7H	Reserved
8H	Reserved
9H	Reserved
AH	Reserved
BH	Reserved
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

Note: Do not access no allocated name address.

[1] Port (2/2)

Address	Name
0080H	
1H	P1DR
2H	
3H	
4H	P4DR
5H	P5DR
6H	P6DR
7H	P7DR
8H	P8DR
9H	P9DR
AH	PADR
BH	
CH	PCDR
DH	
EH	
FH	PFDR

Address	Name
0090H	PGDR
1H	
2H	
3H	PJDR
4H	PKDR
5H	PLDR
6H	PMDR
7H	PNDR
8H	PPDR
9H	PRDR
AH	
BH	PTDR
CH	
DH	PVDR
EH	
FH	PXDR

Address	Name
00A0H	PT
1H	Reserved
2H	PTCR
3H	PTFC
4H	
5H	
6H	
7H	
8H	PV
9H	PVFC2
AH	PVCR
BH	PVFC
CH	
DH	
EH	
FH	

Address	Name
00B0H	PX
1H	PXFC2
2H	PXCR
3H	PXFC
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[2] INTC

Address	Name
00D0H	INTE12
1H	INTE34
2H	INTE56
3H	INTE7
4H	INTETA01
5H	INTETA23
6H	INTETA45
7H	INTETA67
8H	INTETB0
9H	INTETB1
AH	
BH	INTES0
CH	INTES1
DH	
EH	
FH	

Address	Name
00E0H	INTESBIADM
1H	INTESPI
2H	Reserved
3H	INTEUSB
4H	Reserved
5H	INTEALM
6H	Reserved
7H	
8H	INTERTC
9H	INTEKEY
AH	INTELCD
BH	INTEI2S0
CH	INTENDFC
DH	Reserved
EH	INTEP0
FH	INTEAD

Address	Name
00F0H	INTE0
1H	INTETC01 /INTEDMA01
2H	INTETC23 /INTEDMA23
3H	INTETC45 /INTEDMA45
4H	INTETC67
5H	SIMC
6H	IIMC0
7H	INTWDT
8H	INTCLR
9H	
AH	IIMC1
BH	
CH	
DH	
EH	
FH	Reserved

Address	Name
0100H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	DMA4V
5H	DMA5V
6H	DMA6V
7H	DMA7V
8H	DMAB
9H	DMAR
AH	DMASEL
BH	
CH	
DH	
EH	
FH	

[3] MEMC

Address	Name
0140H	B0CSL
1H	B0CSH
2H	MAMR0
3H	MSAR0
4H	B1CSL
5H	B1CSH
6H	MAMR1
7H	MSAR1
8H	B2CSL
9H	B2CSH
AH	MAMR2
BH	MSAR2
CH	B3CSL
DH	B3CSH
EH	MAMR3
FH	MSAR3

Address	Name
0150H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	BEXCSL
9H	BEXCSH
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0160H	
1H	
2H	
3H	
4H	
5H	
6H	PMEMCR
7H	
8H	CSTMGCR
9H	WRTMGCR
AH	RDTMGCR0
BH	RDTMGCR1
CH	BROMCR
DH	RAMCR
EH	
FH	

[4] TSI

Address	Name
01F0H	TSICR0
1H	TSICR1
2H	Reserved
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[5] SDRAMC

Address	Name
0250H	SDACR
1H	SDCISR
2H	SDRCR
3H	SDCMM
4H	SDBLS
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] LCDC

Address	Name
0280H	LCDMODE0
1H	LCDMODE1
2H	
3H	LCDDVM0
4H	LCDSIZE
5H	LCDCTL0
6H	LCDCTL1
7H	LCDCTL2
8H	LCDDVM1
9H	
AH	LCDHSP
BH	LCDHSP
CH	LCDVSP
DH	LCDVSP
EH	LCDPRVSP
FH	LCDHSDLY

Address	Name
0290H	LCDHSDLY
1H	LCDO0DLY
2H	LCDO1DLY
3H	LCDO2DLY
4H	LCDHSW
5H	LCDLDW
6H	LCDHO0W
7H	LCDHO1W
8H	LCDHO2SW
9H	LCDHWB8
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
02A0H	LSAML
1H	LSAMM
2H	LSAMH
3H	
4H	LSASL
5H	LSASM
6H	LSASH
7H	
8H	LSAHX
9H	LSAHX
AH	LSAHY
BH	LSAHY
CH	LSASS
DH	LSASS
EH	LSACS
FH	LSACS

[7] PMC

Address	Name
02F0H	PMCCTL
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[8] USBC (1/2)

Address	Name
0500H	Descriptor
to	RAM
067FH	(384 byte)

Address	Name
0780H	ENDPOINT0
1H	ENDPOINT1
2H	ENDPOINT2
3H	ENDPOINT3
4H	
5H	
6H	
7H	
8H	
9H	EP1_MODE
AH	EP2_MODE
BH	EP3_MODE
CH	
DH	
EH	
FH	

Address	Name
0790H	EP0_STATUS
1H	EP1_STATUS
2H	EP2_STATUS
3H	EP3_STATUS
4H	
5H	
6H	
7H	
8H	EP0_SIZE_L_A
9H	EP1_SIZE_L_A
AH	EP2_SIZE_L_A
BH	EP3_SIZE_L_A
CH	
DH	
EH	
FH	

Address	Name
07A0H	
1H	EP1_SIZE_L_B
2H	EP2_SIZE_L_B
3H	EP3_SIZE_L_B
4H	
5H	
6H	
7H	
8H	Reserved
9H	EP1_SIZE_H_A
AH	EP2_SIZE_H_A
BH	EP3_SIZE_H_A
CH	
DH	
EH	
FH	

Address	Name
07B0H	
1H	EP1_SIZE_H_B
2H	EP2_SIZE_H_B
3H	EP3_SIZE_H_B
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
07C0H	bmRequestType
1H	bRequest
2H	wValue_L
3H	wValue_H
4H	wIndex_L
5H	wIndex_H
6H	wLength_L
7H	wLength_H
8H	SetupReceived
9H	Current_Config
AH	Standard Request
BH	Request
CH	DATASET1
DH	DATASET2
EH	USB STATE
FH	EOP

Address	Name
07D0H	COMMAND
1H	EPx_SINGLE1
2H	Reserved
3H	EPx_BCS1
4H	Reserved
5H	Reserved
6H	INT_Control
7H	Reserved
8H	Standard Request Mode
9H	Request Mode
AH	Reserved
BH	Reserved
CH	Reserved
DH	Reserved
EH	ID_CONTROL
FH	ID_STATE

Note: Do not access no allocated name address.

[8] USBC (2/2)

Address	Name
07E0H	Port Status
1H	FRAME_L
2H	FRAME_H
3H	ADDRESS
4H	Reserved
5H	Reserved
6H	USBREADY
7H	Reserved
8H	Set Descriptor STALL
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
07F0H	USBINTFR1
1H	USBINTFR2
2H	USBINTFR3
3H	USBINTFR4
4H	USBINTMR1
5H	USBINTMR2
6H	USBINTMR3
7H	USBINTMR4
8H	USBCR1
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[9] SPIC

Address	Name
0820H	SPIMD
1H	SPIMD
2H	SPICT
3H	SPICT
4H	SPIST
5H	SPIST
6H	SPICR
7H	SPICR
8H	
9H	
AH	
BH	
CH	SPIIE
DH	SPIIE
EH	
FH	

Address	Name
0830H	SPITD0
1H	SPITD0
2H	SPITD1
3H	SPITD1
4H	SPIRD0
5H	SPIRD0
6H	SPIRD1
7H	SPIRD1
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] MMU

Address	Name
0880H	LOCALPX
1H	LOCALPX
2H	LOCALPY
3H	LOCALPY
4H	LOCALPZ
5H	LOCALPZ
6H	
7H	
8H	LOCALLX
9H	LOCALLX
AH	LOCALLY
BH	LOCALLY
CH	LOCALLZ
DH	LOCALLZ
EH	
FH	

Address	Name
0890H	LOCALRX
1H	LOCALRX
2H	LOCALRY
3H	LOCALRY
4H	LOCALRZ
5H	LOCALRZ
6H	
7H	
8H	LOCALWX
9H	LOCALWX
AH	LOCALWY
BH	LOCALWY
CH	LOCALWZ
DH	LOCALWZ
EH	
FH	

Address	Name
08A0H	LOCALESX
1H	LOCALESX
2H	LOCALESY
3H	LOCALESY
4H	LOCALESZ
5H	LOCALESZ
6H	
7H	
8H	LOCALEDX
9H	LOCALEDX
AH	LOCALEDY
BH	LOCALEDY
CH	LOCALEDZ
DH	LOCALEDZ
EH	
FH	

Address	Name
08B0H	LOCALOSX
1H	LOCALOSX
2H	LOCALOSY
3H	LOCALOSY
4H	LOCALOSZ
5H	LOCALOSZ
6H	
7H	
8H	LOCALODX
9H	LOCALODX
AH	LOCALODY
BH	LOCALODY
CH	LOCALODZ
DH	LOCALODZ
EH	
FH	

Note: Do not access no allocated name address.

[11] NAND-Flash controller

Address	Name
08C0H	NDFMCR0
1H	NDFMCR0
2H	NDFMCR1
3H	NDFMCR1
4H	NDECCRD0
5H	NDECCRD0
6H	NDECCRD1
7H	NDECCRD1
8H	NDECCRD2
9H	NDECCRD2
AH	NDECCRD3
BH	NDECCRD3
CH	NDECCRD4
DH	NDECCRD4
EH	
FH	

Address	Name
08D0H	NDRSCA0
1H	NDRSCA0
2H	NDRSCD0
3H	
4H	NDRSCA1
5H	NDRSCA1
6H	NDRSCD1
7H	
8H	NDRSCA2
9H	NDRSCA2
AH	NDRSCD2
BH	
CH	NDRSCA3
DH	NDRSCA3
EH	NDRSCD3
FH	

Address	Name
1FF0H	NDFDTR0
1H	NDFDTR0
2H	NDFDTR1
3H	NDFDTR1
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[12] DMAC

Address	Name
0900H	HDMAS0
1H	HDMAS0
2H	HDMAS0
3H	
4H	HDMAD0
5H	HDMAD0
6H	HDMAD0
7H	
8H	HDMACA0
9H	HDMACA0
AH	HDMACB0
BH	HDMACB0
CH	HDMAM0
DH	
EH	
FH	

Address	Name
0910H	HDMAS1
1H	HDMAS1
2H	HDMAS1
3H	
4H	HDMAD1
5H	HDMAD1
6H	HDMAD1
7H	
8H	HDMACA1
9H	HDMACA1
AH	HDMACB1
BH	HDMACB1
CH	HDMAM1
DH	
EH	
FH	

Address	Name
0920H	HDMAS2
1H	HDMAS2
2H	HDMAS2
3H	
4H	HDMAD2
5H	HDMAD2
6H	HDMAD2
7H	
8H	HDMACA2
9H	HDMACA2
AH	HDMACB2
BH	HDMACB2
CH	HDMAM2
DH	
EH	
FH	

Address	Name
0930H	HDMAS3
1H	HDMAS3
2H	HDMAS3
3H	
4H	HDMAD3
5H	HDMAD3
6H	HDMAD3
7H	
8H	HDMACA3
9H	HDMACA3
AH	HDMACB3
BH	HDMACB3
CH	HDMAM3
DH	
EH	
FH	

Address	Name
0940H	HDMAS4
1H	HDMAS4
2H	HDMAS4
3H	
4H	HDMAD4
5H	HDMAD4
6H	HDMAD4
7H	
8H	HDMACA4
9H	HDMACA4
AH	HDMACB4
BH	HDMACB4
CH	HDMAM4
DH	
EH	
FH	

Address	Name
0950H	HDMAS5
1H	HDMAS5
2H	HDMAS5
3H	
4H	HDMAD5
5H	HDMAD5
6H	HDMAD5
7H	
8H	HDMACA5
9H	HDMACA5
AH	HDMACB5
BH	HDMACB5
CH	HDMAM5
DH	
EH	
FH	

Address	Name
0970H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	Reserved
DH	Reserved
EH	HDMAE
FH	HDMATR

Note: Do not access no allocated name address.

[13] CGEAR, PLL

Address	Name
10E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	EMCCR2
6H	Reserved
7H	
8H	PLLCR0
9H	PLLCR1
AH	
BH	
CH	
DH	
EH	
FH	

[14] 8-bit timer

Address	Name
1100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA1FFCR
6H	
7H	
8H	TA23RUN
9H	
AH	TA2REG
BH	TA3REG
CH	TA23MOD
DH	TA3FFCR
EH	
FH	

Address	Name
1110H	TA45RUN
1H	
2H	TA4REG
3H	TA5REG
4H	TA45MOD
5H	TA5FFCR
6H	
7H	
8H	TA67RUN
9H	
AH	TA6REG
BH	TA7REG
CH	TA67MOD
DH	TA7FFCR
EH	
FH	

[15] 16-bit timer

Address	Name
1180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

Address	Name
1190H	TB1RUN
1H	
2H	TB1MOD
3H	Reserved
4H	
5H	
6H	
7H	
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

[16] SIO

Address	Name
1200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	SIR0CR
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	SIR1CR

[17] SBI

Address	Name
1240H	SBICR1
1H	SBIDBR
2H	I2CAR
3H	SBICR2/SBISR
4H	SBIBR0
5H	
6H	
7H	SBICR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[18] 10-bit ADC

Address	Name
12A0H	ADREG0L
1H	ADREG0H
2H	ADREG1L
3H	ADREG1H
4H	ADREG2L
5H	ADREG2H
6H	ADREG3L
7H	ADREG3H
8H	ADREG4L
9H	ADREG4H
AH	ADREG5L
BH	ADREG5H
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

Address	Name
12B0H	ADREGSPL
1H	ADREGSPH
2H	Reserved
3H	Reserved
4H	ADCM0REGL
5H	ADCM0REGH
6H	ADCM1REGL
7H	ADCM1REGH
8H	ADMOD0
9H	ADMOD1
AH	ADMOD2
BH	ADMOD3
CH	ADMOD4
DH	ADMOD5
EH	
FH	ADCCLK

[19] WDT

Address	Name
1300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[20] RTC

Address	Name
1320H	SECR
1H	MINR
2H	HOURR
3H	DAYR
4H	DATER
5H	MONTHR
6H	YEARR
7H	PAGER
8H	RESTR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[21] MLD

Address	Name
1330H	ALM
1H	MELALMC
2H	MELFL
3H	MELFH
4H	ALMINT
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[22] I²S

Address	Name
1800H	I2S0BUF
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	I2S0CTL
9H	I2S0CTL
AH	I2S0C
BH	I2S0C
CH	
DH	
EH	
FH	

Address	Name
1810H	Reserved
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	Reserved
9H	Reserved
AH	Reserved
BH	Reserved
CH	
DH	
EH	
FH	

Note: Do not access no allocated name address.

[23] MAC

Address	Name
1BE0H	MACMA
1H	MACMA
2H	MACMA
3H	MACMA
4H	MACMB
5H	MACMB
6H	MACMB
7H	MACMB
8H	MACORL
9H	MACORL
AH	MACORL
BH	MACORL
CH	MACORH
DH	MACORH
EH	MACORH
FH	MACORH

Address	Name
1BF0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	MACCR
DH	
EH	
FH	

(1) I/O ports (1/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1	PORT1	0004H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
			–							
P4	PORT4	0010H	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
P5	PORT5	0014H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
P6	PORT6	0018H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
			–							
P7	PORT7	001CH		P76	P75	P74	P73	P72	P71	P70
				R/W						
				Data from external port (Output latch register is set to “1”)		Data from external port (Output latch register is cleared to “0”)		Data from external port (Output latch register is set to “1”)		1
				–		–		–		–
P8	PORT8	0020H	P87	P86			P83	P82	P81	P80
			R/W				R/W			
			1	1			1	0 (Note)	1	1
			–	–			–	–	–	–
P9	PORT9	0024H	P97	P96				P92	P91	P90
			R					R/W		
			Data from external port					Data from external port (Output latch register is set to “1”)		
			–					–		
PA	PORTA	0028H	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
			Data from external port							
			–							
PC	PORTC	0030H	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
			R/W							
			Data from external port (Output latch register is set to “1”)							
			–							
PF	PORTF	003CH	PF7					PF2	PF1	PF0
			R/W					R/W		
			1					Data from external port (Output latch register is set to “1”)		
			–					–		
PG	PORTG	0040H			PG5	PG4	PG3	PG2	PG1	PG0
					R					
					Data from external port					
					–					
PJ	PORTJ	004CH	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
			R/W							
			1	Data from external port (Output latch register is set to “1”)		1	1	1	1	1
			–	–		–	–	–	–	–

Note: If it is started at boot mode (AM [1:0] = “11”), output latch of P82 is set to “1”.

