TOSHIBA

TOSHIBA Original CMOS 16-Bit Microcontroller

# TLCS-900/L Series

## TMP93CS40/41

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”.

Especially, take care below cautions.

****CAUTION****

**How to release the HALT mode**

Usually, interrupts can release all halts status. However, the interrupts = ($\overline{NMI}$, INT0), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of $f_{FPH}$) with IDLE1 or STOP mode (IDLE2 and RUN are not applicable to this case). (In this case, an interrupt request is kept on hold internally.)

If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

Low Voltage/Low Power CMOS 16-Bit Microcontrollers

# TMP93CS40F/TMP93CS41F
# TMP93CS40DF/TMP93CS41DF

## 1. Outline and Device Characteristics

The TMP93CS40/S41 are high-speed advanced 16-bit microcontrollers developed for controlling medium to large-scale equipment. The TMP93CS41 does not have a ROM; the TMP93CS40 has a built-in ROM. Otherwise, the devices function in the same way.

The TMP93CS40/S41F are housed in a 100-pin flat package.

The device characteristics are as follows:

(1) Original 16-bit CPU (900/L CPU)

- TLCS-90 instruction mnemonic upward compatible
- 16-Mbyte linear address space
- General-purpose registers, register bank system
- 16-bit multiplication/division and bit transfer/arithmetic instructions
- Micro DMA: 4 channels (1.6 μs/2 bytes at 20 MHz)

(2) Minimum instruction execution time: 200 ns at 20 MHz

(3) Internal RAM: 2 Kbytes
Internal ROM:

| | |
|---|---|
| TMP93CS40 | 64-Kbyte ROM |
| TMP93CS41 | None |

(4) External memory expansion

- Can be expanded up to 16 Mbytes (for both programs and data).
- AM8/$\overline{\text{AM16}}$ pin (selects the external data bus width)
- Can mix 8-/16-bit external data buses.
..... Dynamic bus sizing

(5) 8-bit timer: 2 channels

(6) 8-bit PWM timer: 2 channels

(7) 16-bit timer: 2 channels

(8) 4-bit pattern generator: 2 channels

(9) Serial interface: 2 channels

(10) 10-bit AD converter: 8 channels

(11) Watchdog timer

(12) Chip select/wait controller: 3 blocks

(13) Interrupt functions: 29

- 9 CPU interrupts .... SWI instruction, and illegal instruction
- 14 internal interrupts
- 6 external interrupts

7-level priority can be set.

(14) I/O ports
79 pins for TMP93CS40 and 61 pins for TMP93CS41

(15) Standby function: 4 HALT modes (RUN, IDLE2, IDLE1, STOP)

(16) Clock gear function

- Dual clock operation
- Clock gear: High-frequency clock can be varied from fc to fc/16.

(17) Wide operating voltage

- Vcc = 2.7 to 5.5 V

(18) Package

| Type No. | Package |
|---|---|
| TMP93CS40F<br>TMP93CS41F | P-QFP100-1414-0.50 |
| TMP93CS40DF<br>TMP93CS41DF | P-LQFP100-1414-0.50F |

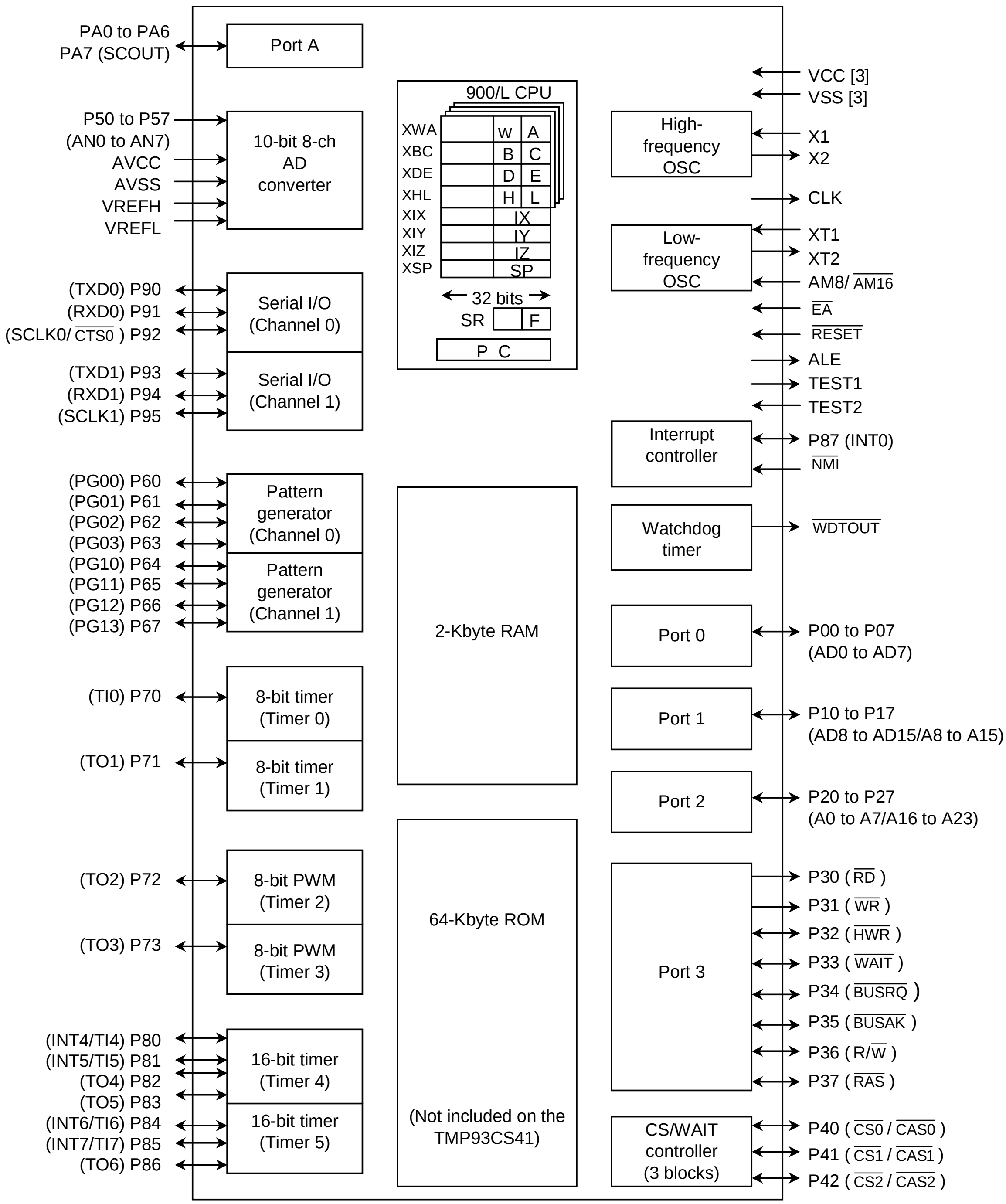


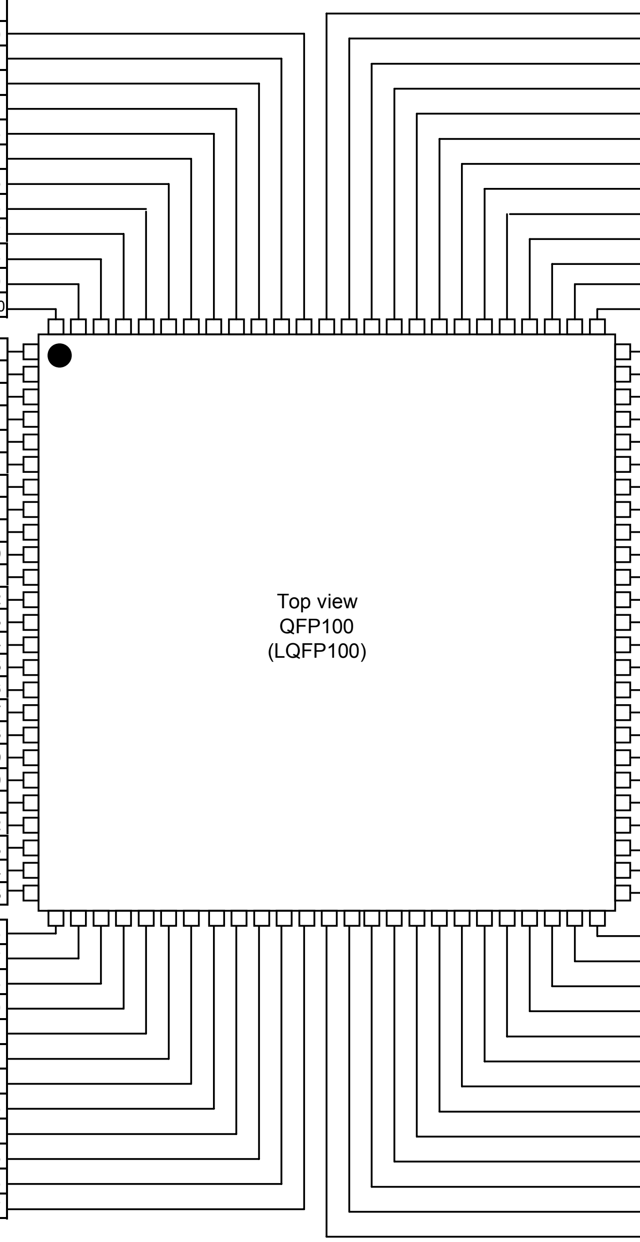

Figure 1.1 TMP93CS40/TMP93CS41 Block Diagram

# 2. Pin Assignment and Functions

The assignment of input/output pins on the TMP93CS40/TMP93CS41, their names and outline functions are described below.

