TOSHIBA

TOSHIBA Original CMOS 32-Bit Microcontroller

# TLCS-900/H1 Series

## TMP92CM22FG



TOSHIBA CORPORATION

Semiconductor Company

# Preface

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

CMOS 32-Bit Microcontrollers

# TMP92CM22FG

## 1. Outline and Device Characteristics

TMP92CM22 is high-speed advanced 32-bit microcontroller developed for controlling equipment, which processes mass data.

TMP92CM22FG is a microcontroller, which has a high-performance CPU (900/H1 CPU) and various built-in I/Os. TMP92CM22F is housed in a 100-pin flat package.

Device characteristics are as follows:

(1) CPU: 32-bit CPU (900/H1 CPU)
- Compatible with TLCS-900, 900/L, 900/L1, 900/H, and 900/H2's instruction code
- 16 Mbytes of linear address space
- General-purpose register and register banks
- Micro DMA: 8 channels (250 ns/4 bytes at $f_{SYS}$ = 20 MHz, best case)

(2) Minimum instruction execution time: 50 ns (at $f_{SYS}$ = 20 MHz)

(3) Internal memory
- Internal RAM: 32 Kbytes (32-bit 1-clock access, programmable)
- Internal ROM: None

(4) External memory expansion
- Expandable up to 16 Mbytes (Shared program/data area)
- Can simultaneously support 8-/16-bit width external data bus
  ···Dynamic data bus sizing
- Separate bus system

(5) Memory controller
- Chip select output: 4 channels

(6) 8-bit timers: 4 channels

(7) 16-bit timers: 2 channels

(8) General-purpose serial interface: 2 channels
- UART/synchronous mode
- IrDA

(9) Serial bus interface: 1 channel
- $I^2C$ bus mode
- Clock synchronous mode

(10) 10-bit AD converter: 8 channels

(11) Watchdog timer

(12) Interrupts: 41 interrupts
- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 25 internal interrupts: Seven selectable priority levels
- 7 external interrupts: Seven selectable priority levels (INT0 to INT5 and $\overline{NMI}$)
  (INT0 to INT3 selectable edge or level interrupt)

(13) Input/output ports: 50 pins (exclude Data bus 8-bit, Address bus 24-bit and $\overline{RD}$ pin)

(14) Standby function
- Three HALT modes: IDLE2 (Programmable), IDLE1, STOP

(15) Dual-clock controller
- PLL: fc = $f_{OSCH}$ × 4 (fc = 40 MHz at $f_{OSCH}$ = 10 MHz)
- Clock gear function: Select a high-frequency clock fc to fc/16

(16) Operating voltage
- DVCC = 3.0 V to 3.6 V (fc max = 40 MHz)

(17) Package
- 100-pin QFP: P-LQFP100-1414-0.50F

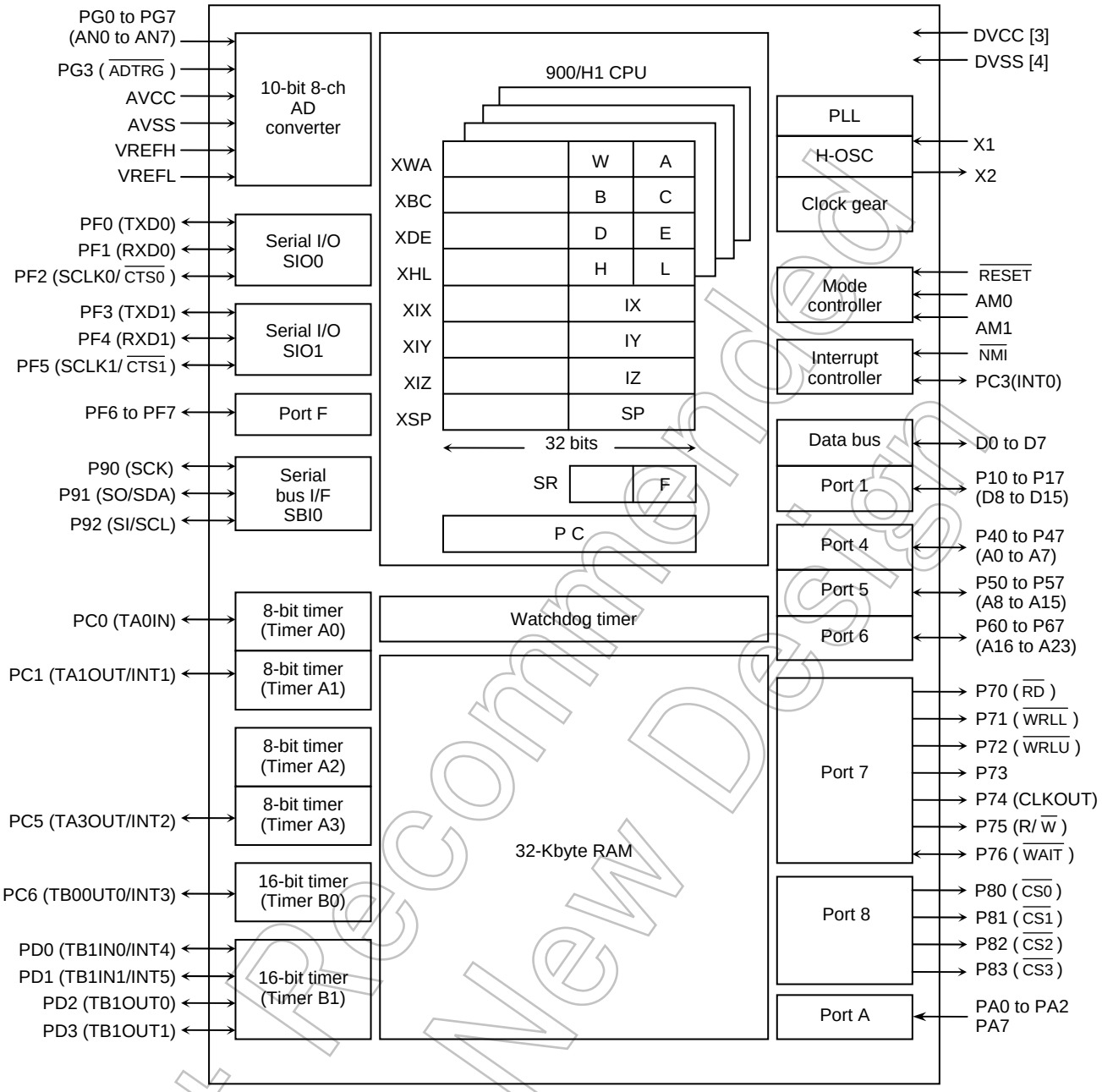



Figure 1.1 TMP92CM22 Block Diagram

# 2. Pin Assignment and Functions

The assignment of input/output pins for the TMP92CM22FG, their names and functions are as follows.

## 2.1 Pin Assignment

Figure 2.1.1 shows the pin assignment of the TMP92CM22FG.

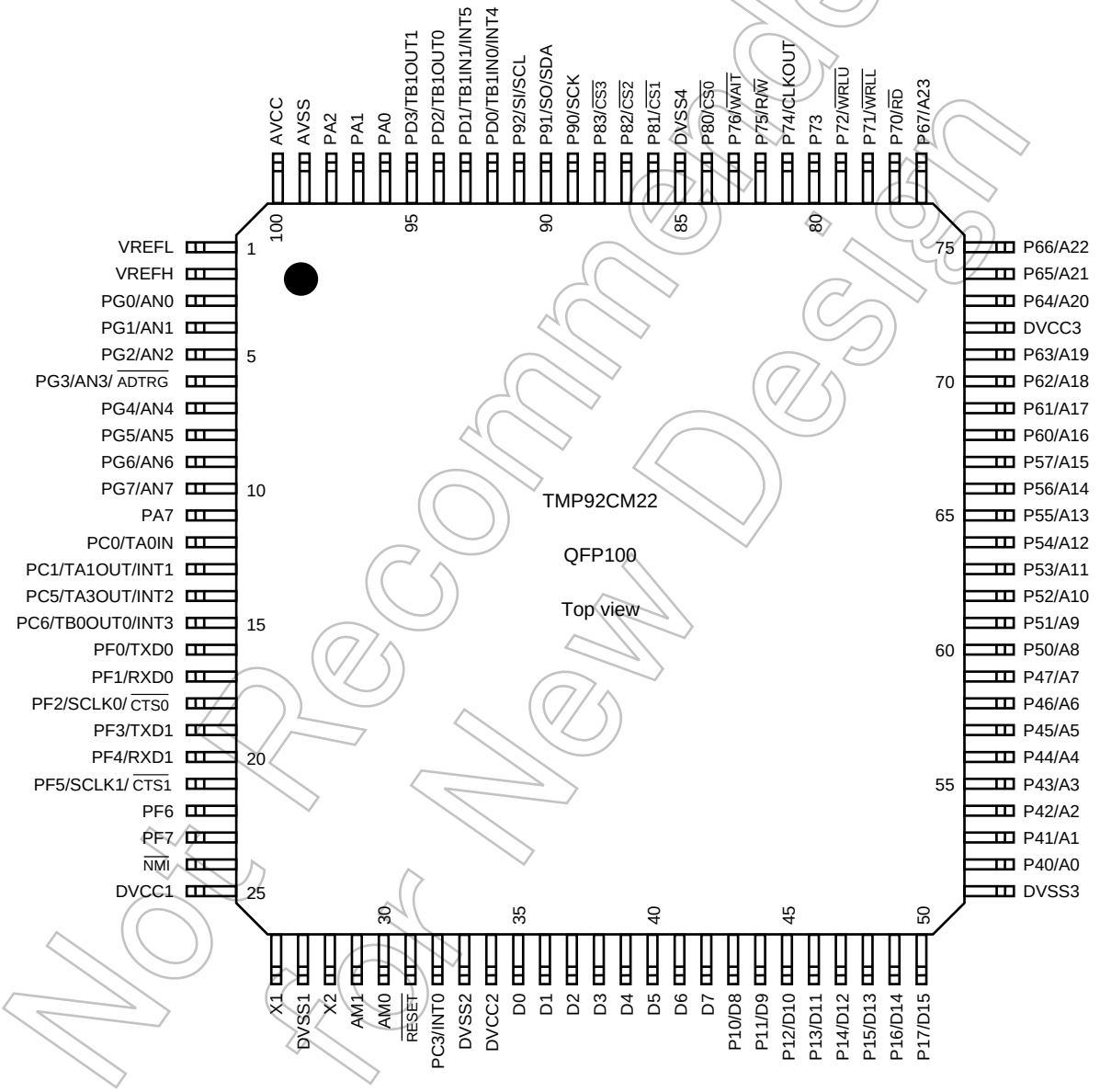





Figure 2.1.1 Pin Assignment Diagram (100-Pin QFP)

## 2.2 Pin Names and Functions

The following tables show the names and functions of the input/output pins.

Table 2.2.1 Pin Names and Functions (1/2)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| D0 to D7 | 8 | I/O | Data (Lower): Data bus D0 to D7. |
| P10 to P17 | 8 | I/O | Port 1: I/O port that allows I/O to be selected at the bit level. (when used to the external 8-bit bus.) |
| D8 to D15 | | I/O | Data: Data bus D8 to D15. |
| P40 to P47 | 8 | I/O | Port 4: I/O port. |
| A0 to A7 | | Output | Address: Address bus A0 to A7. |
| P50 to P57 | 8 | I/O | Port 5: I/O port. |
| A8 to A15 | | Output | Address: Address bus A8 to A15. |
| P60 to P67 | 8 | I/O | Port 6: I/O port. |
| A16 to A23 | | Output | Address: Address bus A16 to A23. |
| P70 | 1 | Output | Port 70: Output port. |
| $\overline{RD}$ | | Output | Read: Strobe signal for reading external memory. |
| P71 | 1 | Output | Port 71: Output port. |
| $\overline{WRLL}$ | | Output | Write: Strobe signal for writing data to pins D0 to D7. |
| P72 | 1 | Output | Port 72: Output port. |
| $\overline{WRLU}$ | | Output | Write: Strobe signal for writing data to pins D8 to D15. |
| P73 | 1 | Output | Port 73: Output port. |
| P74 | 1 | Output | Port 74: Output port. |
| CLKOUT | | Output | Clock: Output system clock. |
| P75 | 1 | Output | Port 75: Output port. |
| R/$\overline{W}$ | | Output | Read/write: This port is 1 when read and dummy cycle. This port is 0 when write cycle. |
| P76 | 1 | I/O | Port 76: I/O port. |
| $\overline{WAIT}$ | | Input | Wait: Pin used to request bus wait to CPU. |
| P80 | 1 | Output | Port 80: Output port. |
| $\overline{CS0}$ | | Output | Chip select 0: Outputs 0 when address is within specified address area. |
| P81 | 1 | Output | Port 81: Output port. |
| $\overline{CS1}$ | | Output | Chip select 1: Outputs 0 when address is within specified address area. |
| P82 | 1 | Output | Port 82: Output port. |
| $\overline{CS2}$ | | Output | Chip select 2: Outputs 0 when address is within specified address area. |
| P83 | 1 | Output | Port 83: Output port. |
| $\overline{CS3}$ | | Output | Chip select 3: Outputs 0 when address is within specified address area. |
| P90 | 1 | I/O | Port 90: I/O port. |
| SCK | | I/O | Serial bus interface clock I/O data at SIO mode. |
| P91 | 1 | I/O | Port 91: I/O port. |
| SO | | Output | Serial bus interface send data at SIO mode. |
| SDA | | I/O | Serial bus interface send/receive data at $I^2C$ mode. (Open-drain output mode by programmable.) |
| P92 | 1 | I/O | Port 92: I/O port. |
| SI | | Input | Serial bus interface receive data at SIO mode. |
| SCL | | I/O | Serial bus interface clock I/O data at $I^2C$ mode. (Open-drain output mode by programmable.) |
| PA0 to PA2, PA7 | 4 | Input | Port A0 to A2, A7: Input port (with pull-up resistor). |

Table 2.2.2 Pin Names and Functions (2/2)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| PC0<br>TA0IN | 1 | I/O<br>Input | Port C0: I/O port.<br>Timer input: 8-bit timer A0 input. |
| PC1<br>INT1<br>TA1OUT | 1 | I/O<br>Input<br>Output | Port C1: I/O port.<br>Interrupt request pin 1: Interrupt request pin with programmable level/rising edge/falling edge.<br>Timer output: 8-bit timer A0 or timer A1 output. |
| PC3<br>INT0 | 1 | I/O<br>Input | Port C3: I/O port.<br>Interrupt request pin 0: Interrupt request pin with programmable level/rising edge/falling edge. |
| PC5<br>INT2<br>TA3OUT | 1 | I/O<br>Input<br>Output | Port C5: I/O port.<br>Interrupt request pin 2: Interrupt request pin with programmable level/rising edge/falling edge.<br>Timer output: 8-bit timer A2 or timer A3 output. |
| PC6<br>INT3<br>TB0OUT0 | 1 | I/O<br>Input<br>Output | Port C6: I/O port.<br>Interrupt request pin 3: Interrupt request pin with programmable level/rising edge/falling edge.<br>Timer output: 16-bit timer B0 output. |
| PD0<br>INT4<br>TB1IN0 | 1 | I/O<br>Input<br>Input | Port D0: I/O port.<br>Interrupt request pin 4: Interrupt request pin with programmable rising edge/falling edge.<br>Timer input: 16-bit timer B1 input 0. |
| PD1<br>INT5<br>TB1IN1 | 1 | I/O<br>Input<br>Input | Port D1: I/O port.<br>Interrupt request pin 5: Interrupt request pin with programmable rising edge/falling edge.<br>Timer input: 16-bit timer B1 input 1. |
| PD2<br>TB1OUT0 | 1 | I/O<br>Output | Port D2: I/O port.<br>Timer output: 16-bit timer B1 output 0. |
| PD3<br>TB1OUT1 | 1 | I/O<br>Output | Port D3: I/O port.<br>Timer output: 16-bit timer B1 output 1. |
| PF0<br>TXD0 | 1 | I/O<br>Output | Port F0: I/O port.<br>Serial send data 0: (Open-drain output mode by programmable.) |
| PF1<br>RXD0 | 1 | I/O<br>Input | Port F1: I/O port.<br>Serial receive data 0. |
| PF2<br>SCLK0<br>$\overline{\text{CTS0}}$ | 1 | I/O<br>I/O<br>Input | Port F2: I/O port.<br>Serial 0 clock I/O.<br>Serial data send enable 0 (Clear to send). |
| PF3<br>TXD1 | 1 | I/O<br>Output | Port F3: I/O port.<br>Serial send data 1: (Open-drain output mode by programmable.) |
| PF4<br>RXD1 | 1 | I/O<br>Input | Port F4: I/O port.<br>Serial receive data 1. |
| PF5<br>SCLK1<br>$\overline{\text{CTS1}}$ | 1 | I/O<br>I/O<br>Input | Port F5: I/O port.<br>Serial 1 clock I/O.<br>Serial data send enable 1 (Clear to send). |
| PF6 to PF7 | 2 | I/O | Port F6 to F7: I/O port. |
| PG0 to PG7<br>AN0 to AN7<br>$\overline{\text{ADTRG}}$ | 8 | Input<br>Input<br>Input | Port G0 to G7: Input port.<br>Analog input 0 to 7: Pin used to input to AD converter.<br>AD trigger; Pin used to request AD converter start (Share with PG3). |
| $\overline{\text{NMI}}$ | 1 | Input | Non-Maskable interrupt request pin. |
| AM0, AM1 | 2 | Input | Operation mode:<br>Fixed to AM1 = “0”, AM0 = “1”: External 16-bit bus start, 8-/16-bit dynamic sizing.<br>Fixed to AM1 = “1”, AM0 = “0”: External 8-bit bus start, 8-/16-bit dynamic sizing. |
| X1/X2 | 2 | I/O | High-frequency oscillator connection pin. |
| $\overline{\text{RESET}}$ | 1 | Input | Reset: Initialize TMP92CM22 (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 (0 V). |
| DVCC | 3 | | Power supply pins (All Vcc pins should be connected with the power supply pin). |
| DVSS | 4 | – | GND pins (0 V) (All DVSS pins should be connected with GND (0 V)). |

# 3. Operation

This section describes the basic components, functions and operation of the TMP92CM22.

## 3.1 CPU

The TMP92CM22 incorporates a high-performance 32-bit CPU (The TLCS-900/H1 CPU). For a description of this CPU's operation, please refer to the section of this data book which describes the TLCS-900/H1 CPU.

The following sub-sections describe functions peculiar to the CPU used in the TMP92CM22; these functions are not covered in the section devoted to the TLCS-900/H1 CPU.

### 3.1.1 Outline

"TLCS-900/H1 CPU" is high-speed and high-performance CPU based on

"TLCS-900/L1 CPU". "TLCS-900/H1 CPU" has expanded 32-bit internal and external data bus to process instructions more quickly.

Outline of "TLCS-900/H1" CPU are as follows:

Table 3.1.1 Outline of CPU

| | | | |
|---|---|---|---|
| Width of CPU address bus | 24 bits | | |
| Width of CPU data bus | 32 bits | | |
| Internal operating frequency | 20 MHz | | |
| Minimum bus cycle | 1-clock access<br>(50 ns at 20 MHz) | | |
| Function of data bus sizing | 8 bits | | |
| Internal RAM | 32 bits<br>1-clock access | | |
| Internal I/O | 8-/16-bit | 2-clock access | 900/H1 I/O |
| | 8-/16-bit | 5-to 6-clock access | 900/H1 I/O |
| External device | 8 bits<br>2-clock access (can insert some waits) | | |
| Minimum instruction execution cycle | 1 clock (50 ns at 20 MHz) | | |
| Conditional jump | 2 clocks (100 ns at 20 MHz) | | |
| Instruction queue buffer | 12 bytes | | |
| Instruction set | Compatible with TLCS-900, 900/L, 900/H, 900/L1, and 900/H2 instruction codes (However, NORMAL, MAX, MIN, and LDX instructions is deleted) | | |
| CPU mode | Only maximum mode | | |
| Micro DMA | 8 channels | | |

### 3.1.2 Reset Operation

When resetting the TMP92CM22 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 RESET input to low for at least 20 system clocks (16 μs at fc = 40 MHz).

When the reset has been accepted, the CPU performs the following:

- Sets the program counter (PC) as follows in accordance with the reset vector stored at address FFFF00H to FFFF02H:

  PC<7:0> ← Data in location FFFF00H
  PC<15:8> ← Data in location FFFF01H
  PC<23:16> ← Data in location FFFF02H
- Sets the stack pointer (XSP) to 00000000H.
- Sets bits <IFF0:2> of the status register (SR) to 111 (Thereby setting the interrupt level mask register to level 7).
- Clears bits <RFP0:1> 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. CPU internal registers not mentioned above do not change when the reset is released.

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 Registers (SFRs)" in Section 5.
- Sets the input or output port to general-purpose input port.

Internal reset is released as soon as external reset is released and $\overline{\text{RESET}}$ input pin is set to "H".
The operation of memory controller cannot be insured until power supply becomes stable after power-on reset. The external RAM data provided before turning on the TMP92CM22 may be spoiled because the control signals are unstable until power supply becomes stable after power on reset.

Figure 3.1.1 shows the timing of a reset for the TMP92CM22.

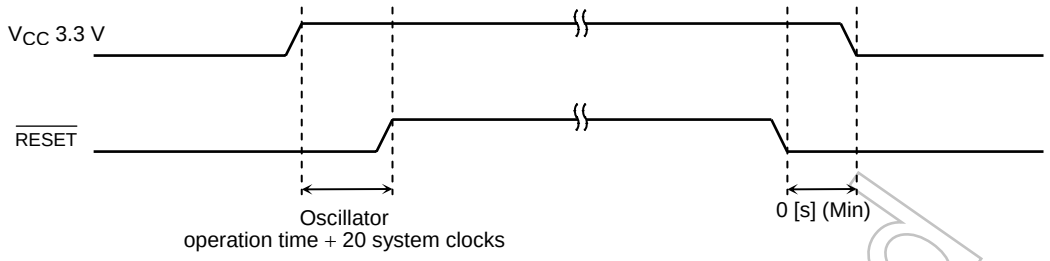



Figure 3.1.1 Reset Timing Example

### 3.1.3 Outline of Operation Mode

Set AM1 and AM0 pins to "10" to use 8-bit external bus, or set it to "01" to use 16-bit external bus.

Table 3.1.2 Operation Mode Setup Table

| Operation | Mode Setting Input Pin | | |
|---|---|---|---|
| | RESET | AM1 | AM0 |
| 16-bit external bus start<br>8-/16-bit dynamic bus sizing | ↑ | 0 | 1 |
| 8-bit external bus start<br>8-/16-bit dynamic bus sizing | | 1 | 0 |

## 3.2 Memory Map

Figure 3.2.1 shows memory map of TMP92CM22.

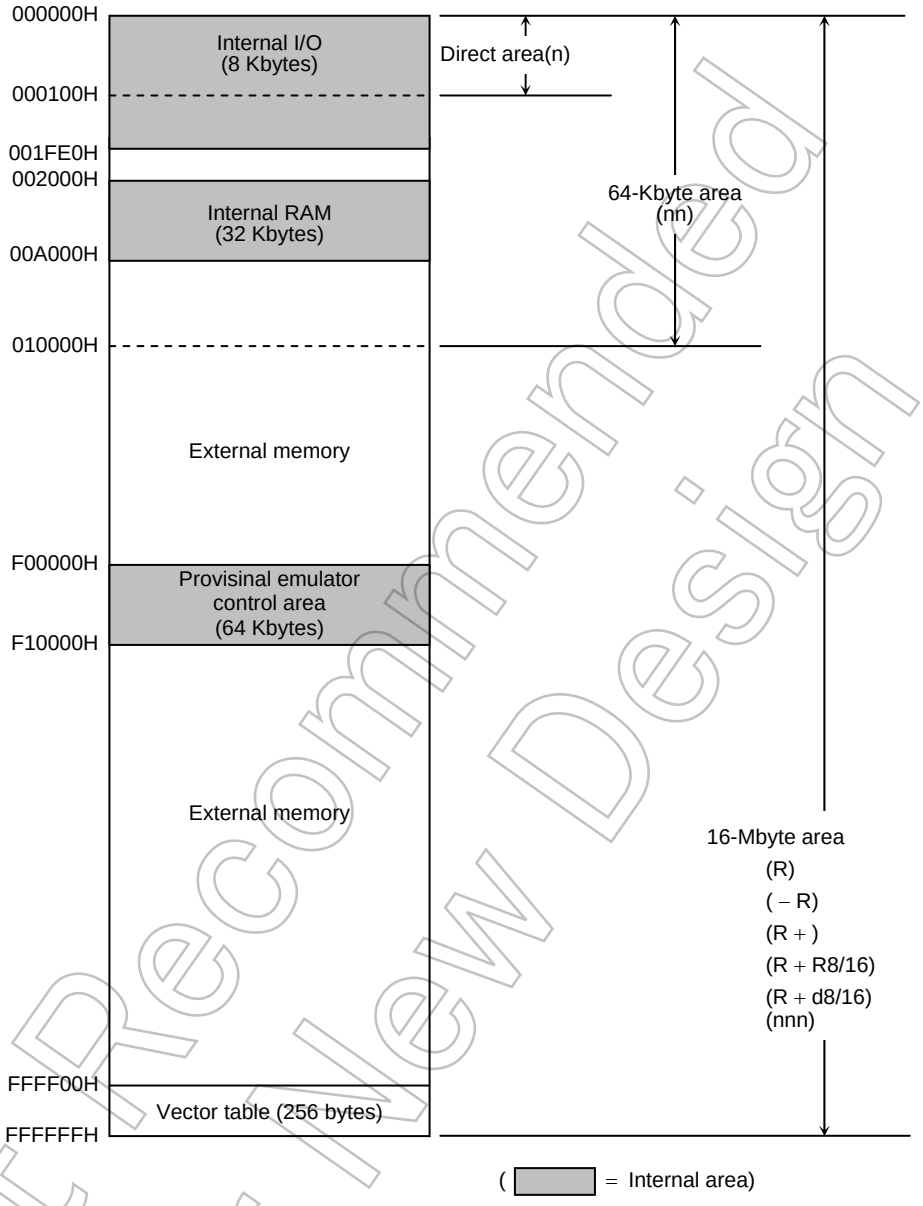



Figure 3.2.1 Memory Map

Note 1: When use emulator, optional 64 Kbytes of 16-Mbyte area are used to control emulator. Therefore, don't use this area.

Note 2: Don't use the last 16-byte area (FFFFF0H to FFFFFFH). This area is reserved.

Note 3: On emulator $\overline{\text{WRLL}}$ signal, $\overline{\text{WRLU}}$ signal and $\overline{\text{RD}}$ signal are asserted, when provisional emulator control area is accessed.

Be careful to use extend memory.

## 3.3 Clock Function and Standby Function

TMP92CM22 contains (1) Clock gear, (2) Standby controller and (3) Noise-reducing circuit. It is used for low-power, low-noise systems.

This chapter is organized as follows:

The clock operating modes are as follows: (a) Single clock mode (X1 and X2 pins only), (b) Dual clock mode (X1, X2 pins and PLL).
Figure 3.3.1 shows a transition figure.

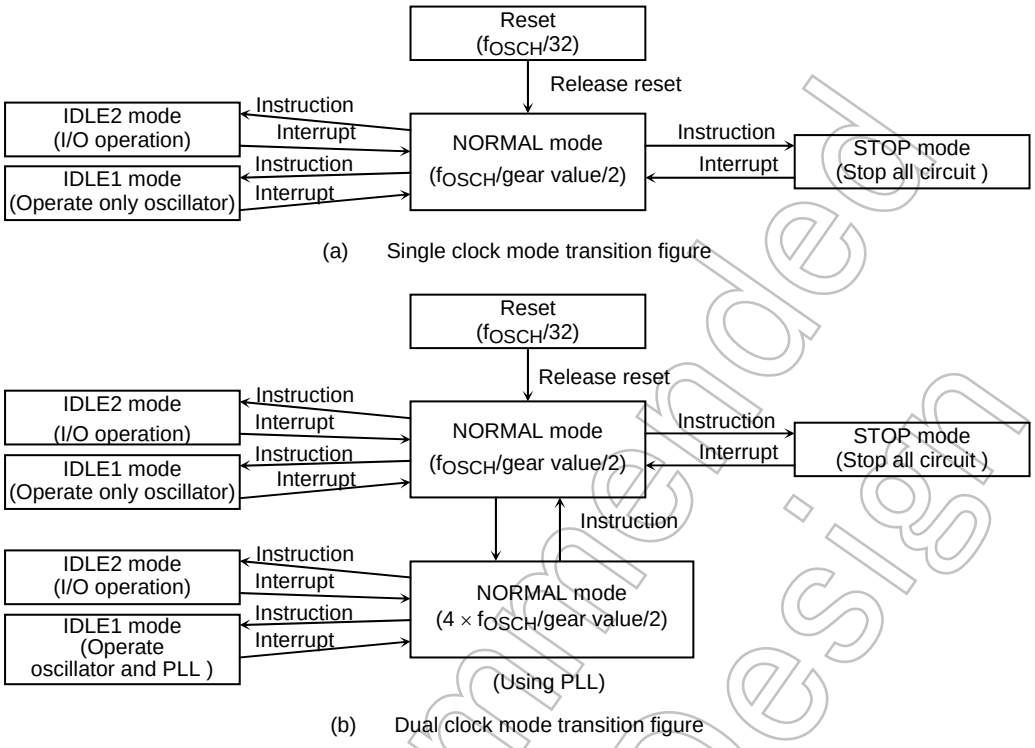



Figure 3.3.1 System Clock Block Diagram

The clock frequency input from the X1 and X2 pins is called $f_{OSCH}$ and the clock frequency selected by SYSCR1<GEAR2:0> is called the clock $f_{FPH}$. The system clock $f_{SYS}$ is defined as the divided 2 clocks of $f_{FPH}$, and one cycle of $f_{SYS}$ is defined to as one state.

### 3.3.1 Block Diagram of System Clock

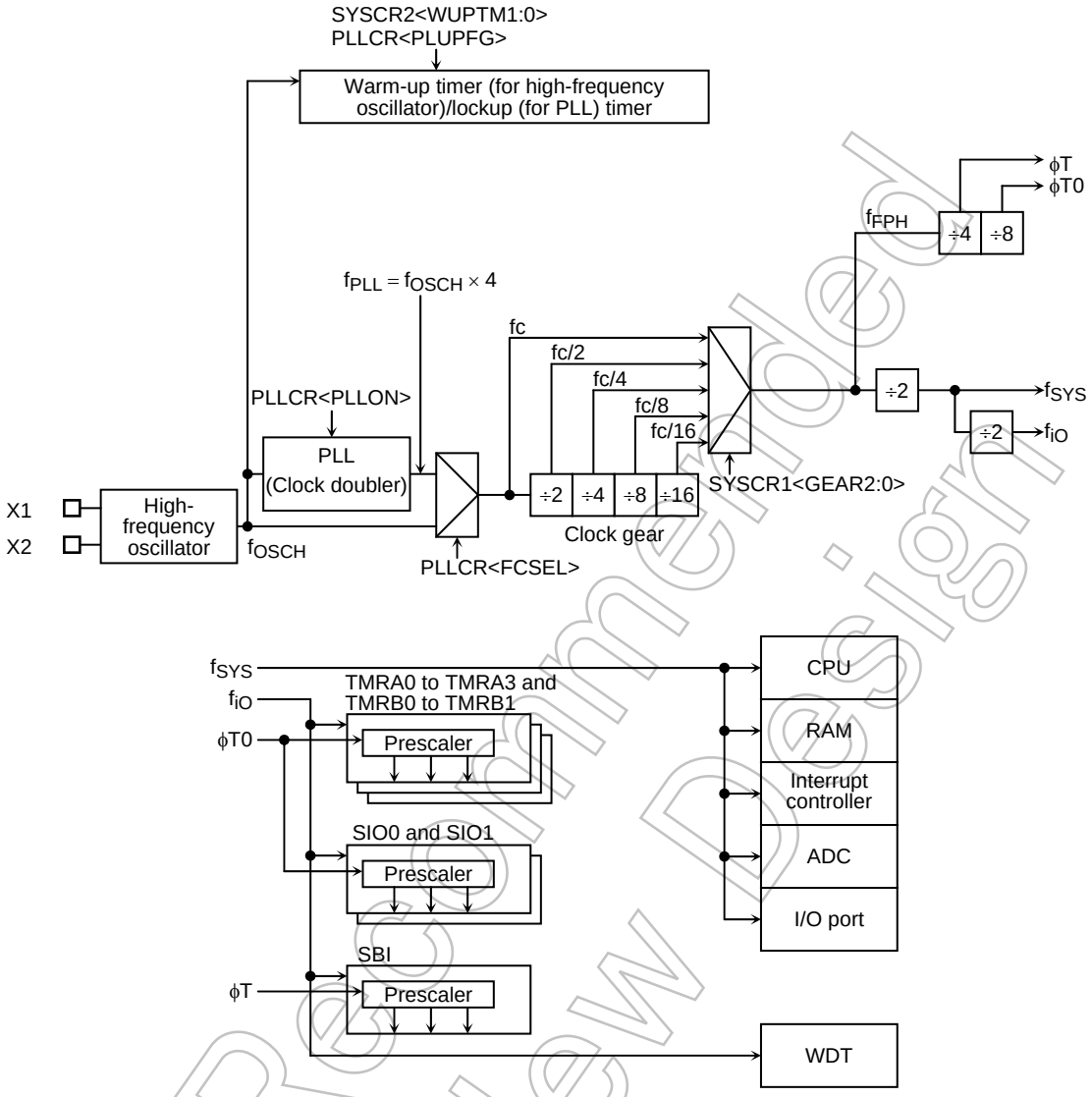



Figure 3.3.2 Block Diagram of Dual Clock and System Clock

### 3.3.2 SFRs

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| SYSCR0 (10E0H) | Bit symbol | – | | | | | – | | |
| | Read/Write | R/W | | | | | R/W | | |
| | After reset | 1 | | | | | 0 | | |
| | Function | Always write “1”. | | | | | Always write “0”. | | |
| SYSCR1 (10E1H) | Bit symbol | | | | | – | GEAR2 | GEAR1 | GEAR0 |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 0 | 1 | 0 | 0 |
| | Function | | | | | Always write “0”. | Select gear value of high-frequency oscillator<br>000: High-frequency oscillator<br>001: High-frequency oscillator/2<br>010: High-frequency oscillator/4<br>011: High-frequency oscillator/8<br>100: High-frequency oscillator/16<br>101:<br>110:<br>111:<br>} Reserved | | |
| SYSCR2 (10E2H) | Bit symbol | – | | WUPTM1 | WUPTM0 | HALTM1 | HALTM0 | SELDRV | DRVE |
| | Read/Write | R/W | | R/W | | | | | |
| | After reset | 0 | | 1 | 0 | 1 | 1 | 0 | 0 |
| | Function | Always write “0”. | | Select WUP time for oscillator<br>00: Reserved<br>01: $2^8$/Input frequency<br>10: $2^{14}$/Input frequency<br>11: $2^{16}$/Input frequency | | Select HALT mode<br>00: Reserved<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE2 mode | | <DRVE><br>Select using mode<br>0: STOP<br>1: IDLE1 | 1: Pin state control in STOP/ IDLE1 mode |

Note: The unassigned register, SYSCR0<bit6:3>, SYSCR0<bit1:0>, SYSCR1<bit7:4>, and SYSCR2<bit6> are RD as undefined value.

Figure 3.3.3 SFR for System Clock

PLLCR (10E8H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PLLON | FCSEL | LWUPFG | | | | | |
| Read/Write | R/W | | R | | | | | |
| After reset | 0 | 0 | 0 | | | | | |
| Function | 0: PLL stop<br>1: PLL run | 0: fc = OSCH<br>1: fc = PLL (× 4) | PLL warm-up flag<br>0: Don't end up or stop<br>1: End up | | | | | |

Note: Logic of PLLCR<LWUPFG> is different DFM of 900/L1.

Figure 3.3.4 SFR for PLL

EMCCR0 (10E3H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PROTECT | | | | | EXTIN | DRVOSCH | – |
| Read/Write | R | | | | | R/W | | |
| After reset | 0 | | | | | 0 | 1 | 1 |
| Function | Protect<br>0: OFF<br>1: ON | | | | | 1: fc external clock | fc oscillator driver ability<br>1: Normal<br>0: Weak | Always write "1". |

EMCCR1 (10E4H)

| | |
|---|---|
| Bit symbol | Switching the protect ON/OFF by write to following 1st-KEY, 2nd-KEY<br>1st-KEY: EMCCR1 = 5AH, EMCCR2 = A5H in succession write<br>2nd-KEY: EMCCR1 = A5H, EMCCR2 = 5AH in succession write |
| Read/Write | |
| After reset | |
| Function | |

EMCCR2 (10E5H)

| | |
|---|---|
| Bit symbol | |
| Read/Write | |
| After reset | |
| Function | |

Note: In case restarting the oscillator in the stop oscillation state (e.g. Restart the oscillator in STOP mode), set EMCCR0<DRVOSCH>, <DRVOSCL>= "1".

Figure 3.3.5 SFR for Noise

### 3.3.3 System Clock Controller

The system clock controller generates the system clock signal ($f_{SYS}$) for the CPU core and internal I/O. It is used as input that fc outputted from high-frequency oscillation circuit and PLL (Clock doubler) SYSCR1<GEAR2:0>, SYSCR1<GEAR2:0> sets the high-frequency clock gear to either 1, 2, 4, 8, or 16 (fc, fc/2, fc/4, fc/8, or fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

Single clock mode is set by resetting, initialized to <GEAR2:0> = "100". This setting will cause the system clock ($f_{SYS}$) to be set to fc/32 (fc/16×1/2).

For example, $f_{SYS}$ is set to 1.25 MHz when the 40MHz oscillator is connected to the X1 and X2 pins.

(1) Clock gear controller

$f_{FPH}$ 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 $f_{FPH}$ reduces power consumption.

Example:
Changing to a high-frequency gear

```
SYSCR1    EQU   10E1H
          LD    (SYSCR1), XXXX0100B    ;  Changes system clock fSYS to fc/32.
```

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 the warm-up time until changing after writing the register value.

There is the possibility that the instruction next to the clock gear changing instruction is executed by the clock gear before changing. To execute the instruction next to 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), XXXX0001B    ;  Changes fSYS to fc/4.
          LD    (DUMMY), 00H           ;  Dummy instruction.
          Instruction to be executed
          after clock gear has changed.
```

### 3.3.4 Clock Doubler (PLL)

PLL outputs the $f_{PLL}$ clock signal, which is four times as fast as $f_{OSCH}$. A reset initializes PLL to stop status, setting to PLLCR register is needed before use.

Like an oscillator, this circuit requires time to stabilize. This is called the lockup time.

Note 1: Input frequency limitation for PLL

The limitation of input frequency (High-frequency oscillation) for PLL is the following.

$f_{OSCH}$ = 4 to 10 MHz (Vcc = 3.0 V to 3.6 V)

Note 2: PLLCR<LWUPFG>

The logic of PLLCR<LUPFG> is different from 900/L1's DFM.

Be careful to judge an end of lockup time.

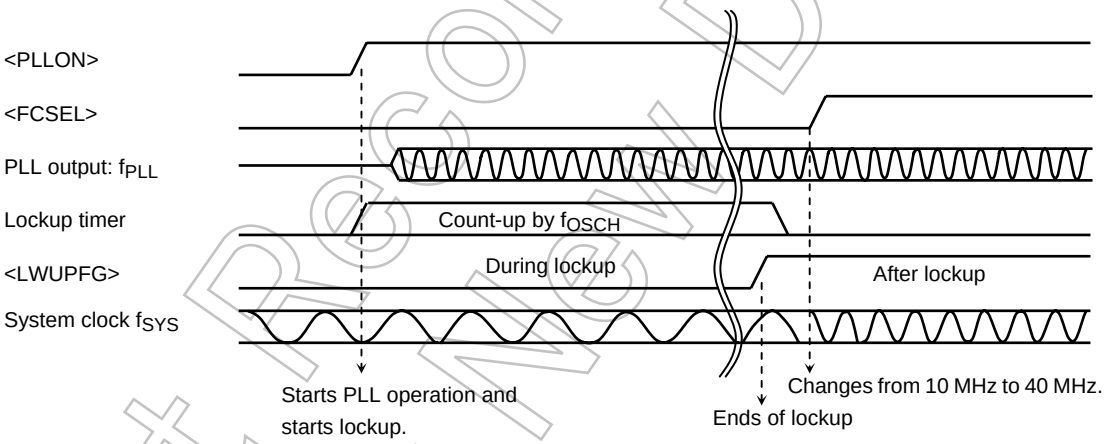
The following is a setting example for PLL starting and PLL stopping.

Example 1: PLL starting

```
PLLCR   EQU   10E8H

        LD    (PLLCR), 10XXXXXXB  ;  Enables PLL operation and starts lockup.
LUP:    BIT   5, (PLLCR)          ;
        JR    Z, LUP              ;  Detects end of lockup.
        LD    (PLLCR), 11XXXXXXB  ;  Changes fc from 10 MHz to 40 MHz.
```

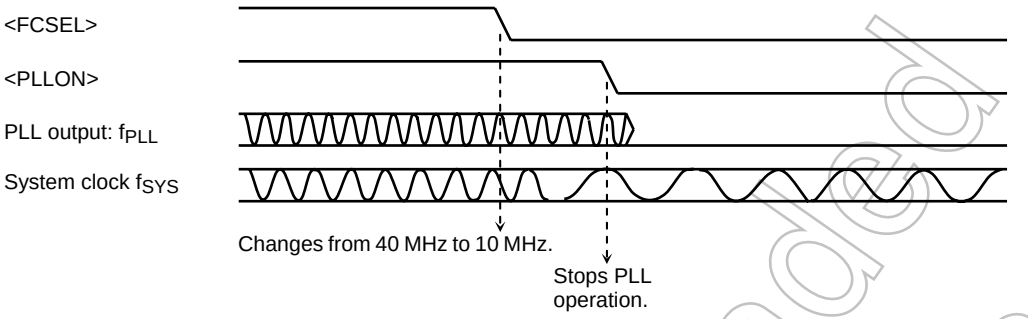
X: Don't care



Example 2: PLL stopping

```
PLLCR    EQU   10E8H
         LD    (PLLCR), 10XXXXXXB    ;   Changes fc from 40 MHz to10 MHz.
         LD    (PLLCR), 00XXXXXXB    ;   Stop PLL.
```

X: Don't care




Limitation point on the use of PLL

1. When PLL is started, don't set fc from $f_{OSCH}$ to $f_{PLL}$ at same time.

   Don't setting:

   LD (PLLCR), 00H
   LD (PLLCR), C0H

2. When PLL is started, don't set fc from $f_{OSCH}$ to $f_{PLL}$ at same time.

   Don't setting:

   LD (PLLCR), C0H
   LD (PLLCR), 00H

### 3.3.5 Noise Reduction Circuits

Noise reduction circuits are built in for reduction EMI (Unnecessary radius noise) and reinforcement EMS (Measure of endure noise), allowing implementation of the following features.

(1) Reduced drivability for high-frequency oscillator

(2) Single drive for high-frequency oscillator

(3) SFR protection of register contents

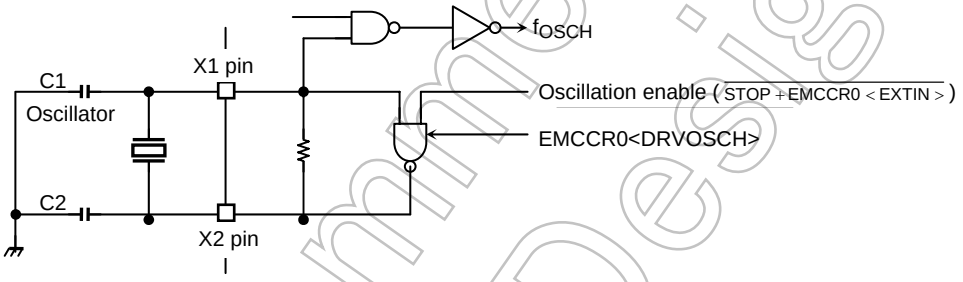
These functions need setting by EMCCR0 to EMCCR2.

(1) Reduced drivability for high-frequency oscillator

(Purpose)

Reduces noise and power for oscillator when connect oscillator to outside.

(Block diagram)




(Setting method)

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

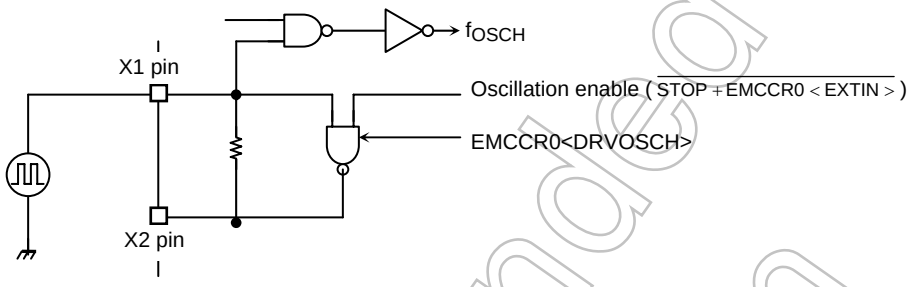
Note: When use drivability reduction function of oscillator, please use in case of $f_{OSCH}$ = 4 MHz to 10 MHz condition.

(2) Single drive for high-frequency oscillator

(Purpose)

Not need twin-drive and protect mistake operation by inputted noise to X2 pin when the 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 is always outputted "1".

By reset, <EXTIN> is initialized to "0".

(3) Runaway provision with SFR protection register

(Purpose)

Provision in runaway of program by noise mixing.

Write operation to specified SFR is prohibited so that provision program in runaway prevents that is in the state which is fetch impossibility by stopping of clock, memory control register (Memory controller) is changed.

And error handling in runaway becomes easy by INTP0 interruption.

Specified SFR list

1. Memory controller
   B0CSL/H, B1CSL/H, B2CSL/H, B3CSL/H, BEXCSL/H,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3, and PMEMCR
2. Clock gear (EMCCR1, EMCCR2 write enable)
   SYSCR0, SYSCR1, SYSCR2, and EMCCR0

(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: Succession writes in 5AH at EMCCR1 and A5H at EMCCR2.
2nd-KEY: Succession writes in A5H at EMCCR1 and 5AH at EMCCR2.

A state of protection can be confirmed by reading EMCCR0<PROTECT>.

By reset, protection becomes OFF.

And INTP0 interruption occurs when write operation to specified SFR was executed with protection on state.

### 3.3.6 Standby Controller

(1) HALT modes

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.

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 3.3.1 shows the registers of setting operation during IDLE2 mode.

Table 3.3.1 SFR Seting Operation during IDLE2 Mode

| Internal I/O | SFR |
|---|---|
| TMRA01 | TA01RUN<I2TA01> |
| TMRA23 | TA23RUN<I2TA23> |
| TMRB0 | TB0RUN<I2TB0> |
| TMRB1 | TB1RUN<I2TB1> |
| SIO0 | SC0MOD1<I2S0> |
| SIO1 | SC1MOD1<I2S1> |
| AD converter | ADMOD1<I2AD> |
| WDT | WDMOD<I2WDT> |
| SBI | SBI0BR0<I2SBI0> |

b. IDLE1: Only internal oscillator operates.

c. STOP: All internal circuit stop.

The operation of each of the different HALT modes is described in Table 3.3.2.

Table 3.3.2 Each Block Operation in HALT Mode

| HALT Mode | | IDLE2 | IDLE1 | STOP |
|---|---|---|---|---|
| SYSCR2<HALTM1:0> | | 11 | 10 | 01 |
| Operation block | CPU | Stop | | |
| | I/O port | Keep the state when the HALT instruction is executed. | Refer Table 3.3.5, Table 3.3.6 | |
| | TMRA, TMRB | * Selection enable operation block to programmable | Stop | |
| | SIO, *SBI | | | |
| | AD converter | | | |
| | WDT | | | |

*: Except clocked-synchronous 8-bit SIO mode for SBI.

(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 between the states of interrupt mask register <IFF2:0> and the HALT modes. The details for release the halt status are shown in Table 3.3.3.

- Released by requesting an interrupt

  The operating released from the HALT mode depends on the interrupt enabled status. When the interrupt request level set before executing the HALT instruction exceeds the value of interrupt mask register, the interrupt due to the source is processed after release the HALT mode, and CPU status executing an 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, release the HALT mode is not executed. (In non-maskable interrupts, interrupt processing is processed after release the HALT mode regardless of the value of the mask register.) However only for INT0 to INT3 interrupts, even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, release the HALT mode is executed. In this case, interrupt processing, and CPU starts executing the instruction next to the HALT instruction, but the interrupt request flag is held at "1".

- Release by resetting

  Release all halt status is executed by resetting.

  When the STOP mode is released by RESET, it is necessary enough resetting time (Refer Table 3.3.4) to set the operation of the oscillator to be stable.

  When release 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. (Release due to interrupts keeps the state before the "HALT" instruction is executed.)

Table 3.3.3 Source of Halt State Release and Halt Release Operation

| Status of Received Interrupt | | | Interrupt Enable (Interrupt level) ≥ (Interrupt mask) | | | Interrupt Disable (Interrupt level) < (Interrupt mask) | | |
|---|---|---|---|---|---|---|---|---|
| HALT Mode | | | Programmable IDLE2 | IDLE1 | STOP | Programmable IDLE2 | IDLE1 | STOP |
| Source of HALT state release | Interrupt | NMI | ♦ | ♦ | ♦ | – | – | – |
| | | INTWDT | ♦ | × | × | – | – | – |
| | | INT0 to 3 (Note1) | ♦ | ♦ | ♦*1 | ○ | ○ | ○*1 |
| | | INT4 to 5 | ♦ | × | × | × | × | × |
| | | INTTA0 to 3, | ♦ | × | × | × | × | × |
| | | INTTB00, 01, 10, 11, O0, O1 | ♦ | × | × | × | × | × |
| | | INTRX0 to 1, TX0 to 1 | ♦ | × | × | × | × | × |
| | | INTAD | ♦ | × | × | × | × | × |
| | | INTSBE0 | ♦ | × | × | × | × | × |
| | Reset | | Initialize LSI | | | | | |

♦: After release the HALT mode, CPU starts interrupt processing.

○: After release the HALT mode, CPU resumes executing starting from instruction following the HALT instruction. (Interrupt don't process.)

×: It can not 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. There is not this combination type.

*1: Release the HALT mode is executed after passing the warm-up time.

Note 1: When the HALT mode is released by INT0 to INT3 interrupts of the level mode in the interrupt enabled status, hold this level until starting interrupt processing. Changing level before holding level, interrupt processing is correctly started.

Note 2: When use external interrupt INT4 to INT5 are used during IDLE2 mode, set 16-bit timer RUN register TB1RUN<I2TB1> to "1".

(Example release HALT mode)

An INT0 interrupt release the halt state when the device is in IDLE1 mode.

```
Address
8203H   LD      (IIMC), 00H     ;   Selects INT0 interrupt rising edge.
8206H   LD      (INTE0AD), 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.

INT0                                        INT0 interrupt routine
                                                        RETI
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 3.3.6 illustrates an example of the timing for clearance of the IDLE2 mode halt state by an interrupt.

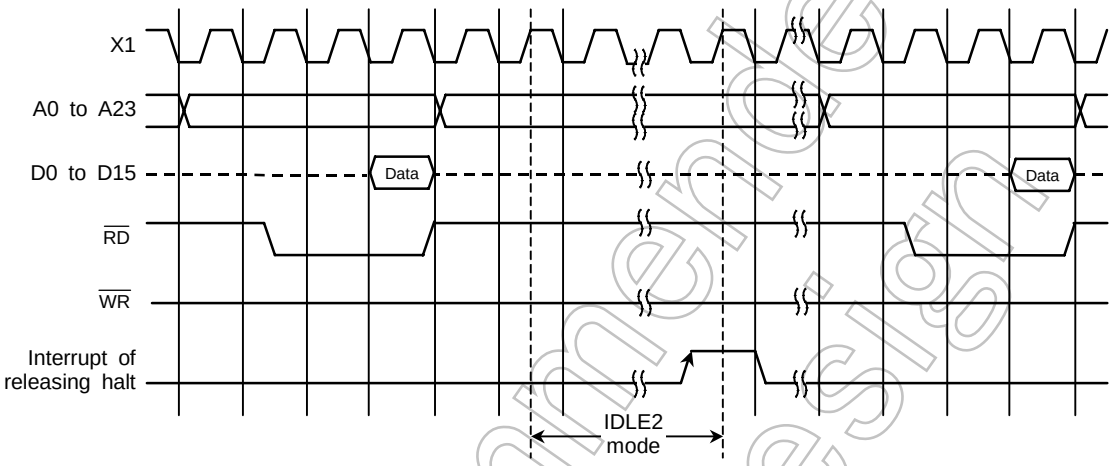



Figure 3.3.6 Timing Chart for IDLE2 Mode Halt State Released by Interrupt

b. IDLE1 mode

In IDLE1 mode, only the internal oscillator operates. The system clock stops.

And, pin state in IDLE1 mode depend on setting SYSCR2<SELDRV, DRVE> register. Table 3.3.5, Table 3.3.6 shows pin state in IDLE1 mode.

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

Figure 3.3.7 shows the timing for release of the IDLE1 mode halt state by an interrupt.

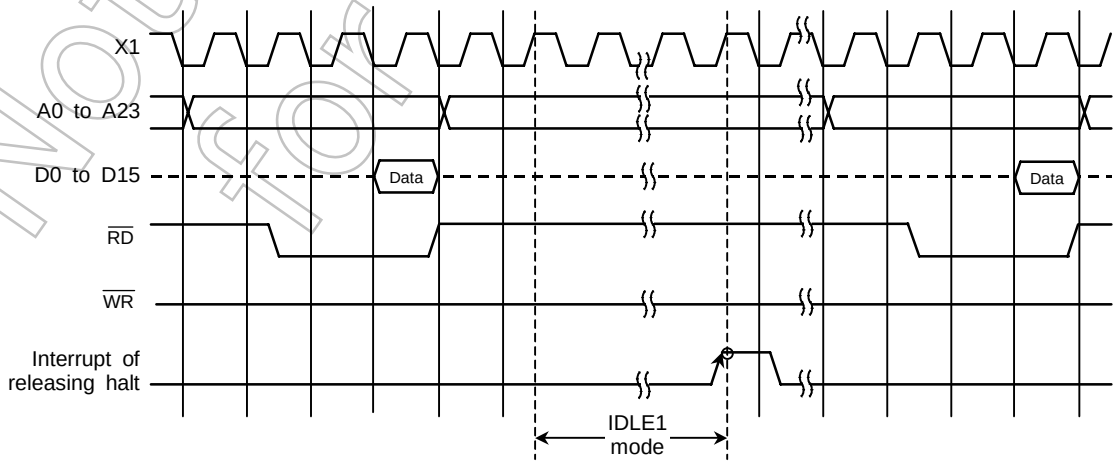



Figure 3.3.7 Timing Chart for IDLE1 Mode Halt State Released by Interrupt

c. STOP mode

When STOP mode is selected, all internal circuits stop, including the internal oscillator pin status in STOP mode depends on the settings in the SYSCR2<SELDRV, DRVE> register. Table 3.3.5, Table 3.3.6 shows the state of these pins in STOP mode.

After STOP mode has been released system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize. Warm-up time set by SYSCR2<WUPTM1:0> register. See the sample warm-up times in Table 3.3.4.

Figure 3.3.8 illustrates the timing for release of the STOP mode halt state by an interrupt.

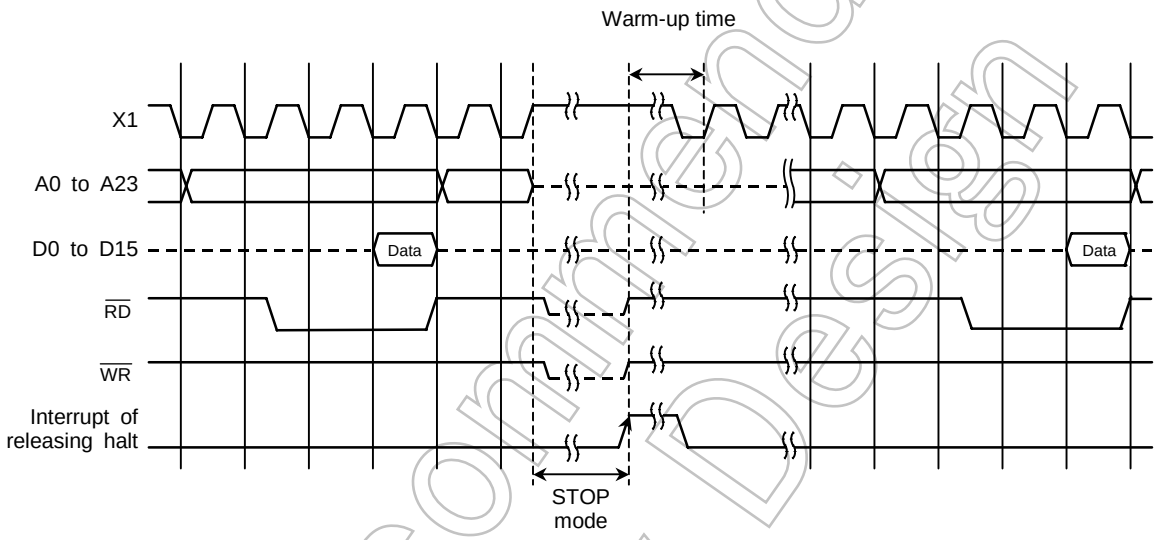



Figure 3.3.8 Timing Chart for STOP Mode Halt State Released by Interrupt

Table 3.3.4 Sample Warm-up Times after Rrelease of STOP Mode

at $f_{OSCH}$ = 10 MHz

| SYSCR2<WUPTM1:0> | | |
|---|---|---|
| 01 ($2^8$) | 10 ($2^{14}$) | 11 ($2^{16}$) |
| 25.6 μs | 1.638 ms | 6.554 ms |

Table 3.3.5 Input Buffer State Table

| Port Name | Input Function Name | Input Buffer State | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | During Reset | Input Buffer State | | Input Buffer State | | In HALT mode (IDLE1/STOP) | | | |
| | | | | | | | Condition A (Note) | | Condition B (Note) | |
| | | | When Used as function Pin | When Used as Input Port | When Used as function Pin | When Used as Input Port | When Used as function Pin | When Used as Input Port | When Used as function Pin | When Used as Input Port |
| D0-D7 | D0-D7 | OFF | ON upon external read | – | OFF | – | OFF | – | OFF | – |
| P10-P17 | D8-D15 | | | ON | | OFF | | OFF | | OFF |
| P40-P47 | – | | – | | – | | – | | – | |
| P50-P57 | – | | | | | | | | | |
| P60-P67 | – | | | | | | | | | |
| P76 | WAIT | ON | ON | | OFF | | OFF | | OFF | |
| P90 | SCK | | | | ON | | | | | |
| P91 | SDA | | | | | | | | | |
| P92 | SI SCL | | | | | | | | | |
| PA0-PA7(*1) | – | | – | | – | ON | – | ON | – | ON |
| PC0 | TA0IN | | ON | | ON | OFF | OFF | OFF | OFF | OFF |
| PC1 | INT1 | | | | | | | | | |
| PC3 | INT0 | | | | | ON | ON | ON | ON | ON |
| PC5 | INT2 | | | | | OFF | OFF | OFF | OFF | OFF |
| PC6 | INT3 | | | | | | | | | |
| PD0 | INT4, TB1IN0 | | | | | | | | | |
| PD1 | INT5, TB1IN1 | | | | | | | | | |
| PD2 | – | | – | | – | | – | | – | |
| PD3 | – | | | | | | | | | |
| PF0 | – | | | | | | | | | |
| PF1 | RXD0 | | ON | | ON | ON | OFF | | OFF | |
| PF2 | SCLK0, CTS0 | | | | | | | | | |
| PF3 | – | | – | | – | OFF | – | | – | |
| PF4 | RXD1 | | ON | | ON | ON | OFF | | OFF | |
| PF5 | SCLK1, CTS1 | | | | | | | | | |
| PF6 | – | | – | | – | | – | | – | |
| PF7 | – | | | | | | | | | |
| PG0-2, PG4-7(*2) | – | OFF | | ON upon port read | | OFF | | | | |
| PG3(*2) | ADTRG | | ON | | ON | | ON | | ON | |
| $\overline{NMI}$ | – | ON | | – | | – | | – | | – |
| RESET(*1) | – | | | | | | | | | |
| AM0,1 | – | | | | | | | | | |
| X1 | | | | | | | | | | |

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

OFF: The buffer is always turned off.

–: No applicable

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

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

Note: Condition A/B are as follows.

| SYSCR2 register setting | | HALT mode | |
|---|---|---|---|
| <DRVE> | <SELDRV> | IDLE1 | STOP |
| 0 | 0 | Condition B | Condition A |
| 0 | 1 | Condition A | |
| 1 | 0 | Condition B | Condition B |
| 1 | 1 | | |

Table 3.3.6 Output Buffer State Table

| Port Name | Output Function Name | Output Buffer State | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | During Reset | When the CPU is Operating | | In HALT mode(IDLE2) | | In HALT mode (IDLE1/STOP) | | | |
| | | | | | | | Condition A (Note) | | Condition B (Note) | |
| | | | When Used as Function Pin | When Used as Output Port | When Used as Function Pin | When Used as Output Port | When Used as Function Pin | When Used as Output Port | When Used as Function Pin | When Used as Output Port |
| D0-D7 | D0-D7 | OFF | ON upon external read | – | OFF | – | OFF | – | OFF | – |
| P10-P17 | D8-D15 | | | ON | | ON | | ON | | ON |
| P40-P47 | A0-A7 | ON | ON | | ON | | | | | |
| P50-P57 | A8-A15 | | | | | | | | | |
| P60-P67 | A16-A23 | | | | | | | | | |
| P70 | $\overline{RD}$ | | | | | | | | | |
| P71 | WRLL | | | | | | | | | |
| P72 | WRLU | | | | | | | | | |
| P73 | WRUL | | | | | | | | | |
| P74 | WRUU | | | | | | | | | |
| P75 | R/W | | | | | | | | | |
| P76 | – | OFF | – | | – | | – | | – | |
| P80 | CS0 | ON | ON | | ON | | OFF | | ON | |
| P81 | CS1 | | | | | | | | | |
| P82 | CS2 | | | | | | | | | |
| P83 | CS3 | | | | | | | | | |
| P90 | SCK | OFF | | | | | | | | |
| P91 | SO | | | | | | | | | |
| P92 | SCL | | | | | | | | | |
| PC0 | – | | – | | – | | – | | – | |
| PC1 | TA1OUT | | ON | | ON | | OFF | | ON | |
| PC3 | – | | – | | – | | – | | – | |
| PC5 | TA3OUT | | ON | | ON | | OFF | | ON | |
| PC6 | TB0OUT | | | | | | | | | |
| PD0 | – | | – | | – | | – | | – | |
| PD1 | – | | | | | | | | | |
| PD2 | TB1OUT0 | | ON | | ON | | OFF | | ON | |
| PD3 | TB1OUT1 | | | | | | | | | |
| PF0 | TXD0 | | ON | | | | | | | |
| PF1 | – | | – | | – | | – | | – | |
| PF2 | SCLK0 | | ON | | ON | | OFF | | ON | |
| PF3 | TXD1 | | | | | | | | | |
| PF4 | – | | – | | – | | – | | – | |
| PF5 | SCLK1 | | ON | | ON | | OFF | | ON | |
| PF6 | – | | – | | – | | – | | – | |
| PF7 | – | | | | | | | | | |
| X2 | – | | | – | ON | – | IDLE1: ON, STOP: High level output | | | |

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

OFF: The buffer is always turned off.

–: No applicable

Note: Condition A/B are as follows.

| SYSCR2 register setting | | HALT mode | |
|---|---|---|---|
| <DRVE> | <SELDRV> | IDLE1 | STOP |
| 0 | 0 | Condition B | Condition A |
| 0 | 1 | Condition A | |
| 1 | 0 | Condition B | Condition B |
| 1 | 1 | | |

## 3.4 Interrupt

Interrupts of TLCS-900/H1 are controlled by the CPU interrupt mask flip-flop (IFF2:0) and by the built-in interrupt controller.

The TMP92CM22 has a total of 41 interrupts divided into the following types:

Interrupts generated by CPU: 9 sources
(Software interrupts: 8 sources, illegal instruction interrupt: 1 source)

External interrupts ($\overline{\text{NMI}}$ and INT0 to INT5): 7 sources

Internal I/O interrupts: 17 sources

High-speed DMA interrupts: 8 sources

A individual interrupt vector number (Fixed) is assigned to each interrupt.

One of six priority level (Variable) can be assigned to each maskable interrupt.

The priority level of non-maskable interrupts are fixed at 7 as the highest level.

When an interrupt is generated, the interrupt controller sends the priority of that interrupt to the CPU. If multiple interrupts is generated simultaneously, the interrupt controller sends the interrupt with the highest priority to the CPU. (The highest priority is level 7 using for non-maskable interrupts.)

The CPU compares the priority level of the interrupt with the value of the CPU interrupts mask register <IFF2:0>. If the priority level of the interrupt is higher than the value of the interrupt mask register, the CPU accepts the interrupt.

The interrupt mask register <IFF2:0> value can be updated using the value of the EI instruction (EI num sets <IFF2:0> data to num).

For example, specifying "EI3" enables the maskable interrupts which priority level set in the interrupt controller is 3 or higher, and also non-maskable interrupts.

Operationally, the DI instruction (<IFF2:0> = 7) is identical to the EI7 instruction. DI instruction is used to disable maskable interrupts because of the priority level of maskable interrupts is 1 to 6. The EI instruction is valid immediately after execution.

In addition to the above general-purpose interrupt processing mode, TLCS-900/H1 has a micro DMA interrupt processing mode as well. The CPU can transfer the data (1/2/4 bytes) automatically in micro DMA mode, therefore this mode is used for speed-up interrupt processing, such as transferring data to the internal or external peripheral I/O. Moreover, TMP92CM22 has software start function for micro DMA processing request by the software not by the hardware interrupt.

Figure 3.4.1 shows the overall interrupt processing flow.

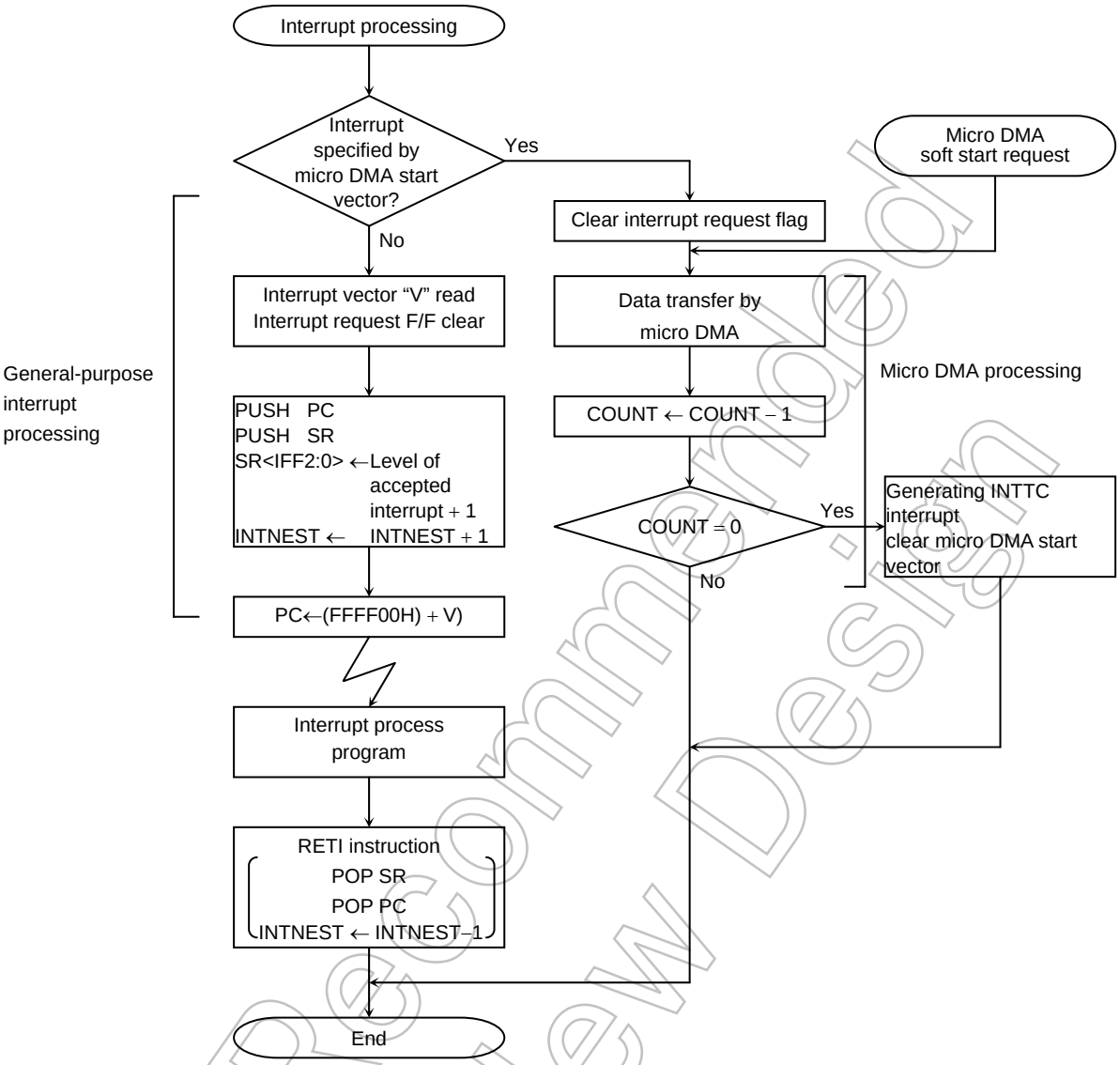



Figure 3.4.1 Interrupt and Micro DMA Processing Sequence

### 3.4.1 General-purpose Interrupt Processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. That is also the same as TLCS-900/L, TLCS-900/H, and TLCS-900/L1.