(1) I/O ports (2/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PK	PORTK	0050H	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
			R/W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
PL	PORTL	0054H	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
			R/W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
PM	PORTM	0058H	PM7					PM2	PM1	
			R/W					R/W		
			1					1	1	
			–					–	–	
PN	PORTN	005CH	PN7	PN6	PN5	PN4	PN3	PN2	PN1	PN0
			R/W							
			Data from external port (Output latch register is cleared to “1”)							
			–							
PP	PORTP	0060H		PP6	PP5	PP4	PP3			
				R/W						
				0	Data from external port (Output latch register is cleared to “0”)					
				–	–					
PR	PORTR	0064H					PR3	PR2	PR1	PR0
							R/W			
							Data from external port (Output latch register is cleared to “0”)			
							–			
PT	PORTT	00A0H	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
			–							
PV	PORTV	00A8H	PV7	PV6						
			R/W							
			Data from external port (Output latch register is cleared to “0”)							
			–							
PX	PORTX	00B0H			PX5	PX4				
					R/W					
					Data from external port (Output latch register is cleared to “0”)					
					–					

(1) I/O ports (3/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1CR	PORT1 control register	0006H (Prohibit RMW)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input 1:Output							
P1FC	PORT1 function register	0007H (Prohibit RMW)								P1F
										W
										0/1
										–
										0: Port 1:Data bus (D8 to D15)
P4FC	PORT4 function register	0013H (Prohibit RMW)	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			–	–	–	–	–	–	–	–
			0: Port 1: Address bus (A0 to A7)							
P5FC	PORT5 function register	0017H (Prohibit RMW)	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			–	–	–	–	–	–	–	–
			0: Port 1: Address bus (A8 to A15)							
P6CR	PORT6 control register	001AH (Prohibit RMW)	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input 1: Output							
P6FC	PORT6 function register	001BH (Prohibit RMW)	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
			–	–	–	–	–	–	–	–
			0: Port 1: Address bus (A16 to A23)							
P7CR	PORT7 control register	001EH (Prohibit RMW)		P76C	P75C	P74C	P73C	P72C	P71C	
				W						
				0	0	0	0	0	0	
				–	–	–	–	–	–	
				0: Input port, $\overline{WAIT}$ 1:Output port	0: Input port, NDR/$\overline{B}$ 1: Output port, R/$\overline{W}$	0: Input port 1: Output port, EA25	0: Input port 1: Output port, EA24	0: Input port 1: Output port, $\overline{NDWE}$ @ <P72> = 0, $\overline{WRLU}$ @ <P72> = 1	0: Input port 1: Output port, $\overline{NDRE}$ @ <P71> = 0, $\overline{WRLL}$ @ <P71> = 1	
P7FC	PORT7 function register	001FH (Prohibit RMW)		P76F	P75F	P74F	P73F	P72F	P71F	P70F
				W						
				0	0	0	0	0	0	0/1
				–	–	–	–	–	–	–
				0: Port 1: $\overline{WAIT}$	0: Port 1:NDR/$\overline{B}$, R/$\overline{W}$	0:Port 1: EA25	0:Port 1: EA24	0: Port 1:$\overline{NDWE}$ @ <P72> = 0, $\overline{WRLU}$ @ <P72> = 1	0: Port 1: $\overline{NDRE}$ @ <P71> = 0, $\overline{WRLL}$ @ <P71> = 1	0: Port 1: $\overline{RD}$

(1) I/O ports (4/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P8FC	PORT8 function register	0023H (Prohibit RMW)	P87F	P86F			P83F	P82F	P81F	P80F
							W			
			0	0			0	0	0	0
			–	–			–	–	–	–
			0: Port 1:<P87F2>	0: Port 1:<P86F2>			0: Port 1: $\overline{\text{CS3}}$, $\overline{\text{CSXA}}$	0: Port, $\overline{\text{CSZA}}$ 1: $\overline{\text{CS2}}$, $\overline{\text{SDCS}}$	0: Port 1: $\overline{\text{CS1}}$	0: Port 1: $\overline{\text{CS0}}$
P8FC2	PORT8 function fegister2	0021H (Prohibit RMW)	P87F2	P86F2			P83F2	P82F2	P81F2	
			W				W			
			0	0			0	0	0	
			–	–			–	–	–	
			0: $\overline{\text{CSXB}}$ 1: $\overline{\text{ND1CE}}$	0: $\overline{\text{CSZD}}$ 1: $\overline{\text{ND0CE}}$			0: Output Port, $\overline{\text{CS3}}$ 1: $\overline{\text{CSXA}}$	0: Output port, $\overline{\text{CS2}}$ 1: $\overline{\text{CSZA}}$, $\overline{\text{SDCS}}$	0: <P81F> 1: $\overline{\text{SDCS}}$	
P9CR	PORT9 control register	0026H (Prohibit RMW)						P92C	P91C	P90C
								W		
								0	0	0
								–	–	–
								0 Input port, $\overline{\text{CTS0/1}}$, SCLK0/1 1: Output port, SCLK0,1	0: Input port, RXD0/1 1: Output port,	0: Input port, 1: Output port, TXD0/1
P9FC	PORT9 function register	0027H (Prohibit RMW)		P96F				P92F		P90F
				W				W		W
				0				0		0
				–				–		–
				0: Input port, 1: INT4				0:Port, $\overline{\text{CTS0/1}}$,SCLK0/1 1:SCLK0		0:Port 1:TXD0/1
P9FC2	PORT9 function register2	0025H (Prohibit RMW)	–		P95F2	P94F2	P93F2	–		P90FC2
			W		W	W	W	W		W
			0		0	0	0	0		0
			–		–	–	–	–		–
			Always write “0”		P92 SCLK selection 0: SCLK0 1: SLCK1 SIO0 SCLK, $\overline{\text{CTS}}$ input selection 0: P92 1: PP5	SIO0 RXD selection 0: P91 1: PP4	P90 TXD selection 0: TXD0 1: TXD1	Always write “0”		0:CMOS 1:Open -Drain

(1) I/O ports (5/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PAFC	PORTA function register	002BH (Prohibit RMW)	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Key-in disable 1: Key-in enable							
PCCR	PORTC control register	0032H (Prohibit RMW)	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input port, 1: Output port, KO output (Open -drain)	0: Input port, EA28 1: Output port, SPCLK output	0: Input port, EA27 1: Output port, SPDO output	0: Input port, EA26 1: Output port, SPDI input	0: Input port, INT3 1: Output port, TA2IN	0: Input port, INT2 1: Output port	0: Input port, INT1 1: Output port, TA0IN	0: Input port, INT0 1: Output port
PCFC	PORTC function register	0033H (Prohibit RMW)	PC7F	PC6F	PC5F	PC4F	PC3F	PC2F	PC1F	PC0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Port 1:KO output (Open -Drain)	0: Port 1:EA28, SPCLK output	0:Port 1:EA27, SPDO output	0:Port 1:EA26, SPDI input	0:Port 1:INT3 ,TA2IN	0: Port 1: INT2	0: Port 1: INT1, TA0IN	0: Port 1:INT0
PCFC2	PORTC function 2 register	0031H (Prohibit RMW)				PC4F2				
						W				
						0				
						–				
						SPDI selection 0: PR0 1: PC4				
PFCR	PORTF control register	003EH (Prohibit RMW)						PF2C	PF1C	PF0C
								W		
								0	0	0
								–	–	–
								0: Input, 1: Output		
PFFC	PORTF function register	003FH (Prohibit RMW)	PF7F					PF2F	PF1F	PF0F
			W					W		
			1					0	0	0
			–					–	–	–
			0:Output port 1: SDCLK					0:Port 1:I2S0WS	0:Port 1:I2S0DO	0:Port 1:I2S0CKO

(1) I/O ports (6/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PGFC	PORTG function register	0043H (Prohibit RMW)					PG3F			
							W			
							0			
							–			
							0:Input port,AN3 1: $\overline{ADTRG}$			
PJCR	PORTJ control register	004EH (Prohibit RMW)		PJ6C	PJ5C					
				W						
				0	0					
				–	–					
				0:Input port 1:output port, NDCLE output	0:Input port 1:output port, NDALE output					
PJFC	PORTJ function register	004FH (Prohibit RMW)	PJ7F	PJ6F	PJ5F	PJ4F	PJ3F	PJ2F	PJ1F	PJ0F
					W					
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Port 1: SDCKE	0: Port 1:NDCLE output	0: Port 1:NDALE output	0: Port 1:SDLUDQM	0: Port 1:SDLLDQM	0: Port 1: $\overline{SDWE}$, $\overline{SRWR}$	0: Port 1: $\overline{SDCAS}$, $\overline{SRLUB}$	0: Port 1: $\overline{SDRAS}$, $\overline{SRLLB}$
PKFC	PORTK function register	0053H (Prohibit RMW)	PK7F	PK6F	PK5F	PK4F	PK3F	PK2F	PK1F	PK0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Port 1: LGOE2	0: Port 1: LGOE1	0: Port 1: LGOE0	0: Port 1: LHSYNC	0: Port 1: LVSYNC	0: Port 1: LFR	0: Port 1: LLOAD	0: Port 1: LCP0
PKFC	PORTL control register	0057H (Prohibit RMW)	PL7C	PL6FC	PL5C	PL4C	PL3C	PL2C	PL1C	PL0C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input 1: Output							
PLFC	PORTL function register	0057H (Prohibit RMW)	PL7F	PL6F	PL5F	PL4F	PL3F	PL2F	PL1F	PL0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Port 1: Data bus for LCDC (LD7~LD0)							
PMFC	PORTM function register	005BH (Prohibit RMW)	PM7F					PM2F	PM1F	
			W					W		
			0					0	0	
			–					–	–	
			0: Port 1: PWE					0: Port 1: $\overline{ALARM}$ at <PM2>=1, $\overline{MLDALM}$ at <PM2>=1	0: Port 1:MLDALM at <PM1>=1 TA1OUT at <PM1>=0	

(1) I/O ports (7/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PNCR	PORTN control register	005EH (Prohibit RMW)	PN7C	PN6C	PN5C	PN4C	PN3C	PN2C	PN1C	PN0C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input 1: Output							
PNFC	PORTN function register	005FH (Prohibit RMW)	PN7F	PN6F	PN5F	PN4F	PN3F	PN2F	PN1F	PN0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0:CMOS output 1:Open-Drain output							
PPCR	PORTP control register	0062H (Prohibit RMW)			PP5C	PP4C	PP3C			
					W					
					0	0	0			
					–	–	–			
					0: Input 1: Output					
PPFC	PORTP function register	0063H (Prohibit RMW)		PP6F	PP5F	PP4F	PP3F			
				W						
				0	0	0	0			
				–	–	–	–			
				0: Port 1: TB0OUT0	0: Port 1: INT7, TB1IN0@<PP3F2>=0	0: Port 1: INT6, TB0IN0@<PP2F2>=0	0: Port 1: INT5, TA7OUT@<PP1F2>=0			
PPFC2	PORTP function 2 register	0061H (Prohibit RMW)		PP6F2	PP5F2	PP4F2	PP3F2	PP2F2	PP1F2	PP0F2
				W						
				0	0	0	0	0	0	0
				–	–	–	–	–	–	–
				PP5 SCLK output 0: SCLK1 1: SCLK0 SIO1 SCLK, CTS input 0: PP5 1: P92	SIO1 RXD selection 0: PP4 1: P91	PP3 selection 0: TXD1 1: TXD0	PP5 selection 0: Others 1:SCLK, CTS input or SCLK output	PP4 selection 0: Others 1: RXD input	PP3 selection 0: Others 1: TXD output	PP3 selection 0: CMOS 1:Open -drain
PRCR	PORTR control register	0066H (Prohibit RMW)					PR3C	PR2C	PR1C	PR0C
							W			
							0	0	0	0
							–	–	–	–
							0: Input, 1: Output			
PRFC	PORTR function register	0067H (Prohibit RMW)					PR3F	PR2F	PR1F	PR0F
							W			
							0	0	0	0
							–	–	–	–
							0: Port 1: SPCLK	0: Port 1: SPCS	0: Port 1: SPDO	0: Port 1: SPDI
PTCR	PORTT control register	00A2H (Prohibit RMW)	PT7C	PT6C	PT5C	PT4C	PT3C	PT2C	PT1C	PT0C
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Input 1: Output							
PTFC	PORTT function register	00A3H (Prohibit RMW)	PT7F	PT6F	PT5F	PT4F	PT3F	PT2F	PT1F	PT0F
			W							
			0	0	0	0	0	0	0	0
			–	–	–	–	–	–	–	–
			0: Port 1: Data bus for LCDC (LD15 to LD8)							

(1) I/O ports (8/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PVCR	PORTV control register	00AAH (Prohibit RMW)	PV7C	PV6C						
			W							
			0	0						
			–	–						
			0: Input 1: Output							
PVFC	PORTV function register	00ABH (Prohibit RMW)	PV7F	PV6F						
			W							
			0	0						
			–	–						
			0: Port 1: SCL	0: Port 1: SDA						
PVFC2	PORTV function register 2	00A9H (Prohibit RMW)	PV7F2	PV6F2						
			W							
			0	0						
			–	–						
			0: CMOS 1:Open -drain	0: CMOS 1:Open -drain						
PXCR	PORTX control register	00B2H (Prohibit RMW)			PX5C					
					W					
					0					
					–					
					0: Input 1: Output					
PXFC	PORTX function register	00B3H (Prohibit RMW)			PX5F	PX4F				
					W	W				
					0	0				
					–	–				
					0: Port 1: X1USB input at <PX5C>=0, X1D4 output at <PX5C>=1, <PX5>=1	0: Port 1: CLKOUT at <PX4>=0, LDIV at <PX4>=1				
PXFC2	PORT X function register 2	00B1H (Prohibit RMW)			PX5F2	PX4F2				
					W					
					0	0				
					–	–				
					X1D4 output clock selection 00: X1 pin ×1/8 01: X1 pin ×1/4 10: X1 pin ×1/2 11: X1 pin ×1/1					

(1) I/O ports (9/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1DR	PORT1 drive register	0081H	P17D	P16D	P15D	P14D	P13D	P12D	P11D	P10D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
P4DR	PORT4 drive register	0084H	P47D	P46D	P45D	P44D	P43D	P42D	P41D	P40D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
P5DR	PORT5 drive register	0085H	P57D	P56D	P55D	P54D	P53D	P52D	P51D	P50D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
P6DR	PORT6 drive register	0086H	P67D	P66D	P65D	P64D	P63D	P62D	P61D	P60D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
P7DR	PORT7 drive register	0087H		P76D	P75D	P74D	P73D	P72D	P71D	P70D
				R/W						
				1	1	1	1	1	1	1
				–	–	–	–	–	–	–
				Input/Output buffer drive register for standby mode						
P8DR	PORT8 drive register	0088H	P87D	P86D			P83D	P82D	P81D	P80D
			R/W				R/W			
			1	1			1	1	1	1
			–	–			–	–	–	–
			Input/Output buffer drive register for standby mode							
P9DR	PORT9 drive register	0089H	P97D	P96D				P92D	P91D	P90D
			R/W					R/W		
			1	1				1	1	1
			–	–				–	–	–
			Input/Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		
PADR	PORTA drive register	008AH	PA7D	PA6D	PA5D	PA4D	PA3D	PA2D	PA1D	PA0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PCDR	PORTC drive register	008CH	PC7D	PC6D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PFDR	PORTF drive register	008FH	PF7D					PF2D	PF1D	PF0D
			R/W							
			1					1	1	1
			–					–	–	–
			Input /Output buffer drive register for standby mode					Input/Output buffer drive register for standby mode		

(1) I/O ports (10/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PGDR	PORTG drive register	0090H					PG3D	PG2D		
							R/W			
							1	1		
							–	–		
							Input/Output buffer drive register for standby mode			
PJDR	PORTJ drive register	0093H	PJ7D	PJ6D	PJ5D	PJ4D	PJ3D	PJ2D	PJ1D	PJ0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PKDR	PORTK drive register	0094H	PK7D	PK6D	PK5D	PK4D	PK3D	PK2D	PK1D	PK0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PLDR	PORTL drive register	0095H	PL7D	PL6D	PL5D	PL4D	PL3D	PL2D	PL1D	PL0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PMDR	PORTM drive register	0096H	PM7D					PM2D	PM1D	
			R/W					R/W		
			1					1	1	
			–					–	–	
			Input/Output buffer drive register for standby mode							
PNDR	PORTN drive register	0097H	PN7D	PN6D	PN5D	PN4D	PN3D	PN2D	PN1D	PN0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							
PPDR	PORTP drive register	0098H		PP6D	PP5D	PP4D	PP3D			
				R/W						
				1	1	1	1			
				–	–	–	–			
				Input/Output buffer drive register for standby mode						
PRDR	PORTR drive register	0099H					PR3D	PR2D	PR1D	PR0D
							R/W			
							1	1	1	1
							–	–	–	–
							Input/Output buffer drive register for standby mode			
PTDR	PORTT drive register	009BH	PT7D	PT6D	PT5D	PT4D	PT3D	PT2D	PT1D	PT0D
			R/W							
			1	1	1	1	1	1	1	1
			–	–	–	–	–	–	–	–
			Input/Output buffer drive register for standby mode							

(1) I/O ports (11/11)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PVDR	PORTV drive register	009DH	PV7D	PV6D						
			R/W							
			1	1						
			–	–						
			Input/Output buffer drive register for standby mode							
PXDR	PORTX drive register	009FH			PX5D	PX4D				
					R/W					
					1	1				
					–	–				
					Input/Output buffer drive register for standby mode					

(2) Interrupt control (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0	INT0 enable	00F0H	–				INT0			
			–	–	–	–	I0C	I0M2	I0M1	I0M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTE12	INT1 & INT2 enable	00D0H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	00D1H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	00D2H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE7	INT7 enable	00D3H	–				INT7			
			–	–	–	–	I7C	I7M2	I7M1	I7M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	00D4H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	00D5H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA45	INTTA4 & INTTA5 enable	00D6H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA67	INTTA6 & INTTA7 enable	00D7H	INTTA7 (TMRA7)				INTTA6 (TMRA6)			
			ITA7C	ITA7M2	ITA7M1	ITA7M0	ITA6C	ITA6M2	ITA6M1	ITA6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB0	INTTB00 & INTTB01 enable	00D8H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	INTTB10 & INTTB11 enable	00D9H	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES0	INTRX0 & INTTX0 enable	00DBH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	INTRX1 & INTTX1 enable	00DCH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESBIADM	INTSBI & INTADM enable	00E0H	INTADM				INTSBI			
			IADM0C	IADMM2	IADMM1	IADMM0	ISBI0C	ISBIM2	ISBIM1	ISBIM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

(2) Interrupt control (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTESPI	INTSPI enable	00E1H	INTSPITX				INTSPIRX			
			ISPITC	ISPITM2	ISPITM1	ISPITM0	ISPIRC	ISPIRM2	ISPIRM1	ISPIRM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEUSB	INTUSB enable	00E3H	–				INTUSB			
			–	–	–	–	IUSBC	IUSBM2	IUSBM1	IUSBM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEALM	INTALM enable	00E5H	–				INTALM			
			–	–	–	–	IALMC	IALMM2	IALMM1	IALMM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTERTC	INTRTC enable	00E8H	–				INTRTC			
			–	–	–	–	IRC	IRM2	IRM1	IRM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEKEY	INTKEY enable	00E9H	–				INTKEY			
			–	–	–	–	IKC	IKM2	IKM1	IKM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTELCD	INTLCD enable	00EAH	–				INTLCD			
			–	–	–	–	ILCD1C	ILCDM2	ILCDM1	ILCDM0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEI2S0	INTI2S0 enable	00EBH	–				INTI2S0			
			–	–	–	–	II2S0C	II2S0M2	II2S0M1	II2S0M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTENDFC	INTRSC & INTRDY enable	00ECH	INTRSC				INTRDY			
			IRSCC	IRSCM2	IRSCM1	IRSCM0	IRDYC	IRDYM2	IRDYM1	IRDYM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEP0	INTP0 enable	00EEH	–				INTP0			
			–	–	–	–	IP0C	IP0M2	IP0M1	IP0M0
			–	–			R	R/W		
			Always write “0”				0	0	0	0
INTEAD	INTAD & INTADHP enable	00EFH	INTADHP				INTAD			
			IADHPC	IADHPM2	IADHPM1	IADHPM0	IADC	IADM2	IADM1	IADM0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