## 2.1 Pin Assignment

Figure 2.1.1 shows the pin assignment for the TMP93CS40F/S41F and TMP93CS40DF/S41DF.




| Programmable Pull up | Programmable Pull down | TMP93CS40 | Pin no. |
|---|---|---|---|
| ○ | | P66/PG12 | 89 |
| ○ | | P67/PG13 | 90 |
| | | VSS | 91 |
| | | P50/AN0 | 92 |
| | | P51/AN1 | 93 |
| | | P52/AN2 | 94 |
| | | P53/AN3 | 95 |
| | | P54/AN4 | 96 |
| | | P55/AN5 | 97 |
| | | P56/AN6 | 98 |
| | | P57/AN7 | 99 |
| | | VREFH | 100 |
| | | VREFL | 1 |
| | | AVSS | 2 |
| | | AVCC | 3 |
| | | $\overline{\text{NMI}}$ | 4 |
| ○ | | P70/TI0 | 5 |
| ○ | | P71/TO1 | 6 |
| ○ | | P72/TO2 | 7 |
| ○ | | P73/TO3 | 8 |
| ○ | | P80/INT4/TI4 | 9 |
| ○ | | P81/INT5/TI5 | 10 |
| ○ | | P82/TO4 | 11 |
| ○ | | P83/TO5 | 12 |
| ○ | | P84/INT6/TI6 | 13 |
| ○ | | P85/INT7/TI7 | 14 |
| ○ | | P86/TO6 | 15 |
| ○ | | P87/INT0 | 16 |
| ○ | | P90/TXD0 | 17 |
| ○ | | P91/RXD0 | 18 |
| ○ | | P92/ $\overline{\text{CTS0}}$ /SCLK0 | 19 |
| ○ | | P93/TXD1 | 20 |
| ○ | | P94/RXD1 | 21 |
| ○ | | P95/SCLK1 | 22 |
| | | AM8/ $\overline{\text{AM16}}$ | 23 |
| | | CLK | 24 |
| | | VCC | 25 |
| | | VSS | 26 |
| | | X1 | 27 |
| | | X2 | 28 |
| | | $\overline{\text{EA}}$ | 29 |
| | | $\overline{\text{RESET}}$ | 30 |
| | | P96/XT1 | 31 |
| | | P97/XT2 | 32 |
| | | TEST1 | 33 |
| | | TEST2 | 34 |
| | | PA0 | 35 |
| | | PA1 | 36 |
| | | PA2 | 37 |

| Pin no. | TMP93CS40 | Programmable Pull down | Programmable Pull up |
|---|---|---|---|
| 88 | P65/PG11 | | ○ |
| 87 | P64/PG10 | | ○ |
| 86 | P63/PG03 | | ○ |
| 85 | P62/PG02 | | ○ |
| 84 | P61/PG01 | | ○ |
| 83 | P60/PG00 | | ○ |
| 82 | P42/ $\overline{\text{CS2}}$ / $\overline{\text{CAS2}}$ | ○ | |
| 81 | P41/ $\overline{\text{CS1}}$ / $\overline{\text{CAS1}}$ | | ○ |
| 80 | P40/ $\overline{\text{CS0}}$ / $\overline{\text{CAS0}}$ | | ○ |
| 79 | P37/ $\overline{\text{RAS}}$ | | ○ |
| 78 | P36/ R/ $\overline{\text{W}}$ | | ○ |
| 77 | P35/ $\overline{\text{BUSAK}}$ | | ○ |
| 76 | P34/ $\overline{\text{BUSRQ}}$ | | ○ |
| 75 | P33/ $\overline{\text{WAIT}}$ | | ○ |
| 74 | P32/ $\overline{\text{HWR}}$ | | ○ |
| 73 | P31/ $\overline{\text{WR}}$ | | |
| 72 | P30/ $\overline{\text{RD}}$ | | |
| 71 | P27/A7/A23 | ○ | |
| 70 | P26/A6/A22 | ○ | |
| 69 | P25/A5/A21 | ○ | |
| 68 | P24/A4/A20 | ○ | |
| 67 | P23/A3/A19 | ○ | |
| 66 | P22/A2/A18 | ○ | |
| 65 | P21/A1/A17 | ○ | |
| 64 | P20/A0/A16 | ○ | |
| 63 | VCC | | |
| 62 | VSS | | |
| 61 | $\overline{\text{WDTOUT}}$ | | |
| 60 | P17/AD15/A15 | | |
| 59 | P16/AD14/A14 | | |
| 58 | P15/AD13/A13 | | |
| 57 | P14/AD12/A12 | | |
| 56 | P13/AD11/A11 | | |
| 55 | P12/AD10/A10 | | |
| 54 | P11/AD9/A9 | | |
| 53 | P10/AD8/A8 | | |
| 52 | P07/AD7 | | |
| 51 | P06/AD6 | | |
| 50 | P05/AD5 | | |
| 49 | P04/AD4 | | |
| 48 | P03/AD3 | | |
| 47 | P02/AD2 | | |
| 46 | P01/AD1 | | |
| 45 | P00/AD0 | | |
| 44 | VCC | | |
| 43 | ALE | | |
| 42 | PA7/SCOUT | | |
| 41 | PA6 | | |
| 40 | PA5 | | |
| 39 | PA4 | | |
| 38 | PA3 | | |

Note: Because the TMP93CS41 does not have an internal ROM, pins P00 to P17 are tied to AD0 to AD15 (when AM8/ $\overline{\text{AM16}}$ = 0), or to AD0 to AD7 and A8 to A15 (when AM8/ $\overline{\text{AM16}}$ = 1). P30 is tied to $\overline{\text{RD}}$, P31 to $\overline{\text{WR}}$ .

Figure 2.1.1 Pin Assignment (100-Pin QFP and 100-Pin LQFP)

## 2.2 Pin Names and Functions

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

Table 2.2.1 to Table 2.2.4 show pin names and functions.

Table 2.2.1 Pin Names and Functions (1/4)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| P00 to P07<br>AD0 to AD7 | 8 | I/O<br>Tri-state | Port 0: I/O port that allows at the bit level<br>Address/data (lower): Bits 0 to 7 of address/data bus |
| P10 to P17<br>AD8 to AD15<br>A8 to A15 | 8 | I/O<br>Tri-state<br>Output | Port 1: I/O port that allows at the bit level<br>Address data (upper): Bits 8 to 15 of address/data bus<br>Address: Bits 8 to 15 of address bus |
| P20 to P27<br><br>A0 to A7<br>A16 to A23 | 8 | I/O<br><br>Output<br>Output | Port 2: I/O port that allows selection of I/O at the bit level<br>(with pull-down resistor)<br>Address: bits 0 to 7 of address bus<br>Address: bits 16 to 23 of address bus |
| P30<br>$\overline{\text{RD}}$ | 1 | Output<br>Output | Port 30: Output port<br>Read: Strobe signal for reading external memory |
| P31<br>$\overline{\text{WR}}$ | 1 | Output<br>Output | Port 31: Output port<br>Write: Strobe signal for writing data on pins AD0 to AD7 |
| P32<br>$\overline{\text{HWR}}$ | 1 | I/O<br>Output | Port 32: I/O port (with pull-up resistor)<br>High write: Strobe signal for writing data on pins AD8 to AD15 |
| P33<br>$\overline{\text{WAIT}}$ | 1 | I/O<br>Input | Port 33: I/O port (with pull-up resistor)<br>Wait: in used to request CPU bus wait |
| P34<br>$\overline{\text{BUSRQ}}$ | 1 | I/O<br>Input | Port 34: I/O port (with pull-up resistor)<br>Bus request: Signal used to request bus release |
| P35<br>$\overline{\text{BUSAK}}$ | 1 | I/O<br>Output | Port 35: I/O port (with pull-up resistor)<br>Bus acknowledge: Signal used to acknowledge bus release |
| P36<br>R/$\overline{\text{W}}$ | 1 | I/O<br>Output | Port 36: I/O port (with pull-up resistor)<br>Read/write: 1 represents read or dummy cycle; 0 represents write cycle. |
| P37<br>$\overline{\text{RAS}}$ | 1 | I/O<br>Output | Port 37: I/O port (with pull-up resistor)<br>Row address strobe: Outputs $\overline{\text{RAS}}$ strobe for DRAM. |
| P40<br>$\overline{\text{CS0}}$<br>$\overline{\text{CAS0}}$ | 1 | I/O<br>Output<br>Output | Port 40: I/O port (with pull-up resistor)<br>Chip select 0: Outputs 0 when address is within specified address area.<br>Column address strobe 0: Outputs $\overline{\text{CAS}}$ strobe for DRAM when address is within specified address area. |

Note: This device's built-in memory or built-in I/O cannot be accessed by an external DMA controller using the BUSRQ and BUSAK signals.

Table 2.2.2 Pin Names and Functions (2/4)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| P41<br>$\overline{CS1}$<br>$\overline{CAS1}$ | 1 | I/O<br>Output<br>Output | Port 41: I/O port (with pull-up resistor)<br>Chip select 1: Outputs 0 if address is within specified address area.<br>Column address strobe 1: Outputs $\overline{CAS}$ strobe for DRAM if address is within specified address area. |
| P42<br>$\overline{CS2}$<br>$\overline{CAS2}$ | 1 | I/O<br>Output<br>Output | Port 42: I/O port (with pull-down resistor)<br>Chip select 2: Outputs 0 if address is within specified address area.<br>Column address strobe 2: Outputs $\overline{CAS}$ strobe for DRAM if address is within specified address area. |
| P50 to P57<br>AN0 to AN7 | 8 | Input<br>Input | Port 5: Input port<br>Analog input: Analog signal input for AD converter |
| VREFH | 1 | Input | Pin for high level reference voltage input to AD converter |
| VREFL | 1 | Input | Pin for low level reference voltage input to AD converter |
| P60 to P63<br><br>PG00 to PG03 | 4 | I/O<br><br>Output | Ports 60 to 63: I/O ports that allow selection of I/O at the bit level<br>(with pull-up resistor)<br>Pattern generator ports: 00 to 03 |
| P64 to P67<br><br>PG10 to PG13 | 4 | I/O<br><br>Output | Ports 64 to 67: I/O ports that allow selection of I/O on a bit basis<br>(with pull-up resistor)<br>Pattern generator ports: 10 to 13 |
| P70<br>TI0 | 1 | I/O<br>Input | Port 70: I/O port (with pull-up resistor)<br>Timer input 0: Timer 0 input |
| P71<br>TO1 | 1 | I/O<br>Output | Port 71: I/O port (with pull-up resistor)<br>Timer output 1: Timer 0 or timer 1 output |
| P72<br>TO2 | 1 | I/O<br>Output | Port 72: I/O port (with pull-up resistor)<br>PWM output 2: 8-bit PWM timer 2 output |
| P73<br>TO3 | 1 | I/O<br>Output | Port 73: I/O port (with pull-up resistor)<br>PWM output 3: 8-bit PWM timer 3 output |
| P80<br>TI4<br>INT4 | 1 | I/O<br>Input<br>Input | Port 80: I/O port (with pull-up resistor)<br>Timer input 4: Timer 4 count/capture trigger signal input Interrupt request pin 4:<br>Interrupt request pin with programmable rising/falling edge |
| P81<br>TI5<br>INT5 | 1 | I/O<br>Input<br>Input | Port 81: I/O port (with pull-up resistor)<br>Timer input 5: Timer 5 count/capture trigger signal input<br>Interrupt request pin 5: Interrupt request pin with rising edge |
| P82<br>TO4 | 1 | I/O<br>Output | Port 82: I/O port (with pull-up resistor)<br>Timer output 4: Timer 4 output pin |
| P83<br>TO5 | 1 | I/O<br>Output | Port 83: I/O port (with pull-up resistor)<br>Timer output 5: Timer 4 output pin |