(1) The CPU reads the interrupt vector from the interrupt controller.
If the same level interrupts occur simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt request.
(The default priority is already fixed for each interrupt: The smaller vector value has the higher priority level.)

(2) The CPU pushes the value of program counter (PC) and status register (SR) onto the stack area (indicated by XSP).

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

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

(5) The CPU jumps to the address indicated by the data at address "FFFF00H + Interrupt vector" and starts the interrupt processing routine.

When the CPU completed the interrupt processing, use the RETI instruction to return to the main routine. RETI restores the contents of program counter (PC) and status register (SR) from the stack and decreases the interrupt nesting counter INTNEST by 1(–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 which has a priority level equal to or greater than the value of the CPU interrupt mask register <IFF2:0> comes out, the CPU accepts its interrupt. Then, the CPU interrupt mask register <IFF2:0> is set to the value of the priority level for the accepted interrupt plus 1(+1).
Therefore, if an interrupt is generated with a higher level than the current interrupt during it's processing, the CPU accepts the later interrupt and goes to the nesting status of interrupt processing.

Moreover, if the CPU receives another interrupt request while performing the said (1) to (5) processing steps of the current interrupt, the latest interrupt request is sampled immediately after execution of the first instruction of the current interrupt processing routine. Specifying DI as the start instruction disables maskable interrupt nesting.

A reset initializes the interrupt mask register <IFF2:0> to "7", disabling all maskable interrupts.

Table 3.4.1 shows the TMP92CM22 interrupt vectors and micro DMA start vectors. The address FFFF00H to FFFFFFH (256 bytes) is assigned for the interrupt vector area.

Table 3.4.1 TMP92CM22 Interrupt Vectors and Micro DMA Start Vectors

| Default Priority | Type | Interrupt Source | Vector Value | Address Refer to Vector | Micro DMA 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 | | NMI: External interrupt input pin | 0020H | FFFF20H | |
| 10 | | INTWD: Watchdog Timer | 0024H | FFFF24H | |
| - | Maskable | Micro DMA (Note 2) | – | – | – |
| 11 | | INT0: External interrupt input pin | 0028H | FFFF28H | 0AH (Note 1) |
| 12 | | INT1: External interrupt input pin | 002CH | FFFF2CH | 0BH (Note 1) |
| 13 | | INT2: External interrupt input pin | 0030H | FFFF30H | 0CH (Note 1) |
| 14 | | INT3: External interrupt input pin | 0034H | FFFF34H | 0DH (Note 1) |
| 15 | | (Reserved) | 0038H | FFFF38H | 0EH |
| 16 | | (Reserved) | 003CH | FFFF3CH | 0FH |
| 17 | | (Reserved) | 0040H | FFFF40H | 10H |
| 18 | | (Reserved) | 0044H | FFFF44H | 11H |
| 19 | | (Reserved) | 0048H | FFFF48H | 12H |
| 20 | | (Reserved) | 004CH | FFFF4CH | 13H |
| 21 | | INTP0: Protect 0 (WR to SFR) | 0050H | FFFF50H | 14H |
| 22 | | (Reserved) | 0054H | FFFF54H | 15H |
| 23 | | INTTA0: 8-bit timer 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 | | INTTB00: 16-bit timer 0 | 0068H | FFFF68H | 1AH |
| 28 | | INTTB01: 16-bit timer 0 | 006CH | FFFF6CH | 1BH |
| 29 | | (Reserved) | 0070H | FFFF70H | 1CH |
| 30 | | (Reserved) | 0074H | FFFF74H | 1DH |
| 31 | | INTTBO0: 16-bit timer 0 (Overflow) | 0078H | FFFF78H | 1EH |
| 32 | | (Reserved) | 007CH | FFFF7CH | 1FH |
| 33 | | INTRX0: Serial 0 (SIO0) receive | 0080H | FFFF80H | 20H (Note 1) |
| 34 | | INTTX0: Serial 0 (SIO0) transmission | 0084H | FFFF84H | 21H |
| 35 | | INTRX1: Serial 1 (SIO1) receive | 0088H | FFFF88H | 22H (Note 1) |
| 36 | | INTTX1: Serial 1 (SIO1) transmission | 008CH | FFFF8CH | 23H |
| 37 | | (Reserved) | 0090H | FFFF90H | 24H |
| 38 | | (Reserved) | 0094H | FFFF94H | 25H |
| 39 | | (Reserved) | 0098H | FFFF98H | 26H |
| 40 | | (Reserved) | 009CH | FFFF9CH | 27H |
| 41 | | (Reserved) | 00A0H | FFFFA0H | 28H |
| 42 | | INT4: External interrupt input pin | 00A4H | FFFFA4H | 29H |
| 43 | | INT5: External interrupt input pin | 00A8H | FFFFA8H | 2AH |
| 44 | | INTTB10: 16-bit timer 1 | 00ACH | FFFFACH | 2BH |
| 45 | | INTTB11: 16-bit timer 1 | 00B0H | FFFFB0H | 2CH |
| 46 | | INTTBO1: 16-bit timer 1 (Overflow) | 00B4H | FFFFB4H | 2DH |
| 47 | | (Reserved) | 00B8H | FFFFB8H | 2EH |
| 48 | | INTSBE0: SBI $I^2C$ bus transfer end (Channel 0) | 00BCH | FFFFBCH | 2FH |
| 49 | | (Reserved) | 00C0H | FFFFC0H | 30H |
| 50 | | (Reserved) | 00C4H | FFFFC4H | 31H |
| 51 | | (Reserved) | 00C8H | FFFFC8H | 32H |

| Default Priority | Type | Interrupt Source | Vector Value | Address Refer to Vector | Micro DMA Start Vector |
|---|---|---|---|---|---|
| 52 | Maskable | INTAD: AD conversion end | 00CCH | FFFFCCH | 33H |
| 53 | | INTTC0: Micro DMA end (Channel 0) | 00D0H | FFFFD0H | 34H |
| 54 | | INTTC1: Micro DMA end (Channel 1) | 00D4H | FFFFD4H | 35H |
| 55 | | INTTC2: Micro DMA end (Channel 2) | 00D8H | FFFFD8H | 36H |
| 56 | | INTTC3: Micro DMA end (Channel 3) | 00DCH | FFFFDCH | 37H |
| 57 | | INTTC4: Micro DMA end (Channel 4) | 00E0H | FFFFE0H | 38H |
| 58 | | INTTC5: Micro DMA end (Channel 5) | 00E4H | FFFFE4H | 39H |
| 59 | | INTTC6: Micro DMA end (Channel 6) | 00E8H | FFFFE8H | 3AH |
| 60 | | INTTC7: Micro DMA end (Channel 7) | 00ECH | FFFFECH | 3BH |
| | | (Reserved) | 00F0H<br>:<br>00FCH | FFFFF0H<br>:<br>FFFFFCH | – |

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

Note 2 : Micro DMA default priority.

Micro DMA initiation takes priority over other maskable interrupts

### 3.4.2 Micro DMA

In addition to general-purpose interrupt processing, the TMP92CM22 also includes a micro DMA function. 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 by a Halt instruction, the requirements of the micro DMA will be ignored (pending).

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

(1) Micro DMA operation

When an interrupt request is generated by an interrupt source specified by the micro DMA start vector register, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request. The eight micro DMA channels allow micro DMA processing to be set for up to eight 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 start vector register is cleared to 0, the next micro DMA operation is disabled and micro DMA processing terminates.

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).

If an interrupt request is triggered for the interrupt source in use during the interval between the time at which the micro DMA 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 (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. (Note) In this case, edge triggered interrupts are the only kinds of general interrupts which can be accepted.

Note: 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 theFigure 3.4.1) 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

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 eight bits of a 32-bit address are not valid).

Three micro DMA transfer modes are supported: one-byte transfers, two-byte (one-word) transfer and four-byte transfer. 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 3.4.2 (1), 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 34 different interrupts – the 33 interrupts shown in the micro DMA start vectors in Table 3.4.1 and a micro DMA soft start.

Figure 3.4.2 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 multiples by 4 numbered source and destination addresses.)

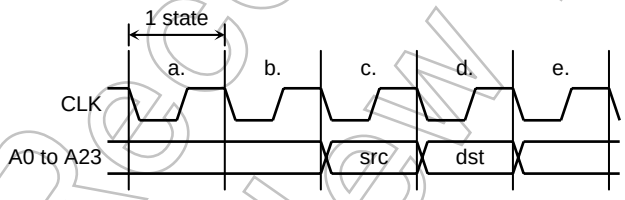



Figure 3.4.2 Timing for Micro DMA Cycle

States 1 to 2: Instruction fetches cycle (Gets next address code).
If the instruction queue buffer is FULL , this cycle becomes a dummy cycle.
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

In addition to starting the micro DMA function by interrupts, TMP92CM22 includes a micro DMA software start function that starts micro DMA on the generation of the write cycle to the DMAR register.

Writing "1" to each bit of DMAR register causes micro DMA once (If write "0" to each bit, micro DMA doesn't operate). At the end of transfer, the corresponding bit of the DMAR register is automatically cleared to "0".

Only one channel can be set for DMA request at once. (Do not write 1 to more than one bit.)

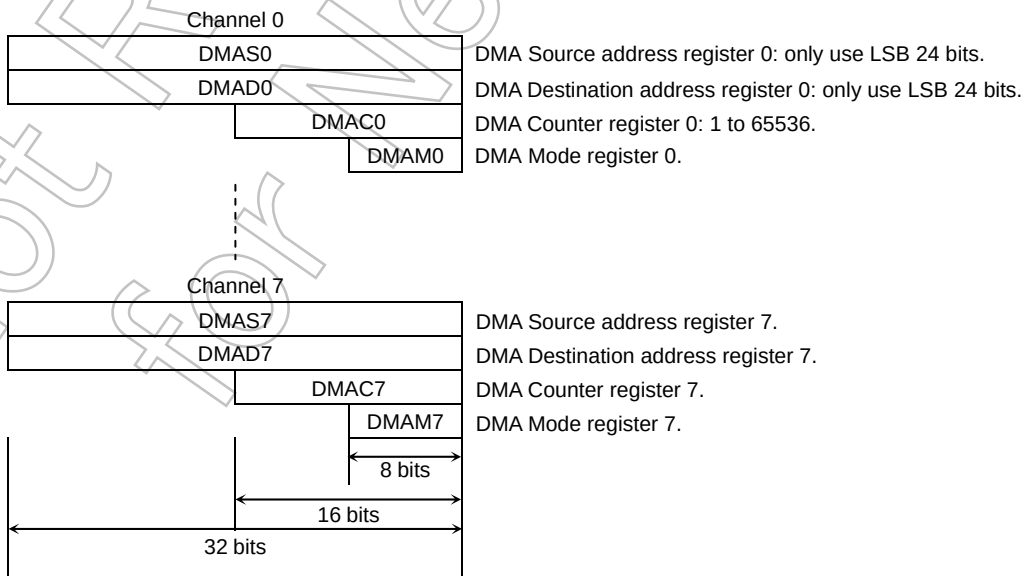
When writing again 1 to the DMAR register, check whether the bit is 0 before writing 1. If read "1", micro DMA transfer isn't started yet.

When a burst is specified by DMAB register, data is continuously transferred until the value in the micro DMA transfer counter is "0" after start up of the micro DMA. 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.

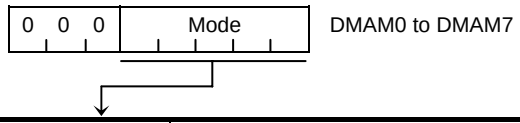
| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| DMAR | DMA request | 109H (Prohibit RMW) | DREQ7 | DREQ6 | DREQ5 | DREQ4 | DREQ3 | DREQ2 | DREQ1 | DREQ0 |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: DMA request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

(3) Transfer control registers

The transfer source address and the transfer destination address are set in the following registers. Data setting for these registers is done by an "LDC cr, r" instruction.



(4) Detailed description of the transfer mode register




| DMAM [4:0] | Operation | Execution Time |
|---|---|---|
| 000 zz | Destination address INC mode<br>(DMADn +) ← (DMASn)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |
| 001 zz | Source address DEC mode<br>(DMADn –) ← (DMASn)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |
| 010 zz | Source address INC mode<br>(DMADn) ← (DMASn +)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |
| 011 zz | Source address DEC mode<br>(DMADn) ← (DMASn –)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |
| 100 zz | Source address INC mode<br>(DMADn +) ← (DMASn +)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 6 states |
| 101 zz | Source address DEC mode<br>(DMADn –) ← (DMASn –)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 6 states |
| 110 zz | Destination address fixed mode<br>(DMADn) ← (DMASn)<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |
| 111 00 | Counter mode<br>DMASn ← DMASn + 1<br>DMACn ← DMACn – 1<br>If DMACn = 0 then INTTC | 5 states |

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

Note 1: The execution state number shows number of best case (1-state memory access).
1 state = 50 ns (at internal 20 MHz)

Note 2: “n” shows micro DMA channel number (0 to 7).

### 3.4.3 Interrupt Controller Operation

The block diagram in Figure 3.4.3 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 33 interrupts channels there is an interrupt request flag (Consisting of a flip-flop), an interrupt priority setting register and a micro DMA start vector register. The interrupt request flag latches interrupt requests from the peripherals.

The flag is cleared to 0 in the following cases:

When reset occurs

When the CPU reads the channel vector after accepted its interrupt

When executing an instruction that clears the interrupt (Write DMA start vector to INTCLR register)

When the CPU receives a micro DMA request

When the micro DMA burst transfer is terminated

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0AD or INTE12). 6 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 interrupt request with the same level are generated at the same time, 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.

The interrupt controller sends the interrupt request with the highest priority among the simultaneous interrupts and its vector address to the CPU. The CPU compares the priority value <IFF2:0> in the status register by the interrupt request signal with the priority value set; if the latter is higher, the interrupt is accepted. Then the CPU sets a value higher than the priority value by 1 (+1) in the CPU SR<IFF2:0>. Interrupt request where the priority value equals or is higher than the set value are accepted simultaneously during the previous interrupt routine.

When interrupt processing is completed (after execution of the RETI instruction), the CPU restores the priority value saved in the stack before the interrupt was generated to the CPU SR<IFF2:0>.

The interrupt controller also has registers (8 channels) used to store the micro DMA start vector. Writing the start vector of the interrupt source for the micro DMA processing (See Table 3.4.1), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter register (e.g., DMAS and DMAD) prior to the micro DMA processing.

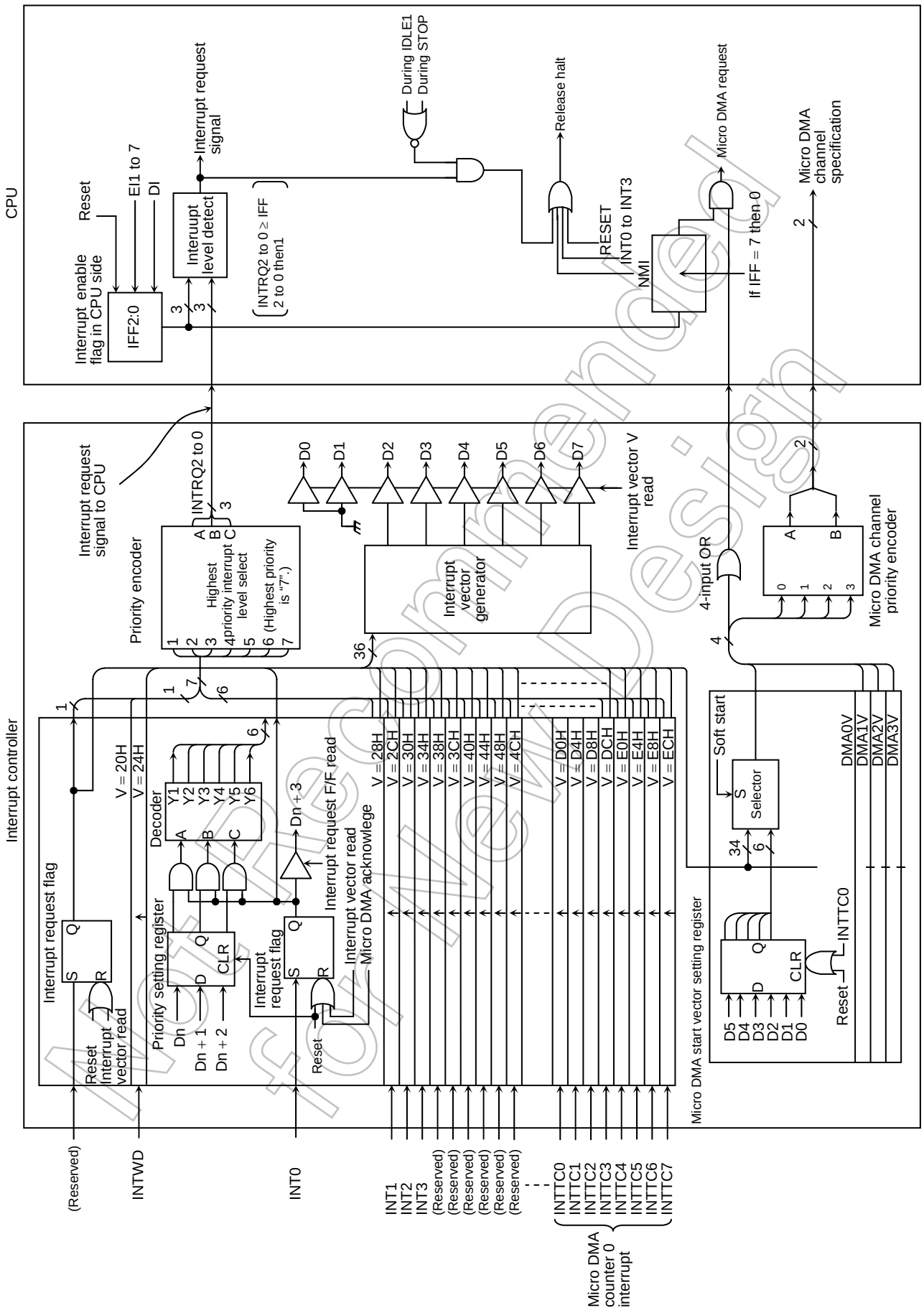



Figure 3.4.3 Block Diagram of Interrupt Controller

(1) Interrupt priority setting registers

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 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 |
| INTE3 | INT3 enable | D1H | – | | | | INT3 | | | |
| | | | – | – | – | – | I3C | I3M2 | I3M1 | I3M0 |
| | | | – | – | | | R | R/W | | |
| | | | Note: 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 | INTAT3 (TMRA3) | | | | INTAT2 (TMRA2) | | | |
| | | | ITA3C | ITA3M2 | ITA3M1 | ITA3M0 | ITA2C | ITA2M2 | ITA2M1 | ITA2M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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 |
| INTETBO0 | INTTBO0 (Overflow) enable | DAH | – | | | | INTTBO0 (TMRB0) | | | |
| | | | – | – | – | – | ITBO0C | ITBO0M2 | ITBO0M1 | ITBO0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | Note: Always write “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 |
| INTE45 | INT4& INT5 enable | E0H | INT5 | | | | INT4 | | | |
| | | | I5C | I5M2 | I5M1 | I5M0 | I4C | I4M2 | I4M1 | I4M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTETB1 | INTTB10& INTTB11 enable | E1H | 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 |
| INTETBO1 | INTTBO1 (Overflow) enable | E2H | – | | | | INTTBO1 (TMRB1) | | | |
| | | | – | – | – | – | ITBO1C | ITBO1M2 | ITBO1M1 | ITBO1M0 |
| | | | – | – | | | R | R/W | | |
| | | | Note: Always write “0”. | | | | 0 | 0 | 0 | 0 |
| INTESB0 | INTSBE0 enable | E3H | – | | | | INTSBE0 | | | |
| | | | – | – | – | – | ISBE0C | ISBE0M2 | ISBE0M1 | ISBE0M0 |
| | | | – | – | | | R | R/W | | |
| | | | Note: Always write “0”. | | | | 0 | 0 | 0 | 0 |
| INTEP0 | INTP0 enable | EEH | – | | | | INTP0 | | | |
| | | | – | – | – | – | IP0C | IP0M2 | IP0M1 | IP0M0 |
| | | | – | – | | | R | R/W | | |
| | | | Note: Always write “0”. | | | | 0 | 0 | 0 | 0 |

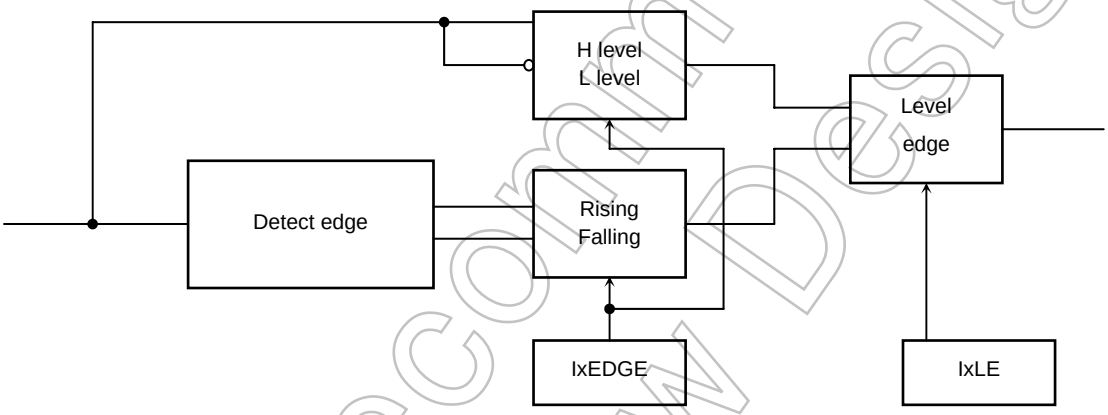
| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| INTE0AD | INT0&INTAD enable | F0H | INTAD | | | | INT0 | | | |
| | | | IADC | IADM2 | IADM1 | IADM0 | I0C | I0M2 | I0M1 | I0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTETC01 | INTTC0& INTTC1 enable | F1H | INTTC1 (DMA1) | | | | INTTC0 (DMA0) | | | |
| | | | ITC1C | ITC1M2 | ITC1M1 | ITC1M0 | ITC0C | ITC0M2 | ITC0M1 | ITC0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTETC23 | INTTC2& INTTC3 enable | F2H | INTTC3 (DMA3) | | | | INTTC2 (DMA2) | | | |
| | | | ITC3C | ITC3M2 | ITC3M1 | ITC3M0 | ITC2C | ITC2M2 | ITC2M1 | ITC2M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTETC45 | INTTC4& INTTC5 enable | F3H | INTTC5 (DMA5) | | | | INTTC4 (DMA4) | | | |
| | | | ITC5C | ITC5M2 | ITC5M1 | ITC5M0 | ITC4C | ITC4M2 | ITC4M1 | ITC4M0 |
| | | | 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 | – | | |
| | | | Note: Always write “0”. | | | | 0 | – | – | – |

Interrupt request flag

| IxxM2 | IxxM1 | IxxM0 | Function (Write) |
|---|---|---|---|
| 0 | 0 | 0 | Disables interrupt request. |
| 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 request. |

(2) External interrupt control

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| IIMC | Interrupt input mode control | 00F6H (Prohibit RMW) | | | I3EDGE | I2EDGE | I1EDGE | I0EDGE | I0LE | NMIREE |
| | | | | | W | | | | R/W | |
| | | | | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | INT3EDGE 0: Rising/ high 1: Falling/ low | INT2EDGE 0: Rising/ high 1: Falling/ low | INT1EDGE 0: Rising/ high 1: Falling/ low | INT0EDGE 0: Rising/ high 1: Falling/ low | INT0 0: Edge 1: Level | NMI 0: Falling edge 1: Falling and rising edges |
| IIMC2 | Interrupt input mode control2 | 00FAH (Prohibit RMW) | | | | | I3LE | I2LE | I1LE | |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | |
| | | | | | | | INT3 0: Edge 1: Level | INT2 0: Edge 1: Level | INT1 0: Edge 1: Level | |




Note 1: Disable INT0 to INT3 before changing INT0 to 3 pins mode from “level” to “edge”.