(2) Interrupt control (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETC01 /INTEDMA01	INTTC0/INTDMA0 & INTTC1/INTDMA1 enable	00F1H	INTTC1/INTDMA1				INTTC0/INTDMA0			
			ITC1C /IDMA1C	ITC1M2 /IDMA1M2	ITC1M1 /IDMA1M1	ITC1M0 /IDMA1M0	ITC0C /IDMA0C	ITC0M2 /IDMA0M2	ITC0M1 /IDMA0M1	ITC0M0 /IDMA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23 /INTEDMA23	INTTC2/INTDMA2 & INTTC3/INTDMA3 enable	00F2H	INTTC3/INTDMA3				INTTC2/INTDMA2			
			ITC3C /IDMA3C	ITC3M2 /IDMA3M2	ITC3M1 /IDMA3M1	ITC3M0 /IDMA3M0	ITC2C /IDMA2C	ITC2M2 /IDMA2M2	ITC2M1 /IDMA2M1	ITC2M0 /IDMA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC45 /INTEDMA45	INTTC4/INTDMA4 & INTTC5/INTDMA5 enable	00F3H	INTTC5/INTDMA5				INTTC4/INTDMA4			
			ITC5C /IDMA5C	ITC5M2 /IDMA5M2	ITC5M1 /IDMA5M1	ITC5M0 /IDMA5M0	ITC4C /IDMA4C	ITC4M2 /IDMA4M2	ITC4M1 /IDMA4M1	ITC4M0 /IDMA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC67	INTTC6 & INTTC7 enable	00F4H	INTTC7 (DMA7)				INTTC6 (DMA6)			
			ITC7C	ITC7M2	ITC7M1	ITC7M0	ITC6C	ITC6M2	ITC6M1	ITC6M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
SIMC	SIO interrupt mode control	00F5H (Prohibit RMW)	–	–					IR1LE	IR0LE
			W	W					W	W
			0	0					1	1
			Always write "0"	Always write "0"					0: INTRX1 edge mode 1: INTRX1 level mode	0: INTRX0 edge mode 1: INTRX0 level mode
IIMC0	Interrupt input mode control 0	00F6H (Prohibit RMW)	I5EDGE	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	–
			W	W	W	W	W	W	R/W	R/W
			0	0	0	0	0	0	0	0
			INT5 edge 0: Rising 1: Falling	INT4 edge 0: Rising 1: Falling	INT3 edge 0: Rising 1: Falling	INT2 edge 0: Rising 1: Falling	INT1 edge 0: Rising 1: Falling	INT0 edge 0: Rising 1: Falling	0: INT0 edge mode 1:INT0 level mode	Always write "0"
INTWDT	INTWD enable	00F7H	–				INTWD			
			–	–	–	–	ITCWD	–	–	–
			–	–			R	–	–	–
			Always write "0"				0	–	–	–
INTCLR	Interrupt clear control	00F8H (Prohibit RMW)	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt vector							
IIMC1	Interrupt input mode control 1	00FAH (Prohibit RMW)							I7EDGE	I6EDGE
									W	W
									0	0
									INT7 edge 0: Rising 1: Falling	INT6 edge 0: Rising 1: Falling

(2) Interrupt control (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	0100H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	0101H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	0102H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	0103H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
DMA4V	DMA4 start vector	0104H			DMA4V5	DMA4V4	DMA4V3	DMA4V2	DMA4V1	DMA4V0
					R/W					
					0	0	0	0	0	0
					DMA4 start vector					
DMA5V	DMA5 start vector	0105H			DMA5V5	DMA5V4	DMA5V3	DMA5V2	DMA5V1	DMA5V0
					R/W					
					0	0	0	0	0	0
					DMA5 start vector					
DMA6V	DMA6 start vector	0106H			DMA6V5	DMA6V4	DMA6V3	DMA6V2	DMA6V1	DMA6V0
					R/W					
					0	0	0	0	0	0
					DMA6 start vector					
DMA7V	DMA7 start vector	0107H			DMA7V5	DMA7V4	DMA7V3	DMA7V2	DMA7V1	DMA7V0
					R/W					
					0	0	0	0	0	0
					DMA7 start vector					
DMAB	DMA burst	0108H	DBST7	DBST6	DBST5	DBST4	DBST3	DBST2	DBST1	DBST0
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request on burst mode							
DMAR	DMA request	0109H (Prohibit RMW)	DREQ7	DREQ6	DREQ5	DREQ5	DREQ4	DREQ3	DREQ2	DREQ1
			R/W							
			0	0	0	0	0	0	0	0
			1: DMA request in software							
DMASEL	Micro DMA/HDMA Select	010AH			DMASEL5	DMASEL4	DMASEL3	DMASEL2	DMASEL1	DMASEL0
					R/W					
					0	0	0	0	0	0
					0:Micro DMA5 1:HDMA5	0: Micro DMA4 1:HDMA4	0: Micro DMA3 1:HDMA3	0: Micro DMA2 1:HDMA2	0: Micro DMA1 1:HDMA1	0: Micro DMA0 1:HDMA0

(3) Memory controller (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CSL	BLOCK0 CS/WAIT control register low	0140H	B0WW3	B0WW2	B0WW1	B0WW0	B0WR3	B0WR2	B0WR1	B0WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved			
B0CSH	BLOCK0 CS/WAIT control register high	0141H	B0E			B0REC	B0OM1	B0OM0	B0BUS1	B0BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
B1CSL	BLOCK1 CS/WAIT control register low	0144H	B1WW3	B1WW2	B1WW1	B1WW0	B1WR3	B1WR2	B1WR1	B1WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved			
B1CSH	BLOCK1 CS/WAIT control register high	0145H	B1E			B1REC	B1OM1	B1OM0	B1BUS1	B1BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
B2CSL	BLOCK2 CS/WAIT control register low	0148H	B2WW3	B2WW2	B2WW1	B2WW0	B2WR3	B2WR2	B2WR1	B2WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{WAIT}$ pin input mode Others: Reserved			
B2CSH	BLOCK2 CS/WAIT control register high	0149H	B2E	B2M		B2REC	B2OM1	B2OM0	B2BUS1	B2BUS0
			R/W			R/W				
			1	0		0	0	0	0	1
			CS select 0: Disable 1: Enable	0: 16 MB 1: Sets area		Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: SDRAM		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	

(3) Memory controller (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B3CSL	BLOCK3 CS/WAIT control register low	014CH	B3WW3	B3WW2	B3WW1	B3WW0	B3WR3	B3WR2	B3WR1	B3WR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved			
B3CSH	BLOCK3 CS/WAIT control register high	014DH	B3E			B3REC	B3OM1	B3OM0	B3BUS1	B3BUS0
			R/W			R/W				
			0			0	0	0	0	0
			CS select 0: Disable 1: Enable			Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	
BEXCSL	BLOCK EX CS/WAIT control register low	0158H	BEXWW3	BEXWW2	BEXWW1	BEXWW0	BEXWR3	BEXWR2	BEXWR1	BEXWR0
			R/W							
			0	0	1	0	0	0	1	0
			Write waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved				Read waits 0001: 0 waits 0010: 1 wait 0101: 2 waits 0110: 3 waits 0111: 4 waits 1000: 5 waits 1001: 6 waits 1010: 7 waits 1011: 8 waits 1100: 9 waits 1101: 10 waits 1110: 12 waits 1111: 16 waits 0100: 20 waits 0011: 6 states + $\overline{\text{WAIT}}$ pin input mode Others: Reserved			
BEXCSH	BLOCK EX CS/WAIT control register high	0159H				BEXREC	BEXOM1	BEXOM0	BEXBUS1	BEXBUS0
						R/W				
						0	0	0	0	0
						Dummy cycle 0:No insert 1: Insert	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 00: 8 bits 01: 16 bits 10: Reserved 11: Don't set	

(3) Memory controller (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MAMR0	Memory address mask register 0	0142H	M0V20	M0V19	M0V18	M0V17	M0V16	M0V15	M0V14-9	M0V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR0	Memory start address register 0	0143H	M0S23	M0S22	M0S21	M0S20	M0S19	M0S18	M0S17	M0S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR1	Memory address mask register 1	0146H	M1V21	M1V20	M1V19	M1V18	M1V17	M1V16	MV15-9	M1V8
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR1	Memory start address register 1	0147H	M1S23	M1S22	M1S21	M1S20	M1S19	M1S18	M1S17	M1S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR2	Memory address mask register 2	014AH	M2V22	M2V21	M2V20	M2V19	M2V18	M2V17	M2V16	M2V15
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR2	Memory start address register 2	014BH	M2S23	M2S22	M2S21	M2S20	M2S19	M2S18	M2S17	M2S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							
MAMR3	Memory address mask register 3	014EH	M3V22	M3V21	M3V20	M3V19	M3V18	M3V17	M3V16	M3V15
			R/W							
			1	1	1	1	1	1	1	1
			0: Compare enable 1: Compare disable							
MSAR3	Memory start address register 3	014FH	M3S23	M3S22	M3S21	M3S20	M3S19	M3S18	M3S17	M3S16
			R/W							
			1	1	1	1	1	1	1	1
			Set start address A23 to A16							

(3) Memory controller (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
PMEMCR	Page ROM control register	0166H				OPGE	OPWR1	OPWR0	PR1	PR0
						R/W				
						0	0	0	1	0
						ROM page access 0: Disable 1: Enable	Wait number on page 00: 1 CLK (n-1-1-1 mode) 01: 2 CLK (n-2-2-2 mode) 10: 3CLK (n-3-3-3 mode) 11: Reserved		Byte number in a page 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes	
CSTMGCR	Adjust for Timing of control signal	0168H			TACSEL1	TACSEL0			TAC1	TAC0
					R/W				R/W	
					0	0			0	0
					Select area to change timing 00: CS0 01: CS1 10: CS2 11: CS3				Select delay time(TAC) 00: 0 × 1/f_{SYS} 01: 1 × 1/f_{SYS} 10: 2 × 1/f_{SYS} 11: Reserved	
WRTMGCR	Adjust for Timing of control signal	0169H			TCWSEL1	TCWSEL0	TCWS1	TCWS0	TCWH1	TCWH0
					R/W					
					0	0	0	0	0	0
					Select area to change timing 00: CS0 01: CS1 10: CS2 11: CS3		Select delay time(TCWS) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}		Select delay time(TCWH) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}	
RDTMGCR0	Adjust for Timing of control signal	016AH	B1TCRS1	B1TCRS0	B1TCRH1	B1TCRH0	B0TCRS1	B0TCRS0	B0TCRH1	B0TCRH0
			R/W							
			0	0	0	0	0	0	0	0
			Select delay time(TCRS) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}		Select delay time(TCRH) 00: 0 × 1/f_{SYS} 01: 1 × 1/f_{SYS} 10: 2 × 1/f_{SYS} 11: 3 × 1/f_{SYS}		Select delay time(TCRS) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}		Select delay time(TCRH) 00: 0 × 1/f_{SYS} 01: 1 × 1/f_{SYS} 10: 2 × 1/f_{SYS} 11: 3 × 1/f_{SYS}	
RDTMGCR1	Adjust for Timing of control signal	016BH	B3TCRS1	B3TCRS0	B3TCRH1	B3TCRH0	B2TCRS1	B2TCRS0	B2TCRH1	B2TCRH0
			R/W							
			0	0	0	0	0	0	0	0
			Select delay time(TCRS) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}		Select delay time(TCRH) 00: 0 × 1/f_{SYS} 01: 1 × 1/f_{SYS} 10: 2 × 1/f_{SYS} 11: 3 × 1/f_{SYS}		Select delay time(TCRS) 00: 0.5 × 1/f_{SYS} 01: 1.5 × 1/f_{SYS} 10: 2.5 × 1/f_{SYS} 11: 3.5 × 1/f_{SYS}		Select delay time(TCRH) 00: 0 × 1/f_{SYS} 01: 1 × 1/f_{SYS} 10: 2 × 1/f_{SYS} 11: 3 × 1/f_{SYS}	
BROMCR	Boot Rom control register	016CH						CSDIS	ROMLESS	VACE
								R/W		
								1	0/1	1/0
								Nand-Flash Area CS Output 0: Enable 1: Disable	Boot ROM 0: Use 1: Bypass	Vector address 0: Disable 1: Enable
RAMCR	RAM control register	016DH								–
										R/W
										1
										Always write "1"

(4) TSI

<table>
<tr><th>Symbol</th><th>Name</th><th>Address</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="4">TSICR0</td><td rowspan="4">TSI control register0</td><td rowspan="4">01F0H</td><td>TSI7</td><td>INGE</td><td>PTST</td><td>TWIEN</td><td>PYEN</td><td>PXEN</td><td>MYEN</td><td>MXEN</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0: Disable
1: Enable</td><td>Input gate control of Port 96,97
0: Enable
1: Disable</td><td>Detection condition
0: no touch
1: touch</td><td>INT4 interrupt control
0: Disable
1: Enable</td><td>SPY
0: OFF
1: ON</td><td>SPX
0: OFF
1: ON</td><td>SMY
0: OFF
1: ON</td><td>SMX
0: OFF
1: ON</td></tr>
<tr><td rowspan="5">TSICR1</td><td rowspan="5">TSI control register1</td><td rowspan="5">01F1H</td><td>DBC7</td><td>DB1024</td><td>DB256</td><td>DB64</td><td>DB8</td><td>DB4</td><td>DB2</td><td>DB1</td></tr>
<tr><td colspan="8">R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td rowspan="2">0: Disable
1: Enable</td><td>1024</td><td>256</td><td>64</td><td>8</td><td>4</td><td>2</td><td>1</td></tr>
<tr><td colspan="7">Debounce time is set by the formula “(N*64-16) / fSYS”.
“N” is the number of bits between bit6 and bit0 which are set to “1”.</td></tr>
</table>

(5) SDRAM controller

Symbol	Name	Address	7	6	5	4	3	2	1	0
SDACR	SDRAM access control register	0250H	SRDS	–	SMUXW1	SMUXW0	SPRE			SMAC
			R/W							R/W
			1	0	0	0	0			0
			Read data shift function 0: Disable 1: Enable	Always write “0”	Address multiplex type 00: Type A (A9-) 01: Type B (A10-) 10: Type C (A11-) 11: Reserved		Read/Write commands 0: Without auto pre-charge 1: With auto precharge			SDRAM controller 0: Disable 1: Enable
SDCISR	SDRAM Command Interval Setting Register	0251H		STMRD	STWR	STRP	STRCD	STRC2	STRC1	STRC0
				R/W						
				1	1	1	1	1	0	0
				TMRD 0: 1 CLK 1: 2 CLK	TWR 0: 1 CLK 1: 2 CLK	TRP 0: 1 CLK 1: 2 CLK	TRCD 0: 1 CLK 1: 2 CLK	TRC 000: 1 CLK 100: 5 CLK 001: 2 CLK 101: 6 CLK 010: 3 CLK 110: 7 CLK 011: 4 CLK 111: 8 CLK		
SDRCR	SDRAM refresh control register	0252H	–			SSAE	SRS2	SRS1	SRS0	SRC
			R/W			R/W				
			0			1	0	0	0	0
			Always write “0”			Self Refresh auto exit function 0: Disable 1: Enable	Refresh interval 000: 47 states 100: 468 states 001: 78 states 101: 624 states 010: 156 states 110: 936 states 011: 312 states 111: 1248 states			Auto Refresh 0: Disable 1: Enable
SDCMM	SDRAM command register	0253H						SCMM2	SCMM1	SCMM0
								R/W		
								0	0	0
								Command issue 000: Don’t care 001: Initialization sequence a. Precharge All command b. Eight Auto Refresh commands c. Mode Register Set command 010: Precharge All command 100: Reserved 101: Self Refresh Entry command 110: Self Refresh Exit command Others: Reserved		
SDBLS	SDRAM HDRAM burst length register	0254H			SDBL5	SDBL4	SDBL3	SDBL2	SDBL1	SDBL0
					0	0	0	0	0	0
					For HDMA5	For HDMA4	For HDMA3	For HDMA2	For HDMA1	For HDMA0
					HDMA burst length 0: 1 Word Read / Single Write 1: Full Page Read / Burst Write					