Table 2.2.3 Pin Names and Functions (3/4)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| P84<br>TI6<br>INT6 | 1 | I/O<br>Input<br>Input | Port 84: I/O port (with pull-up resistor)<br>Timer input 6: Timer 5 count/capture trigger signal input<br>Interrupt request pin 6: Interrupt request pin with programmable rising/falling edge |
| P85<br>TI7<br>INT7 | 1 | I/O<br>Input<br>Input | Port 85: I/O port (with pull-up resistor)<br>Timer input 7: Timer 5 count/capture trigger signal input<br>Interrupt request pin 7: Interrupt request pin with rising edge |
| P86<br>TO6 | 1 | I/O<br>Output | Port 86: I/O port (with pull-up resistor)<br>Timer output 6: Timer 5 output pin |
| P87<br>INT0 | 1 | I/O<br>Input | Port 87: I/O port (with pull-up resistor)<br>Interrupt request pin 0: Interrupt request pin with programmable level/rising edge |
| P90<br>TXD0 | 1 | I/O<br>Output | Port 90: I/O port (with pull-up resistor)<br>Serial data send 0 |
| P91<br>RXD0 | 1 | I/O<br>Input | Port 91: I/O port (with pull-up resistor)<br>Serial data receive 0 |
| P92<br>$\overline{\text{CTS0}}$<br>SCLK0 | 1 | I/O<br>Input<br>I/O | Port 92: I/O port (with pull-up resistor)<br>Serial data send enable 0 (Clear to send)<br>Serial Clock I/O 0 |
| P93<br>TXD1 | 1 | I/O<br>Output | Port 93: I/O port (with pull-up resistor)<br>Serial data send 1 |
| P94<br>RXD1 | 1 | I/O<br>Input | Port 94: I/O port (with pull-up resistor)<br>Serial data receive 1 |
| P95<br>SCLK1 | 1 | I/O<br>I/O | Port 95: I/O port (with pull-up resistor)<br>Serial clock I/O 1 |
| PA0 to PA6 | 7 | I/O | Ports A0 to A6: I/O ports |
| PA7<br>SCOUT | 1 | I/O<br>Output | Port A7: I/O port<br>System clock output: Outputs $f_{FPH}$ or $f_{SYS}$ clock. |
| $\overline{\text{WDTOUT}}$ | 1 | Output | Watchdog timer output pin |
| $\overline{\text{NMI}}$ | 1 | Input | Non-maskable interrupt request pin: Interrupt request pin with programmable falling edge or with both edges programmable. |
| CLK | 1 | Output | Clock output: Outputs [$f_{SYS}$ ÷ 2] clock.<br>Pulled-up during reset.<br>Can be disabled to reduce noise. |
| $\overline{\text{EA}}$ | 1 | Input | External access: On the TMP93CS41, the Vss pin should be connected.<br>On the TMP93CS40, the Vcc pin should be connected. |

Table 2.2.4 Pin Names and Functions (4/4)

| Pin Names | Number of Pins | I/O | Functions |
|---|---|---|---|
| AM8/ $\overline{\text{AM16}}$ | 1 | Input | Address mode: Selects external data bus width.<br>(On the TMP93CS40)<br>The Vcc pin should be connected. The data bus width for external access is set by the chip select/WAIT control register, port 1 control register.<br>(On the TMP93CS41)<br>The Vss pin should be connected to access either fixed 16-bit bus width, or 16-bit bus interchangeable with 8-bit bus.<br>The Vcc pin should be connected to access a fixed 8-bit bus width. |
| ALE | 1 | Output | Address latch enable<br>(Can be disabled to reduce noise.) |
| $\overline{\text{RESET}}$ | 1 | Input | Reset: Initializes TMP93CS40/TMP93CS41. (with pull-up resistor) |
| X1/X2 | 2 | I/O | High-frequency oscillator connecting pin |
| P96<br>XT1 | 1 | I/O<br>Input | Port 96: I/O port (open-drain output)<br>Low-frequency oscillator connecting pin |
| P97<br>XT2 | 1 | I/O<br>Output | Port 97: I/O port (open drain output)<br>Low-frequency oscillator connecting pin |
| TEST1/TEST2 | 2 | Output/Input | TEST1 pin should be connected to TEST2 pin.<br>Don't connect to any other pins. |
| VCC | 3 | | Power supply pin (All VCC pins should be connected to the power supply pin.) |
| VSS | 3 | | GND pin (0 V) (All VSS pins should be connected to GND (0 V).) |
| AVCC | 1 | | Power supply pin for AD converter |
| AVSS | 1 | | GND pin for AD converter (0 V) |

Note: All pins that have built-in pull-up/pull-down resistors (other than the $\overline{\text{RESET}}$ pin) can be disconnected from the built-in pull-up/pull-down resistor by software.

# 3. Operation

This section describes in blocks the functions and basic operations of TMP93CS40 and TMP93CS41 devices. Please also refer to section 7. Precautions in use, which describes some points requiring careful attention.

## 3.1 CPU

TMP93CS40 and TMP93CS41 devices have a built-in high-performance 16-bit CPU (900/L CPU). (For basics of the CPU operation, see the information on the TLCS-900/L CPU in the previous chapter.)

This section describes some CPU functions unique to the TMP93CS40 and TMP93CS41, that are not described in the previous chapter, entitled TLCS-900/L CPU.

### 3.1.1 Reset

When resetting the TMP93CS40 and TMP93CS41 microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then set the $\overline{\text{RESET}}$ input to low level at least for 10 system clocks (16 μs at 20 MHz). Thus, when turn on the switch, be set to the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to low level at least for 10 system clocks.

Clock gear is initialized 1/16 mode by reset operation. It means that the system clock mode $f_{SYS}$ is set to fc/32 (= fc/16 × 1/2).

When a reset signal is accepted, the CPU sets itself as follows:

- The program counter (PC) is set according to the reset vector that is stored from 8000H to 8002H.

  PC<7:0> ← Data in location 8000H
  PC<15:8> ← Data in location 8001H
  PC<23:16> ← Data in location 8002H

- The stack pointer (XSP) for system mode is set to 100H.
- The <IFF2:0> bits of the status register SR are set to 111. (Sets mask register to interrupt level 7.)
- The <MAX> bit of SR is set to 1. (Sets to maximum mode. See previous chapter.)
- The <RFP2:0> bits of SR are set to 000. (Clears register banks to 0.)

When the reset is released, instruction execution starts from PC (The reset vector). The reset makes no changes in any CPU internal registers other than those specifically mentioned above.

When a reset is received, signal and data processing for built-in I/Os, ports, and other pins is affected as follows:

- Initializes built-in I/O registers as per specifications.
- Sets port pins (including pins also used as built-in I/Os) to general-purpose input/output port mode.
- Sets the $\overline{\text{WDTOUT}}$ pin to 0. (The watchdog timer is set to enable after reset.)
- Pulls up the CLK pin to 1.
- Sets the ALE pin to 0 in the case of the TMP93CS41, and to high impedance (High-Z) in the case of TMP93CS40.

Note 1: Resetting makes no change in any register in the CPU except the program counter (PC), status register (SR) and stack pointer (XSP), nor in the data in the internal RAM.

Note 2: The CLK pin is pulled up during reset. When the voltage is externally reduced, there is a possibility of causing malfunctions.

Figure 3.1.1 and Figure 3.1.2 show the reset timing chart of the TMP93CS41 and TMP93CS40.