Setting example for case of INT0:

```
DI
LD (IIMC) ,XXXXXX0-B      ;  Change from “level” to “edge”.
LD (INTCLR),0AH           ;  Clear 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: When release halt by INT0 to INT3 interrupt of level-mode in interrupt request enable, keep setting level by <IxEDGE> until be started interrupt process. If changed “level” before interrupt process starting, interrupt isn’t processed correctly.

Example: Case of set “H” level interrupt (<IxLE> = 1, <IxEDGE> = 0).
Keep “H” level until be started interrupt process. If changed to “L” level before interrupt process starting, interrupt isn’t processed correctly.

Table 3.4.2 Function Setting of External Interrupt Pin

| Interrupt Pin | Shared Pin | Mode | Setting Method |
|---|---|---|---|
| INT0 | PC3 | Rising edge | IIMC<I0LE> = 0, INT0EDGE = 0 |
| | | Falling edge | IIMC<I0LE> = 0, INT0EDGE = 1 |
| | | High level | IIMC<I0LE> = 1, INT0EDGE = 0 |
| | | Low level | IIMC<I0LE> = 1, INT0EDGE = 1 |
| INT1 | PC1 | Rising edge | IIMC2<I1LE> = 0, INT1EDGE = 0 |
| | | Falling edge | IIMC2<I1LE> = 0, INT1EDGE = 1 |
| | | High level | IIMC2<I1LE> = 1, INT1EDGE = 0 |
| | | Low level | IIMC2<I1LE> = 1, INT1EDGE = 1 |
| INT2 | PC5 | Rising edge | IIMC2<I2LE> = 0, INT2EDGE = 0 |
| | | Falling edge | IIMC2<I2LE> = 0, INT2EDGE = 1 |
| | | High level | IIMC2<I2LE> = 1, INT2EDGE = 0 |
| | | Low level | IIMC2<I2LE> = 1, INT2EDGE = 1 |
| INT3 | PC6 | Rising edge | IIMC2<I3LE> = 0, INT3EDGE = 0 |
| | | Falling edge | IIMC2<I3LE> = 0, INT3EDGE = 1 |
| | | High level | IIMC2<I3LE> = 1, INT3EDGE = 0 |
| | | Low level | IIMC2<I3LE> = 1, INT3EDGE = 1 |
| INT4 | PD0 | Rising edge | TB1MOD<TB1CPM1:0> = 0, 0 or 0,1 or 1, 0 |
| | | Falling edge | TB1MOD<TB1CPM1:0> = 1, 0 |
| INT5 | PD1 | Rising edge | – |

(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 | |
| | | | | | | | | | 1 | 1 |
| | | | | | | | | | 0: INTRX1 edge mode<br>1: INTRX1 level mode | 0: INTRX0 edge mode<br>1: INTRX0 level mode |

*INTRX1 level enables

| | |
|---|---|
| 0 | Detect edge INTRX1 |
| 1 | "H" level INTRX1 |

*INTRX0 rising edge enable

| | |
|---|---|
| 0 | Detect edge INTRX0 |
| 1 | "H" Level INTRX0 |

(4) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA start vector, as given in Table 3.4.1, 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) | | | CLRV5 | CLRV4 | CLRV3 | CLRV2 | CLRV1 | CLRV0 |
| | | | | | W | | | | | |
| | | | | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | Interrupt clear | | | | | |

(5) Micro DMA start vector registers

This register assigns micro DMA processing to which interrupt source. The interrupt source with a micro DMA start vector that matches the vector set in this register is assigned as the micro DMA start source.

When the micro DMA transfer counter value reaches "0", the micro DMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA start vector register is cleared, and the micro DMA start source for the channel is cleared. Therefore, to continue micro DMA processing, set the micro DMA start vector register again during the processing of the micro DMA transfer end interrupt.

If the same vector is set in the micro DMA start vector registers of more than one channel, the channel with the lowest number has a higher priority. Accordingly, if the same vector is set in the micro DMA start vector registers of two channels, the interrupt generated in the channel with the lower number is executed until micro DMA transfer is completed. If the micro DMA start vector for this channel is not set again, the next micro DMA is started for the channel with the higher number (Micro DMA 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) 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 | | | | | | | |

(7) 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 (Note), 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 placed after 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. Thus, when be changed interrupt request level to "0", change it after cleared corresponding interrupt request by INTCLR instruction.

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

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

| | |
|---|---|
| INT0 to INT3 level mode | In level mode INT0 to INT3 are not an edge-triggered interrupt. Hence, in level mode the interrupt request flip-flop for INT0 to INT3 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.<br>If the CPU enters the interrupt response sequence as a result of INT x (x = 0, 1, 2, or 3) going from 0 to 1, INTx must then be held at 1 until the interrupt response sequence has been completed. If INTx is set to Level mode so as to release a Halt state, INTx must be held at 1 from the time INTx 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 INTx to revert to 0 before the Halt state has been released.)<br>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.<br>DI<br>LD (IIMC), 00H ; Changes from level to edge.<br>LD (INTCLR), 0AH ; Clears interrupt request flag.<br>NOP ; Wait EI execution.<br>NOP<br>NOP<br>EI |
| INTRX | 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. |

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

INT0 to INT 3: Instructions which switch to level mode after an interrupt request has been generated in edge mode.

The pin input change from high to low after interrupt request has been generated in level mode. ("H" → "L", "L" →"H")

INTRX: Instruction which read the receive buffer.

## 3.5 Port Function

The TMP92CM22 features 50-bit settings which relate to the various I/O ports.

As well as general-purpose I/O port functionality, the port pins also have I/O functions which relate to the built-in CPU and internal I/Os. Table 3.5.1 lists the functions of each port pin.

Table 3.5.2 and Table 3.5.3 lists I/O registers and their specifications.

Table 3.5.1 Port Function (R: U = with pull-up resistor)

| Port Names | Pin Names | Number of Pins | Direction | R | Direction Setting Unit | Pin Names for Built-In Function |
|---|---|---|---|---|---|---|
| Port 1 | P10 to P17 | 8 | I/O | – | Bit | D8 to D15 |
| Port 4 | P40 to P47 | 8 | I/O* | – | Bit* | A0 to A7 |
| Port 5 | P50 to P57 | 8 | I/O* | – | Bit* | 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 | Output | – | (Fixed) | $\overline{WRLL}$ |
| | P72 | 1 | Output | – | (Fixed) | $\overline{WRLU}$ |
| | P73 | 1 | Output | – | (Fixed) | |
| | P74 | 1 | Output | – | (Fixed) | CLKOUT |
| | P75 | 1 | Output | – | (Fixed) | R/$\overline{W}$ |
| | P76 | 1 | I/O | – | Bit | $\overline{WAIT}$ |
| Port 8 | P80 | 1 | Output | – | (Fixed) | $\overline{CS0}$ |
| | P81 | 1 | Output | – | (Fixed) | $\overline{CS1}$ |
| | P82 | 1 | Output | – | (Fixed) | $\overline{CS2}$ |
| | P83 | 1 | Output | – | (Fixed) | $\overline{CS3}$ |
| Port 9 | P90 | 1 | I/O | – | Bit | SCK |
| | P91 | 1 | I/O | – | Bit | SO, SDA |
| | P92 | 1 | I/O | – | Bit | SI, SCL |
| Port A | PA0 | 1 | Input | U | (Fixed) | |
| | PA1 | 1 | Input | U | (Fixed) | |
| | PA2 | 1 | Input | U | (Fixed) | |
| | PA7 | 1 | Input | U | (Fixed) | |
| Port C | PC0 | 1 | I/O | – | Bit | TA0IN |
| | PC1 | 1 | I/O | – | Bit | INT1, TA1OUT |
| | PC3 | 1 | I/O | – | Bit | INT0 |
| | PC5 | 1 | I/O | – | Bit | INT2, TA3OUT |
| | PC6 | 1 | I/O | – | Bit | INT3, TB0OUT0 |
| Port D | PD0 | 1 | I/O | – | Bit | INT4, TB1IN0 |
| | PD1 | 1 | I/O | – | Bit | INT5, TB1IN1 |
| | PD2 | 1 | I/O | – | Bit | TB1OUT0 |
| | PD3 | 1 | I/O | – | Bit | TB1OUT1 |
| Port F | PF0 | 1 | I/O | – | Bit | TXD0 |
| | PF1 | 1 | I/O | – | Bit | RXD0 |
| | PF2 | 1 | I/O | – | Bit | SCLK0, $\overline{CTS0}$ |
| | PF3 | 1 | I/O | – | Bit | TXD1 |
| | PF4 | 1 | I/O | – | Bit | RXD1 |
| | PF5 | 1 | I/O | – | Bit | SCLK1, $\overline{CTS1}$ |
| | PF6 | 1 | I/O | – | Bit | |
| | PF7 | 1 | I/O | – | Bit | |
| Port G | PG0 | 1 | Input | – | (Fixed) | AN0 |
| | PG1 | 1 | Input | – | (Fixed) | AN1 |
| | PG2 | 1 | Input | – | (Fixed) | AN2 |
| | PG3 | 1 | Input | – | (Fixed) | AN3, $\overline{ADTRG}$ |
| | PG4 | 1 | Input | – | (Fixed) | AN4 |
| | PG5 | 1 | Input | – | (Fixed) | AN5 |
| | PG6 | 1 | Input | – | (Fixed) | AN6 |
| | PG7 | 1 | Input | – | (Fixed) | AN7 |

*: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port.
All of general-purpose I/O ports except for port that used as address bus are operated as output port.
Please be careful when using this setting.

Table 3.5.2 I/O Port Setting List (1/2)

| Ports | Input Pins | Specification | I/O Register Setting Value | | | |
|---|---|---|---|---|---|---|
| | | | Pn | PnCR | PnFC | PnODE |
| Port 1 | P10 to P17 | Input port | × | 0 | 0 | None |
| | | Output port | × | 1 | | |
| | | D8 to D15 bus | × | × | 1 | |
| Port 4 | P40 to P47 | Input port* | × | 0* | 0 | None |
| | | Output port* | × | 1* | | |
| | | A0 to A7 output | × | × | 1 | |
| Port 5 | P50 to P57 | Input port* | × | 0* | 0 | None |
| | | Output port* | × | 1* | | |
| | | A8 to A15 output | × | × | 1 | |
| Port 6 | P60 to P67 | Input port* | × | 0* | 0 | None |
| | | Output port* | × | 1* | | |
| | | A16 to A23 output | × | × | 1 | |
| Port 7 | P70 to P75 | Output port | × | None | 0 | None |
| | P70 | $\overline{RD}$ output | × | None | 1 | |
| | P71 | $\overline{WRLL}$ output | | | | |
| | P72 | $\overline{WRLU}$ output | | | | |
| | P74 | CLKOUT output | | | | |
| | P75 | R/$\overline{W}$ output | | | | |
| | P76 | Input port | × | 0 | 0 | |
| | | Output port | × | 1 | 0 | |
| | | $\overline{WAIT}$ Input | × | 0 | 1 | |
| Port 8 | P80 to P83 | Output port | × | None | 0 | None |
| | P80 | $\overline{CS0}$ output | × | | 1 | |
| | P81 | $\overline{CS1}$ output | × | | 1 | |
| | P82 | $\overline{CS2}$ output | × | | 1 | |
| | P83 | $\overline{CS3}$ output | × | | 1 | |
| Port 9 | P90 to P92 | Input port | × | 0 | 0 | 0 |
| | | Output port | × | 1 | 0 | 0 |
| | P90 | SCK input | × | 0 | 0 | 0 |
| | | SCK output | × | × | 1 | 0/1 |
| | P91 | SO output | × | 1 | 1 | 0/1 |
| | | SDA | × | × | 1 | 1 |
| | P92 | SI input | × | 0 | 0 | 0 |
| | | SCL (Open drain) | × | × | 1 | 1 |

X: Don't care

*: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port.
All of general-purpose I/O ports except for port that used as address bus are operated as output port.
Please be careful when using this setting.

Table 3.5.3 I/O Port Setting List (2/2)

| Ports | Input Pins | Specification | I/O Register Setting Value | | | |
|---|---|---|---|---|---|---|
| | | | Pn | PnCR | PnFC | PnODE |
| Port A | PA0, PA1, PA2, PA7 | Input port | × | None | None | None |
| Port C | PC0, PC1, PC3, PC5, PC6 | Input port | × | 0 | 0 | None |
| | | Output port | × | 1 | 0 | |
| | PC0 | TA0IN input | × | × | 1 | |
| | PC1 | TA1OUT output | × | 1 | 1 | |
| | | INT1 input | × | 0 | 1 | |
| | PC3 | INT0 input | × | × | 1 | |
| | PC5 | INT2 input | × | 0 | 1 | |
| | | TA3OUT | × | 1 | 1 | |
| | PC6 | INT3 input | × | 0 | 1 | |
| | | TB0OUT0 | × | 1 | 1 | |
| Port D | PD0 to PD3 | Input port | × | 0 | 0 | None |
| | | Output port | × | 1 | 0 | |
| | PD0 | TB1IN0, INT4 input | × | 0 | 1 | |
| | PD1 | TB1IN1, INT5 input | × | 0 | 1 | |
| | PD2 | TB0OUT0 output | × | 1 | 1 | |
| | PD3 | TB0OUT1 output | × | 1 | 1 | |
| Port F | PF0 to PF7 | Input port | × | 0 | 0 | None |
| | | Output port | × | 1 | 0 | |
| | PF0 | TXD0 (Open drain) | × | 0 | 1 | |
| | | TXD0 | × | 1 | 1 | |
| | PF1 | RXD0 input | × | 0 | None | |
| | PF2 | SCLK0 input/output | × | 0/1 | 1 | |
| | | $\overline{\text{CTS0}}$ input | × | 0 | 1 | |
| | PF3 | TXD1 (Open drain) | × | 0 | 1 | |
| | | TXD1 | × | 1 | 1 | |
| | PF4 | RXD1 input | × | 0 | None | |
| | PF5 | SCLK1 input/output | × | 0/1 | 1 | |
| | | $\overline{\text{CTS1}}$ input | × | 0 | 1 | |
| Port G | PG0 to PG7 | Input port | × | None | None | None |
| | | AN0 to AN7 input | × | | | |
| | PG3 | $\overline{\text{ADTRG}}$ input | × | | | |

X: Don't care

By resetting, these port pins become general-purpose input port.
I/O pin is reset to input pin. When use built-in function, process all function by software.

### 3.5.1 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).

After released reset, device set port1 to pins of follow function by combination of AM1 and AM0 pins.

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

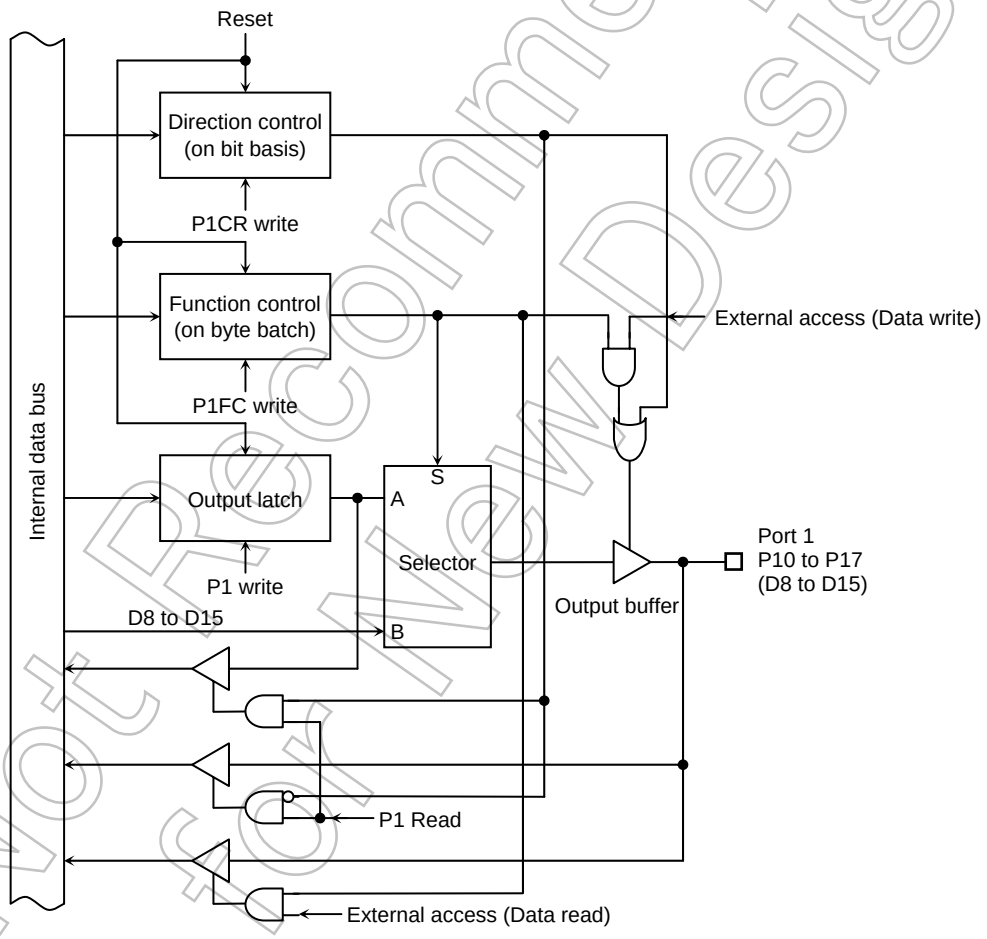



Figure 3.5.1 Port 1

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 | | | | | | | |
| After reset | Data from external port (Output latch register is clear to “0”.) | | | | | | | |

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 | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Refer to port 1 function setting | | | | | | | |

Port 1 Function Register

P1FC (0007H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | | | P1F |
| Read/Write | | | | | | | | W |
| After reset | | | | | | | | 0/1 Note3 |
| Function | Refer to port 1 function setting | | | | | | | |

Port 1 Function setting

| P1FC<P1F> / P1CR<P1xC> | 0 | 1 |
|---|---|---|
| 0 | Input port | Data bus (D15 to D8) |
| 1 | Output port | Don’t use this setting |

Note 1: Read-modify-write instruction is prohibited for registers P1FC and P1CR.

Note 2: <P1XC> shows “X bit” of P1CR register.

Note 3: It is set to “Port” or “Data bus” by AM pin setting.

Figure 3.5.2 Register for Port 1

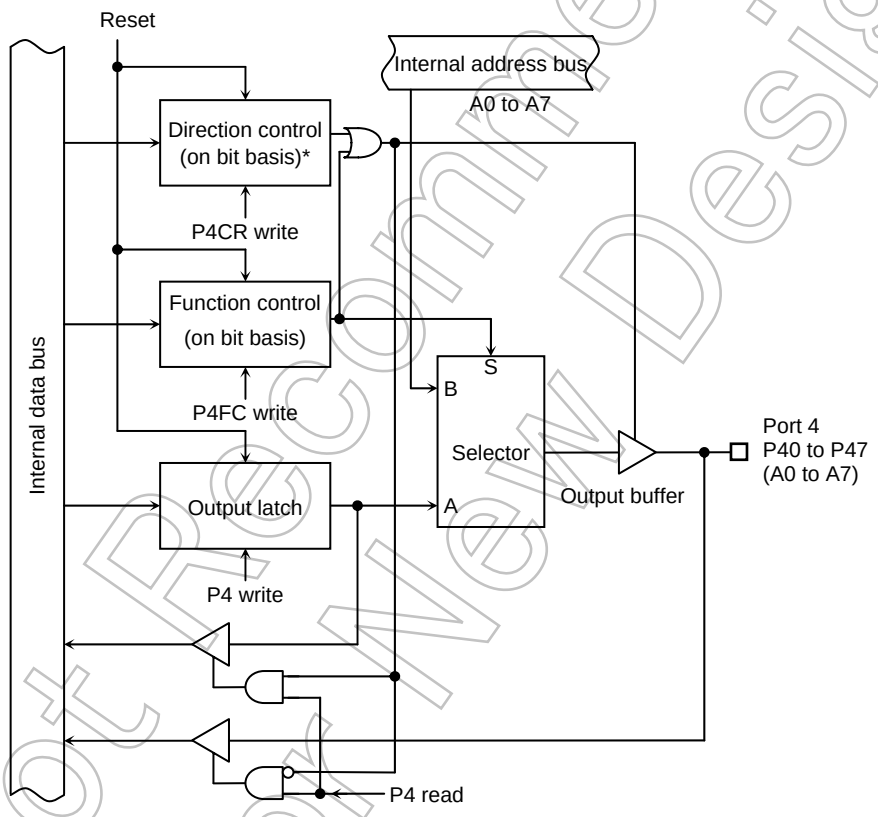
### 3.5.2 Port 4 (P40 to P47)

Port 4 is an 8-bit general-purpose I/O port*. Bits can be individually set as either inputs or outputs by control register P4CR and function register P4FC*.

In addition to functioning as a general-purpose I/O port, port 4 can also function as a address bus (A0 to A7).

After released reset, device set Port 4 to pins of follow function by combination of AM1 and AM0 pins.

| AM1 | AM0 | Function Setting after Reset |
|---|---|---|
| 0 | 0 | Don't use this setting |
| 0 | 1 | Address bus (A0 to A7) |
| 1 | 0 | Address bus (A0 to A7) |
| 1 | 1 | Don't use this setting |




*: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port.
All of general-purpose I/O ports except for port that used as address bus are operated as output port.
Please be careful when using this setting.

Figure 3.5.3 Port 4

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 | | | | | | | |
| After reset | Data from external port (Output latch register is cleared to “0”.) | | | | | | | |

Port 4 Control Register

P4CR (0012H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P47C | P46C | P45C | P44C | P43C | P42C | P41C | P40C |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | 0: Input 1: Output (Note2) | | | | | | | |

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 | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Function | 0: Port 1: Address bus (A0 to A7) | | | | | | | |

Note1: Read-modify-write instruction is prohibited for registers P4CR and P4FC.

Note2: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port. All of general-purpose I/O ports except for port that used as address bus are operated as output port. Please be careful when using this setting.

Figure 3.5.4 Register for Port 4

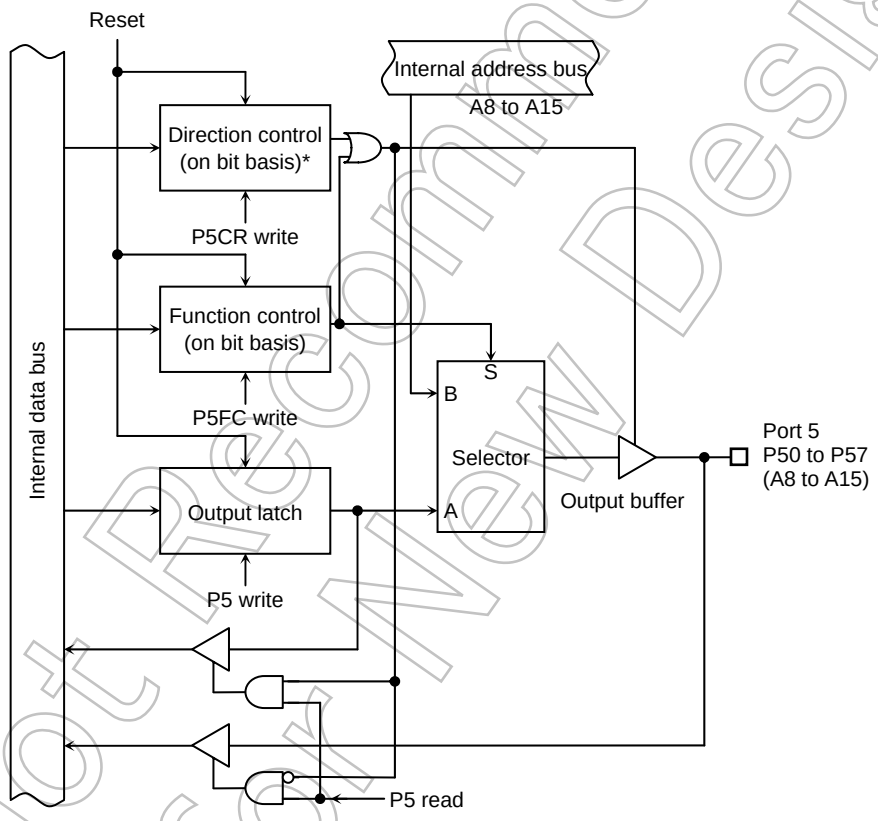
### 3.5.3 Port 5 (P50 to P57)

Port 5 is an 8-bit general-purpose I/O port*. Bits can be individually set as either inputs or outputs by control register P5CR and function register P5FC*.

In addition to functioning as a general-purpose I/O port, port 5 can also function as an address bus (A8 to A15).

After released reset, device set port 5 to pins of follow function by combination of AM1 and AM0 pins.

| AM1 | AM0 | Function Setting after Reset |
|---|---|---|
| 0 | 0 | Don't use this setting |
| 0 | 1 | Address bus (A8 to A15) |
| 1 | 0 | Address bus (A8 to A15) |
| 1 | 1 | Don't use this setting |




*: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port.
All of general-purpose I/O ports except for port that used as address bus are operated as output port.
Please be careful when using this setting.

Figure 3.5.5 Port 5

Port 5 Register

P5 (0014H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 |
| Read/Write | R/W | | | | | | | |
| After reset | Data from external port (Output latch register is cleared to “0”.) | | | | | | | |

Port 5 Control Register

P5CR (0016H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P57C | P56C | P55C | P54C | P53C | P52C | P51C | P50C |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | 0: Input 1: Output (Note2) | | | | | | | |

Port 5 Function Register

P5FC (0017H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P57F | P56F | P55F | P54F | P53F | P52F | P51F | P50F |
| Read/Write | W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Function | 0: Port 1: Address bus (A8 to A15) | | | | | | | |

Note1: Read-modify-write instruction is prohibited for registers P5CR and P5FC.

Note2: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port. All of general-purpose I/O ports except for port that used as address bus are operated as output port. Please be careful when using this setting.

Figure 3.5.6 Register for Port 5

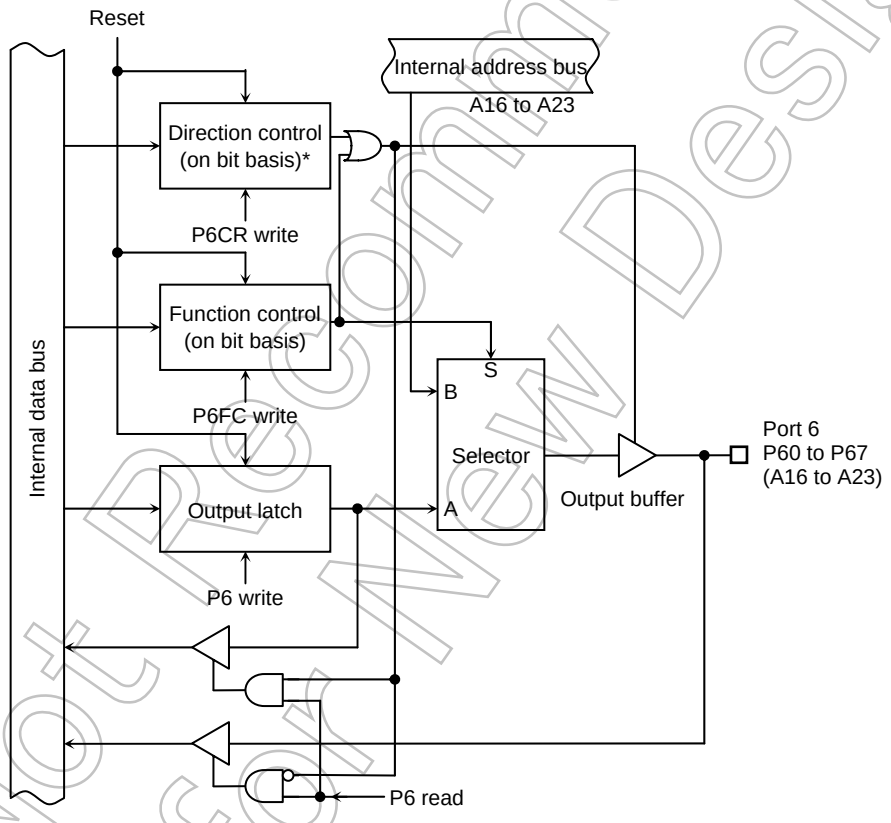
### 3.5.4 Port 6 (P60 to P67)

Port 6 is an 8-bit general-purpose I/O port*. Bits can be individually set as either inputs or outputs by control register P6CR and function register P6FC*.

In addition to functioning as a general-purpose I/O port, port 6 can also function as an address bus (A16 to A23).

After released reset, device set port 6 to pins of follow function by combination of AM1 and AM0 pins.

| AM1 | AM0 | Function Setting after Reset |
|---|---|---|
| 0 | 0 | Don't use this setting |
| 0 | 1 | Address bus (A16 to A23) |
| 1 | 0 | Address bus (A16 to A23) |
| 1 | 1 | Don't use this setting |




*: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port.
All of general-purpose I/O ports except for port that used as address bus are operated as output port.
Please be careful when using this setting.

Figure 3.5.7 Port 6

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 | | | | | | | |
| After reset | Data from external port (Output latch register is cleared to “0”.) | | | | | | | |

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 | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | 0: Input 1: Output (Note2) | | | | | | | |

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 | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Function | 0: Port 1: Address bus (A16 to A23) | | | | | | | |

Note1: Read-modify-write instruction is prohibited for registers P6CR and P6FC.

Note2: When these ports are used as general-purpose I/O port, each bit can be set individually for input or output. However, each bit cannot be set individually for input or output even if 1bit or more bits are used as address bus in same port. All of general-purpose I/O ports except for port that used as address bus are operated as output port. Please be careful when using this setting.

Figure 3.5.8 Register for Port 6

### 3.5.5 Port 7 (P70 to P76)

Port 7 is a 7-bit general-purpose I/O port (P70 to P75 are 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 P73 pins can also function as output pin of read/write strobe signals to connect with an external memory. P74 pin can also function as CLKOUT output pin when outputted internal clock. P76 pin can also function as wait input.

After reset, P71 to P75 pins are set to output port mode, and P76 pin is set to input port mode.

P70 pin set port 1 to pins of follow function by combination of AM1 and AM0 pins.

| AM1 | AM0 | Function Setting after Reset |
|---|---|---|
| 0 | 0 | Don't use this setting |
| 0 | 1 | CPU control pin ( $\overline{RD}$ ) |
| 1 | 0 | CPU control pin ( $\overline{RD}$ ) |
| 1 | 1 | Don't use this setting |

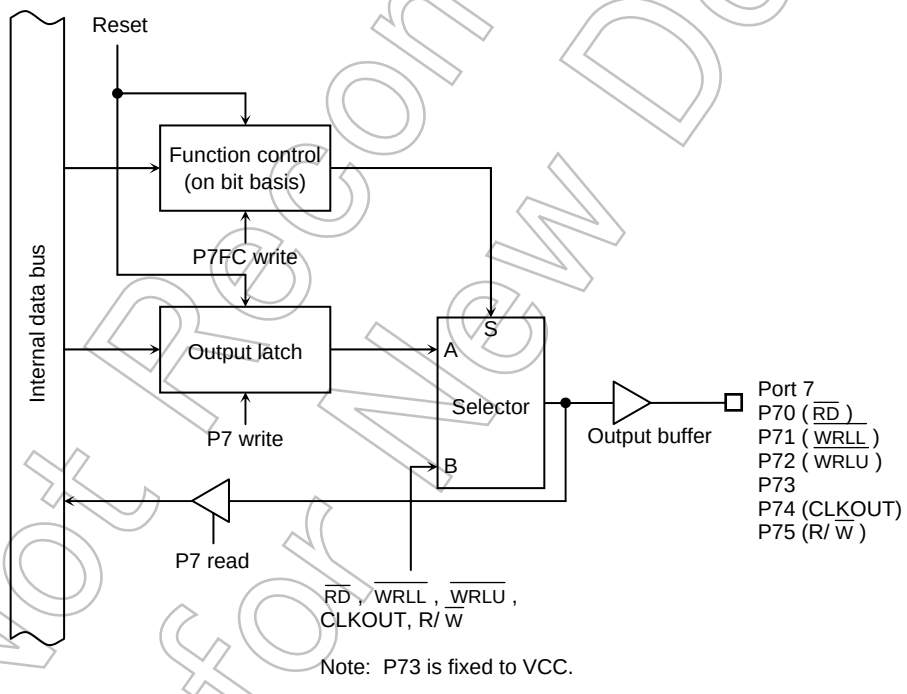



Figure 3.5.9 Port 7 (P70 to P75)

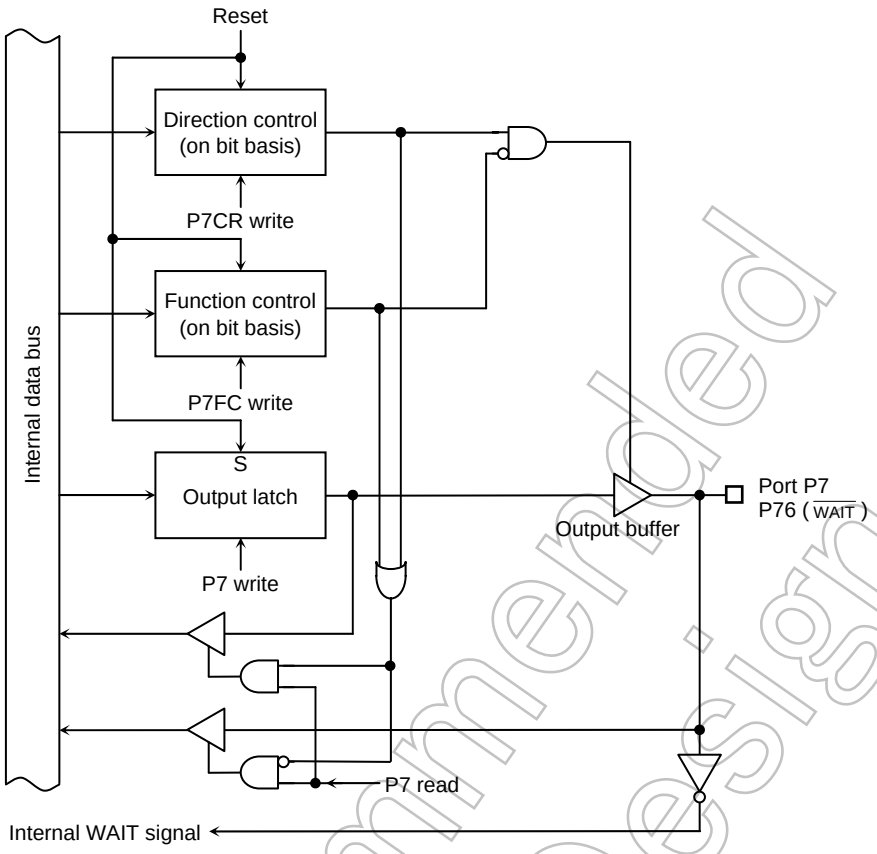



Figure 3.5.10 Port 7 (P76)

Port 7 Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | P76 | P75 | P74 | P73 | P72 | P71 | P70 |
| Read/Write | | R/W | | | | | | |
| After reset | | Data from external port (Note) | 1 | 1 | 1 | 1 | 1 | 1 |

P7 (001CH)

Note: Output latch register is cleared to 0.

Port 7 Control Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | P76C | | | | | | |
| Read/Write | | W | | | | | | |
| After reset | | 0 | | | | | | |
| Function | | 0: Input<br>1: Output | | | | | | |

P7CR (001EH)

Port 7 Function Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | P76F | P75F | P74F | P73F | P72F | P71F | P70F |
| Read/Write | | W | | | | | | |
| After reset | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Function | | 0: Port<br>1: $\overline{WAIT}$ | 0: Port<br>1: R/$\overline{W}$ | 0: Port<br>1: CLKOUT | 0: Port<br>1: Don't set | 0: Port<br>1: $\overline{WRLU}$ | 0: Port<br>1: $\overline{WRLL}$ | 0: Port<br>1: $\overline{RD}$ |

P7FC (001FH)

Note: Read-modify-write instruction is prohibited for registers P7CR and P7FC.

Figure 3.5.11 Register for Port 7

### 3.5.6 Port 8 (P80 to P83)

Port 8 is 4-bit output port. Resetting sets output latch of P82 to "0" and set output latches of P80, P81, and P83 to "1".

In addition to functioning as a output port, port 8 can also function as a output chip select signal ($\overline{CS0}$ to $\overline{CS3}$).

These settings operate by programming "1" to the corresponding bit of P8FC.

Resetting set all bits of P8FC to "0", these pits set output mode.

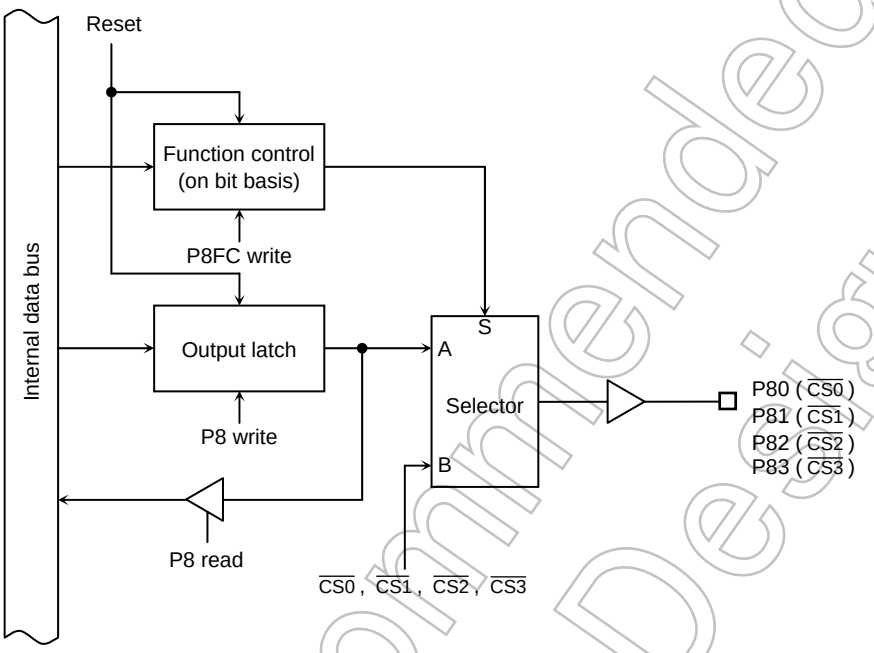



Figure 3.5.12 Port 8

Port 8 Register

| P8 (0020H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | P83 | P82 | P81 | P80 |
| Read/Write | | | | | R/W | | | |
| After reset | | | | | 1 | 0 | 1 | 1 |

Port 8 Function Register

| P8FC (0023H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | P83F | P82F | P81F | P80F |
| Read/Write | | | | | W | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | 0: Port<br>1: $\overline{CS3}$ | 0: Port<br>1: $\overline{CS2}$ | 0: Port<br>1: $\overline{CS1}$ | 0: Port<br>1: $\overline{CS0}$ |

Note 1: Read-modify-write instruction is prohibited for the registers P8FC.

Note 2: When set P82 pin as $\overline{CS2}$ after release reset, set function register (P8FC<P82F> = 1) in keep output latch of P82 to "0" (P8<P82> = 0).

If set function register (P8FC<P82F> = 1) after set output latch to "1" (P8<P82> = 1), maybe operation become to error because $\overline{CS2}$ output don't output correctly.

Figure 3.5.13 Register for Port 8

### 3.5.7 Port 9 (P90 to P92)

Port 9 is 3-bit general-purpose I/O port. Each bit can be set individually for input or output.

In addition to functioning as a general-purpose I/O port, port 9 can also function as a serial bus interface input (SCK (Clock signal in SIO mode), SO (Data output signal in SIO mode), SDA (Data signal in I²C bus mode), SI (Data input signal in SIO mode) and SCL (Clock signal in I²C bus mode)).

These settings operate by programming to the corresponding bit of P9FC.

Resetting set value of P9CR and P9FC to "0", all bits are set to input port. And all bits of output latch are set to "1".

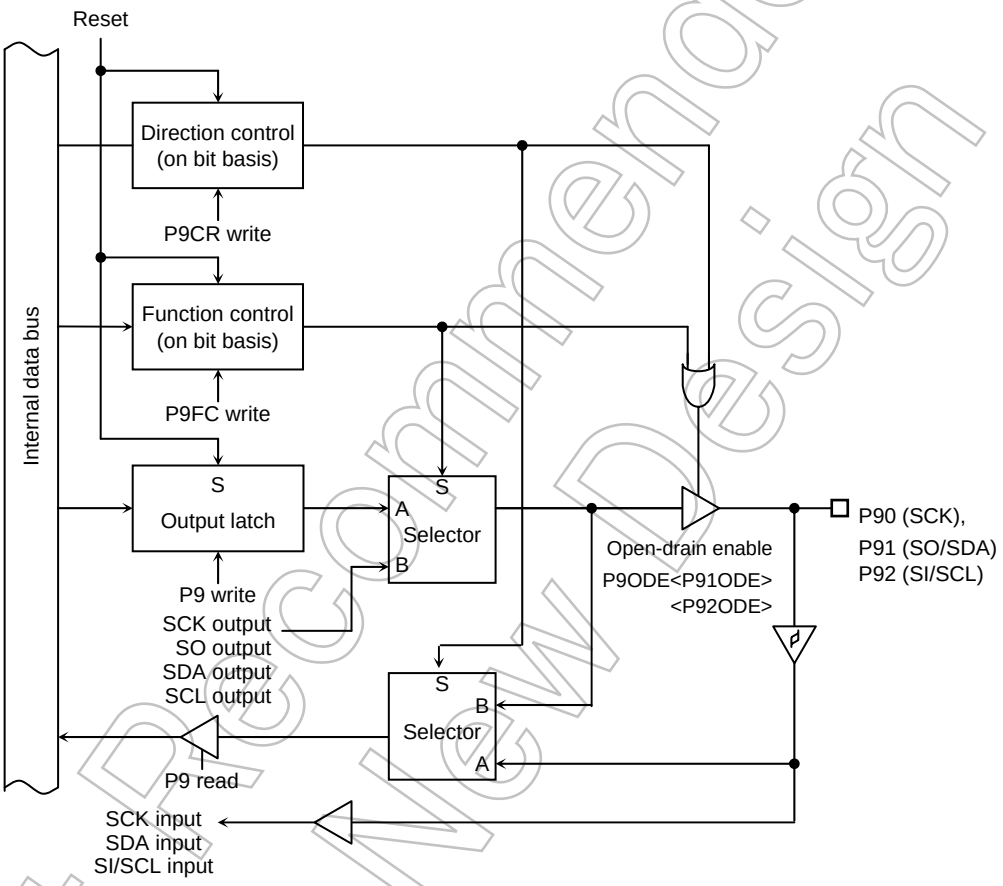



Figure 3.5.14 Port 9 (P90 to P92)

Port 9 Register

P9 (0024H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | P92 | P91 | P90 |
| Read/Write | | | | | | R/W | | |
| After reset | | | | | | Data from external port (Output latch register is set to 1) | | |

Port 9 Control Register

P9CR (0026H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | P92C | P91C | P90C |
| Read/Write | | | | | | W | | |
| After reset | | | | | | 0 | 0 | 0 |
| Function | | | | | | 0: Input 1: Output | | |

Port 9 Function Register

P9FC (0027H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | P92F | P91F | P90F |
| Read/Write | | | | | | W | | |
| After reset | | | | | | 0 | 0 | 0 |
| Function | | | | | | 0: Port, SI<br>1: SCL<br>Note | 0: Port<br>1: SO, SDA | 0: Port, SCK input<br>1:SCK output<br>Note |

Port 9 ODE Register

P9ODE (0025H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | P92ODE | P91ODE | |
| Read/Write | | | | | | W | | |
| After reset | | | | | | 0 | 0 | |
| Function | | | | | | 1:Open drain | 1:Open drain | |

Note1: Read-modify-write instruction is prohibited for the registers P9CR, P9FC, and P9ODE.

Note2: When using SI and SCK input function, set P9FC<P92F,P90F> to “0” (Function setting).

Figure 3.5.15 Register for Port 9

### 3.5.8 Port A (PA0 to PA2, PA7)

Port A is 4-bit general-purpose input port with pull-up resistor.

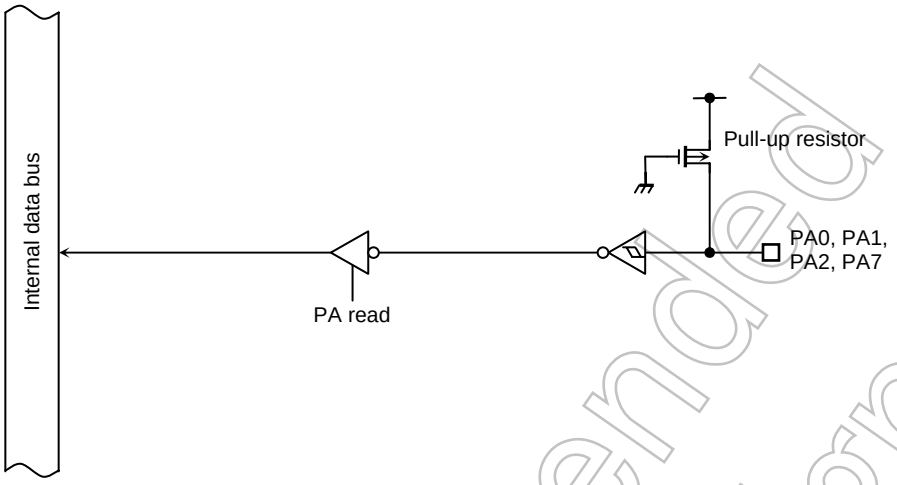



Figure 3.5.16 Port A

Port A Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| PA (0028H) | Bit symbol | PA7 | | | | | PA2 | PA1 | PA0 |
| | Read/Write | R | | | | | R | | |
| | After reset | Data from external port | | | | | Data from external port | | |

Figure 3.5.17 Register for Port A

### 3.5.9 Port C (PC0, PC1, PC3, PC5, and PC6)

Port C is 5-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets port C to input port.

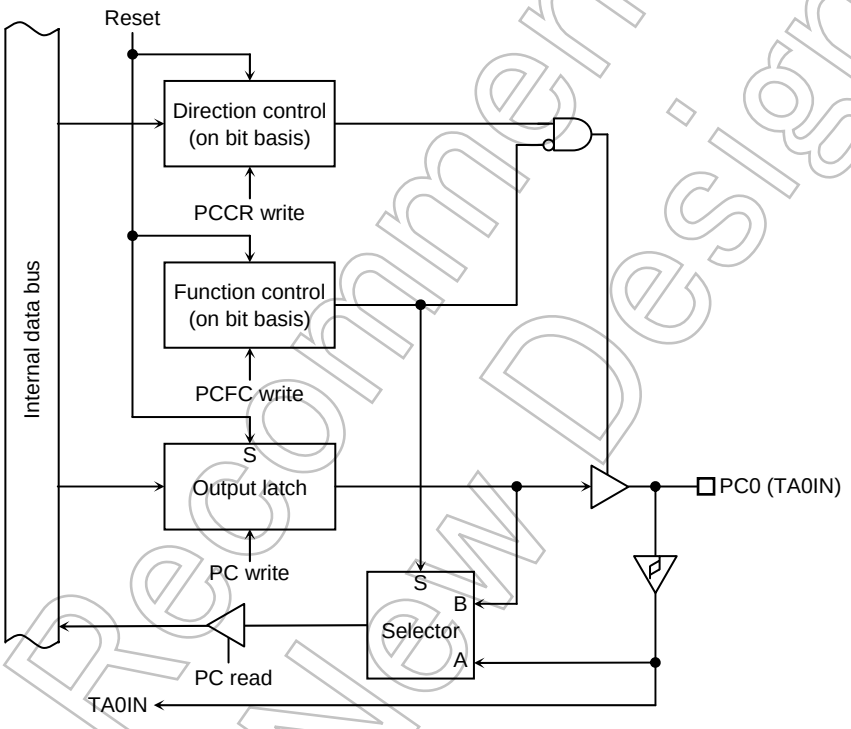
In addition to functioning as a general-purpose I/O port, port C can also function as a input/output pin (TA0IN, TA1OUT, TA3OUT, and TB0OUT0) and external interrupt pin (INT0 to INT3).

These settings operate by programming "1" to the corresponding bit of PCCR and PCFC.

Resetting resets the PCCR and PCFC to "0", and sets all bits to input port.

(1) PC0 (TA0IN)

In addition to function as I/O port, port PC0 can also function as input pin TA0IN of timer channel 0.

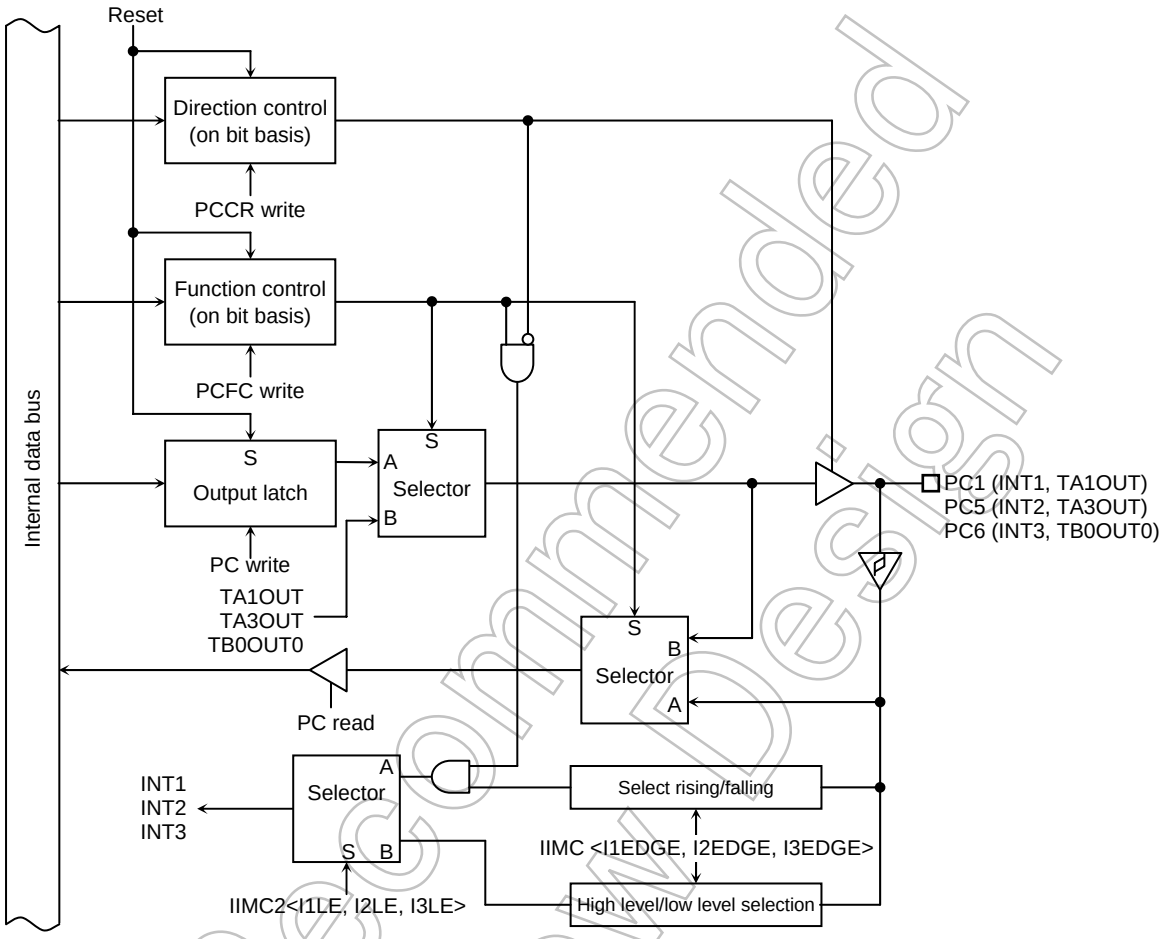



Note: Can not read the output latch data when output mode.

Figure 3.5.18 Port C (PC0)

(2) PC1 (INT1, TA1OUT), PC5 (INT2, TA3OUT), PC6 (INT3, TB0OUT0)

In addition to function as I/O port, port PC1, PC5, and PC6 can also function as external interrupt input pin INT1 to INT3 and output pin of timer channel TA1OUT, TA3OUT, and TB0OUT0.




Note: Can not read the output latch data when output mode.

Figure 3.5.19 Port C (PC1, PC5, and PC6)

(3) PC3 (INT0)

In addition to function as I/O port, port PC3 can also function as external interrupt pin INT0.

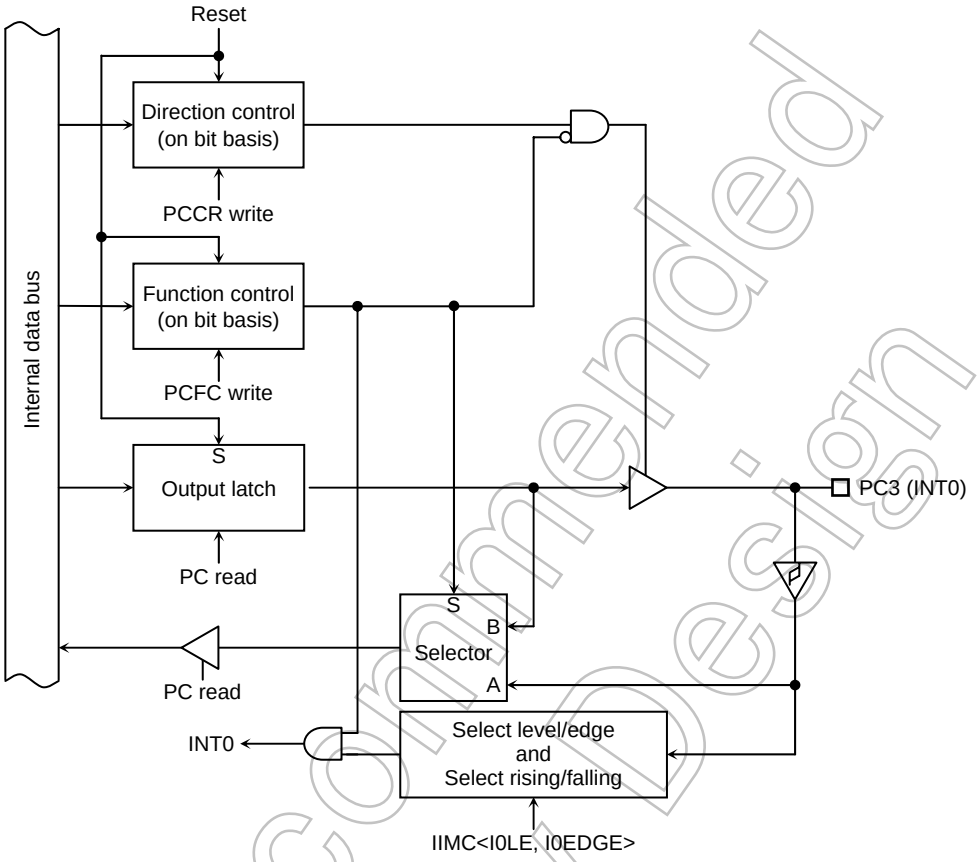



Figure 3.5.20 Port C (PC3)

Port C Register

PC (0030H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | PC6 | PC5 | | PC3 | | PC1 | PC0 |
| Read/Write | | R/W | | | R/W | | R/W | |
| After reset | | Data from external port (Note) | | | Data from external port (Note) | | Data from external port (Note) | |

Note: Output latch register is set to 1.

Port C Control Register

PCCR (0032H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | PC6C | PC5C | | PC3C | | PC1C | PC0C |
| Read/Write | | W | | | W | | W | |
| After reset | | 0 | 0 | | 0 | | 0 | 0 |
| Function | | 0: Input 1: Output | | | 0: Input 1: Output | | 0: Input 1: Output | |

Port C Function Register

PCFC (0033H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | PC6F | PC5F | | PC3F | | PC1F | PC0F |
| Read/Write | | W | | | W | | W | |
| After reset | | 0 | 0 | | 1 | | 0 | 0 |
| Function | | 0: Port 1: INT3 TB0OUT0 | 0: Port 1: INT2 TA3OUT | | 0: Port 1: INT0 | | 0: Port 1: INT1 TA1OUT | 0: Port 1: TA0IN |

INT1, TA1OUT setting

| <PC1C> / <PC1F> | 0 | 1 |
|---|---|---|
| 0 | Input port | Output port |
| 1 | INT1 | TA1OUT |

INT2, TA3OUT Setting

| <PC5C> / <PC5F> | 0 | 1 |
|---|---|---|
| 0 | Input port | Output port |
| 1 | INT2 | TA3OUT |

INT3, TB0OUT0 setting

| <PC6C> / <PC6F> | 0 | 1 |
|---|---|---|
| 0 | Input port | Output port |
| 1 | INT3 | TB0OUT0 |

Note 1: Read-modify-write instruction is prohibited for the registers PCCR and PCFC.

Note 2: PC0/TA0IN pins do not have a register changing PORT/FUNCTION. For example, when it is used as an input port, the input signal is inputted to 8-bit timer as the input 0.

Note 3: Can not read the output latch data when PC0, PC1, PC5, and PC6 are output mode.

Figure 3.5.21 Register for Port C

### 3.5.10 Port D (PD0 to PD3)

Port D is 4-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets port D to input port.

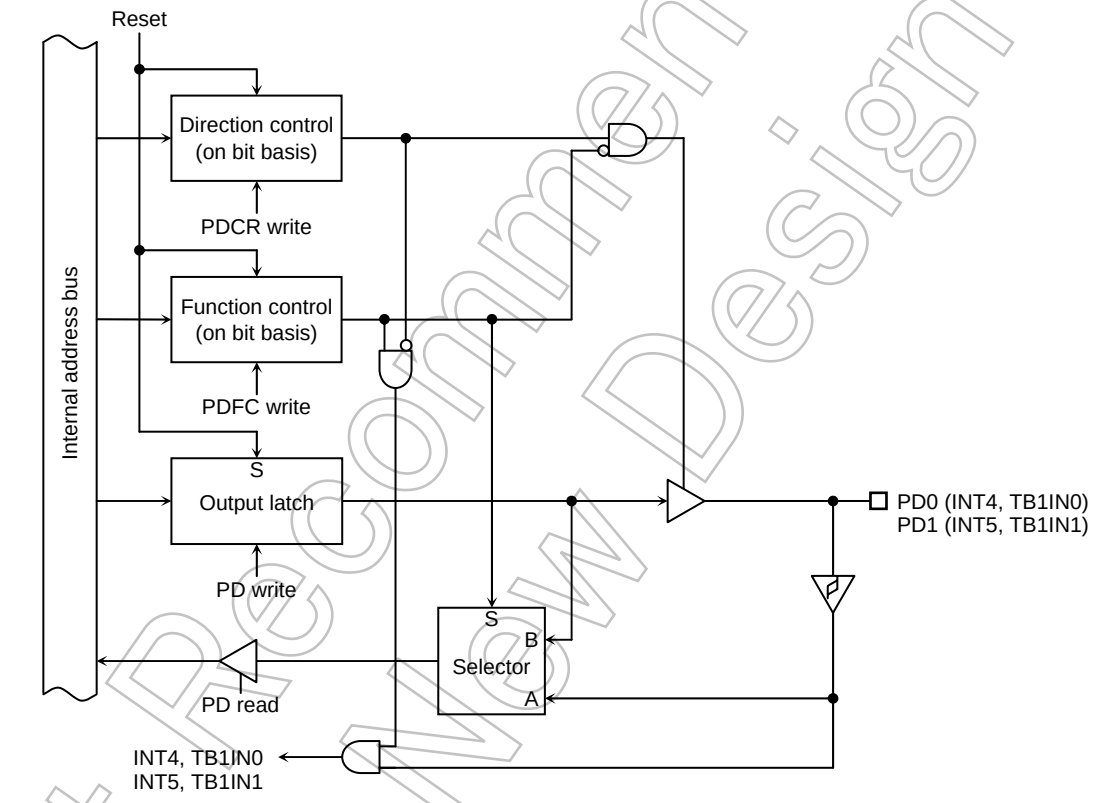
In addition to functioning as a general-purpose I/O port, port D can also function as an input pin (INT4 and INT5)/output pin (TB0IN, TB1OUT, TB3OUT, and TB1OUT1).

These settings operate by programming "1" to the corresponding bit of PDCR and PDFC.

Resetting resets the PDCR and PDFC to "0", and sets all bits to input port.

(1) PD0 (INT4, TB1IN0), PD1 (INT5, TB1IN1)

In addition to function as I/O port, port PD0 and PD1 can also function as external interrupt input pins INT4, INT5, timer channel input pins TB1IN0 and TB1IN1.




Note: Can not read the output latch data when output mode.

Figure 3.5.22 Port D (PD0 and PD1)

(2) PD2 (TB1OUT0) and PD3 (TB1OUT1)

In addition to function as I/O port, port PD0 and PD1 can also function as timer channel output pins TB1OUT0 and TB1OUT1.

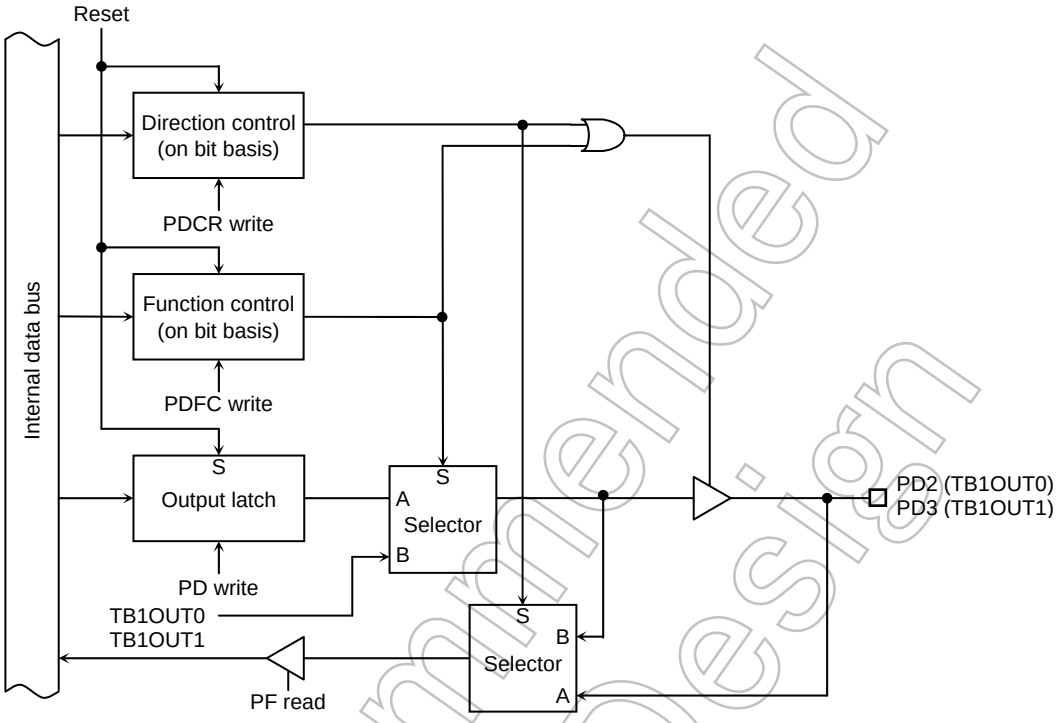



Figure 3.5.23 Port D (PD2 and PD3)

Port D Register

PD (0034H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | PD3 | PD2 | PD1 | PD0 |
| Read/Write | | | | | R/W | | | |
| After reset | | | | | Data from external port (Output latch register is set to 1) | | | |

Port D Control Register

PDCR (0036H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | PD3C | PD2C | PD1C | PD0C |
| Read/Write | | | | | W | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | 0: Input 1: Output | 0: Input 1: Output | 0: Input 1: Output | 0: Input 1: Output |

→ Port D I/O setting

| | |
|---|---|
| 0 | Input |
| 1 | Output |

Port D Function Register

PDFC (0037H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | PD3F | PD2F | PD1F | PD0F |
| Read/Write | | | | | W | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | 0: Port 1: TB1OUT1 | 0: Port 1: TB1OUT0 | 0: Port 1: TB0IN1 INT5 Input | 0: Port 1: TB0IN0 INT4 Input |

→ PD2 output setting asTB1OUT0

| | |
|---|---|
| PDFC<PD2F> | 1 |
| PDCR<PD2C> | 1 |

→ PD3 output setting as TB1OUT1

| | |
|---|---|
| PDFC<PD3F> | 1 |
| PDCR<PD3C> | 1 |

Note 1: Read-modify-write instruction is prohibited for the registers PDFC and PDCR.

Note 2: Can not read the output latch data when PD0 and PD1 are output mode.

Figure 3.5.24 Register for Port D

### 3.5.11 Port F (PF0 to PF7)

Port F is 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting resets the PFCR and PFFC to "0", and sets all bits to input port. And all bits of output latch register to "1".

In addition to functioning as a general-purpose I/O port, port F can also function as I/O function of serial channel 0 and 1.

These settings operate by writing "1" to the corresponding bit of PFFC.

Resetting resets the PDCR and PDFC to "0", and sets all bits to input port.

(1) Port PF0 and PF3 (TXD0/TXD1)

In addition to function as I/O port, port PF0 and PF3 can also function as TXD output pin of serial channel.

Thus, output buffer feature a programmable open-drain function, and setting enable by PFFC<PF0F, PF3F> and PFCR<PF0C, PF3C> register.

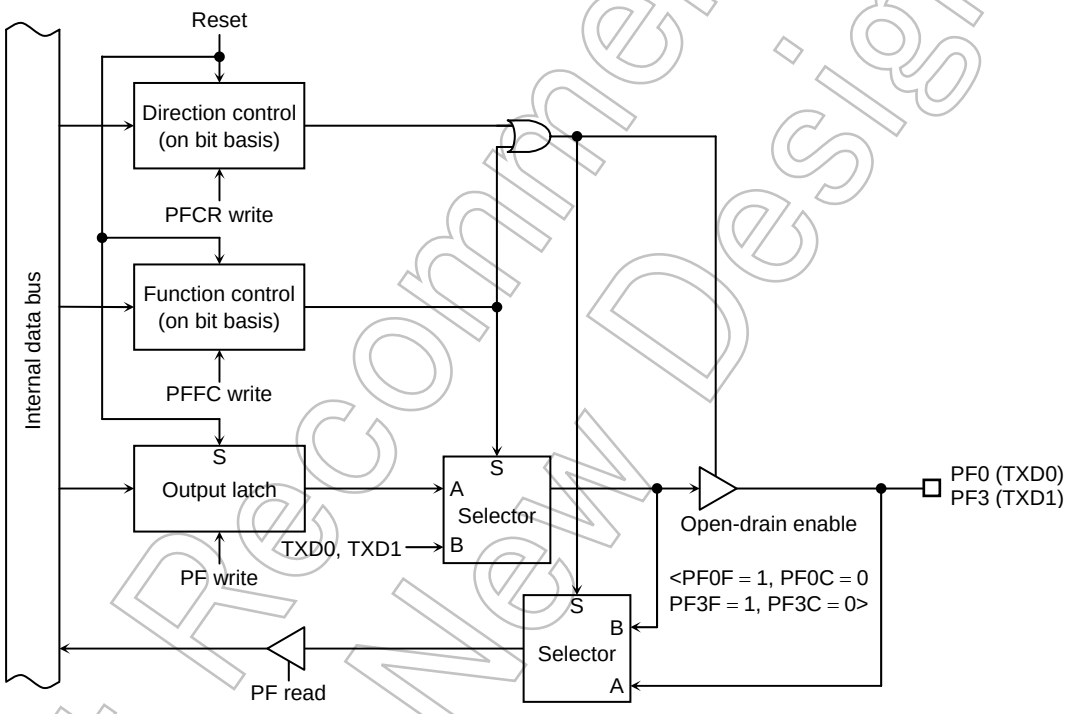



Figure 3.5.25 Port F (PF0 and PF3)

(2) Ports PF1 and PF4 (RXD0 and XD1)

In addition to function as I/O port, port PF1 and PF4 can also function as RXD input pin of serial channel.

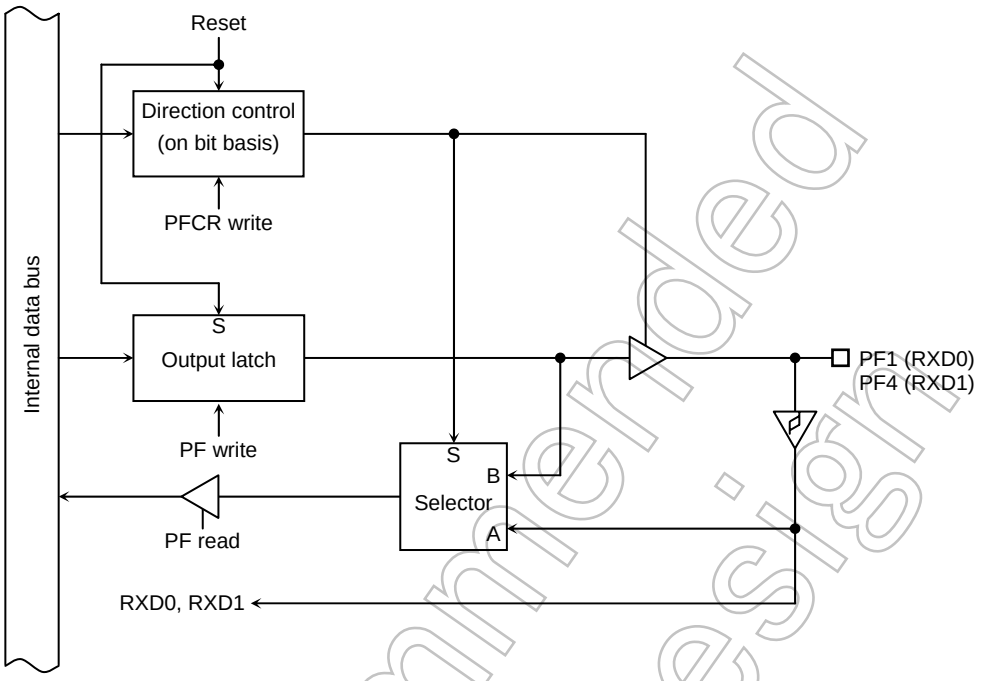



Figure 3.5.26 Port F (PF and PF4)

(3) Port PF2 ($\overline{CTS0}$, SCLK0) and port PF5 ($\overline{CTS1}$, SCLK1)

In addition to function as I/O port, port PF2 and PF5 can also function as $\overline{CTS}$ input pin of serial channel or SCLK I/O pin.

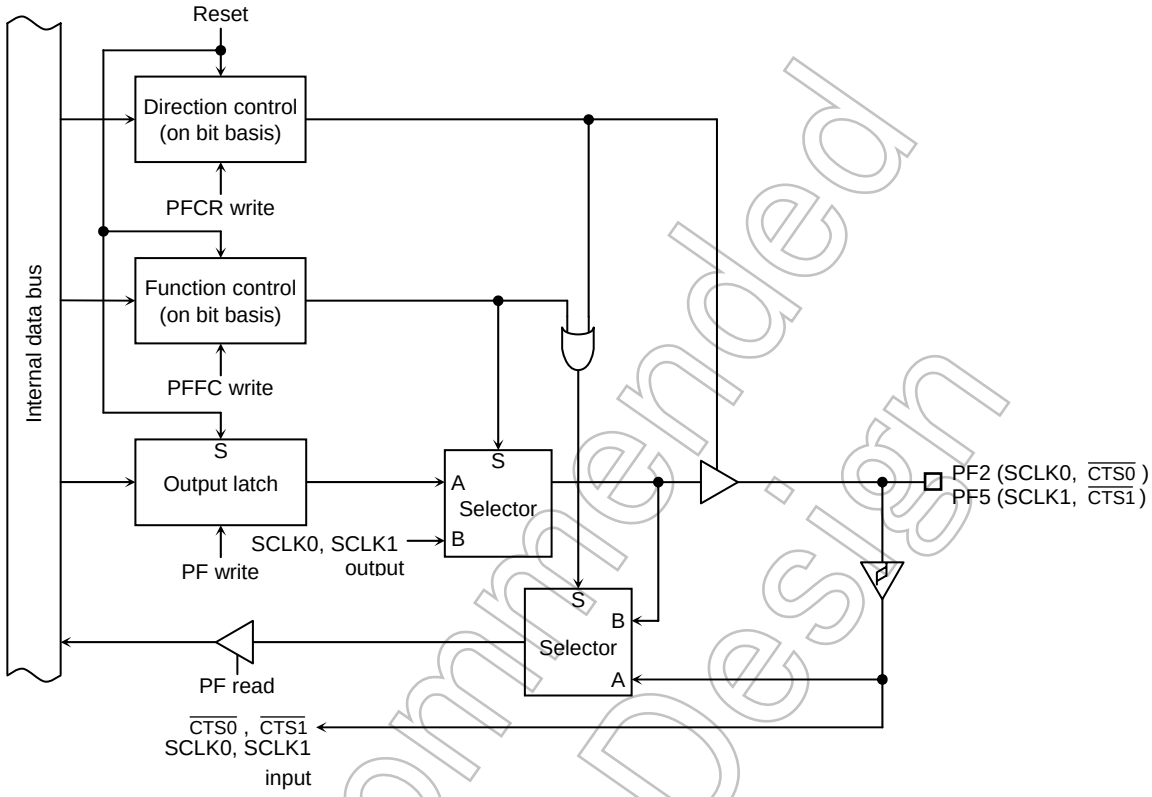



Figure 3.5.27 Port F (PF2 and PF5)

(4) Port PF6 and port PF7

These ports are general-purpose I/O port.

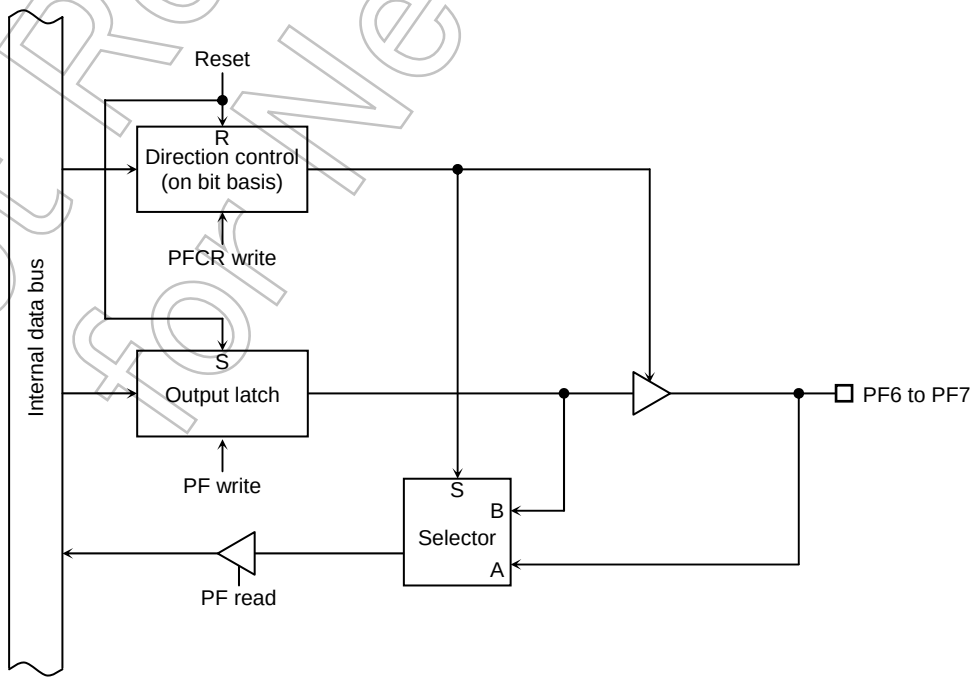



Figure 3.5.28 Port F (PF6 and PF7)

Port F Register

PF (003CH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PF7 | PF6 | PF5 | PF4 | PF | PF2 | PF1 | PF0 |
| Read/Write | R/W | | | | | | | |
| After reset | Data from external port (Output latch register is set to 1) | | | | | | | |

Port F Control Register

PFCR (003EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PF7C | PF6C | PF5C | PF4C | PF3C | PF2C | PF1C | PF0C |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | 0: Input 1: Output | | | | | | | |

Port F Function Register

PFFC (003FH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | – | PF5F | | PF3F | PF2F | | PF0F |
| Read/Write | W | | | | W | | | W |
| After reset | 0 | 0 | 0 | | 0 | 0 | | 0 |
| Function | Always write “0”. | Always write “0”. | 0: Port 1: SCLK1 output | | 0: Port 1: TXD1 | 0: Port 1: SCLK0 output | | 0: Port 1: TXD0 |

Port function setting

| <PF3C> / <PF3F> | 0 | 1 |
|---|---|---|
| 0 | Input port | Output port |
| 1 | TXD1 (Open drain) | TXD1 |

| <PF0C> / <PF0F> | 0 | 1 |
|---|---|---|
| 0 | Input port | Output port |
| 1 | TXD0 (Open drain) | TXD0 |

Note 1: Read-modify-write instruction is prohibited for the registers PFCR and PFFC.

Note 2: PF1/RXD0 and PF4/RXD1 pins do not have a register changing PORT/FUNCTION. For example, when it is used as an input port, the input signal is inputted to SIO as the serial receive data.

Note 3: PF0 and PF3 pins do not have a register (PFODE) for open-drain setting. Please conduct the open-drain setting according to above setting.

Figure 3.5.29 Register for Port F

### 3.5.12 Port G (PG0 to PG7)

Port G is 8-bit input port and can also be used as the analog input pins for the internal AD converter. PG3 can also be used as ADTRG pin for the AD converter.

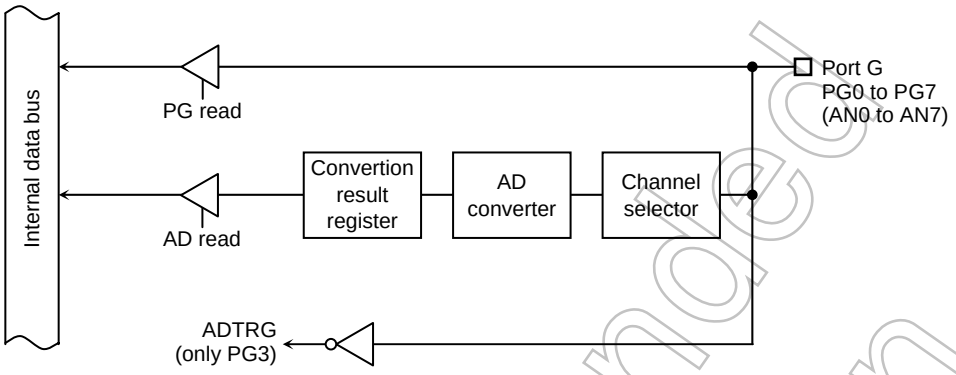



Figure 3.5.30 Port G

Port G Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| PG (0040H) | Bit symbol | PG7 | PG6 | PG5 | PG4 | PG3 | PG2 | PG1 | PG0 |
| | Read/Write | R | | | | | | | |
| | After reset | Data from external port | | | | | | | |

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

Figure 3.5.31 Register for Port G

## 3.6 Memory Controller

### 3.6.1 Function

TMP92CM22 has a memory controller with a variable 4-block address area that controls as follows.

(1) 4-block address area support
Specifies a start address and a block size for 4-block address area.

(2) Connecting memory specifications
Specifies SRAM and ROM as memories to connect with the selected address areas.

(3) Data bus size selection
Whether 8-bit or 16-bit is selected as the data bus size of the respective block address areas.

(4) Wait control
Wait specification bit in the control register and WAIT input pin control the number of waits in the external bus cycle. Read cycle and write cycle can specify the number of waits individually.
The number of waits is controlled in 6 mode mentioned below.

| |
|---|
| 0 waits, 1 wait, |
| 2 waits, 3 waits, 4 waits |
| N waits (Control with $\overline{\text{WAIT}}$ pin) |

### 3.6.2 Control Register and Operation after Reset Release

This section describes the registers to control the memory controller, the state after reset release and necessary settings.

(1) Control register

The control registers of the memory controller are as follows.

- Control register: BnCSH/BnCSL (n = 0 to 3, EX)
  Sets the basic functions of the memory controller, that is the connecting memory type, the number of waits to be read and written.
- Memory start address register: MSARn (n = 0 to 3)
  Sets a start address in the selected block address areas.
- Memory address mask register: MAMRn (n = 0 to 3)
  Sets a block size in the selected address areas.

In addition to setting of the above-mentioned registers, it is necessary to set the following registers to control ROM page mode access.

- Page ROM control register: PMEMCR
  Sets to executed ROM page mode accessing.

(2) Operation after reset release

The start data bus width is determined depending on state of AM1 and AM0 pins just after reset release. Then, the external memory is accessed as follows.

| AM1 | AM0 | Start Mode |
|---|---|---|
| 0 | 0 | Don't use this setting |
| 0 | 1 | Start with 16-bit data bus |
| 1 | 0 | Start with 8-bit data bus |
| 1 | 1 | Don't use this setting |

AM1/AM0 pins are valid only just after reset release. In the other cases, the data bus width is set to the value set to BnBUS bit of the control register.

By reset, only control register (B2CSH/B2CSL) of the block address area 2 is automatically effective (B2CSH<B2E> is set to "1" by reset).

The data bus width which is specified by AM1/AM0 pin is loaded to the bit to specify the bus width of the control register in the block address area 2.

The block address area 2 is set to address 000000H to FFFFFFH by reset.

After reset release, the block address areas are specified by the memory start address register (MSAR) and the memory address mask register (MAMR). Then the control register (BnCS) is set.

Set the enable bit (BnE) of the control register to "1" to enable the setting.

### 3.6.3 Basic Functions and Register Setting

In this section, setting of the block address area, the connecting memory and the number of waits out of the memory controller's functions are described.

(1) Block address area specification

The block address area is specified by two registers.

The memory start address register (MSAR) sets the start address of the block address areas. The memory controller compares between the register value and the address every bus cycles. The address bit which is masked by the memory address mask register (MAMR) is not compared by the memory controller. The block address area size is determined by setting the memory address mask register. The set value in the register is compared with the block address area on the bus. If the compared result is a match, the memory controller sets the chip select signal ($\overline{CS}$) to "low".

(i) Setting memory start address register

The MS23 to MS16 bits of the memory start address register respectively correspond with addresses A23 to A16. The lower start address A15 to A0 are always set to address 0000H.

Therefore the start address of the block address area are set to addresses 000000H to FF0000H every 64 Kbytes.

(ii) Setting memory address mask registers

The memory address mask register sets whether an address bit is compared or not. Set the register to "0" to compare, or to "1" not to compare.

The address bit to be set is depended on the block address area.

Block address area 0: A20 to A8

Block address area 1: A21 to A8

Block address area 2 to 3: A22 to A15

The above-mentioned bits are always compared. The block address area size is determined by the compared result.

The size to be set depending on the block address area is as follows.

| Size (bytes) / CS area | 256 | 512 | 32 K | 64 K | 128 K | 256 K | 512 K | 1 M | 2 M | 4 M | 8 M |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CS0 | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | | |
| CS1 | ○ | ○ | | ○ | ○ | ○ | ○ | ○ | ○ | ○ | |
| CS2 to CS3 | | | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

Note: After reset release, only the control register of the block address area 2 is valid. The control register of the block address area 2 has <B2M> bit. Setting <B2M> bit to "0" sets the block address area 2 to addresses 000000H to FFFFFFH. State of after reset release is set this. Setting <B2M> bit to "1" specifies the start address and the address area size as it is in the other block address area.

(iii) Example of register setting

To set the block address area 1 to 512 bytes from address 110000H, set the register as follows.

MSAR1 Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | M1S23 | M1S22 | M1S21 | M1S20 | M1S19 | M1S18 | M1S17 | M1S16 |
| Setting value | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

M1S23 to M1S16 bits of the memory start address register MSAR1 correspond with address A23 to A16.

A15 to A0 are cleared to "0". Therefore setting MSAR1 to the above-mentioned value specifies the start address of the block address area to address 110000H.

The start address is set as it is in the other block address areas.

MAMR1 Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | M1V21 | M1V20 | M1V19 | M1V18 | M1V17 | M1V16 | M1V15-9 | M1V8 |
| Setting value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

M1V21 to M1V16 and M1V8 bits of the memory address mask register MAMR1 set whether address A21 to A16 and A8 are compared or not. Set the register to "0" to compare, or to "1" not to compare. A23 and A22 are always compared.

Setting the above-mentioned compares A23 to A9 with the values set as the start addresses. Therefore 512 bytes of addresses 110000H to 1101FFH are set as the block address area 1, and compared with the addresses on the bus. If the compared result is a match, the chip select signal $\overline{CS1}$ is set to "low".

The other block address area sizes are specified like this.

Similarly, A23 is always compared in block address areas 2 to 3. Whether A22 to A15 are compared or not is set to register.

Note: When the set block address area overlaps with the built-in memory area, or both two address areas overlap, the block address area is processed according to priority as follows.

Built-in I/O > Built-in memory > Block address area 0 >1 > 2 > 3 > CSEX

Also that any accessed areas outside the address spaces set by $\overline{CS0}$ to $\overline{CS3}$ are processed as the CSEX space. Therefore, settings of CSEX apply for the control of wait cycles, data bus width, etc.

(2) Connection memory specification

Setting the BnOM1 to BnOM0 bit of the control register (BnCSH) specifies the memory type to be connected with the block address areas. The interface signal is output according to the set memory as follows. TMP92CM22 prohibit changing default (SRAM/ROM).

BnOM1, BnOM0 Bit (BnCSH register)

| BnOM1 | BnOM0 | Function |
|---|---|---|
| 0 | 0 | SRAM/ROM (Default) |
| 0 | 1 | (Reserved) |
| 1 | 0 | (Reserved) |
| 1 | 1 | (Reserved) |

(3) Data bus width specification

The data bus width is set for every block address area. The bus size is set by the BnBUS1 and BnBUS0 bits of the control register (BnCSH) as follows.

BnBUS Bit (BnCSH register)

| BnBUS1 | BnBUS0 | Function |
|---|---|---|
| 0 | 0 | 8-bit bus mode (Default) |
| 0 | 1 | 16-bit bus mode |
| 1 | 0 | (Reserved) |
| 1 | 1 | (Reserved) |

This way of changing the data bus size depending on the address being accessed is called “dynamic bus sizing”. The part where the data is output to is depended on the data size, the bus width and the start address.

Note: Since there is a possibility of abnormal writing/reading of the data if two memories with different bus width are put in consecutive addresses, do not execute an access to placed on both memories with one command.

| Data Size (Bit) | Start Address | Data Width in Memory Side (Bit) | CPU Address | CPU Data | |
|---|---|---|---|---|---|
| | | | | D15 to D8 | D7 to D0 |
| 8 | 4n + 0 | 8/16 | 4n + 0 | xxxxx | b7 to b0 |
| | 4n + 1 | 8 | 4n + 1 | xxxxx | b7 to b0 |
| | | 16 | 4n + 1 | b7 to b0 | xxxxx |
| | 4n + 2 | 8/16 | 4n + 2 | xxxxx | b7 to b0 |
| | 4n + 3 | 8 | 4n + 3 | xxxxx | b7 to b0 |
| | | 16 | 4n +3 | b7 to b0 | xxxxx |
| 16 | 4n + 0 | 8 | (1) 4n + 0 | xxxxx | b7 to b0 |
| | | | (2) 4n + 1 | xxxxx | b15 to b8 |
| | | 16 | 4n + 0 | b15 to b8 | b7 to b0 |
| | 4n + 1 | 8 | (1) 4n + 1 | xxxxx | b7 to b0 |
| | | | (2) 4n + 2 | xxxxx | b15 to b8 |
| | | 16 | (1) 4n + 1 | b7 to b0 | xxxxx |
| | | | (2) 4n + 2 | xxxxx | b15 to b8 |
| | 4n + 2 | 8 | (1) 4n + 2 | xxxxx | b7 to b0 |
| | | | (2) 4n + 1 | xxxxx | b15 to b8 |
| | | 16 | 4n + 2 | b15 to b8 | b7 to b0 |