(6) LCD controller (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDMODE0	LCD mode0 register	0280H	RAMTYPE1	RAMTYPE0	SCPW1	SCPW0	MODE3	MODE2	MODE1	MODE0
			R/W							
			0	0	1	1	0	0	0	0
			Display RAM 00: Internal RAM 01: External SRAM 10: SDRAM 11: Reserved		LD bus transfer speed SCPW2= 0 00: 2-clock 01: 4-clock 10: 8-clock 11: 16-clock SCPW2= 1 00: 6-clock 01: 12-clock 10: 24-clock 11: 48-clock		Mode setting 0000: Reserved 0001: SR (mono) 0010: SR (4Gray) 0011: Reserved 0100: SR (16Gray) 0101: SR (64Gray) 0110: STN (256 color) 0111: STN (4096 color) 1000: STN (64k color) 1001: Reserved 1010: TFT (256 color) 1011: TFT (4096 color) 1100: TFT (64k color) 1101: Reserved 1110: TFT (mono) 1111: Reserved			
LCD MODE1	LCD mode1 register	0281H	LDC2	LDC1	LDC0	LDINV	AUTOINV	INTMODE	FREDGE	SCPW2
			R/W						W	W
			0	0	0	0	0	0	0	0
			Data rotation function (Supported for 64K-color: 16bps only) 000: Normal 001: Horizontal flip 010: Vertical flip 011: Horizontal & vertical flip 100: 90-degree 101: Reserved 110: Reserved 111: Reserved			LD bus Inversion 0: Normal 1: Inversion	Auto bus inversion 0: Disable 1: enable (Valid only for TFT)	Interrupt selection 0:LLOAD 1:LVSYNC	FR edge 0: LHSYNC front edge 1:LHSYNC back edge	LD bus transfer speed 0: normal 1: 1/3
LCDDVM0	LCD divide frame0 register	0283H	FMP3	FMP2	FMP1	FMP0	FML3	FML2	FML1	FML0
			R/W							
			0	0	0	0	0	0	0	0
			LCP0 DVM (bits 3-0)				LHSYNC DVM (bits 3-0)			
LCDDVM1	LCD divide frame1 register	0288H	FMP7	FMP6	FMP5	FMP4	FML7	FML6	FML5	FML4
			R/W							
			0	0	0	0	0	0	0	0
			LCP0 DVM (bits 7-4)				LHSYNC DVM (bit 7-4)			
LCDSIZE	LCD size register	0284H	COM3	COM2	COM1	COM0	SEG3	SEG2	SEG1	SEG0
			R/W				R/W			
			0	0	0	0	0	0	0	0
			Common setting 0000: Reserved 0001: 64 0010: 96 0011: 120 0100: 128 0101: 160 0110: 200 0111: 240 1000: 320 1001: 480 1010: Reserved 1011: Reserved 1100: Reserved 1101: Reserved 1110: Reserved 1111: Reserved				Segment setting 0000: Reserved 0001: 64 0010: 128 0011: 160 0100: 240 0101: 320 0110: 480 0111: 640 1000: Reserved 1001: Reserved 1010: Reserved 1011: Reserved 1100: Reserved 1101: Reserved 1110: Reserved 1111: Reserved			
LCDCTL0	LCD control0 register	0285H	PIPE	ALL0	FRMON	–		DLS	LCP0OC	START
			R/W					R/W		
			0	0	0	0		0	0	0
			PIP function 0: Disable 1: Enable	Segment Data 0: Normal 1: Always output “0”	FR divide setting 0: Disable 1: Enable	Always write “0”		FR signal LCP0/Line selection 0:Line 1:LCP0	LCP0 0: Always output 1: At valid data only LLOAD width 0: At setting in register 1: At valid data only	LCDC operation 0: Stop 1: Start

(6) LCD controller (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDCTL1	LCD control1 register	0286H	LCP0P	LHSP	LVSP	LLDP			LVSW1	LVSW0
			R/W	R/W	R/W	R/W			R/W	R/W
			1	0	1	0			0	0
			LCP0 phase 0: Rising 1: Falling	LHSYNC phase 0: Rising 1: Falling	LVSYNC phase 0: Rising 1: Falling	LLOAD phase 0: Rising 1: Falling			LVSYNC enable time control 00: 1 clock of LHSYNC 01: 2 clocks of LHSYNC 10: 3 clocks of LHSYNC 11: Reserved	
LCDCTL2	LCD control2 register	0287H	LGOE2P	LGOE1P	LGOE0P					
			R/W							
			0	0	0					
			LGOE2 phase 0: Rising 1: Falling	LGOE1 phase 0: Rising 1: Falling	LGOE0 phase 0: Rising 1: Falling					
LCDHSP	LHSYNC Pulse register	028AH	LH7	LH6	LH5	LH4	LH3	LH2	LH1	LH0
			W							
			0	0	0	0	0	0	0	0
			LHSYNC period (bits 7-0)							
LCDHSP	LHSYNC Pulse register	028BH	LH15	LH14	LH13	LH12	LH11	LH10	LH9	LH8
			W							
			0	0	0	0	0	0	0	0
			LHSYNC period (bits 15-8)							
LCDVSP	LVSYNC Pulse register	028CH	LVP7	LVP6	LVP5	LVP4	LVP3	LVP2	LVP1	LVP0
			W							
			0	0	0	0	0	0	0	0
			LVSYNC period (bits 7-0)							
LCDVSP	LVSYNC Pulse register	028DH							LVP9	LVP8
									W	
									0	0
									LVSYNC period (bits 9-8)	
LCDPRVSP	LVSYNC Pre Pulse register	028EH		PLV6	PLV5	PLV4	PLV3	PLV2	PLV1	PLV0
				W						
				0	0	0	0	0	0	0
				Front dummy LVSYNC (bits 6-0)						
LCDHSDLY	LHSYNC Delay register	028FH		HSD6	HSD5	HSD4	HSD3	HSD2	HSD1	HSD0
				W						
				0	0	0	0	0	0	0
				LHSYNC delay (bits 6-0)						
LCDLDDLY	LLOAD Delay register	0290H	PDT	LDD6	LDD5	LDD4	LDD3	LDD2	LDD1	LDD0
			R/W	W						
			0	0	0	0	0	0	0	0
			Data output timing 0: Sync with LLOAD 1: 1 clock later than LLOAD	LLOAD delay (bits 6-0)						

(6) LCD controller (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDO0DLY	LGOE0 Delay register	0291H		OE0D6	OE0D5	OE0D4	OE0D3	OE0D2	OE0D1	OE0D0
				W						
				0	0	0	0	0	0	0
				OE0 delay (bits 6-0)						
LCDO1DLY	LGOE1 Delay register	0292H		OE1D6	OE1D5	OE1D4	OE1D3	OE1D2	OE1D1	OE1D0
				W						
				0	0	0	0	0	0	0
				OE1 delay (bits 6-0)						
LCDO2DLY	LGOE2 Delay register	0293H		OE2D6	OE2D5	OE2D4	OE2D3	OE2D2	OE2D1	OE2D0
				W						
				0	0	0	0	0	0	0
				OE2 delay (bits 6-0)						
LCDHSW	LHSYNC Width register	0294H	HSW7	HSW6	HSW5	HSW4	HSW3	HSW2	HSW1	HSW0
			W							
			0	0	0	0	0	0	0	0
			Setting bit7-0 for LHSYNC Width							
LCDLDW	LLOAD width register	0295H	LDW7	LDW6	LDW5	LDW4	LDW3	LDW2	LDW1	LDW0
			W							
			0	0	0	0	0	0	0	0
			LHSYNC width (bits 7-0)							
LCDHO0W	LGOE0 width register	0296H	O0W7	O0W6	O0W5	O0W4	O0W3	O0W2	O0W1	O0W0
			W							
			0	0	0	0	0	0	0	0
			LLOAD width (bits 7-0)							
LCDHO1W	LGOE1 width register	0297H	O1W7	O1W6	O1W5	O1W4	O1W3	O1W2	O1W1	O1W0
			W							
			0	0	0	0	0	0	0	0
			LGOE1 width (bits 7-0)							
LCDHO2W	LGOE2 width register	0298H	O2W7	O2W6	O2W5	O2W4	O2W3	O2W2	O2W1	O2W0
			W							
			0	0	0	0	0	0	0	0
			LGOE2 width (bits 7-0)							
LCDHWB8	Bit8,9 for signal width register	0299H	O2W9	O2W8	O1W9	O1W8	O0W8	LDW9	LDW8	HSW8
			W							
			0	0	0	0	0	0	0	0
			LGOE2 width (bits 9-8)		LGOE1 width (bits 9-8)		LGOE0 width (bit 8)	LLOAD width (bits 9-8)		LHSYNC width (bit 8)

(6) LCD controller (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LSAML	Start address register LCD main-L	02A0H	LMSA7	LMSA6	LMSA5	LMSA4	LMSA3	LMSA2	LMSA1	
			R/W							
			0	0	0	0	0	0	0	
			LCD main area start address (A7-A1)							
LSAMM	Start address register LCD main-M	02A1H	LMSA15	LMSA14	LMSA13	LMSA12	LMSA11	LMSA10	LMA9	LMSA8
			R/W							
			0	0	0	0	0	0	0	0
			LCD main area start address (A15-A8)							
LSAMH	Start address register LCD main-H	02A2H	LMSA23	LMSA22	LMSA21	LMSA20	LMSA19	LMSA18	LMSA17	LMSA16
			R/W							
			0	1	0	0	0	0	0	0
			LCD main area start address (A23-A16)							
LSASL	Start address register LCD sub-L	02A4H	LSSA7	LSSA6	LSSA5	LSSA4	LSSA3	LSSA2	LSSA1	
			R/W							
			0	0	0	0	0	0	0	
			LCD sub area start address (A7-A1)							
LSASM	Start address register LCD sub -M	02A5H	LSSA15	LSSA14	LSSA13	LSSA12	LSSA11	LSSA10	LSSA9	LSSA8
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area start address (A15-A8)							
LSASH	Start address register LCD sub -H	02A6H	LSSA23	LSSA22	LSSA21	LSSA20	LSSA19	LSSA18	LSSA17	LSSA16
			R/W							
			0	1	0	0	0	0	0	0
			LCD sub area start address (A23-A16)							
LSAHX	Hot point register LCD sub -X	02A8H	SAHX7	SAHX6	SAHX5	SAHX4	SAHX3	SAHX2	SAHX1	SAHX0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area HOT point (7-0)							
LSAHX	Hot point register LCD sub -X	02A9H							SAHX9	SAHX8
									R/W	
									0	0
									LCD sub area HOT point (9-8)	
LSAHY	Hot point register LCD sub -Y	02AAH	SAHY7	SAHY6	SAHY5	SAHY4	SAHY3	SAHY2	SAHY1	SAHY0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area HOT point (7-0)							
LSAHY	Hot point register LCD sub -Y	02ABH								SAHY8
										R/W
										0
										LCD sub area HOT point (8)
LSASS	Segment size register LCD sub	02ACH	SAS7	SAS6	SAS5	SAS4	SAS3	SAS2	SAS1	SAS0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area segment size (7-0)							
LSASS	Segment size register LCD sub	02ADH							SAS9	SAS8
									R/W	
									0	0
									LCD sub area segment size (9-8)	
LSACS	Common size register LCD sub	02AEH	SAC7	SAC6	SAC5	SAC4	SAC3	SAC2	SAC1	SAC0
			R/W							
			0	0	0	0	0	0	0	0
			LCD sub area common size (7-0)							
LSACS	Common size register LCD sub	02AFH								SAC8
										R/W
										0
										LCD sub area common size (8)

(7) PMC

Symbol	Name	Address	7	6	5	4	3	2	1	0
PMCCTL	PMC Control Register	02A0H	PCM_ON					–	WUTM1	WUTM0
			R/W					W	R/W	R/W
		System Reset State	0					0	0	0
		Hot Reset State	Data retained					–	Data retained	Data retained
			Power Cut Mode 0: Disable 1: Enable					Must be written as 0 Always read as "0	Warm-up time 00: 2^9 (15.625 ms) 01: 2^{10} (31.25 ms) 10: 2^{11} (62.5 ms) 11: 2^{12} (125 ms)	

(8) USB controller (1/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Descriptor RAM0	Descriptor RAM 0 register	0500H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM1	Descriptor RAM 1 register	0501H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM2	Descriptor RAM 2 register	0502H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM3	Descriptor RAM 3 register	0503H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
: :	: :	: :	: :							
Descriptor RAM381	Descriptor RAM 381 register	067DH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM382	Descriptor RAM 382 register	067EH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Descriptor RAM383	Descriptor RAM 383 register	067FH	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint0	Endpoint 0 register	0780H	EP0_DATA7	EP0_DATA6	EP0_DATA5	EP0_DATA4	EP0_DATA3	EP0_DATA2	EP0_DATA1	EP0_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint1	Endpoint 1 register	0781H	EP1_DATA7	EP1_DATA6	EP1_DATA5	EP1_DATA4	EP1_DATA3	EP1_DATA2	EP1_DATA1	EP1_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint2	Endpoint 2 register	0782H	EP2_DATA7	EP2_DATA6	EP2_DATA5	EP2_DATA4	EP2_DATA3	EP2_DATA2	EP2_DATA1	EP2_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Endpoint3	Endpoint 3 register	0783H	EP3_DATA7	EP3_DATA6	EP3_DATA5	EP3_DATA4	EP3_DATA3	EP3_DATA2	EP3_DATA1	EP3_DATA0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
EP1_MODE	Endpoint 1 mode register	0789H			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP2_MODE	Endpoint 2 mode register	078AH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0
EP3_MODE	Endpoint 3 mode register	078BH			Payload[2]	Payload[1]	Payload[0]	Mode[1]	Mode[0]	Direction
					R/W					
					0	0	0	0	0	0

(8) USB controller (2/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP0_STATUS	Endpoint 0 status register	0790H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP1_STATUS	Endpoint 1 status register	0791H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP2_STATUS	Endpoint 2 status register	0792H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP3_STATUS	Endpoint 3 status register	0793H		TOGGLE	SUSPEND	STATUS[2]	STATUS[1]	STATUS[0]	FIFO_DISABLE	STAGE_ERR
				R						
				0	0	1	1	1	0	0
EP0_SIZE_L_A	Endpoint 0 size register Low A	0798H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_A	Endpoint 0 size register Low A	0799H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP2_SIZE_L_A	Endpoint 2 size register Low A	079AH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP3_SIZE_L_A	Endpoint 3 size register Low A	079BH	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			1	0	0	0	1	0	0	0
EP1_SIZE_L_B	Endpoint 1 size register Low B	07A1H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP2_SIZE_L_B	Endpoint 2 size register Low B	07A2H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP3_SIZE_L_B	Endpoint 3 size register Low B	07A3H	PKT_ACTIVE	DATASIZE6	DATASIZE5	DATASIZE4	DATASIZE3	DATASIZE2	DATASIZE1	DATASIZE0
			R							
			0	0	0	0	1	0	0	0
EP1_SIZE_H_A	Endpoint 1 size register High A	07A9H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_A	Endpoint 2 size register High A	07AAH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_A	Endpoint 3 size register HighA	07ABH						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0

(8) USB controller (3/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
EP1_SIZE_H_B	Endpoint 1 size register High B	07B1H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP2_SIZE_H_B	Endpoint 2 size register High B	07B2H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
EP3_SIZE_H_B	Endpoint 0 size register High B	07B3H						DATASIZE9	DATASIZE8	DATASIZE7
								R		
								0	0	0
bmRequestType	bmRequest-Type register	07C0H	DIRECTION	REQ_TYPE1	REQ_TYPE0	RECIPIENT4	RECIPIENT3	RECIPIENT2	RECIPIENT1	RECIPIENT0
			R							
			0	0	0	0	0	0	0	0
bRequest	bRequest register	07C1H	REQUEST7	REQUEST6	REQUEST5	REQUEST4	REQUEST3	REQUEST2	REQUEST1	REQUEST0
			R							
			0	0	0	0	0	0	0	0
wValue_L	wValue register Low	07C2H	VALUE_L7	VALUE_L6	VALUE_L5	VALUE_L4	VALUE_L3	VALUE_L2	VALUE_L1	VALUE_L0
			R							
			0	0	0	0	0	0	0	0
wValue_H	wValue register High	07C3H	VALUE_H7	VALUE_H6	VALUE_H5	VALUE_H4	VALUE_H3	VALUE_H2	VALUE_H1	VALUE_H0
			R							
			0	0	0	0	0	0	0	0
wIndex_L	wIndex register Low	07C4H	INDEX_L7	INDEX_L6	INDEX_L5	INDEX_L4	INDEX_L3	INDEX_L2	INDEX_L1	INDEX_L0
			R							
			0	0	0	0	0	0	0	0
wIndex_H	wIndex register High	07C5H	INDEX_H7	INDEX_H6	INDEX_H5	INDEX_H4	INDEX_H3	INDEX_H2	INDEX_H1	INDEX_H0
			R							
			0	0	0	0	0	0	0	0
wLength_L	wLength register Low	07C6H	LENGTH_L7	LENGTH_L6	LENGTH_L5	LENGTH_L4	LENGTH_L3	LENGTH_L2	LENGTH_L1	LENGTH_L0
			R							
			0	0	0	0	0	0	0	0
wLength_H	wLength register High	07C7H	LENGTH_H7	LENGTH_H6	LENGTH_H5	LENGTH_H4	LENGTH_H3	LENGTH_H2	LENGTH_H1	LENGTH_H0
			R							
			0	0	0	0	0	0	0	0

(8) USB controller (4/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SetupReceived	SetupRec-eived register	07C8H	D7	D6	D5	D4	D3	D2	D1	D0
			W							
			0	0	0	0	0	0	0	0
Current_Config	Current_ Config register	07C9H	REMOTEWAKEUP		ALTERNATE[1]	ALTERNATE[0]	INTERFACE[1]	INTERFACE[0]	CONFIG[1]	CONFIG[0]
			R		R					
			0		0	0	0	0	0	0
Standard Request	Standard-Request register	07CAH	S_INTERFACE	G_INTERFACE	S_CONFIG	G_CONFIG	G_DESCRIPT	S_FEATURE	C_FEATURE	G_STATUS
			R							
			0	0	0	0	0	0	0	0
Request	Request register	07CBH		SOFT_RESET	G_PORT_STS	G_DEVICE_ID	VENDOR	CLASS	ExSTANDARD	STANDARD
				R						
				0	0	0	0	0	0	0
DATASET1	DATASET 1 register	07CCH	EP3_DSET_B	EP3_DSET_A	EP2_DSET_B	EP2_DSET_A	EP1_DSET_B	EP1_DSET_A		EP0_DSET_A
			R							R
			0	0	0	0	0	0		0
DATASET2	DATASET 2 register	07CDH	EP7_DSET_B	EP7_DSET_A	EP6_DSET_B	EP6_DSET_A	EP5_DSET_B	EP5_DSET_A	EP4_DSET_B	EP4_DSET_A
			R							
			0	0	0	0	0	0	0	0
USB_STATE	USB state register	07CEH						Configured	Addressed	Default
								R/W	R	
								0	0	1
EOP	EOP register	07CFH	EP7_EOPB	EP6_EOPB	EP5_EOPB	EP4_EOPB	EP3_EOPB	EP2_EOPB	EP1_EOPB	EP0_EOPB
			W							
			1	1	1	1	1	1	1	1
COMMAND	Command register	07D0H		EP[2]	EP[1]	EP[0]	Command[3]	Command[2]	Command[1]	Command[0]
				W						
				0	0	0	0	0	0	0
EPx_SINGLE1	Endpoint 1 single register	07D1H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_SINGLE	EP2_SINGLE	EP1_SINGLE	
			R/W				R/W			
			0	0	0		0	0	0	
EPx_BCS1	Endpoint 1 BCS register	07D3H	EP3_SELECT	EP2_SELECT	EP1_SELECT		EP3_BCS	EP2_BCS	EP1_BCS	
			R/W				R/W			
			0	0	0		0	0	0	
INT_Control	Interrupt control register	07D6H								Status_nak
										R/W
										0
Standard Request Mode	Standard Request mode register	07D8H	S_Interface	G_Interface	S_Config	G_Config	G_Descript	S_Feature	C_Feature	G_Status
			R/W							
			0	0	0	0	0	0	0	0
Request Mode	Request mode register	07D9H		Soft_Reset	G_Port_Sts	G_DeviceId				
				R/W						
				0	0	0				