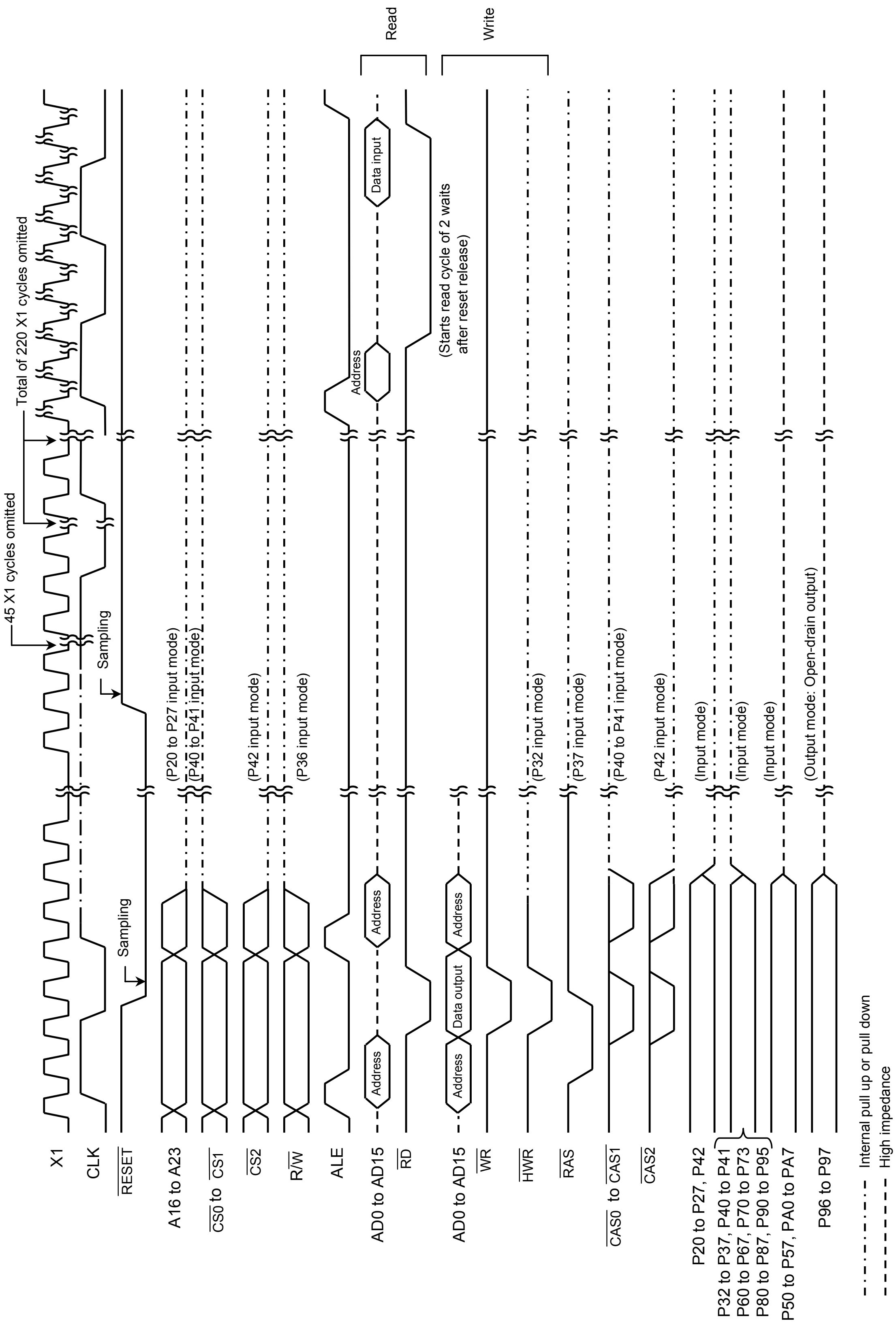



Figure 3.1.1 TMP93CS41 Reset Timing Chart

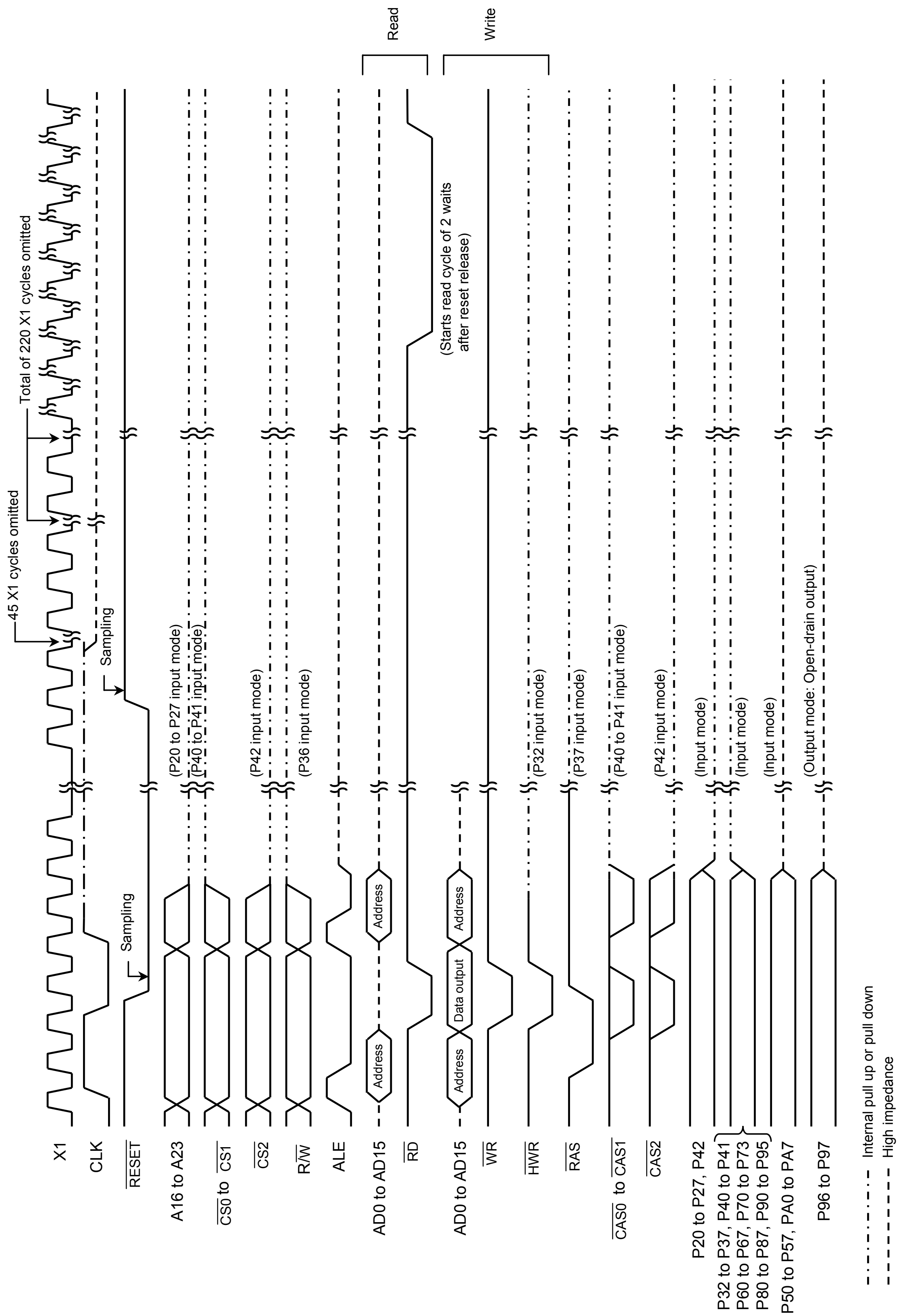



Figure 3.1.2 TMP93CS40 Reset Timing Chart

### 3.1.2 AM8/$\overline{\text{AM16}}$ Pin

(1) TMP93CS40

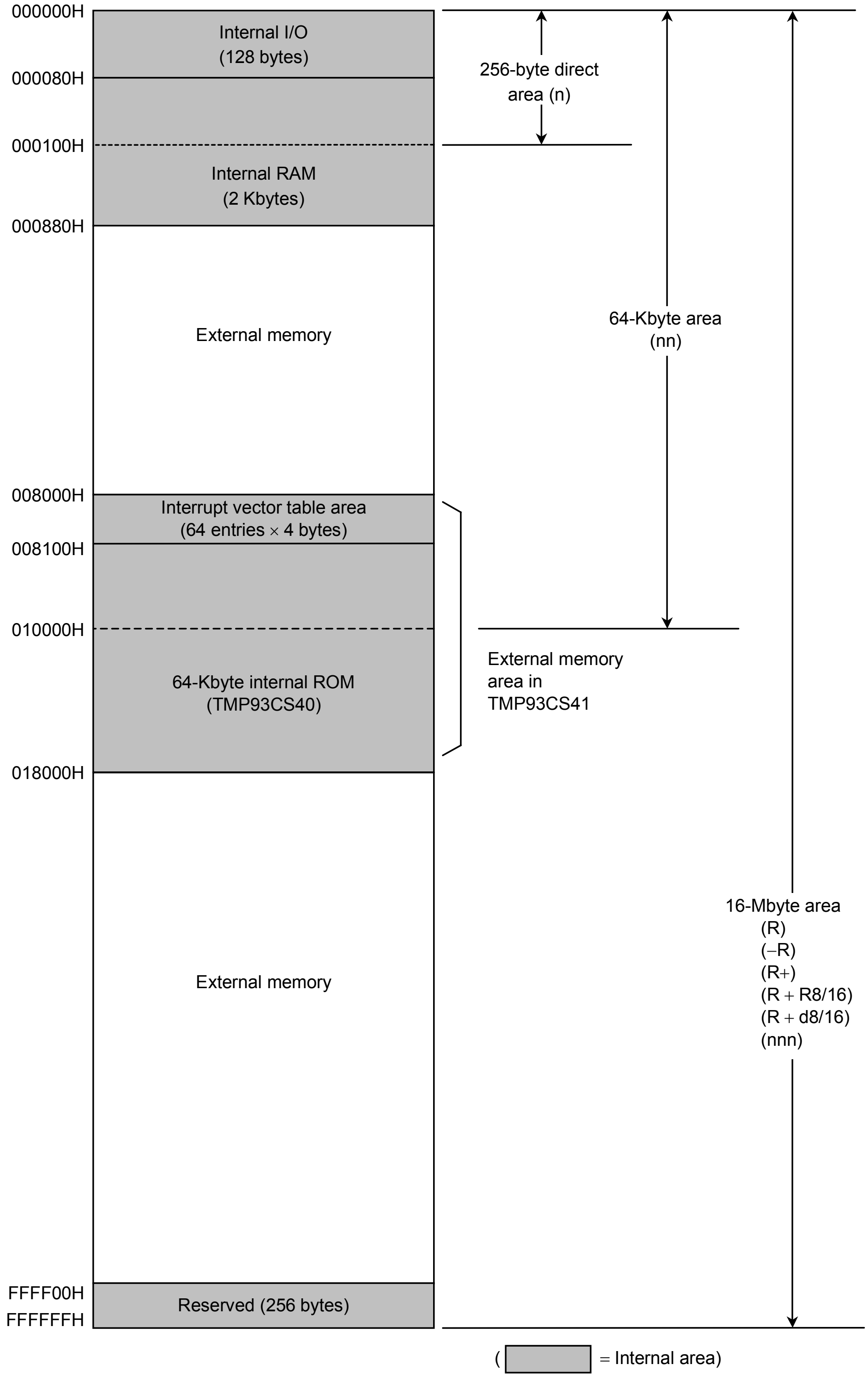
Set this pin to 1. Resetting accesses a built-in ROM via the internal 16-bit bus. When accessing externally, the bus width is set by the chip select/wait control register described in 3.6.3, and the registers of port 1.

(2) TMP93CS41

1. With fixed 16-bit data bus or with 16-bit data bus interchangeable with 8-bit data bus.
   Set this pin to 0. Port 1, AD8 to AD15 or A8 to A15 pins are fixed to AD8 to AD15 functions. Any values set in the port 1 control register or the port 1 function register are invalid.
   The external data bus width is set by the chip select/wait control register.
   After reset, it is necessary to set the program memory to be accessed, to 16-bit data bus.
2. With fixed external 8-bit data bus
   Set this pin to 1. Port 1, AD8 to AD15 or A8 to A15 pins are fixed to A8 to A15 functions. Any values set in the port 1 control register or the port 1 function register are invalid.
   The values of Bit4 <B0BUS>, <B1BUS> and <B2BUS> in the chip select/wait control register, which are described in 3.6.2, are invalid. The external 8-bit data bus is fixed.

## 3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP93CS40 and TMP93CS41.




Note: The 256-byte area from FFFF00H to FFFFFFH can not be used.