| | 4n + 3 | 8 | (1) 4n + 3 | xxxxx | b7 to b0 |
| | | | (2) 4n + 4 | xxxxx | b15 to b8 |
| | | 16 | (1) 4n + 3 | b7 to b0 | xxxxx |
| | | | (2) 4n + 4 | xxxxx | b15 to b8 |
| 32 | 4n + 0 | 8 | (1) 4n + 0 | xxxxx | b7 to b0 |
| | | | (2) 4n + 1 | xxxxx | b15 to b8 |
| | | | (3) 4n + 2 | xxxxx | b23 to b16 |
| | | | (4) 4n + 3 | xxxxx | b31 to b24 |
| | | 16 | (1) 4n + 0 | b15 to b8 | b7 to b0 |
| | | | (2) 4n + 2 | b31 to b24 | b23 to b16 |
| | 4n + 1 | 8 | (1) 4n + 0 | xxxxx | b7 to b0 |
| | | | (2) 4n + 1 | xxxxx | b15 to b8 |
| | | | (3) 4n + 2 | xxxxx | b23 to b16 |
| | | | (4) 4n + 3 | xxxxx | b31 to b24 |
| | | 16 | (1) 4n + 1 | b7 to b0 | xxxxx |
| | | | (2) 4n + 2 | b23 to b16 | b15 to b8 |
| | | | (3) 4n + 4 | xxxxx | b31 to b24 |
| | 4n + 2 | 8 | (1) 4n + 2 | xxxxx | b7 to b0 |
| | | | (2) 4n + 3 | xxxxx | B15 to b8 |
| | | | (3) 4n + 4 | xxxxx | b23 to b16 |
| | | | (4) 4n + 5 | xxxxx | b31 to b24 |
| | | 16 | (1) 4n + 2 | b15 to b8 | b7 to b0 |
| | | | (2) 4n + 4 | b31 to b24 | b23 to b16 |
| | 4n + 3 | 8 | (1) 4n + 3 | xxxxx | b7 to b0 |
| | | | (2) 4n + 4 | xxxxx | b15 to b8 |
| | | | (3) 4n + 5 | xxxxx | b23 to b16 |
| | | | (4) 4n + 6 | xxxxx | b31 to b24 |
| | | 16 | (1) 4n + 3 | b7 to b0 | xxxxx |
| | | | (2) 4n + 4 | b23 to b16 | b15 to b8 |
| | | | (3) 4n + 6 | xxxxx | b31 to b24 |

xxxxx: During a read, data input to the bus ignored. At write, the bus is at high impedance and the write strobe signal remains inactive.

(4) Wait control

The external bus cycle completes a wait of two states at least (100 ns at $f_{SYS}$ = 20 MHz).

Setting the <BnWW2:0> and <BnWR2:0> of BnCSL specifies the number of waits in the read cycle and the write cycle. BnWW is set with the same method as BnWR.

BnWW/BnWR Bit (BnCSL Register)

| BnWW2<br>BnWR2 | BnWR1<br>BnWW1 | BnWW0<br>BnWR0 | Function |
|---|---|---|---|
| 0 | 0 | 1 | 2 states (0 waits) access fixed mode |
| 0 | 1 | 0 | 3 states (1 wait) access fixed mode (Default) |
| 1 | 0 | 1 | 4 states (2 waits) access fixed mode |
| 1 | 1 | 0 | 5 states (3 waits) access fixed mode |
| 1 | 1 | 1 | 6 states (4 waits) access fixed mode |
| 0 | 1 | 1 | $\overline{\text{WAIT}}$ pin input mode |
| Others | | | (Reserved) |

(i) Waits number fixed mode

The bus cycle is completed with the set states. The number of states is selected from 2 states (0 waits) to 6 states (4 waits).

(ii) $\overline{\text{WAIT}}$ pin input mode

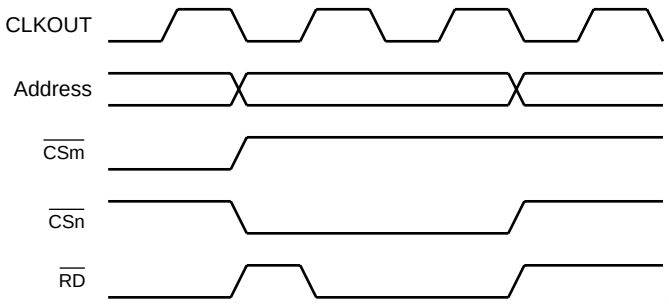
This mode samples the $\overline{\text{WAIT}}$ input pins. It continuously samples the $\overline{\text{WAIT}}$ pin state and inserts a wait if the pin is active. The bus cycle is minimum 2 states. The bus cycle is completed when the wait signal is non-active ("High" level) at 2 states. The bus cycle extends if the wait signal is active at 2 states and more.

If a lot of connected pertain ROM and etc. (Much data-output-floating-time (tDF)), each other's data-bus-output-recovery-time is trouble. However, by setting BnREC of control register (BnCSH), can to insert dummy cycle of 1-state just before first bus cycle of starting access another block address.

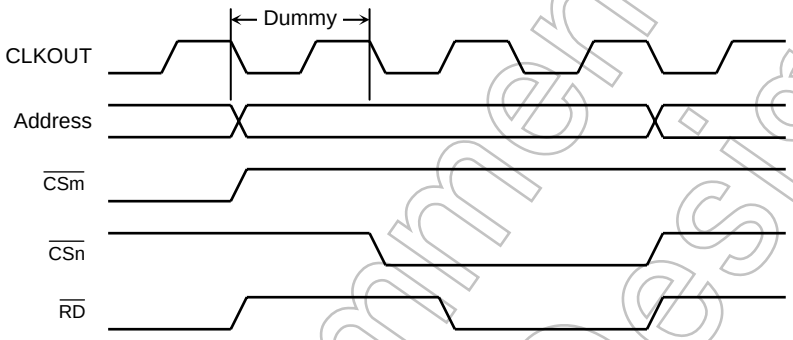
BnREC Bit (BnCSH register)

| | |
|---|---|
| 0 | No dummy cycle is inserted (Default). |
| 1 | Dummy cycle is inserted. |

- When not inserting a dummy (0 waits)

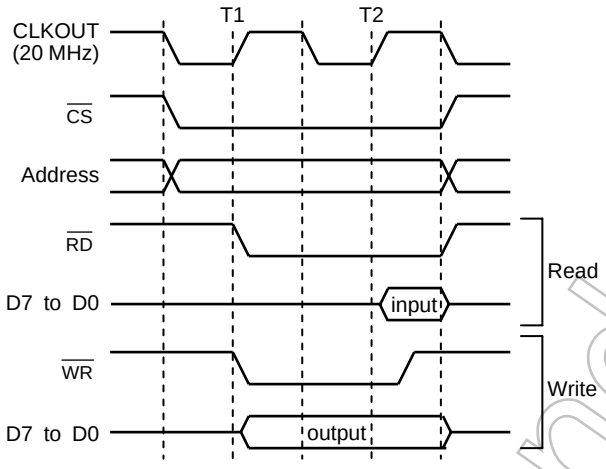



- When inserting a dummy cycle (0 waits)

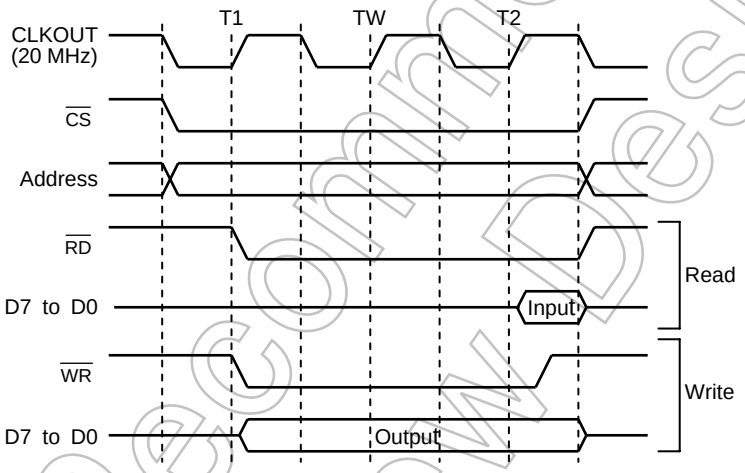


(5) Bus access timing

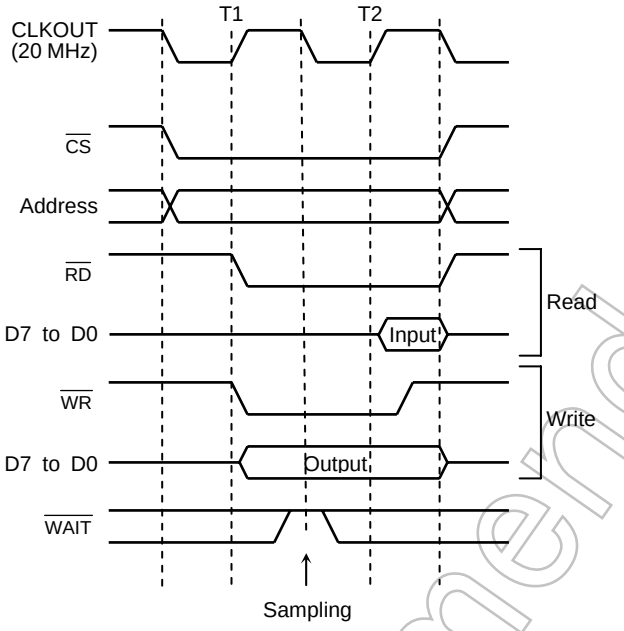
- External read/write bus cycle (0 waits)



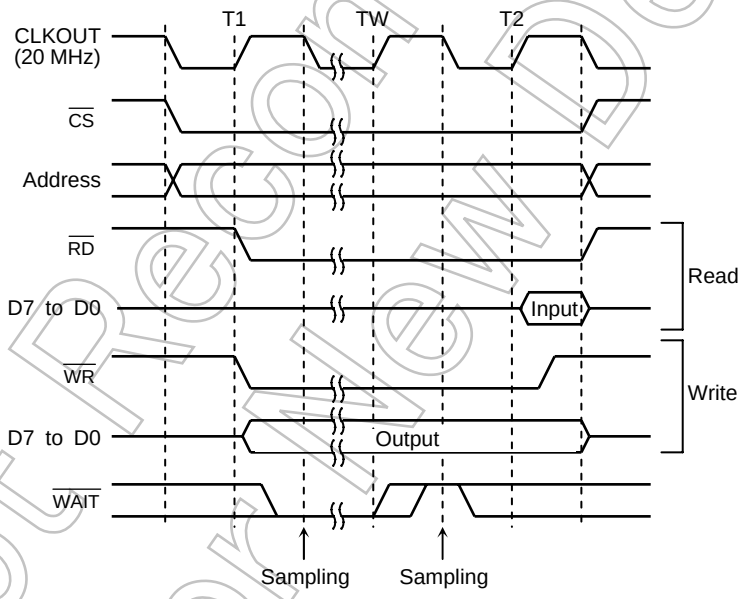

- External read/write bus cycle (1 wait)



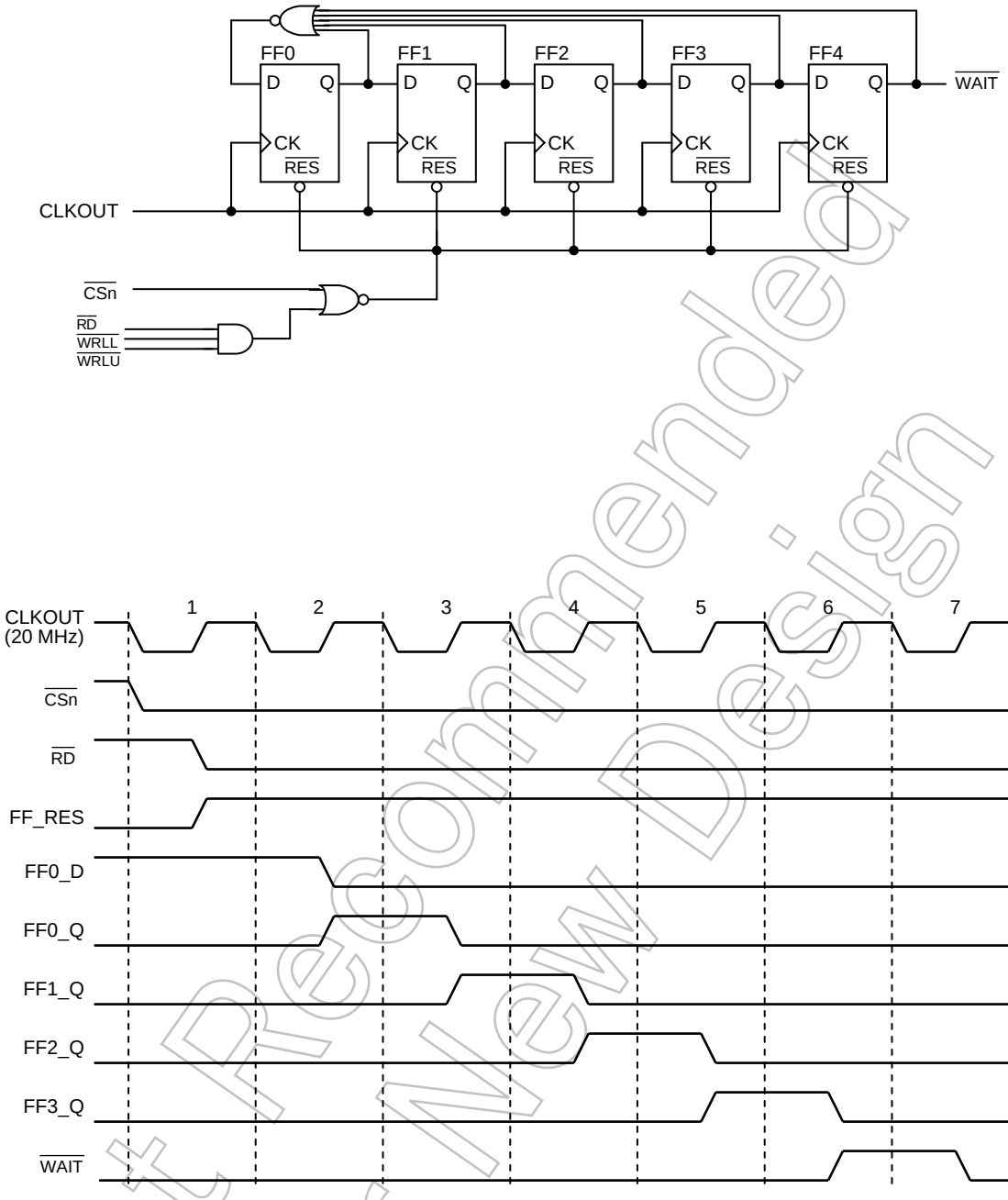
- External read/write bus cycle (0 waits at $\overline{\text{WAIT}}$ pin input mode)




- External read/write bus cycle (n waits at $\overline{\text{WAIT}}$ pin input mode)



Example of $\overline{\text{WAIT}}$ input cycle (5 waits)



(6) Connecting external memory

Figure 3.6.1 shows an example of how to connect external memory to the TMP92CM22.

This example connects ROM and SRAM in 16-bit width.

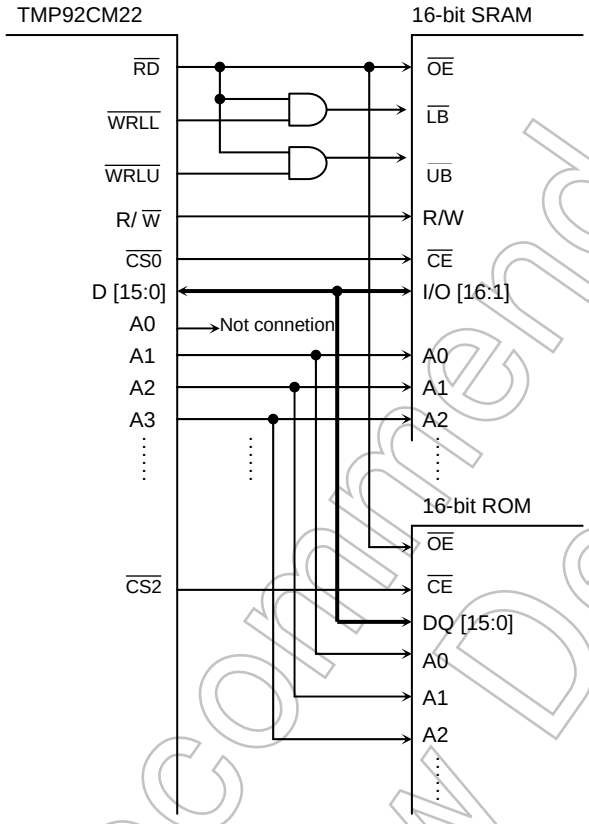



Figure 3.6.1 Example of External Memory

By resetting, TMP92CM22 function as output port. Output latch of P82 ($\overline{CS2}$) is cleared to "0", and output "L". Output latch of P80 ($\overline{CS0}$), P81 ($\overline{CS1}$) and P83 ($\overline{CS3}$) are set to "1", and output "H".

When set port 8 from port function to CS function, set need bit of P8FC register to "1".

Note: When set P82 as $\overline{CS2}$ after release reset, set function register remain output latch of P82 is "0" (P8<P82> = 0). (P8FC<P82F> = 1)

If set function register (P8FC<P82F> = 1) after set output latch of P82 to "1" (P8<P82> = 1), maybe don't read ROM data during changing from port function to $\overline{CS2}$.

### 3.6.4 ROM Control (Page mode)

This section describes ROM page mode accessing and how to set registers. ROM page mode is set by the page ROM control register.

(1) Operation and how to set the registers

The TMP92CM22 supports ROM access of the page mode. ROM access of the page mode is specified only in block address area 2.

ROM page mode is set by the page ROM control register (PMEMCR). Setting <OPGE> of the PMEMCR register to "1" sets the memory access of the block address area to ROM page mode access.

The number of read cycles is set by the <OPWR1:0> bit of the PMEMCR register.

OPWR1/OPWR0 Bit (PMEMCR register)

| OPWR1 | OPWR0 | Number of Cycle in A Page |
|---|---|---|
| 0 | 0 | 1 state (n-1-1-1 mode) (n ≥ 2) |
| 0 | 1 | 2 states (n-2-2-2 mode) (n ≥ 3) |
| 1 | 0 | 3states (n-3-3-3 mode) (n ≥ 4) |
| 1 | 1 | (Reserved) |

Note: Set the number of waits ("n") using the control register (BnCSL) in each block address area.

The page size (The number of bytes) of ROM in the CPU side is set by the <PR1:0> of the PMEMCR register. When data is read out up to the border of the set page, the controller completes the page reading operation. The start data of the next page is read in the normal cycle. The following data is set to page read again.

PR1/PR0 Bit (PMEMCR register)

| PR1 | PR0 | ROM Page Size |
|---|---|---|
| 0 | 0 | 64 bytes |
| 0 | 1 | 32 bytes |
| 1 | 0 | 16 bytes |
| 1 | 1 | 8 bytes |

(2) Signal pulse

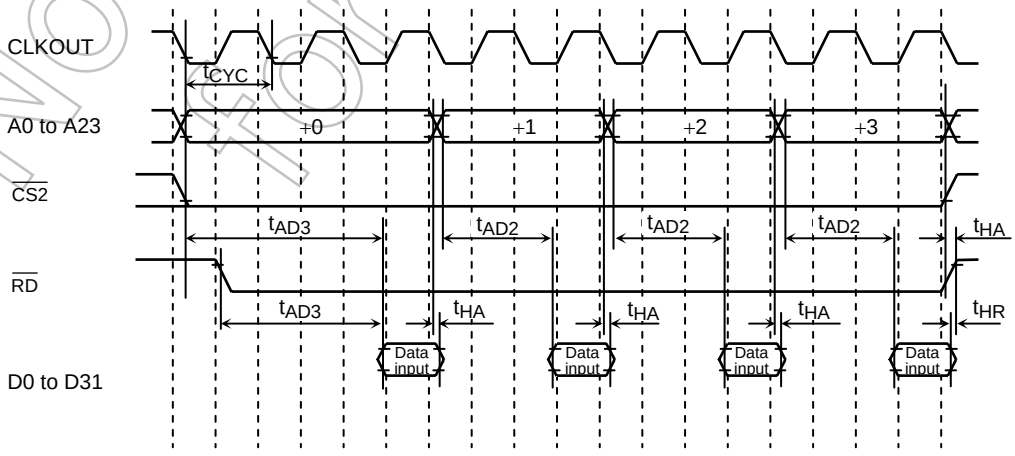



Figure 3.6.2 Page mode access Timing (8-byte example)

### 3.6.5 List of Registers

The memory control registers and the settings are described as follows. For the addresses of the registers, see list of special function registers in section 5.

(1) Control registers

The control register is a pair of BnCSL and BnCSH. ("n" is a number of the block address area.) BnCSL has the same configuration regardless of the block address areas. In BnCSH, only B2CSH which is corresponded to the block address area 2 has a different configuration from the others.

BnCSL

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | BnWW2 | BnWW1 | BnWW0 | | BnWR2 | BnWR1 | BnWR0 |
| Read/Write | | W | | | | W | | |
| After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |

BnWW[2:0] Specifies the number of write waits.

001 = 2 states (0 waits) access
010 = 3 states (1 wait) access
101 = 4 states (2 waits) access
110 = 5 states (3 waits) access
111 = 6 states (4 waits) access
011 = $\overline{\text{WAIT}}$ pin input mode
Others = (Reserved)

BnWR[2:0] Specifies the number of read waits.

001 = 2 states (0 waits) access
010 = 3 states (1 wait) access
101 = 4 states (2 waits) access
110 = 5 states (3 waits) access
111 = 6 states (4 waits) access
011 = $\overline{\text{WAIT}}$ pin input mode
Others = (Reserved)

B2CSH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | B2E | B2M | | B2REC | B2OM1 | B2OM0 | B2BUS1 | B2BUS0 |
| Read/Write | W | | | W | | | | |
| After reset | 1 | 0 | | 0 | 0 | 0 | 0 | 0 |

B2E Enable bit.

0 = No chip select signal output
1 = Chip select signal output (Default)

Note: After reset release, only the enable bit B2E of B2CSH register is valid ("1").

B2M Specifies the block address area.

0 = Sets the block address area of CS2 to addresses 000000H to FFFFFFH (Default)
1 = Sets the block address area of CS2 to programmable

Note: After reset release, the block address area 2 is set to addresses 000000H to FFFFFFH.

B2REC Sets the dummy cycle for data output recovery time.

0 = Not insert a dummy cycle (Default)

1 = Insert a dummy cycle

B2OM[1:0]

00 = SRAM or ROM (Default)

Others = (Reserved)

B2BUS[1:0] Sets the data bus width.

00 = 8 bits (Default)

01 = 16 bits

10 = (Reserved)

11 = (Reserved)

Note: The value of B2BUS bit is set according to the state of AM[1:0] pin after reset release.

BnCSH (n = 0, 1, 3)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | BnE | | | BnREC | BnOM1 | BnOM0 | BnBUS1 | BnBUS0 |
| Read/Write | W | | | W | | | | |
| After reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

BnE Enable bit.

0 = No chip select signal output (Default)

1 = Chip select signal output

Note: After reset release, only the enable bit B2E of B2CSH register is valid (“1”).

BnREC Sets the dummy cycle for data output recovery time.

0 = Not insert a dummy cycle (Default)

1 = Insert a dummy cycle

BnOM[1:0]

00 = SRAM or ROM (Default)

01 = (Reserved)

10 = (Reserved)

11 = (Reserved)

BnBUS[1:0] Sets the data bus width.

00 = 8 bits (Default)

01 = 16 bits

10 = (Reserved)

11 = (Reserved)

BEXCSL

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | BEXWW2 | BEXWW1 | BEXWW0 | | BEXWR2 | BEXWR1 | BEXWR0 |
| Read/Write | | W | | | | W | | |
| After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |

BEXWW[2:0] Specifies the number of write waits.

001 = 2 states (0 waits) access
010 = 3 states (1 wait) access
101 = 4 states (2 waits) access
110 = 5 states (3 waits) access
111 = 6 states (4 waits) access
011 = $\overline{\text{WAIT}}$ pin input mode
Others = (Reserved)

BEXWR[2:0] Specifies the number of read waits.

001 = 2 states (0 waits) access
010 = 3 states (1 wait) access
101 = 4 states (2 waits) access
110 = 5 states (3 waits) access
111 = 6 states (4 waits) access
011 = $\overline{\text{WAIT}}$ pin input mode
Others = (Reserved)

BEXCSH

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit Symbol | | – | – | – | BEXOM1 | BEXOM0 | BEXBUS1 | BEXBUS0 |
| Read/Write | | W | | | W | | | |
| After reset | | Always write 0. | | | 0 | 0 | 0 | 0 |

BEXOM[1:0]

00 = SRAM or ROM (Default)
01 = (Reserved)
10 = (Reserved)
11 = (Reserved)

BEXBUS[1:0]

00 = 8 bits (Default)
01 = 16 bits
10 = (Reserved)
11 = (Reserved)

(1) Block address area specification register

A start address and range in the block address are specified by the memory start address register (MSARn) and the memory address mask register (MAMRn). The memory start address register sets all start address similarly regardless of the block address areas. The bit to be set by the memory address mask register is depended on the block address area.

MSARn (n = 0 to 3)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | MnS23 | MnS22 | MnS21 | MnS20 | MnS19 | MnS18 | MnS17 | MnS16 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

MnS<23:16>

Sets a start address.

Sets the start address of the block address areas. The bit is corresponding to the address A23 to A16.

MAMR0

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | M0V20 | M0V19 | M0V18 | M0V17 | M0V16 | M0V15 | M0V14-9 | M0V8 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

M0V<20:8>

Enables or masks comparison of the addresses. M0V20 to M0V8 are corresponding to addresses A20 to A8. The bit of M0V14 to M0V9 is corresponding to address A14 to A9 by 1 bit. If "0" is set, the comparison between the value of the address bus and the start address is enabled. If "1" is set, the comparison is masked.

MAMR1

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | M1V21 | M1V20 | M1V19 | M1V18 | M1V17 | M1V16 | M1V15-9 | M1V8 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

M1V<21:8>

Enables or masks comparison of the addresses. M1V21 to M1V8 are corresponding to addresses A21 to A8. The bits of M1V15 to M1V9 are corresponding to address A15 to A9 by 1 bit. If "0" is set, the comparison between the value of the address bus and the start address is enabled. If "1" is set, the comparison is masked.

MAMRn (n = 2 to 3)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | MnV22 | MnV21 | MnV20 | MnV19 | MnV18 | MnV17 | MnV16 | MnV15 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

MnV<22:15>

Enables or masks comparison of the addresses. MnV22 to MnV15 are corresponding to addresses A22 to A15. If "0" is set, the comparison between the value of the address bus and the start address is enabled. If "1" is set, the comparison is masked.

After a reset, MASR0 to MASR3 and MAMR0 to MAMR3 are set to "FFH". B0CSH<B0E>, B1CSH<B1E>, and B3CSH<B3E> are reset to "0". This disables the CS0, CS1, and CS3 areas. However, B2CSH<B2M> is reset to "0" and B2CSH<B2E> to "1", and CS2 is enabled 000000H to FFFFFFH. Also the bus width and the number of waits specified in BEXCSH/L are used for accessing address except the specified CS0 to CS3 area.

(2) Page ROM control register (PMEMCR)

The page ROM control register sets page ROM accessing. ROM page accessing is executed only in block address area 2.

PMEMCR

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | OPGE | OPWR1 | OPWR0 | PR1 | PR0 |
| Read/Write | | | | R/W | | | | |
| After reset | | | | 0 | 0 | 0 | 1 | 0 |

OPGE Enable bit.

0 = No ROM page mode accessing (Default)

1 = ROM page mode accessing

OPWR [1:0] Specifies the number of waits.

00 = 1 state (n-1-1-1 mode) (n ≥ 2) (Default)

01 = 2 states (n-2-2-2 mode) (n ≥ 3)

10 = 3 states (n-3-3-3 mode) (n ≥ 4)

11 = (Reserved)

Note: Set the number of waits “n” to the control register (BnCSL) in each block address area.

PR [1:0] ROM page size.

00 = 64 bytes

01 = 32 bytes

10 = 16 bytes (Default)

11 = 8 bytes

Table 3.6.1 Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| B0CSL | Bit symbol | | B0WW2 | B0WW1 | B0WW0 | | B0WR2 | B0WR1 | B0WR0 |
| (0140H) | Read/Write | | W | | | | W | | |
| | After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |
| B0CSH | Bit symbol | B0E | – | – | B0REC | B0OM1 | B0OM0 | B0BUS1 | B0BUS0 |
| (0141H) | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 (Note) | 0 (Note) | 0 | 0 | 0 | 0 | 0 |
| MAMR0 | Bit symbol | M0V20 | M0V19 | M0V18 | M0V17 | M0V16 | M0V15 | M0V14-V9 | M0V8 |
| (0142H) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| MSAR0 | Bit symbol | M0S23 | M0S22 | M0S21 | M0S20 | M0S19 | M0S18 | M0S17 | M0S16 |
| (0143H) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B1CSL | Bit symbol | | B1WW2 | B1WW1 | B1WW0 | | B1WR2 | B1WR1 | B1WR0 |
| (0144H) | Read/Write | | W | | | | W | | |
| | After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |
| B1CSH | Bit symbol | B1E | – | – | B1REC | B1OM1 | B1OM0 | B1BUS1 | B1BUS0 |
| (0145H) | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 (Note) | 0 (Note) | 0 | 0 | 0 | 0 | 0 |
| MAMR1 | Bit symbol | M1V21 | M1V20 | M1V19 | M1V18 | M1V17 | M1V16 | M1V15-V9 | M1V8 |
| (0146H) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| MSAR1 | Bit symbol | M1S23 | M1S22 | M1S21 | M1S20 | M1S19 | M1S18 | M1S17 | M1S16 |
| (0147H) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B2CSL | Bit symbol | | B2WW2 | B2WW1 | B2WW0 | | B2WR2 | B2WR1 | B2WR0 |
| (0148H) | Read/Write | | W | | | | W | | |
| | After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |
| B2CSH | Bit symbol | B2E | B2M | – | B2REC | B2OM1 | B2OM0 | B2BUS1 | B2BUS0 |
| (0149H) | Read/Write | W | | | | | | | |
| | After reset | 1 | 0 | 0 (Note) | 0 | 0 | 0 | 0 | 0 |
| MAMR2 | Bit symbol | M2V22 | M2V21 | M2V20 | M2V19 | M2V18 | M2V17 | M2V16 | M2V15 |
| (014AH) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| MSAR2 | Bit symbol | M2S23 | M2S22 | M2S21 | M2S20 | M2S19 | M2S18 | M2S17 | M2S16 |
| (014BH) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B3CSL | Bit symbol | | B3WW2 | B3WW1 | B3WW0 | | B3WR2 | B3WR1 | B3WR0 |
| (014CH) | Read/Write | | W | | | | W | | |
| | After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |
| B3CSH | Bit symbol | B3E | – | – | B3REC | B3OM1 | B3OM0 | B3BUS1 | B3BUS0 |
| (014DH) | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 (Note) | 0 (Note) | 0 | 0 | 0 | 0 | 0 |
| MAMR3 | Bit symbol | M3V22 | M3V21 | M3V20 | M3V19 | M3V18 | M3V17 | M3V16 | M3V15 |
| (014EH) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| MSAR3 | Bit symbol | M3S23 | M3S22 | M3S21 | M3S20 | M3S19 | M3S18 | M3S17 | M3S16 |
| (014FH) | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| BEXCSH | Bit symbol | | | | | BEXOM1 | BEXOM0 | BEXBUS1 | BEXBUS0 |
| (0159H) | Read/Write | W | | | | | | | |
| | After reset | | | | | 0 | 0 | 0 | 0 |
| BEXCSL | Bit symbol | | BEXWW2 | BEXWW1 | BEXWW0 | | BEXWR2 | BEXWR1 | BEXWR0 |
| (0158H) | Read/Write | | W | | | | W | | |
| | After reset | | 0 | 1 | 0 | | 0 | 1 | 0 |
| PMEMCR | Bit symbol | | | | OPGE | OPWR1 | OPWR0 | PR1 | PR0 |
| (0166H) | Read/Write | | | | R/W | | | | |
| | After reset | | | | 0 | 0 | 0 | 1 | 0 |

Note1: Always write “0”.

Note2: Read-modify-write instruction is prohibited for BnCSL, BnCSH registers (n=0 to 3, EX).

### 3.6.6 Caution

If the parasitic capacitance of the read signal (Output enable signal) is greater than that of the chip select signal, it is possible that an unintended read cycle occurs due to a delay in the read signal. Such an unintended read cycle may cause a trouble as in the case of (a) in Figure 3.6.3

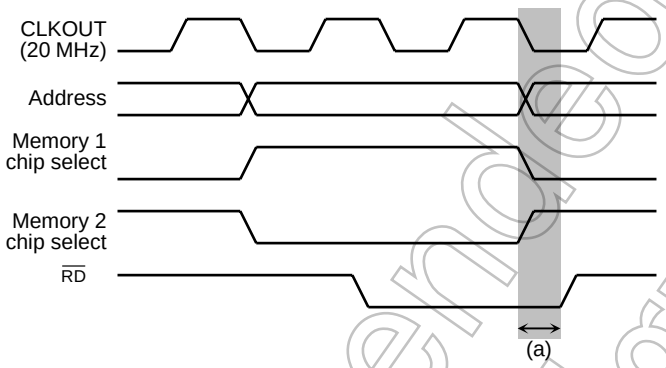



Figure 3.6.3 Read Signal Delay Read Cycle

Example: When using an externally connected flash EEPROM which users JEDEC standard commands, note that the toggle bit may not be read out correctly. If the read signal in the cycle immediately preceding the access to the flash EEPROM does not go "high" in time, as shown in Figure 3.6.4 an unintended read cycle like the one shown in (b) may occur.

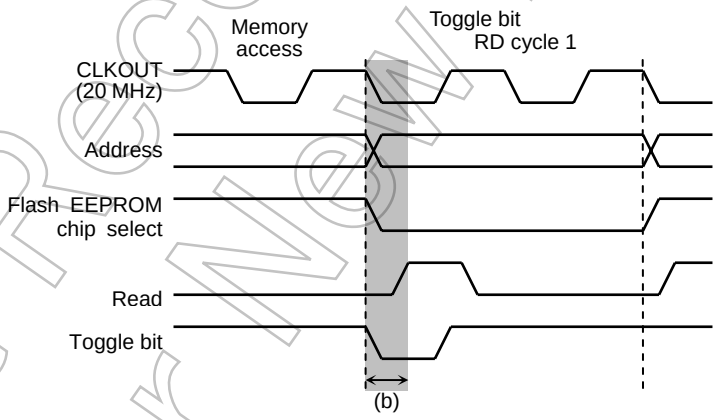



Figure 3.6.4 Flash EEPROM Toggle Bit Read Cycle

When the toggle bit reverse with this unexpected read cycle, TMP92CM22 always reads same value of the toggle bit, and cannot read the toggle bit correctly. To avoid this phenomenon, the data polling control recommended.

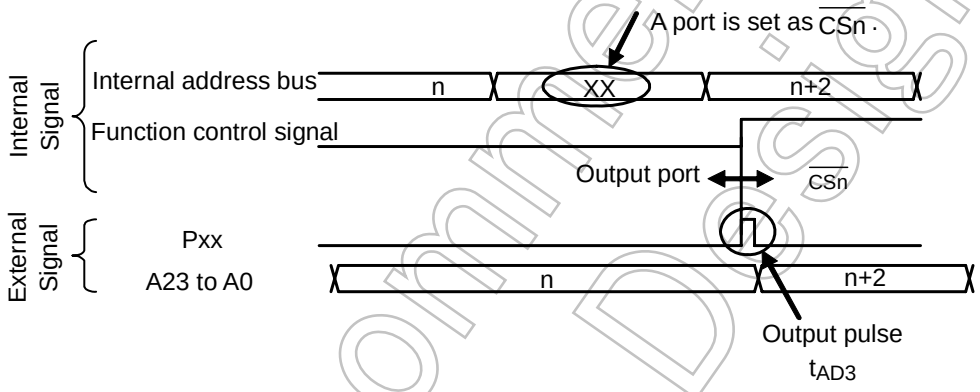
(2) The cautions at the time of the functional change of a $\overline{CSn}$.

A chip select signal output has the case of a combination terminal with a general-purpose port function. In this case, an output latch register and a function control register are initialized by the reset action, and an object terminal is initialized by the port output ("1" or "0") by it.

Functional change

Although an object terminal is changed from a port to a chip select signal output by setting up a function control register (PnFC register), the short pulse for several ns may be outputted to the changing timing. Although it does not become especially a problem when using the usual memory, it may become a problem when using a special memory.

* XX is a function register address.(When an output port is initialized by "0")

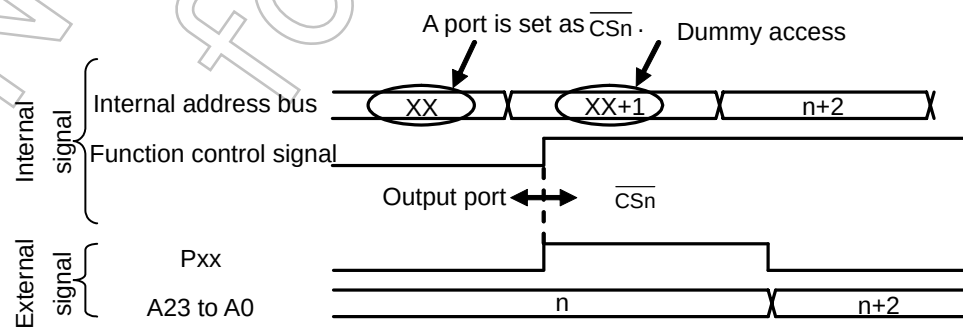



The measure by software

The countermeasures in S/W for avoiding this phenomenon are explained.

Since CS signal decodes the address of the access area and is generated, an unnecessary pulse is outputted by access to the object CS area immediately after setting it as a CSn function. Then, if internal area is accessed also immediately after setting a port as CS function, an unnecessary pulse will not output.

1. Prohibition of use of an NMI function
2. The ban on interruption under functional change (DI command)
3. A dummy command is added in order to carry out continuous internal access.
4. (Access to a functional change register is corresponded by 16-bit command. (LDW command))



## 3.7 8-Bit Timers (TMRA)

The TMP92CM22 features 4 built-in 8-bit timers.
These timers are paired into four modules: TMRA01 and TMRA23. Each module consists of two channels and can operate in any of the following four 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 3.7.1 and Figure 3.7.2 show block diagrams for TMRA01 and TMRA23.

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 five controls SFR (Special-function registers).

Each of the two modules (TMRA01 and TMRA23) 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 3.7.1 Registers and Pins for Each Module

| Specification \ Module | | Timer A01 | Timer A23 |
|---|---|---|---|
| External pin | Input pin for external clock | TA0IN (Shared with PC0) | None |
| | Output pin for timer flip-flop | TA1OUT (Shared with PC1) | TA3OUT (Shared with PC5) |
| SFR (Address) | Timer RUN register | TA01RUN (1100H) | TA23RUN (1108H) |
| | Timer register | TA0REG (1102H) TA1REG (1103H) | TA2REG (110AH) TA3REG (110BH) |
| | Timer mode register | TA01MOD (1104H) | TA23MOD (110CH) |
| | Timer flip-flop control register | TA1FFCR (1105H) | TA3FFCR (110DH) |

### 3.7.1 Block Diagrams

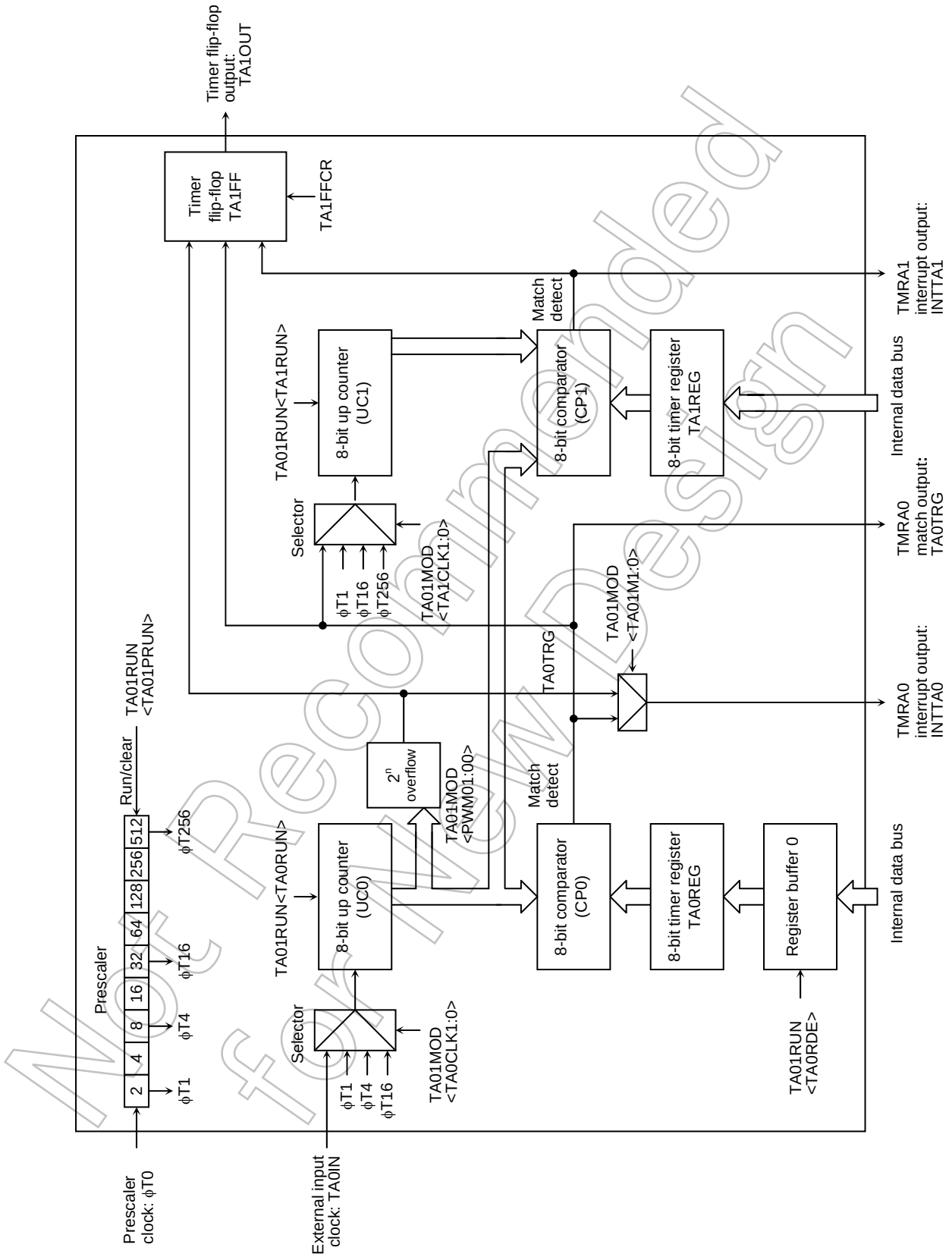





Figure 3.7.1 TMRA01 Block Diagram

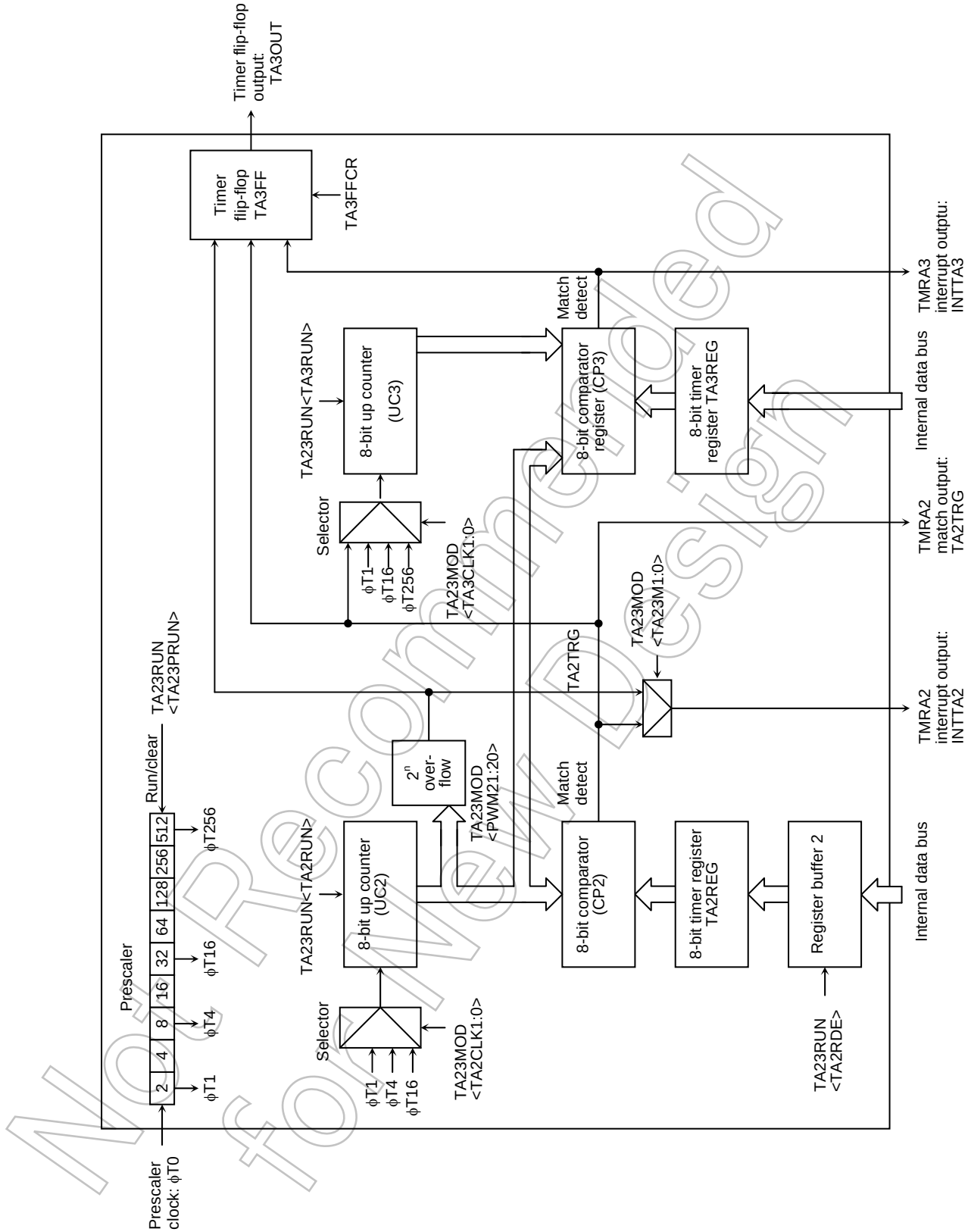



Figure 3.7.2 TMRA23 Block Diagram

### 3.7.2 Operation of Each Circuit

(1) Prescaler

A 9-bit prescaler generates the input clock to TMRA01.

The prescaler's operation can be controlled using TA01RUN<TA0PRUN> in the timer control register. Setting <TA0PRUN> to "1" starts the count; setting <TA0PRUN> to "0" clears the prescaler to "0" and stops operation. Table 3.7.2 shows the various prescaler output clock resolutions.

Table 3.7.2 Prescaler Output Clock Resolution

| Clock gear selection SYSCR1 <GEAR2:0> | System clock selection SYSCR1 <SYSCK> | – | Timer counter input clock TMRA prescaler TAxMOD<TAxCLK1:0> | | | |
|---|---|---|---|---|---|---|
| | | | φT1(1/2) | φT4(1/8) | φT16(1/32) | φT256(1/512) |
| 000 (1/1) | 0 (fc) | 1/8 | fc/16 | fc/64 | fc/256 | fc/4096 |
| 001 (1/2) | | | fc/32 | fc/128 | fc/512 | fc/8192 |
| 010 (1/4) | | | fc/64 | fc/256 | fc/1024 | fc/16384 |
| 011 (1/8) | | | fc/128 | fc/512 | fc/2048 | fc/32768 |
| 100 (1/16) | | | fc/256 | fc/1024 | fc/4096 | fc/65536 |

(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 release both up counters, stopping the timers.

(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.

The TA0REG are double buffer structure, each of which makes a pair with 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.

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 3.7.3 show the configuration of TA0REG

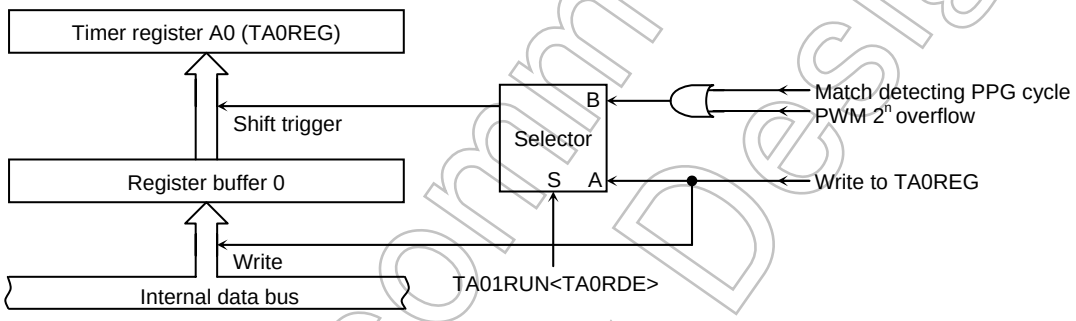



Figure 3.7.3 Timer Register A0 (TA0REG)

Note: The same memory address is allocated to the timer register and the register buffer. When <TA0RDE> = 0, the same value is written to the register buffer and the timer register; when <TA0RDE> = 1, only the register buffer is written to.

The address of each timer register is as follows.

TA0REG: 001102H TA1REG: 001103H

TA2REG: 00110AH TA3REG: 00110BH

All these registers are write-only and cannot be read.

(4) Comparator (CP0)

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.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detects signal (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". Programming "01" or "10" to TA1FFCR<TA1FFC1:0> sets TA1FF to 0 or 1. Programming "00" to these bits inverts the value of TA1FF. (This is known as software inversion.)

The TA1FF signal is output via the TA1OUT pin (which can also be used as PC1). When this pin is used as the timer output, the timer flip-flop should be set beforehand using the port C function register PCFC.

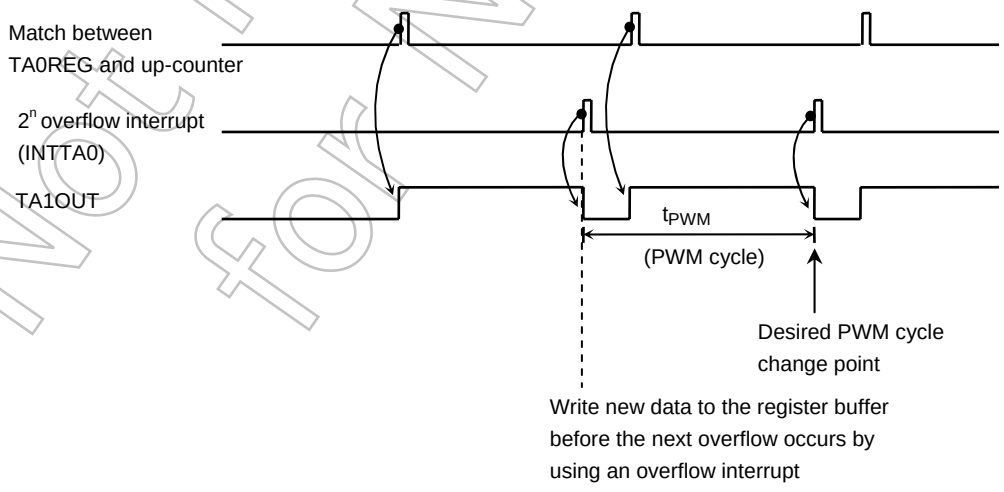
Note: When the double buffer is enabled for an 8-bit timer in PWM or PPG mode, caution is required as explained below.

If new data is written to the register buffer immediately before an overflow occurs by a match between the timer register value and the up-counter value, the timer flip-flop may output an unexpected value.

For this reason, make sure that in PWM mode new data is written to the register buffer by six cycles ($f_{SYS} \times 6$) before the next overflow occurs by using an overflow interrupt.

When using PPG mode, make sure that new data is written to the register buffer by six cycles before the next cycle compare match occurs by using a cycle compare match interrupt.

Example when using PWM mode



### 3.7.3 SFRs

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 | | | |
| After reset | 0 | | | | 0 | 0 | 0 | 0 |
| Function | 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 double buffer control

| | |
|---|---|
| 0 | Disable |
| 1 | Enable |

Count operation

| | |
|---|---|
| 0 | Stop and clear |
| 1 | Count |

Note: The values of bits 4 to 6 of TA01RUN are undefined 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 | | | |
| After reset | 0 | | | | 0 | 0 | 0 | 0 |
| Function | 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 double buffer control

| | |
|---|---|
| 0 | Disable |
| 1 | Enable |

Count operation

| | |
|---|---|
| 0 | Stop and clear |
| 1 | Count |

Note: The values of bits 4 to 6 of TA23RUN are undefined when read.

Figure 3.7.4 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 | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | Operation mode<br>00: 8-bit timer mode<br>01: 16-bit timer mode<br>10: 8-bit PPG mode<br>11: 8-bit PWM mode | | PWM cycle<br>00: Reserved<br>01: $2^6$<br>10: $2^7$<br>11: $2^8$ | | TMRA1 source clock<br>00: TA0TRG<br>01: φT1<br>10: φT16<br>11: φT256 | | TMRA0 source clock<br>00: TA0IN pin input (Note)<br>01: φT1<br>10: φT4<br>11: φT16 | |

TMRA0 input clock

| | |
|---|---|
| 00 | TA0IN (External input) |
| 01 | φT1 (Prescaler) |
| 10 | φT4 (Prescaler) |
| 11 | φT16 (Prescaler) |

TMRA1 input clock

| | TA01MOD <TA01M1:0> ≠ 01 | TA01MOD <TA01M1:0> = 01 |
|---|---|---|
| 00 | Matching output for TMRA0 | Overflow output for TMRA0 |
| 01 | φT1 | |
| 10 | φT16 | |
| 11 | φT256 | (16-bit timer mode) |

Select cycle in PWM mode

| | |
|---|---|
| 00 | Reserved |
| 01 | $2^6$ × Source clock |
| 10 | $2^7$ × Source clock |
| 11 | $2^8$ × Source clock |

Select operation mode for TMR01

| | |
|---|---|
| 00 | 8-bit timers × 2ch |
| 01 | 16-bit timer |
| 10 | 8-bit PPG |
| 11 | 8-bit PWM (TMRA0),<br>8-bit timer (TMRA1) |

Note: When set TA0IN pin, set TA01MOD after set port C.

Figure 3.7.5 Register for TMRA01

TMRA23 Mode Register

TA23MOD (110CH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TA23M1 | TA23M0 | PWM21 | PWM20 | TA3CLK1 | TA3CLK0 | TA2CLK1 | TA2CLK0 |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Operation mode<br>00: 8-bit timer mode<br>01: 16-bit timer mode<br>10: 8-bit PPG mode<br>11: 8-bit PWM mode | | PWM cycle<br>00: Reserved<br>01: $2^6$<br>10: $2^7$<br>11: $2^8$ | | TMRA3 source clock<br>00: TA2TRG<br>01: φT1<br>10: φT16<br>11: φT256 | | TMRA2 source clock<br>00: Reserved<br>01: φT1<br>10: φT4<br>11: φT16 | |

→TMRA2 input clock

| | |
|---|---|
| 00 | Don't set |
| 01 | φT1 (Prescaler) |
| 10 | φT4 (Prescaler) |
| 11 | φT16 (Prescaler) |

→TMRA3 input clock

| | TA23MOD <TA23M1:0> ≠ 01 | TA23MOD <TA23M1:0> = 01 |
|---|---|---|
| 00 | Matching output for TMRA2 | Overflow output for TMRA2<br>(16−bit timer mode) |
| 01 | φT1 | |
| 10 | φT16 | |
| 11 | φT256 | |

→Select cycle in PWM mode

| | |
|---|---|
| 00 | Reserved |
| 01 | $2^6$ × Source clock |
| 10 | $2^7$ × Source clock |
| 11 | $2^8$ × Source clock |

→Select operation mode for TMRA23

| | |
|---|---|
| 00 | 8-bit timer × 2ch |
| 01 | 16-bit timer |
| 10 | 8-bit PPG |
| 11 | 8-bit PWM (TMRA2),<br>8-bit timer (TMRA3) |



Figure 3.7.6 Register for TMRA23

TMRA1 Flip Flop Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| TA1FFCR (1105H) | Bit symbol | | | | | TA1FFC1 | TA1FFC0 | TA1FFCIE | TA1FFCIS |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 1 | 1 | 0 | 0 |
| Read-modify -write instruction is prohibited. | Function | | | | | 00: Invert TA1FF<br>01: Set TA1FF to “1”<br>10: Clear TA1FF to “0”<br>11: Don’t care | | TA1FF control for inversion<br>0: Disable<br>1: Enable | TA1FF Inversion signal select<br>0: TMRA0<br>1: TMRA1 |

Inversion signal for timer flip-flop 1 (TA1FF)
(Don’t care except in 8-bit timer mode)

| | |
|---|---|
| 0 | Inversion by TMRA0 |
| 1 | Inversion by TMRA1 |

TA1FF control for inversion

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

TFF1 control

| | |
|---|---|
| 00 | Invert TA1FF |
| 01 | Set TA1FF to “1” |
| 10 | Clear TA1FF to “0” |
| 11 | Don’t care |

Note: The values of bits 4 to 7 of TA1FFCR are undefined when read.

Figure 3.7.7 Register for TMRA

TMRA3 Flip-Flop Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| TA3FFCR (110DH) | Bit symbol | | | | | TA3FFC1 | TA3FFC0 | TA3FFCIE | TA3FFCIS |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 1 | 1 | 0 | 0 |
| Read-modify -write instruction is prohibited. | Function | | | | | 00: Invert TA3FF<br>01: Set TA3FF to “1”<br>10: Clear TA3FF to “0”<br>11: Don’t care | | TA3FF control for inversion<br>0: Disable<br>1: Enable | TA3FF inversion select<br>0: TMRA2<br>1: TMRA3 |

Inverse signal for timer flip-flop 3 (TA3FF)
(Don’t care except in 8-bit timer mode)

| | |
|---|---|
| 0 | Invert TMRA2 |
| 1 | Invert TMRA3 |

TA3FF control for inversion

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

TA3FF control

| | |
|---|---|
| 00 | Invert TA3FF |
| 01 | Set TA3FF to “1” |
| 10 | Clear TA3FF to “0” |
| 11 | Don’t care |

Note: The values of bits 4 to 7 of TA3FFCR are undefined when read.

Figure 3.7.8 Register for TMRA

Timer Register (TA0REG to TA3REG)

| Symbol | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| TA0REG | 1102H | – | | | | | | | |
| | | W | | | | | | | |
| | | Undefined | | | | | | | |
| TA1REG | 1103H | – | | | | | | | |
| | | W | | | | | | | |
| | | Undefined | | | | | | | |
| TA2REG | 110AH | – | | | | | | | |
| | | W | | | | | | | |
| | | Undefined | | | | | | | |
| TA3REG | 110BH | – | | | | | | | |
| | | W | | | | | | | |
| | | Undefined | | | | | | | |

Note: Read-modify-write instruction is prohibited for above registers.

Figure 3.7.9 Register for TMRA

### 3.7.4 Operation in Each Mode

(1) 8-bit timer mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

When set function and count data, TMRA0 and TMRA1 should be stopped.

1. 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 40 μs at $f_C$ = 40 MHz, set each register as follows:

| | 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 | – | – | Select 8-bit timer mode and select φT1 (=(16/fc)s at $f_C$ = 40MHz) as the input clock. |
| TA1REG | ← 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | Set 40 μs ÷ φT1 = 100 = 64H to TAREG. |
| INTETA01 | ← X | 1 | 0 | 1 | – | – | – | – | 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 refers to Table 3.7.3.

Table 3.7.3 Selecting Interrupt Interval and the Input Clock Using 8-Bit Timer

| Input clock | Interrupt Interval (at $f_{SYS}$ = 20 MHz) | Resolution |
|---|---|---|
| φT1 (8/$f_{SYS}$) | 0.4 μs to 102.4 μs | 0.4 μs |
| φT4 (32/$f_{SYS}$) | 1.6 μs to 409.6 μs | 1.6 μs |
| φT16 (128/$f_{SYS}$) | 6.4 μs to 1.638 ms | 6.4 μs |
| φT256 (2048/$f_{SYS}$) | 102.4 μs to 26.21 ms | 102.4 μs |

Note: The input clocks for TMRA0 and TMRA1 differ as follows:

TMRA0: Uses TMRA0 input (TA0IN) and can be selected from φT1, φT4, or φT16.

TMRA1: Matches output of TMRA0 (TA0TRG) and can be selected from φT1, φT16, φT256.

2. Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF1) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 2.4 μs square wave pulse from the TA1OUT pin at $f_C$ = 40 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

| | 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 | – | – | Select 8-bit timer mode and select φT1 (=(16/fc)s at $f_C$ = 40MHz) as the input clock. |
| TA1REG | ← 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | Set the timer register to 2.4 μs ÷ φT1 ÷ 2 = 3. |
| 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. |
| PCCR | ← X | – | – | X | – | X | 1 | – | Set PC1 to function as the TA1OUT pin. |
| PCFC | ← X | – | – | X | – | X | 1 | – | |
| TA01RUN | ← – | X | X | X | – | 1 | 1 | – | Start TMRA1 counting. |

X : Don't care, – : No change

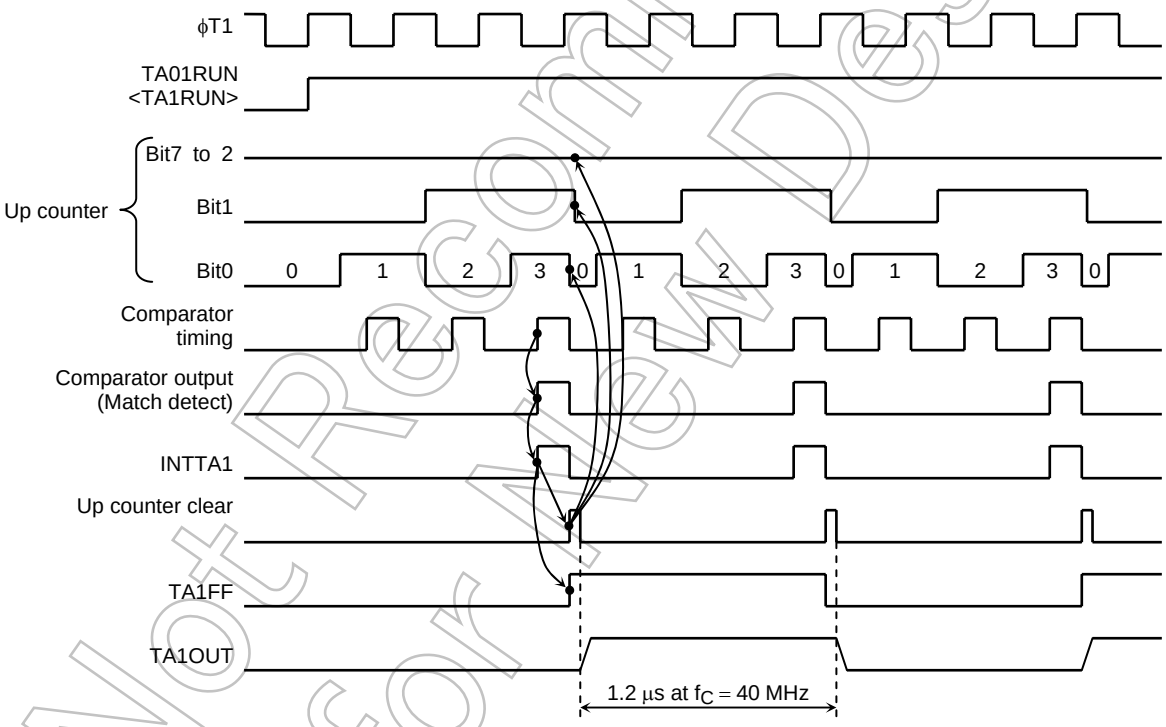



Figure 3.7.10 Square Wave Output Timing Chart (50% duty)

3. 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.

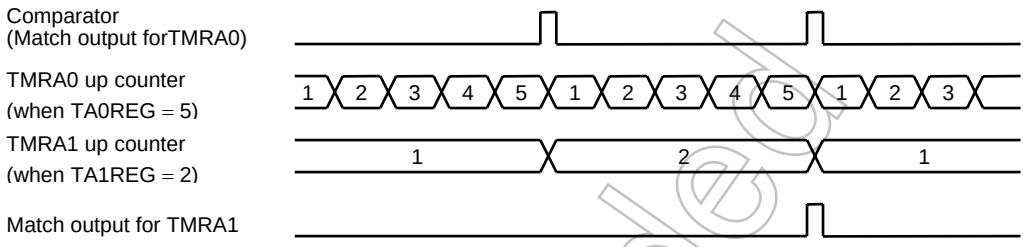



Figure 3.7.11 TMRA1 Count up on Signal from TMRA0

(2) 16-bit timer mode

A 16-bit interval timer is configured by pairing the two 8-bit timers TMRA0 and TMRA1.

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 3.7.3 shows the relationship between the timer (Interrupt) cycle and the input clock selection.

To set the timer interrupt interval, set the lower eight bits in timer register TA0REG and the upper eight bits in TA1REG. Be sure to set TA0REG first (As entering data in TA0REG temporarily disables the compare, while entering data in TA1REG starts the compare).

Example: To generate an INTTA1 interrupt every 0.4 s at $f_C$ = 40 MHz, set the timer registers TA0REG and TA1REG as follows:

If $\phi$T16 (= (256/fc)s at $f_C$ = 40MHz) is used as the input clock for counting, set the following value in the registers:

0.4 s ÷ (256/fc)s = 62500 = F424H;

e.g., set TA1REG to F4H and TA0REG to 24H.

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.

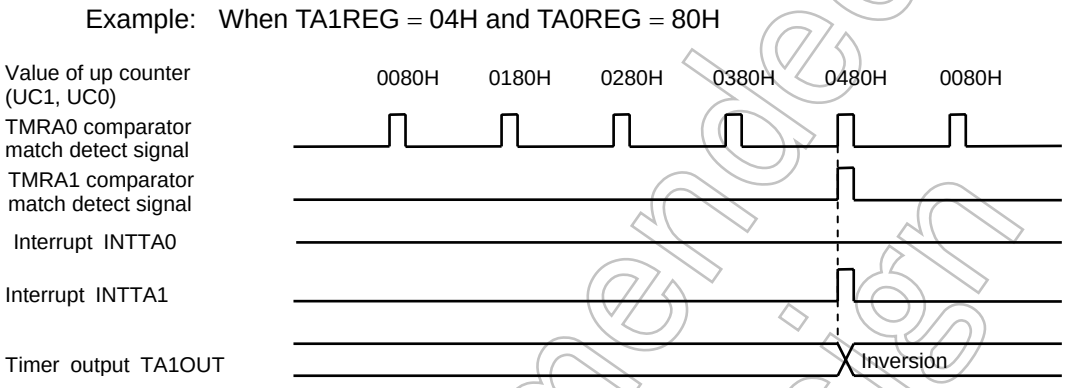



Figure 3.7.12 Timer Output by 16-Bit Timer Mode

(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 (Shared with PC1).

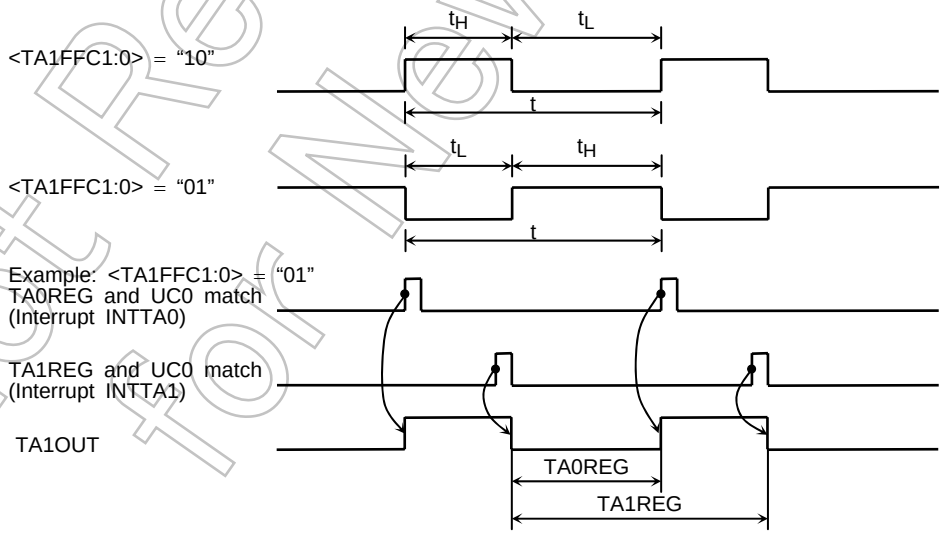



Figure 3.7.13 8-Bit PPG Output Waveforms

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 3.7.14 shows a block diagram representing this mode.

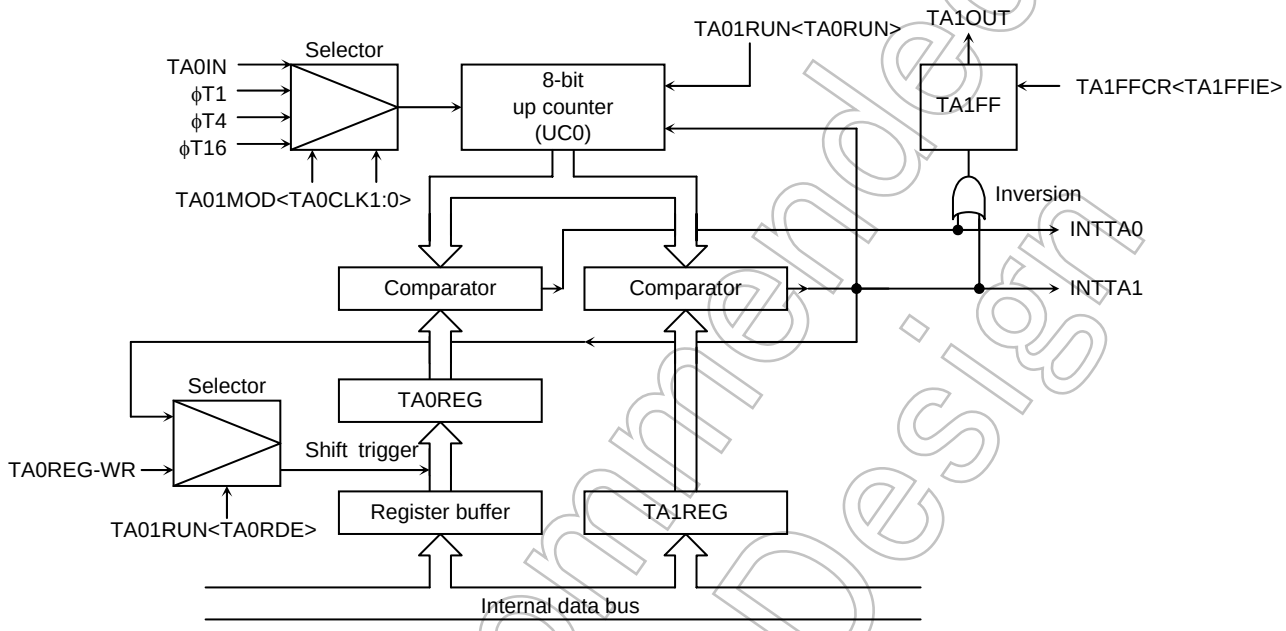



Figure 3.7.14 Block Diagram of 8-Bit PPG Output Mode

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).

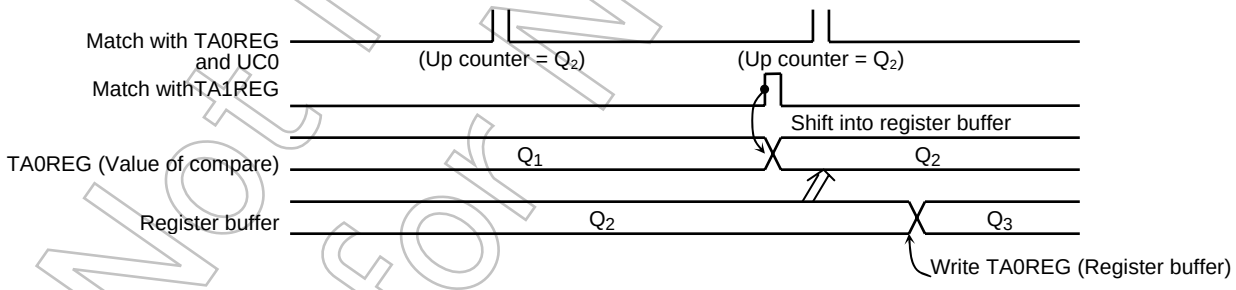
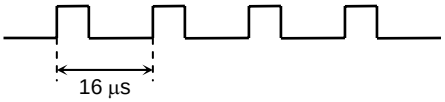



Figure 3.7.15 Operation of Register Buffer

Example: To generate 1/4 duty 62.5 kHz pulses (at $f_C$= 40 MHz):




Calculate the value that should be set in the timer register.

To obtain a frequency of 62.5 kHz, the pulse cycle t should be: t = 1/62.5 kHz = 16 μs

φT1 (= (16/fc)s (at $f_C$ = 40MHz);

16 μs/(16/fc)s = 40

Therefore set TA1REG to 40 (28H)

The duty is to be set to 1/4: t × 1/4 = 16 μs × 1/4 = 4 μs

4 μs/(16/fc)s = 10

Therefore, set TA0REG = 10 = 0AH

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TA01RUN | ← | 0 | X | X | X | – | 0 | 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 0AH. |
| TA1REG | ← | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | Write 28H. |
| TA1FFCR | ← | X | X | X | X | 0 | 1 | 1 | X | Set TA1FF and set inversion to enable. |
| | | | | | | | | | | → Writing “10” provides negative logic pulse. |
| PCCR | ← | X | – | – | X | – | X | 1 | – | Set PC1 to TA1OUT pin. |
| PCFC | ← | X | – | – | X | – | X | 1 | – | |
| 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 (which is also used as PC1). 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 of set for $2^n$ counter overflow

Value set in TA0REG ≠ 0

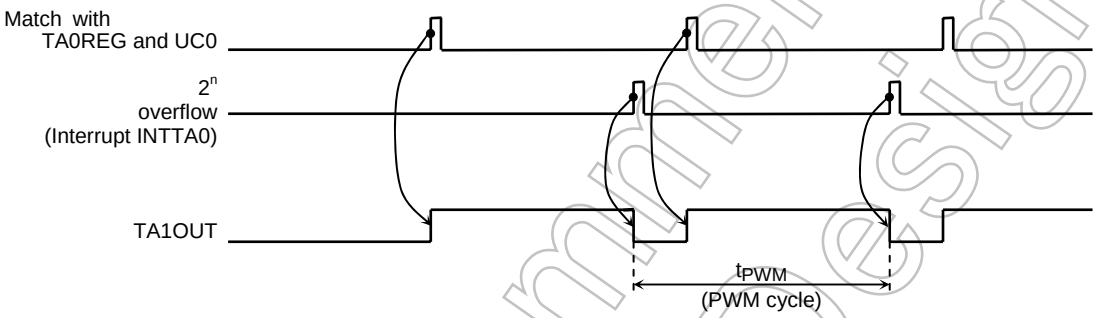



Figure 3.7.16 8-Bit Output Wave Form

Figure 3.7.17 shows a block diagram representing this mode.

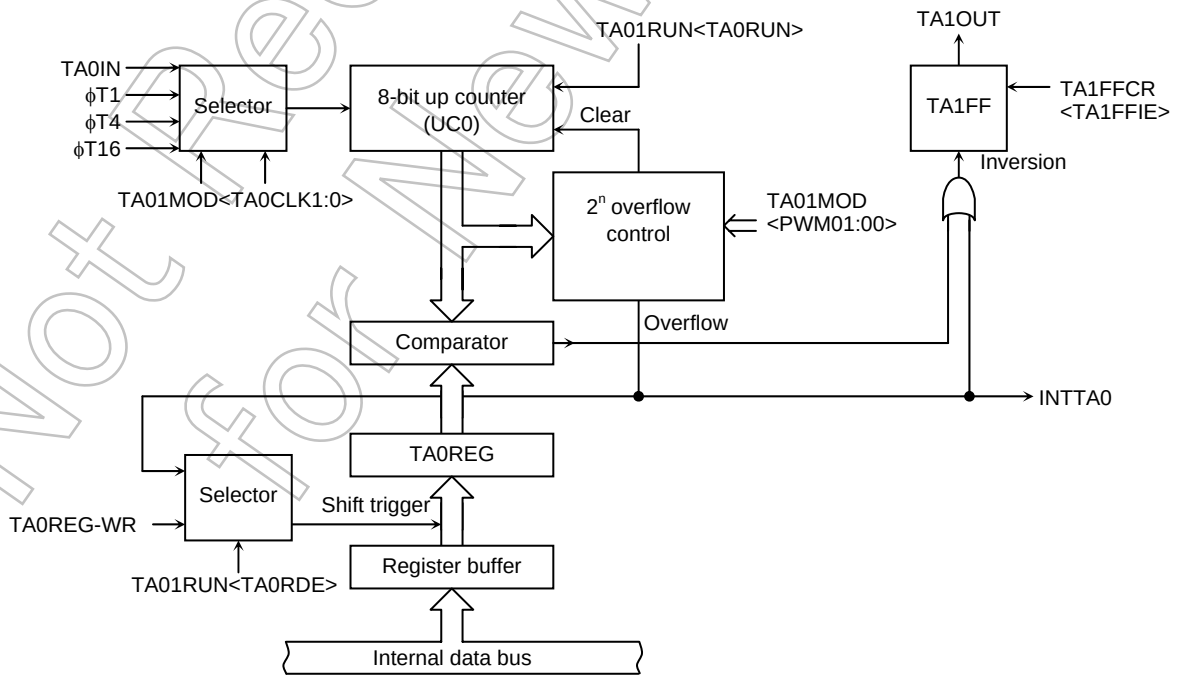



Figure 3.7.17 Block Diagram of 8-Bit PWM Output Mode

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.

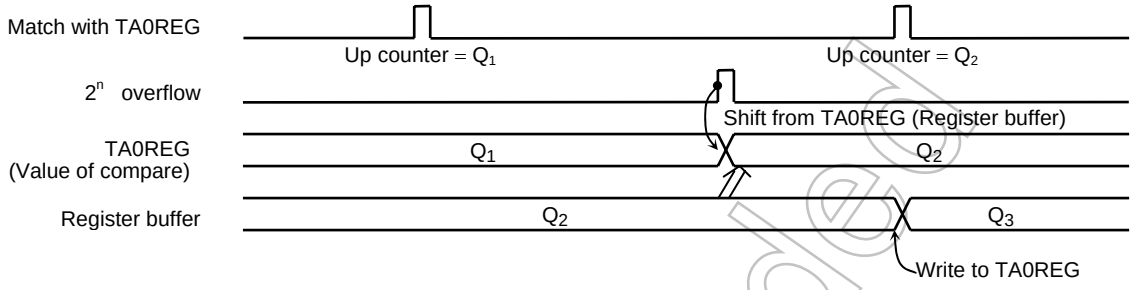
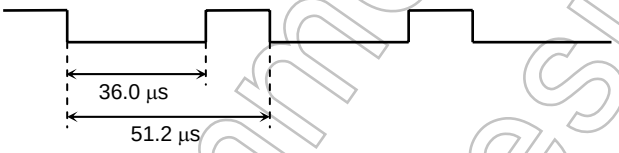



Figure 3.7.18 Operation of Register Buffer

Example: To output the following PWM waves on the TA1OUT pin at $f_C$ = 40 MHz:




To achieve a 51.2 μs PWM cycle by setting φT1=(16/fc)s (at $f_C$ = 40 MHz):

51.2 μs/(16/fc)s = 128 = $2^n$

Therefore n should be set to 7.

Since the low-level period is 36.0 μs when φT1 = (16/fc)s,

set the following value for TA0REG:

36.0 μs/(16/fc)s = 90 = 5AH

| | MSB 7 | 6 | 5 | 4 | 3 | 2 | 1 | LSB 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TA01RUN | ← – | X | X | X | – | – | – | 0 | Stop TMRA0 and clear it to 0. |
| TA01MOD | ← 1 | 1 | 1 | 0 | – | – | 0 | 1 | Select 8-bit PWM mode (cycle: $2^7$) and select φT1 as the input clock. |
| TA0REG | ← 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | Write 5AH. |
| TA1FFCR | ← X | X | X | X | 1 | 0 | 1 | X | Clear TA1FF to 0; set inversion to enable. |
| PCCR | ← X | – | – | X | – | X | 1 | – | Set PC1 to TA1OUT pin. |
| PCFC | ← X | – | – | X | – | X | 1 | – | |
| TA01RUN | ← 1 | X | X | X | – | 1 | – | 1 | Start TMRA0 counting. |

X : Don't care, – : No change

Table 3.7.4 Relationship of PWM Cycle and $2^n$ Counter

| Clock gear SYSCR1 <GEAR2:0> | System clock SYSCR0 <SYSCK> | – | 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) |
| 000(x1) | 0(fc) | ×8 | 1024/fc | 4096/fc | 16384/fc | 2048/fc | 8192/fc | 32768/fc | 4096/fc | 16384/fc | 65536/fc |
| 001(x2) | | | 2048/fc | 8192/fc | 32768/fc | 4096/fc | 16384/fc | 65536/fc | 8192/fc | 32768/fc | 131072/fc |
| 010(x4) | | | 4096/fc | 16384/fc | 65536/fc | 8192/fc | 32768/fc | 131072/fc | 16384/fc | 65536/fc | 262144/fc |
| 011(x8) | | | 8192/fc | 32768/fc | 131072/fc | 16384/fc | 65536/fc | 262144/fc | 32768/fc | 131072/fc | 524288/fc |
| 100(x16) | | | 16384/fc | 65536/fc | 262144/fc | 32768/fc | 131072/fc | 524288/fc | 65536/fc | 262144/fc | 1048576/fc |

(5) Mode settings

Table 3.7.5 shows the SFR settings for each mode.

Table 3.7.5 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 inversion select |
| 8-bit timer × 2 channels | 00 | – | Lower timer match, φT1, φT16, φT256 (00, 01, 10, 11) | External φT1, φT4, φT16 (00, 01, 10, 11) | 0: Lower timer output<br>1: Upper timer output |
| 16-bit timer mode | 01 | – | – | External φT1, φT4, φT16 (00, 01, 10, 11) | – |
| 8-bit PPG × 1 channel | 10 | – | – | External φT1, φT4, φT16 (00, 01, 10, 11) | – |
| 8-bit PWM × 1 channel | 11 | $2^6$, $2^7$, $2^8$ (01, 10, 11) | – | External φT1, φT4, φT16 (00, 01, 10, 11) | – |
| 8-bit timer × 1 channel | 11 | – | φT1, φT16, φT256 (01, 10, 11) | – | Output disable |

– : Don't care

## 3.8 16-Bit Timer/Event Counters (TMRB)

The TMP92CM22 contains 2 channels 16-bit timer/event counter (TMRB) which have the following operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable square wave pulse generation output mode (PPG: Variable duty cycle with variable period)

Can be used following operation modes by capture function:

- Frequency measurement mode
- Pulse width measurement mode
- Time differential measurement mode

Figure 3.8.1 to Figure 3.8.2 show block diagram of TMRB0 and TMRB1. Each timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (One of them with a double-buffer structure), two 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and a control circuit.

Each timer/event counter is controlled by 11-byte control register (SFR).

This chapter consists of the following items:



Table 3.8.1 Pins and SFR of TMRB

| Spec \ Channel | | TMRB0 | TMRB1 |
|---|---|---|---|
| External pin | External clock/ Caputre triggr input pin | None | TB1IN0 (Share with PD0)<br>TB1IN1 (Share with PD1) |
| | Timer flip-flop output pin | TB0OUT0<br>(Share with PC6) | TB1OUT0 (Share with PD2)<br>TB1OUT1 (Share with PD3) |
| SFR (Address) | Timre run register | TB0RUN (1180H) | TB1RUN (1190H) |
| | Timrer mode register | TB0MOD (1182H) | TB1MOD (1192H) |
| | Timre flip-flop control register | TB0FFCR (1183H) | TB1FFCR (1193H) |
| | Timer register | TB0RG0L (1188H) | TB1RG0L (1198H) |
| | | TB0RG0H (1189H) | TB1RG0H (1199H) |
| | | TB0RG1L (118AH) | TB1RG1L (119AH) |
| | | TB0RG1H (118BH) | TB1RG1H (119BH) |
| | Capture register | TB0CP0L (118CH) | TB1CP0L (119CH) |
| | | TB0CP0H (118DH) | TB1CP0H (119DH) |
| | | TB0CP1L (118EH) | TB1CP1L (119EH) |
| | | TB0CP1H (118FH) | TB1CP1H (119FH) |

### 3.8.1 Block Diagram

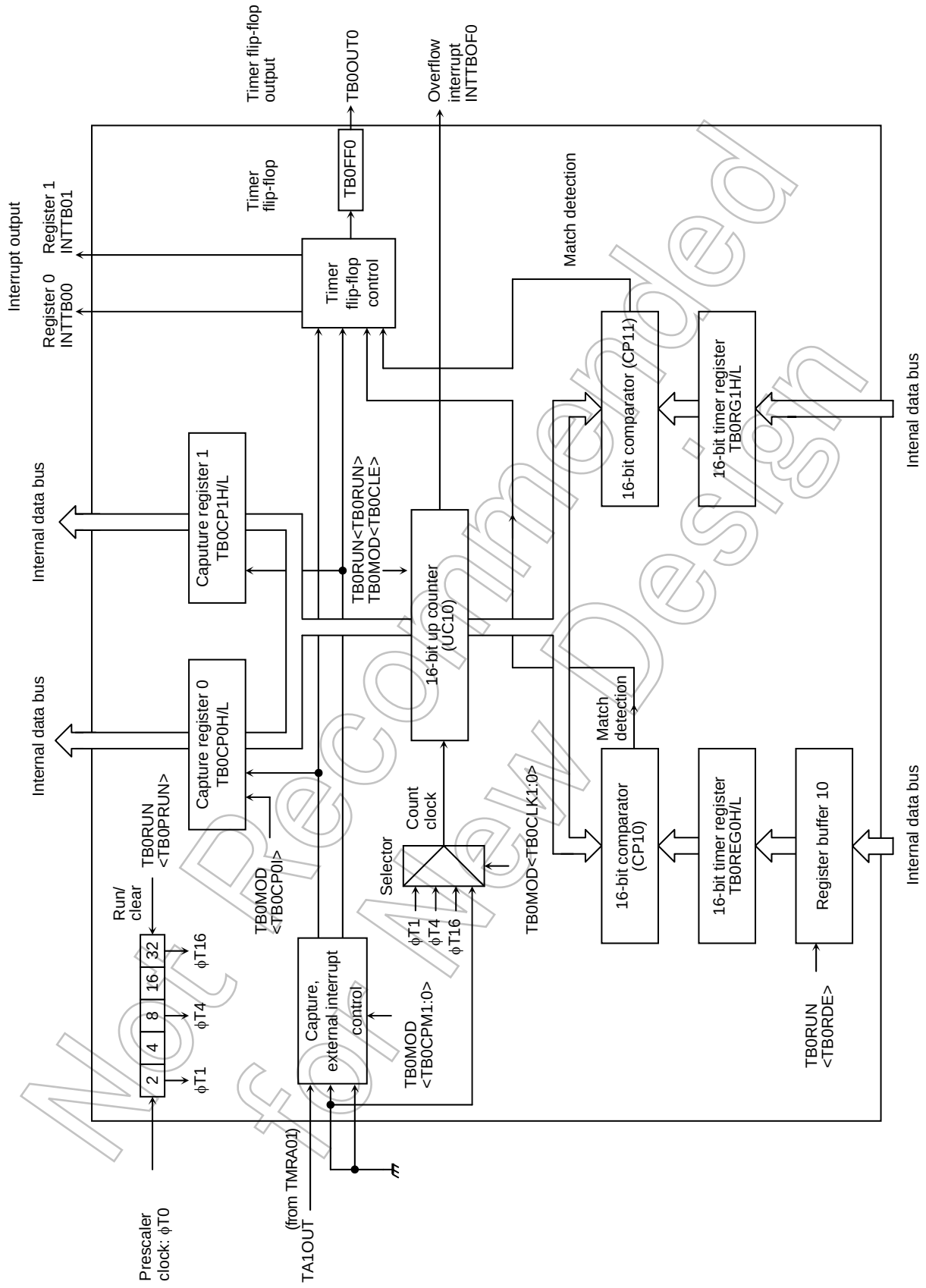



Figure 3.8.1 Block Diagram of TMRB0

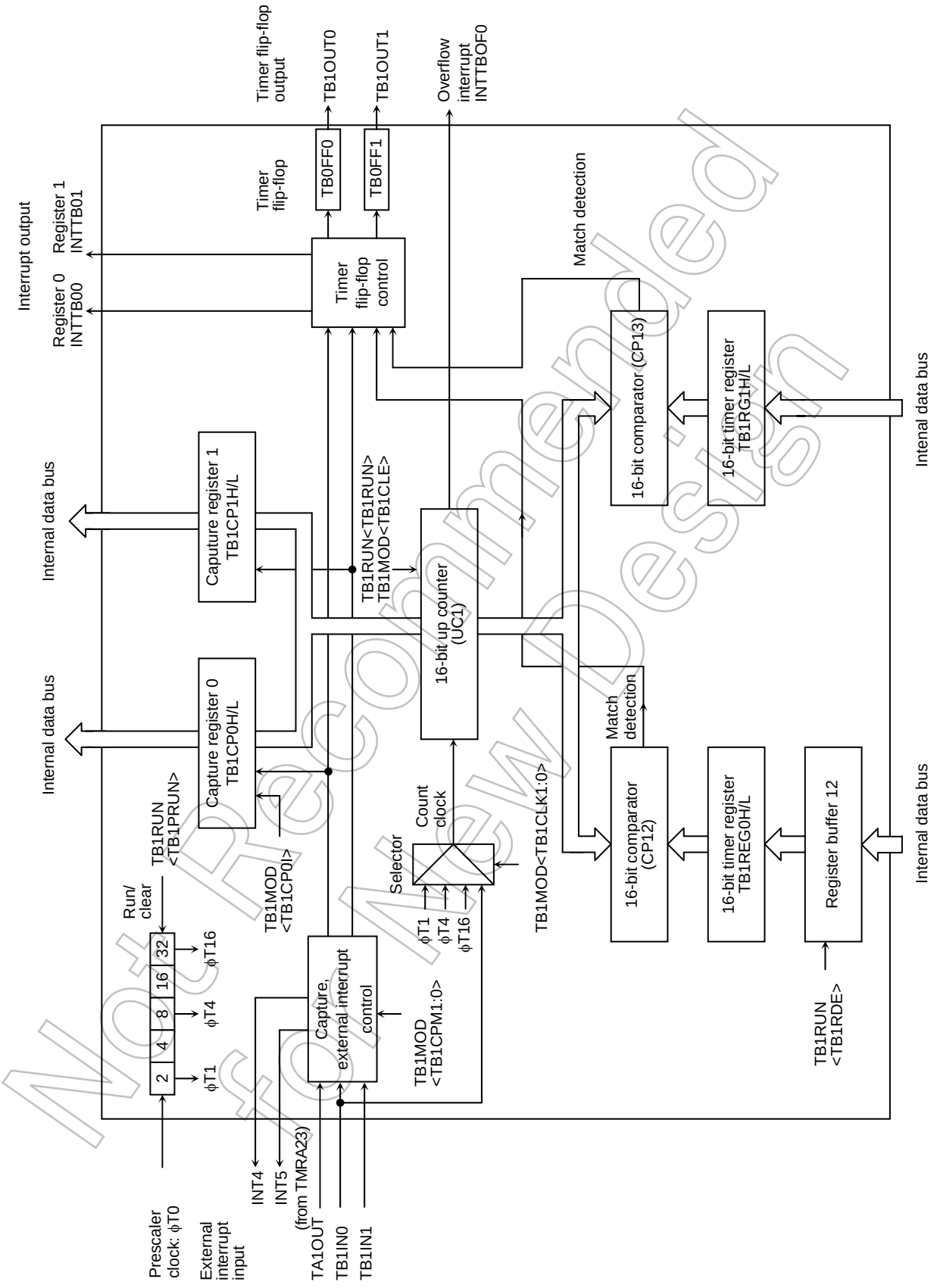



Figure 3.8.2 Block Diagram of TMRB1

### 3.8.2 Operation

(1) Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (ϕT0) is a divided clock (Divided by 8) from selected clock by the register SYSCR1<GEAR1:0> of clock gear.

This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0PRUN> is set to 1; the prescaler is cleared to zero and stops operation when <TB0PRUN> is cleared to 0.

Table 3.8.2 show prescaler output clock resolution.

Table 3.8.2 Prescaler Output Clock Resolution

| Clock gear selection SYSCR1 <GEAR2:0> | – | Timer counter input clock TMRB prescaler TB0MOD<TB0CLK1:0> | | |
|---|---|---|---|---|
| | | ϕT1(1/2) | ϕT4 (1/8) | ϕT16 (1/32) |
| 000 (1/1) | 1/8 | fc/16 | fc/64 | fc/256 |
| 001 (1/2) | | fc/32 | fc/128 | fc/512 |
| 010 (1/4) | | fc/64 | fc/256 | fc/1024 |
| 011 (1/8) | | fc/128 | fc/512 | fc/2048 |
| 100 (1/16) | | fc/256 | fc/1024 | fc/4096 |

(2) Up counter (UC10)

UC10 is a 16-bit binary counter that counts up according to input from the clock specified by TB0MOD<TB0CLK1:0> register.

As the input clock, one of the prescaler internal clocks ϕT1, ϕT4, and ϕT16 can be selected. Counting or stopping and clearing of the counter is controlled by timer operation control register TB0RUN<TB0RUN>. And an external clock from TB1IN0 pin can be selected in TB1MOD.

When clearing is enabled, the up counter UC10 will be cleared to zero each time its value matches the value in the timer register TB0RG1H/L. Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

If clearing is disabled, the counter operates as a free-running counter.

A timer overflow interrupt (INTTBOF0) is generated when UC10 overflow occurs.

(3) Timer registers (TB0RG0H/L and 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 TB0RG0H/L and TB0RG1H/L is always needed. For example, either using 2-byte data transfer instruction or using 1-byte data transfer instruction twice for lower 8 bits and upper 8 bits in order.

The TB0RG0H/L timer register has a double-buffer structure, which is paired with 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.

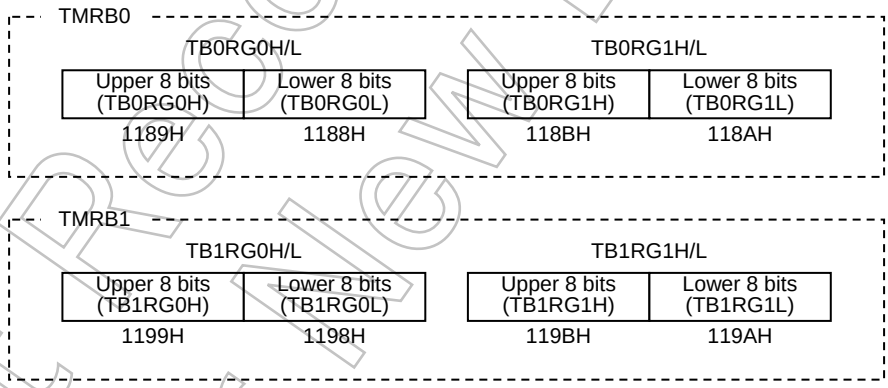
When the double buffer is enabled, data is transferred from the register buffer to the timer register when the values in the up counter (UC10) and the timer register TB0RG1H/L match.

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 as shown below.

TB0RG0H/L and the register buffer both have the same memory addresses (001188H and 001189H) allocated to them. If <TB0RDE> = 0, the value is written to both the timer register and the register buffer. If <TB0RDE> = 1, the value is written to the register buffer only.

The addresses of the timer registers are as follows:



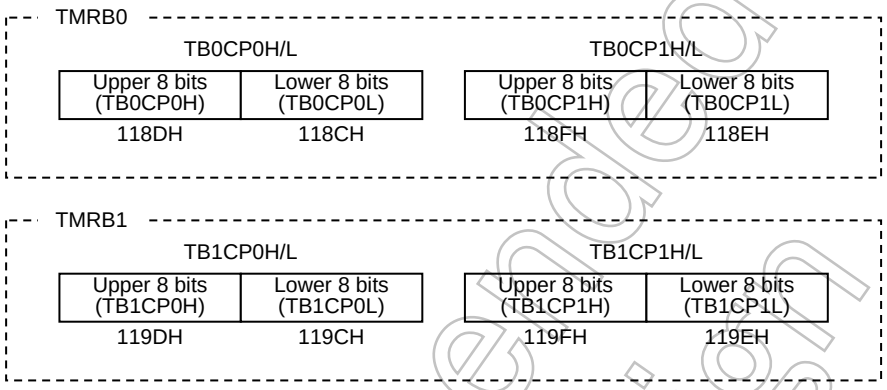

The timer registers are write-only registers and thus cannot be read.

(4) Capture registers (TB0CP0H/L, TB0CP1H/L, TB1CP0H/L and TB1CP1H/L)

These 16-bit registers are used to latch the values in the up counters UC10.

Data in the capture registers should be read both upper and lower all 16 bits. For example, using 2-byte data transfer instruction or using 1-byte data transfer instruction twice for lower 8 bits and upper 8 bits in order.

The addresses of the capture registers are as follows:




The capture registers are read-only registers and thus cannot be written.

(5) Capture and external interrupt control

This circuit controls the timing to latch the value of up counter UC10 into TB0CP0H/L, TB0CP1H/L and generating for external interrupt.

Interrupt timing of capture register and selection edge of external interrupt are set by TB0MOD<TB0CPM1:0>. (TMRB0 does not include the selection edge of external interrupt.) External interrupt INT5 is fixed to rising edge.

The value in the up counter (UC10) can be loaded into a capture register by software. Whenever 0 is programmed to TB0MOD<TB0CP0I>, the current value in the up counter is loaded into capture register TB0CP0. It is necessary to keep the prescaler in Run mode (e.g., TB0RUN<TB0PRUN> must be held at a value of 1).

(6) Comparators (CP10 and 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-flop (TB0FF0 and TB0FF1)

These flip-flops (TB0FF0 and TB0FF1) 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<TB0C0T1, TB0E1T1, TB0E0T1>.

After a reset the values of TB0FF0 and TB0FF1 are undefined. If "00" is programmed to TB0FFCR<TB0FF0C1:0> or <TB0FF1C1:0>, TB0FF0 will be inverted. If "01" is programmed to the capture registers, the value of TB0FF0 will be set to "1". If "10" is programmed to the capture registers, the value of TB0FF0 will be cleared to "0".

The values of TB0FF0 can be output via the timer output pins TB0OUT0 (which is shared with PC6). Timer output should be specified using the port C function register.

### 3.8.3 SFRs

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 |
| After reset | 0 | 0 | | | 0 | 0 | | 0 |
| Function | Double buffer 0: Disable 1: Enable | Always write “0”. | | | IDLE2 0: Stop 1: Operate | TMRB0 Prescaler 0: Stop and clear 1: Run (Count) | | Up counter UC10 |

Count operation

| | |
|---|---|
| 0 | Stop and clear |
| 1 | Count |

Note: The values of bits 1, 4, and 5 of TB0RUN are undefined when read.

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 |
| After reset | 0 | 0 | | | 0 | 0 | | 0 |
| Function | Double buffer 0: Disable 1: Enable | Always write “0”. | | | IDLE2 0: Stop 1: Operate | TMRB1 Prescaler 0: Stop and clear 1: Run (Count) | | Up counter UC12 |

Count operation

| | |
|---|---|
| 0 | Stop and clear |
| 1 | Count |

Note: The values of bits 1, 4, and 5 of TB1RUN are undefined when read.

Figure 3.8.3 Register for TMRB

TMRB0 Mode Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | – | TB0CP0I | TB0CPM1 | TB0CPM0 | TB0CLE | TB0CLK1 | TB0CLK0 |
| Read/Write | R/W | | W | R/W | | | | |
| After reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | Always write “0”. | Always write “0”. | Software capture control<br>0: Software capturer<br>1: Undefined | Capture timing<br>00: Disable<br>01: (Reserved)<br>10: (Reserved)<br>11: TA1OUT↑ TA1OUT↓ | | Up counter control<br>0: Clear disable<br>1: Clear enable | TMRB0 source clock<br>00: (Reserved)<br>01: ϕT1<br>10: ϕT4<br>11: ϕT16 | |

TB0MOD (1182H)

Read-modify-write instruction is prohibited

→ Input clock

| | |
|---|---|
| 00 | Reserved |
| 01 | ϕT1 |
| 10 | ϕT4 |
| 11 | ϕT16 |

→ Clear up counter 0(UC0)

| | |
|---|---|
| 0 | Disable |
| 1 | Enable clearing on match with TB0RG1H/L |

→ Capture/interrupt timing

| | Capture control |
|---|---|
| 00 | Disable |
| 01 | (Reserved) |
| 10 | (Reserved) |
| 11 | Capture to TB0CP0H/L at rising edge of TA1OUT<br>Capture to TB0CP1H/L at falling edge of TA1OUT |

→ Software capture

| | |
|---|---|
| 0 | Capture value of up counter to TB0CP0H/L |
| 1 | Undefined |



Figure 3.8.4 Register for TMRB

TMRB1 Mode Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TB1CT1 | TB1ET1 | TB1CP0I | TB1CPM1 | TB1CPM0 | TB1CLE | TB1CLK1 | TB1CLK0 |
| Read/Write | R/W | | W | R/W | | | | |
| After reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | TB1FF1 Inversion trigger<br>0: Trigger disable<br>1: Trigger enable<br>Invert when UC12 is loaded into TB1CP1H/L | Invert when UC12 matches with TB1RG1H/L | Software capture control<br>0: Software capture<br>1: Undefined | Capture timing<br>00: Disable INT4 is rising edge<br>01: TB1N0 ↑ TB1IN1 ↑ INT4 is falling edge<br>10: TB1IN0 ↑ TB1IN0 ↓ INT4 is falling edge<br>11: TA1TRG ↑ TA1TRG ↓ INT4 is rising edge | | Up counter control<br>0: Clear disable<br>1: Clear enable | TMRB1 source clock<br>00: TB1IN0 pin input<br>01: ϕT1<br>10: ϕT4<br>11: ϕT16 | |

TB1MOD (1192H)

Read-modify-write instruction is prohibited

Input clock

| | |
|---|---|
| 00 | TB1IN0 pin input |
| 01 | ϕT1 |
| 10 | ϕT4 |
| 11 | ϕT16 |

Clear up counter (UC12)

| | |
|---|---|
| 0 | Clear disable |
| 1 | Clear by matching with TB1RG1H/L |

Capture/interrupt timing

| | Capture control | INT4 control |
|---|---|---|
| 00 | Capture disable | Generate INT4 by TB1IN0 rising |
| 01 | Capture to TB1CP0H/L at rising edge of TB1IN0<br>Capture to TB1CP1H/L at rising edge of TB1IN1 | |
| 10 | Capture to TB1CP0H/L at rising edge of TB1IN0<br>Capture to TB1CP1H/L at falling edge of TB1IN1 | Generate INT4 by TB1IN0 falling |
| 11 | Capture to TB1CP0H/L at rising edge of TA1OUT<br>Capture to TB1CP1H/L at falling edge of TA1OUT | Generate INT4 by TB1IN0 rising |

Software capture

| | |
|---|---|
| 0 | Capture value of up counter to TB1CP0H/L |
| 1 | Undefined |



Figure 3.8.5 Register for TMRB

TMRB0 Flip-flop Control Register

TB0FFCR (1183H)

Read-modify -write instruction is prohibited

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | – | TB0C1T1 | TB0C0T1 | TB0E1T1 | TB0E0T1 | TB0FFC1 | TB0FFC0 |
| Read/Write | W | | R/W | | | | W* | |
| After reset | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| Function | Always write “11”. | | TB0FF0 inversion trigger<br>0: Trigger disable<br>1: Trigger enable | | | | TB0FF0 control<br>00: Invert<br>01: Set<br>10: Clear<br>11: Don’t care<br>* Always read as “11”. | |
| | | | Invert when the UC10 value is loaded into TB0CP1H/L | Invert when the UC10 value is loaded into TB0CP0H/L | Invert when the UC10 matches with TB0RG1H/L | Invert when the UC10 match with TB0RG0H/L | | |

Timer flip-flop TB0 (TB0FF0) control

| | |
|---|---|
| 00 | Invert |
| 01 | Set to “1”. |
| 10 | Set to “0”. |
| 11 | Don’t care. |

Inverted when the UC10 value matches the value in TB0RG0H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

Inverted when the UC10 value matches the value in TB0RG1H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

Inverted when the UC10 value is loaded into TB0CP0H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

Inverted when the UC10 value is loaded into TB0CP1H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |



Figure 3.8.6 Register for TMRB

TMRB1 Flip-flop Control Register

TB1FFCR (1193H)

Read-modify -write instruction is prohibited

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TB1FF1C1 | TB1FF1C0 | TB1C1T1 | TB1C0T1 | TB1E1T1 | TB1E0T1 | TB1FFC1 | TB1FFC0 |
| Read/Write | W* | | R/W | | | | W* | |
| After reset | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| Function | TB1FF1 control<br>00: Invert<br>01: Set<br>10: Clear<br>11: Don't care<br>* Always read as "11". | | TB1FF0 inversion trigger<br>0: Trigger disable<br>1: Trigger enable | | | | TB1FF0 control<br>00: Invert<br>01: Set<br>10: Clear<br>11: Don't care<br>* Always read as "11". | |
| | | | Invert when the UC12 value is loaded into TB1CP1H/L. | Invert when the UC12 value is loaded into TB1CP0H/L. | Invert when the UC12 matches with TB1RG1H/L. | Invert when the UC12 match with TB1RG0H/L. | | |

Timer flip-flop TB1 (TB1FF0) control

| | |
|---|---|
| 00 | Invert |
| 01 | Set to "1". |
| 10 | Set to "0". |
| 11 | Don't care |

Inverted when the UC12 value matches the value in TB1RG0H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

Inverted when the UC12 value matches the value in TB1RG1H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

Inverted when the UC12 value is loaded into TB1CP0H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

I Inverted when the UC12 value is loaded into TB1CP1H/L

| | |
|---|---|
| 0 | Disable inversion |
| 1 | Enable inversion |

TB1FF1 control

| | |
|---|---|
| 00 | Invert value of TB1FF1 |
| 01 | Set TB1FF1 to "1". |
| 10 | Set TB1FF1 to "0". |
| 11 | Don't care |



Figure 3.8.7 Register for TMRB

TMRB0 register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| TB0RG0L (1188H) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0RG0H (1189H) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0RG1L (118AH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0RG1H (118BH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0CP0L (118CH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0CP0H (118DH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0CP1L (118EH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB0CP1H (118FH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |

TMRB1 register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| TB1RG0L (1198H) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1RG0H (1199H) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1RG1L (119AH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1RG1H (119BH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1CP0L (119CH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1CP0H (119DH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1CP1L (119EH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |
| TB1CP1H (119FH) | bit Symbol | – | | | | | | | |
| | Read/Write | W | | | | | | | |
| | After reset | Undefined | | | | | | | |

Note: All registers are prohibited to execute read-modify-write instruction.

Figure 3.8.8 Register for TMRB

### 3.8.4 Operation in Each Mode

(1) 16-bit interval timer mode

Generating interrupts at fixed intervals in this example, the interval time is set the timer register TB0RG1H/L to generate the interrupt INTTB01.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TB0RUN | ← | 0 | 0 | X | X | – | 0 | X | 0 | Stop TMRB0. |
| INTETB0 | ← | X | 1 | 0 | 0 | X | 0 | 0 | 0 | Enable INTTB01 and 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 | * | * | Set input clock to prescaler clock, and set capture function to disable. |
| | | | | | | (** = 01, 10, 11) | | | | |
| TB0RG1 | ← | * | * | * | * | * | * | * | * | Set the interval time. |
| | | * | * | * | * | * | * | * | * | (16 bits) |
| TB0RUN | ← | 0 | 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 (TB1IN0 pin input) as the input clock.

Up counter counting up by rising edge of TB1IN0 pin input. And execution software capture and reading capture value enable reading count value.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TB1RUN | ← | 0 | 0 | X | X | – | 0 | X | 0 | Stop TMRB1. |
| PDCR | ← | X | X | X | X | – | – | – | 0 | Set PD0 to TB1IN0 input mode. |
| PDFC | ← | X | X | X | X | – | – | – | 1 | |
| INTETB1 | ← | X | 1 | 0 | 0 | X | 0 | 0 | 0 | Set INTTB11 to enable (Interrupt level4). Set INTTB00 to disable. |
| TB1FFCR | ← | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | Set trigger to disable. |
| TB1MOD | ← | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | Set input clock to TB1IN0 pin input. |
| TB1RG1 | ← | * | * | * | * | * | * | * | * | Set number of count. |
| | | * | * | * | * | * | * | * | * | (16 bits) |
| TB1RUN | ← | 0 | 0 | X | X | – | 1 | X | 1 | Start TMRB1. |

X: Don't care, –: No change

Note: When used as an event counter, set the prescaler to "RUN" (TB1RUN<TB1PRUN> = "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 to be enabled by the match of the up counter UC10 with timer register TB0RG0H/L or TB0RG1H/L and to be output to TB0OUT0. In this mode, the following conditions must be satisfied.

(Set value of TB0RG0H/L) < (Set value of TB0RG1H/L)

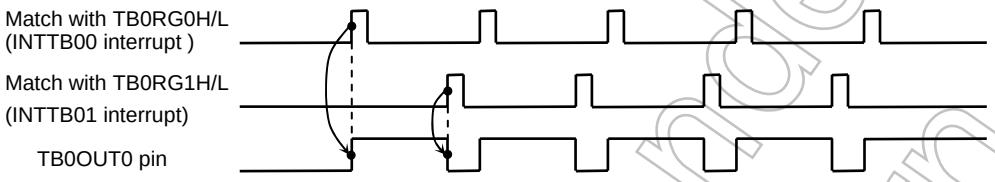



Figure 3.8.9 Programmable Pulse Generation (PPG) Output Waveforms

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 makes easy the handling of low-duty waves.

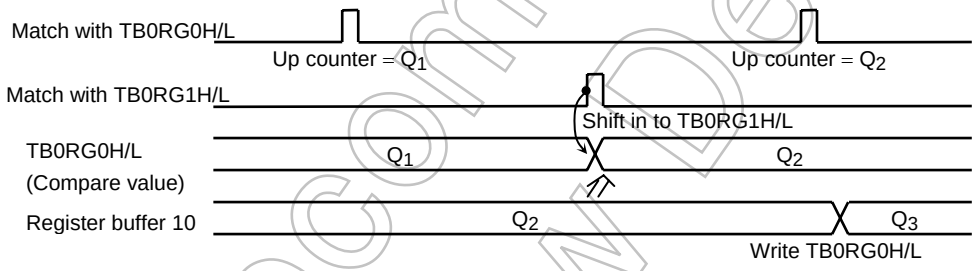



Figure 3.8.10 Operation of Register Buffer

The following block diagram illustrates this mode.

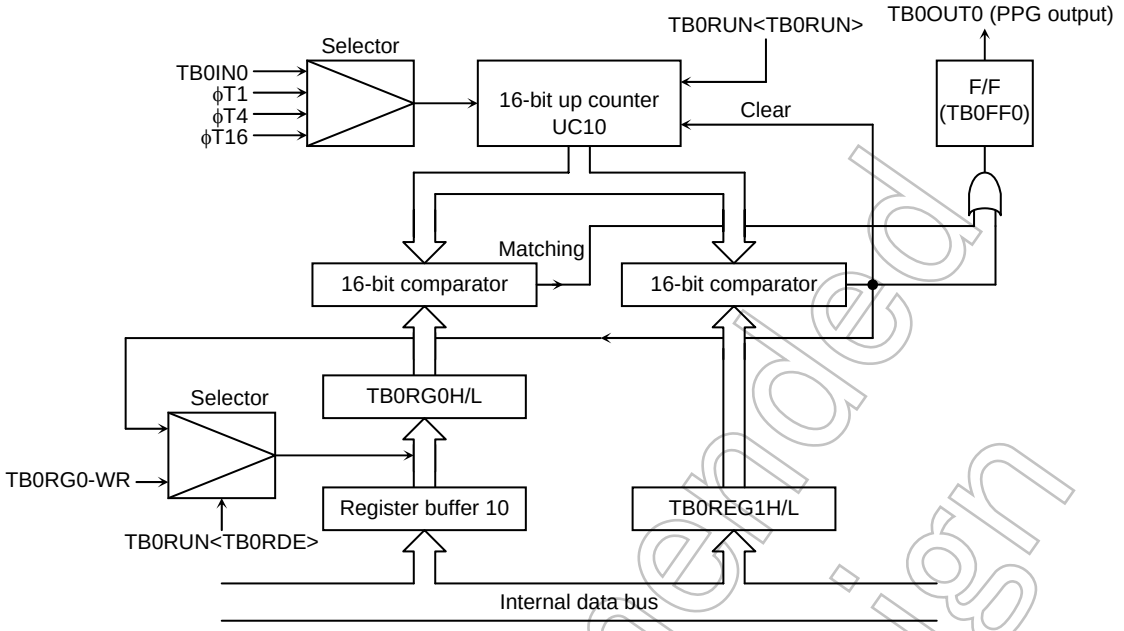



Figure 3.8.11 Block Diagram of 16-Bit PPG Mode

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 | – | 0 | X | 0 | Disable the TB0RG0H/L double buffer and stop TMRB0. |
| TB0RG0H/L | ← | * | * | * | * | * | * | * | * | Set the duty ratio. |
| | | * | * | * | * | * | * | * | * | (16 bits) |
| TB0RG1H/L | ← | * | * | * | * | * | * | * | * | Set the frequency. |
| | | * | * | * | * | * | * | * | * | (16 bits) |
| TB0RUN | ← | 1 | 0 | X | X | – | 0 | X | 0 | Enable the TB0RG0 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. Clear TB0FF0 to 0. |
| TB0MOD | ← | 0 | 0 | 1 | 0 | 0 | 1 | * | * | Set input clock to prescaler output clock and disable the capture function. |
| | | | | | | (** = 01, 10, 11) | | | | |
| PCCR | ← | X | 1 | – | X | – | X | – | – | Set PC6 to function as TB0OUT0. |
| PCFC | ← | X | 1 | – | X | – | X | – | – | |
| TB0RUN | ← | 1 | 0 | X | X | – | 1 | X | 1 | Start TMRB0. |

X : Don't care, – : No change

(4) Capture function examples

Used capture function, they can be applicable in many ways, for example:

1. One-shot pulse output from external trigger pulse
2. Frequency measurement
3. Pulse width measurement
4. Measurement of difference time

1. One-shot pulse output from external trigger pulse

Set the up counter UC12 in free-running mode with the internal input clock, input the external trigger pulse from TB1IN0 pin, and load the value of up counter into capture register TB1CP0H/L at the rise edge of external trigger pulse.

When the interrupt INT4 is generated at the rise edge of external trigger pulse, set the TB1CP0H/L value (c) plus a delay time (d) to TB1RG0H/L (= c + d), and set the above set value (c + d) plus a one-shot width (p) to TB1RG1H/L (= c + d + p). And, set "11" to timer flip-flop control register TB1FFCR<TB1E1T1, TB1E0T1>. Set to trigger enable for be inverted timer flip-flop TB1FF0 by UC12 matching with TB1RG0H/L and with TB1RG1H/L. When interrupt INTTB11 occurs, this inversion will be disabled after one-shot pulse is output.

The (c), (d), and (p) correspond to c, d, and p in Figure 3.8.12.

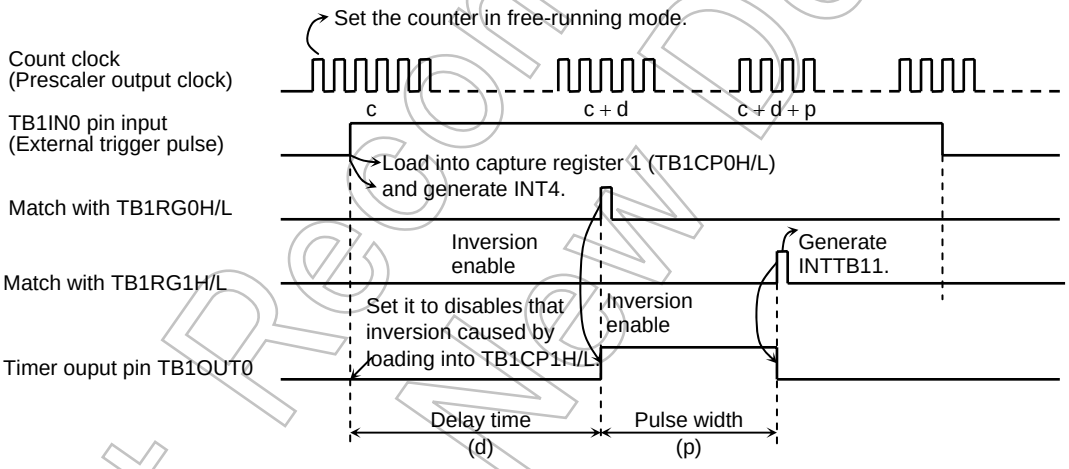
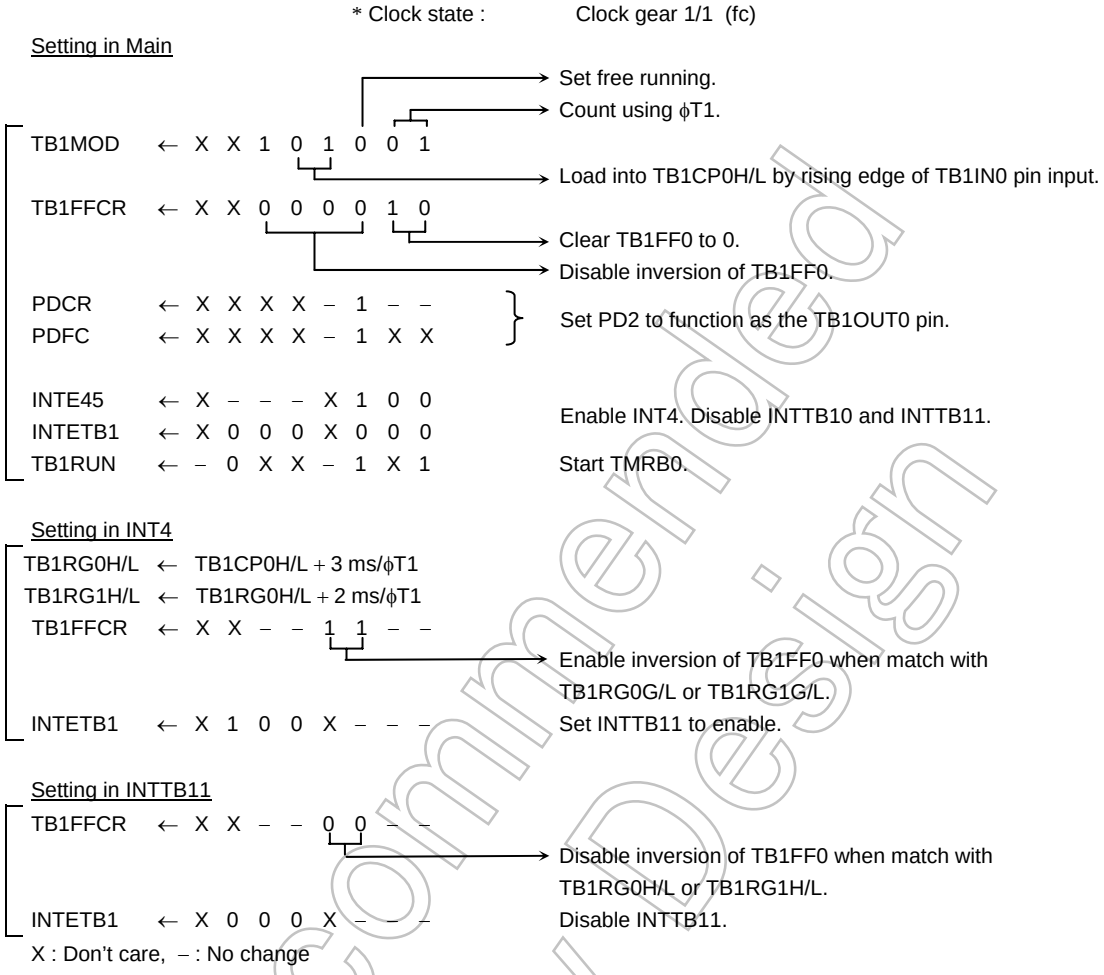



Figure 3.8.12 One-shot Pulse Output (with delay)

Example: To output a 2 [ms] one-shot pulse with a 3 [ms] delay to the external trigger pulse via the TB1IN0 pin.




When delay time is unnecessary, invert timer flip-flop TB1FF0 when up counter value is loaded into capture register (TB1CP0H/L), and set the TB1CP0H/L value (c) plus the one-shot pulse width (p) to TB0RG1H/L when the interrupt INT4 occurs. The TB1FF0 inversion should be enable when the up counter (UC12) value matches TB1RG1H/L, and disabled when generating the interrupt INTTB11.

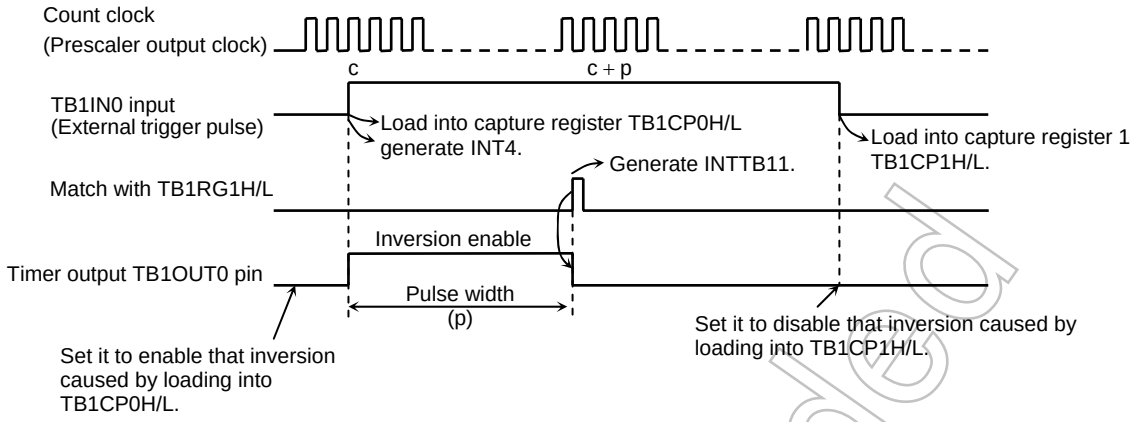



Figure 3.8.13 One-shot Pulse Output (without delay)

2. Frequency measurement

The frequency of the external clock can be measured in this mode. Frequency is measured by the 8-bit timers TMRA23 and the 16-bit timer/event counter.

TMRA23 is used to setting of measurement time by inversion TA3FF.

Counter clock in TMRB0 select TB1IN0 pin input, and count by external clock input. Set to TB1MOD<TB1CPM1:0> = "11". The value of the up counter (UC12) is loaded into the capture register TB0CP0H/L at the rise edge of the timer flip-flop TA1FF of 8-bit timers (TMRA1), and into TB0CP1H/L at its fall edge.

The frequency is calculated by difference between the loaded values in TB1CP0H/L and TB1CP1H/L when the interrupt (INTTA2 or INTTA3) is generates by either 8-bit timer.

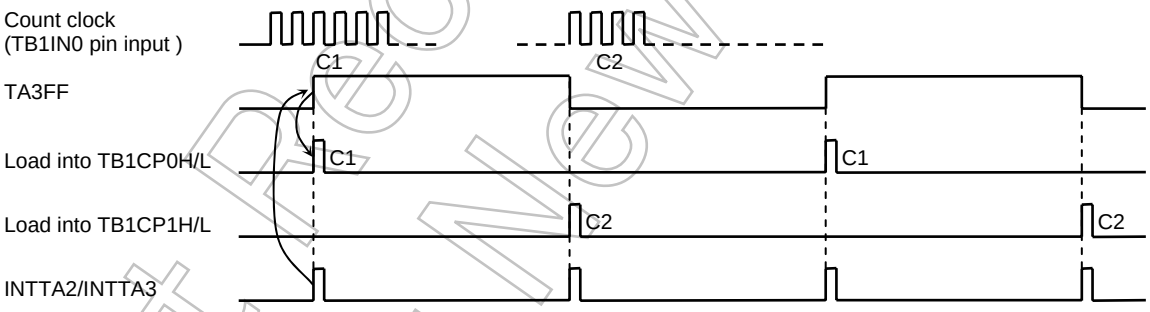



Figure 3.8.14 Frequency Measurement

For example, if the value for the level 1 width of TA3FF of the 8-bit timer is set to 0.5 s and the difference between the values in TB1CP0H/L and TB1CP1H/L is 100, the frequency is 100 ÷ 0.5 s = 200 Hz.

3. Pulse width measurement

This mode allows measuring the high level width of an external pulse. While keeping the 16-bit timer/event counter counting (Free running) with the prescaler output clock input, external pulse is input through the TB1IN0 pin. Then the capture function is used to load the UC12 values into TB1CP0H/L and TB1CP1H/L at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT4 occurs at the falling edge of TB1IN0.

The pulse width is obtained from the difference between the values of TB1CP0H/L and TB1CP1H/L and the internal clock cycle.

For example, if the prescaler output clock is 0.8 μs and the difference between TB1CP0H/L and TB1CP1H/L is 100, the pulse width will be 100 × 0.8 μs = 80 μs.

Additionally, the pulse width that is over the UC12 maximum count time specified by the clock source can be measured by changing software.

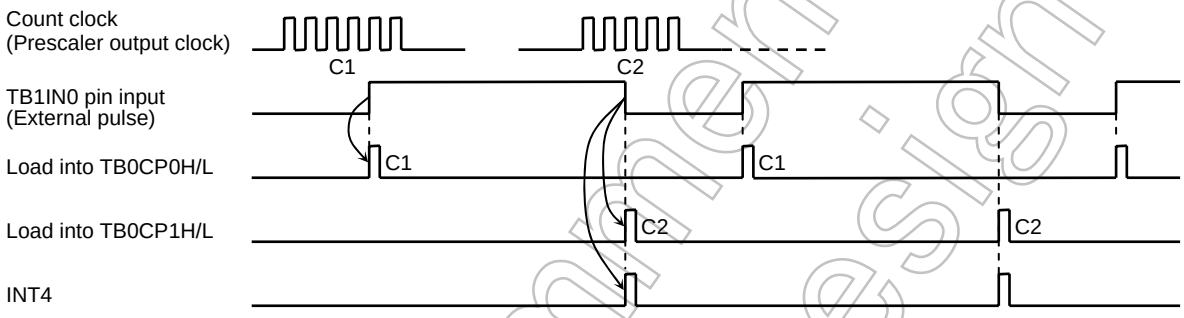



Figure 3.8.15 Pulse Width Measurement

Note: Pulse Width measure by setting “10” to TB1MOD<TB1CPM1:0>. The external interrupt INT4 is generated in timing of falling edge of TB1IN0 input. In other modes, it is generated in timing of rising edge of TB1IN0 input.

The width of low level can be measured from the difference between the first C2 and the second C1 at the second INT4 interrupt.

4. Measurement of difference time

This mode is used to measure the difference in time between the rising edges of external pulses input through TB1IN0 and TB1IN1.

Keep the 16-bit timer/event counter (TMRB1) counting (Free running) with the prescaler output clock, and load the UC12 value into TB1CP0H/L at the rising edge of the input pulse to TB1IN0. Then the interrupt INT4 is generated.

Similarly, the UC012 value is loaded into TB1CP1H/L at the rising edge of the input pulse to TB1IN1, generating the interrupt INT5.

The time difference between these pulses can be obtained by multiplying the value subtracted TB1CP0H/L from TB1CP1H/L and the internal clock cycle together at which loading the UC12 value into TB1CP0H/L and TB1CP1H/L has been done.

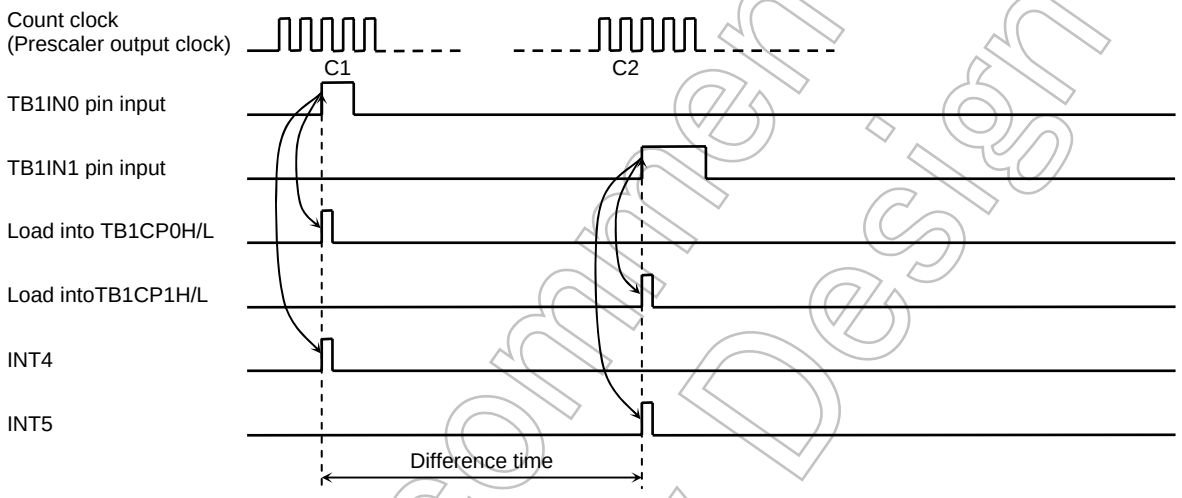



Figure 3.8.16 Measurement of Difference Time

## 3.9 Serial Channels (SIO)

The TMP92CM22 includes 2 serial I/O channels. Each channel is called SIO0 and SIO1. For both channels either UART Mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission) can be selected.

- 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 (Multi-controller system).

Figure 3.9.2 and Figure 3.9.3 are block diagrams for each channel. Each channel is structured in prescaler, serial clock generation circuit, receiving buffer and control circuit, and transfer buffer and control circuit.

Serial channels 0 and 1 can be used independently.

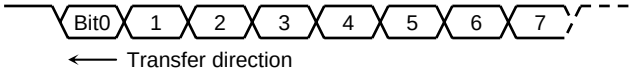
Both channels operate in the same function except for the following points; hence only the operation of channel 0 is explained below.

Table 3.9.1 Differences between Channels 0 to 1

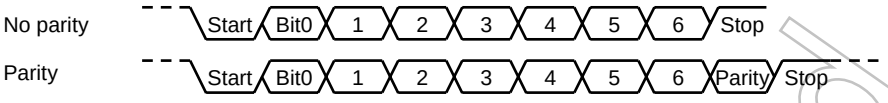
| | Channel 0 | Channel 1 |
|---|---|---|
| Pin name | TXD0 (PF0)<br>RXD0 (PF1)<br>$\overline{CTS0}$ /SCLK0 (PF2) | TXD1 (PF3)<br>RXD1 (PF4)<br>$\overline{CTS1}$ /SCLK1 (PF5) |
| IrDA mode | Yes | No |

This chapter contains the following sections:

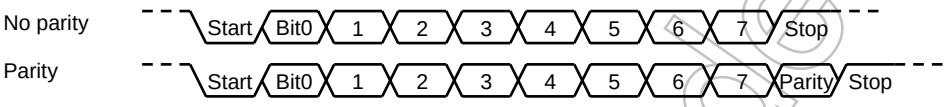
- Mode 0 (I/O interface mode)




- Mode 1 (7-bit UART mode)




- Mode 2 (8-bit UART mode)




- Mode 3 (9-bit UART mode)

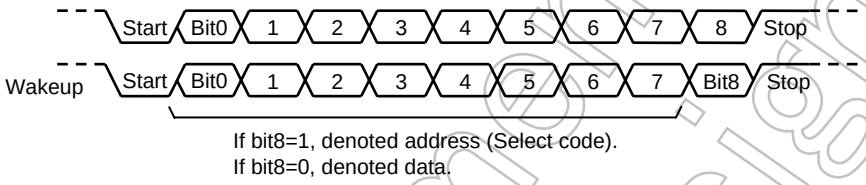



Figure 3.9.1 Data Format

### 3.9.1 Block Diagram

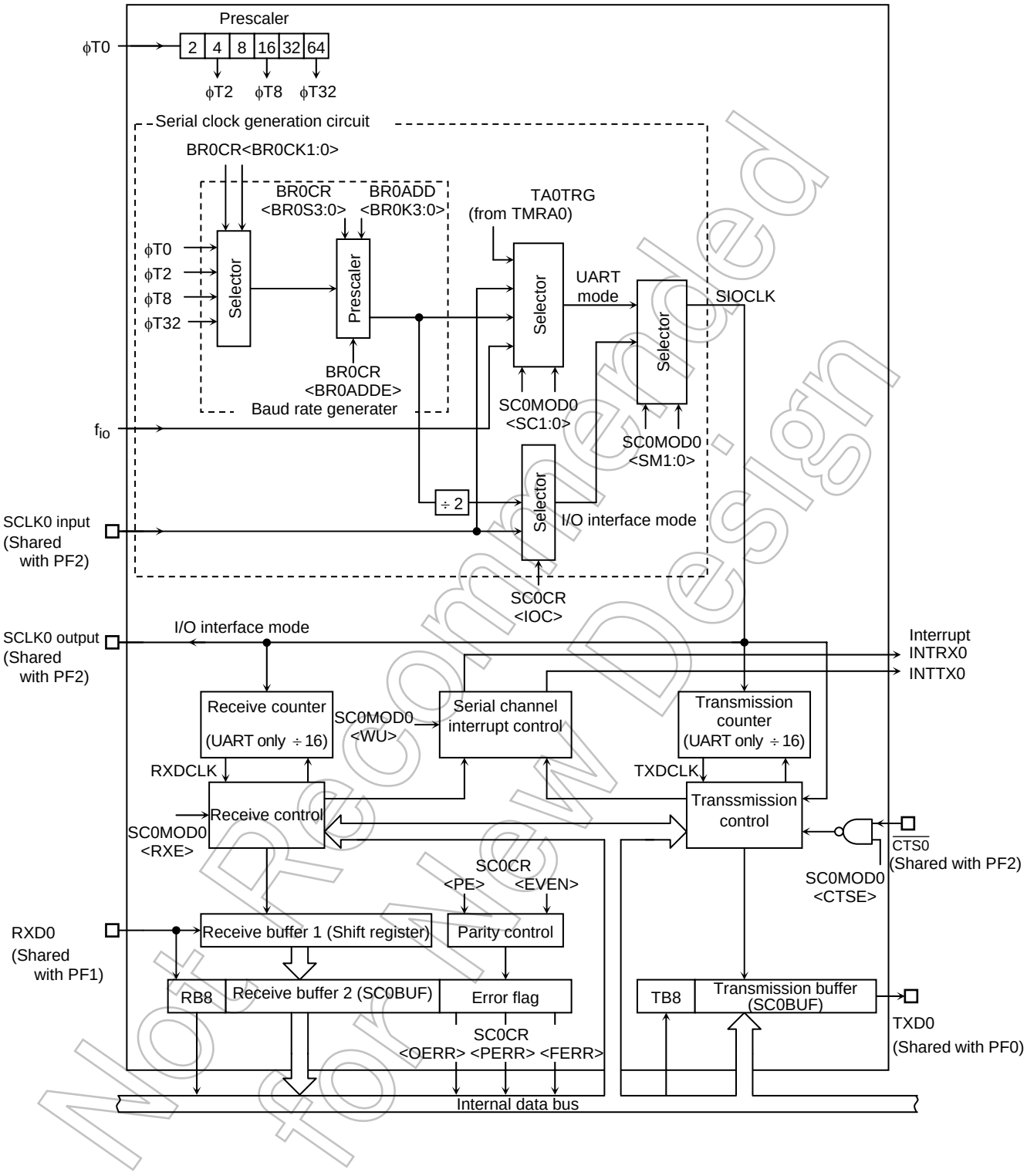





Figure 3.9.2 Block Diagram of SIO0

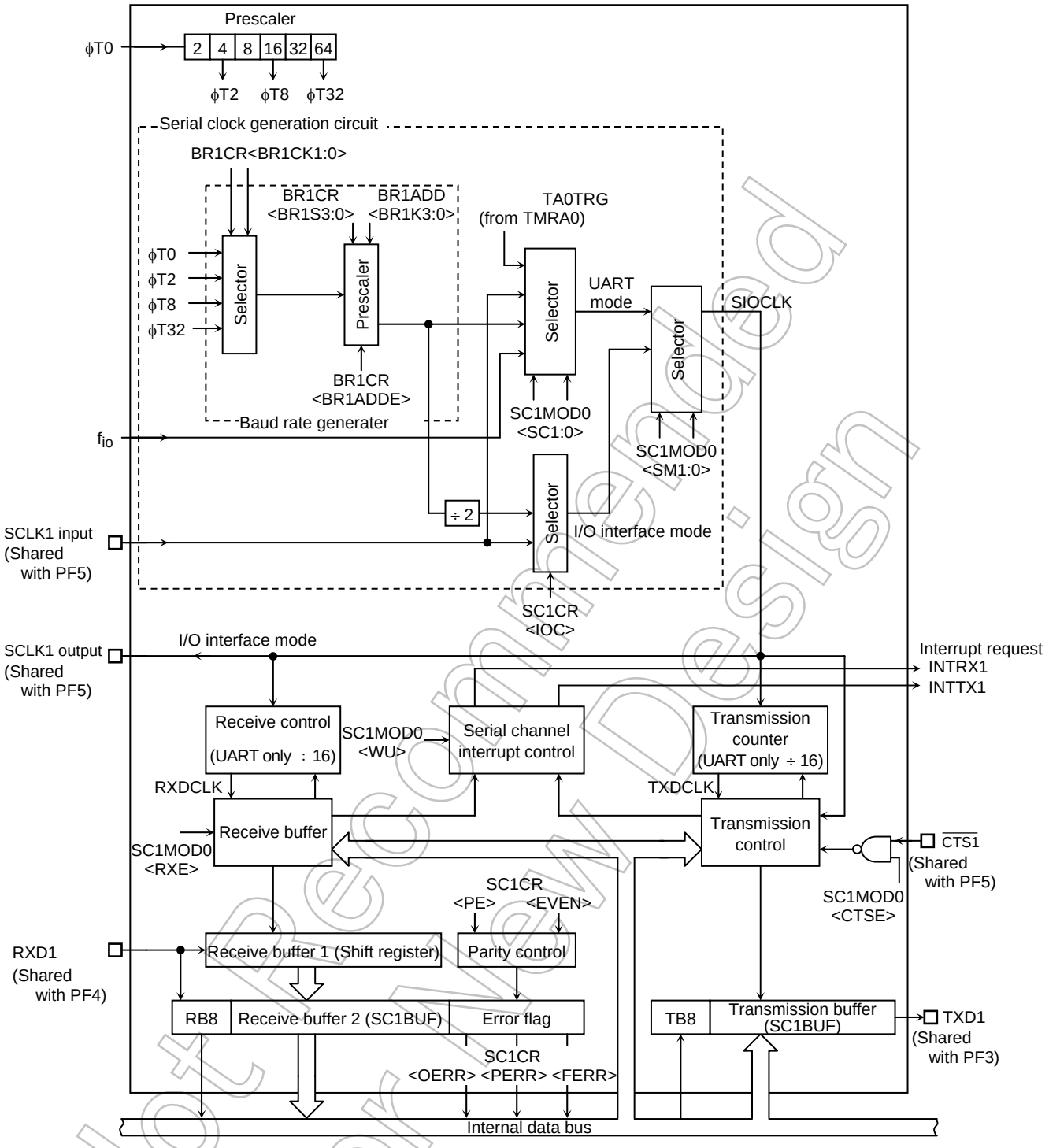





Figure 3.9.3 Block Diagram of SIO1

### 3.9.2 Operation of Each Circuit

(1) Prescaler

There is a 6-bit prescaler for generating a clock to SIO0. The clock selected using SYSCR1<GEAR2:0> is divided by 8 and input to the prescaler as ϕT0. The prescaler can be run only case of selecting the baud rate generator as the serial transfer clock.

Table 3.9.2 shows prescaler clock resolution into the baud rate generator.

Table 3.9.2 Prescaler Clock Resolution to Baud Rate Generator

| – | Clock Gear SYSCR1 <GEAR2:0> | – | Clock Resolution BR0CR<BR0CK1:0> | | | |
|---|---|---|---|---|---|---|
| | | | ϕT0 | ϕT2 | ϕT8 | ϕT32 |
| fc | 000(1/1) | 1/8 | fc/8 | fc/32 | fc/128 | fc/512 |
| | 001(1/2) | | fc/16 | fc/64 | fc/256 | fc/1024 |
| | 010(1/4) | | fc/32 | fc/128 | fc/512 | fc/2048 |
| | 011(1/8) | | fc/64 | fc/256 | fc/1024 | fc/4096 |
| | 100(1/16) | | fc/128 | fc/512 | fc/2048 | fc/8192 |

The serial interface 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 a circuit that generates transmission and receiving clocks that determine 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, determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3:0> and BR0ADD<BR0K3:0>.

- In UART mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N (N = 1, 2, 3 ... 16), which is set in BR0CR<BR0S3:0>.

(2) 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 (N = 2, 3 ... 15) set in BR0CR<BR0S3:0> and the value of K (K = 1, 2, 3 ... 15) set in BR0ADD<BR0K3:0>.

Note: If N = 1 and N = 16, the N + (16 – K)/16 division function is disabled. Clear BR0CR<BR0ADDE> register 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.

- UART mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

- 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 $f_C$ = 39.3216 MHz, the input clock frequency = $\phi$T2, 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 ($f_C$)

$$\text{Baud rate} = \frac{f_C/32}{8} \div 16$$

$$= 39.3216 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

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 $f_C$ = 31.9488 MHz, the input clock frequency = $\phi$T2, the frequency divider N (BR0CR<BR0S3:0>) = 6, K (BR0ADD<BR0K3:0>) = 8, and BR0CR<BR0ADDE> = 1, the baud rate is as follows:

  * Clock state

    [ Clock gear: 1/1 ($f_C$)

$$\text{Baud rate} = \frac{f_C/32}{6 + \frac{(16-8)}{16}} \div 16$$

$$= 31.9488 \times 10^6 \div 32 \div (6 + \frac{8}{16}) \div 16 = 9600 \text{ (bps)}$$

Table 3.9.3 show examples of UART mode transfer rates.

Additionally, the external clock input is available in the serial clock (Serial channels 0 and 1). 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 3.9.3 UART Baud Rate Selection

(when using baud rate generater and BR0CR<BR0ADDE> = 0)

Unit (kbps)

| $f_{SYS}$ [MHz] | Input Clock / Frequency Divider | φT0 ($f_{SYS}$/4) | φT2 ($f_{SYS}$/16) | φT8 ($f_{SYS}$/64) | φT32 ($f_{SYS}$/256) |
|---|---|---|---|---|---|
| 9.8304 | 2 | 76.800 | 19.200 | 4.800 | 1.200 |
| ↑ | 4 | 38.400 | 9.600 | 2.400 | 0.600 |
| ↑ | 8 | 19.200 | 4.800 | 1.200 | 0.300 |
| ↑ | 10 | 9.600 | 2.400 | 0.600 | 0.150 |
| 12.2880 | 5 | 38.400 | 9.600 | 2.400 | 0.600 |
| ↑ | A | 19.200 | 4.800 | 1.200 | 0.300 |
| 14.7456 | 2 | 115.200 | 28.800 | 7.200 | 1.800 |
| ↑ | 3 | 76.800 | 19.200 | 4.800 | 1.200 |
| ↑ | 6 | 38.400 | 9.600 | 2.400 | 0.600 |
| ↑ | C | 19.200 | 4.800 | 1.200 | 0.300 |
| 19.6608 | 1 | 307.200 | 76.800 | 19.200 | 4.800 |
| ↑ | 2 | 153.600 | 38.400 | 9.600 | 2.400 |
| ↑ | 4 | 76.800 | 19.200 | 4.800 | 1.200 |
| ↑ | 8 | 38.400 | 9.600 | 2.400 | 0.600 |
| ↑ | 10 | 19.200 | 4.800 | 1.200 | 0.300 |
| 22.1184 | 3 | 115.200 | 28.800 | 7.200 | 1.800 |
| 24.5760 | 1 | 384.000 | 96.000 | 24.000 | 6.000 |
| ↑ | 2 | 192.000 | 48.000 | 12.000 | 3.000 |
| ↑ | 4 | 96.000 | 24.000 | 6.000 | 1.500 |
| ↑ | 5 | 76.800 | 19.200 | 4.800 | 1.200 |
| ↑ | 8 | 48.000 | 12.000 | 3.000 | 0.750 |
| ↑ | A | 38.400 | 9.600 | 2.400 | 0.600 |
| ↑ | 10 | 24.000 | 6.000 | 1.500 | 0.375 |

Note 1: 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

Note: 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 clocks, the internal system clock $f_{IO}$, the trigger output signal from TMRA0 or the external clock (SCLK0 pin) is used to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART mode that 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 are taken to be 0.

(5) Receiving control

- In I/O interface mode

  In SCLK output mode with the setting SC0CR<IOC> = 0, the RXD0 pin 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 pin is sampled on the rising or falling edge of the SCLK input, according to the SC0CR<SCLKS> setting.

- In UART mode

  The receiving control block has a circuit that 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.

(7) Transmission counter

The transmission counter is a 4-bit binary counter that is 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.

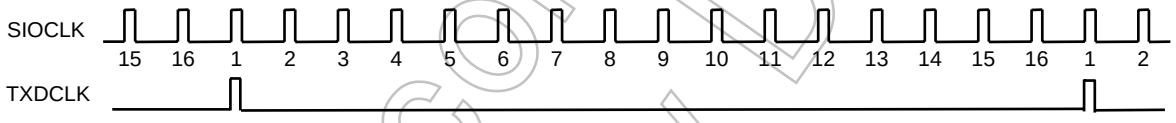



Figure 3.9.4 Generation of Transmission Clock

(8) 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 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 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, generating a transmission shift clock TXDSFT.

Handshake function

Use of $\overline{CTS0}$ pin allows data to be sent in units of one data format; thus, overrun errors can be avoided. The handshake function is enabled or disabled by the SC0MOD0<CTSE> setting.

When the $\overline{CTS0}$ pin condition is high level, after completed the current data transmission, data transmission is halted until the $\overline{CTS0}$ pin state is low again. However, the INTTX0 interrupt is generated, and it requests the next send from data to the CPU. The next data is written in the transmission buffer and data transmission 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 receive interrupt routine.

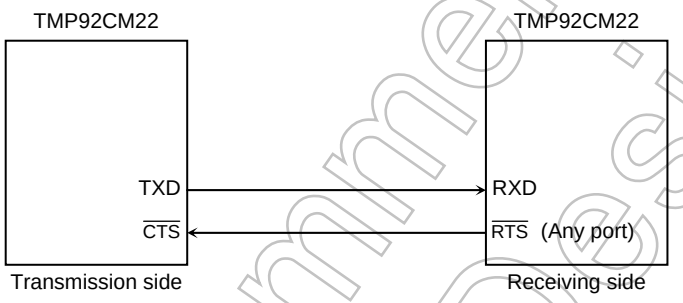
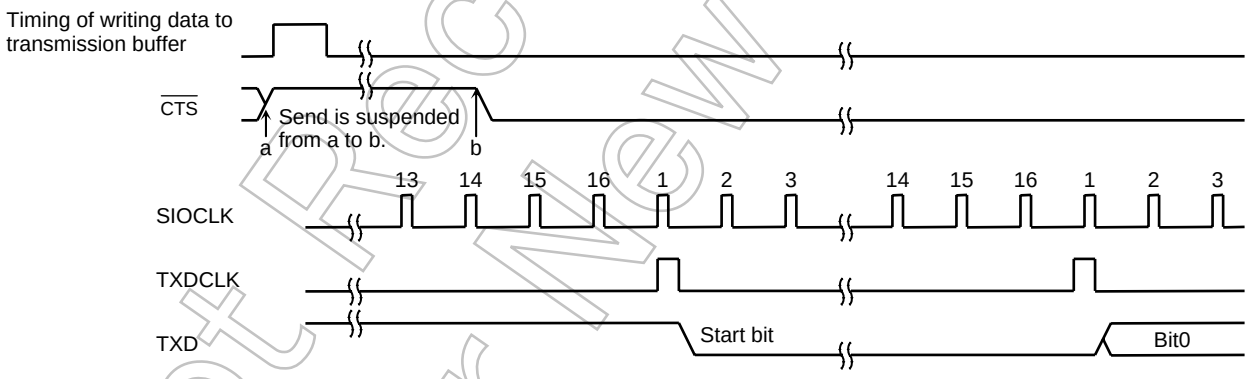



Figure 3.9.5 Handshake Function




Note 1: If the $\overline{CTS}$ signal goes high during transmission, will be stop next transmission data after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{CTS}$ signal has fallen.

Figure 3.9.6 $\overline{CTS}$ (Clear to send) Signal Timing

(9) Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU form the least significant bit in order. When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

(10) 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.

(11) 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
4) Set to disable receiving (Program "0" to SC0MOD0<RXE>)
5) Wait to terminate current frame
6) Read receiving buffer
7) Read error flag
8) Set to enable receiving (Program "1" to SC0MOD0<RXE>)
9) Request to transmit again
10) Other

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.
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.

(12) Timing generation

1. In UART mode

Receiving

| Mode | 9 Bits | 8 Bits + Parity | 8 Bits, 7 Bits + Parity, 7 Bits |
|---|---|---|---|
| Interrupt generation timing | Center of last bit (Bit8) | Center of last bit (Parity bit) | Center of stop bit |
| Framing error generation timing | Center of stop bit | Center of stop bit | Center of stop bit |
| Parity error generation timing | – | Center of last bit (Parity bit) | Center of stop bit |
| Overrun error generation timing | Center of last bit (Bit8) | Center of last bit (Parity bit) | Center of stop bit |

Note1: In 9 Bits mode and 8 Bits + Parity mode, 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.

Note2: The higher the transfer rate, the later than the middle receive interrupts and errors occur.

Transmission

| Mode | 9 Bits | 8 Bits + Parity | 8 Bits, 7 Bits + Parity, 7 Bits |
|---|---|---|---|
| Interrupt generation timing | Just before stop bit is transmitted | ← | ← |

2. In I/O interface mode

| | | |
|---|---|---|
| Transmission interrupt timing | SCLK output mode | Immediately after last bit data. (See Figure 3.9.19.) |
| | SCLK input mode | Immediately after rise of last SCLK signal rising mode, or immediately after fall in falling mode. (See Figure 3.9.20.) |
| Receiving interrupt timing | SCLK output mode | Timing used to transfer received to data receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 3.9.21.) |
| | SCLK input mode | Timing used to transfer received data to receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 3.9.22.) |

### 3.9.3 SFRs

SC0MOD0 (1202H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TB8 | CTSE | RXE | WU | SM1 | SM0 | SC1 | SC0 |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Transfer data bit8 | Handshake function control<br>0: CTS disable<br>1: CTS enable | Receive control<br>0: Receive disable<br>1: Receive enable | Wakeup function<br>0: Disable<br>1: Enable | Serial transmission mode<br>00: I/O interface mode<br>01: 7-bit UART mode<br>10: 8-bit UART mode<br>11: 9-bit UART mode | | Serial transmission clock (UART)<br>00: Timer A0 trigger<br>01: Baud rate generator<br>10: Internal clock $f_{IO}$<br>11: External clcok (SCLK0 input) | |

Serial transmission clock source (UART)

| | |
|---|---|
| 00 | TMRA0 trigger output 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 control register (SC0CR).

Serial transmission mode

| | | |
|---|---|---|
| 00 | I/O interface mode | |
| 01 | UART mode | 7-bit mode |
| 10 | | 8-bit mode |
| 11 | | 9-bit mode |

Wakeup function

| | 9-bit UART | Other modes |
|---|---|---|
| 0 | Interrupt generated when data is received | Don't care |
| 1 | Interrupt generated only when SC0CR<RB8> = 1 | |

Receiving function

| | |
|---|---|
| 0 | Receive disabled |
| 1 | Receive enabled |

Handshake function ( $\overline{CTS}$ pin)

| | |
|---|---|
| 0 | Disabled (Always transferable) |
| 1 | Enabled |

Transmission data bit8



Figure 3.9.7 Serial Mode Control Register 0 (for SIO0 and 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 | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Transfer data bit8 | Handshake function control<br>0: CTS disable<br>1: CTS enable | Receive control<br>0: Receive disable<br>1: Receive enable | Wakeup function<br>0: Disable<br>1: Enable | Serial transmission mode<br>00: I/O interface mode<br>01: 7-bit UART mode<br>10: 8-bit UART mode<br>11: 9-bit UART mode | | Serial transmission clock (UART)<br>00: Timer A0 trigger<br>01: Baud rate generator<br>10: Internal clock $f_{IO}$<br>11: External clcok (SCLK1 input) | |

Serial transmission clock source (UART)

| | |
|---|---|
| 00 | TMRA0 trigger output 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 control register (SC1CR).

Serial transmission mode

| | | |
|---|---|---|
| 00 | I/O interface mode | |
| 01 | UART mode | 7-bit mode |
| 10 | | 8-bit mode |
| 11 | | 9-bit mode |

Wakeup function

| | 9-bit UART | Other modes |
|---|---|---|
| 0 | Interrupt generated when data is received | Don't care |
| 1 | Interrupt generated only when SC1CR<RB8> = 1 | |

Receiving function

| | |
|---|---|
| 0 | Receive disabled |
| 1 | Receive enabled |

Handshake function ( $\overline{CTS}$ pin)

| | |
|---|---|
| 0 | Disabled (Always transferable) |
| 1 | Enabled |

Transmission data bit8



Figure 3.9.8 Serial Mode Control Register (for SIO1 and SC1MOD)

SC0CR (1201H)

| | 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 | |
| After reset | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Received data bit8 | Parity 0: Odd 1: Even | Parity addition 0: Disable 1: Enable | 1: Error Overrun | 1: Error Parity | 1: Error Framing | 0: SCLK0 (↑) 1: SCLK0 (↓) | 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 SCLK0 pin (I/O mode)

| | |
|---|---|
| 0 | Transmits and receivers data on rising edge of SCLK0. (↑) |
| 1 | Transmits and receivers 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 bit8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.9.9 Serial Control Register (for SIO0 and SC0CR)

SC1CR (1209H)

| | 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 | |
| After reset | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Received data bit8 | Parity 0: Odd 1: Even | Parity addition 0: Disable 1: Enable | 1: Error | | | 0: SCLK1 (↑) 1: SCLK1 (↓) | 0: Baud rate generator 1: SCLK1 pin input |
| | | | | Overrun | Parity | Framing | | |

I/O interface input clock selection

| | |
|---|---|
| 0 | Baud rate generator |
| 1 | SCLK1 pin input |

Edge selection for SCKL1 pin (I/O mode)

| | |
|---|---|
| 0 | Transmits and receives data on rising edge of SCLK1. |
| 1 | Transmits and receives data on falling edge of 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 bit8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.9.10 Serial Control Register (for SIO1 and SC1CR)

BR0CR (1203H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | BR0ADDE | BR0CK1 | BR0CK0 | BR0S3 | BR0S2 | BR0S1 | BR0S0 |
| Read/Write | R/W | | | | | | | |
| After reset | 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 divisions 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 | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | Sets frequency divisor "K" (Divided by N + (16 – K)/16). | | | |

Sets baud rate generator frequency divisor

| BR0CR <BR0S3:0> / BR0ADD <BR0K3:0> | BR0CR<BR0ADDE> = 1 | | BR0CR<BR0ADDE> = 0 |
|---|---|---|---|
| 0000 | 0000 (N = 16) or 0001 (N = 1) | 0010 (N = 2) ~ 1111 (N = 15) | 0001 (N = 1) (UART only) ~ 1111 (N = 15) 0000 (N = 16) |
| 0000 | Disable | Disable | Divided by N |
| 0001 (K = 1) ~ 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 + (16-K)/16 division function is used. Writes to unused bits in the BR0ADD register do not affect operation, and undefined data is read from these unused bits.

Figure 3.9.11 Baud Rate Generator Control (for SIO0, BR0CR, and BR0ADD)

BR1CR (120BH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | BR1ADDE | BR1CK1 | BR1CK0 | BR1S3 | BR1S2 | BR1S1 | BR1S0 |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Always write “0”. | + (16 – K)/16 division<br>0: Disable<br>1: Enable | 00: φT0<br>01: φT2<br>10: φT8<br>11: φT32 | | Divided frequency setting | | | |

+ (16 – K)/16 divisions enable

| | |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

Input clock selection for 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 | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | Set frequency divisor “K”<br>(Divided by N + (16 – K)/16). | | | |

Baud rate generator frequency divisor setting

| | BR1CR<BR1ADDE> = 1 | | BR1CR<BR1ADDE> = 0 |
|---|---|---|---|
| BR1CR <BR1S3:0><br>BR1ADD <BR1K3:0> | 0000 (N = 16)<br>or<br>0001 (N = 1) | 0010 (N = 2)<br>~<br>1111 (N = 15) | 0001 (N = 1) (UART only)<br>to<br>1111 (N = 15)<br>0000 (N = 16) |
| 0000 | Disable | Disable | Divided by N |
| 0001 (K = 1)<br>~<br>1111 (K = 15) | Disable | Divided by<br>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 BR1CR <BR1ADDE> to 1 after setting K (K = 1 to 15) to BR1ADD<BR1K3:0> when +(16-K)/16 division function is used. Writes to unused bits in the BR1ADD register do not affect operation, and undefined data is read from these unused bits.

Figure 3.9.12 Baud Rate Generater Control (for SIO1, BR1CR, and BR1ADD)

SC0BUF (1200H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| TB7 | TB6 | TB5 | TB4 | TB3 | TB2 | TB1 | TB0 | (for transmission) |
| RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | (for receiving) |

Note: Prohibit read-modify-write for SC0BUF

Figure 3.9.13 Serial Transmission/Receiving Buffer Register (for SIO0 and SC0BUF)

SC0MOD1 (1205H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | I2S0 | FDPX0 | | | | | | |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | IDLE2<br>0: Stop<br>1:Run | Duplex<br>0: Half<br>1: Full | | | | | | |

Figure 3.9.14 Serial Mode Control Regsiter (for SIO and SC0MOD1)

SC1BUF (1208H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| TB7 | TB6 | TB5 | TB4 | TB3 | TB2 | TB1 | TB0 | (for transmission) |
| RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | (for receiving) |

Note: Prohibit read-modify-write for SC1BUF

Figure 3.9.15 Serial Transmission/Receiving Buffer Register (for SIO1 and SC1BUF)

SC1MOD1 (120DH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | I2S1 | FDPX1 | | | | | | |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | IDLE2<br>0: Stop<br>1: Run | Duplex<br>0: Half<br>1: Full | | | | | | |

Figure 3.9.16 Serial Mode Control Regsiter (for SIO and SC1MOD1)

### 3.9.4 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.

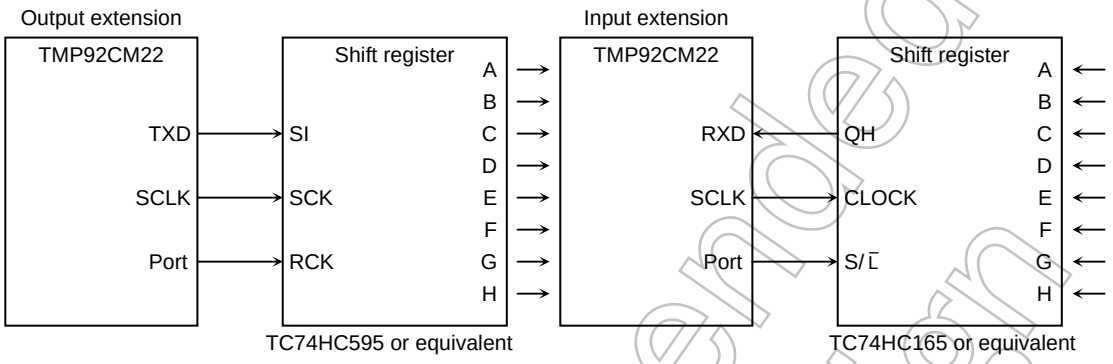



Figure 3.9.17 Example of SCLK Output Mode Connection

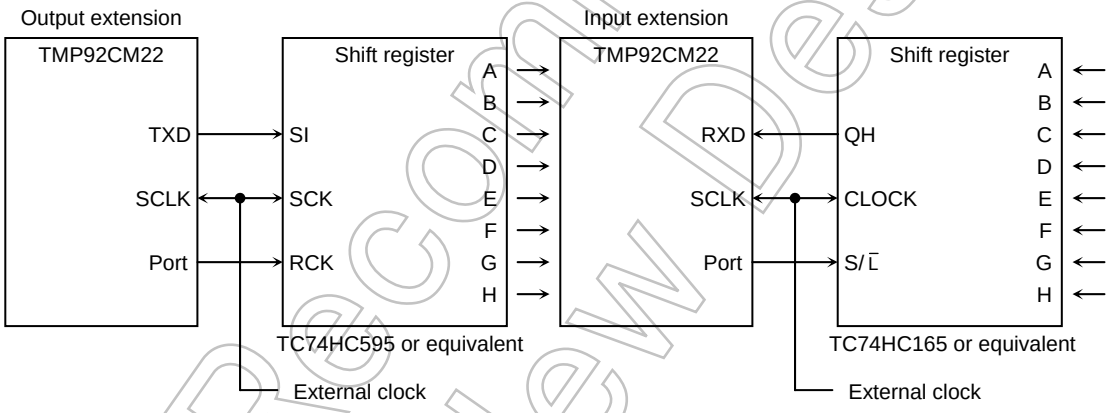



Figure 3.9.18 Example of SCLK Output Mode Connection

1. 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 outputted, INTES0<ITX0C> will be set to generate the INTTX0 interrupt.

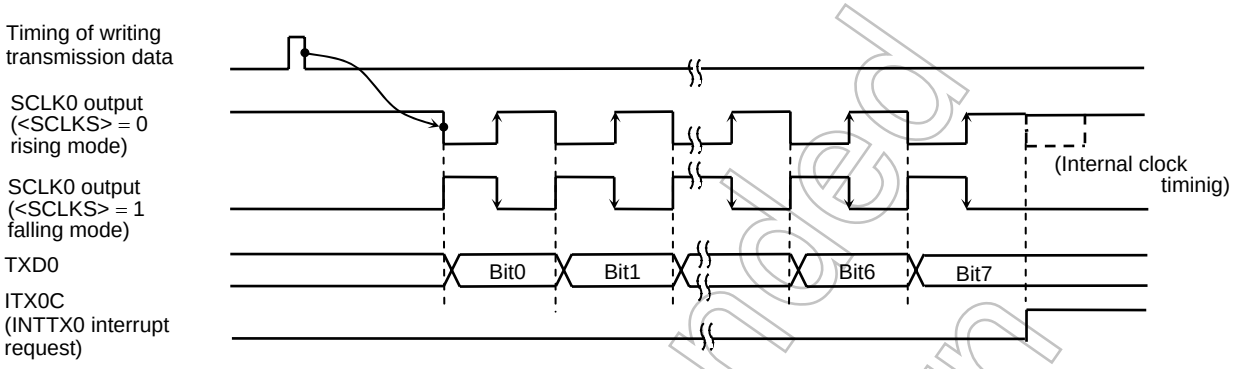



Figure 3.9.19 Transmission Operation in I/O Interface Mode (SCLK0 output mode)

In SCLK input mode, 8-bit data is output from 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 outputted, INTES0<ITX0C> will be set to generate INTTX0 interrupt.

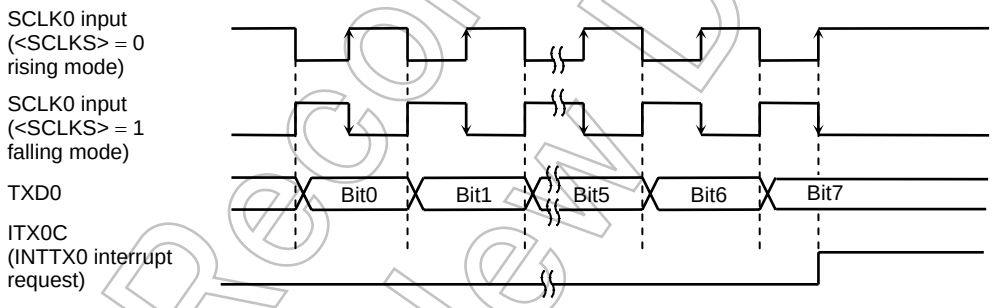



Figure 3.9.20 Transmission Operation in I/O Interface Mode (SCLK0 input mode)

2. Receiving

In SCLK output mode, the synchronous clock is outputted from SCLK0 pin and the data is shifted to receiving buffer 1. This starts when the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data are received, the data will be transferred to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set to generate INTRX0 interrupt.

The outputting for the first SCLK0 starts by setting SC0MOD0<RXE> to 1.

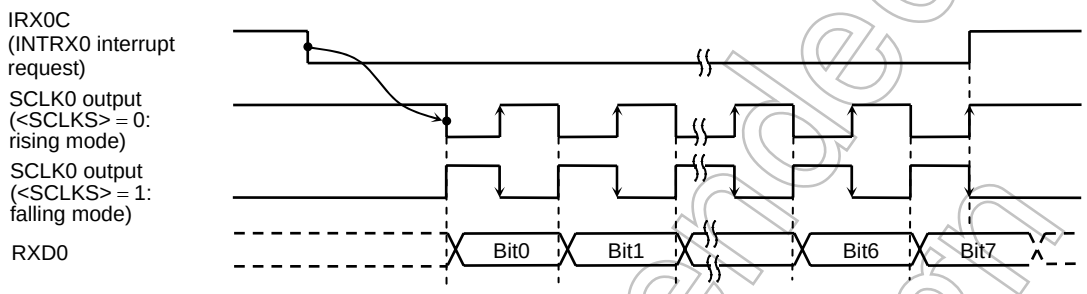



Figure 3.9.21 Receiving Operation in I/O Interface Mode (SCLK0 output mode)

In SCLK input mode, the data is shifted to receiving buffer 1 when the SCLK input becomes active after the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data is received, the data will be shifted to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set again to be generate INTRX0 interrupt.

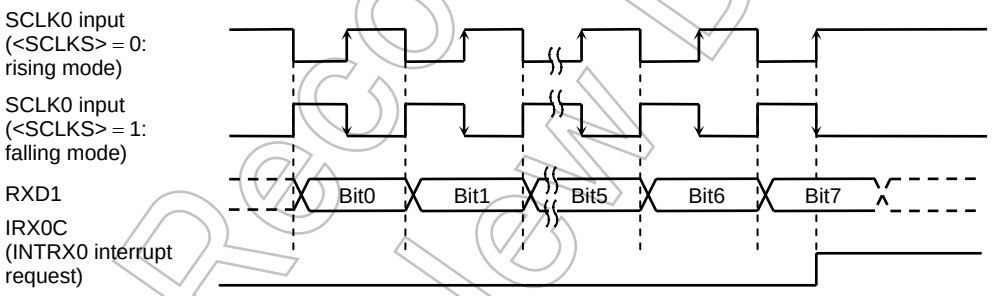



Figure 3.9.22 Receiving Operation in I/O Interface Mode (SCLK0 input mode)

Note: If receiving, set to the receive enable state (SC0MOD0<RXE> = 1) in both SCLK input mode and output mode.

3. Transmission and receiving (Full duplex mode)

When the full duplex mode is used, set the level of receive interrupt to "0" and set enable the interrupt level (1 to 6) to the transfer interrupts. In the transfer interrupt program, the receiving operation should be done like the below example before setting the next transfer data.

Example: Channel 0, SCLK output

Baud rate = 9600 bps

$f_C$ = 4.9152 MHz

* Clock state: Clock gear 1/1(fc)

Main routine

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| INTES0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | Set transmission interrupt level to 1, and disable receiving interrupt level to 0. |
| PFCR | – | – | – | – | – | 1 | 0 | 1 | Set to PF0 (TXD0), PF1 (RXD0), and PF2 (SCLK0). |
| PFFC | – | – | – | – | – | 1 | – | 1 | |
| SC0MOD0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Set to I/O interface mode. |
| SC0MOD1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Set to full duplex mode. |
| SC0CR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Output SCLK, select rising edge. |
| BR0CR | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Set to 9600 bps. |
| SC0MOD0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Set receive to enable. |
| SC0BUF | * | * | * | * | * | * | * | * | Set transmission data. |

Transmission interrupt routine

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Acc SC0BUF | | | | | | | | | Read receiving data. |
| SC0BUF | * | * | * | * | * | * | * | * | Set transmission data. |

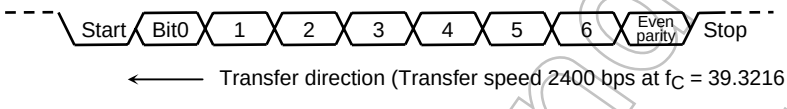
X: Don't care, –: No change

(2) Mode 1 (7-bit UART mode)

7-bit UART mode is selected by setting serial channel mode register SC0MOD0<SM1:0> 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).

Example: When transmitting data of the following format, the control registers should be set as described below. This explanation applies to channel 0.




← Transfer direction (Transfer speed 2400 bps at $f_C$ = 39.3216 MHz)

* Clock state: Clock gear 1/1(fc)

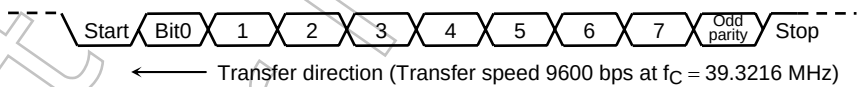
| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| PFCR | ← | – | – | – | – | – | – | – | 1 | Set PF0 to as TXD0 pin. |
| PFFC | ← | – | – | – | – | – | – | – | 1 | |
| SC0MOD | ← | X | 0 | – | X | 0 | 1 | 0 | 1 | Set to 7-bit UART mode. |
| SC0CR | ← | X | 1 | 1 | X | X | X | 0 | 0 | Add even parity. |
| BR0CR | ← | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | Set to 2400 bps. |
| INTES0 | ← | 1 | 1 | 0 | 0 | – | – | – | – | Set INTTX0 interrupt to enable, set to level 4. |
| SC0BUF | ← | * | * | * | * | * | * | * | * | Set transmission data. |

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).

Example: When receiving data of the following format, the control registers should be set as described below.




← Transfer direction (Transfer speed 9600 bps at $f_C$ = 39.3216 MHz)

* Clock state: Clock gear 1/1(fc)

Main routine

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| PFCR | ← | – | – | – | – | – | – | 0 | – | Set PF1 (RXD0) to input pin. |
| SC0MOD | ← | – | 0 | 1 | X | 1 | 0 | 0 | 1 | Set to 8-bit UART mode, set receives to enable. |
| SC0CR | ← | X | 0 | 1 | X | X | X | 0 | 0 | Add odd parity. |
| BR0CR | ← | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | Set to 9600 bps. |
| INTES0 | ← | – | – | – | – | 1 | 1 | 0 | 0 | Set INTTX0 interrupt to enable, set to level 4. |

Interrupt routine processing

| | | | |
|---|---|---|---|
| Acc | ← | SC0CR AND 00011100 | Check for error. |
| if Acc | ≠ | 0 then ERROR | |
| Acc | ← | SC0BUF | Read receiving data. |

X : Don't care, – : No change

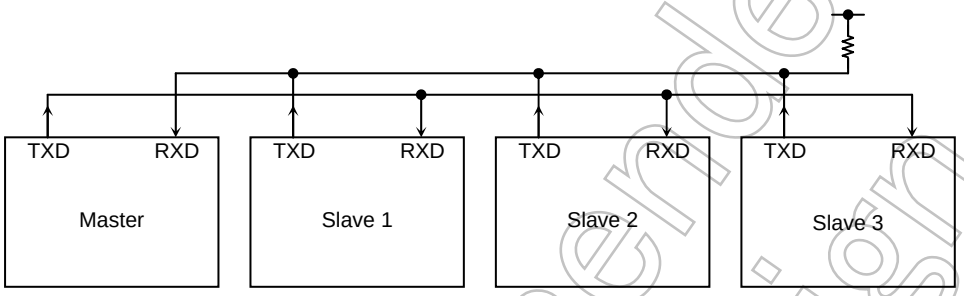
(4) Mode 3 (9-bit UART mode)

9-bit UART mode is selected by setting SC0MOD0<SM1:0> to 11. In this mode parity bit cannot be added.

In the case of transmission the MSB (9th bit) is programmed to SC0MOD0<TB8>. In the case of receiving it is stored in SC0CR<RB8>. When the buffer is written and read, the MSB is read or written first, before the rest of the SC0BUF data.

Wakeup function

In 9-bit UART mode, the wakeup function for slave controllers is enabled by setting SC0MOD0<WU> to 1. The interrupt INTRX0 occurs only when <RB8> = 1.

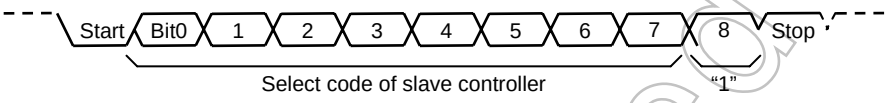



Note: The TXD pin of each slave controller must be in open-drain output mode.

Figure 3.9.23 Serial Link Using Wakeup Function

Protocol

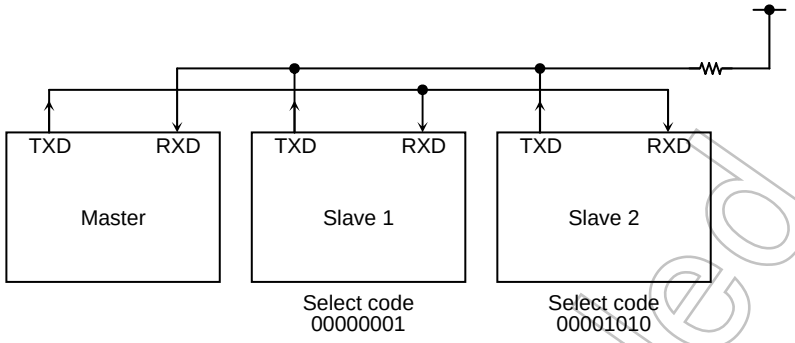
1. Select 9-bit UART mode on the master and slave controllers.
2. Set the SC0MOD0<WU> bit on each slave controller to 1 to enable data receiving.
3. The master controller transmits one-frame data including the 8-bit select code for the slave controllers. The MSB (Bit8) <TB8> is set to "1".




4. Each slave controller receives the above frame. If it matches with own select code, clears <WU> bit to "0".
5. The master controller transmits data to the specified slave controller whose SC0MOD0<WU> bit is cleared to "0". The MSB (Bit8) <TB8> is cleared to "0".


6. The other slave controllers (whose <WU> bits remain at 1) ignore the received data because their MSB (Bit8 or <RB8>) are set to "0", disabling INTRX0 interrupts. The slave controller (<WU> bit = "0") can transmit data to the master controller, and it is possible to indicate the end of data receiving to the master controller by this transmission.

Example: To link two slave controllers serially with the master controller using the system clock $f_{IO}$ as the transfer clock.




- Master controller setting