(8) USB controller (5/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
Port Status	Port status register	07E0H	Reserved7	Reserved6	PaperError	Select	NotError	Reserved2	Reserved1	Reserved0
			W							
			0	0	0	1	1	0	0	0
FRAME_L	Frame register Low	07E1H	–	T[6]	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
			R							
			0	0	0	0	0	0	0	0
FRAME_H	Frame register H	07E2H	T[10]	T[9]	T[8]	T[7]		CREATE	FRAME_STS1	FRAME_STS0
			R					R		
			0	0	0	0		0	1	0
ADDRESS	Address register	07E3H		A6	A5	A4	A3	A2	A1	A0
				R						
				0	0	0	0	0	0	0
USBREADY	USB ready register	07E6H								USBREADY
										R/W
										0
Set Descriptor STALL	Set-Descriptor stall register	07E8H								S_D_STALL
										W
										0
USBINTFR1	USB interrupt flag register 1	07F0H (Prohibit RMW)	INT_URST_STR	INT_URST_END	INT_SUS	INT_RESUME	INT_CLKSTOP	INT_CLKON		
			R/W							
			0	0	0	0	0	0		
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							
USBINTFR2	USB interrupt flag register 2	07F1H (Prohibit RMW)	EP1_FULL_A	EP1_Empty_A	EP1_FULL_B	EP1_Empty_B	EP2_FULL_A	EP2_Empty_A	EP2_FULL_B	EP2_Empty_B
			R/W							
			0	0	0	0	0	0	0	0
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							
USBINTFR3	USB interrupt flag register 3	07F2H (Prohibit RMW)	EP3_FULL_A	EP3_Empty_A	EP3_FULL_B	EP3_Empty_B				
			R/W							
			0	0	0	0				
			When read 0:Not generate interrupt 1:Generate interrupt When write 0: Clear flag 1: –							
USBINTFR4	USB interrupt flag register 4	07F3H (Prohibit RMW)	INT_SETUP	INT_EP0	INT_STAS	INT_STASN	INT_EP1N	INT_EP2N	INT_EP3N	
			R/W							
			0	0	0	0	0	0	0	
			When read 0: Not generate interrupt 1: Generate interrupt When write 0: Clear flag 1: –							

(8) USB controller (6/6)

Symbol	Name	Address	7	6	5	4	3	2	1	0
USBINTMR1	USB interrupt mask register 1	07F4H	MSK_URST_STR	MSK_URST_END	MSK_SUS	MSK_RESUME	MSK_CLKSTOP	MSK_CLKON		
			R/W							
			1	1	1	1	1	1		
			0: Be not masked 1: Be masked							
USBINTMR2	USB interrupt mask register 2	07F5H	EP1_MSK_FA	EP1_MSK_EA	EP1_MSK_FB	EP1_MSK_EB	EP2_MSK_FA	EP2_MSK_EA	EP2_MSK_FB	EP2_MSK_EB
			R/W							
			1	1	1	1	1	1	1	1
			0: Be not masked 1: Be masked							
USBINTMR3	USB interrupt mask register 3	07F6H	EP3_MSK_FA	EP3_MSK_EA						
			R/W							
			1	1						
			0: Be not masked 1: Be masked							
USBINTMR4	USB interrupt mask register 4	07F7H	MSK_SETUP	MSK_EP0	MSK_STAS	MSK_STASN	MSK_EP1N	MSK_EP2N	MSK_EP3N	
			R/W							
			1	1	1	1	1	1	1	
			0: Be not masked 1: Be masked							
USBCR1	USB control register 1	07F8H	TRNS_USE	WAKEUP					SPEED	USBCLKE
			R/W						R/W	
			0	0					1	0
			Transceiver 0:disable 1:enble	Wake up 0: – 1:Start						

(9) SPIC (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SPIMD	SPI Mode Setting register	0820H (Prohibit RMW)	SWRST	XEN				CLKSEL2	CLKSEL1	CLKSEL0
			W	R/W				R/W		
			0	0				1	0	0
			Software reset 0: Don't care 1: Reset	SYSCK 0: Disable 1: Enable				Select Baud Rate 000:Reserved 100: $f_{SYS}/8$ 001: $f_{SYS}/2$ 101: $f_{SYS}/16$ 010: $f_{SYS}/3$ 110: $f_{SYS}/64$ 011: $f_{SYS}/4$ 111: $f_{SYS}/256$		
		0821H (Prohibit RMW)	LOOPBACK	MSB1ST	DOSTAT		TCPOL	RCPOL	TDINV	RDINV
			R/W				R/W			
			0	1	1		0	0	0	0
			LOOPBACK Test mode 0:Disbale 1:Enable	Start bit for Transmit / Receive 0:LSB 1:MSB	SPDO pin state (no transmit) 0:Fixed to "0" 1:Fixed to "1"		Synchronous clock edge during transmitting 0: Falling 1: Rising	Synchronous clock edge during receiving 0: Falling 1: Rising	Invert data During transmitting 0: Disable 1: Enable	Invert data During receiving 0: Disable 1: Enable
SPICT	SPI Control register	0822H	CEN	SPCS_B	UNIT16	TXMOD	TXE	FDPXE	RXMOD	RXE
			R/W							
			0	1	0	0	0	0	0	0
			Communicat-ion control 0: Disable 1: Enable	$\overline{SPCS}$ pin 0: Output "0" 1: Output "1"	Data length 0: 8bit 1: 16bit	Transmit mode 0: UNIT 1:Sequential	Transmit control 0: Disable 1: Enable	Alignment in Full duplex 0: Disable 1: Enable	Receive Mode 0: UNIT 1:Sequential	Receive control 0: Disable 1: Enable
		0823H	CRC16_7_B	CRCRX_TX_B	CRCRESET_B					
			R/W							
			0	0	0					
			CRC select 0: CRC7 1: CRC16	CRC data 0: Transmit 1: Receive	CRC calculate register 0:Reset 1:Release Reset					
SPIST	SPI Status register	0824H					TEMP		TEND	REND
							R		R	
							1		1	0
							Transmit FIFO Status 0: No space 1: Having space		Transmit Status 0: During transmis-sion or having transmis-sion data 1: Finish	Receive Status 0: During receiving or not having receiving data 1: Finish or not having space
		0825H								
SPIIE	SPI Interrupt enable register	082CH					TEMPIE	RFULIE	TENDIE	RENDIE
							R/W			
							0	0	0	0
							TEMP interrupt 0:Enable 1:Disable	RFUL interrupt 0:Enable 1:Disable	TEND interrupt 0:Enable 1:Disable	REND interrupt 0:Enable 1:Disable
		082DH								

(9) SPIC (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SPICR	SPI CRC register	0826H	CRCD7	CRCD6	CRCD5	CRCD4	CRCD3	CRCD2	CRCD1	CRCD0
			R							
			0	0	0	0	0	0	0	0
			CRC result register [7:0]							
		0827H	CRCD15	CRCD14	CRCD13	CRCD12	CRCD11	CRCD10	CRCD9	CRCD8
			R							
			0	0	0	0	0	0	0	0
			CRC result register [15:8]							
SPITD0	SPI Transmis -sion data0 register	0830H	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [7:0]							
		0831H	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [15:8]							
SPITD1	SPI Transmis -sion data1 register	0832H	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [7:0]							
		0833H	TXD15	TXD14	TXD13	TXD12	TXD11	TXD10	TXD9	TXD8
			R/W							
			0	0	0	0	0	0	0	0
			Transmit data register [15:8]							
SPIRD0	SPI receive data0 register	0834H	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
			R							
			0	0	0	0	0	0	0	0
			Receive data register [7:0]							
		0835H	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
			R							
			0	0	0	0	0	0	0	0
			Receive data register [15:8]							
SPIRD1	SPI receive data1 register	0836H	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
			R							
			0	0	0	0	0	0	0	0
			Receive data register [7:0]							
		0837H	RXD15	RXD14	RXD13	RXD12	RXD11	RXD10	RXD9	RXD8
			R							
			0	0	0	0	0	0	0	0
			Receive data register [15:8]							

(10) MMU (1/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALPX	LOCALX register for program	0880H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		0881H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALPY	LOCALY register for program	0882H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		0883H	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALPZ	LOCALZ register for program	0884H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		0885H	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

(10) MMU (2/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALLX	LOCALX register for LCD	0888H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		0889H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALLY	LOCALY register for LCD	088AH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		088BH	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALLZ	LOCALZ register for LCD	088CH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		088DH	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited			100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD			

(10) MMU (3/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALRX	LOCALX register for read	0890H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		0891H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALRY	LOCALY register for read	0892H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		0893H	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALRZ	LOCALZ register for read	0894H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		0895H	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

(10) MMU (4/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALWX	LOCALX register for write	0898H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		0899H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALWY	LOCALY register for write	089AH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		089BH	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALWZ	LOCALZ register for write	089CH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		089DH	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

(10) MMU (5/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALESX	LOCALX register for DMA source	08A0H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		08A1H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALESY	LOCALY register for DMA source	08A2H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		08A3H	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALESZ	LOCALZ register for DMA source	08A4H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3)							
		08A5H	LZE							Z8
			R/W							R/W
			0							0
			BANK for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to 111111111 CSZD						

(10) MMU (6/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALEDX	LOCALX register for DMA destination	08A8H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		08A9H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALEDY	LOCALY register for DMA destination	08AAH			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		08ABH	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALEDZ	LOCALZ register for DMA destination	08ACH	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		08ADH	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000~101111111 Setting prohibited 110000000~111111111 CSZD						

(10) MMU (7/8)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LOCALOSX	LOCALX register for DMA source	08B0H	X7	X6	X5	X4	X3	X2	X1	X0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-X area (Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)							
		08B1H	LXE							X8
			R/W							R/W
			0							0
			Bank for LOCAL-X 0: Disable 1: Enable	Specify the bank number for the LOCAL-X area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 011111111 CSXA 100000000 to 111111111 CSXB						
LOCALOSY	LOCALY register for DMA source	08B2H			Y5	Y4	Y3	Y2	Y1	Y0
					R/W					
					0	0	0	0	0	0
					Specify the bank number for the LOCAL-Y area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)					
		08B3H	LYE							
			R/W							
			0							
			Bank for LOCAL-Y 0: Disable 1: Enable							
LOCALOSZ	LOCALZ register for DMA source	08B4H	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
			R/W							
			0	0	0	0	0	0	0	0
			Specify the bank number for the LOCAL-Z area (Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)							
		08B5H	LZE							Z8
			R/W							R/W
			0							0
			Bank for LOCAL-Z 0: Disable 1: Enable	Specify the bank number for the LOCAL-Z area Settings of the X8 through X0 bits and their corresponding chip select signals 000000000 to 001111111 CSZA 010000000 to 011111111 Setting prohibited 100000000 to 101111111 Setting prohibited 110000000 to111111111 CSZD						

(10) MMU (8/8)

<table>
<tr><th>Symbol</th><th>Name</th><th>Address</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="8">LOCALODX</td><td rowspan="8">LOCALX register for DMA destination</td><td rowspan="4">08B8H</td><td>X7</td><td>X6</td><td>X5</td><td>X4</td><td>X3</td><td>X2</td><td>X1</td><td>X0</td></tr>
<tr><td colspan="8">R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="8">Specify the bank number for the LOCAL-X area
(Since bank 0 is overlapping with the COMMON area, this filed must not be specified as 0.)</td></tr>
<tr><td rowspan="4">08B9H</td><td>LXE</td><td></td><td></td><td></td><td></td><td></td><td></td><td>X8</td></tr>
<tr><td>R/W</td><td></td><td></td><td></td><td></td><td></td><td></td><td>R/W</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td></tr>
<tr><td>Bank for LOCAL-X
0: Disable
1: Enable</td><td colspan="7">Specify the bank number for the LOCAL-X area
Settings of the X8 through X0 bits and their corresponding chip select signals
000000000 to 011111111 CSXA
100000000 to 111111111 CSXB</td></tr>
<tr><td rowspan="8">LOCALODY</td><td rowspan="8">LOCALY register for DMA destination</td><td rowspan="4">08BAH</td><td></td><td></td><td>Y5</td><td>Y4</td><td>Y3</td><td>Y2</td><td>Y1</td><td>Y0</td></tr>
<tr><td></td><td></td><td colspan="6">R/W</td></tr>
<tr><td></td><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td></td><td></td><td colspan="6">Specify the bank number for the LOCAL-Y area
(Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)</td></tr>
<tr><td rowspan="4">08BBH</td><td>LYE</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>R/W</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>BANK for LOCAL-Y
0: Disable
1: Enable</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="8">LOCALODZ</td><td rowspan="8">LOCALZ register for DMA destination</td><td rowspan="4">08BCH</td><td>Z7</td><td>Z6</td><td>Z5</td><td>Z4</td><td>Z3</td><td>Z2</td><td>Z1</td><td>Z0</td></tr>
<tr><td colspan="8">R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="8">Specify the bank number for the LOCAL-Z area
(Since bank 3 is overlapping with the COMMON area, this filed must not be specified as 3.)</td></tr>
<tr><td rowspan="4">08BDH</td><td>LZE</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Z8</td></tr>
<tr><td>R/W</td><td></td><td></td><td></td><td></td><td></td><td></td><td>R/W</td></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td></tr>
<tr><td>Bank for LOCAL-Z
0: Disable
1: Enable</td><td colspan="7">Specify the bank number for the LOCAL-Z area
Settings of the X8 through X0 bits and their corresponding chip select signals
000000000 to 001111111 CSZA 100000000 to 101111111 Setting prohibited
010000000 to 011111111 Setting prohibited 110000000 to 111111111 CSZD</td></tr>
</table>

(11) NAND-Flash controller (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDFMCR0	NANDF Control0 Register	08C0H (Prohibit RMW)	WE	ALE	CLE	CE0	CE1	ECCE	BUSY	ECCRST
			R/W							
			0	0	0	0	0	0	0	0
			WE enable 0: Disable 1: Enable	ALE control 0: "L" out 1: "H" out	CLE control 0: "L" out 1: "H" out	$\overline{CE0}$ control 0: "H" out 1: "L" out	$\overline{CE1}$ control 0: "H" out 1: "L" out	ECC circuit control 0: Disable 1: Enable	NAND Flash state 1: Busy 0: Ready	ECC reset control 0: – 1: Reset *Always read as "0".
		08C1H (Prohibit RMW)	SPLW1	SPLW0	SPHW1	SPHW0	RSECCL	RSEDN	RSESTA	RSECGW
			R/W						W	R/W
			0	0	0	0	0	0	0	0
			Strobe pulse width (Low width of $\overline{NDRE}$, $\overline{NDWE}$) Inserted width = (f_{SYS}) × (set value)		Strobe pulse width (High width of $\overline{NDRE}$, $\overline{NDWE}$) Inserted width = (f_{SYS}) × (set value)		Reed-Solomon ECC latch 0: Disable 1: Enable	Reed-Solomon operation 0: Encode (Write) 1: Decode (Read)	Reed-Solomon error calculation start 0: – 1: Start *Always read as "0".	Reed-Solomon ECC generator write control 0: Disable 1: Enable
NDFMCR1	NANDF Control1 Register	08C2H	INTERDY	INTRSC				BUSW	ECCS	SYSCKE
			R/W	R/W				R/W	R/W	R/W
			0	0				0	0	0
			Ready interrupt 0: Disable 1: Enable	Reed-Solomon calculation end interrupt 0: Disable 1: Enable				Data bus width 0: 8-bit 1: 16-bit	ECC calculation 0:Hamming 1: Reed-Solomon	Clock control 0: Disable 1: Enable
		08C3H	STATE3	STATE2	STATE1	STATE0	SEER1	SEER0		
			R							
			0	0	0	0	Undefined	Undefined		
			Status read (See the table below.)							
NDECCRD0	NANDF Code ECC Register0	08C4H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08C5H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							
NDECCRD1	NANDF Code ECC Register1	08C6H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08C7H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							

(11) NAND-Flash controller (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDECCRD2	NANDF Code ECC Register2	08C8H	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08C9H	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							
NDECCRD3	NANDF Code ECC Register3	08CAH	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08CBH	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							
NDECCRD4	NANDF Code ECC Register4	08CCH	ECCD7	ECCD6	ECCD5	ECCD4	ECCD3	ECCD2	ECCD1	ECCD0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (7-0)							
		08CDH	ECCD15	ECCD14	ECCD13	ECCD12	ECCD11	ECCD10	ECCD9	ECCD8
			R							
			0	0	0	0	0	0	0	0
			NAND Flash ECC Register (15-8)							

(11) NAND-Flash controller (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDRSCA0	NANDF read solomon Result address Register0	08D0H	RS0A7	RS0A6	RS0A5	RS0A4	RS0A3	RS0A2	RS0A1	RS0A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D1H							RS0A9	RS0A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD0	NANDF read solomon Result data Register0	08D2H	RS0D7	RS0D6	RS0D5	RS0D4	RS0D3	RS0D2	RS0D1	RS0D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDRSCA1	NANDF read solomon Result address Register1	08D4H	RS1A7	RS1A6	RS1A5	RS1A4	RS1A3	RS1A2	RS1A1	RS1A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D5H							RS1A9	RS1A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD1	NANDF read solomon Result data Register1	08D6H	RS1D7	RS1D6	RS1D5	RS1D4	RS1D3	RS1D2	RS1D1	RS1D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDRSCA2	NANDF read solomon Result address Register2	08D8H	RS2A7	RS2A6	RS2A5	RS2A4	RS2A3	RS2A2	RS2A1	RS2A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08D9H							RS2A9	RS2A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD2	NANDF read solomon Result data Register2	08DAH	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							