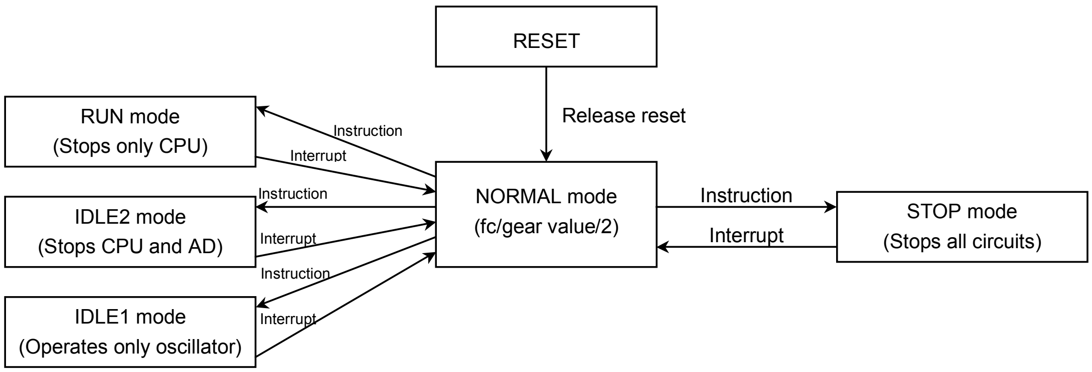
Figure 3.2.1 Memory Map

## 3.3 Dual Clock, Standby Function

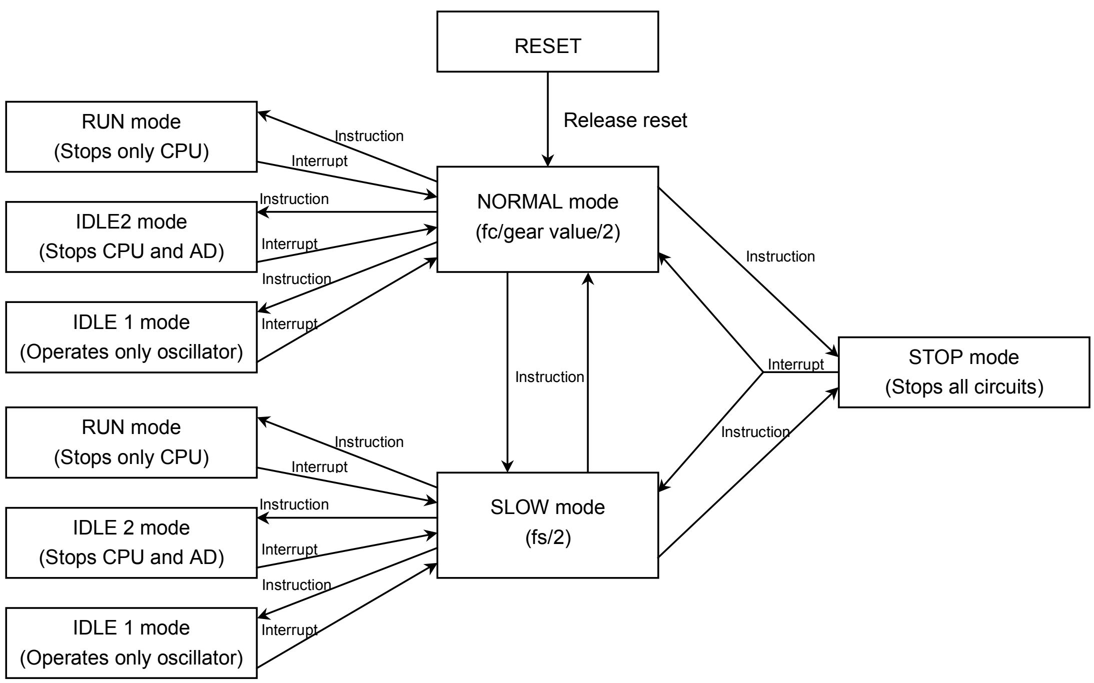
Dual clock, standby control circuits are comprised of a system clock controller, prescaler clock controller, internal clock pin output function and standby controller.

The oscillator operating modes are classified as either (a) Single clock mode (using only the X1 and X2 pins), or (b) Dual clock mode (using the X1, X2, XT1, and XT2 pins).

Figure 3.3.1 shows state diagrams for the two clock modes. Figure 3.3.2 shows the corresponding block diagram, Figure 3.3.3 displays functions of the I/O registers and Table 3.3.1 lists correspondences between alternative states of the system clock and those of the CPU, oscillator and internal I/O components.




(a) Single Clock Mode State Diagram




(b) Dual Clock Mode State Diagram

Figure 3.3.1 State Diagrams

The clock frequency input from the X1 and X2 pins is called fc, and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<SYSCK> and <GEAR2:0> is called the system clock $f_{FPH}$. The divided clock of $f_{FPH}$ is defined as the system clock $f_{SYS}$, and one cycle of $f_{SYS}$ is defined as one state.

Table 3.3.1 Relations between System Clock States and Other Internal Operations

<table>
<tr><th colspan="2" rowspan="2">Operating Mode</th><th colspan="2">Oscillator</th><th rowspan="2">CPU</th><th rowspan="2">Internal I/O</th><th rowspan="2">System Clock $f_{SYS}$</th></tr>
<tr><th>High Frequency (fc)</th><th>Low Frequency (fs)</th></tr>
<tr><td rowspan="6">Single clock</td><td>RESET</td><td rowspan="5">Oscillation</td><td rowspan="6">Stop</td><td>Reset</td><td>Reset</td><td>fc/32</td></tr>
<tr><td>NORMAL</td><td>Operate</td><td rowspan="2">Operate</td><td rowspan="4">Programmable (fc/2, fc/4, fc/8, fc/16, fc/32)</td></tr>
<tr><td>RUN</td><td rowspan="4">Stop</td></tr>
<tr><td>IDLE2</td><td>Stop only AD</td></tr>
<tr><td>IDLE1</td><td rowspan="2">Stop</td></tr>
<tr><td>STOP</td><td>Stop</td><td>Stop</td></tr>
<tr><td rowspan="7">Dual clock</td><td>RESET</td><td rowspan="2">Oscillation</td><td>Stop</td><td>Reset</td><td>Reset</td><td>fc/32</td></tr>
<tr><td>NORMAL</td><td>Programmable</td><td rowspan="2">Operate</td><td rowspan="3">Operate</td><td>Programmable (fc/2, fc/4, fc/8, fc/16, fc/32)</td></tr>
<tr><td>SLOW</td><td>Programmable</td><td>Oscillation</td><td>fs/2</td></tr>
<tr><td>RUN</td><td colspan="2" rowspan="3">Oscillator being used as system clock: Oscillation<br>Other oscillator: Programmable</td><td rowspan="4">Stop</td><td rowspan="3">Programmable (fc/2, fc/4, fc/8, fc/16, fc/32, fs/2)</td></tr>
<tr><td>IDLE2</td><td>Stop only AD</td></tr>
<tr><td>IDLE1</td><td rowspan="2">Stop</td></tr>
<tr><td>STOP</td><td colspan="2">Stop</td><td>Stop</td></tr>
</table>

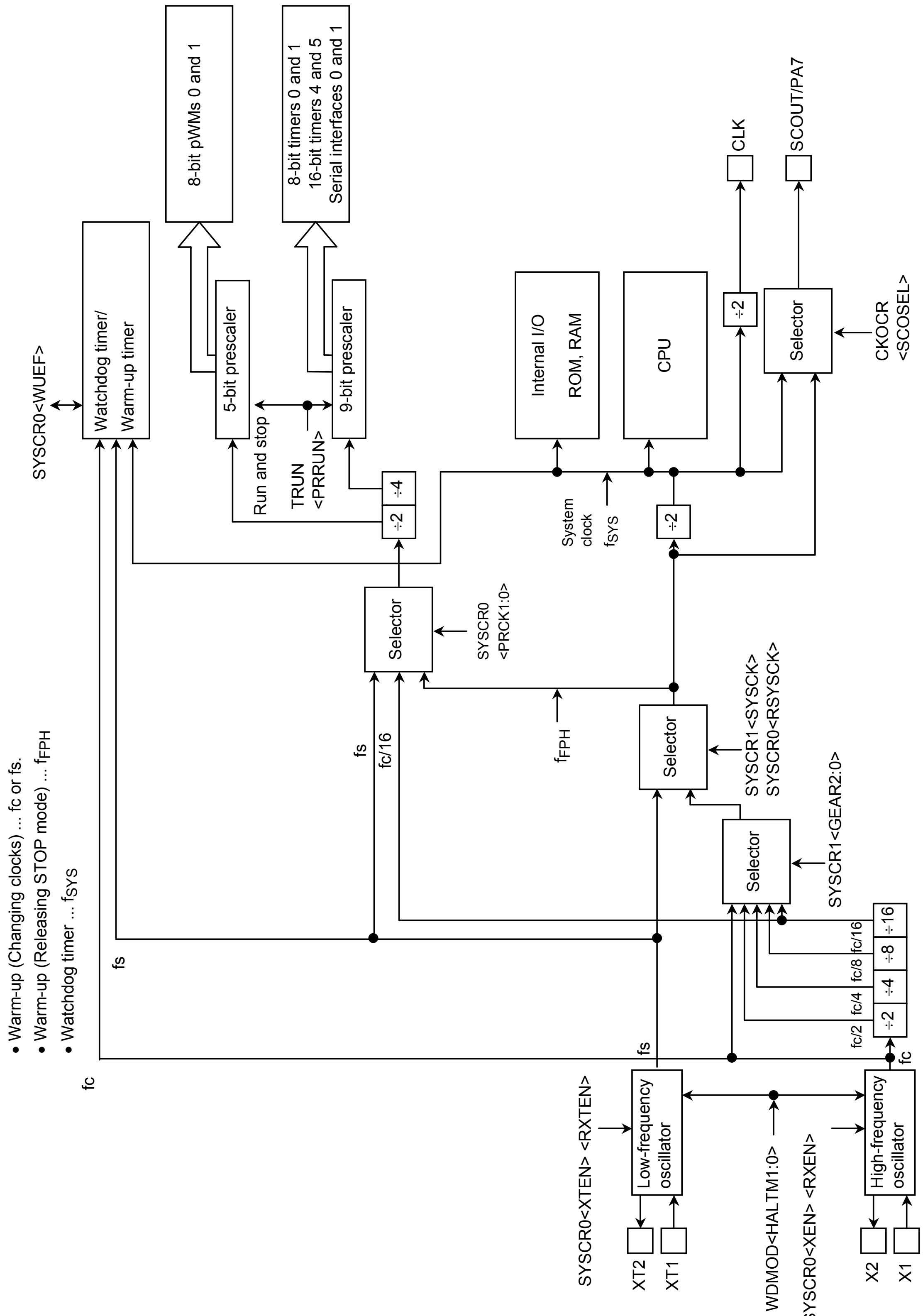



Figure 3.3.2 Block Diagram of Dual Clock and Standby Circuits

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| SYSCR0 (006EH) | Bit symbol | XEN | XTEN | RXEN | RXTEN | RSYSCK | WUEF | PRCK1 | PRCK0 |
| | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | Function | High-frequency oscillator (fc)<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs)<br>0: Stop<br>1: Oscillation | High-frequency oscillator (fc) after released STOP mode<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs) after released STOP mode<br>0: Stop<br>1: Oscillation | Select clock after STOP mode is released<br>0: fc<br>1: fs | Warm-up timer<br>(Write)<br>0:Don't care<br>1: Start timer<br>(Read)<br>0: Warm up complete<br>1: Continue warm up | Select prescaler clock<br>00: $f_{FPH}$<br>01: fs<br>10: fc/16<br>11: (Reserved) | |
| SYSCR1 (006FH) | Bit symbol | | | | | SYSCK | GEAR2 | GEAR1 | GEAR0 |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 0 | 1 | 0 | 0 |
| | Function | | | | | Select system clock<br>0: fc<br>1: fs | 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) | | |
| WDMOD (005CH) | Bit symbol | WDTE | WDTP1 | WDTP0 | WARM | HALTM1 | HALTM0 | RESCR | DRVE |
| | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | WDT control<br>1: Enable | WDT detection time<br>00: $2^{15}/f_{SYS}$<br>01: $2^{17}/f_{SYS}$<br>10: $2^{19}/f_{SYS}$<br>11: $2^{21}/f_{SYS}$ | | Warm-up timer<br>0: $2^{14}$/ frequency input<br>1: $2^{16}$/ frequency input | Standby mode<br>00: RUN mode<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE mode | | 1: Connects WDT output to reset pin internally. | 1: Drives pin even in STOP mode |
| CKOCR (006DH) | Bit symbol | | | | | SCOSE | SCOEN | ALEEN | CLKEN |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 0 | 0 | 0/1 (Note 2) | 0/1 (Note 2) |
| | Function | | | | | SCOUT select<br>0: $f_{FPH}$<br>1: $f_{SYS}$ | SCOUT output control<br>0: I/O port<br>1: SCOUT output | ALE pin output control<br>0: HZ port<br>1: ALE output | CLK pin output control<br>0: HZ port<br>1: CLK output |

Note 1: SYSCR1<bit7:4> are always 1.