```
Main routine
PFCR      ← – – – – – – 0 1   }  Set PF0 to TXD0, and set PF1 to RXD0 pin.
PFFC      ← – – – – – – X 1   }
INTES0    ← 1 1 0 0 1 1 0 1      Set INTTX0 to enable, and set interrupt level to level 4.
                                 Set INTRX0 to enable, and set interrupt level to level 5.
SC0MOD0   ← 1 0 1 0 1 1 1 0      Set to 9-bit UART mode, and set transfer clock to fIO.
SC0BUF    ← 0 0 0 0 0 0 0 1      Set select code of slave 1.

Interrupt routine (INTTX0)
SC0MOD0   ← 0 – – – – – – –      Set TB8 to "0".
SC0BUF    ← * * * * * * * *      Set transmission data.
```

- Slave setting

```
Main routine
PFCR      ← – – – – – – 0 0   }  Set PF0 to TXD (open-drain output), and PC1 to RXD.
PFFC      ← – – – – – – X 1   }

INTES0    ← 1 1 0 1 1 1 1 0      Set INTTX0 and INTRX0 to enable.
SC0MOD0   ← 0 0 1 1 1 1 1 0      Set to <WU> = "1" in 9-bit UART mode transfer clock fIO.

Interrupt routine (INTRX0)

Acc ← SC0BUF
if  Acc = Select code
Then      ← – – – 0 – – – –      Clear to <WU> = "0".
SC0MOD0
```

### 3.9.5 Support for IrDA Mode

SIO0 includes support for the IrDA 1.0 infrared data communication specification. Figure 3.9.24 shows the block diagram.

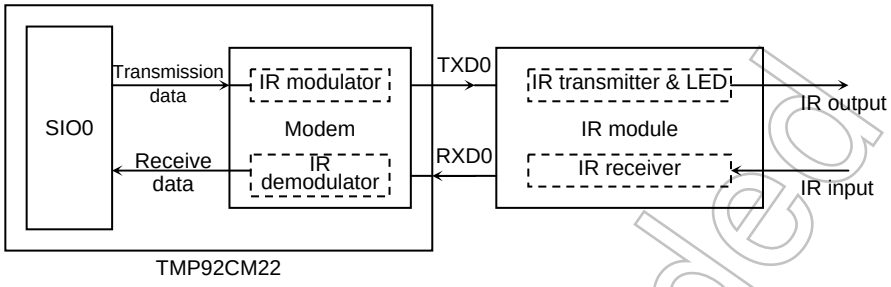



Figure 3.9.24 Block Diagram of IrDA

(1) Modulation of transmission data

When the transmission data is 0, output "H" level with either 3/16 or 1/16 times for width of baud-rate (Selectable in software). When data is "1", modem output "L" level.

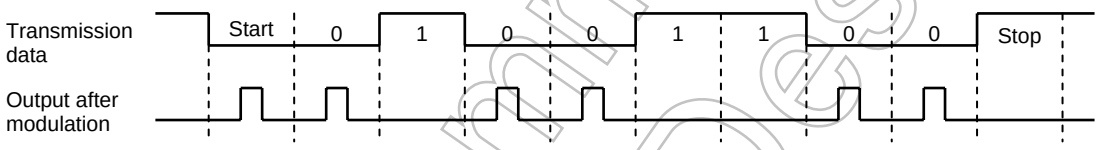



Figure 3.9.25 Example of Modulation of Transmission Data

(2) Modulation of receiving data

When the receive data has the effective high level pulse width (Software selectable), the modem outputs "0" to SIO0. Otherwise modem outputs "1" to SIO0. Receive pulse logic is selectable by SIRCR<RXSEL>.

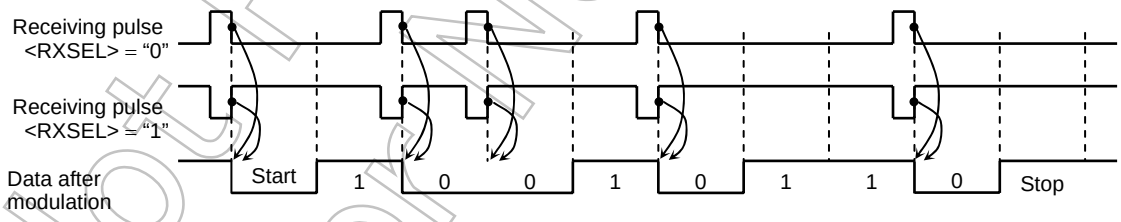



Figure 3.9.26 Example of Modulation of Receiving Data

(3) Data format

Format of transmission/receiving must set to data length 8-bit, without parity bit, 1 bit of stop bit.

Any other settings don't guarantee the normal operation.

(4) SFR

Figure 3.9.27 shows the control register SIRCR. If change setting this register, must set it after set operation of transmission/receiving to disable (Both <TXEN> and <RXEN> of this register should be clear to 0).

Any changing for this register during transmission or receiving operation doesn't guarantee the normal operation.

The following example describes how to set this register:

1) SIO setting ; Set SIO side.
↓
2) LD (SIRCR), 07H ; Set receiving effect pulse width to 16X.
3) LD (SIRCR), 37H ; TXEN, RXEN enable the transmission and receiving.
↓
4) Transmission/receiving ; The modem operates as follows:
   - SIO0 starts transmitting.
   - IR receiver starts receiving.

(5) Notes

1. Making baud rate when using IrDA

   In baud rate during using IrDA, must set "01" to SC0MOD0<SC1:0> in SIO by using baud rate generator.

   TA0TRG, $f_{IO}$, SCLK0 input of except for it can not using.

2. Output pulse width and baud rate generator during transmission IrDA

   As the IrDA 1.0 physical layer specification, the data transfer speed and infra-red pulse width is specified.

Table 3.9.4 Specification of Transfer Rate and Pulse Width

| Transfer Rate | Modulation | Transfer Rate Tolerance (% of Rate) | Minimum of Pulse Width | Typical of Pulse Width 3/16 | Maximum of Pulse Width |
|---|---|---|---|---|---|
| 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 T × 3/16 pulse width when baud rate is 115.2 kbps).

The TMP92CM22 has function which is selectable the transmission pulse width either 3/16 or 1/16. But T × 1/16 pulse width can be selected when the baud rate is equal or less than 38.4 kbps only. When 57.6 kbps and 115.2 kbps, the output pulse width should not be set to T × 1/16.

As the same reason, + (16 – K)/16 division function in the baud rate generator of SIO0 cannot be used to generate 115.2 kbps baud rate.

Also when the 38.4 kbps and 1/16 pulse width, + (16 – K)/16 division function cannot be used.

Table 3.9.5 shows baud rate and pulse width for (16 – K)/16 division function.

Table 3.9.5 Baud Rate and Pulse Width for (16 – K)/16 Division Function

| Output 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 | × | ○ | ○ | ○ | ○ | ○ |
| T × 1/16 | – | – | × | ○ | ○ | ○ |

○: Can be used (16 – K)/16 division function.
×: Cannot be used (16 – K)/16 division function.
–: Cannot be set to T × 1/16 pulse width.

SIRCR (1207H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PLSEL | RXSEL | TXEN | RXEN | SIRWD3 | SIRWD2 | SIRWD1 | SIRWD0 |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Selection transmission pulse width<br>0: 3/16<br>1: 1/16 | Receiving data logic<br>0: "H" pulse<br>1: "L" pulse | Transmission data<br>0: Disable<br>1: Enable | Receiving operation<br>0: Disable<br>1: Enable | Select receiving effective pulse width<br>Set effective pulse width for equal or more than 2x × (Value + 1) + 100 ns<br>Can be set: 1 to 14<br>Cannot be set: 0, 15 | | | |

Select receiving effective pulse width
Formula:
Receiving effective pulse width ≥ 2x × (Setting value + 1) + 100 ns x = $1/f_{SYS}$

| | |
|---|---|
| 0000 | Cannot be set. |
| 0001 | Pulse width of equal or more than 4x + 100 ns is effective. |
| ↓ | |
| 1110 | Pulse width of equal or more than 30x + 100 ns is effective. |
| 1111 | Cannot be set. |

Enable of receiving operation

| | |
|---|---|
| 0 | Disable receiving operation.<br>(Received input is ignored.) |
| 1 | Enable receiving operation. |

Enable of transmission operation

| | |
|---|---|
| 0 | Disable transmission operation<br>(Input from SIO is ignored.) |
| 1 | Enable transmission operation. |

Select transmission pulse width

| | |
|---|---|
| 0 | Pulse width of 3/16 |
| 1 | Pulse width of 1/16 |

Note: If a pulse width complying with the IrDA 1.0 standard (1.6μs min.) can be guaranteed with a low baud rate, setting this bit to "1" will result in reduced power dissipation.



Figure 3.9.27 IrDA Control Register

## 3.10 Serial Bus Interface (SBI)

The TMP92CM22 has a 1-channel serial bus interface. Serial bus interface (SBI0) include following 2 operation modes.

- I²C bus mode (Multi master)
- Clocked-synchronous 8-bit SIO mode

The serial bus interface is connected to an external device through P91 (SDA) and P92 (SCL) in the I²C bus mode; and through P90 (SCK), P91 (SO), and P92 (SI) in the clocked-synchronous 8-bit SIO mode.

Each pin is specified as follows.

| | P9ODE<br><P92ODE, P91ODE> | P9CR<br><P92C, P91C, P90C> | P9FC<br><P92F, P91F, P90F> |
|---|---|---|---|
| I²C bus mode | 11 | 11X | 11X |
| Clocked-synchronous 8-bit SIO mode | XX | 011<br>010 | 011<br>010 (Note) |

X: Don't care

Note: When using SI and SCK input function, set P9FC<P92F,P90F> to "0" (Function setting).

### 3.10.1 Configuration

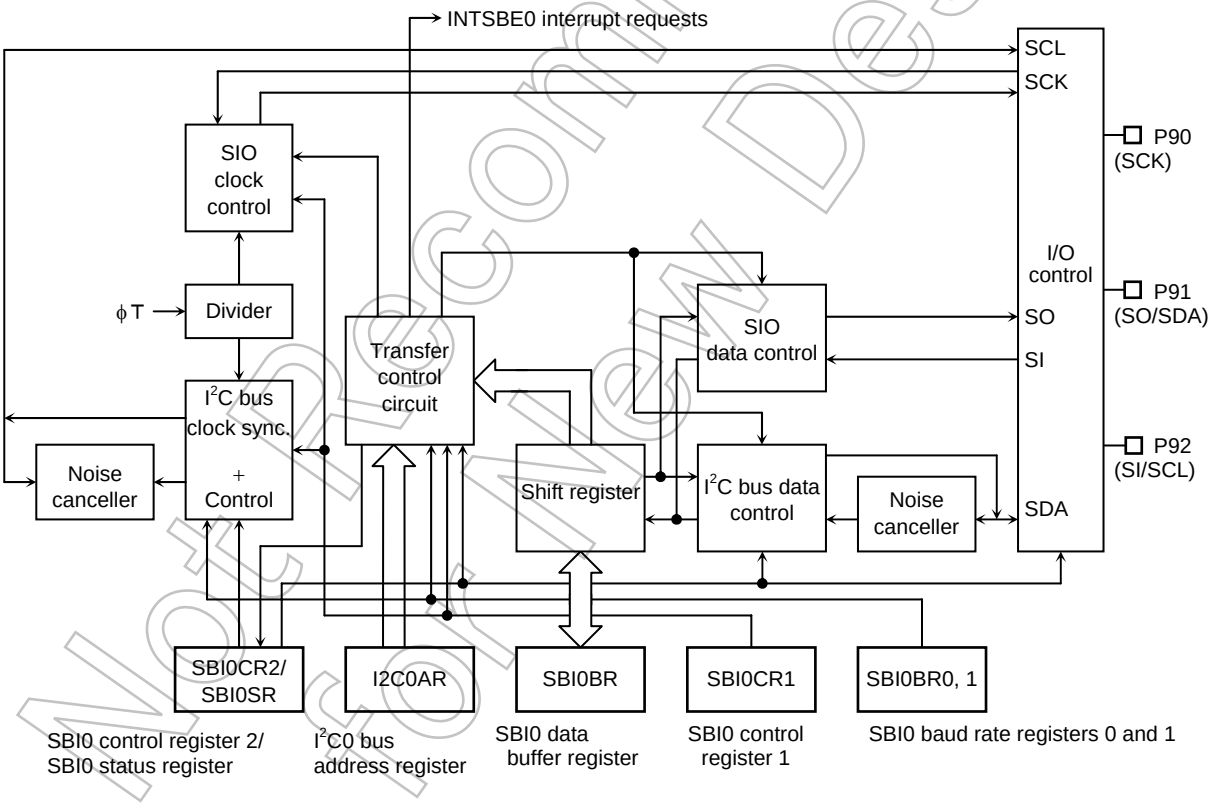



Figure 3.10.1 Serial Bus Interface 0 (SBI0)

### 3.10.2 Control

The following registers are used to control the serial bus interface and monitor the operation status.

- Serial bus interface 0 control register 1 (SBI0CR1)
- Serial bus interface 0 control register 2 (SBI0CR2)
- Serial bus interface 0 data buffer register (SBI0DBR)
- I²C bus 0 address register (I2C0AR)
- Serial bus interface 0 status register (SBI0SR)
- Serial bus interface 0 baud rate register 0 (SBI0BR0)
- Serial bus interface 0 baud rate register 1 (SBI0BR1)

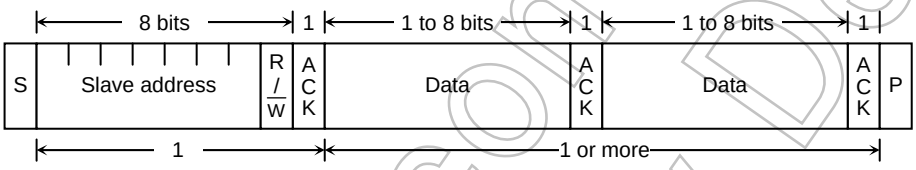
The above registers differ depending on a mode to be used.

Refer to Section 3.10.4 "I²C Bus Mode Control Register" and 3.10.7 "Clocked-synchronous 8-Bit SIO Mode Control".

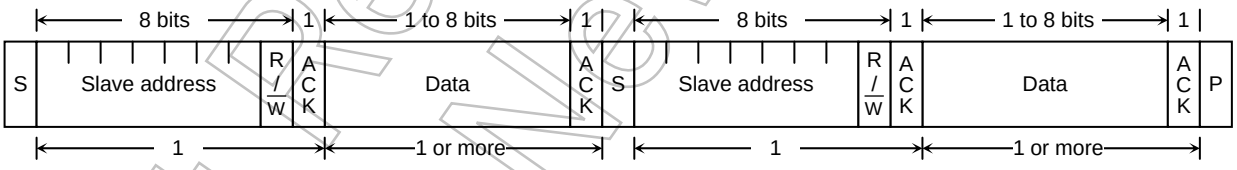
### 3.10.3 Data Format in I²C Bus Mode

Data format in I²C bus mode is shown Figure 3.10.3.

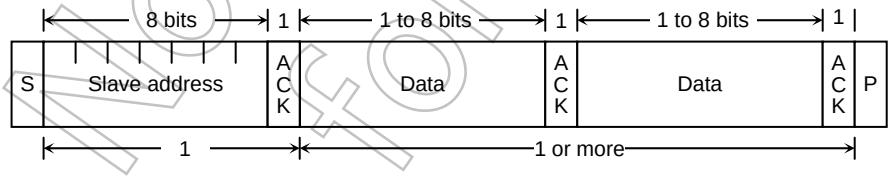
(a) Addressing format




(b) Addressing format (with restart)




(c) Free data format (transfer format transfer from master device to slave device)




S: Start condition

R/$\overline{W}$ : Direction bit

ACK: Acknowledge bit

P: Stop condition

Figure 3.10.2 Data Format in I²C Bus Mode

### 3.10.4 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 1

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| SBI0CR1 (1240H) | Bit symbol | BC2 | BC1 | BC0 | ACK | | SCK2 | SCK1 | SCK0/ SWRMON |
| | Read/Write | W | | | R/W | | W | | R/W |
| | After reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0/1 (Note 3) |
| Read-modify-write instruction is prohibited. | Function | Select number of transferred bits (Note 1) | | | Acknowledge mode specification 0: Not generate 1: Generate | | Internal serial clock selection and software reset monitor (Note 2) | | |

Internal serial clock selection <SCK2:0> at write

| | | |
|---|---|---|
| 000 | n = 5 | – kHz (Note4) |
| 001 | n = 6 | – kHz (Note4) |
| 010 | n = 7 | – kHz (Note4) |
| 011 | n = 8 | 75.8 kHz |
| 100 | n = 9 | 38.5 kHz |
| 101 | n = 10 | 19.4 kHz |
| 110 | n = 11 | 9.73 kHz |
| 111 | (Reserved) | (Reserved) |

System clock: $f_{SYS}$

$f_{SYS}$ = 20 MHz (output to SCL pin)

Frequency = $\frac{f_{SYS}}{2^{n+8}}$ [Hz]

Software reset state monitor <SWRMON> at read

| | |
|---|---|
| 0 | During software reset |
| 1 | Initial data |

Acknowledge mode selection

| | |
|---|---|
| 0 | Not generate clock pulse for acknowledge signal |
| 1 | Generate clock for acknowledge signal |

Select number of bits transferred

| <BC2:0> | <ACK> = 0 | | <ACK> = 1 | |
|---|---|---|---|---|
| | Number of clock pulses | Data length | Number of clock pulses | Data length |
| 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 |

Note 1: Set the <BC2:0> to "000" before switching to a clocked-synchronous 8-bit SIO mode.

Note 2: For the frequency of the SCL line clock, see section 3.10.5 (3) "Serial clock".

Note 3: Initial data of SCK0 is "0", SWRMON is "1".

Note 4: This I²C bus circuit does not support Fast mode, it supports standard mode only. Although the I²C bus circuit itself allows the setting of a baud rate over 100 kbps, the compliance with the I²C specification is not guaranteed in that case.

Figure 3.10.3 Register for I²C Bus Mode

Serial Bus Interface Control Register 2

SBI0CR2 (1243H)

Read-modify-write instruction is prohibited.

| | 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) | |
| After reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Function | Master/ slave selection | Transmitter/ receiver selection | Start/stop condition generation | Release INTSBE0 interrupt request | Serial bus interface operation mode selection (Note 2)<br>00: Port mode<br>01: SIO mode<br>10: I²C bus mode<br>11: (Reserved) | | Software reset control write “10” and “01” in order, then an internal software reset signal is generated. | |

Serial bus interface operating mode selection (Note 2)

| | |
|---|---|
| 00 | Port mode (Serial bus interface output disabled) |
| 01 | Clocked-synchronous 8-bit SIO mode |
| 10 | I²C bus mode |
| 11 | (Reserved) |

INTSBE0 interrupt request

| | |
|---|---|
| 0 | – |
| 1 | Release interrupt request |

Start/stop condition generation

| | |
|---|---|
| 0 | Generates the stop condition |
| 1 | Generates the start condition |

Transmitter/receiver selection

| | |
|---|---|
| 0 | Receiver |
| 1 | Transmitter |

Master/slave selection

| | |
|---|---|
| 0 | Slave |
| 1 | Master |

Note 1: Reading this register function as SBI0SR register.

Note 2: Switch a mode to port mode after confirming that the bus is free.
Switch a mode between I²C bus mode and clocked-synchronous 8-bit SIO mode after confirming that input signals via port are high level.

Figure 3.10.4 Register for I²C Bus Mode

Serial Bus Interface Status Register

SBI0SR (1243H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | MST | TRX | BB | PIN | AL | AAS | AD0 | LRB |
| Read/Write | R | | | | | | | |
| After reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Function | Master/ slave status selection monitor | Transmitter/ receiver status selection monitor | $I^2C$ bus status monitor | INTSBE0 interrupt request monitor | 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 | Undetected |
| 1 | Slave address match or GENERAL CALL detected |

Arbitration lost detection monitor

| | |
|---|---|
| 0 | – |
| 1 | Arbitration lost |

INTSBE0 interrupt request monitor

| | |
|---|---|
| 0 | Interrupt requested |
| 1 | Interrupt released |

$I^2C$ bus status monitor

| | |
|---|---|
| 0 | Free |
| 1 | Busy |

Transmitter/receiver status monitor

| | |
|---|---|
| 0 | Receiver |
| 1 | Transmitter |

Master/slave status monitor

| | |
|---|---|
| 0 | Slave |
| 1 | Master |

Note: Writing in this register functions as SBI0CR2.



Figure 3.10.5 Register for $I^2C$ Bus Mode

Serial Bus Interface Baud Rate Register 0

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | I2SBI0 | | | | | | |
| Read/Write | W | R/W | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | Always write "0". | IDLE2<br>0: Stop<br>1: Run | | | | | | |

SBI0BR0 (1244H)

Read-modify-write instruction is prohibited.

Operation during IDLE 2 mode

| | |
|---|---|
| 0 | Stop |
| 1 | Run |

Serial Bus Interface Baud Rate Register 1

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P4EN | – | | | | | | |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | Internal clock<br>0: Stop<br>1: Run | Always write "0". | | | | | | |

SBI0BR1 (1245H)

Read-modify-write instruction is prohibited.

Operation during IDLE 2 mode

| | |
|---|---|
| 0 | Stop |
| 1 | Run |

Sirial Bus Interface Data Buffer Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| Read/Write | R (Receiving)/W (Transmission) | | | | | | | |
| After reset | Undefined | | | | | | | |

SBI0DBR (1241H)

Read-modify-write instruction is prohibited.

Note 1: When writing transmission data, start from the MSB (Bit7). Receiving data is placed from LSB (Bit0).

Note 2: SBI0DBR can't be read the written data. Therefore read-modify-write instruction (e.g., "BIT" instruction) is prohibited.

$I^2C$ Bus Address Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | SA6 | SA5 | SA4 | SA3 | SA2 | SA1 | SA0 | ALS |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Slave address selection for when device is operating as slave device | | | | | | | Address recognition mode specification |

I2C0AR (1242H)

Read-modify-write instruction is prohibited.

Address recognition mode specification

| | |
|---|---|
| 0 | Slave address recognition |
| 1 | Non slave address recognition |

Figure 3.10.6 Register for $I^2C$ Bus Mode

### 3.10.5 Control in I²C Bus Mode

(1) Acknowledge mode specification

Set the SBI0CR1<ACK> to 1 for operation in the acknowledge mode. The TMP92CM22 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 TMP92CM22 does not generate a clock pulse for the acknowledge signal when operating in the master mode.

(2) Select number of transfer bits

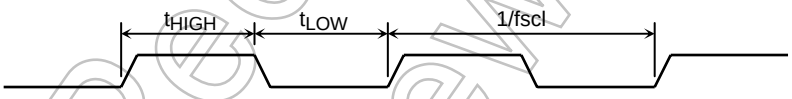
The SBI0CR1<BC2:0> is used to select a number of bits for next transmission/ receiving data.

Since the <BC2:0> is cleared to 000 as a start condition, a slave address and direction bit are transferred in 8 bits. Other than these, the <BC2:0> retains a specified value.

(3) Serial clock

1. Clock source

The SBI0CR1<SCK2:0> is used to select a maximum transfer frequency outputted on the SCL pin in master mode. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I²C bus, such as the smallest pulse width of $t_{LOW}$.




$t_{LOW} = 2^{n-1}/f_{SBI}$

$t_{HIGH} = 2^{n-1}/f_{SBI} + 8/f_{SBI}$

$fscl = 1/(t_{LOW} + t_{HIGH})$

$= \frac{f_{SBI}}{2^n + 8}$

| SBI0CR1<SCK2:0> | n |
|---|---|
| 000 | 5 |
| 001 | 6 |
| 010 | 7 |
| 011 | 8 |
| 100 | 9 |
| 101 | 10 |
| 110 | 11 |

Note: $f_{SBI}$ shows $f_{SYS}$.

Figure 3.10.7 Clock Source

2. 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 TMP92CM22 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.

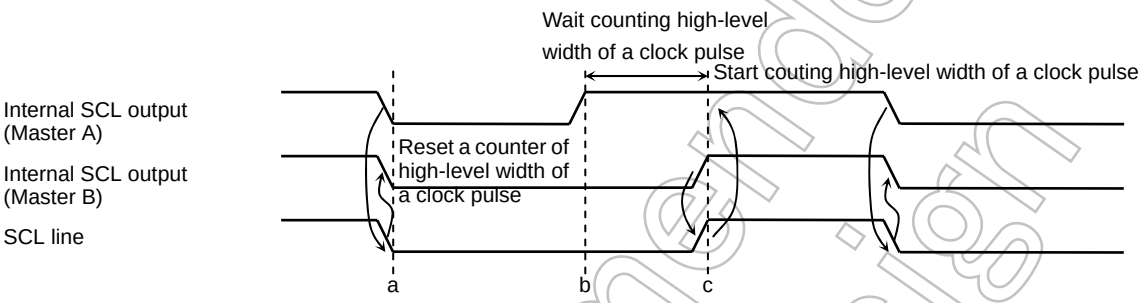



Figure 3.10.8 Clock Synchronization

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 waits 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 TMP92CM22 is used as a slave device, set the slave address <SA6:0> and <ALS> to the I2C0AR. Clear the <ALS> to “0” for the address recognition mode.

(5) Master/slave selection

Set the SBI0CR2<MST> to “1” for operating the TMP92CM22 as a master device. Clear the SBI0CR2<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 SBI0CR2<TRX> to "1" for operating the TMP92CM22 as a transmitter. Clear the <TRX> to "0" for operation as a receiver. In slave mode, when transfer data in addressing format, when received slave address is same value with setting value to I2C0AR, or 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 <TRX> 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 bus is detected or arbitration is lost.

(7) Start/stop condition generation

When programmed "1111" to SBI0CR2 <MST, TRX, BB, PIN> in during SBI0SR<BB> is "0", slave address and direction bit which are set to SBI0DBR and start condition are output on a bus. And it is necessary to set transmitted data to the data buffer register (SBI0DBR) and set "1" to <ACK> beforehand.

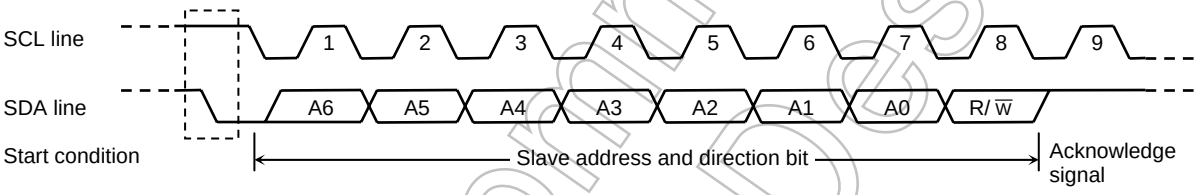



Figure 3.10.9 Generation of Start Condition and Slave Address

When programmed "0" to SBI0CR2<BB> and "111" to <MST, TRX, PIN> in during SBI0SR<BB> is "1", start a sequence of stop condition output. Do not modify the contents of <MST, TRX, BB, and PIN> until a stop condition is generated on a bus.

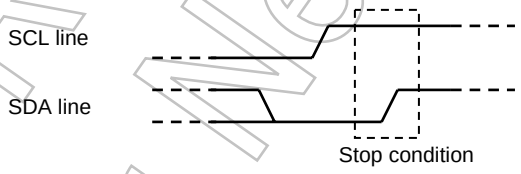



Figure 3.10.10 Generation of Stop Condition

The state of the bus can be ascertained by reading the contents of SBI0SR<BB>. SBI0SR<BB> will be set to 1 (Bus busy status) if a start condition has been detected on the bus, and will be cleared to 0 if a stop condition has been detected (Bus free status).

(8) Interrupt service requests and interrupt cancellation

When a serial bus interface interrupt request 0 (INTSBE0) occurs, the SBI0SR2 <PIN> is cleared to "0". During the time that the SBI0SR2<PIN> is "0", the SCL line is pulled down to the low level.

The <PIN> is cleared to "0" when end of transmission or receiving 1 word of data. And when writing data to SBI0DBR or reading data from SBI0DBR, <PIN> is set to "1".

The time from the <PIN> being set to "1" until the SCL line is released takes $t_{LOW}$.

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 I2C0AR or when a GENERAL CALL is received (All 8-bit data are "0" after a start condition). Although SBI0CR2<PIN> can be set to "1" by the program, the <PIN> is not clear it to "0" when it is programmed "0".

(9) Serial bus interface operation mode selection

SBI0CR2<SBIM1:0> is used to specify the serial bus interface operation mode. Set SBI0CR2<SBIM1:0> to "10" when the device is to be used in $I^2C$ 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 $I^2C$ bus mode, a bus arbitration procedure has been implemented in order to guarantee the integrity of transferred data.

Data on the SDA line is used for $I^2C$ 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 that 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.

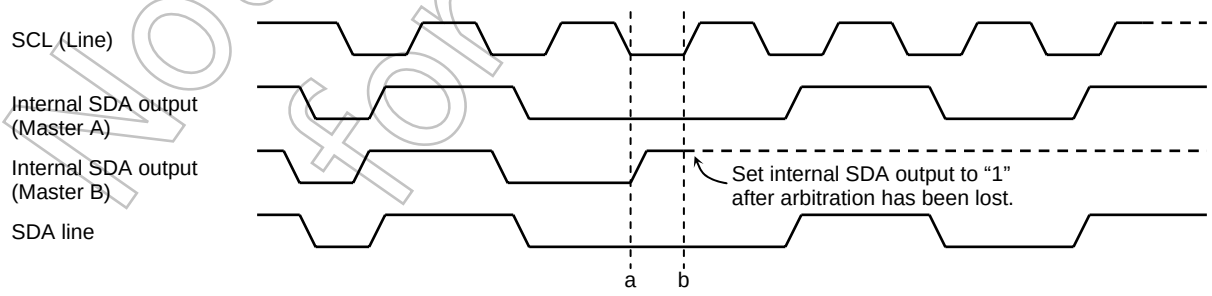



Figure 3.10.11 Arbitration Lost

The TMP92CM22 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 SBI0SR<AL> is set to "1".

When SBI0SR<AL> is set to "1", SBI0SR<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".

SBI0SR <AL> is cleared to "0" when data is written to or read from SBI0DBR or when data is written to SBI0CR2.

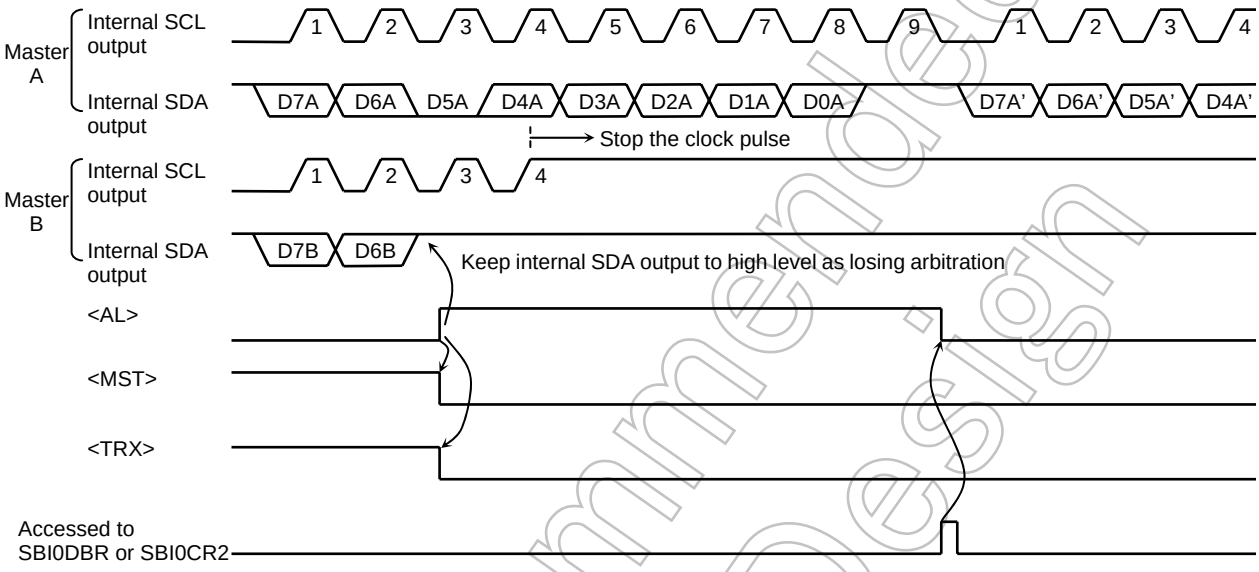



Figure 3.10.12 Example of when TMP92CM22 is a Master Device B (D7A = D7B, D6A = D6B)

(11) Slave address match detection monitor

SBI0SR<AAS> operates following in during slave mode; In address recognition mode (e.g., when I2C0AR<ALS> = "0"), when received GENERAL CALL or same slave address with value set to I2C0AR, SBI0SR<AAS> is set to "1". When <ALS> = "1", SBI0SR<AAS> is set to "1" after the first word of data has been received. SBI0SR<AAS> is cleared to "0" when data is written to SBI0DBR or read from SBI0DBR.

(12) GENERAL CALL detection monitor

SBI0SR<AD0> operates following in during slave mode; when received GENERAL CALL (all 8-bit data is "0", after a start condition), SBI0SR<AD0> is set to "1". And SBI0SR<AD0> is cleared to "0" when a start condition or stop condition on the bus is detected.

(13) Last received bit monitor

The value on the SDA line detected on the rising edge of the SCL line is stored in the SBI0SR<LRB>. In the acknowledge mode, immediately after an INTSBE0 interrupt request has been generated, an acknowledge signal is read by reading the contents of the SBI0SR<LRB>.

(14) Software reset function

The software reset function is used to initialize the SBI circuit, when SBI is rocked by external noises, etc.

When write first "10" next "01" to SBI0CR2<SWRST1:0>, reset signal is inputted to serial bus interface circuit, and circuit is initialized. All command registers except SBI0CR2<SBIM1:0> and status flag except SBI0CR2<SBIM1:0> are initialized to value of just after reset. SBI0CR1<SWRMON> is set to "1" automatically when completed initialization of serial bus interface.

(15) Serial bus interface data buffer register (SBI0DBR)

The received data can be read and transmission data can be written by reading or writing SBI0DBR.

In the master mode, after the slave address and the direction bit are set in this register, the start condition is generated.

(16) I²C bus address register (I2C0AR)

I2C0AR<SA6:0> is used to set the slave address when the TMP92CM22 functions as a slave device.

The slave address outputted from the master device is recognized by setting the I2C0AR<ALS> to "0". And, the data format becomes the addressing format. When set <ALS> to "1", the slave address is not recognized, the data format becomes the free data format.

(17) Baud rate register (SBI0BR1)

Write "1" to baud rate circuit control register SBI0BR1<P4EN> before using I²C bus.

(18) Setting register for IDLE2 mode operation (SBI0BR0)

SBI0BR0<I2SBI0> is the register setting operation/stop during IDLE2 mode. Therefore, setting <I2SBI0> is necessary before the HALT instruction is executed.

### 3.10.6 Data Transfer in $I^2C$ Bus Mode

(1) Device initialization

In first, set the SBI0BR1<P4EN>, SBI0CR1<ACK, SCK2:0>. Set SBI0BR1<P4EN> to "1" and clear bits 7 to 5 and 3 in the SBI0CR1 to "0".

Next, set a slave address <SA6:0> and the <ALS> (<ALS> = "0" when an addressing format) to the I2C0AR.

And, write "000" to SBI0CR2<MST, TRX, BB>, "1" to <PIN>, "10" to <SBIM1:0> and "00" to <SWRST1:0>. Set initialization status to slave receiver mode by this setting.

(2) Start condition and slave address generation

1. Master mode

In the master mode, the start condition and the slave address are generated as follows.

In first, check a bus free status (when SBI0SR<BB> = "0").
Set the SBI0CR1<ACK> to "1" (Acknowledge mode) and specify a slave address and a direction bit to be transmitted to the SBI0DBR.

When SBI0SR<BB> = "0", the start condition are generated by writing "1111" to SBI0CR2<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 SBI0DBR. At the 9th clock, the SDA line is released and the acknowledge signal is received from the slave device.

An INTSBE interrupt request generate 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 is generated, the <TRX> is changed according to the direction bit only when an acknowledge signal is returned from the slave device.

2. 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 I2C0AR is received, the SDA line is pulled down to the low level at the 9th clock, and the acknowledge signal is output.

An INTSBE interrupt request is generated 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".

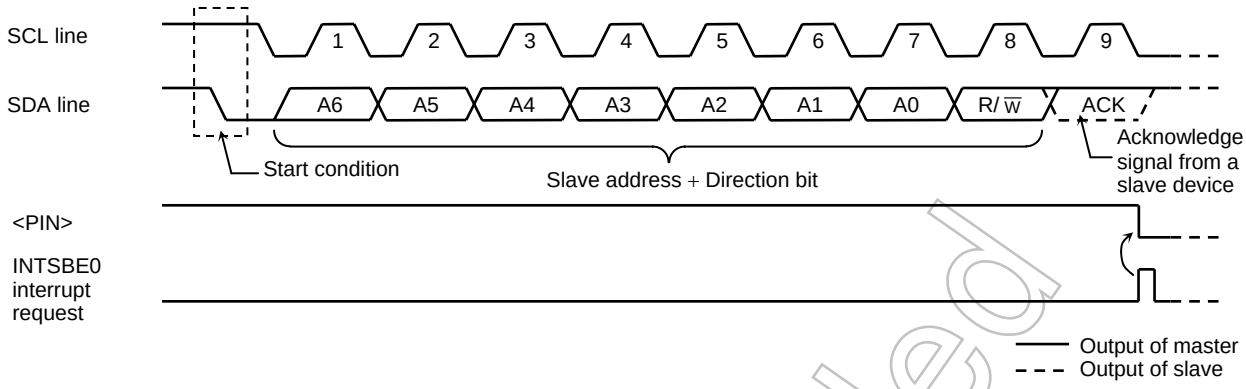



Figure 3.10.13 Start Condition and Slave Address Generation

(3) 1-word data transfer

Check the <MST> by the INTSBE0 interrupt process after the 1-word data transfer is completed, and determine whether the mode is a master or slave.

1. 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 3.10.6 (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 SBI0DBR. When the next transmitted data is other than 8 bits, set the <BC2:0> <ACK> and write the transmitted data to SBI0DBR. 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 INTSBE interrupt request generates. 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.

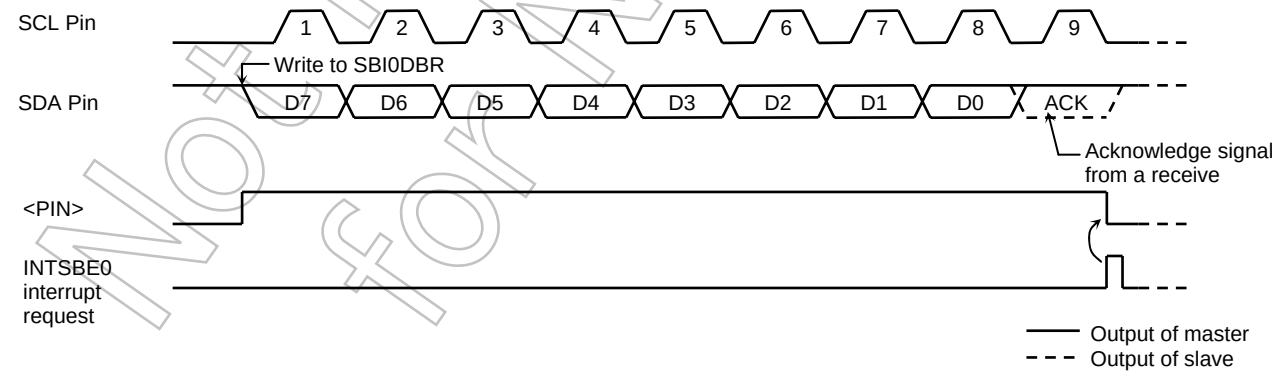



Figure 3.10.14 Example in which <BC2:0> = "000" and <ACK> = "1" (Transmitter mode)

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 SBI0DBR 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 INTSBE interrupt request then generates and the <PIN> becomes "0", Then the TMP92CM22 pulls down the SCL pin to the low level. The TMP92CM22 outputs a clock pulse for 1 word of data transfer and the acknowledge signal each time that received data is read from the SBI0DBR.

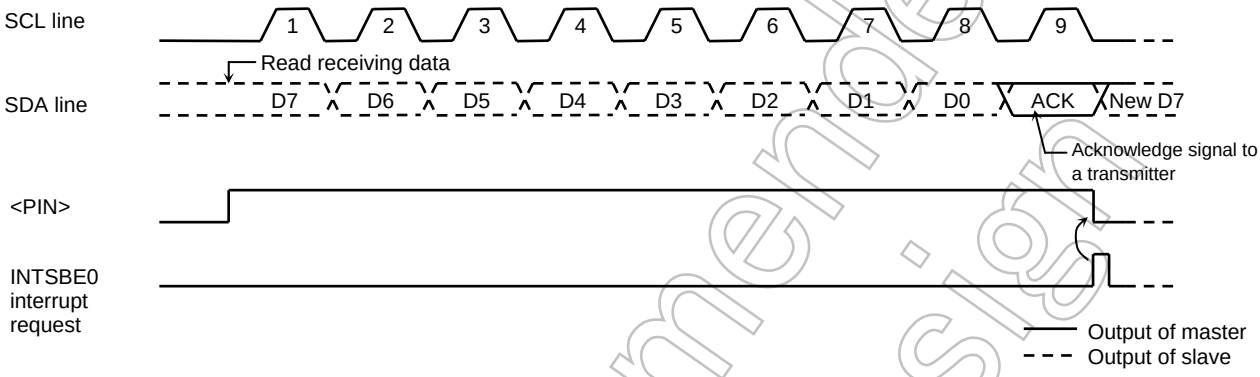



Figure 3.10.15 Example of when <BC2:0> = "000", <ACK> = "1" (Receiver mode)

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 TMP92CM22 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 receives the high signal as an ACK signal. The receiver indicates to the transmitter that the data transfer is completed.

After the one data bit has been received and an interrupt request has been generated, the TMP92CM22 generates a stop condition (See section 3.10.6 (4)) and terminates data transfer.

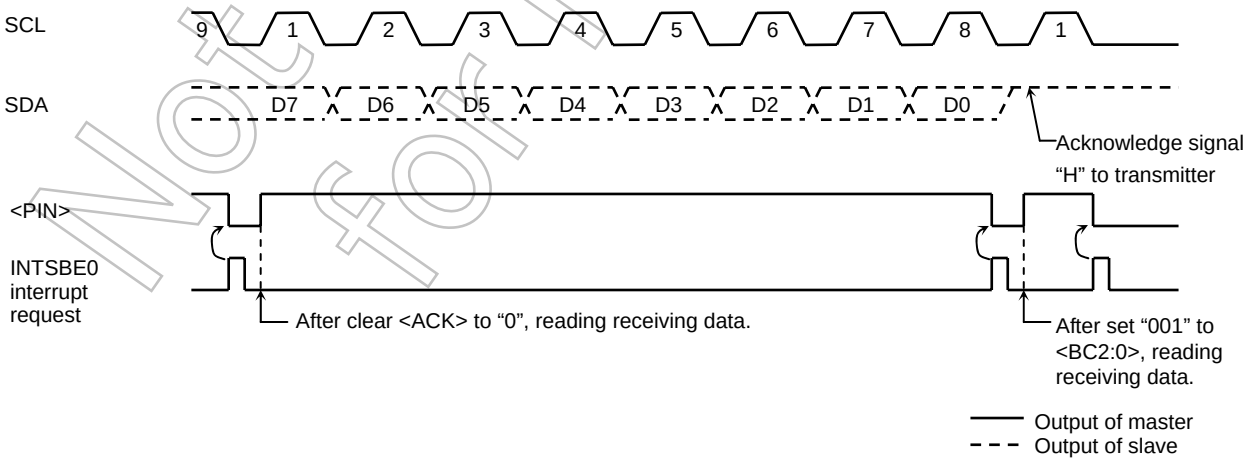



Figure 3.10.16 Termination of Data Transfer (Master receiver mode)

2. If <MST> = 0 (Slave mode)

In the slave mode the TMP92CM22 operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBE0 interrupt request generate when the TMP92CM22 receives a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is completed, or after matching received address. In the master mode, the TMP92CM22 operates in a slave mode if it losing arbitration. An INTSBE0 interrupt request is generated when a word data transfer terminates after losing arbitration. When an INTSBE0 interrupt request is generated the <PIN> is cleared to "0" and the SCL pin is pulled down to the low level. Either reading/writing from/to the SBI0DBR or setting the <PIN> to "1" will release the SCL pin after taking $t_{LOW}$ time.

Check the SBI0SR<AL>, <TRX>, <AAS>, and <AD0> and implements processes according to conditions listed in the next table.

Table 3.10.1 Operation in the Slave Mode

| <TRX> | <AL> | <AAS> | <AD0> | Conditions | Process |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | The TMP92CM22 detects arbitration lost 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 of single word to <BC2:0>, and write the transmit data to SBI0DBR. |
| | 0 | 1 | 0 | In salve receiver mode, the TMP92CM22 receives a slave address for which the value of the direction bit sent from the master is "1". | |
| | | 0 | 0 | In salve transmitter mode, transmission of data of single word is terminated. | Check the <LRB>, If <LRB> is set to "1", set <PIN> to "1", reset "0" to <TRX> and release the bus for the receiver no request next data. If <LRB> was cleared to "0", set bit number of single word to <BC2:0> and write the transmit data to SBI0DBR for the receiver requests next data. |
| 0 | 1 | 1 | 1/0 | The TMP92CM22 detects arbitration lost 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 SBI0DBR for setting the <PIN> to "1" (Reading dummy data) or set the <PIN> to "1". |
| | | 0 | 0 | The TMP92CM22 detects arbitration lost when transmitting a slave address or data, and transfer of word terminates. | |
| | 0 | 1 | 1/0 | In slave receiver mode the TMP92CM22 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 TMP92CM22 terminates receiving word data. | Set bit number of single word to <BC2:0>, and read the receiving data from SBI0DBR. |

(4) Stop condition generation

When SBI0SR<BB> = 1, the sequence for generating a stop condition is started by writing "111" to SBI0CR2<MST, TRX, PIN> and "0" to SBI0CR2<BB>. Do not modify the contents of SBI0CR2<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 TMP92CM22 generates a stop condition when the other device has released the SCL line and SDA pin rising.

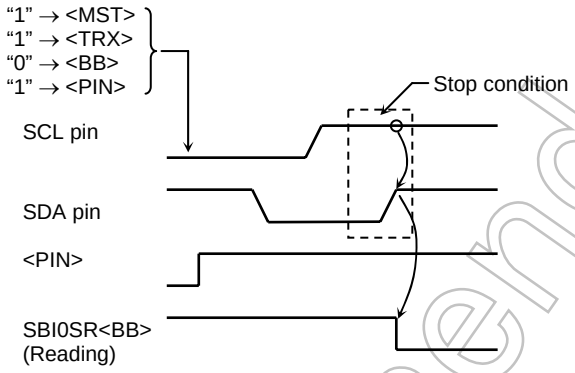



Figure 3.10.17 Stop Condition Generation (Single master)

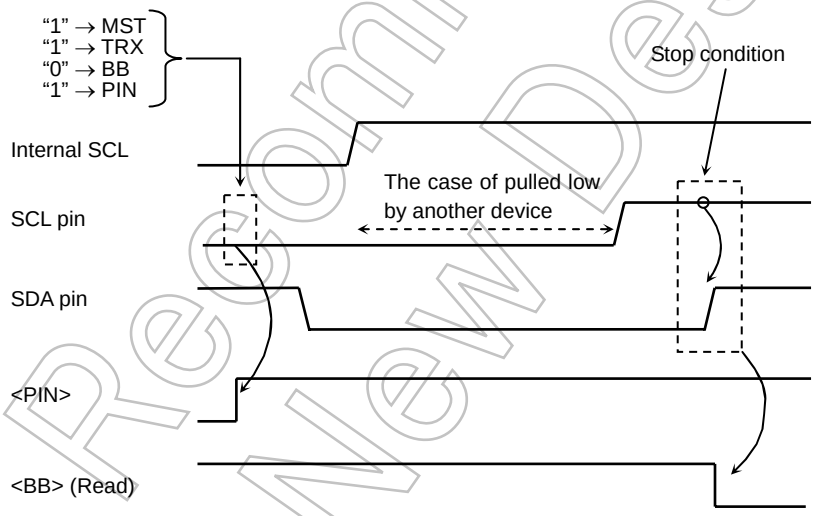



Figure 3.10.18 Stop Condition Generation (Multi master)

(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 this device is in the master mode.

Clear the SBI0CR2<MST, TRX, BB> to "000" and set the SBI0CR2<PIN> to "1" to release the bus. The SDA line remains the high level and the SCL pin is released. Since a stop condition is not generated on the bus, other devices assume the bus to be in a busy state. Check the SBI0SR<BB> until it becomes "0" to check that the SCL pin of this device is released. 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 stays in a free state, generate a start condition with procedure described in 3.10.6 (2).

In order to meet setup time 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.

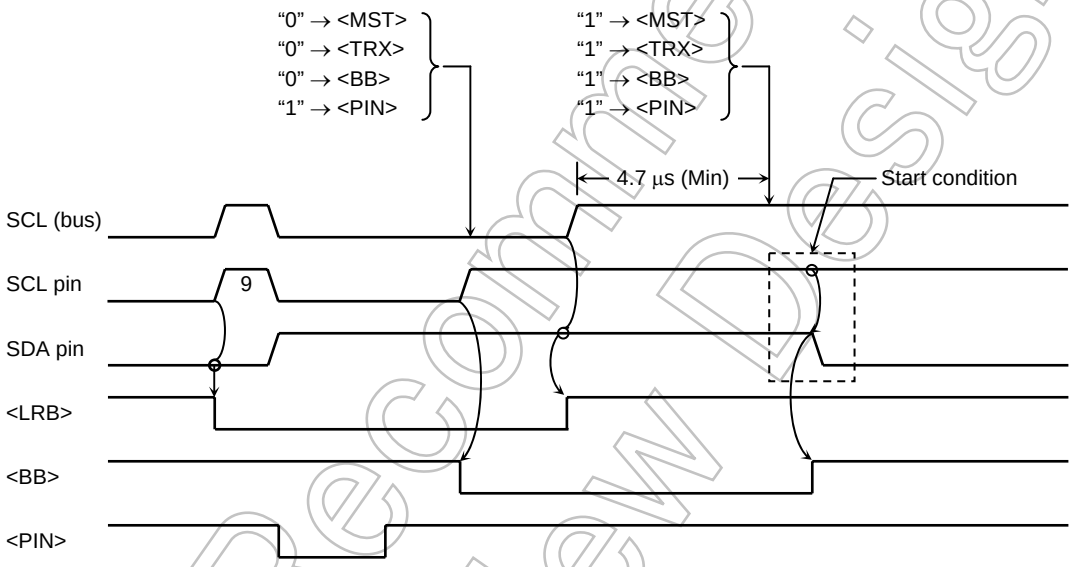



Figure 3.10.19 Timing Diagram when Restarting

### 3.10.7 Clocked-synchronous 8-bit SIO Mode Control

The following registers are used to control and monitor the operation status when the serial bus interface (SBI) is being operated in clocked synchronous 8-bit SIO mode.

Serial Bus Interface 0 Control Register 1

SBI0CR1 (1240H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | SIOS | SIOINH | SIOM1 | SIOM0 | | SCK2 | SCK1 | SCK0 |
| Read/Write | W | | | | | W | | W |
| After reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| Function | Transfer start<br>0: Stop<br>1: Start | Continue/ abort transfer<br>0: Continue transfer<br>1: Abort transfer | Transfer mode select<br>00: Transmit mode<br>01: (Reserved)<br>10: Transmit/receive mode<br>11: Receive mode | | | Serial clock selection and reset monitor | | |

Serial clock selection <SCK2:0> at write

| | | |
|---|---|---|
| 000 | n = 4 | 1.25 MHz |
| 001 | n = 5 | 625 kHz |
| 010 | n = 6 | 313 kHz |
| 011 | n = 7 | 156 kHz |
| 100 | n = 8 | 78.1 kHz |
| 101 | n = 9 | 39.1 kHz |
| 110 | n = 10 | 19.5 kHz |
| 111 | – | (External clock : SCK0) |

System clock: $f_{SYS}$
$f_{SYS}$ = 20 MHz (SCL output to SCK pin)
$fscl = \frac{f_{SYS}}{2^n}$ [Hz]

Transfer mode selection

| | |
|---|---|
| 00 | 8-bit transmit mode |
| 01 | (Reserved) |
| 10 | 8-bit transmit/receive mode |
| 11 | 8-bit receive mode |

Continue/abort transfer

| | |
|---|---|
| 0 | Continue transfer |
| 1 | Abort transfer (Automatically cleared after transfer aborted) |

Indicate transfer start/stop

| | |
|---|---|
| 0 | Stop |
| 1 | Start |

Note: Set the transfer mode and the serial clock after setting <SIOS> to “0” and <SIOINH> to “1”.

Serial Bus Interface 0 Data Buffer Register

SBI0DBR (1241H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| Read/Write | R (Receiver)/W (Transfer) | | | | | | | |
| After reset | Undefined | | | | | | | |

Figure 3.10.20 Register for the SIO Mode

Serial Bus Interface 0 Control Register 2

SBI0CR2 (1243H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | SBIM1 | SBIM0 | – | – |
| Read/Write | | | | | W | | | |
| After reset | | | | | 0 | 0 | 0 | 0 |
| Function | | | | | Serial bus interface operation mode selection<br>00: Port mode<br>01: SIO mode<br>10: $I^2C$ bus mode<br>11: (Reserved) | | (Note 2) | (Note 2) |

Note 1: Set the SBI0CR1<BC2:0> to "000" before switching to a clocked-synchronous 8-bit SIO mode.

Note 2: Please always write "00" to SBICR2<1:0>.

Serial bus interface operation mode selection

| | |
|---|---|
| 00 | Port mode (Serial bus interface output disabled) |
| 01 | Clocked-synchronous 8-bit SIO mode |
| 10 | $I^2C$ bus mode |
| 11 | (Reserved) |

Serial Bus Interface 0 Status Register

SBI0SR (1243H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | SIOF | SEF | | |
| Read/Write | | | | | R | | | |
| After reset | | | | | 0 | 0 | | |
| Function | | | | | Serial transfer operation status monitor | Shift operation status monitor | | |

Serial transfer operating status monitor

| | |
|---|---|
| 0 | Transfer terminated |
| 1 | Transfer in progress |

Shift operation status monitor

| | |
|---|---|
| 0 | Shift operation terminated |
| 1 | Shift operation in progress |

Serial Bus Interface 0 Baud Rate Register 0

SBI0BR0 (1244H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | – | | | | | | |
| Read/Write | W | R/W | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | Always write "0". | Always write "0". | | | | | | |

Note: Clocked-syncronous mode cannot operate in IDEL2 mode.

Serial Bus Interface 0 Baud Rate Register 1

SBI0BR1 (1245H)

Read-modify-write instruction is prohibited.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P4EN | – | | | | | | |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | | | | | | |
| Function | Internal clock<br>0: Stop<br>1: Operate | Always write "0". | | | | | | |

Baud rate clock control

| | |
|---|---|
| 0 | Stop |
| 1 | Operate |



Figure 3.10.21 Register for the SIO Mode

(1) Serial Clock

1. Clock source

SBI0CR1<SCK2:0> is used to select the following functions:

Internal clock

In internal clock mode one of seven frequencies can be selected. The serial clock signal is output to the outside on the SCK pin.

When the device is writing (in transmit mode) or reading (in receive mode), data cannot follow the serial clock rate, so an automatic wait function is executed which automatically stops the serial clock and holds the next shift operation until reading or writing has been completed.

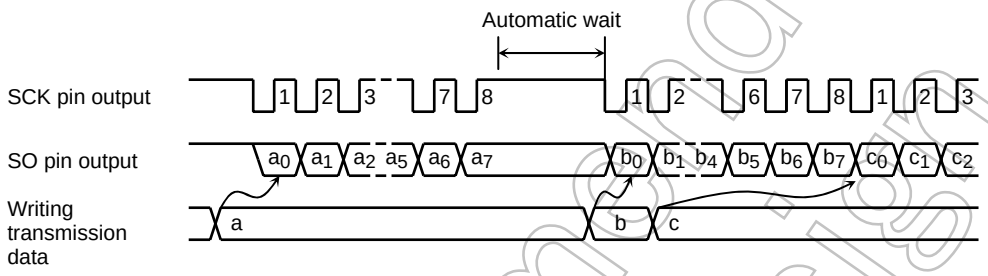



Figure 3.10.22 Automatic Wait Function

External clock (<SCK2:0> = "111")

An external clock input via the SCK pin is used as the serial clock. In order to ensure the integrity of shift operations, both the high and low-level serial clock pulse widths shown below must be maintained. The maximum data transfer frequency is 1.25 MHz (when $f_{SYS}$ = 20 MHz).

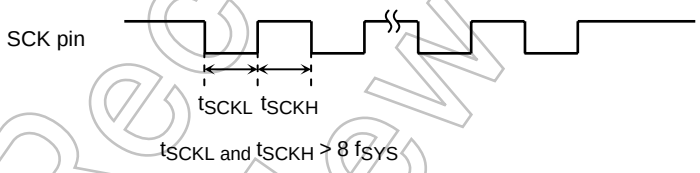



Figure 3.10.23 Maximum Data Transfer Frequency when External Clock Input

2. Shift edge

Data is transmitted on the leading edge of the clock and received on the trailing edge.

Leading edge shift

Data is shifted on the leading edge of the serial clock (on the falling edge of the SCK pin input/output).

Trailing edge shift

Data is shifted on the trailing edge of the serial clock (on the rising edge of the SCK pin input/output).

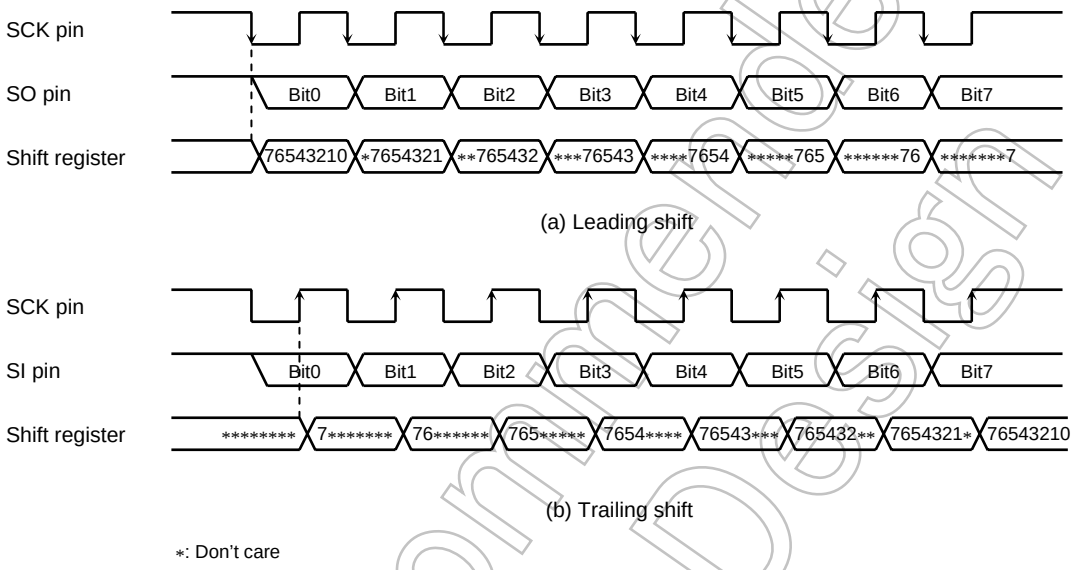



Figure 3.10.24 Shift Edge

(2) Transfer modes

The SBI0CR1<SIOM1:0> is used to select a transmit, receive or transmit/receive mode.

1. 8-bit transmit mode

Set a control register to a transmit mode and write transmission data to the SBI0DBR.

After the transmit data has been written, set the SBI0CR1<SIOS> to "1" to start data transfer. The transmitted data is transferred from the SBI0DBR to the shift register and output, starting with the least significant bit (LSB), via the SO pin and synchronized with the serial clock. When the transmission data has been transferred to the shift register, the SBI0DBR becomes empty. The INTSBE0 (Buffer empty) interrupt request is generated to request new data.

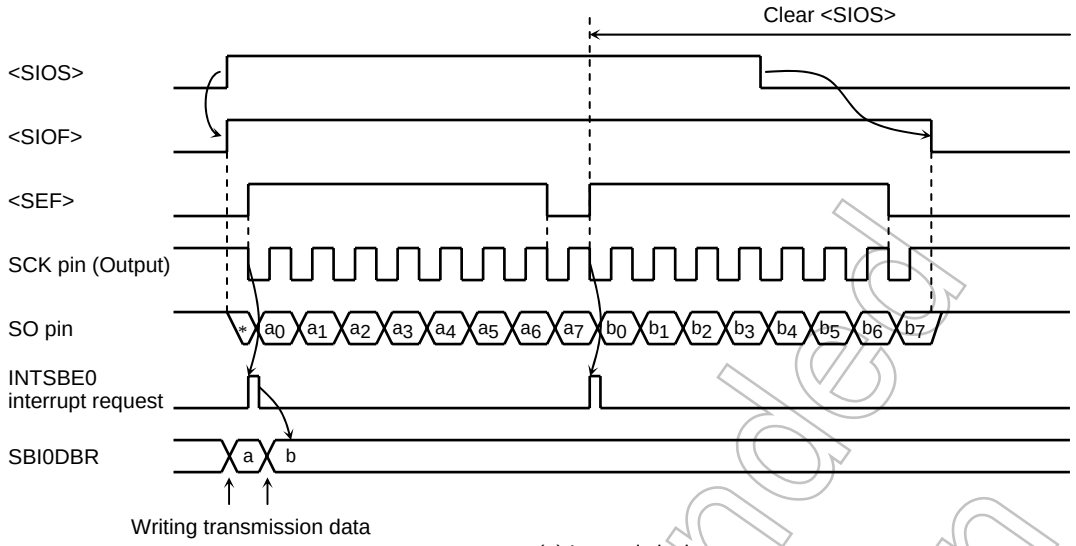
When the internal clock is used, the serial clock will stop and the automatic wait function will be initiated if new data is not loaded to the data buffer register after the specified 8-bit data is transmitted. When new transmission data is written, the automatic wait function is canceled.

When the external clock is used, data should be written to the SBI0DBR before new data is shifted. The transfer speed is determined by the maximum delay time between the time when an interrupt request is generated and the time when data is written to the SBI0DBR by the interrupt service program.

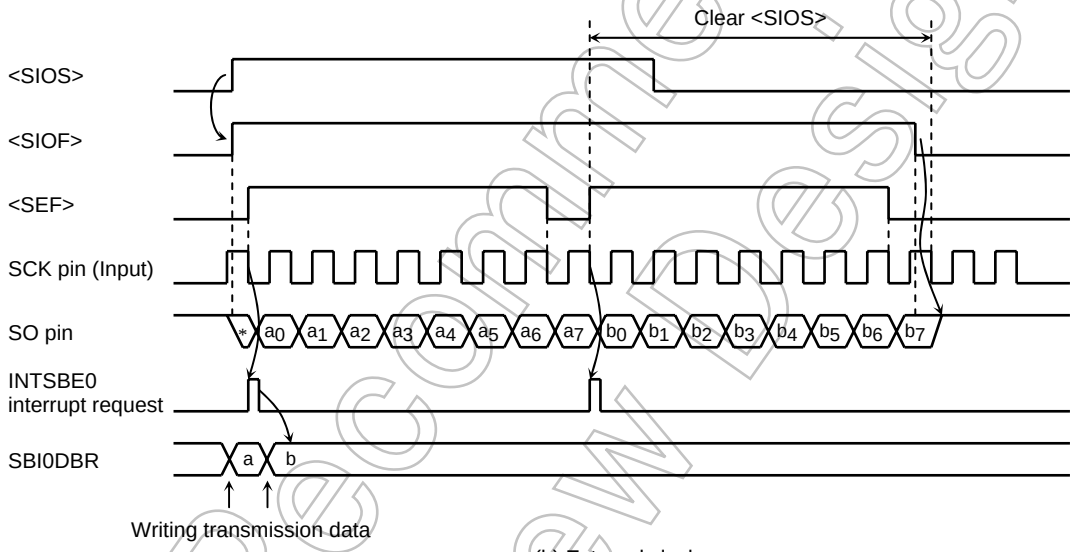
When the transmit is started, after the SBI0SR<SIOF> goes "1" output from the SO pin holds final bit of the last data until falling edge of the SCK.

For stopping data transmission, when the <SIOS> is cleared to "0" by the INTSBE0 interrupt service program or when the <SIOINH> is set to "1". When the <SIOS> is cleared to "0", the transmitted mode ends when all data is output. In order to confirm whether data is being transmitted properly by the program, the <SIOF> to be sensed. The SBI0SR<SIOF> is cleared to "0" when transmission has been completed. When the <SIOINH> is set to "1", transmitting datat stops. The <SIOF> turns "0".

When the external clock is used, it is also necessary to clear the <SIOS> to "0" before new data is shifted; otherwise, dummy data is transmitted and operation ends.




(a) Internal clock




(b) External clock

Figure 3.10.25 Transmission Mode

Example: Program to stop data transmission (when an external clock is used)