(11) NAND-Flash controller (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
NDRSCA3	NANDF read solomon Result address Register3	08DCH	RS3A7	RS3A6	RS3A5	RS3A4	RS3A3	RS3A2	RS3A1	RS3A0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Address Register (7-0)							
		08DDH							RS3A9	RS3A8
									R	
									0	0
									NAND Flash Reed-Solomon Calculation Result Address Register (9-8)	
NDRSCD3	NANDF read solomon Result data Register3	08DEH	RS2D7	RS2D6	RS2D5	RS2D4	RS2D3	RS2D2	RS2D1	RS2D0
			R							
			0	0	0	0	0	0	0	0
			NAND Flash Reed-Solomon Calculation Result Data Register (7-0)							
NDFDTR0	NANDF Data Register0	1FF0H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (7-0)							
		1FF1H	D15	D14	D13	D12	D11	D10	D9	D8
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (15-8)							
NDFDTR1	NANDF Data Register1	1FF2H	D7	D6	D5	D4	D3	D2	D1	D0
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (7-0)							
		1FF3H	D15	D14	D13	D12	D11	D10	D9	D8
			R/W							
			Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
			NAND-Flash Data Register (15-8)							

(12) DMAC (1/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS0	DMA source address Register0	0900H	D0SA7	D0SA6	D0SA5	D0SA4	D0SA3	D0SA2	D0SA1	D0SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (7:0)							
		0901H	D0SA15	D0SA14	D0SA13	D0SA12	D0SA11	D0SA10	D0SA9	D0SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (15:8)							
		0902H	D0SA23	D0SA22	D0SA21	D0SA20	D0SA19	D0SA18	D0SA17	D0SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA0 (23:16)							
HDMAD0	DMA destination address Register0	0904H	D0DA7	D0DA6	D0DA5	D0DA4	D0DA3	D0DA2	D0DA1	D0DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (7:0)							
		0905H	D0DA15	D0DA14	D0DA13	D0DA12	D0DA11	D0DA10	D0DA9	D0DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (15:8)							
		0906H	D0DA23	D0DA22	D0DA21	D0DA20	D0DA19	D0DA18	D0DA17	D0DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA0 (23:16)							
HDMACA0	DMA Transfer count number A Register0	0908H	D0CA7	D0CA6	D0CA5	D0CA4	D0CA3	D0CA2	D0CA1	D0CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA0 (7:0)							
		0909H	D0CA15	D0CA14	D0CA13	D0CA12	D0CA11	D0CA10	D0CA9	D0CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA0 (15:8)							
HDMACB0	DMA Transfer count number B Register0	090AH	D0CB7	D0CB6	D0CB5	D0CB4	D0CB3	D0CB2	D0CB1	D0CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA0 (7:0)							
		090BH	D0CB15	D0CB14	D0CB13	D0CB12	D0CB11	D0CB10	D0CB9	D0CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA0 (15:8)							
HDMAM0	DMA transfer Mode Register0	090CH				D0M4	D0M3	D0M2	D0M1	D0M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (2/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS1	DMA source address Register1	0910H	D1SA7	D1SA6	D1SA5	D1SA4	D1SA3	D1SA2	D1SA1	D1SA0
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (7:0)							
		0911H	D1SA15	D1SA14	D1SA13	D1SA12	D1SA11	D1SA10	D1SA9	D1SA8
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (15:8)							
		0912H	D1SA23	D1SA22	D1SA21	D1SA20	D1SA19	D1SA18	D1SA17	D1SA16
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA1 (23:16)							
HDMAD1	DMA destination address Register1	0914H	D1DA7	D1DA6	D1DA5	D1DA4	D1DA3	D1DA2	D1DA1	D1DA0
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (7:0)							
		0915H	D1DA15	D1DA14	D1DA13	D1DA12	D1DA11	D1DA10	D1DA9	D1DA8
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (15:8)							
		0916H	D1DA23	D1DA22	D1DA21	D1DA20	D1DA19	D1DA18	D1DA17	D1DA16
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA1 (23:16)							
HDMACA1	DMA Transfer count number A Register1	0918H	D1CA7	D1CA6	D1CA5	D1CA4	D1CA3	D1CA2	D1CA1	D1CA0
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number A for DMA1 (7:0)							
		0919H	D1CA15	D1CA14	D1CA13	D1CA12	D1CA11	D1CA10	D1CA9	D1CA8
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number A for DMA1 (15:8)							
HDMACB1	DMA Transfer count number B Register1	091AH	D1CB7	D1CB6	D1CB5	D1CB4	D1CB3	D1CB2	D1CB1	D1CB0
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number B for DMA1 (7:0)							
		091BH	D0CB15	D0CB14	D0CB13	D0CB12	D0CB11	D0CB10	D0CB9	D0CB8
			R/W							
			0	0	0	0	0	0	0	0
			Set transfer-count-number B for DMA1 (15:8)							
HDMAM1	DMA transfer Mode Register1	091CH				D1M4	D1M3	D1M2	D1M1	D1M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (3/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS2	DMA source address Register2	0920H	D2SA7	D2SA6	D2SA5	D2SA4	D2SA3	D2SA2	D2SA1	D2SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (7:0)							
		0921H	D2SA15	D2SA14	D2SA13	D2SA12	D2SA11	D2SA10	D2SA9	D2SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (15:8)							
		0922H	D2SA23	D2SA22	D2SA21	D2SA20	D2SA19	D2SA18	D2SA17	D2SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA2 (23:16)							
HDMAD2	DMA destination address Register2	0924H	D2DA7	D2DA6	D2DA5	D2DA4	D2DA3	D2DA2	D2DA1	D2DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (7:0)							
		0925H	D2DA15	D2DA14	D2DA13	D2DA12	D2DA11	D2DA10	D2DA9	D2DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (15:8)							
		0926H	D2DA23	D2DA22	D2DA21	D2DA20	D2DA19	D2DA18	D2DA17	D2DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA2 (23:16)							
HDMACA2	DMA Transfer count number A Register2	0928H	D2CA7	D2CA6	D2CA5	D2CA4	D2CA3	D2CA2	D2CA1	D2CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA2 (7:0)							
		0929H	D2CA15	D2CA14	D2CA13	D2CA12	D2CA11	D2CA10	D2CA9	D2CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA2 (15:8)							
HDMACB2	DMA Transfer count number B Register2	092AH	D2CB7	D2CB6	D2CB5	D2CB4	D2CB3	D2CB2	D2CB1	D2CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA2 (7:0)							
		092BH	D2CB15	D2CB14	D2CB13	D2CB12	D2CB11	D2CB10	D2CB9	D2CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA2 (15:8)							
HDMAM2	DMA transfer Mode Register2	092CH				D2M4	D2M3	D2M2	D2M1	D2M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (4/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS3	DMA source address Register3	0930H	D3SA7	D3SA6	D3SA5	D3SA4	D3SA3	D3SA2	D3SA1	D3SA0
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (7:0)							
		0931H	D3SA15	D3SA14	D3SA13	D3SA12	D3SA11	D3SA10	D3SA9	D3SA8
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (15:8)							
		0932H	D3SA23	D3SA22	D3SA21	D3SA20	D3SA19	D3SA18	D3SA17	D3SA16
			R/W							
			0	0	0	0	0	0	0	0
			Set source address for DMA3 (23:16)							
HDMAD3	DMA destination address Register3	0934H	D3DA7	D3DA6	D3DA5	D3DA4	D3DA3	D3DA2	D3DA1	D3DA0
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (7:0)							
		0935H	D3DA15	D3DA14	D3DA13	D3DA12	D3DA11	D3DA10	D3DA9	D3DA8
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (15:8)							
		0936H	D3DA23	D3DA22	D3DA21	D3DA20	D3DA19	D3DA18	D3DA17	D3DA16
			R/W							
			0	0	0	0	0	0	0	0
			Set destination address for DMA3 (23:16)							
HDMACA3	DMA Transfer count number A Register3	0938H	D3CA7	D3CA6	D3CA5	D3CA4	D3CA3	D3CA2	D3CA1	D3CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA3 (7:0)							
		0939H	D3CA15	D3CA14	D3CA13	D3CA12	D3CA11	D3CA10	D3CA9	D3CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA3 (15:8)							
HDMACB3	DMA Transfer count number B Register3	093AH	D3CB7	D3CB6	D3CB5	D3CB4	D3CB3	D3CB2	D3CB1	D3CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA3 (7:0)							
		093BH	D3CB15	D3CB14	D3CB13	D3CB12	D3CB11	D3CB10	D3CB9	D3CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA3 (15:8)							
HDMAM3	DMA transfer Mode Register3	093CH				D3M4	D3M3	D3M2	D3M1	D3M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (5/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS4	DMA source address Register4	0940H	D4SA7	D4SA6	D4SA5	D4SA4	D4SA3	D4SA2	D4SA1	D4SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (7:0)							
		0941H	D4SA15	D4SA14	D4SA13	D4SA12	D4SA11	D4SA10	D4SA9	D4SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (15:8)							
		0942H	D4SA23	D4SA22	D4SA21	D4SA20	D4SA19	D4SA18	D4SA17	D4SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA4 (23:16)							
HDMAD4	DMA destination address Register4	0944H	D4DA7	D4DA6	D4DA5	D4DA4	D4DA3	D4DA2	D4DA1	D4DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (7:0)							
		0945H	D4DA15	D4DA14	D4DA13	D4DA12	D4DA11	D4DA10	D4DA9	D4DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (15:8)							
		0946H	D4DA23	D4DA22	D4DA21	D4DA20	D4DA19	D4DA18	D4DA17	D4DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA4 (23:16)							
HDMACA4	DMA Transfer count number A Register4	0948H	D4CA7	D4CA6	D4CA5	D4CA4	D4CA3	D4CA2	D4CA1	D4CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA4 (7:0)							
		0949H	D4CA15	D4CA14	D4CA13	D4CA12	D4CA11	D4CA10	D4CA9	D4CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA4 (15:8)							
HDMACB4	DMA Transfer count number B Register4	094AH	D4CB7	D4CB6	D4CB5	D4CB4	D4CB3	D4CB2	D4CB1	D4CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA4 (7:0)							
		094BH	D4CB15	D4CB14	D4CB13	D4CB12	D4CB11	D4CB10	D4CB9	D4CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA4 (15:8)							
HDMAM4	DMA transfer Mode Register4	094CH				D4M4	D4M3	D4M2	D4M1	D4M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (6/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAS5	DMA source address Register5	0950H	D5SA7	D5SA6	D5SA5	D5SA4	D5SA3	D5SA2	D5SA1	D5SA0
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (7:0)							
		0951H	D5SA15	D5SA14	D5SA13	D5SA12	D5SA11	D5SA10	D5SA9	D5SA8
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (15:8)							
		0952H	D5SA23	D5SA22	D5SA21	D5SA20	D5SA19	D5SA18	D5SA17	D5SA16
			R/W							
			0	0	0	0	0	0	0	0
			Source address for DMA5 (23:16)							
HDMAD5	DMA destination address Register5	0954H	D5DA7	D5DA6	D5DA5	D5DA4	D5DA3	D5DA2	D5DA1	D5DA0
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (7:0)							
		0955H	D5DA15	D5DA14	D5DA13	D5DA12	D5DA11	D5DA10	D5DA9	D5DA8
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (15:8)							
		0956H	D5DA23	D5DA22	D5DA21	D5DA20	D5DA19	D5DA18	D5DA17	D5DA16
			R/W							
			0	0	0	0	0	0	0	0
			Destination address for DMA5 (23:16)							
HDMACA5	DMA Transfer count number A Register5	0958H	D5CA7	D5CA6	D5CA5	D5CA4	D5CA3	D5CA2	D54CA1	D5CA0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA5 (7:0)							
		0959H	D5CA15	D5CA14	D5CA13	D5CA12	D5CA11	D5CA10	D5CA9	D5CA8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count A for DMA5 (15:8)							
HDMACB5	DMA Transfer count number B Register5	095AH	D5CB7	D5CB6	D5CB5	D5CB4	D5CB3	D5CB2	D5CB1	D5CB0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA5 (7:0)							
		095BH	D5CB15	D5CB14	D5CB13	D5CB12	D5CB11	D5CB10	D5CB9	D5CB8
			R/W							
			0	0	0	0	0	0	0	0
			Transfer count B for DMA5 (15:8)							
HDMAM5	DMA transfer Mode Register5	095CH				D5M4	D5M3	D5M2	D5M1	D5M0
						R/W				
						0	0	0	0	0
						DMA transfer mode 000: Destination INC (I/O → MEM) 001: Destination DEC (I/O → MEM) 010: Source INC (MEM → I/O) 011: Source DEC (MEM → I/O) 100: Source/destination INC (MEM → MEM) 101: Source/destination DEC (MEM → MEM) 110: Source/destination fixed (I/O→ I/O) 111: Reserved			Transfer data size 00: 1 byte 01: 2 bytes 10: 4 bytes 11: Reserved	

(12) DMAC (7/7)

Symbol	Name	Address	7	6	5	4	3	2	1	0
HDMAE	DMA enable Register	097EH			DMAE5	DMAE4	DMAE3	DMAE2	DMAE1	DMAE0
					R/W					
					0	0	0	0	0	0
					DMA channel operation 0: Disable 1: Enable					
HDMATR	DMA timer Register	097FH	DMATE	DMATR6	DMATR5	DMATR4	DMATR3	DMATR2	DMATR1	DMATR0
			R/W							
			0	0	0	0	0	0	0	0
			Timer operation 0: Disable 1: Enable	Maximum bus occupancy time setting The value to be set in <DMATR6:0> should be obtained by “Maximum bus occupancy time / (256/f_{SYS})”. “00H” cannot be set.						

(13) Clock gear, PLL

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register0	10E0H		XTEN	USBCLK1	USBCLK0		WUEF		PRCK
				R/W				R/W		R/W
				1	0	0		0		0
				Low -frequency oscillator circuit (fs) 0: Stop 1: Oscillation	Select the clock of USB(f_{USB}) 00: Disable 01: Reserved 10: X1USB 11: f_{PLLUSB}			Warm-up timer		Select Prescaler clock 0: f_{SYS}/2 1: f_{SYS}/8
SYSCR1	System clock control register1	10E1H						GEAR2	GEAR1	GEAR0
								R/W		
								1	0	0
								Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8	101: Reserved 110: Reserved 111: Reserved 100: fc/16	
SYSCR2	System clock control register2	10E2H	–	CKOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		
			R/W							
			0	0	1	0	1	1		
			Always write "0".	Select CLKOUT 0: f_{SYS} 1: fs	Warm-Up Timer 00: Reserved 01: 2^8/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			
EMCCR0	EMC control register0	10E3H	PROTECT				–	EXTIN	DRVOSCH	DRVOSCL
			R				R/W	R/W	R/W	R/W
			0				0	0	1	1
			Protect flag 0: OFF 1: ON				Always write "0".	1: External clock	fc oscillator drive ability 1: NORMAL 0: WEAK	fs oscillator drive ability 1: NORMAL 0: WEAK
EMCCR1	EMC control register1	10E4H	Switching the protect ON/OFF by write to following 1st-KEY,2nd-KEY 1st-KEY: EMCCR1=5AH,EMCCR2=A5H in succession write 2nd-KEY: EMCCR1=A5H,EMCCR2=5AH in succession write							
EMCCR2	EMC control register2	10E5H								
PLLCR0	PLL control register0	10E8H		FCSEL	LUPFG					
				R/W	R					
				0	0					
				Select fc clock 0: f_{OSCH} 1: f_{PLL}	Lock-up timer Status flag 0: not end 1: end					
PLLCR1	PLL control register1	10E9H	PLL0	PLL1	LUPSEL					PLLTIMES
			R/W							R/W
			0	0	0					0
			PLL0 for CPU 0: Off 1: On	PLL1 for USB 0: Off 1: On	Select stage of Lock up counter 0: 12 stage (for PLL0) 1:13 stage (for PLL1)					Select the number of PLL 0: ×12 1: ×16

(14) 8-bit timer (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA01RUN	TMRA01 RUN register	1100H	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA01 prescaler	Up counter (UC1)	Up counter (UC0)
								0: Stop and clear 1: Run (Count up)		
TA0REG	8-bit timer register 0	1102H (Prohibit RMW)	–							
			W							
			0							
TA1REG	8-bit timer register 1	1103H (Prohibit RMW)	–							
			W							
			0							
TA01MOD	TMRA01 MODE register	1104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA1 00: TA0TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA0 00: TA0IN pin 01: φT1 10: φT4 11: φT16	
TA1FFCR	TMRA1 Flip-Flop control register	1105H (Prohibit RMW)					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don’t care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1
TA23RUN	TMRA23 RUN register	1108H	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA23 prescaler	Up counter (UC3)	Up counter (UC2)
								0: Stop and clear 1: Run (Count up)		
TA2REG	8-bit timer register 2	110AH (Prohibit RMW)	–							
			W							
			0							
TA3REG	8-bit timer register 3	110BH (Prohibit RMW)	–							
			W							
			0							
TA23MOD	TMRA23 MODE register	110CH	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA3 00: TA2TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA2 00: Reserved 01: φT1 10: φT4 11: φT16	
TA3FFCR	TMRA3 Flip-Flop control register	110DH (Prohibit RMW)					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don’t care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