Note 2: In the TMP93CS40, resetting sets the <ALEEN> and <CLKEN> bits to 0. In the TMP93CS41, resetting sets the <ALEEN> and <CLKEN> bits to 1 (Output ALE and CLK). The CLK pin is internally pulled up during reset, regardless of the product types.

Note 3: Writing 0 to SYSCR1<SYSCK> enables the high-frequency oscillator regardless of the value of SYSCR0<XEN>. Additionally, writing 1 to SYSCR1<SYSCK> enables the low-frequency oscillator regardless of the value of SYSCR0<XTEN>.

Figure 3.3.3 I/O Register about Dual Clock, Standby

### 3.3.1 System Clock Controller

The system clock controller generates the system clock signal ($f_{SYS}$) for the CPU core and internal I/O. It contains two oscillation circuits and a clock gear circuit for high frequency (fc). The register SYSCR1<SYSCK> changes the system clock to either fc or fs, SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling each oscillator, and SYSCR1<GEAR2:0> changes 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.

The system clock ($f_{SYS}$) is set to fc/32 (fc/16 × 1/2) by the setting of <XEN> = 1, <XTEN> = 0, <SYSCK> = 0, and <GEAR2:0> = 100 upon resetting.

For example, $f_{SYS}$ is set to 0.625 MHz by resetting in the case where the 20 MHz oscillator is connected to the X1 and X2 pins.

The high frequency (fc) and low frequency (fs) clock signals can be easily obtained by connecting a resonator to the X1 and X2, XT1 and XT2 pins, respectively. Clock input from an external oscillator is also possible.

The XT1 and XT2 pins can also function as ports 96 and 97. Therefore in the case of single clock mode, the XT1 and XT2 pins can be used as I/O port pins.

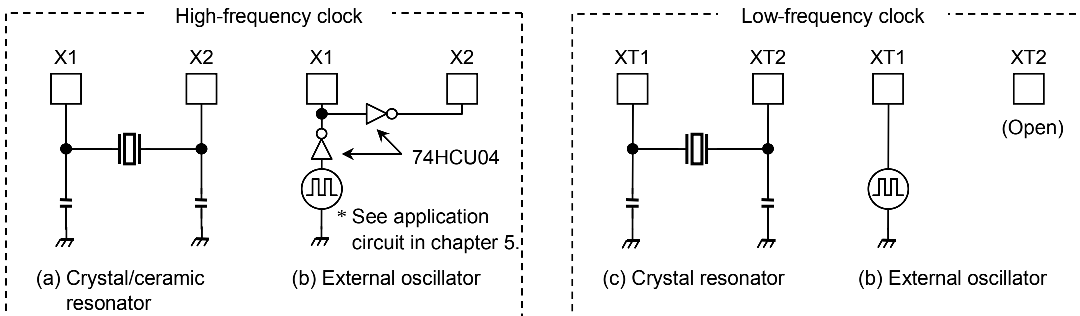



Figure 3.3.4 Examples of Resonator Connection

Note 1: Note on using the low-frequency oscillation circuit.
In connecting the low-frequency resonator to ports 96 and 97, it is necessary to make the following settings to reduce the power consumption.
(Connecting with resonators) P9CR<P96C, P97C> = 11, P9<P96:97> = 00
(Connecting with oscillators) P9CR<P96C, P97C> = 11, P9<P96:97> = 10

Note 2: Accurate adjustment of the oscillation frequency
The CLK pin output at 1/2 the system clock frequency ($f_{SYS}$/2) is used to monitor the oscillation clock.
With a system requiring adjustment of the oscillation frequency, an adjusting program must be written.

(1) Switching from NORMAL to SLOW mode

When the resonator is connected to the X1 and X2, or to the XT1 and XT2 pins, the warm-up timer is used to change the operation frequency after stable oscillation is attained.

The warm-up time can be selected by WDMOD<WARM>.

This starting and stopping of the warm-up timer are performed by programming as in the following examples 1 and 2.

Note 1: The warm-up timer is also used as a watchdog timer. So when it is to be used as a warm-up timer, the watchdog timer function must be disabled.

Note 2: In the case of using an oscillator (not resonator) with stable oscillation, a warm-up timer is not needed.

Note 3: The warm-up timer is operated by an oscillation clock. Therefore there is an error in, warm-up time.

Table 3.3.2 Warm-up Time

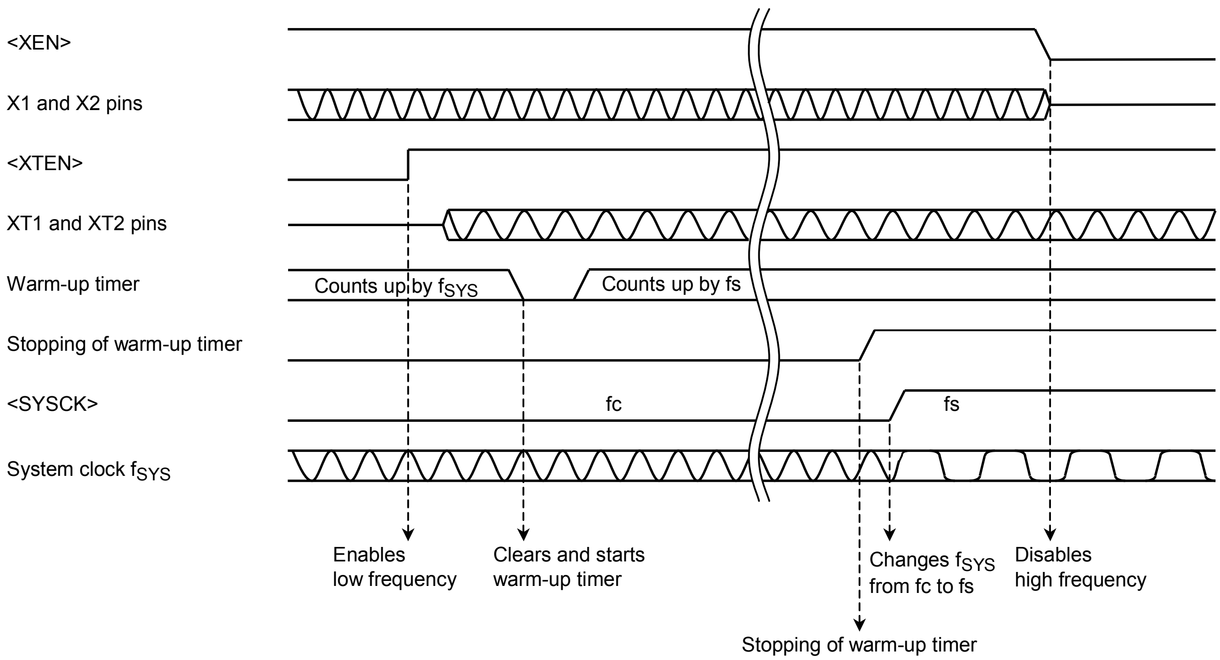
| Warm-up time WDMOD<WARM> | Change to NORMAL (fc) | Change to SLOW (fs) |
|---|---|---|
| 0 ($2^{14}$/frequency) | 0.8192 ms | 500 ms |
| 1 ($2^{16}$/frequency) | 3.2768 ms | 2000 ms |

at fc = 20 MHz,
fs = 32.768 kHz

Clock setting example 1:

Changing from high frequency (fc) to low frequency (fs).

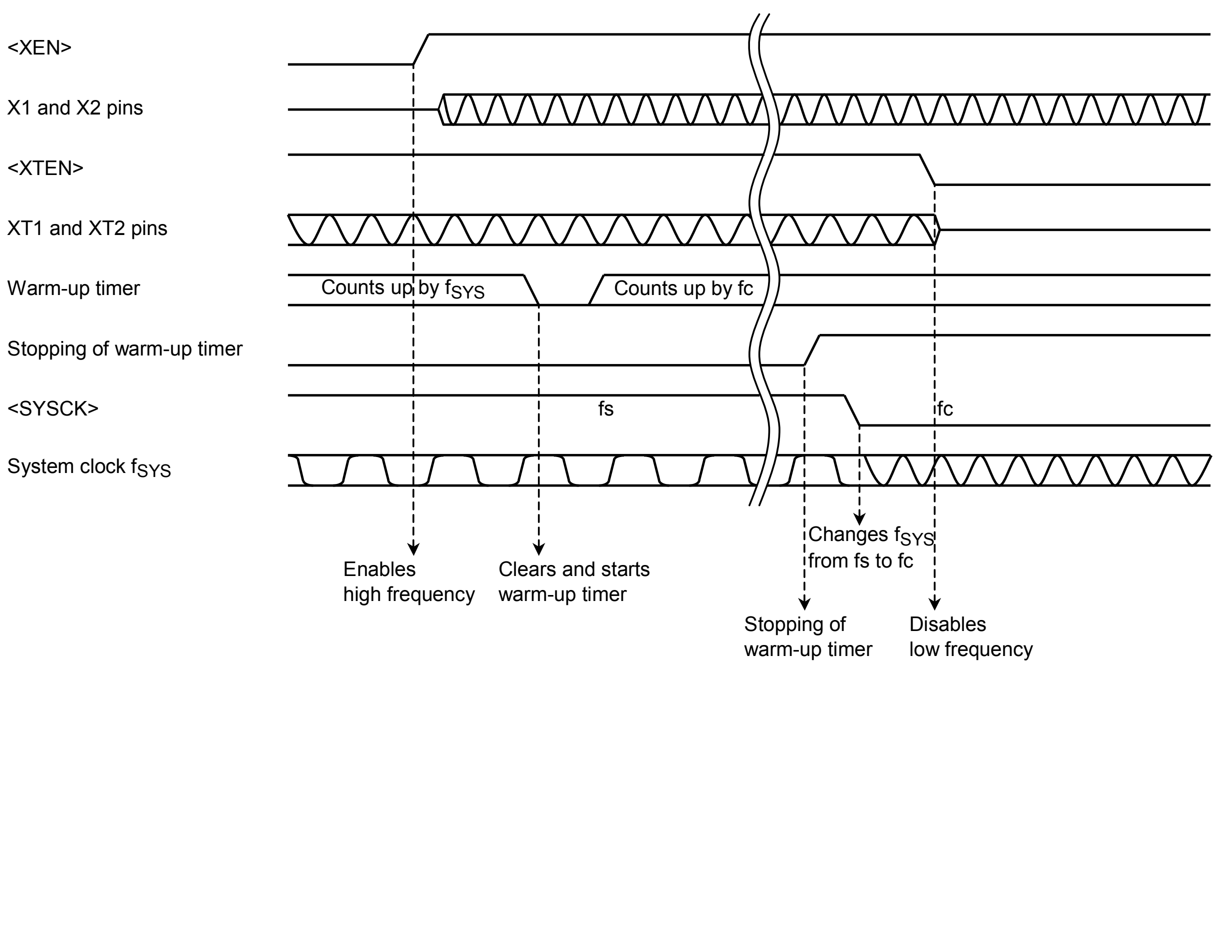
```
SYSCR0  EQU   006EH
SYSCR1  EQU   006FH
WDCR    EQU   005DH
WDMOD   EQU   005CH
        RES   7, (WDMOD)     ; } Disables watchdog timer.
        LD    (WDCR), B1H    ; }
        SET   4, (WDMOD)     ; Sets warm-up time to 2^16/fs.
        SET   6, (SYSCR0)    ; Enables low-frequency oscillation.
        SET   2, (SYSCR0)    ; Clears and starts warm-up timer.
WUP:    BIT   2, (SYSCR0)    ; } Detects stopping of the warm-up timer.
        JR    NZ, WUP        ; }
        SET   3, (SYSCR1)    ; Changes fSYS from fc to fs.
        RES   7, (SYSCR0)    ; Disables high-frequency oscillation.
        SET   7, (WDMOD)     ; Enables watchdog timer.
```