```
STEST1 :    BIT     2, (SBI0SR)              ;  If <SEF> = 1 then loop.
            JR      NZ, STEST1
STEST2 :    BIT     0, (P9)                  ;  If SCK = 0 then loop.
            JR      Z, STEST2
            LD      (SBI0CR1), 00000111B     ;  <SIOS> ← 0
```

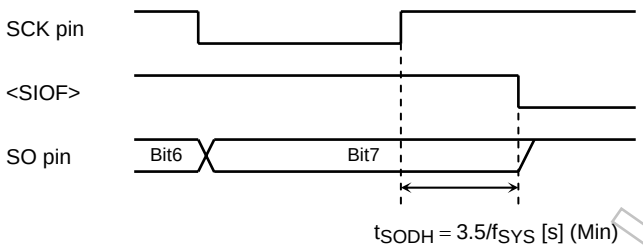



Figure 3.10.26 Transmission Data Hold Time at End Transmit

2. 8-bit receive mode

Set the control register to receive mode and set the SBI0CR1<SIOS> to "1" for switching to receive mode. Data is received into the shift register via the SI pin and synchronized with the serial clock, starting from the least significant bit (LSB). When the 8-bit data is received, the data is transferred from the shift register to the SBI0DBR. The INTSBE0 (Buffer full) interrupt request is generated to request that the received data be read. The data is then read from the SBI0DBR by the interrupt service program.

When the internal clock is used, the serial clock will stop and the automatic wait function will be in effect until the received data is read from the SBI0DBR.

When the external clock is used, since shift operation is synchronized with an external clock pulse, the received data should be read from the SBI0DBR before the next serial clock pulse is input. If the received data is not read, further data to be received is canceled. The maximum transfer speed when an external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when the received data is read.

Receiving of data ends when the <SIOS> is cleared to "0" by the INTSBE0 interrupt service program or when the <SIOINH> is set to "1". If <SIOS> is cleared to "0", received data is transferred to the SBI0DBR in complete blocks. The received mode ends when the transfer is completed. In order to confirm whether data is being received properly by the program, the SBI0SR<SIOF> to be sensed. The <SIOF> is cleared to "0" when receiving is completed. When it is confirmed that receiving has been completed, the last data is read. When the <SIOINH> is set to "1", data receiving stops. The <SIOF> is cleared to "0" (The received data becomes invalid, therefore no need to read it).

Note: When the transfer mode is changed, the contents of the SBI0DBR will be lost. If the mode must be changed, conclude data receiving by clearing the <SIOS> to "0", read the last data, then change the mode.

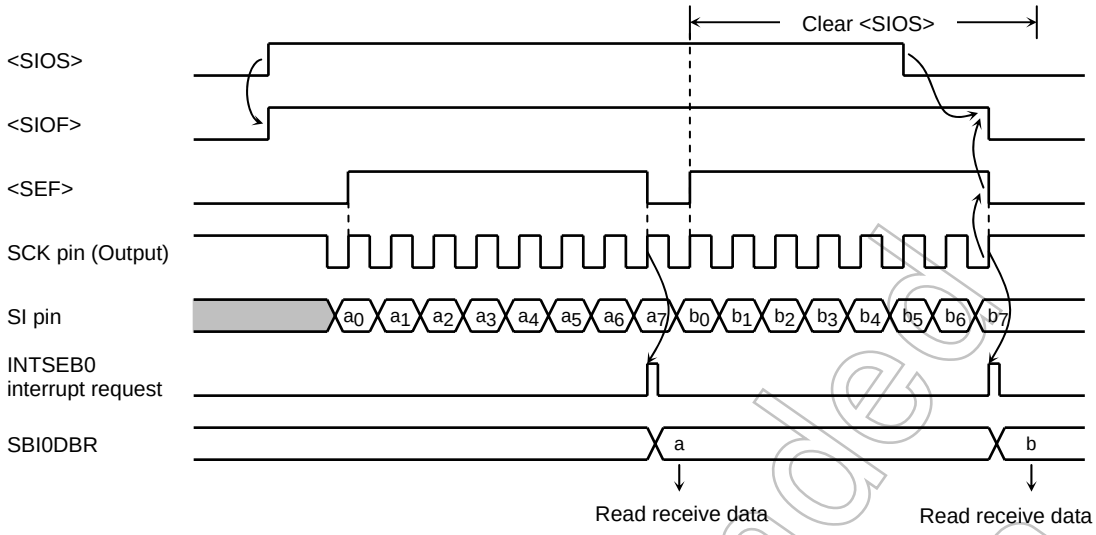



Figure 3.10.27 Receiver Mode (Example: Internal clock)

3. 8-bit transmit/receive mode

Set a control register to a transmit/receive mode and write data to the SBI0DBR. After the data is written, set the SBI0CR<SIOS> to "1" to start transmitting/receiving. When data is transmitted, the data is output from the SO pin, starting from the least significant bit (LSB) and synchronized with the leading edge of the serial clock signal. When data is received, the data is input via the SI pin on the trailing edge of the serial clock signal. 8-bit data is transferred from the shift register to the SBI0DBR and the INTSBE0 interrupt request is generated. The interrupt service program reads the received data from the data buffer register and writes the data which is to be transmitted. The SBI0DBR is used for both transmitting and receiving. Transmitted data should always be written after received data is read.

When the internal clock is used, the automatic wait function will be in effect until the received data is read and the new data is written.

When the external clock is used, since the shift operation is synchronized with the external clock, the received data is read and transmitted data is written before a new shift operation is executed. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time at which received data is read and transmitted data is written.

When the transmit is started, after the SBI0SR<SIOF> goes "1" output from the SO pin holds final bit of the last data until falling edge of the SCK.

Transmitting/receiving data ends when the <SIOS> is cleared to "0" by the INTSBE0 interrupt service program or when the SBI0CR1<SIOINH> is set to "1". When the <SIOS> is cleared to "0", received data is transferred to the SBI0DBR in complete blocks. The transmit/receive mode ends when the transfer is completed. In order to confirm whether data is being transmitted/received properly by the program, set the SBI0SR to be sensed. The <SIOF> is cleared to "0" when transmitting/receiving is completed. When the <SIOINH> is set to "1", data transmitting/receiving stops. The <SIOF> is then cleared to "0".

Note: When the transfer mode is changed, the contents of the SBI0DBR will be lost. If the mode must be changed, conclude data transmitting/receiving by clearing the <SIOS> to "0", read the last data, and then change the transfer mode.

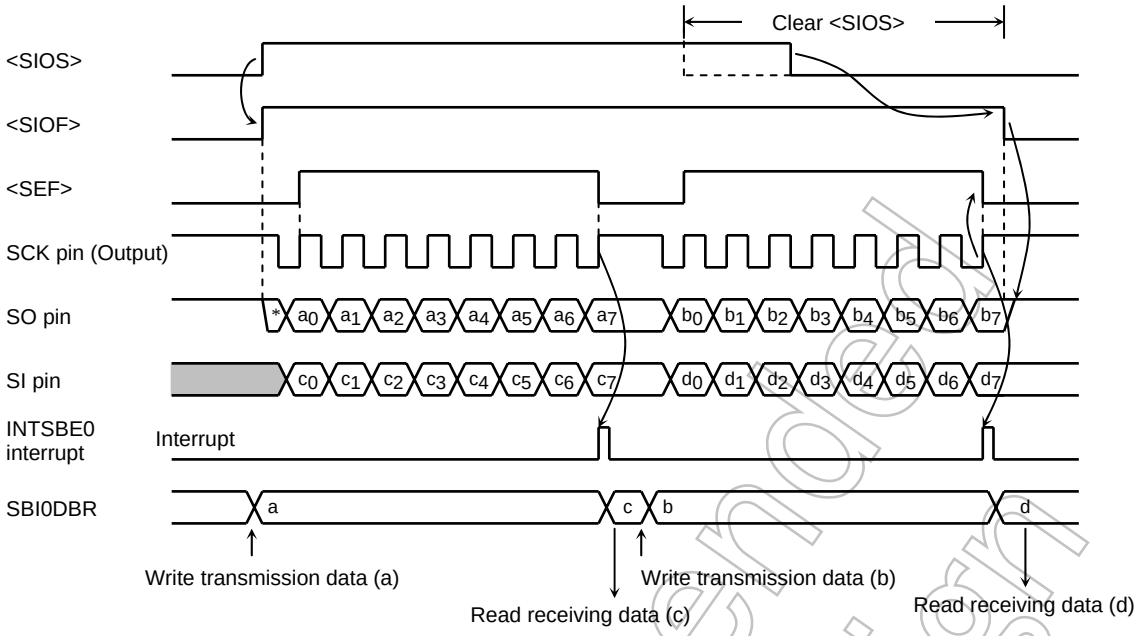



Figure 3.10.28 Transmission/Receiving Mode (when an external clock is used)

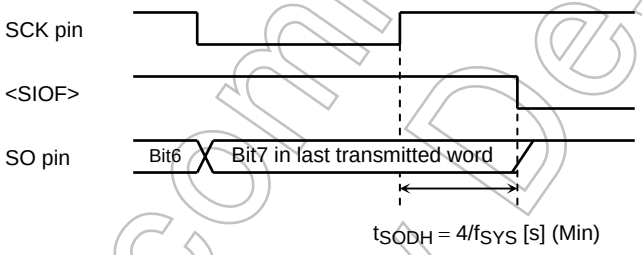



Figure 3.10.29 Transmission Data Hold Time at End of Transmission/Receiving (Transmission/receiving mode)

## 3.11 Analog/Digital Converter

The TMP92CM22 incorporates a 10-bit successive approximation-type analog/digital converter (AD converter) with 8-channel analog input.

Figure 3.11.1 is a block diagram of the AD converter. The 8-channel analog input pins (AN0 to AN7) are shared with the input-only port G so they can be used as an input port.

Note: When IDLE2, IDLE1, or STOP mode is selected, as to reduce the power, with some timings the system may enter a standby mode even though the internal comparator is still enabled. Therefore be sure to check that AD converter operations are halted before a HALT instruction is executed.

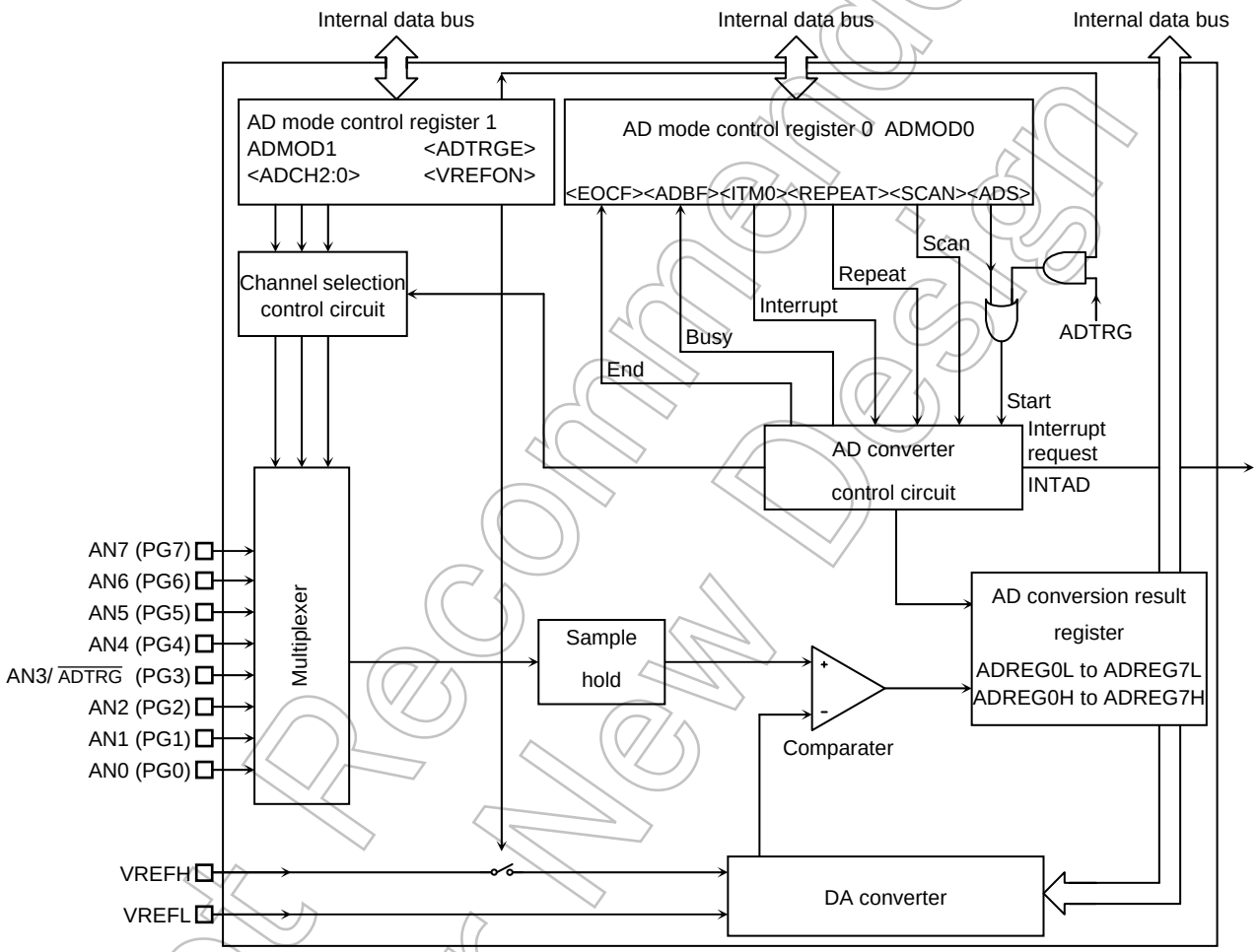



Figure 3.11.1 Block Diagram of AD Converter

### 3.11.1 Analog/Digital Converter Registers

The AD converter is controlled by the three AD mode control registers: ADMOD0, ADMOD1, and ADMOD2. The eight AD conversion data result registers (ADREG0H/L to ADREG7H/L) store the results of AD conversion.

Figure 3.11.2 shows the registers related to the AD converter.

AD Mode Control Register 0

ADMOD0 (12B8H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | EOCF | ADBF | – | – | ITM0 | REPEAT | SCAN | ADS |
| Read/Write | R | | R/W | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | AD conversion end flag<br>0: Conversion in progress<br>1: Conversion complete | AD conversion busy flag<br>0: Conversion stopped<br>1: Conversion in progress | Always write "0". | Always write "0". | Interrupt specification in conversion channel fixed repeat mode<br>0: Every conversion<br>1: Every fourth conversion | Repeat mode specification<br>0: Single conversion<br>1: Repeat conversion mode | Scan mode specification<br>0: Conversion channel fixed mode<br>1: Conversion channel scan mode | AD conversion start<br>0: Don't care<br>1: Start conversion<br>Always 0 when read. |

AD conversion start

| | |
|---|---|
| 0 | Don't care |
| 1 | Start AD conversion. |

Note: Always read as 0.

AD scan mode setting

| | |
|---|---|
| 0 | AD conversion channel fixed mode |
| 1 | AD conversion channel scan mode |

AD repeat mode setting

| | |
|---|---|
| 0 | AD single conversion mode |
| 1 | AD repeat conversion 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 | Generates interrupt every fourth conversion. |

AD conversion busy flag

| | |
|---|---|
| 0 | AD conversion stopped |
| 1 | AD conversion in progress |

AD conversion in progress

| | |
|---|---|
| 0 | Before or during AD conversion |
| 1 | AD conversion complete |



Figure 3.11.2 Register for AD Converter

AD Mode Control Register 1

ADMOD1 (12B9H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | VREFON | I2AD | – | – | – | ADCH2 | ADCH1 | ADCH0 |
| Read/Write | R/W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | VREF application control<br>0: OFF<br>1: ON | IDLE2<br>0: Stop<br>1: Operate | Always write “0”. | Always write “0”. | Always write “0”. | Analog input channel selection | | |

Analog input channel selection

| <ADCH2:0> \ <SCAN> | 0 (Channel fixed) | 1 (Channel scanned) |
|---|---|---|
| 000 | AN0 | AN0 |
| 001 | AN1 | AN0 → AN1 |
| 010 | AN2 | AN0 → AN1 → AN2 |
| 011 (Note) | AN3 | AN0 → AN1 → AN2 → AN3 |
| 100 (Note) | AN4 | AN0 → AN1 → AN2 → AN3 → AN4 |
| 101 (Note) | AN5 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 |
| 110 (Note) | AN6 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 → AN6 |
| 111 (Note) | AN7 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 → AN6 → AN7 |

IDLE2 control

| | |
|---|---|
| 0 | Stopped |
| 1 | In operation |

Control of application of reference voltage to AD converter

| | |
|---|---|
| 0 | OFF |
| 1 | ON |

Before starting conversion (before writing 1 to ADMOD0<ADS>), set the <VREFON> bit to 1.

AD Mode Control Register 2

ADMOD2 (12BAH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | | | ADTRGE |
| Read/Write | | | | | | | | R/W |
| After reset | | | | | | | | 0 |
| Function | | | | | | | | AD conversion trigger start control<br>0: Disable<br>1. Enable |

AD conversion start control by external trigger ( $\overline{\text{ADTRG}}$ input )

| | |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

Note: As pin AN3 also functions as the $\overline{\text{ADTRG}}$ input pin, do not set <ADCH2:0> = “011, 100,101,110,111” when using $\overline{\text{ADTRG}}$ with <ADTRGE> set to “1”.

Figure 3.11.3 Register for AD Converter

AD Conversion Result Register 0 Low

ADREG0L (12A0H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR01 | ADR00 | | | | | | ADR0RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result | | | | | | | AD conversion data storage flag 1: Conversion result stored |

AD Conversion Result Register 0 High

ADREG0H (12A1H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR09 | ADR08 | ADR07 | ADR06 | ADR05 | ADR04 | ADR03 | ADR02 |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits AD conversion result. | | | | | | | |

AD Conversion Result Register 1 Low

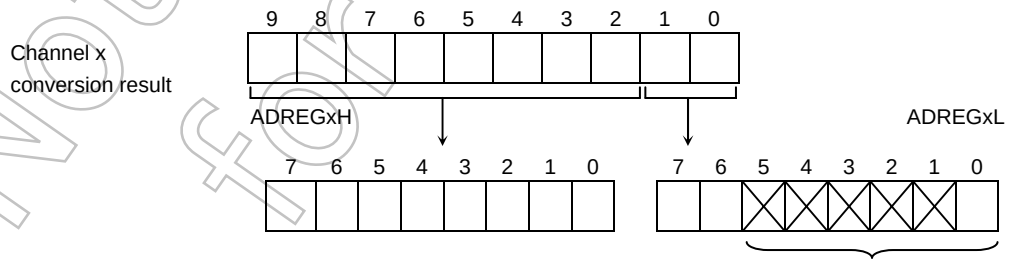
ADREG1L (12A2H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR11 | ADR10 | | | | | | ADR1RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result | | | | | | | AD conversion data storage flag 1: Conversion result stored |

AD Conversion Result Register 1 High

ADREG1H (12A3H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR19 | ADR18 | ADR17 | ADR16 | ADR15 | ADR14 | ADR13 | ADR12 |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |




- Bits 5 to 1 are always read as 1.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.4 Register for AD Converter

AD Conversion Result Register 2 Low

ADREG2L (12A4H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR21 | ADR20 | | | | | | ADR2RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1: Conversion result stored |

AD Conversion Result Register 2 High

ADREG2H (12A5H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR29 | ADR28 | ADR27 | ADR26 | ADR25 | ADR24 | ADR23 | ADR22 |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |

AD Conversion Result Register 3 Low

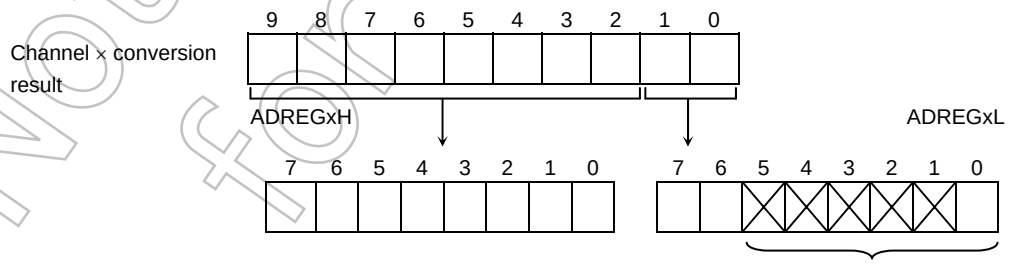
ADREG3L (12A6H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR31 | ADR30 | | | | | | ADR3RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1:Conversion result 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 | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |




- Bits 5 to 1 are always read as 1.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.5 Register for AD Converter

AD Conversion Result Register 4 Low

ADREG4L (12A8H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR41 | ADR40 | | | | | | ADR4RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1: Conversion result 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 | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |

AD Conversion Result Register 5 Low

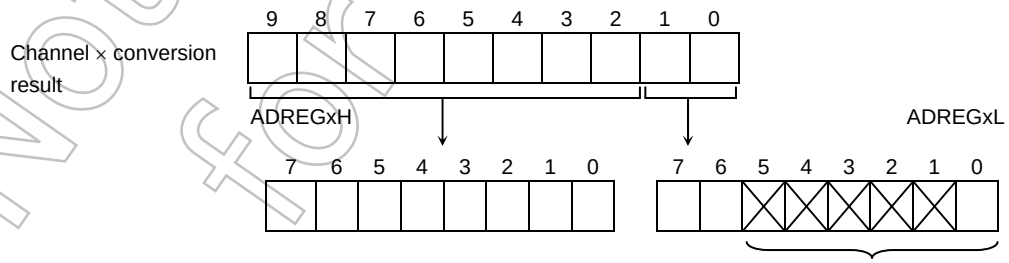
ADREG5L (12AAH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR51 | ADR50 | | | | | | ADR5RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1: Conversion result 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 | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |




- Bits 5 to 1 are always read as 1.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.6 Register for AD Converter

AD Conversion Result Register 6 Low

ADREG6L (12ACH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR61 | ADR60 | | | | | | ADR6RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1:Conversion result stored |

AD Conversion Result Register 6 High

ADREG6H (12ADH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR69 | ADR68 | ADR67 | ADR66 | ADR65 | ADR64 | ADR63 | ADR62 |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |

AD Conversion Result Register 7 Low

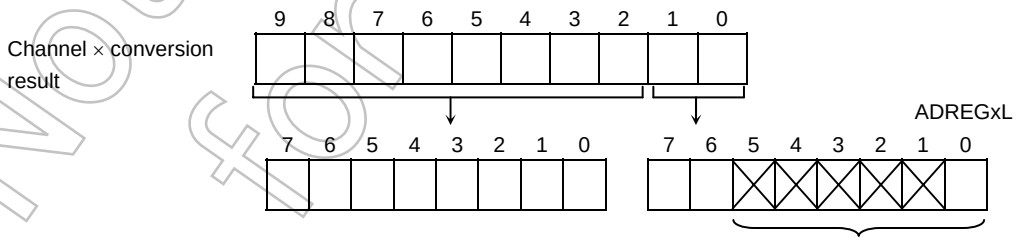
ADREG7L (12AEH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR71 | ADR70 | | | | | | ADR7RF |
| Read/Write | R | | | | | | | R |
| After reset | Undefined | | | | | | | 0 |
| Function | Stores lower 2 bits of AD conversion result. | | | | | | | AD conversion data storage flag 1: Conversion result stored |

AD Conversion Result Register 7 High

ADREG7H (12AFH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR79 | ADR78 | ADR77 | ADR76 | ADR75 | ADR74 | ADR73 | ADR72 |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Stores upper 8 bits of AD conversion result. | | | | | | | |




- Bits 5 to 1 are always read as 1.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.7 Register for AD Converter

### 3.11.2 Description of Operation

(1) Analog reference voltage

A high-level analog reference voltage is applied to the VREFH pin; a low-level analog reference voltage is applied to the VREFL pin. To perform AD conversion, the reference voltage, the difference between VREFH and VREFL, is divided by 1024 using string resistance. The result of the division is then compared with the analog input voltage.

To turn off the switch between VREFH and VREFL, program a 0 to ADMOD1<VREFON> in AD mode control register 1. To start AD conversion in the OFF state, first write a 1 to ADMOD1<VREFON>, wait 3 μs until the internal reference voltage stabilizes (This is not related to $f_{SYS}$), then set ADMOD0<ADS> to 1.

(2) Analog input channel selection

The analog input channel selection varies depends on the operation mode of the AD converter.

- In analog input channel fixed mode (ADMOD0<SCAN> = 0)
  Setting ADMOD1<ADCH2:0> selects one of the input pins AN0 to AN7 as the input channel.
- In analog input channel scan mode (ADMOD0<SCAN> = 1)
  Setting ADMOD1<ADCH2:0> selects one of the eight scan modes.

Table 3.11.1 illustrates analog input channel selection in each operation mode.

On a reset, ADMOD0<SCAN> is set to "0" and ADMOD1<ADCH2:0> is initialized to "000". Thus pin AN0 is selected as the fixed input channel. Pins not used as analog input channels can be used as standard input port pins.

Table 3.11.1 Analog Input Channel Selection

| <ADCH2:0> | Channel fixed <SCAN> = "0" | Channel scan <SCAN> = "1" |
|---|---|---|
| 000 | AN0 | AN0 |
| 001 | AN1 | AN0 → AN1 |
| 010 | AN2 | AN0 → AN1 → AN2 |
| 011 | AN3 | AN0 → AN1 → AN2 → AN3 |
| 100 | AN4 | AN0 → AN1 → AN2 → AN3 → AN4 |
| 101 | AN5 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 |
| 110 | AN6 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 → AN6 |
| 111 | AN7 | AN0 → AN1 → AN2 → AN3 → AN4 → AN5 → AN6 → AN7 |

(3) Starting AD conversion

To start AD conversion, program "1" to ADMOD0<ADS> in AD mode control register 0, or ADMOD1<ADTRGE> in AD mode control register 1 and input falling edge on ADTRG pin. When AD conversion starts, the AD conversion busy flag ADMOD0<ADBF> will be set to "1", indicating that AD conversion is in progress.

(4) AD conversion modes and the AD conversion end interrupt

The four AD conversion modes are:

- Channel fixed single conversion mode
- Channel scan single conversion mode
- Channel fixed repeat conversion mode
- Channel scan repeat conversion mode

The ADMOD0<REPEAT> and ADMOD0<SCAN> settings in AD mode control register 0 determine the AD mode setting.

Completion of AD conversion triggers an INTAD AD conversion end interrupt request. Also, ADMOD0<EOCF> will be set to 1 to indicate that AD conversion has been completed.

1. Channel fixed single conversion mode

   Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to "00" selects conversion channel fixed single conversion mode.

   In this mode data on one specified channel is converted once only. When the conversion has been completed, the ADMOD0<EOCF> flag is set to 1, ADMOD0<ADBF> is cleared to "0", and an INTAD interrupt request is generated.

2. Channel scan single conversion mode

   Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to "01" selects conversion channel scan single conversion mode.

   In this mode data on the specified scan channels is converted once only. When scan conversion has been completed, ADMOD0<EOCF> is set to "1", ADMOD0<ADBF> is cleared to "0", and an INTAD interrupt request is generated.

3. Channel fixed repeat conversion mode

   Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to “10” selects conversion channel fixed repeat conversion mode.

   In this mode data on one specified channel is converted repeatedly. When conversion has been completed, ADMOD0<EOCF> is set to “1” and ADMOD0<ADBF> is not cleared to “0” but held at “1”. INTAD interrupt request generation timing is determined by the setting of ADMOD0<ITM0>.

   Clearing <ITM0> to “0” generates an interrupt request every time an AD conversion is completed.

   Setting <ITM0> to “1” generates an interrupt request on completion of every fourth conversion.

4. Channel scan repeat conversion mode

   Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to “11” selects conversion channel scan repeat conversion mode.

   In this mode data on the specified scan channels is converted repeatedly. When each scan conversion has been completed, ADMOD0<EOCF> is set to “1” and an INTAD interrupt request is generated. ADMOD0<ADBF> is not cleared to “0” but held at “1”.

   To stop conversion in a repeat conversion mode (e.g., in cases c and d), program a "0” to ADMOD0<REPEAT>. After the current conversion has been completed, the repeat conversion mode terminates and ADMOD0<ADBF> is cleared to “0”.

   Switching to a halt state (IDLE2 mode with ADMOD1<I2AD> cleared to “0”, IDLE1 mode or STOP mode) immediately stops operation of the AD converter even when AD conversion is still in progress. In repeat conversion modes (e.g., in cases c and d), when the halt is released, conversion restarts from the beginning. In single conversion modes (e.g., in cases a and b), conversion does not restart when the halt is released (The converter remains stopped).

   Table 3.11.2 shows the relationship between the AD conversion modes and interrupt requests.

Table 3.11.2 Relationship between the AD Conversion Modes and Interrupt Requests AD

| Mode | Interrupt Request Generation | ADMOD0 | | |
|---|---|---|---|---|
| | | <ITM0> | <REPEAT> | <SCAN> |
| Channel fixed single conversion mode | After completion of conversion | X | 0 | 0 |
| Channel scan single conversion mode | After completion of scan conversion | X | 0 | 1 |
| Channel fixed repeat conversion mode | Every conversion | 0 | 1 | 0 |
| | Every forth conversion | 1 | | |
| Channel scan repeat conversion mode | After completion of every scan conversion | X | 1 | 1 |

X: Don’t care

(5) AD conversion time

84 states (8.4 μs at $f_{SYS}$ = 20 MHz) are required for the AD conversion of one channel.

(6) Storing and reading the results of AD conversion

The AD conversion data upper and lower registers (ADREG0H/L to ADREG7H/L) store the results of AD conversion. (ADREG0H/L to ADREG7H/L are read-only registers.)

In channel fixed repeat conversion mode, the conversion results are stored successively in registers ADREG0H/L to ADREG3H/L. In other modes the AN0, AN1, AN2, AN3, AN4 AN5, AN6, AN7 conversion results are stored in ADREG0H/L, ADREG1H/L, ADREG2H/L, ADREG3H/L, ADREG4H/L, ADREG5H/L, ADREG6H/L, ADREG7H/L respectively.

Table 3.11.3 shows the correspondence between the analog input channels and the registers which are used to hold the results of AD conversion.

Table 3.11.3 Correspondence between Analog Input Channel and AD Conversion Result Register

| Analog Input Channel (Port G) | AD Conversion Result Register | |
|---|---|---|
| | Conversion Modes Other than at Right | Channel Fixed Repeat Conversion Mode (ADMOD0<ITM0>= "1") |
| AN0 | ADREG0H/L | ADREG0H/L → ADREG1H/L → ADREG2H/L → ADREG3H/L → ADREG0H/L |
| AN1 | ADREG1H/L | |
| AN2 | ADREG2H/L | |
| AN3 | ADREG3H/L | |
| AN4 | ADREG4H/L | |
| AN5 | ADREG5H/L | |
| AN6 | ADREG6H/L | |
| AN7 | ADREG7H/L | |

<ADRxRF>, bit0 of the AD conversion data lower register, is used as the AD conversion data storage flag. The storage flag indicates whether the AD conversion result register has been read or not. When a conversion result is stored in the AD conversion result register, the flag is set to "1". When either of the AD conversion result registers (ADREGxH or ADREGxL) is read, the flag is cleared to "0".

Reading the AD conversion result also clears the AD conversion end flag ADMOD0<EOCF> to "0".

Example:

1. Convert the analog input voltage on the AN3 pin and write the result, to memory address 0800H using the AD interrupt (INTAD) processing routine.

```
Setting of main routine
                    7  6  5  4  3  2  1  0
INTE0AD   ←         X  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 the analog input channel.
ADMOD0    ←         X  X  0  0  0  0  0  1    Start conversion in channel fixed single conversion
                                               mode.

Interrupt routine processing example
WA        ← ADREG3                            Read value of ADREG3L, ADREG3H to general
                                               purpose register WA (16 bits).
WA        > > 6                               Shift contents read into WA six times to right and zero-fill
                                               upper bits.
(0800H)   ← WA                                Write contents of WA to memory address 0800H.
```

2. Converts repeatedly the analog input voltages on the three pins AN0, AN1, and AN2, using channel scan repeat conversion mode.

```
INTE0AD   ←         X  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.