(14) 8-bit timer (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA45RUN	TMRA45 RUN register	1110H	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA45 prescaler	Up counter (UC5)	Up counter (UC4)
								0: Stop and clear 1: Run (Count up)		
TA4REG	8-bit timer register 4	1112H (Prohibit RMW)	–							
			W							
			0							
TA5REG	8-bit timer register 5	1113H (Prohibit RMW)	–							
			W							
			0							
TA45MOD	TMRA45 MODE register	1114H	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA4 00: 32KHz clock 01: ϕT1 10: ϕT4 11: ϕT16	
TA5FFCR	TMRA5 Flip-Flop control register	1115H (Prohibit RMW)					TA5FFC1	TA5FFC0	TA5FFIE	TA5FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1: TMRA5
TA67RUN	TMRA67 RUN register	1118H	TA6RDE				I2TA67	TA67PRUN	TA7RUN	TA6RUN
			R/W				R/W			
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA67 prescaler	Up counter (UC7)	Up counter (UC6)
								0: Stop and clear 1: Run (Count up)		
TA6REG	8-bit timer register 2	111AH (Prohibit RMW)	–							
			W							
			0							
TA7REG	8-bit timer register 3	111BH (Prohibit RMW)	–							
			W							
			0							
TA67MOD	TMRA67 MODE register	111CH	TA67M1	TA67M0	PWM61	PWM60	TA7CLK1	TA7CLK0	TA6CLK1	TA6CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Source clock for TMRA7 00: TA6TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA6 00: 32KHz clock 01: ϕT1 10: ϕT4 11: ϕT16	
TA7FFCR	TMRA7 Flip-Flop control register	111DH (Prohibit RMW)					TA7FFC1	TA7FFC0	TA7FFIE	TA7FFIS
							W		R/W	
							1	1	0	0
							00: Invert TA7FF 01: Set TA7FF 10: Clear TA7FF 11: Don't care		TA7FF control for inversion 0: Disable 1: Enable	TA7FF inversion select 0: TMRA6 1: TMRA7

(15) 16-bit timer (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	TMRB0 RUN register	1180H	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffer 0: disable 1: enable	Always write “0”.			IDLE2 0: Stop 1: Operate	TMRB0 prescaler		Up counter (UC10)
								0: Stop and clear 1: Run (Count up)		
TB0MOD	TMRB0 MODE register	1182H (Prohibit RMW)	–	–	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			Always write “00”.		Software capture control 0: Execute 1:Undefined	Capture timing 00: Disable INT6 occurs at rising edge 01: TB0IN0 ↑ INT6 occurs at rising edge 10: TB0IN0 ↑ TB0IN0 ↓ INT6 occurs at falling edge 11: TA1OUT ↑ TA1OUT ↓ INT6 occurs at rising edge		Control Up counter 0:Clear Disable 1:Clear Enable	TMRB1 source clock 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
TB0FFCR	TMRB0 Flip-Flop control register	1183H (Prohibit RMW)	–	–	TB0CT1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			Always write “11”. *Always read as “11”.		TB1FF0 inversion trigger 0: Disable trigger 1: Enable trigger				Control TB1FF0 00: Invert 01: Set 10: Clear 11: Don’t care * Always read as “11”.	
					When capture UC10 to TB0CP1H/L	When capture UC10 to TB0CP0H/L	When UC10 matches with TB0RG1H/L	When UC10 matches with TB0RG0H/L		
TB0RG0L	16 bit timer register 0 low	1188H (Prohibit RMW)	–							
			W							
			0							
TB0RG0H	16 bit timer register 0 high	1189H (Prohibit RMW)	–							
			W							
			0							
TB0RG1L	16 bit timer register low	118AH (Prohibit RMW)	–							
			W							
			0							
TB0RG1H	16 bit timer register 1 high	118BH (Prohibit RMW)	–							
			W							
			0							
TB0CP0L	Capture register 0 low	118CH	–							
			R							
			Undefined							
TB0CP0H	Capture register 0 high	118DH	–							
			R							
			Undefined							
TB0CP1L	Capture register 1 low	118EH	–							
			R							
			Undefined							
TB0CP1H	Capture register 1 high	118FH	–							
			R							
			Undefined							

(15) 16-bit timer (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB1RUN	TMRB1 RUN register	1190H	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffer 0: disable 1: enable	Always write “0”.			IDLE2 0: Stop 1: Operate	TMRB1 prescaler		Up counter (UC12)
								0: Stop and clear 1: Run (Count up)		
TB1MOD	TMRB1 MODE register	1192H (Prohibit RMW)	–	–	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			Always write “00”.		Software capture control 0: Execute 1: Undefined	Capture timing 00: Disable INT7 occurs at rising edge 01: TB1IN0 ↑ INT7 occurs at rising edge 10: TB1IN0 ↑ TB1IN0 ↓ INT7 occurs at falling edge 11: TA3OUT ↑ TA3OUT ↓ INT7 occurs at rising edge		Control Up counter 0:Clear Disable 1:Clear Enable	TMRB1 source clock 00: TB1IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
TB1RG0L	16 bit timer register 0 low	1198H (Prohibit RMW)	–							
			W							
			0							
TB1RG0H	16 bit timer register 0 high	1199H (Prohibit RMW)	–							
			W							
			0							
TB1RG1L	16 bit timer register low	119AH (Prohibit RMW)	–							
			W							
			0							
TB1RG1H	16 bit timer register 1 high	119BH (Prohibit RMW)	–							
			W							
			0							
TB1CP0L	Capture register 0 low	119CH	–							
			R							
			Undefined							
TB1CP0H	Capture register 0 high	119DH	–							
			R							
			Undefined							
TB1CP1L	Capture register 1 low	119EH	–							
			R							
			Undefined							
TB1CP1H	Capture register 1 high	119FH	–							
			R							
			Undefined							

(16) UART/Serial channels (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer register	1200H (Prohibit RMW)	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R (Receive) /W (Transmission)							
			Undefined							
SC0CR	Serial channel 0 control register	1201H (Prohibit RMW)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to 0 when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK0↑ 1: SCLK0↓	0:baud rate generator 1: SCLK0 pin input
SC0MOD0	Serial channel 0 mode 0 register	1202H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer data bit 8	0: CTS disable 1: CTS enable	Receive function 0:Receive disable 1:Receive enable	Wake up 0: Disable 1: Enable	00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{IO} 11: External clock (SCLK0 input)	
BR0CR	Serial channel 0 baud rate control register	1203H	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write "0".	(16−K) /16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency "N" setting 0~F			
BR0ADD	Serial channel 0 K setting register	1204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Sets frequency divisor "K" (1~F)			
SC0MOD1	Serial channel 0 mode 1 register	1205H	I2S0	FDPX0						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						
SIR0CR	IrDA 0 control register	1207H	PLSEL	RXSEL	TXEN	RXEN	SIR0WD3	SIR0WD2	SIR0WD1	SIR0WD0
			R/W							
			0	0	0	0	0	0	0	0
			Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0:"H" pulse 1: "L" pulse	Transmit 0: Disable 1: Enable	Receive 0: Disable 1: Enable	Select receive pulse width Set the valid SIRR×D pulse width for equal or more than 2x × (setting value + 1) + 100ns Can be set: 1~14 Can not be set: 0, 15			

UART/Serial channels (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial channel 1 buffer register	1208H (Prohibit RMW)	RB7 TB7	RB6 TB6	RB5 TB5	RB4 TB4	RB3 TB3	RB2 TB2	RB1 TB1	RB0 TB0
			R (Receive) /W (Transmission)							
			Undefined							
SC1CR	Serial channel 1 control register	1209H (Prohibit RMW)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to 0 when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK1↑ 1: SCLK1↓	0:baud rate generator 1: SCLK1 pin input
SC1MOD0	Serial channel 1 mode 0 register	120AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transfer data bit 8	0: CTS disable 1: CTS enable	Receive function 0:Receive disable 1:Receive enable	Wake up 0: Disable 1: Enable	00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{IO} 11: External clock (SCLK1 input)	
BR1CR	Serial channel 1 baud rate control register	120BH	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write "0".	(16–K) /16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Divided frequency "N" setting 0~F			
BR1ADD	Serial channel 1 K setting register	120CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Sets frequency divisor "K" (1~F)			
SC1MOD1	Serial channel 1 mode 1 register	120DH	I2S1	FDPX1						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						
SIR1CR	IrDA 1 control register	120FH	PLSEL	RXSEL	TXEN	RXEN	SIR1WD3	SIR1WD2	SIR1WD1	SIR1WD0
			R/W							
			0	0	0	0	0	0	0	0
			Select transmit pulse width 0: 3/16 1: 1/16	Receive data 0:"H" pulse 1: "L" pulse	Transmit 0: Disable 1: Enable	Receive 0: Disable 1: Enable	Select receive pulse width Set the valid SIRRxD pulse width for equal or more than 2x × (setting value + 1) + 100ns Can be set: 1~14 Can not be set: 0, 15			

(17) SBI

<table>
<tr><th>Symbol</th><th>Name</th><th>Address</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="4">SBICR1</td><td rowspan="4">Serial bus interface control register 1</td><td rowspan="4">1240H
(Prohibit RMW)</td><td>BC2</td><td>BC1</td><td>BC0</td><td>ACK</td><td>–</td><td>SCK2</td><td>SCK1</td><td>SCK0
/SWRMON</td></tr>
<tr><td colspan="3">R/W</td><td>R/W</td><td>R</td><td colspan="2">R/W</td><td>R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0/1</td></tr>
<tr><td colspan="3">Number of transfer bits
000: 8 001: 1 010: 2
011: 3 100: 4 101: 5
110: 6 111: 7</td><td>Acknowledge mode specification
0: Disable
1: Enable</td><td>Always read as “1”.</td><td colspan="3">Setting for the divisor value “n”
(When writing)
000: 4 001: 5 010: 6
011: 7 100: 8 101: 9
110: 10 111: Reserved</td></tr>
<tr><td rowspan="3">SBIDBR</td><td rowspan="3">SBI buffer register</td><td rowspan="3">1241H
(Prohibit RMW)</td><td>DB7</td><td>DB6</td><td>DB5</td><td>DB4</td><td>DB3</td><td>DB2</td><td>DB1</td><td>DB0</td></tr>
<tr><td colspan="8">R (receive)/W (Transmit)</td></tr>
<tr><td colspan="8">Undefined</td></tr>
<tr><td rowspan="4">I2CAR</td><td rowspan="4">I^2C BUS Address register</td><td rowspan="4">1242H
(Prohibit RMW)</td><td>SA6</td><td>SA5</td><td>SA4</td><td>SA3</td><td>SA2</td><td>SA1</td><td>SA0</td><td>ALS</td></tr>
<tr><td colspan="8">R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td colspan="7">Slave Address setting</td><td>Address recognition
0: Enable
1: Disable</td></tr>
<tr><td rowspan="4">SBISR
When read</td><td rowspan="4">Serial bus interface status register</td><td rowspan="5">1243H
(Prohibit RMW)</td><td>MST</td><td>TRX</td><td>BB</td><td>PIN</td><td>AL/SBIM1</td><td>AAS/SBIM0</td><td>AD0/
SWRST1</td><td>LRB/
SWRST0</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td rowspan="2">Master/ Slave status monitor
0:Slave
1:Master</td><td rowspan="2">Transmitter/ Receiver status monitor
0:Receiver
1:Transmit-ter</td><td>I^2C bus status monitor
0: Free
1: Busy</td><td>INTSBI request monitor
0: Request
1: Cancel</td><td>Arbitration lost detection monitor
0: –
1: Detected</td><td>Slave Address match detection monitor
0: Undetected
1: Detected</td><td>General call detection monitor
0: Undetected
1: Detected</td><td>Last receive bit monitor
0: “0”
1: “1”</td></tr>
<tr><td>SBICR2
When write</td><td>Serial bus interface control register 2</td><td>Start/Stop condition
0: Stop condition
1: Busy condition</td><td>Cancel INTSBI interrupt request
0:Don’t care
1:Cancel interrupt request</td><td colspan="2">Serial bus interface operation mode selection
00: Port mode
01: Reserved
10: I^2C bus mode
11: Reserved</td><td colspan="2">Software reset generate write “10” and “01”, then an internal reset signal is generated.</td></tr>
<tr><td rowspan="4">SBIBR0</td><td rowspan="4">Serial bus interface baud rate register 0</td><td rowspan="4">1244H
(Prohibit RMW)</td><td>–</td><td>I2SBI</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>W</td><td>R/W</td><td colspan="5">R</td><td>R/W</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>Always read “0”</td><td>IDLE2
0: Stop
1: Operate</td><td colspan="5">Always read as “1”.</td><td>Always write “0”.</td></tr>
<tr><td rowspan="4">SBICR0</td><td rowspan="4">Serial bus interface control register 0</td><td rowspan="4">1247H
(Prohibit RMW)</td><td>SBIEN</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>R/W</td><td colspan="7">R</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SBI operation
0: Disable
1: Enable</td><td colspan="7">Always read as “0”.</td></tr>
</table>

(18) AD converter (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADREG0L	AD Conversion Result register 0 low	12A0H	ADR01	ADR00					OVR0	ADR0RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN0 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREG0H	AD conversion result register 0 high	12A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN0 conversion result							
ADREG1L	AD conversion result register 1 low	12A2H	ADR11	ADR10					OVR1	ADR1RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN1 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREG1H	AD conversion result register 1 high	12A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN1 conversion result							
ADREG2L	AD conversion result register 2 low	12A4H	ADR21	ADR20					OVR2	ADR2RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN2 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREG2H	AD conversion result register 2 high	12A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN2 conversion result							
ADREG3L	AD conversion result register 3 low	12A6H	ADR31	ADR30					OVR3	ADR3RF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN3 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREG3H	AD conversion result register 3 high	12A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN3 conversion result							
ADREG4L	AD conversion result register 4 low	12A8H	ADR4	ADR4					OVR4	ADR4F
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN4 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREG4H	AD conversion result register 4high	12A9H	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN4 conversion result							
ADREG5L	AD conversion result register 5 low	12AAH	ADR5	ADR5					OVR5	ADR5F
			R						R	R
			0	0					0	0
			Store Lower 2 bits of AN5 AD conversion result						Overrun flag 0:No generate 1: Generate	AD conversion result store flag 1: Stored
ADREG5H	AD conversion result register 5 high	12ABH	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AN5 conversion result							

(18) AD converter (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADREGSPL	High priority Conversion Register SP low	12B0H	ADRSP1	ADRSP0					OVSRP	ADRSPRF
			R						R	R
			0	0					0	0
			Store Lower 2 bits of an AD conversion result						Overrun 0:No generate 1: Generate	AD conversion result store flag 1:Stored
ADREGSPH	High priority Conversion Register SP high	12B1H	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
			R							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result							
ADCM0REGL	AD Conversion Result Compare Criterion Register 0 Low	12B4H	ADR21	ADR20						
			R/W							
			0	0						
			Store Lower 2 bits of an AD conversion result compare criterion							
ADCM0REGH	AD Conversion Result Compare Criterion Register 0 High	12B5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R/W							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result compare criterion							
ADCM1REGL	AD Conversion Result Compare Criterion Register 1 Low	12B6H	ADR21	ADR20						
			R/W							
			0	0						
			Store Lower 2 bits of an AD conversion result compare criterion							
ADCM1REGH	AD Conversion Result Compare Criterion Register 1 High	12B7H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R/W							
			0	0	0	0	0	0	0	0
			Store Upper 8 bits of an AD conversion result compare criterion							
ADMOD0	AD mode control register 0	12B8H	EOS	BUSY		I2AD	ADS	HTRGE	TSEL1	TSEL0
			R			R/W				
			0	0		0	0	0	0	0
			Normal AD conversion end flag 0:During conversion sequence or before starting 1:Complete conversion sequence	Normal AD conversion BUSY Flag 0:Stop conversion 1:During conversion		AD conversion when IDLE2 mode 0: Stop 1: Operate	Start Normal AD conversion 0: Don't Care 1:Start AD conversion Always read as"0".	Normal AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB00 interrupt 01: Reserved 10: $\overline{\text{ADTRG}}$ 11: Reserved	

(18) AD converter (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD1	AD mode control register 1	12B9H	DACON	ADCH2	ADCH1	ADCH0	LAT	ITM	REPEAT	SCAN
			R/W							
			0	0	0	0	0	0	0	0
			DAC and VREF application control	Analog input channel select			Latency 0: No Wait 1:Start after reading conversion result store Register of last channel	Interrupt specification when conversion channel fixed repeat mode	Repeat mode specification 0:Single conversion 1:Repeat conversion	Scan mode specification 0: Channel fixed mode 1: Channel scan mode
ADMOD2	AD mode control register 2	12BAH	HEOS	HBUSY			HADS	HHTRGE	HTSEL1	HTSEL0
			R				R/W			
			0	0			0	0	0	0
			High-priority AD conversion sequence FLAG 0: During conversion sequence or before starting 1: Complete conversion sequence	High-priority AD conversion BUSY Flag 0:Stop conversion 1:During conversion			Start High-priority AD conversion 0: Don't Care 1: Start AD conversion Always read as"0".	High-priority AD conversion at Hard ware trigger 0: Disable 1: Enable	Select Hard ware trigger 00: INTTB10 interrupt 01: Reserved 10: $\overline{ADTRG}$ 11: I^{2}S Sampling Counter Output	
ADMOD3	AD mode control register 3	12BBH	–	HADCH2	HADCH1	HADCH0				–
			R/W							R/W
			0	0	0	0				0
			Always write "0".	High-priority analog input channel select						Always write "0".
ADMOD4	AD mode control register 4	12BCH	CMEN1	CMEN0	CMP1C	CMP0C	IRQEN1	IRQEN0	CMPINT1	CMPINT0
			R/W							
			0	0	0	0	0	0	0	0
			AD Monitor function1 0: Disable 1: Enable	AD Monitor function0 0: Disable 1: Enable	Generation condition of AD monitor function interrupt 1 0: less than 1: Greater than or Equal	Generation condition of AD monitor function interrupt 0 0: less than 1: Greater than or Equal	AD monitor function interrupt 1 0: Disable 1: Enable	AD monitor function interrupt 0 0: Disable 1: Enable	Status of AD monitor function interrupt 1 0: No generation 1:Generation	Status of AD monitor function interrupt 0 0: No generation 1: Generation
ADMOD5	AD mode control register 5	12BDH		CM1H2	CM1CH1	CM1CH0		CM0CH2	CM0CH1	CM0CH0
				R/W				R/W		
				0	0	0		0	0	0
				Select analog channel for AD monitor function 1 000: AN0 100: AN4 001: AN1 101: AN5 010: AN2 110: Reserved 011: AN3 111: Reserved				Select analog channel for AD monitor function 1 000: AN0 100: AN4 001: AN1 101: AN5 010: AN2 110: Reserved 011: AN3 111: Reserved		
ADCCLK	AD Conversion Clock Setting Register	12BFH					–	ADCLK2	ADCLK1	ADCLK0
							R/W	R/W	R/W	R/W
							0	0	0	0
							Always write "0"	Select clock for AD conversion 000: Reserved 100: $f_{IO}/4$ 001: $f_{IO}/1$ 101: $f_{IO}/5$ 010: $f_{IO}/2$ 110: $f_{IO}/6$ 011: $f_{IO}/3$ 111: $f_{IO}/7$		