Clock setting example 2:

Changing from low frequency (fs) to high frequency (fc).

```
SYSCR0   EQU   006EH
SYSCR1   EQU   006FH
WDCR     EQU   005DH
WDMOD    EQU   005CH
         RES   7, (WDMOD)    ; } Disables watchdog timer.
         LD    (WDCR), B1H   ; }
         RES   4, (WDMOD)    ; Sets warm-up time to 2^14/fc.
         SET   7, (SYSCR0)   ; Enables high-frequency oscillation (fc).
         SET   2, (SYSCR0)   ; Clears and starts warm-up timer.
WUP:     BIT   2, (SYSCR0)   ; } Detects stopping of the warm-up timer.
         JR    NZ, WUP       ; }
         RES   3, (SYSCR1)   ; Changes fSYS from fs to fc.
         RES   6, (SYSCR0)   ; Disables low-frequency oscillation.
         SET   7, (WDMOD)    ; Enables watchdog timer.
```



(2) Clock gear controller

When the high-frequency clock fc is selected at SYSCR1<SYSCK> = “0”, the clock gear select register SYSCR1<GEAR2:0> sets $f_{FPH}$ to either fc, fc/2, fc/4, fc/8, or fc/16. Switching $f_{FPH}$ with the clock gear reduces the power consumption.

Clock setting example 3:

Changing gear value of the high-frequency clock

```
SYSCR1    EQU    006FH
                 LD (SYSCR1), XXXX0000B        ; Changes fSYS to fc/2.
                 X: Don’t care
```

(High-frequency clock gear changing)

To change the frequency of the clock gear, write the value to the SYSCR1<GEAR2:0> register. It is necessary to continue the warm-up time until changing $f_{FPH}$ after writing the register value.

There is a possibility that the instruction immediately following the clock-gear-changing instruction will be executed by the clock gear before executing its gear change. To ensure that the instruction immediately following the clock-gear-changing instruction will only be executed by the clock gear after changing its gear ratio, input a dummy instruction (An instruction to execute a write cycle) as follows.

Example:

Instruction to be executed by the clock gear after changing its gear ratio.

```
SYSCR1    EQU    006FH
                 LD (SYSCR1), XXXX0001B        ; Changes fSYS to fc/4.
                 LD (DUMMY), 00H               ; Dummy instruction.
```

Instruction to be executed by the clock gear after changing.

### 3.3.2 Prescaler Clock Controller

The 9-bit prescaler provides a clock signal to the 8-bit timer 0 and timer 1, 16-bit timer 4 and timer 5, and serial interface 0 and serial interface 1. The 5-bit prescaler provides a clock signal to the 8-bit PWM timer 0 and 1.

The clock input to the 5-bit prescaler is a clock signal which is selected as either $f_{FPH}$, fc/16, or fs according to the value in the SYSCR0<PRCK1:0> register, and divided by 2.

The clock input to the 9-bit prescaler is a clock signal which is selected as either $f_{FPH}$, fc/16, or fs according to the value in the SYSCR0<PRCK1:0> register, and divided by 4.

The <PRCK1:0> register is initialized to 00 by resetting.

When the IDLE1 mode (Operating only the oscillator) is being used, set TRUN<PRRUN> to 0 to reduce the power consumption before a HALT instruction is executed.

### 3.3.3 Internal Clock Pin Output Function

(1) PA7/SCOUT pin

The PA7/SCOUT pin outputs the internal clock signals $f_{FPH}$ or $f_{SYS}$.

One bit in the port A control register PACR<PA7C>, and two bits in the clock output control register CKOCR<SCOEN and SCOSEL> specify the clock and the pins. The PA7/SCOUT pin is assigned as the input port in resetting.

Table 3.3.3 shows states of the SCOUT pin in the alternative operation modes which it can assume on condition that the PA7/SCOUT pin is specified as SCOUT output.

Table 3.3.3 SCOUT Pin States in Alternative Operation Modes

| Operation Mode / Output Clock | NORMAL, SLOW | HALT Mode | |
|---|---|---|---|
| | | RUN, IDLE2, IDLE1 | STOP |
| $f_{FPH}$ | Outputs $f_{FPH}$ clock. | | Fixed to 0 or 1. |
| $f_{SYS}$ | Outputs $f_{SYS}$ clock. | | |

(2) CLK pin

The CLK pin outputs the internal clock signal $f_{SYS}$ divided by 2.

The type of output is determined by one bit in the clock output control register, CKOCR<CLKEN>. Writing 1 sets the clock output, and writing “0” sets the CLK pin to high impedance. CKOCR<CLKEN> is set to “0” upon resetting.

During resetting, the CLK pin is internally pulled up regardless of the value of the <CLKEN> register.

Table 3.3.4 States of the <CLKEN> Register, and CLK Pin Operation after Reset

| Type No. | CKOCR<CLKEN> | CLK Pin Operation |
|---|---|---|
| TMP93CS40 | 0 | High impedance |
| TMP93CS41 | 1 | $f_{SYS}$/2 clock output |

Note: To set CKOCR<CLKEN> = “0” and set the CLK pin to high impedance, an external pull-up resistor is needed to prevent current from flowing to the input buffer of the CLK pin.

### 3.3.4 Standby Controller

(1) HALT mode

When the HALT instruction is executed, the operating mode changes to RUN, IDLE2, IDLE1 or STOP mode depending on the contents of the HALT mode setting register WDMOD<HALTM1:0>. Figure 3.3.5 shows the alternative states of the watchdog timer mode registers.

Watchdog Timer Mode Register

WDMOD (005CH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | WDTE | WDTP1 | WDTP0 | WARM | HALTM1 | HALTM0 | RESCR | DRVE |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Watchdog timer control<br>0: Disable<br>1: Enable | Watchdog timer detect time selection<br>00: $2^{15}/f_{SYS}$<br>01: $2^{17}/f_{SYS}$<br>10: $2^{19}/f_{SYS}$<br>11: $2^{21}/f_{SYS}$ | | Warm-up timer<br>0: $2^{14}$/clock frequency selection<br>1: $2^{16}$/clock frequency selection | HALT mode selection<br>00: RUN mode<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE2 mode | | Runaway detection internal reset control<br>1: Executes internal reset by runaway detection | STOP mode pin control<br>1: Drive pins in STOP mode |

Pin state control in STOP mode

| | |
|---|---|
| 0 | I/O off |
| 1 | Retains the state before HALT |

HALT mode setting

| | |
|---|---|
| 00 | RUN mode (only CPU stops) |
| 01 | STOP mode (all circuits stop) |
| 10 | IDLE1 mode (only oscillator operating) |
| 11 | IDLE2 mode (partial I/O operation) |

Warm-up time selection at returning from the STOP mode (see table 3.3.7)

| | |
|---|---|
| 0 | $2^{14}$/select clock frequency |
| 1 | $2^{16}$/select clock frequency |

Figure 3.3.5 Watchdog Timer Mode Register

The features of the RUN, IDLE2, IDLE1, and STOP modes are as follows.

1. RUN: Only the CPU HALTs.
2. IDLE2: The built-in oscillator and the specified I/O operates.
3. IDLE1: Only the built-in oscillator operates, while all other built-in circuits stop.
4. STOP: All internal circuits including the built-in oscillator stop. This greatly reduces power consumption.

The operations in the halt state are described in Table 3.3.5.

Table 3.3.5 I/O Operation during HALT Mode

<table>
<tr><th colspan="2">HALT mode</th><th>RUN</th><th>IDLE2</th><th>IDLE1</th><th>STOP</th></tr>
<tr><td colspan="2">WDMOD<HALTM1:0></td><td>00</td><td>11</td><td>10</td><td>01</td></tr>
<tr><td rowspan="10">Block</td><td>CPU</td><td colspan="4">HALT</td></tr>
<tr><td>I/O port</td><td colspan="3">Maintain the state when the HALT instruction was executed.</td><td>See Table 3.3.8</td></tr>
<tr><td>8-bit timer</td><td colspan="2" rowspan="5">Operate</td><td colspan="2" rowspan="8">STOP</td></tr>
<tr><td>8-bit PWM timer</td></tr>
<tr><td>16-bit timer</td></tr>
<tr><td>Pattern generator</td></tr>
<tr><td>Serial interface</td></tr>
<tr><td>AD converter</td><td></td><td></td></tr>
<tr><td>Watchdog timer</td><td colspan="2" rowspan="2"></td></tr>
<tr><td>Interrupt controller</td></tr>
</table>

(2) How to release the HALT mode

These halt states can be released by resetting or by requesting an interrupt. The halt release sources are determined by the combinations between the states of the interrupt mask register<IFF2:0> and the HALT modes. The details for releasing the halt status are shown in Table 3.3.6.

- Release by requesting an interrupt

  This method of releasing operation from the HALT mode depends on the interrupt-enabled status being in force. When the interrupt request level set before executing the HALT instruction exceeds the value of the interrupt mask register, the interrupt due to that source is processed after releasing the HALT mode, and then the CPU starts executing the next 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 of the HALT mode is not executed. (In non-maskable interrupts, interrupt processing is preformed after releasing the HALT mode regardless of the value of the mask register.)

  INT0 interrupts are a special case in which release of the HALT mode is executed even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register. In this case interrupt processing is not performed, and the CPU starts executing the next instruction that follows the HALT instruction, but the interrupt request flag is held at 1.

  Note: Usually, interrupts can release all halts status. However, the interrupts = ($\overline{\text{NMI}}$, INT0) which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of $f_{FPH}$) with IDLE1 or STOP mode (IDLE2 and RUN are not applicable to this case). (In this case, an interrupt request is kept on hold internally.)
  If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

- Release by resetting

  Resetting releases all halt status settings.