X : Don't care, – : No change
```

## 3.12 Watchdog Timer (Runaway detection timer)

The TMP92CM22 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.)

### 3.12.1 Configuration

Figure 3.12.1 is a block diagram of the watchdog timer (WDT).

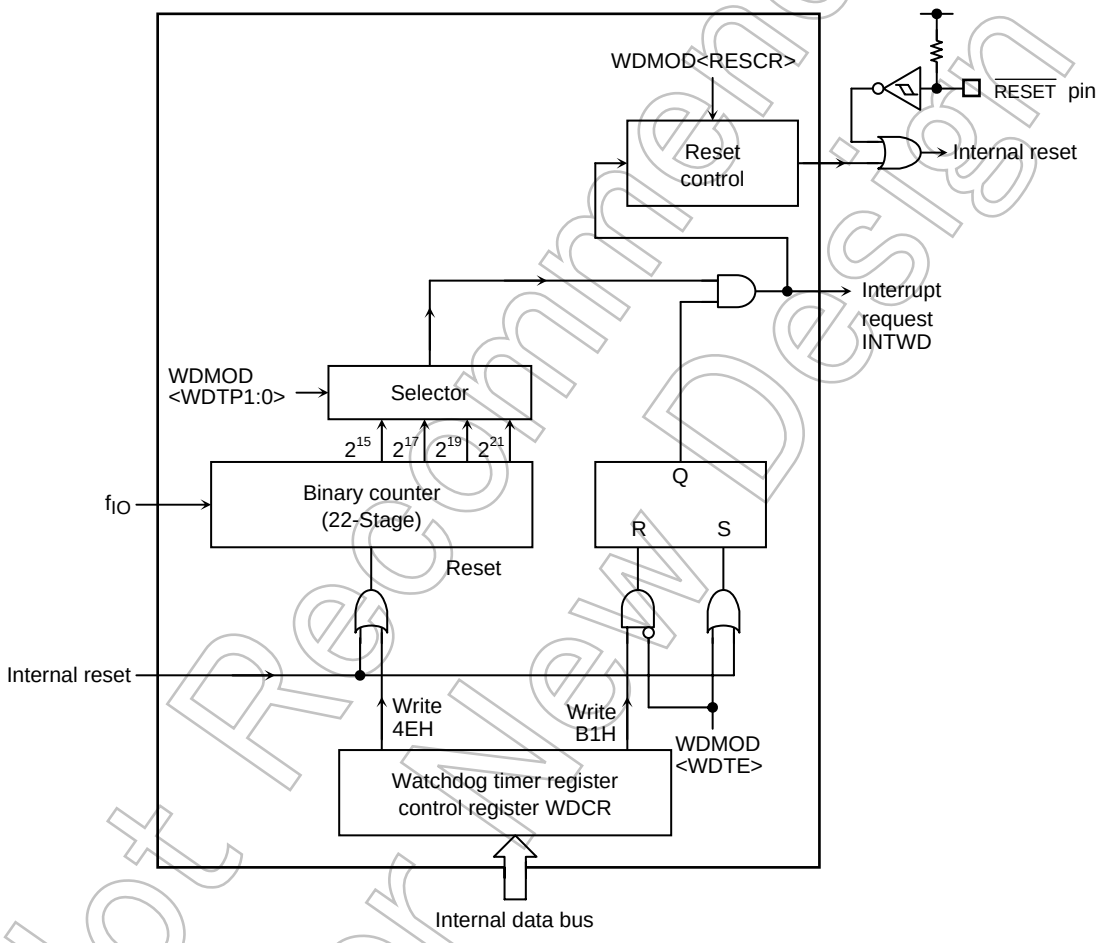



Figure 3.12.1 Block Diagram of Watchdog Timer

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.

### 3.12.2 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 $\phi$ ($2/f_{IO}$) as the input clock. The binary counter can output $2^{15}/f_{IO}$, $2^{17}/f_{IO}$, $2^{19}/f_{IO}$ and $2^{21}/f_{IO}$.

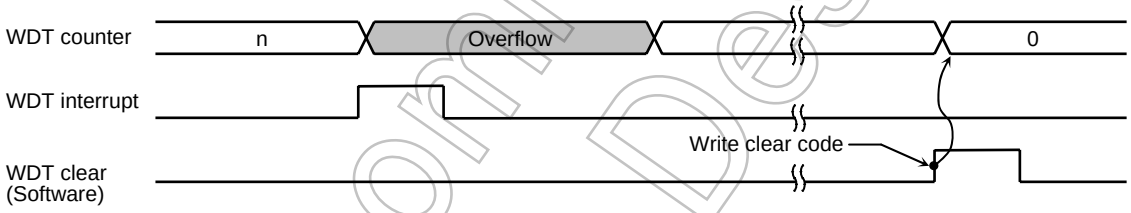



Figure 3.12.2 Normal Mode

The runaway detection result can also be connected to the reset pin internally.
In this case, the reset time will be between 22 and 29 system clocks (35.2 to 46.4 μs at $f_{OSCH}$ = 40 MHz) as shown inFigure 3.12.3. After a reset, the $f_{SYS}$ clock is $f_{FPH}/2$, where $f_{FPH}$ is generated by dividing the high-speed oscillator clock ($f_{OSCH}$) by sixteen through the clock gear function

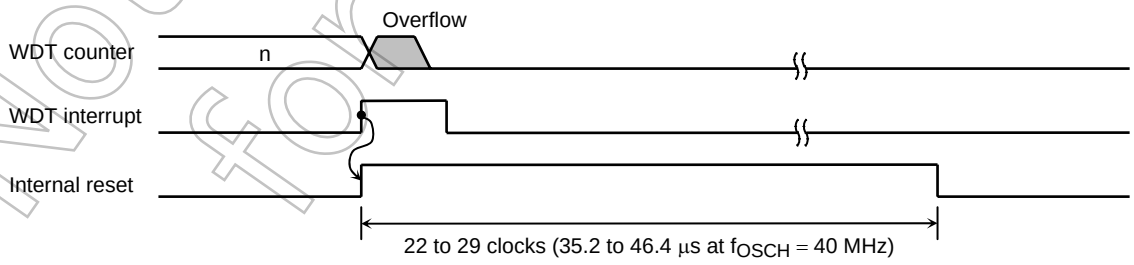



Figure 3.12.3 Reset Mode

### 3.12.3 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

(1) Watchdog timer mode register (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 $2^{15}/f_{SYS}$ [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 set 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 | – | – | – | 0 | – | – | 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, set the disable code (B1H) to WDCR after weirint the clear code (4EH) once. (Please refer to setting example.)

Note2: If the Watchdog timer setting, 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 | | | |
| After reset | 1 | 0 | 0 | | 0 | 0 | 0 | 0 |
| Function | WDT control<br>1: Enable | Select detecting time<br>00: $2^{15}/f_{IO}$<br>01: $2^{17}/f_{IO}$<br>10: $2^{19}/f_{IO}$<br>11: $2^{21}/f_{IO}$ | | | Always write “0” | IDLE2<br>0: Stop<br>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 3.28 ms at $f_{OSCH}$ = 40 MHz) |
| 01 | $2^{17}/f_{IO}$ (Approximately 1.31 ms at $f_{OSCH}$ = 40 MHz) |
| 10 | $2^{19}/f_{IO}$(Approximately 52.4 ms at $f_{OSCH}$ = 40 MHz) |
| 11 | $2^{21}/f_{IO}$ (Approximately 210 ms at $f_{OSCH}$ = 40 MHz) |

Watchdog timer enable/disable control

| | |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

Figure 3.12.4 Watchdog Timer Mode Register

WDCR (1302H)

Read -modify -write instruction is prohibited

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | | | | | | | |
| Read/Write | W | | | | | | | |
| After reset | – | | | | | | | |
| Function | B1H: WDT disable code<br>4EH: WDT clear code | | | | | | | |

WDT disable/clear control

| | |
|---|---|
| B1H | Disable code |
| 4EH | Clear code |
| Others | Don’t care |

Figure 3.12.5 Watchdog Timer Control Register

# 4. Electrical Characteristics

## 4.1 Absolute Maximum Ratings

| Parameter | Symbol | Rating | Unit |
|---|---|---|---|
| Power supply voltage | Vcc | −0.5 to 4.0 | V |
| Input voltage | VIN | −0.5 to Vcc + 0.5 | |
| Output current (1 pin) | IOL | 2 | mA |
| Output current (1 pin) | IOH | −2 | |
| Output current (Total) | ΣIOL | 80 | |
| Output current (Total) | ΣIOH | −80 | |
| Power dissipation (Ta = 85°C) | PD | 600 | mW |
| Soldering temperature (10 s) | TSOLDER | 260 | °C |
| Storage temperature | TSTG | −65 to 150 | |
| Operation temperature | TOPR | −40 to 85 | |

Note: 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, the 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 of lead-free products

| Test parameter | Test condition | Note |
|---|---|---|
| Solderability | (1) Use of Sn-37Pb solder Bath<br>Solder bath temperature =230°C, Dipping time = 5 seconds<br>The number of times = one, Use of R-type flux | Pass:<br>solderability rate until forming ≥ 95% |
| | (2) Use of Sn-3.0Ag-0.5Cu solder bath<br>Solder bath temperature =245°C, Dipping time = 5 seconds<br>The number of times = one, Use of R-type flux (use of lead-free) | |

DC Characteristics (1/2)

Vcc = 3.3 ± 0.3 V/fc = 4 to 40 MHz/Ta = –40 to 85°C

| Parameter | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Power supply voltage (DVCC = AVCC) (DVSS = AVSS = 0 V) | $V_{CC}$ | fc = 4 to 40 MHz ($f_{SYS}$ = 125 kHz to 20 MHz) | 3.0 | | 3.6 | V |
| Input low voltage P00 to P07 (D0 to D7) P10 to P17 (D8 to D15) | $V_{IL0}$ | | –0.3 | | 0.6 | V |
| Input low voltage P40 to P47 P50 to P57 P60 to P67 P76 PD2, PD3 PF0, PF3, PF6, PF7 PG0 to PG7 | $V_{IL1}$ | | | | 0.3 × VCC | |
| Input low voltage P90 to P92 PA0 to PA2, PA7 PC0, PC1, PC3, PC5, PC6 PD0, PD1 PF1, PF2, PF4, PF5 RESET, $\overline{NMI}$ | $V_{IL2}$ | | | | 0.25 × VCC | |
| Input low voltage AM0, AM1 | $V_{IL3}$ | | | | 0.3 | |
| Input low voltage X1 | $V_{IL4}$ | | | | 0.2 × VCC | |
| Input high voltage P00 to P07 (D0 to D7) P10 to P17 (D8 to D15) | $V_{IH0}$ | | 2.0 | | VCC + 0.3 | V |
| Input high voltage P40 to P47 P50 to P57 P60 to P67 P76 PD2, PD3 PF0, PF3, PF6, PF7 PG0 to PG7 | $V_{IH1}$ | | 0.7 × VCC | | | |
| Input high voltage P90 to P92 PA0 to PA2, PA7 PC0, PC1, PC3, PC5, PC6 PD0, PD1 PF1, PF2, PF4, PF5 RESET, $\overline{NMI}$ | $V_{IH2}$ | | 0.75 × VCC | | | |
| Input high voltage AM0, AM1 | $V_{IH3}$ | | VCC – 0.3 | | | |
| Input high voltage X1 | $V_{IH4}$ | | 0.8 × VCC | | | |

DC Characteristics (2/2)

Vcc = 3.3 ± 0.3 V/fc = 4 to 40 MHz/Ta = –40 to 85°C

| Parameter | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Output low voltage | $V_{OL}$ | IOL = 1.6 mA | | | 0.45 | V |
| Output high voltage | $V_{OH}$ | IOH = –400 μA | 2.4 | | | |
| Input leakage current | $I_{LI}$ | 0.0 ≤ Vin ≤ VCC | | 0.02 | 5 | μA |
| Output leakage current | $I_{LO}$ | 0.2 ≤ Vin ≤ VCC – 0.2 | | 0.05 | 10 | |
| Power down voltage (at STOP, RAM backup) | $V_{STOP}$ | VIL2 = 0.2 × Vcc, VIH2 = 0.8 × Vcc | 1.8 | | 3.6 | V |
| $\overline{\text{RESET}}$ pull-up resistor | $R_{RST}$ | | 100 | | 400 | kΩ |
| Programmable pull-up resistor | $R_{KH}$ | | | | | |
| Pin capacitance | $C_{IO}$ | fc = 1 MHz | | | 10 | pF |
| Schmitt width | $V_{TH}$ | P90 to P92<br>PA0 to PA2, PA7<br>PC0, PC1, PC3, PC5, PC6<br>PD0, PD1<br>PF1, PF2, PF4, PF5<br>$\overline{\text{RESET}}$ , $\overline{\text{NMI}}$ | 0.4 | 1.0 | | V |
| NORMAL | ICC | $V_{CC}$ = 3.6 V, X1 = 40 MHz (Internal 20 MHz) | | 30 | 42 | mA |
| IDLE2 mode | $ICC_{IDLE2}$ | | | 17 | 25 | |
| IDLE1 mode | $ICC_{IDLE1}$ | | | 3 | 5 | |
| STOP | $ICC_{STOP}$ | Vcc = 3.6 V | | 1 | 100 | μA |

## 4.2 AC Characteristics

### 4.2.1 Basis Bus Cycle

Read cycle

Vcc = 3.3 ± 0.3 V/fc = 4 to 40 MHz/Ta = –40 to 85°C

| No. | Parameter | Symbol | Min | Max | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | Unit |
|---|---|---|---|---|---|---|---|
| 1 | OSC period (X1/X2) | $t_{OSC}$ | 25 | 250 | 25 | 250 | ns |
| 2 | System clock period (= T) | $t_{CYC}$ | 50 | 8000 | 50 | 8000 | ns |
| 3 | CLKOUT low width | $t_{CL}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 4 | CLKOUT high width | $t_{CH}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 5-1 | A0 to A23 valid → D0 to D15 input at 0 waits | $t_{AD}$ | | 2.0T – 30 | 70 | 15970 | ns |
| 5-2 | A0 to A23 valid → D0 to D15 input at 1 wait | $t_{AD3}$ | | 3.0T – 30 | 120 | 23970 | ns |
| 6-1 | $\overline{RD}$ fall → D0 to D15 input at 0 waits | $t_{RD}$ | | 1.5T – 30 | 45 | 11970 | ns |
| 6-2 | $\overline{RD}$ fall → D0 to D15 input at 1 wait | $t_{RD3}$ | | 2.5T – 30 | 95 | 19970 | ns |
| 7-1 | $\overline{RD}$ low width at 0 waits | $t_{RR}$ | 1.5T – 20 | | 55 | 11980 | ns |
| 7-2 | $\overline{RD}$ low width at 1 wait | $t_{RR3}$ | 2.5T – 20 | | 105 | 19980 | ns |
| 8 | A0 to A23 valid → $\overline{RD}$ fall | $t_{AR}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 9 | $\overline{RD}$ fall → CLKOUT fall | $t_{RK}$ | 0.5T – 20 | | 5 | 3980 | ns |
| 10 | A0 to A23 valid → D0 to D15 hold | $t_{HA}$ | 0 | | 0 | 0 | ns |
| 11 | $\overline{RD}$ rise → D0 to D15 hold | $t_{HR}$ | 0 | | 0 | 0 | ns |
| 12 | $\overline{WAIT}$ setup time | $t_{TK}$ | 15 | | 15 | 15 | ns |
| 13 | $\overline{WAIT}$ hold time | $t_{KT}$ | 5 | | 5 | 5 | ns |

Write cycle

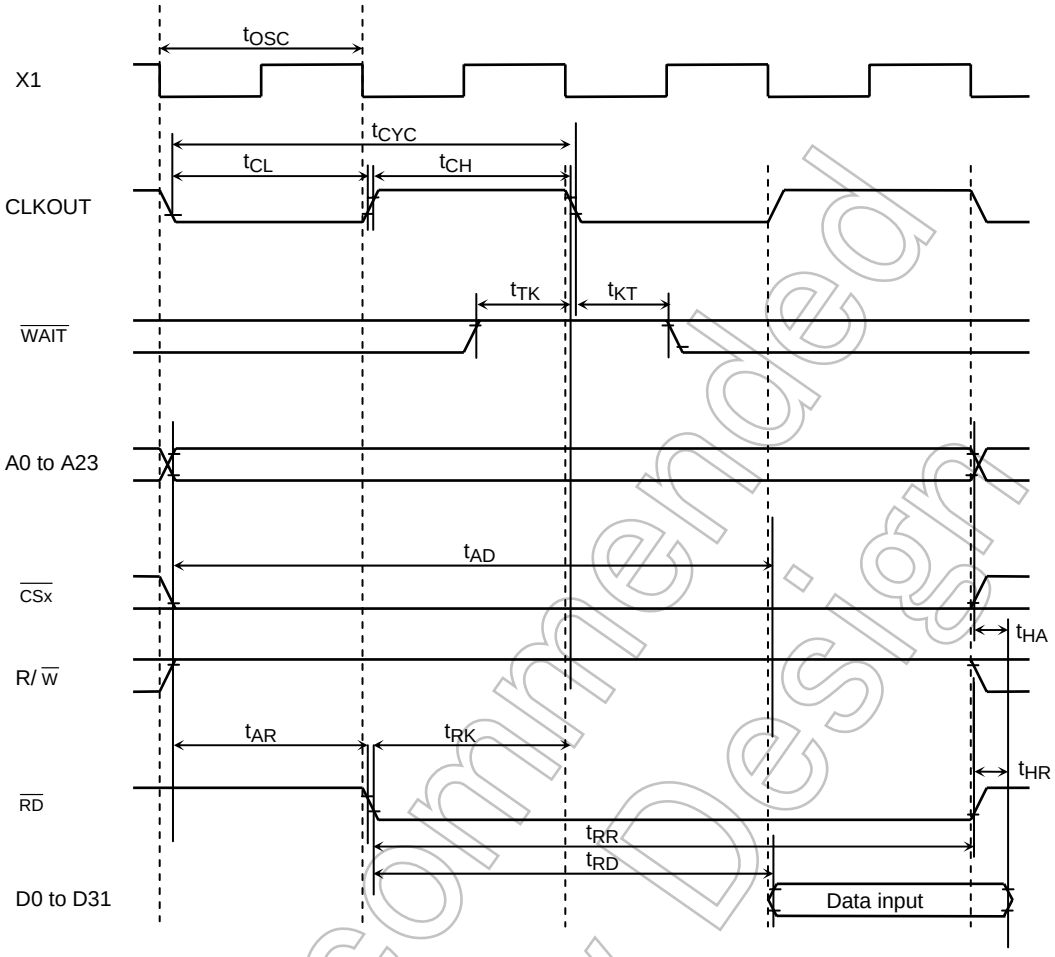
Vcc = 3.3 ± 0.3 V/fc = 4 to 40 MHz/Ta = –40 to 85°C

| No. | Parameter | Symbol | Min | Max | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | Unit |
|---|---|---|---|---|---|---|---|
| 1 | OSC period (X1/X2) | $t_{OSC}$ | 25 | 250 | 25 | 250 | ns |
| 2 | System clock period (= T) | $t_{CYC}$ | 50 | 8000 | 50 | 8000 | ns |
| 3 | CLKOUT low width | $t_{CL}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 4 | CLKOUT high width | $t_{CH}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 5-1 | D0 to D15 valid → $\overline{WRxx}$ rise at 0 waits | $t_{DW}$ | 1.25T – 35 | | 28 | 9965 | ns |
| 5-2 | D0 to D15 valid → $\overline{WRxx}$ rise at 1 wait | $t_{DW3}$ | 2.25T – 35 | | 78 | 17965 | ns |
| 6-1 | $\overline{WRxx}$ low width at 0 waits | $t_{WW}$ | 1.25T – 30 | | 33 | 9970 | ns |
| 6-2 | $\overline{WRxx}$ low width at 1 wait | $t_{WW3}$ | 2.25T – 30 | | 83 | 17970 | ns |
| 7 | A0 to A23 valid → $\overline{WR}$ fall | $t_{AW}$ | 0.5T – 15 | | 10 | 3985 | ns |
| 8 | $\overline{WRxx}$ fall → CLKOUT fall | $t_{WK}$ | 0.5T – 20 | | 5 | 3980 | ns |
| 9 | $\overline{WRxx}$ rise → A0 to A23 hold | $t_{WA}$ | 0.25T – 5 | | 8 | 1995 | ns |
| 10 | $\overline{WRxx}$ rise → D0 to D15 hold | $t_{WD}$ | 0.25T – 3 | | 10 | 1997 | ns |
| 11 | $\overline{WAIT}$ setup time | $t_{TK}$ | 15 | | 15 | 15 | ns |
| 12 | $\overline{WAIT}$ hold time | $t_{KT}$ | 5 | | 5 | 5 | ns |
| 13 | $\overline{RD}$ rise → D0 to D15 output | $t_{RDO}$ | 0.5T – 5 | | 20 | 3995 | ns |

AC condition

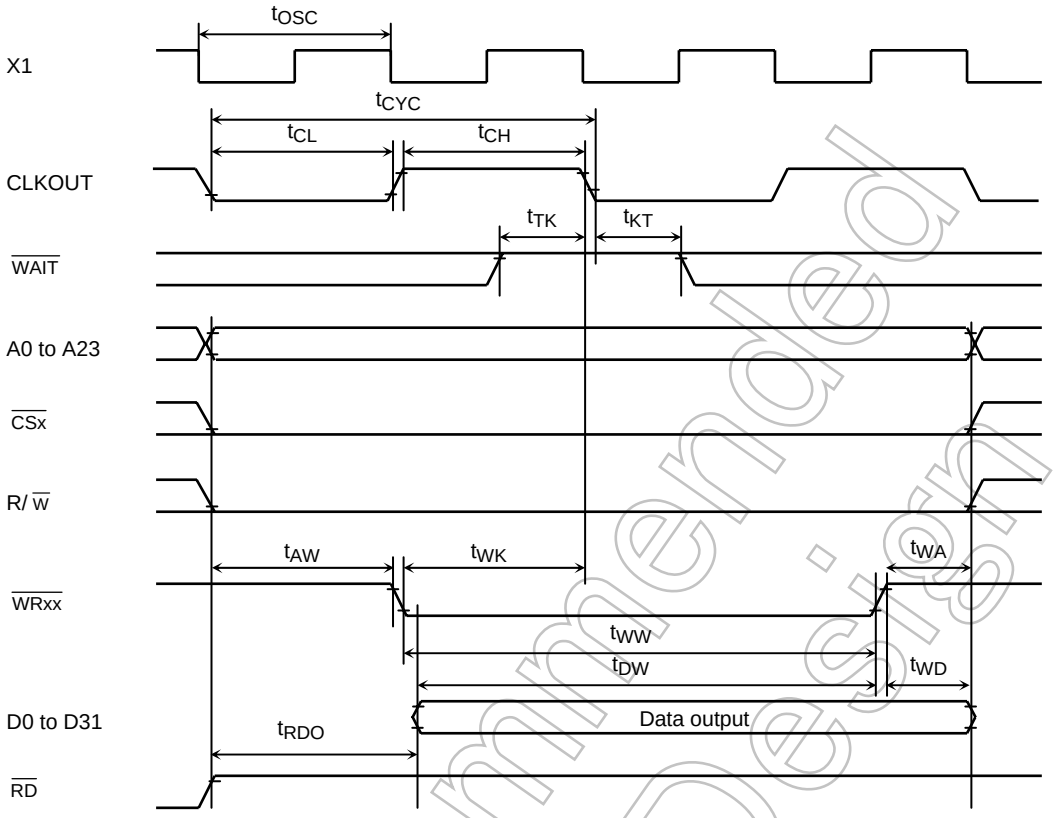
- Output : High = 0.7Vcc, Low = 0.3Vcc, $C_L$ = 50 pF
- Input : High = 0.9Vcc, Low = 0.1Vcc

(1) Read cycle (0 waits, fc = $f_{OSCH}$, $f_{FPH}$ = fc/1)



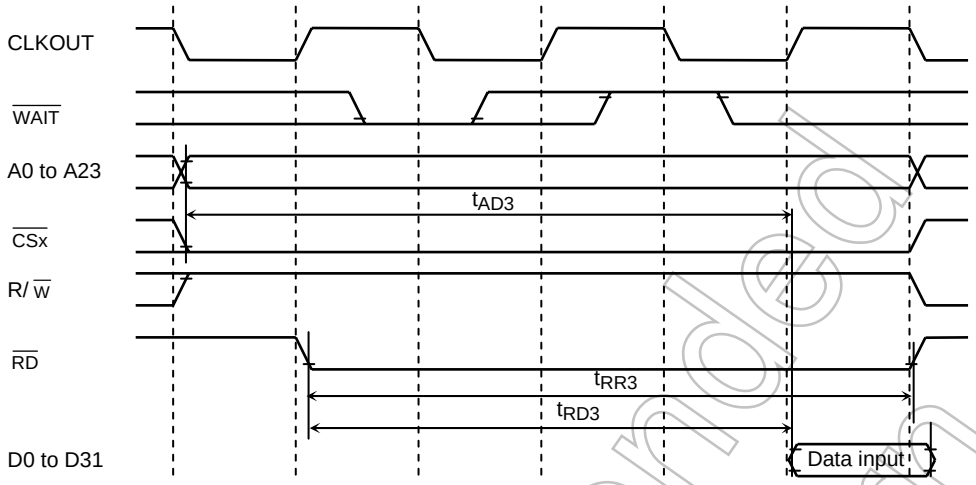

Note: The phase relation between X1 input signal and the other signals is unsettled. The timing chart above is an example.

(2) Write cycle (0 waits, fc = $f_{OSCH}$, $f_{FPH}$ = fc/1)

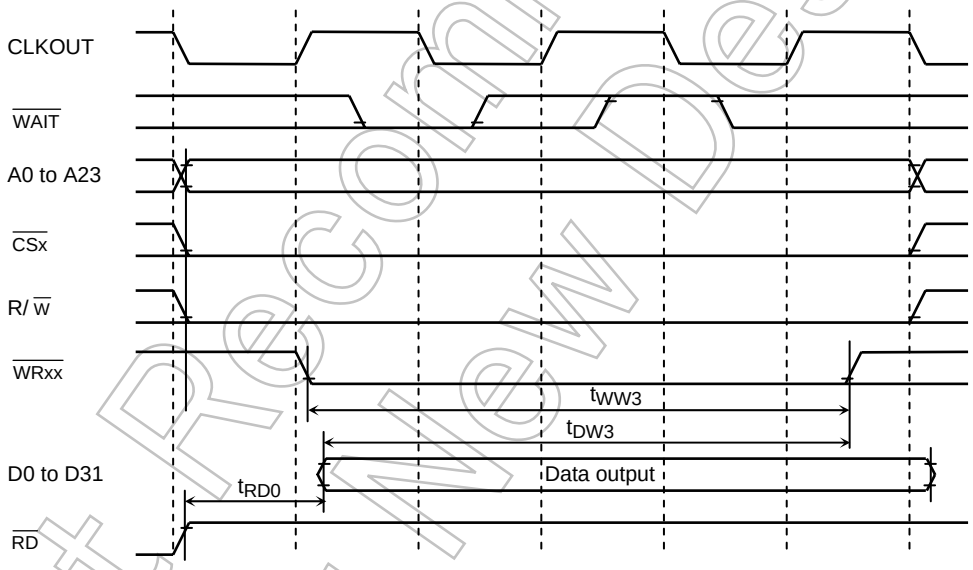



Note: The phase relation between X1 input signal and the other signals is unsettled.
The timing chart above is an example.

(3) Read cycle (1 wait)




(4) Write cycle (1 wait)



### 4.2.2 Page ROM Read Cycle

(1) 3-2-2-2 mode

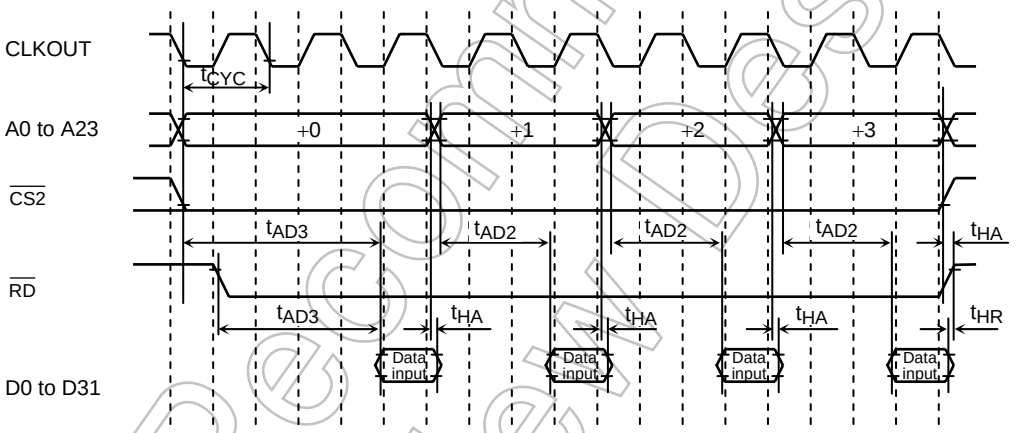
Vcc = 3.3 ± 0.3 V/fc = 4 to 40 MHz/Ta = –40 to 85°C

| No. | Parameter | Symbol | Min | Max | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | Unit |
|---|---|---|---|---|---|---|---|
| 1 | System clock period ( = T) | $t_{CYC}$ | 50 | 8000 | 50 | 8000 | ns |
| 2 | A0, A1 → D0 to D31 input | $t_{AD2}$ | | 2.0T – 50 | 50 | 15950 | ns |
| 3 | A2 to A23 → D0 to D31 input | $t_{AD3}$ | | 3.0T – 50 | 100 | 23950 | ns |
| 4 | RD falling → D0 to D31 input | $t_{RD3}$ | | 2.5T – 45 | 80 | 19955 | ns |
| 5 | A0 to A23 invalid → D0 to D31 hold | $t_{HA}$ | 0 | | 0 | 0 | ns |
| 6 | $\overline{RD}$ rising → D0 to D31 hold | $t_{HR}$ | 0 | | 0 | 0 | ns |

AC condition

- Output: High = 0.7 Vcc, Low = 0.3 Vcc, $C_L$ = 50 pF
- Input: High = 0.9 Vcc, Low = 0.1 Vcc

(2) Page ROM read cycle (3-2-2-2 mode)



## 4.3 AD Conversion Characteristics

| Parameter | Symbol | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|
| Analog reference voltage (+) | $V_{REFH}$ | VCC − 0.2 | VCC | VCC | V |
| Analog reference voltage (−) | $V_{REFL}$ | VSS | VSS | VSS + 0.2 | |
| AD converter power supply voltage | $A_{VCC}$ | VCC | VCC | VCC | |
| AD converter power supply ground | $A_{VSS}$ | VSS | VSS | VSS | |
| Analog input voltage | $A_{VIN}$ | VREFL | | VREFH | |
| Analog current for analog reference voltage <VREFON> = 1 | $I_{REF}$ | | 1.0 | 1.2 | mA |
| Analog current for analog reference voltage <VREFON> = 0 | | | 0.02 | 5.0 | UA |
| Total error (Include quantize error of ± 0.5 LSB ) | $E_T$ | | ±1.0 | ±4.0 | LSB |

## 4.4 Event Counter (TA0IN, TB1IN0, and TB1IN1)

| Parameter | Symbol | Variable | | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| Clock cycle | $T_{VCK}$ | 8X + 100 | | 500 | | 64100 | | ns |
| Low-level clock width | $T_{VCKL}$ | 4X + 40 | | 240 | | 32040 | | ns |
| High-level clock width | $T_{VCKH}$ | 4X + 40 | | 240 | | 32040 | | ns |

Note: Symbol “x” in the above table means the period of clock “$f_{SYS}$”, it’s same period of the system clock “$f_{SYS}$” for CPU core. The period of $f_{SYS}$ depends on the clock gear setting or changing high-speed oscillator/low-speed oscillator and so on.

## 4.5 Serial Channel Timing (I/O interface mode)

Note: Symbol "X" in the following table means the period of clock "$f_{SYS}$", it's same period of the system clock "$f_{SYS}$" for CPU core. The period of $f_{SYS}$ depends on the clock gear setting or changing high-speed oscillator/low-speed oscillator and so on.

(1) SCLK input mode

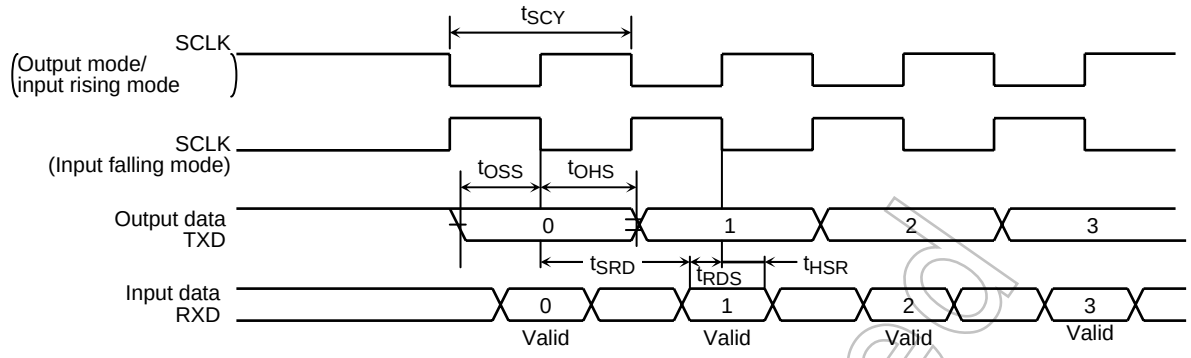
| Parameter | Symbol | Variable | | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| SCLK period | $t_{SCY}$ | 16X | | 0.8 | | 128 | | μs |
| Output data → SCLK rising/falling* | $t_{OSS}$ | $t_{SCY}/2 - 4X - 110$ | | 90 | | 31890 | | ns |
| SCLK rising/falling* → Output data hold | $t_{OHS}$ | $t_{SCY}/2 + 2X + 0$ | | 500 | | 80000 | | ns |
| SCLK rising/falling* → Input data hold | $t_{HSR}$ | $3X + 10$ | | 160 | | 24010 | | ns |
| SCLK rising/falling → Valid data input | $t_{SRD}$ | | $t_{SCY} - 0$ | | 800 | | 128000 | ns |
| Valid data input → SCLK rising/falling | $t_{RDS}$ | 0 | | 0 | | 0 | | ns |

*) SCLK rinsing/falling edge: The rising edge is used in SCLK rising mode.
The falling edge is used in SCLK falling mode.

Note: Value of $f_{SYS}$ = 20 MHz, 125 kHz is value if $t_{SCY}$ = 16X.

(2) SCLK output mode

| Parameter | Symbol | Variable | | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| SCLK period | $t_{SCY}$ | 16X | 8192X | 0.8 | 409.6 | 128 | 65536 | μs |
| Output data → SCLK rising/falling* | $t_{OSS}$ | $t_{SCY}/2 - 40$ | | 360 | | 3960 | | ns |
| SCLK rising/falling* → Output data hold | $t_{OHS}$ | $t_{SCY}/2 - 40$ | | 360 | | 3960 | | ns |
| SCLK rising/falling* → Input data hold | $t_{HSR}$ | 0 | | 0 | | 0 | | ns |
| SCLK rising/falling → Valid data input | $t_{SRD}$ | | $t_{SCY} - 1X - 180$ | | 409.4 | | 65528 | ns |
| Valid data input → SCLK rising/falling | $t_{RDS}$ | $1X + 180$ | | | 230 | | 8180 | ns |




## 4.6 Interrupt, Capture

Note: Symbol "X" in the following table means the period of clock "$f_{SYS}$", it's same period of the system clock "$f_{SYS}$" for CPU core. The period of $f_{SYS}$ depends on the clock gear setting or changing high-speed oscillator/low-speed oscillator and so on.

(1) $\overline{NMI}$ and INT0 to INT3 interrupts

| Parameter | Symbol | Variable | | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | | $f_{SYS}$ = 125 kHz (fc = 4 MHz) | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| INT0 to INT3 low width | $T_{INTAL}$ | 4X + 40 | | 240 | | 32040 | | ns |
| INT0 to INT3 high width | $T_{INTAH}$ | 4X + 40 | | 240 | | 32040 | | |

(2) INT4 to INT5 interrupts

| $t_{INTBL}$ (INT4 to INT5 Low Level Pulse Width) | | $t_{INTBH}$ (INT4 to INT5 High Level Pulse Width) | | Unit |
|---|---|---|---|---|
| Variable | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | Variable | $f_{SYS}$ = 20 MHz (fc = 40 MHz) | |
| Min | Min | Min | Min | |
| 8X + 100 | 500 | 8X + 100 | 500 | ns |

## 4.7 Recommended Oscillation Circuit

TMP92CM22 is evaluated by below oscillator vender. When selecting external parts, make use of this information.

Note 1: Total loads value of oscillation is sum of external (or internal) loads (C1 and C2) and floating loads of actual assemble board. There is a possibility of miss operating using C1 and C2 values in below table. When designing board, it should design minimum length pattern around oscillator. And we recommend that oscillator evaluation try on your actual using board.

Note 2: When use function of reduced drivability for high-frequency oscillator, must be used at $f_{OSCH}$ = 4 to 10 MHz.

(1) Example of oscillation connection circuit

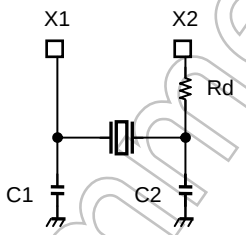



Figure 4.7.1 High-frequency Oscillator

(2) TMP92CM22 recommended ceramic oscillator: Murata Manufacturing Co., Ltd.

Following table shows circuit parameter recommended.

| IC Name | Oscillation Frequency [MHz] | Type | Item of Oscillator (Old number) | Parameter of Elements | | | | Running Condition | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | C1 [pF] | C2 [pF] | Rf [Ω] | Rd [Ω] | Voltage of Power [V] | Tc [°C] |
| TMP92CM22FG | 4.000 | SMD | CSTCR4M00G55-R0 (New and old is same product No.) | (39) | (39) | Open | 0 | 3.0 to 3.6 | −40 to +85 |
| | | Lead | CSTLS4M00G56-B0 (CSTS0400MG06) | (47) | (47) | Open | 0 | | |
| | 6.000 | SMD | CSTCR6M00G55-R0 (New and old is same product No.) | (39) | (39) | Open | 0 | | |
| | | Lead | CSTLS6M00G56-B0 (CSTS0600MG06) | (47) | (47) | Open | 0 | | |
| | 10.000 | SMD | CSTCE10M0G55-R0 (New and old is same product No.) | (33) | (10) | Open | 0 | | |
| | | Lead | CSTLS10M0G53-B0 (CSTS1000MG03) | (15) | (15) | Open | 0 | | |
| | 20.000 | SMD (New) | CSTCG20M0V51-R0 (New and old is same product No.) | (6) | (15) | Open | 0 | | |
| | | SMD | CSCTW20M0X51-R0 (CSTCW2000MX01) | (5) | (5) | Open | 0 | | |
| | 36.000 | SMD | CSTCW36M0X51-R0 (CSTCW3600MX01) | (6) | (6) | 15 k | 0 | | |
| | 40.000 | 2-pin SMD | CSACW40M0X51-R0) (CSACW4000MX01) | 3 | 3 | 15 k | 0 | | |

Note 1 : "()" of C1 and C2 are built in condenser type.

Note 2 : The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL:

http://www.murata.co.jp

# 5. Table of Special Function Registers (SFRs)

The SFRs include the I/O ports and peripheral control registers allocated to the 8 Kbytes address space from 000000H to 001FFFH.

(1) I/O port

(2) Interrupt controller

(3) DMA controller

(4) Memory controller

(5) Clock gear/PLL

(6) 8-bit timer

(7) 16-bit timer

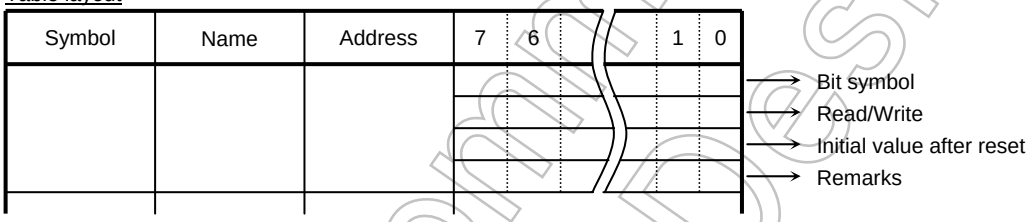
(8) UART/SIO

(9) I²C bus/SIO

(10) 10-bit ADC

(11) WDT

Table layout




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 as 1).

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 prohibit when controlling the pull-up resistor.

Table 5.1 I/O Register Address Map

[1] I/O port

| 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 | P4CR |
| 3H | P4FC |
| 4H | P5 |
| 5H | |
| 6H | P5CR |
| 7H | P5FC |
| 8H | P6 |
| 9H | |
| AH | P6CR |
| BH | P6FC |
| CH | P7 |
| DH | |
| EH | P7CR |
| FH | P7FC |

| Address | Name |
|---|---|
| 0020H | P8 |
| 1H | |
| 2H | |
| 3H | P8FC |
| 4H | P9 |
| 5H | P9ODE |
| 6H | P9CRP |
| 7H | 9FC |
| 8H | PA |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

| Address | Name |
|---|---|
| 0030H | PC |
| 1H | |
| 2H | PCCR |
| 3H | PCFC |
| 4H | PD |
| 5H | |
| 6H | PDCR |
| 7H | PDFC |
| 8H | |
| 9H | |
| AH | |
| BH | |
| CH | PF |
| DH | |
| EH | PFCR |
| FH | PFFC |

| Address | Name |
|---|---|
| 0040H | PG |
| 1H | |
| 2H | |
| 3H | |
| 4H | |
| 5H | |
| 6H | |
| 7H | |
| 8H | |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

Note: Do not access un-named addresses.

[2] Interrupt controller

| Address | Name |
|---|---|
| 00D0H | INTE12 |
| 1H | INTE3 |
| 2H | |
| 3H | |
| 4H | INTETA01 |
| 5H | INTETA23 |
| 6H | |
| 7H | |
| 8H | INTETB0 |
| 9H | |
| AH | INTETBO0 |
| BH | INTES0 |
| CH | INTES1 |
| DH | |
| EH | |
| FH | |

| Address | Name |
|---|---|
| 00E0H | INTE45 |
| 1H | INTETB1 |
| 2H | INTETBO1 |
| 3H | INTESB0 |
| 4H | |
| 5H | |
| 6H | |
| 7H | |
| 8H | |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | INTEP0 |
| FH | |

| Address | Name |
|---|---|
| 00F0H | INTE0AD |
| 1H | INTETC01 |
| 2H | INTETC23 |
| 3H | INTETC45 |
| 4H | INTETC67 |
| 5H | SIMC |
| 6H | IIMC |
| 7H | INTWDT |
| 8H | INTCLR |
| 9H | |
| AH | IIMC2 |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

[3] DMA controller

| Address | Name |
|---|---|
| 0100H | DMA0V |
| 1H | DMA1V |
| 2H | DMA2V |
| 3H | DMA3V |
| 4H | DMA4V |
| 5H | DMA5V |
| 6H | DMA6V |
| 7H | DMA7V |
| 8H | DMAB |
| 9H | DMAR |
| AH | Reserved |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

[4] Memory controller

| 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 | |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

[5] Clock gear/PLL

| Address | Name |
|---|---|
| 10E0H | SYSCR0 |
| 1H | SYSCR1 |
| 2H | SYSCR2 |
| 3H | EMCCR0 |
| 4H | EMCCR1 |
| 5H | EMCCR2 |
| 6H | Reserved |
| 7H | |
| 8H | PLLCR |
| 9H | Reserved |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

[6] 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 | |

[7] 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 | TB1FFCR |
| 4H | |
| 5H | |
| 6H | |
| 7H | |
| 8H | TB1RG0L |
| 9H | TB1RG0H |
| AH | TB1RG1L |
| BH | TB1RG1H |
| CH | TB1CP0L |
| DH | TB1CP0H |
| EH | TB1CP1L |
| FH | TB1CP1H |

[8] UART/SIO

| Address | Name |
|---|---|
| 1200H | SC0BUF |
| 1H | SC0CRS |
| 2H | C0MOD0 |
| 3H | BR0CR |
| 4H | BR0ADD |
| 5H | SC0MOD1 |
| 6H | |
| 7H | SIRCR |
| 8H | SC1BUF |
| 9H | SC1CR |
| AH | SC1MOD0 |
| BH | BR1CR |
| CH | BR1ADD |
| DH | SC1MOD1 |
| EH | |
| FH | |

[9] I²C bus/SIO

| Address | Name |
|---|---|
| 1240H | SBI0CR1 |
| 1H | SBI0DBR |
| 2H | I2C0AR |
| 3H | SBI0CR2/SBI0SR |
| 4H | SBI0BR0 |
| 5H | SBI0BR1 |
| 6H | |
| 7H | |
| 8H | |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

[10] 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 | ADREG6L |
| DH | ADREG6H |
| EH | ADREG7L |
| FH | ADREG7H |

| Address | Name |
|---|---|
| 12B0H | |
| 1H | |
| 2H | |
| 3H | |
| 4H | |
| 5H | |
| 6H | |
| 7H | |
| 8H | ADMOD0 |
| 9H | ADMOD1 |
| AH | ADMOD2 |
| BH | Reserved |
| CH | |
| DH | |
| EH | |
| FH | |

[11] WDT

| Address | Name |
|---|---|
| 1300H | WDMOD |
| 1H | WDCR |
| 2H | |
| 3H | |
| 4H | |
| 5H | |
| 6H | |
| 7H | |
| 8H | |
| 9H | |
| AH | |
| BH | |
| CH | |
| DH | |
| EH | |
| FH | |

(1) I/O port (1/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P1 | Port 1 | 0004H | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| | | | R/W | | | | | | | |
| | | | Data from external port (Output latch register is cleared to “0”) | | | | | | | |
| P4 | Port 4 | 0010H | P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |
| | | | R/W | | | | | | | |
| | | | Data from external port (Output latch register is cleared to “0”) | | | | | | | |
| P5 | Port 5 | 0014H | P57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 |
| | | | R/W | | | | | | | |
| | | | Data from external port (Output latch register is cleared to “0”) | | | | | | | |
| P6 | Port 6 | 0018H | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 |
| | | | R/W | | | | | | | |
| | | | Data from external port (Output latch register is cleared to “0”) | | | | | | | |
| P7 | Port 7 | 001CH | | P76 | P75 | P74 | P73 | P72 | P71 | P70 |
| | | | | R/W | | | | | | |
| | | | | Data from external port (Output latch register is cleared to “0”) | 1 | 1 | 1 | 1 | 1 | 1 |
| P8 | Port 8 | 0020H | | | | | P83 | P82 | P81 | P80 |
| | | | | | | | R/W | | | |
| | | | | | | | 1 | 0 | 1 | 1 |
| P9 | Port 9 | 0024H | | | | | | P92 | P91 | P90 |
| | | | | | | | | R/W | | |
| | | | | | | | | Data from external port (Output latch register is set to “1”) | | |
| PA | Port A | 0028H | PA7 | | | | | PA2 | PA1 | PA0 |
| | | | R | | | | | R | | |
| | | | Data from external port | | | | | Data from external port | | |
| PC | Port C | 0030H | | PC6 | PC5 | | PC3 | | PC1 | PC0 |
| | | | | R/W | | | R/W | | R/W | |
| | | | | Data from external port (Output latch register is set to “1”) | | | Data from external port (Output latch register is set to “1”) | | Data from external port (Output latch register is set to “1”) | |
| PD | Port D | 0034H | | | | | PD3 | PD2 | PD1 | PD0 |
| | | | | | | | R/W | | | |
| | | | | | | | Data from external port (Output latch register is set to “1”) | | | |
| PF | Port F | 003CH | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 |
| | | | R/W | | | | | | | |
| | | | Data from external port (Output latch register is set to “1”) | | | | | | | |
| PG | Port G | 0040H | PG7 | PG6 | PG5 | PG4 | PG3 | PG2 | PG1 | PG0 |
| | | | R | | | | | | | |
| | | | Data from external port | | | | | | | |

I/O port (2/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P1CR | Port 1 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 | Port 1 function register | 0007H (Prohibit RMW) | | | | | | | | P1F |
| | | | | | | | | | | W |
| | | | | | | | | | | 0/1 |
| | | | | | | | | | | 0: Port 1: Data bus (D8 to D15) |
| P4CR | Port 4 control register | 0012H (Prohibit RMW) | P47C | P46C | P45C | P44C | P43C | P42C | P41C | P40C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| P4FC | Port 4 function register | 0013H (Prohibit RMW) | P47F | P46F | P45F | P44F | P43F | P42F | P41F | P40F |
| | | | W | | | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | 0: Port 1: Address bus (A0 to A7) | | | | | | | |
| P5CR | Port 5 control register | 0016H (Prohibit RMW) | P57C | P56C | P55C | P54C | P53C | P52C | P51C | P50C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| P5FC | Port 5 function register | 0017H (Prohibit RMW) | P57F | P56F | P55F | P54F | P53F | P52F | P51F | P50F |
| | | | W | | | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | 0: Port 1: Address bus (A8 to A15) | | | | | | | |
| P6CR | Port 6 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 | Port 6 function register | 001BH (Prohibit RMW) | P67F | P66F | P65F | P64F | P63F | P62F | P61F | P60F |
| | | | W | | | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | 0: Port 1: Address bus (A16 to A23) | | | | | | | |
| P7CR | Port 7 control register | 001EH (Prohibit RMW) | | P76C | | | | | | |
| | | | | W | | | | | | |
| | | | | 0 | | | | | | |
| | | | | 0: Input 1: Output | | | | | | |
| P7FC | Port 7 function register | 001FH (Prohibit RMW) | | P76F | P75F | P74F | P73F | P72F | P71F | P70F |
| | | | | W | | | | | | |
| | | | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | 0: Port 1: $\overline{WAIT}$ | 0: Port 1: R/$\overline{W}$ | 0: Port 1: CLKOUT | 0: Port 1: Don't set. | 0: Port 1: $\overline{WRLU}$ | 0: Port 1: $\overline{WRLL}$ | 0: Port 1: $\overline{RD}$ |
| P8FC | Port 8 control register | 0023H (Prohibit RMW) | | | | | P83F | P82F | P81F | P80F |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | 0: Port 1: $\overline{CS3}$ | 0: Port 1: $\overline{CS2}$ | 0: Port 1: $\overline{CS1}$ | 0: Port 1: $\overline{CS0}$ |

I/O port (3/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P9CR | Port 9 control register | 0026H (Prohibit RMW) | | | | | | P92C | P91C | P90C |
| | | | | | | | | W | | |
| | | | | | | | | 0 | 0 | 0 |
| | | | | | | | | 0: Input 1: Output | | |
| P9FC | Port 9 function register | 0027H (Prohibit RMW) | | | | | | P92F | P91F | P90F |
| | | | | | | | | W | | |
| | | | | | | | | 0 | 0 | 0 |
| | | | | | | | | 0: Port, SI 1: SCL Note | 0: Port 1: SO, SDA | 0: Port, SCK input 1: SCK output Note |
| P9ODE | Port 9 ODE register | 0025H (Prohibit RMW) | | | | | | P92ODE | P91ODE | |
| | | | | | | | | W | | |
| | | | | | | | | 0 | 0 | |
| | | | | | | | | 1: Open drain | 1: Open drain | |
| PCCR | Port C control register | 0032H (Prohibit RMW) | | PC6C | PC5C | | PC3C | | PC1C | PC0C |
| | | | | W | | | W | | W | |
| | | | | 0 | 0 | | 0 | | 0 | 0 |
| | | | | 0: Input 1: Output | | | 0: Input 1: Output | | 0: Input 1: Output | |
| PCFC | Port C function register | 0033H (Prohibit RMW) | | PC6F | PC5F | | PC3F | | PC1F | PC0F |
| | | | | W | | | W | | W | |
| | | | | 0 | 0 | | 1 | | 0 | 0 |
| | | | | 0: Port 1: INT3 TB0OUT0 | 0: Port 1: INT2 TA3OUT | | 0: Port 1: INT0 | | 0: Port 1: INT1 TA1OUT | 0: Port 1: TA0IN |
| PDCR | Port D control register | 0036H (Prohibit RMW) | | | | | PD3C | PD2C | PD1C | PD0C |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | 0: Input 1: Output | 0: Input 1: Output | 0: Input 1: Output | 0: Input 1: Output |
| PDFC | Port D function register | 0037H (Prohibit RMW) | | | | | PD3F | PD2F | PD1F | PD0F |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | 0: Port 1: TB1OUT1 | 0: Port 1: TB1OUT0 | 0: Port 1: TB0IN1 INT5 input | 0: Port 1: TB0IN0 INT4 input |
| PFCR | Port F control register | 003EH (Prohibit RMW) | PF7C | PF6C | PF5C | PF4C | PF3C | PF2C | PF1C | PF0C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| PFFC | Port F function register | 003FH (Prohibit RMW) | – | – | PF5F | | PF3F | PF2F | | PF0F |
| | | | W | | | | W | | | W |
| | | | 0 | 0 | 0 | | 0 | 0 | | 0 |
| | | | Always write "0". | Always write "0". | 0: Port 1: SCLK1 output | | 0: Port 1: TXD1 | 0: Port 1: SCLK0 output | | 0: Port 1: TXD0 |

Note: When using SI and SCK input function, set P9FC<P92F,P90F> to "0" (Function setting).

(2) Interrupt control (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 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 |
| | | | 1: INT2 | Interrupt request level. | | | 1: INT1 | Interrupt request level | | |
| INTE3 | INT3 enable | 00D1H | – | | | | INT3 | | | |
| | | | – | – | – | – | I3C | I3M2 | I3M1 | I3M0 |
| | | | – | – | – | – | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Always write “0”. | | | | 1: INT3 | Interrupt request level | | |
| 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 |
| | | | 1: INTTA1 | Interrupt request level | | | 1: INTTA0 | Interrupt request level | | |
| 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 |
| | | | 1: INTTA3 | Interrupt request level | | | 1: INTTA2 | Interrupt request level | | |
| INTETB01 | INTTB00 & INTTB01 enable | 00D8H | INTTB1 (TMRB0) | | | | INTTB0 (TMRB0) | | | |
| | | | ITB1C | ITB1M2 | ITB1M1 | ITB1M0 | ITB0C | ITB0M2 | ITB0M1 | ITB0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INTTB1 | Interrupt request level | | | 1: INTTB0 | Interrupt request level | | |
| INTETBO0 | INTTBO0 (Overflow) enable | 00DAH | – | | | | INTTBO0 (TMRB0) | | | |
| | | | – | – | – | – | ITBO0C | ITBO0M2 | ITBO0M1 | ITBO0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | | | Always write “0”. | | | | 1: INTTBO0 | Interrupt request level | | |
| 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 |
| | | | 1: INTTX0 | Interrupt request level | | | 1: INTRX0 | Interrupt request level | | |
| 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 |
| | | | 1: INTTX1 | Interrupt request level | | | 1: INTRX1 | Interrupt request level | | |
| INTE45 | INT4 & INT5 enable | 00E0H | INT5 | | | | INT4 | | | |
| | | | I5C | I5M2 | I5M1 | I5M0 | I4C | I4M2 | I4M1 | I4M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INT5 | Interrupt request level | | | 1: INT4 | Interrupt request level | | |
| INTETB1 | INTB10 & INTTB11 enable | 00E1H | 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 |
| | | | 1: INTTB11 | Interrupt request level | | | 1: INTTB10 | Interrupt request level | | |
| INTETBO1 | INTTBO1 (Overflow) enable | 00E2H | – | | | | INTTBO1 (TMRB1) | | | |
| | | | – | – | – | – | ITBO1C | ITBO1M2 | ITBO1M1 | ITBO1M0 |
| | | | – | – | – | – | R | R/W | | |
| | | | Always write “0”. | | | | 0 | 0 | 0 | 0 |
| INTESB0 | INTSBE0 enable | 00E3H | – | | | | INTSBE0 | | | |
| | | | – | – | – | – | ISBE0C | ISBE0M2 | ISBE0M1 | ISBE0M0 |
| | | | – | – | – | – | R | R/W | | |
| | | | – | – | – | – | 0 | 0 | 0 | 0 |
| | | | Always write “0”. | | | | 1: INTSBE0 | Interrupt request level | | |

Interrupt control (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| INTEP0 | INTP0 enable | 00EEH | – | | | | INTP0 | | | |
| | | | – | – | – | – | IP0C | IP0M2 | IP0M1 | IP0M0 |
| | | | – | – | – | – | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Always write “0”. | | | | 1: INTP0 | Interrupt request level | | |
| INTE0AD | INT0 & INTAD enable | 00F0H | INTAD | | | | INT0 | | | |
| | | | IADC | IADM2 | IADM1 | IADM0 | I0C | I0M2 | I0M1 | I0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INTAD | Interrupt request level | | | 1: INT0 | Interrupt request level | | |
| INTETC01 | INTTC0 & INTTC1 enable | 00F1H | INTTC1 (DMA1) | | | | INTTC0 (DMA0) | | | |
| | | | ITC1C | ITC1M2 | ITC1M1 | ITC1M0 | ITC0C | ITC0M2 | ITC0M1 | ITC0M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INTTC1 | Interrupt request level | | | 1: INTTC0 | Interrupt request level | | |
| INTETC23 | INTTC2 & INTTC3 enable | 00F2H | INTTC3 (DMA3) | | | | INTTC2 (DMA2) | | | |
| | | | ITC3C | ITC3M2 | ITC3M1 | ITC3M0 | ITC2C | ITC2M2 | ITC2M1 | ITC2M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INTTC3 | Interrupt request level | | | 1: INTTC2 | Interrupt request level | | |
| INTETC45 | INTTC4 & INTTC5 enable | 00F3H | INTTC5 (DMA5) | | | | INTTC4 (DMA4) | | | |
| | | | ITC5C | ITC5M2 | ITC5M1 | ITC5M0 | ITC4C | ITC4M2 | ITC4M1 | ITC4M0 |
| | | | R | R/W | | | R | R/W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: INTTC5 | Interrupt request level | | | 1: INTTC4 | Interrupt request level | | |
| 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 |
| | | | 1: INTTC7 | Interrupt request level | | | 1: INTTC6 | Interrupt request level | | |
| SIMC | SIO interrupt mode control | 00F5H (Prohibit RMW) | | | | | | | IR1LE | IR0LE |
| | | | | | | | | | W | |
| | | | | | | | | | 1 | 1 |
| | | | | | | | | | 0: INTRX1 edge mode<br>1: INTRX1 level mode | 0: INTRX0 edge mode<br>1: INTRX0 level mode |
| IIMC | Interrupt input mode control | 00F6H (Prohibit RMW) | | | I3EDGE | I2EDGE | I1EDGE | I0EDGE | I0LE | NMIREE |
| | | | | | W | | | | | |
| | | | | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | INT3<br>0: Rising/ high<br>1: Falling/ low | INT2<br>0: Rising/ high<br>1: Falling/ low | INT1<br>0: Rising/ high<br>1: Falling/ low | INT0<br>0: Rising/ high<br>1: Falling/ low | INT0<br>0: Edge mode<br>1: Level mode | NMI<br>0: Falling<br>1: Falling and rising |
| INTWDT | INTWD enable | 00F7H | – | – | – | – | INTWD | – | – | – |
| | | | – | – | – | – | R | – | – | – |
| | | | – | – | – | – | 0 | – | – | – |
| | | | Always write “0”. | | | | 1: INTWD | | | |
| INTCLR | Interrupt clear control | 00F8H (Prohibit RMW) | | | CLRV5 | CLRV4 | CLRV3 | CLRV2 | CLRV1 | CLRV0 |
| | | | | | W | | | | | |
| | | | | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | Interrupt vector | | | | | |
| IIMC2 | Interrupt input mode control | 00FAH (Prohibit RMW) | | | | | I3LE | I2LE | I1LE | |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | |
| | | | | | | | INT3<br>0: Edge<br>1: Level | INT2<br>0: Edge<br>1: Level | INT1<br>0: Edge<br>1: Level | |

(3) DMA controller

| 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 | DREQ4 | DREQ3 | DREQ2 | DREQ1 | DREQ0 |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: DMA request in software | | | | | | | |

(4) Memory controller (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| B0CSL | Block 0 MEMC control register low | 0140H (Prohibit RMW) | | B0WW2 | B0WW1 | B0WW0 | | B0WR2 | B0WR1 | B0WR0 |
| | | | | W | | | | W | | |
| | | | | 0 | 1 | 0 | | 0 | 1 | 0 |
| | | | | Write waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | | | Read waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | |
| B0CSH | Block 0 MEMCT control register high | 0141H (Prohibit RMW) | B0E | – | – | B0REC | B0OM1 | B0OM0 | B0BUS1 | B0BUS0 |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | | | CS select<br>0: Disable<br>1: Enable | Always write "0". | Always write "0". | 0: No insert dummy cycle (Default)<br>1: Insert dummy cycle | 00: ROM/SRAM<br>01: Reserved<br>10: Reserved<br>11: Reserved | | Data bus width<br>00: 8 bits<br>01: 16 bits<br>10: Reserved<br>11: Reserved | |
| B1CSL | Block 1 MEMC control register low | 0144H (Prohibit RMW) | | B1WW2 | B1WW1 | B1WW0 | | B1WR2 | B1WR1 | B1WR0 |
| | | | | W | | | | W | | |
| | | | | 0 | 1 | 0 | | 0 | 1 | 0 |
| | | | | Write waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | | | Read waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | |
| B1CSH | Block 1 MEMC control register high | 0145H (Prohibit RMW) | B1E | – | – | B1REC | B1OM1 | B1OM0 | B1BUS1 | B1BUS0 |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | | | CS select<br>0: Disable<br>1: Enable | Always write "0". | Always write "0". | 0: No insert dummy cycle (Default)<br>1: Insert dummy | 00: ROM/SRAM<br>01: Reserved<br>10: Reserved<br>11: Reserved | | Data bus width<br>00: 8 bits<br>01: 16 bits<br>10: Reserved<br>11: Reserved | |
| B2CSL | Block 2 MEMC control register low | 0148H (Prohibit RMW) | | B2WW2 | B2WW1 | B2WW0 | | B2WR2 | B2WR1 | B2WR0 |
| | | | | W | | | | W | | |
| | | | | 0 | 1 | 0 | | 0 | 1 | 0 |
| | | | | Write waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | | | Read waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | |
| B2CSH | Block 2 MEMC control register high | 0149H (Prohibit RMW) | B2E | B2M | – | B2REC | B2OM1 | B2OM0 | B2BUS1 | B2BUS0 |
| | | | W | | | | | | | |
| | | | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | | | CS select<br>0: Disable<br>1: Enable | 0: 16 MB<br>1: Sets area | Always write "0". | 0: No insert dummy cycle (Default)<br>1: Insert dummy cycle | 00: ROM/SRAM<br>01: Reserved<br>10: Reserved<br>11: Reserved | | Data bus width<br>00: 8 bits<br>01: 16 bits<br>10: Reserved<br>11: Reserved | |
| B3CSL | Block 3 MEMC control register low | 014CH (Prohibit RMW) | | B3WW2 | B3WW1 | B3WW0 | | B3WR2 | B3WR1 | B3WR0 |
| | | | | W | | | | W | | |
| | | | | 0 | 1 | 0 | | 0 | 1 | 0 |
| | | | | Write waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | | | Read waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | |
| B3CSH | Block 3 MEMC control register high | 014DH (Prohibit RMW) | B3E | – | – | B3REC | B3OM1 | B3OM0 | B3BUS1 | B3BUS0 |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | | | CS select<br>0: Disable<br>1: Enable | Always write "0". | Always write "0". | 0: No insert dummy cycle (Default)<br>1: Insert dummy cycle | 00: ROM/SRAM<br>01: Reserved<br>10: Reserved<br>11: Reserved | | Data bus width<br>00: 8 bits<br>01: 16 bits<br>10: Reserved<br>11: Reserved | |

Memory controller (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| BEXCSL | Block EX MEMC control register low | 0158H (Prohibit RMW) | | BEXWW2 | BEXWW1 | BEXWW0 | | BEXWR2 | BEXWR1 | BEXWR0 |
| | | | | W | | | | W | | |
| | | | | 0 | 1 | 0 | | 0 | 1 | 0 |
| | | | | Write waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | | | Read waits<br>001: 0 waits 010: 1 wait<br>101: 2 waits 110: 3 waits<br>111: 4 waits 011: $\overline{\text{WAIT}}$ pin<br>Others: Reserved | | |
| BEXCSH | Block EX MEMC control register high | 0159H (Prohibit RMW) | | – | – | – | BEXOM1 | BEXOM0 | BEXBUS1 | BEXBUS0 |
| | | | | W | | | | | | |
| | | | | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | | | | Always write “0”. | Always write “0”. | Always write “0”. | 00: ROM/SRAM<br>01: Reserved<br>10: Reserved<br>11: Reserved | | Data bus width<br>00: 8 bits<br>01: 16 bits<br>10: Reserved<br>11: Reserved | |
| PMEMCR | Page ROM control register | 0166H | | | | OPGE | OPWR1 | OPWR0 | PR1 | PR0 |
| | | | | | | R/W | | | | |
| | | | | | | 0 | 0 | 0 | 1 | 0 |
| | | | | | | ROM page access<br>0: Disable<br>1: Enable | Wait number on page<br>00: 1 state (n-1-1-1 mode)<br>01: 2 states (n-2-2-2 mode)<br>10: 3 states (n-3-3-3 mode)<br>11: (Reserved) | | Byte number in a page<br>00: 64 bytes<br>01: 32 bytes<br>10: 16 bytes<br>11: 8 bytes | |
| MAMR0 | Memory 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 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 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 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 | | | | | | | |

(5) Clock gear

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SYSCR0 | System clock control 0 | 10E0H | – | | | | | – | | |
| | | | R/W | | | | | R/W | | |
| | | | 1 | | | | | 0 | | |
| | | | Always write “1”. | | | | | Always write “0”. | | |
| SYSCR1 | System clock control 1 | 10E1H | | | | | – | GEAR2 | GEAR1 | GEAR0 |
| | | | | | | | R/W | | | |
| | | | | | | | 0 | 1 | 0 | 0 |
| | | | | | | | Always write “0”. | Select gear value of high frequency (fc)<br>000: fc<br>001: fc/2<br>010: fc/4<br>011: fc/8<br>100: fc/16<br>101: (Reserved)<br>110: (Reserved)<br>111: (Reserved) | | |
| SYSCR2 | System clock control 2 | 10E2H | – | | WUPTM1 | WUPTM0 | HALTM1 | HALTM0 | SELDRV | DRVE |
| | | | R/W | | R/W | | | | | |
| | | | 0 | | 1 | 0 | 1 | 1 | 0 | 0 |
| | | | Always write “0”. | | Warm-up timer<br>00: Reserved<br>01: $2^8$/input frequency<br>10: $2^{14}$/input frequency<br>11: $2^{16}$/input frequency | | HALT mode<br>00: Reserved<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE2 mode | | <DRVE> mode select<br>0: STOP<br>1: IDLE1 | Pin state control in STOP mode<br>0: I/O off<br>1: Remains the state before halt |
| PLLCR | PLL control | 10E8H | PLLON | FCSEL | LWUPFG | | | | | |
| | | | R/W | | R | | | | | |
| | | | 0 | 0 | 0 | | | | | |
| | | | 0: PLL off<br>1: PLL on | fc select<br>0: OSCH<br>1: PLL (x4) | PLL warm-up flag<br>0: Don’t end warm up<br>1: End warm up | | | | | |
| EMCCR0 | EMC control register 0 | 10E3H | PROTECT | | | | | EXTIN | DRVOSCH | – |
| | | | R | | | | | R/W | | |
| | | | 0 | | | | | 0 | 1 | 1 |
| | | | Protect flag<br>0: OFF<br>1: ON | | | | | 1: External clock | fc oscillator driver ability<br>1: NORMAL<br>0: WEAK | Always write “1”. |
| EMCCR1 | EMC control register 1 | 10E4H | Switching the protect ON/OFF by write to following 1st-KEY, 2nd-KEY<br>1st-KEY: EMCCR1 = 5AH, EMCCR2 = A5H in succession write<br>2nd-KEY: EMCCR1 = A5H, EMCCR2 = 5AH in succession write | | | | | | | |
| EMCCR2 | EMC control register 2 | 10E5H | | | | | | | | |

(6) 8-bit timer

| 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 0: Stop and clear 1: Run (Count up) | UP counter (UC1) | UP counter (UC0) |
| TA0REG | 8-bit timer register 0 | 1102H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TA1REG | 8-bit timer register 1 | 1103H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| 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: $\phi$T1 10: $\phi$T16 11: $\phi$T256 | | Source clock for TMRA0 00: TA0IN pin 01: $\phi$T1 10: $\phi$T4 11: $\phi$T16 | |
| 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 0: Stop and clear 1: Run (Count up) | UP counter (UC3) | UP counter (UC2) |
| TA2REG | 8-bit timer register 2 | 110AH (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TA3REG | 8-bit timer register 3 | 110BH (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| 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: $\phi$T1 10: $\phi$T16 11: $\phi$T256 | | Source clock for TMRA2 00: Reserved 01: $\phi$T1 10: $\phi$T4 11: $\phi$T16 | |
| 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 |

(7) 16-bit timer (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| TB0RUN | Timer B0 RUN register | 1180H | TB0RDE | – | | | I2TB0 | TB0PRUN | | TB0RUN |
| | | | 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 0: Stop and clear 1: Run (Count up) | | UP counter (UC10) |
| TB0MOD | Timer B0 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 “0”. | Always write “0”. | Software capture control 0:Software capture 1:Undefined | Capture timing 00: Disable 01: Reserved 10: Reserved 11: TA1OUT↑ TA1OUT↓ | | Up counter control 0: Clear disable 1: Clear enable | Timer B0 source clock 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16 | |
| TB0FFCR | Timer B0 flip-flop control register | 1183H (Prohibit RMW) | – | – | TB0C1T1 | TB0C0T1 | TB0E1T1 | TB0E0T1 | TB0FFC1 | TB0FFC0 |
| | | | W* | | R/W | | | | W* | |
| | | | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| | | | Always write “11”. | | TB0FF0 inversion trigger 0: Trigger disable 1: Trigger enable | | | | Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don’t care * Always read as 11. | |
| | | | | | Invert when the UC10 value is loaded in to TB0CP1H/L. | Invert when the UC10 value is loaded in to TB0CP0H/L. | Invert when the UC10 matches with TB0RG1H/L. | Invert when the UC10 matches with TB0RG0H/L. | | |
| TB0RG0L | 16-bit timer register 0 low | 1188H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB0RG0H | 16-bit timer register 0 high | 1189H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB0RG1L | 16-bit timer register 1 low | 118AH (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB0RG1H | 16-bit timer register 1 high | 118BH (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| 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 | | | | | | | |

16-bit timer (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| TB1RUN | Timer B1 RUN register | 1190H | TB1RDE | – | | | I2TB0 | TB1PRUN | | TB1RUN |
| | | | 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 | Timer B1 mode register | 1192H (Prohibit RMW) | TB1CT1 | TB1ET1 | TB1CP0I | TB1CPM1 | TB1CPM0 | TB1CLE | TB1CLK1 | TB1CLK0 |
| | | | R/W | | W | R/W | | | | |
| | | | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | TB1FF1 inversion trigger 0: Disable trigger 1: Enable trigger | | Software capture control 0: Software capture 1: Undefined | Capture timing 00: Disable 01: TB1N0 ↑ TB1IN1 ↑ 10:TB1IN0 ↑ TB1IN0 ↓ 11: TA1OUT ↑ TA1OUT ↓ | | Up counter control 0: Clear disable 1: Clear enable | Timer B1 source clock 00: TB1IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16 | |
| | | | Invert when UC12 is loaded into TB1CP1H/L | Invert when UC12 matches with TB1RG1H/L | | | | | | |
| TB1FFCR | Timer B1 flip-flop control register | 1193H (Prohibit RMW) | TB1FF1C1 | TB1FF1C0 | TB1C1T1 | TB1C0T1 | TB1E1T1 | TB1E0T1 | TB1FFC1 | TB1FFC0 |
| | | | W* | | R/W | | | | W* | |
| | | | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| | | | Control TB1FF1 00: Invert 01: Set 10: Clear 11: Don’t care *Always read as 11. | | TB0FF0 inversion trigger 0: Disable trigger 1: Enable trigger | | | | Control TB1FF0 00: Invert 01: Set 10: Clear 11: Don’t care *Always read as 11. | |
| | | | | | Invert when the UC12 value is loaded in to TB1CP1H/L. | Invert when the UC12 value is loaded in to TB1CP0H/L. | Invert when the UC12 value matches the value in TB1RG1H/L. | Invert when the UC12 value matches the value in TB1RG0H/L. | | |
| TB1RG0L | 16-bit timer register 0 low | 1198H (Prohibit RMW) | — | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB1RG0H | 16-bit timer register 0 high | 1199H (Prohibit RMW) | — | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB1RG1L | 16-bit timer register 1 low | 119AH (Prohibit RMW) | — | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TB1RG1H | 16-bit timer register 1 high | 119BH (Prohibit RMW) | — | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| 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 | | | | | | | |

(8) UART/Serial channel (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(Receiving) / W(Transmission) | | | | | | | |
| | | | Undefined | | | | | | | |
| SC0CR | Serial channel 0 control register | 1201H | RB8 | EVEN | PE | OERR | PERR | FERR | SCLKS | IOC |
| | | | R | R/W | | R (Clear o after reading) | | | R/W | |
| | | | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Receive data bit8 | Parity 0: Odd 1: Even | Parity 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 |
| | | | Transmis-sion data bit8 | 0: CTS disable 1: CTS enable | 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: Timer TA0REG 01: Baud rate generator 10: Internal clock fio 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 divided 0: Disable 1: Enable | 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32 | | Divided frequency setting | | | |
| BR0ADD | Serial channel 0 K setting register | 1204H | | | | | BR0K3 | BR0K2 | BR0K1 | BR0K0 |
| | | | | | | | R/W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | Sets frequency divisor “K” (divided by N + (16 – K)/16). | | | |
| SC0MOD1 | Serial channel 0 mode 1 register | 1205H | I2S0 | FDPX0 | | | | | | |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | | | | | | |
| | | | IDLE2 0: Stop 1: Operate | I/O interface mode 0: Half duplex 1: Full duplex | | | | | | |
| SIRCR | IrDA control register | 1207H | PLSEL | RXSEL | TXEN | RXEN | SIRWD3 | SIRWD2 | SIRWD1 | SIRWD0 |
| | | | 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 effective pulse width for equal or more than 2x × (Value + 1) + 100 ns Can be set: 1 to 14 Can not be set: 0, 15 | | | |

UART/Serial channel (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 (Receiving)/W (Transmission) | | | | | | | |
| | | | Undefined | | | | | | | |
| SC1CR | Serial channel 1 control register | 1209H | RB8 | EVEN | PE | OERR | PERR | FERR | SCLKS | IOC |
| | | | R | R/W | | R (Clear 0 after reading) | | | R/W | |
| | | | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Receive data bit8 | Parity 0: Odd 1: Even | Parity 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 |
| | | | Transmis-sion data bit8 | 0: CTS disable 1: CTS enable | 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: Timer TA0REG 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 divided 0: Disable 1: Enable | 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32 | | Divided frequency setting | | | |
| BR1ADD | Serial channel 1 K setting register | 120CH | | | | | BR1K3 | BR1K2 | BR1K1 | BR1K0 |
| | | | | | | | R/W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | Sets frequency divisor “K” (divided by N + (16 – K)/16). | | | |
| SC1MOD1 | Serial channel 1 mode 1 register | 120DH | I2S1 | FDPX1 | | | | | | |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | | | | | | |
| | | | IDLE2 0: Stop 1: Operate | I/O interface mode 0: Half duplex 1: Full duplex | | | | | | |

(9) I²C bus/Serial channel (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SBI0CR1 | SBI0 control register 1 | 1240H (Prohibit RMW) I²C mode | BC2 | BC1 | BC0 | ACK | | SCK2 | SCK1 | SCK0/ SWRMON |
| | | | W | | | R/W | | W | | R/W |
| | | | 0 | 0 | 0 | 0 | | 0 | 0 | 0/1 |
| | | | Number of transfer bits 000: 8 001: 1 010: 2 011: 3 100: 4 101: 5 110: 6 111: 7 | | | Acknowledge mode 0: Disable 1: Enable | | Setting of the divide value "n" 000: 5 001: 6 010: 7 011: 8 100: 9 101: 10 110: 11 111: Reserved | | |
| | | 1240H (Prohibit RMW) SIO mode | SIOS | SIOINH | SIOM1 | SIOM0 | | SCK2 | SCK1 | SCK0 |
| | | | W | | | | | W | | |
| | | | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| | | | Transfer 0: Stop 1: Start | Transfer 0: Continue 1: Abort | Transfer mode 00: 8-bit transmit 10: 8-bit transmit/receive 11: 8-bit receive | | | Setting of the divide value "n" 000:4 001:5 010:6 011:7 100:8 101:9 110:10 111:External clock SCK0 | | |
| SBI0DBR | SBI0 buffer register | 1241H (Prohibit RMW) | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| | | | R (Receiving)/W (Transmission) | | | | | | | |
| | | | Undefined | | | | | | | |
| I2C0AR | I²C bus0 address register | 1242H (Prohibit RMW | SA6 | SA5 | SA4 | SA3 | SA2 | SA1 | SA0 | ALS |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Setting slave address | | | | | | | Address recognition 0 :Enable 1: Disable |
| SBI0SR when read | Serial bus interface status register | 1243H (I²C mode) (Prohibit RMW) | MST | TRX | BB | PIN | AL/SBIM1 | AAS/ SBIM0 | AD0/ SWRST | LARB/ SWRST0 |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| | | | 0: Slave 1: Master | 0: Receive 1: Transmit | Bus status monitor 0: Free 1: Busy | INTSBE0 request monitor 0: Request 1: Cancel | Arbitration lost detection monitor 0: – 1: Detect | Slave address match detection monitor 0: Undetect 1: Detect | GENERAL CALL detection monitor 0:Undetect 1: Detect | Last received bit monitor 0: 0 1: 1 |
| SBI0CR2 when write | Serial bus interface control register2 | | | | Start/stop condition generation 0: Start condition 1: Stop condition | | Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I²C bus mode 11: (Reserved) | | Software reset generate write "10" and "01", then an internal reset signal is generated. | |
| SBI0SR when read | Serial bus interface status register | 1243H (SIO mode) (Prohibit RMW) | | | | | SIOF/ SBIM1 | SEF/ SBIM2 | – | – |
| | | | | | | | R/W | | W | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | Transmit status monitor 0: Stopped 1: Terminated in progress | Shift operation status monitor 0: Stopped 1: Terminated in progress | | |
| SBI0CR2 when write | Serial bus interface control register2 | | | | | | Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I²C bus mode 11: (Reserved) | | Always write "0". | Always write "0". |

I²C bus/Serial channel (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SBI0BR0 | SBI0 baud rate register 0 | 1244H ($I^2C$ mode) (Prohibit RMW) | – | I2SBI0 | | | | | | |
| | | | W | R/W | | | | | | |
| | | | 0 | 0 | | | | | | |
| | | | Always write “0”. | IDLE2<br>0: Abort<br>1: Operate | | | | | | |
| | | 1244H (SIO mode) (Prohibit RMW) | – | – | | | | | | |
| | | | W | R/W | | | | | | |
| | | | 0 | 0 | | | | | | |
| | | | Always write “0”. | Always write “0”. | | | | | | |
| SBI0BR1 | SBI0 baud rate register 1 | 1245H | P4EN | – | | | | | | |
| | | | R/W | W | | | | | | |
| | | | 0 | 0 | | | | | | |
| | | | Clock control<br>0: Stop<br>1: Operate | Always write “0”. | | | | | | |

(10) AD converter (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| ADMOD0 | AD mode control register 0 | 12B8H | EOCF | ADBF | – | – | ITM0 | REPEAT | SCAN | ADS |
| | | | R | | R/W | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | AD conversion end flag 0: Busy 1: End | AD conversion busy flag 0: End 1: Busy | Always write "0". | Always write "0". | 0: Every 1 time 1: Every 4 times | Repeat mode 0: Single mode 1: Repeat mode | Scan mode 0: Fixed channel mode 1: Channel scan mode | AD conversion start 1: Start Always read as "0". |
| ADMOD1 | AD mode control register 1 | 12B9H | VREFON | I2AD | – | – | – | ADCH2 | ADCH1 | ADCH0 |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Ladder resistance 0: OFF 1: ON | IDLE2 0: Stop 1: Operate | Always write "0". | Always write "0". | Always write "0". | Input channel 000: AN0 AN0 001: AN1 AN0→AN1 010: AN2 AN0→AN1→AN2 011: AN3 AN0→AN1→AN2→AN3 100: AN4 AN0→AN1→AN2→AN3→AN4 101: AN5 AN0→AN1→AN2→AN3→AN4→AN5 110: AN6 AN0→AN1→AN2→AN3→AN4→AN5→AN6 111: AN7 AN0→AN1→AN2→AN3→AN4→AN5→AN6→AN7 | | |
| ADMOD2 | AD mode control register 2 | 12BAH | | | | | | | | ADTRG |
| | | | | | | | | | | R/W |
| | | | | | | | | | | 0 |
| | | | | | | | | | | AD external trigger start control 0: Disable 1: Enable |
| ADREG0L | AD result register 0 low | 12A0H | ADR01 | ADR00 | | | | | | ADR0RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG0H | AD result register 0 high | 12A1H | ADR09 | ADR08 | ADR07 | ADR06 | ADR05 | ADR04 | ADR03 | ADR02 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG1L | AD result register 1 low | 12A2H | ADR11 | ADR10 | | | | | | ADR1RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG1H | AD result register 1 high | 12A3H | ADR19 | ADR18 | ADR17 | ADR16 | ADR15 | ADR14 | ADR13 | ADR12 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG2L | AD result register 2 low | 12A4H | ADR21 | ADR20 | | | | | | ADR2RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG2H | AD result register 2 high | 12A5H | ADR29 | ADR28 | ADR27 | ADR26 | ADR25 | ADR24 | ADR23 | ADR22 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG3L | AD result register 3 low | 12A6H | ADR31 | ADR30 | | | | | | ADR3RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG3H | AD result register 3 high | 12A7H | ADR39 | ADR38 | ADR37 | ADR36 | ADR35 | ADR34 | ADR33 | ADR32 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |

AD converter (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| ADREG4L | AD result register 4 low | 12A8H | ADR41 | ADR40 | | | | | | ADR4RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG4H | AD result register 4 high | 12A9H | ADR49 | ADR48 | ADR47 | ADR46 | ADR45 | ADR44 | ADR43 | ADR42 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG5L | AD result register 5 Low | 12AAH | ADR51 | ADR50 | | | | | | ADR5RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG5H | AD result register 5 high | 12ABH | ADR59 | ADR58 | ADR57 | ADR56 | ADR55 | ADR54 | ADR53 | ADR52 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG6L | AD result register 6 low | 12ACH | ADR61 | ADR60 | | | | | | ADR6RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG6H | AD result register 6 high | 12ADH | ADR69 | ADR68 | ADR67 | ADR66 | ADR65 | ADR64 | ADR63 | ADR62 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| ADREG7L | AD result register 7 low | 12AEH | ADR71 | ADR70 | | | | | | ADR7RF |
| | | | R | | | | | | | R |
| | | | Undefined | | | | | | | 0 |
| ADREG7H | AD result register 7 high | 12AFH | ADR79 | ADR78 | ADR77 | ADR76 | ADR75 | ADR74 | ADR73 | ADR72 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |

(11) 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 | 0 |
| | | | WDT control<br>1: Enable | Select detecting time<br>00: $2^{15}/f_{IO}$<br>01: $2^{17}/f_{IO}$<br>10: $2^{19}/f_{IO}$<br>11: $2^{21}/f_{IO}$ | | | Always write “0”. | IDLE2<br>0: Stop<br>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 | | | | | | | |

# 6. Port Section Equivalent Circuit Diagram

■ Reading the circuit diagram

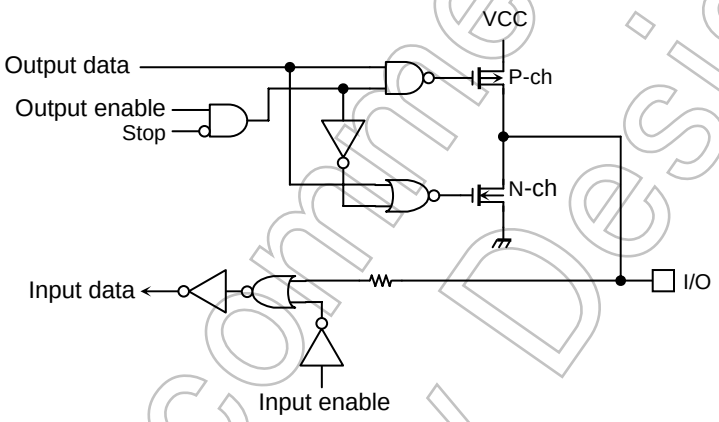
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The dedicated signal is described below.

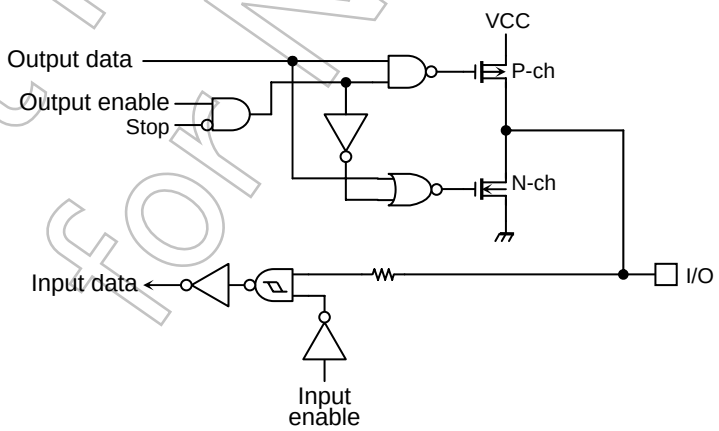
STOP: This signal becomes active "1" when the halt mode setting register is set to the STOP mode and the CPU executes the HALT instruction. When the drive enable bit <DRVE> is set to "1", however, STOP remains at "0".

The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

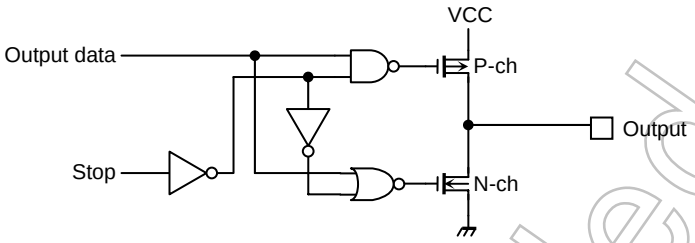
■ Data bus (D0 to D7), P1 (D8 to D15), P4 (A0 to A7), P5 (A8 to A15), P6 (A16 to A23), P76 ($\overline{\text{WAIT}}$), PD2 (TB1OUT0), PD3 (TB1OUT1), PF6, and PF7



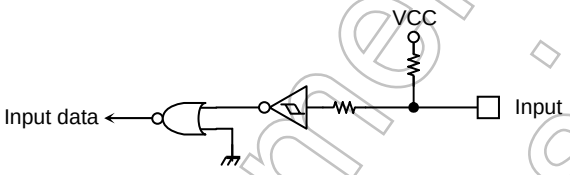

■ P90 (SCK), PC0 (TA0IN), PC1 (TA1OUT/INT1), PC3 (INT0), PC5 (TA3OUT/INT2), PC6 (TB0OUT/INT3), PD0 (INT4/TB1IN0), PD1 (INT5/TB1IN1), PF1 (RXD0), PF2 (SCLK0/$\overline{\text{CTS0}}$), PF4 (RXD1), and PF5 (SCLK1/$\overline{\text{CTS1}}$)



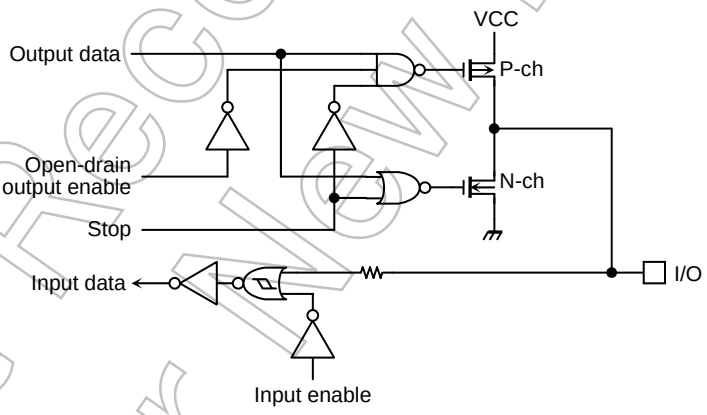
■ P70 ($\overline{RD}$), P71 ($\overline{WRLL}$), P72 ($\overline{WRLU}$), P73, P74 (CLKOUT), P75 (R/$\overline{W}$), P80 ($\overline{CS0}$), P81 ($\overline{CS1}$), P82 ($\overline{CS2}$), and P83 ($\overline{CS3}$)



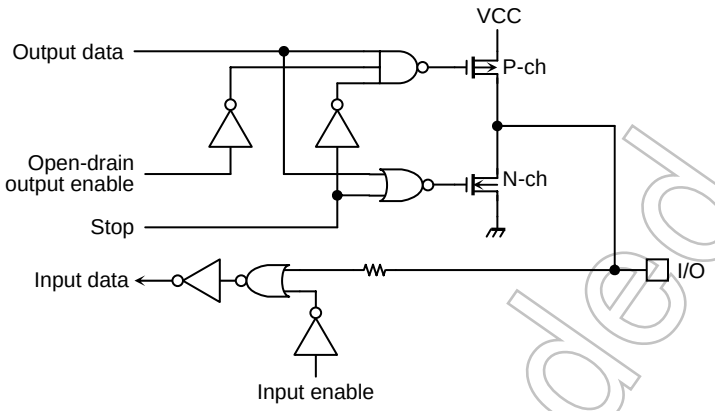

■ PA0, PA1, PA2, and PA7



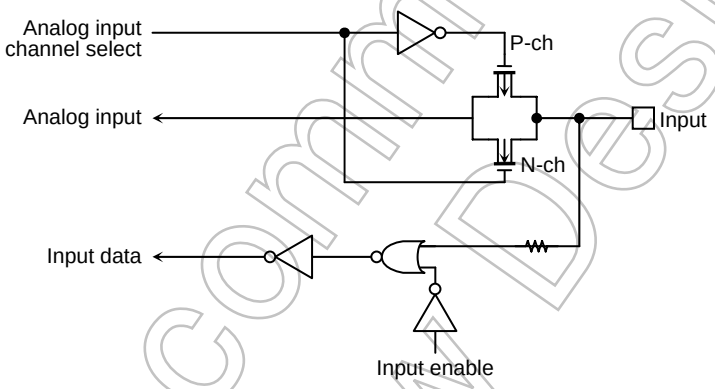

■ P91 (SO/SDA) and P92 (SI/SCL)



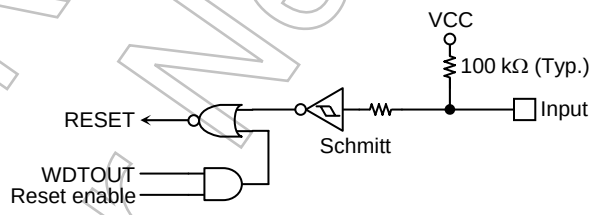
■ PF0 (TXD0) and PF3 (TXD1)



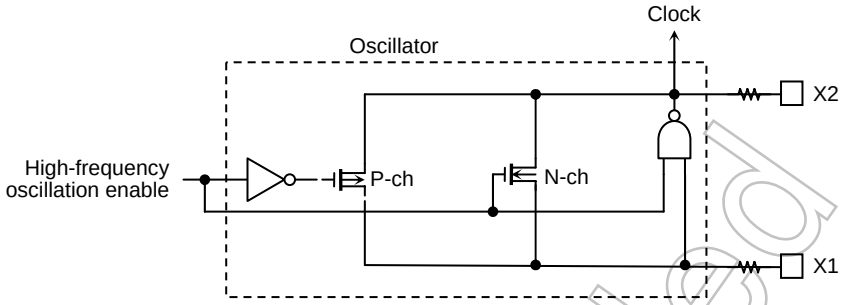

■ PG0 (AN0), PG1 (AN1), PG2 (AN2), PG3 (AN3/$\overline{\text{ADTRG}}$), PG4 (AN4), PG5 (AN5), PG6 (AN6), and PG7 (AN7)



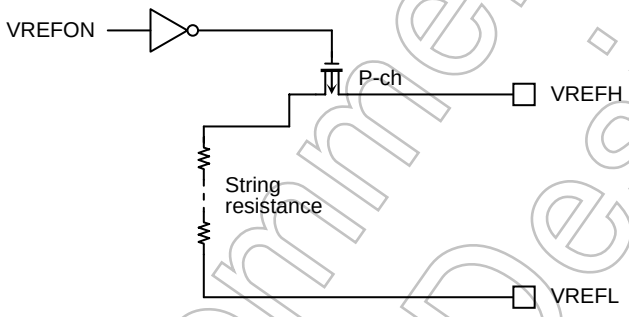

■ $\overline{\text{RESET}}$



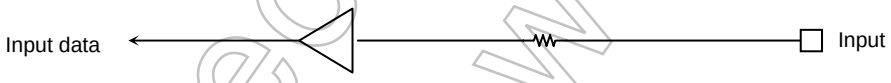
■ X1 and X2



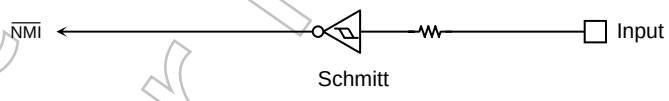

■ VREFH and VREFL




■ AM0 and AM1




■ $\overline{\text{NMI}}$



# 7. Points to Note and Restrictions

(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 that the CPU reads data from memory and writes the data to the same memory location by using 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, $f_{FPH}$, $f_{SYS}$, and one state

   The clock frequency that is inputted from X1 and X2 is called "$f_{OSCH}$". The clock that is selected by PLLCR<FCSEL> register is called "fc".

   The clock that selected by SYSCR1<SYSCK> is called "$f_{FPH}$". The clock frequency that is give by "$f_{FPH}$" divided by 2 is called "$f_{SYS}$".

   One cycle of "$f_{SYS}$" is referred to as one state.

(2) Points to note

a) AM0 and AM1 pins

This pin is connected to the VCC (Power supply level) or VSS (Ground level) pins. Do not alter the level when the pin is active.

b) Reservation area of address area

TMP92CM22 don't include reservation area.

c) Warm-up counter

The warm-up counter 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.

d) Watchdog timer

The watchdog timer starts operation immediately after a reset is released. When the watchdog timer is not to be used, disable it.

e) AD converter

The string resistor between the VREFH and VREFL pins can be cut by a program so as to reduce power consumption. When STOP mode is used, disable the resistor using the program before the HALT instruction is executed.

f) 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)).

g) Undefined SFR bit

The value of an undefined bit in an SFR (Special function register) is undefined when read.

h) POP SR instruction

Please execute the POP SR instruction during DI condition.

# 8. Package Dimensions

P-LQFP100-144-0.50F

Unit: mm