(19) Watchdog timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT mode register	1300H	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
			R/W					R/W		
			1	0	0			0	0	0
			WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{IO}$ 01: $2^{17}/f_{IO}$ 10: $2^{19}/f_{IO}$ 11: $2^{21}/f_{IO}$				IDLE2 0: Stop 1: Operate	1:Internally connects WDT out to the reset pin	Always write "0".
WDCR	WDT control register	1301H (Prohibit RMW)	–							
			W							
			–							
			B1H: WDT disable code 4E: WDT clear code							

(20) RTC (Real-Time Clock)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SECR	Second register	1320H		SE6	SE5	SE4	SE3	SE2	SE1	SE0
				R/W						
				Undefined						
			"0" is read	40 sec.	20 sec.	10 sec.	8 sec.	4 sec.	2 sec.	1 sec.
MINR	Minute register	1321H		MI6	MI5	MI4	MI3	MI2	MI1	MI0
				R/W						
				Undefined						
			"0" is read	40 min.	20 min.	10 min.	8 min.	4 min.	2 min.	1 min.
HOURR	Hour register	1322H			HO5	HO4	HO3	HO2	HO1	HO0
					R/W					
					Undefined					
			"0" is read		20 hours (PM/AM)	10 hours	8 hours	4 hours	2 hours	1 hour
DAYR	Day register	1323H						WE2	WE1	WE0
								R/W		
								Undefined		
			"0" is read					W2	W1	W0
DATER	Date register	1324H			DA5	DA4	DA3	DA2	DA1	DA0
					R/W					
					Undefined					
			"0" is read		20 days	10 days	8 days	4 days	2 days	1 day
MONTHR	Month register	1325H				MO4	MO3	MO2	MO1	MO0
						R/W				
						Undefined				
		PAGE0	"0" is read			10 month	8month	4 month	2 month	1 month
		PAGE1	"0" is read							0:Indicator for 12 hours 1: Indicator for 24 hours
YEARR	Year register	1326H	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
			R/W							
			Undefined							
		PAGE0	80 years	40 years	20 years	10 years	8 years	4 years	2 years	1 year
		PAGE1	"0" is read						Leap year setting 00: Leap year 01: One year after 10: Two years after 11: Three years after	
PAGER	Page register	1327H (Prohibit RMW)	INTENA			ADJUST	ENATMR	ENAALM		PAGE
			R/W			W	R/W			R/W
			0			Undefined	Undefined			Undefined
			Interrupt 0: Disable 1: Enable	"0" is read		0: Don't care 1: Adjust	Clock 0: Disable 1: Enable	ALARM 0: Disable 1: Enable	"0" is read.	PAGE selection
RESTR	Reset register	1328H (Prohibit RMW)	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	–	–	–	–
			W							
			Undefined							
			1Hz 0: Enable 1: Disable	16Hz 0: Enable 1: Disable	1:Clock reset	1: Alarm reset	Always write "0"			

(21) Melody/alarm generator

Symbol	Name	Address	7	6	5	4	3	2	1	0
ALM	Alarm-pattern register	1330H	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
			R/W							
			0	0	0	0	0	0	0	0
			Alarm pattern setting							
MELALMC	Melody/ alarm control register	1331H	FC1	FC0	ALMINV	–	–	–	–	MELALM
			R/W							
			0	0	0	0	0	0	0	0
			Free run counter control 00: Hold 01: Restart 10: Clear and stop 11: Clear and start		Alarm frequency invert 1: Invert	Always write “0”.				Output frequency 0: Alarm 1: Melody
MELFL	Melody frequency L-register	1332H	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
			R/W							
			0	0	0	0	0	0	0	0
			Melody frequency set (Low 8bit)							
MELFH	Melody frequency H-register	1333H	MELON				ML11	ML10	ML9	ML8
			R/W				R/W			
			0				0	0	0	0
			Melody counter control 0: Stop and clear 1: Start				Melody frequency set (Upper 4 bits)			
ALMINT	Alarm interrupt enable register	1334H			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
					R/W					
					0	0	0	0	0	0
					Always write “0”.	1:INTALM4 (1Hz) enable	1:INTALM3 (2Hz) enable	1:INTALM2 (64Hz) enable	1:INTALM1 (512Hz) enable	1:INTALM0 (8192Hz) enable

(22) I²S

Symbol	Name	Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2S0BUF	I²S Transmission Buffer Register0	1800H (Prohibit RMW)	B015	B014	B013	B012	B011	B010	B009	B008	B007	B006	B005	B004	B003	B002	B001	B000
			W															
			Undefined															
			Transmission buffer register (FIFO)															
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			B031	B030	B09	B028	B027	B026	B025	B024	B023	B022	B021	B020	B019	B018	B017	B016
			W															
			Undefined															
			Transmission buffer register (FIFO)															
I2S0CTL	I²S Control Register0	1808H	TXE0		*CNTE0				DIR0		BIT0		DTFMT01		DTFMT00		SYSCKE0	
			R/W		R/W				R/W		R/W		R/W		R/W		R/W	
			0		0				0		0		0		0		0	
			Transmit 0: Stop 1: Start		Counter control 0: Clear 1: Start				Transmis-sion start BIT 0:MSB 1:LSB		Bit length 0: 8 bits 1:16 bits		Output format 00: I²S 10: Right 01: Left 11:Reserved				System clock 0:Disable 1:Enable	
		1809H	CLKS0						FSEL0		TEMP0		WLVL0		EDGE0		CLKE0	
			R/W						R/W		R		R/W		R/W		R/W	
			0						0		1		0		0		0	
			Source clock 0: f_{SYS} 1: f_{PLL}						Stereo /monaural 0: Stereo 1:Monaural		Condition of transmission FIFO 0: data 1: None data		WS level 0:low left 1:high left		Clock edge for data output 0:Falling 1:Rising		Clock enable (After trans-mission) 0:Operate 1:Stop	
I2S0C	I²S0 Divider Value Setting Register	180AH	CK07		CK06		CK05		CK04		CK03		CK02		CK01		CK00	
			R/W															
			0		0		0		0		0		0		0		0	
			Divider value for CK signal (8-bit counter)															
		180BH					WS05		WS04		WS03		WS02		WS01		WS00	
							R/W											
							0		0		0		0		0		0	
							Divider value for WS signal (6-bit counter)											

(23) MAC (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MACMA_LL	Data register Multiplier A-LL	1BE0H	MA7	MA6	MA5	MA4	MA3	MA2	MA1	MA0
			R/W							
			Undefined							
			Multiplier A data register [7:0]							
MACMA_LH	Data register Multiplier A-LH	1BE1H	MA15	MA14	MA13	MA12	MA11	MA10	MA9	MA8
			R/W							
			Undefined							
			Multiplier A data register [15:8]							
MACMA_HL	Data register Multiplier A-HL	1BE2H	MA23	MA22	MA21	MA20	MA19	MA18	MA17	MA16
			R/W							
			Undefined							
			Multiplier A data register [23:16]							
MACMA_HH	Data register Multiplier A-HH	1BE3H	MA31	MA30	MA29	MA28	MA27	MA26	MA25	MA24
			R/W							
			Undefined							
			Multiplier A data register [31:24]							
MACMB_LL	Data register Multiplier B-LL	1BE4H	MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0
			R/W							
			Undefined							
			Multiplier B data register [7:0]							
MACMB_LH	Data register Multiplier B-LH	1BE5H	MB15	MB14	MB13	MB12	MB11	MB10	MB9	MB8
			R/W							
			Undefined							
			Multiplier B data register [15:8]							
MACMB_HL	Data register Multiplier B-HL	1BE6H	MB23	MB22	MB21	MB20	MB19	MB18	MB17	MB16
			R/W							
			Undefined							
			Multiplier B data register [23:16]							
MACMB_HH	Data register Multiplier B-HH	1BE7H	MB31	MB30	MB29	MB28	MB27	MB26	MB25	MB24
			R/W							
			Undefined							
			Multiplier B data register [31:24]							
MACOR_LLL	Data register Multiply and Accumulate -LLL	1BE8H	OR7	OR6	OR5	OR4	OR3	OR2	OR1	OR0
			R/W							
			Undefined							
			Multiply and Accumulate data register [7:0]							
MACOR_LLH	Data register Multiply and Accumulate -LLH	1BE9H	OR15	OR14	OR13	OR12	OR11	OR10	OR9	OR8
			R/W							
			Undefined							
			Multiply and Accumulate data register [15:8]							
MACOR_LHL	Data register Multiply and Accumulate -LGL	1BEAH	OR23	OR22	OR21	OR20	OR19	OR18	OR17	OR16
			R/W							
			Undefined							
			Multiply and Accumulate data register [23:16]							
MACOR_LHH	Data register Multiply and Accumulate -LHH	1BEBH	OR31	OR30	OR29	OR28	OR27	OR26	OR25	OR24
			R/W							
			Undefined							
			Multiply and Accumulate data register [31:24]							

(23) MAC (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MACOR_HLL	Data register Multiply and Accumulate -HLL	1BECH	OR39	OR38	OR37	OR36	OR35	OR34	OR33	OR32
			R/W							
			Undefined							
			Multiply and Accumulate data register [39:32]							
MACOR_HLH	Data register Multiply and Accumulate -HLH	1BEDH	OR47	OR46	OR45	OR44	OR43	OR42	OR41	OR40
			R/W							
			Undefined							
			Multiply and Accumulate data register [47:40]							
MACOR_HHL	Data register Multiply and Accumulate -HHL	1BEEH	OR55	OR54	OR53	OR52	OR51	OR50	OR49	OR48
			R/W							
			Undefined							
			Multiply and Accumulate data register [55:48]							
MACOR_HHH	Data register Multiply and Accumulate -HHH	1BEFH	OR63	OR62	OR61	OR60	OR59	OR58	OR57	OR56
			R/W							
			Undefined							
			Multiply and Accumulate data register [63:56]							
MACCR	MAC Control Register	1BFCH	MOVF	MOPST	MSTTG2	MSTTG1	MSTTG0	MSGMD	MOPMD1	MOPMD0
			R/W	W	R/W			R/W	R/W	
			0	0	0	0	0	0	0	0
			Over flow flag 0:no over flow 1:generate over flow	Start calculation control 0:don't care 1: Start calculation	Select the trigger of start calculation 000: Write to MACMA[7:0] 001: Write to MACMB[7:0] 010: Write to MACMOR[7:0] 011: Write to MACMOR[39:32] 1xx: Write "1" to <MOPST>			Sign mode 0:Unsigned 1:Signed	Calculation Mode 00: 64 + 32 × 32 01: 64 − 32 × 32 10: 32 × 32 − 64 11: Reserved	

6. Notes and Restrictions

6.1 Notation

(1) The notation for built-in I/O registers is as follows: Register symbol <Bit symbol>

Example: TA01RUN<TA0RUN> denotes bit TA0RUN of register TA01RUN.

(2) Read-modify-write instructions (RMW)

An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

Example 1: SET 3, (TA01RUN); Set bit3 of TA01RUN.

Example 2: INC 1, (100H); Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900:

Exchange instruction

EX (mem), R

Arithmetic operations

ADD (mem), R/#	ADC (mem), R/#
SUB (mem), R/#	SBC (mem), R/#
INC #3, (mem)	DEC #3, (mem)

Logic operations

AND (mem), R/#	OR (mem), R/#
XOR (mem), R/#	

Bit manipulation operations

STCF #3/A, (mem)	RES #3, (mem)
SET #3, (mem)	CHG #3, (mem)
TSET #3, (mem)	

Rotate and shift operations

RLC (mem)	RRC (mem)
RL (mem)	RR (mem)
SLA (mem)	SRA (mem)
SLL (mem)	SRL (mem)
RLD (mem)	RRD (mem)

(3) f_{OSCH}, fc, f_{SYS}, f_{IO} and one state

The clock frequency input on pins X1 and 2 is referred to as f_{OSCH}. The clock selected by PLLCR0<FCSEL> is referred to as fc.

The clock selected by SYSCR1<GEAR2:0> is referred to as system clock f_{SYS}. The clock frequency given by f_{SYS} divided by 2 is referred to as f_{IO}.

One cycle of f_{SYS} is referred to as one state.

6.2 Notes

(1) AM0 and AM1 pins

These pins are connected to the V_{CC} (Power supply level) or the V_{SS} (Grand level) pin. Do not alter the level when the pin is active.

(2) Reserved address areas

The 144Kbyte area (022000H~045FFFH) and 16 bytes area (FFFFF0H ~ FFFFFFH) cannot be used since it is reserved for use as internal area. If using an emulator, an optional 64 Kbytes of the 16M bytes area is used for emulator control. Therefore, if using an emulator, this area cannot be used.

(3) Standby mode (IDLE1)

When the HALT instruction is executed in IDLE1 mode (in which only the oscillator operates), RTC (Real-time-clock) and MLD (Melody-alarm-generator) operate. When necessary, stop the circuit before the HALT instruction is executed.

(4) Warm-up timer

The warm-up timer operates when STOP mode is released, even if the system is using an external oscillator. As a result, a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

(5) Watchdog timer

The watchdog timer starts operation immediately after a reset is released. Disable the watchdog timer when it is not to be used.

(6) AD converter

The string resistor between the VREFH and VREFL pins can be cut by program so as to reduce power consumption. When STOP mode is used, disable the resistor using the program before the HALT instruction is executed.

(7) CPU (Micro DMA)

Only the “LDC cr, r” and “LDC r, cr” instructions can be used to access the control registers in the CPU (e.g., the transfer source address register (DMASn).).

(8) Undefined SFR

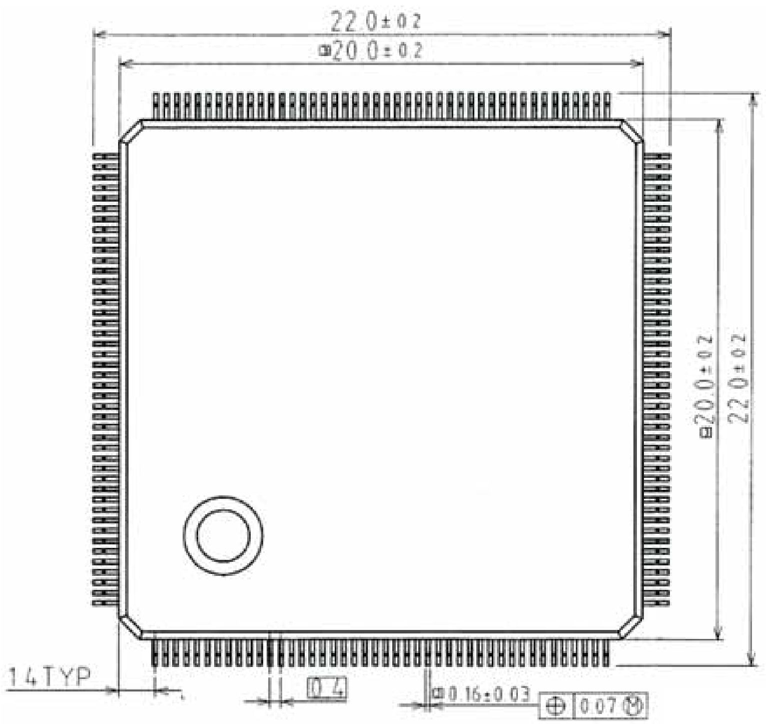
The value of an undefined bit in an SFR is undefined when read.

(9) POP SR instruction

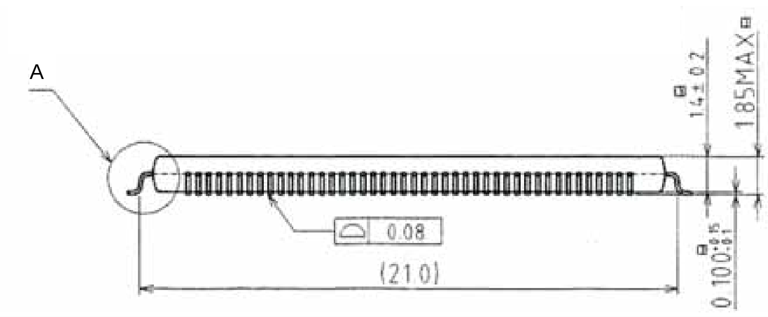
Please execute the POP SR instruction during DI condition.

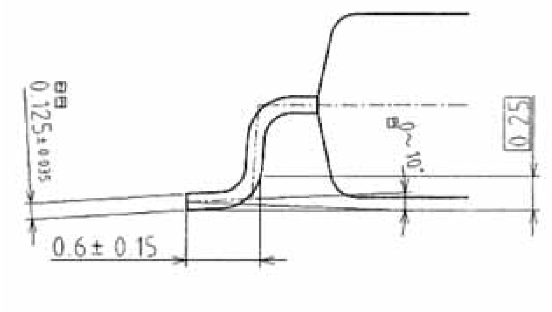
7. Package Dimensions

LQFP176-P-2020-0.40F



TOP VIEW





Detail view of A (25/1)

BOTTOM VIEW

RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively “TOSHIBA”), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively “Product”) without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA’s written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product’s quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before creating and producing designs and using, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the “TOSHIBA Semiconductor Reliability Handbook” and (b) the instructions for the application that Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS’ PRODUCT DESIGN OR APPLICATIONS.**
- Product is intended for use in general electronics applications (e.g., computers, personal equipment, office equipment, measuring equipment, industrial robots and home electronics appliances) or for specific applications as expressly stated in this document. Product is neither intended nor warranted for use in equipment or systems that require extraordinarily high levels of quality and/or reliability and/or a malfunction or failure of which may cause loss of human life, bodily injury, serious property damage or serious public impact (“Unintended Use”). Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. Do not use Product for Unintended Use unless specifically permitted in this document.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. TOSHIBA assumes no liability for damages or losses occurring as a result of noncompliance with applicable laws and regulations.

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