  It is necessary to allow enough resetting time (3 ms or more) for the operation of the oscillator to stabilize.

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

When the HALT mode is released by resetting, the internal RAM data maintains the state it was in before the HALT instruction was executed.

However the other setting contents are initialized. (Release of the HALT mode due to interrupts maintains all setting contents in their states before the HALT instruction was executed.)

Table 3.3.6 Halt Release Sources and Halt Release Operations

| Interrupt Receiving Status | | | Interrupt Enabled (Interrupt level) ≥ (Interrupt mask) | | | | Interrupt Disabled (Interrupt level) < (Interrupt mask) | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| HALT Mode | | | RUN | IDLE2 | IDLE1 | STOP | RUN | IDLE2 | IDLE1 | STOP |
| Halt release source | Interrupt | NMI | ♦ | ♦ | ♦ | ♦ *1 | – | – | – | – |
| | | INTWDT | ♦ | × | × | × | – | – | – | – |
| | | INT0 | ♦ | ♦ | ♦ | ♦ *1 | ○ | ○ | ○ | ○ *1 |
| | | INT4 to INT7 | ♦ | ♦ | × | × | × | × | × | × |
| | | INTT0 to INTT3 | ♦ | ♦ | × | × | × | × | × | × |
| | | INTTR4 to INTTR7 | ♦ | ♦ | × | × | × | × | × | × |
| | | INTRX0, TX0 | ♦ | ♦ | × | × | × | × | × | × |
| | | INTRX1, TX1 | ♦ | ♦ | × | × | × | × | × | × |
| | | INTAD | ♦ | × | × | × | × | × | × | × |
| | RESET | | ♦ | ♦ | ♦ | ♦ | ♦ | ♦ | ♦ | ♦ |

♦: After release of the HALT mode, the CPU starts interrupt processing. (RESET initializes LSI.)

○: After release of the HALT mode, the CPU starts executing the next instruction that follows the HALT instruction.

×: Cannot be used to release the HALT mode.

–: This combination type does not exist because the priority level (Interrupt request level) of non-maskable interrupts is fixed to the highest priority level 7.

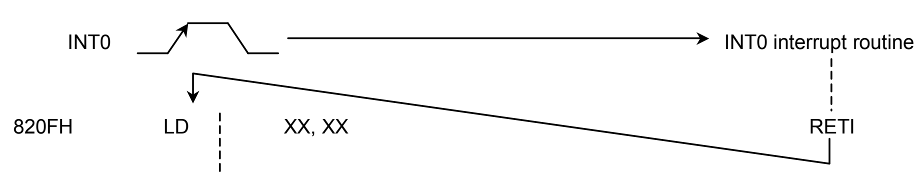
*1: Release the HALT mode is executed after the warm-up cycle is completed.

Note: When release of the HALT mode is executed by an INT0 interrupt of the level mode in the interrupt enabled status, maintain level H until the start of interrupt processing. If level L is set before the start of interrupt processing, interrupt processing is correctly started.

Example of releasing the RUN mode:

An INT0 interrupt releases the halt state when the RUN mode is on.

```
Address
8203H    LD     (IIMC), 00H       ; Select interrupt rising edge.
8206H    LD     (INTE0AD), 06H    ; Sets interrupt level to 6 for INT0.
8209H    EI     5                 ; Sets interrupt level to 5 for CPU.
820BH    LD     (WDMOD), 00H      ; Sets HALT mode to run.
820EH    HALT                     ; Halts CPU.
```



(3) Operation

1. RUN mode

In the RUN mode, the system clock in the MCU continues to operate even after a HALT instruction is executed. Only the CPU stops executing further instructions.

In the halt state, an interrupt request is accepted on the falling edge of the CLK signal.

Release of the RUN mode is executed by the external or internal interrupts. (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

Figure 3.3.6 shows the timing for releasing the halt state by interrupts in the RUN or IDLE2 modes.

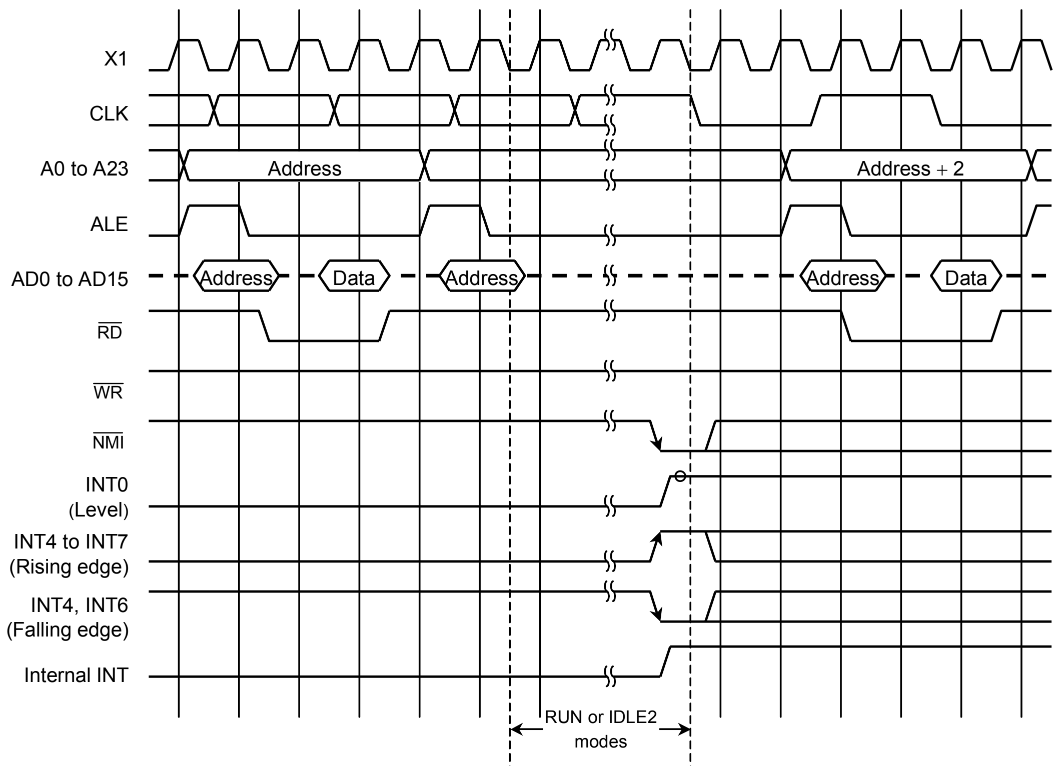



Figure 3.3.6 Timing Chart for Releasing the Halt State by Interrupt in RUN/IDLE2 Modes

2. IDLE2 mode

In the IDLE2 mode, the system clock signal is supplied only to specific internal I/O devices, and the CPU stops executing the current instruction. In the IDLE2 mode, the halt state is released by an interrupt with the same timing as in the RUN mode. The IDLE2 mode is released by external or internal interrupts, except for INTWDT and INTAD interrupts. (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

In the IDLE2 mode, the watchdog timer should be disabled before entering the halt status, to prevent the watchdog timer interrupt from occurring just after release of the HALT mode.

3. IDLE1 mode

In the IDLE1 mode, only the internal oscillator operates. The system clock in the MCU stops, and the CLK pin is fixed at the level H in the output enabled state. (CKOCR<CLKEN> = 1)

In the halt state, an interrupt request is sampled unsyncronously with the system clock, however the halt release (Restart of operation) is performed synchronously with it.

IDLE1 mode is released by external interrupts (NMI, INT0). (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

When the IDLE mode is used, set (TRUN<PRRUN> to 0) to stop the 9 bit and 5 bit prescaler before a HALT instruction is executed, to reduce the power consumption.

Figure 3.3.7 illustrates the timing for releasing the halt state by interrupts in the IDLE1 mode

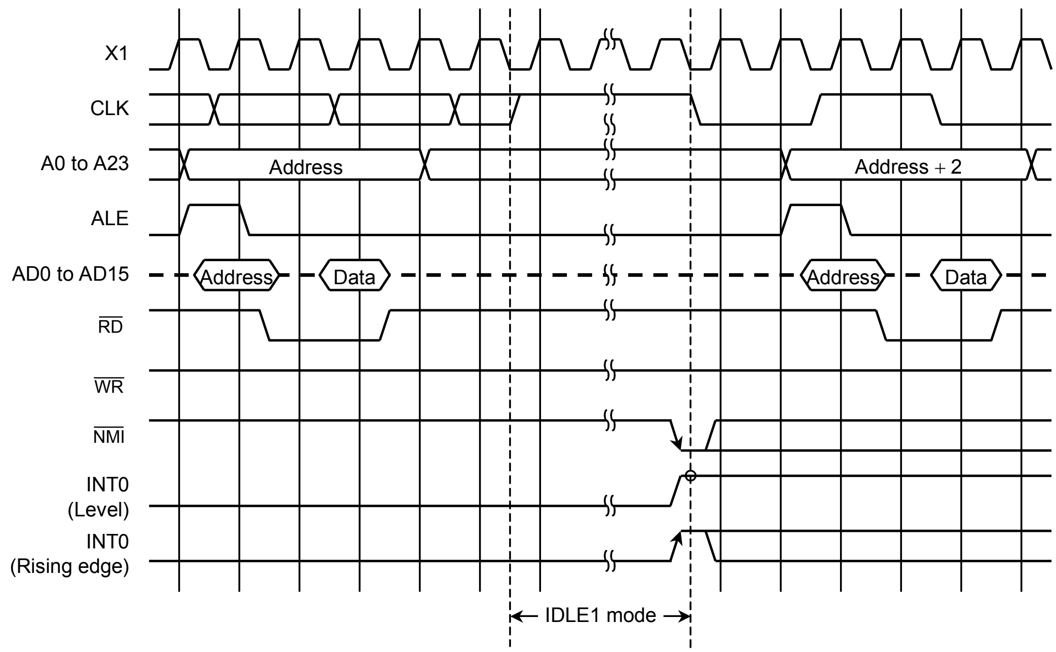



Figure 3.3.7 Timing Chart of Halt State Release by Interrupts in IDLE1 Mode

4. STOP mode

The STOP mode is selected to stop all internal circuits including the internal oscillator.

The pin status in the STOP mode depends on the setting of a bit in the watchdog timer mode register WDMOD<DRVE>. (See Table 3.3.8 for setting of WDMOD<DRVE>.) Table 3.3.8 summarizes the state of these pins in the STOP mode.

The STOP mode is released by external interrupts (NMI, INT0). When the STOP mode is released, the system clock output starts after the warm-up time required to attain stable oscillation.

The warm-up time can be set using WDMOD<WARM>. See the example of warm-up time setting (Table 3.3.7).

In a system which supplies a stable clock signal generated by an external oscillator, the warm-up time can be reduced by using the setting of T45CR<QCU>.

Figure 3.3.8 illustrates the timing for releasing the halt state by interrupts during the STOP mode.

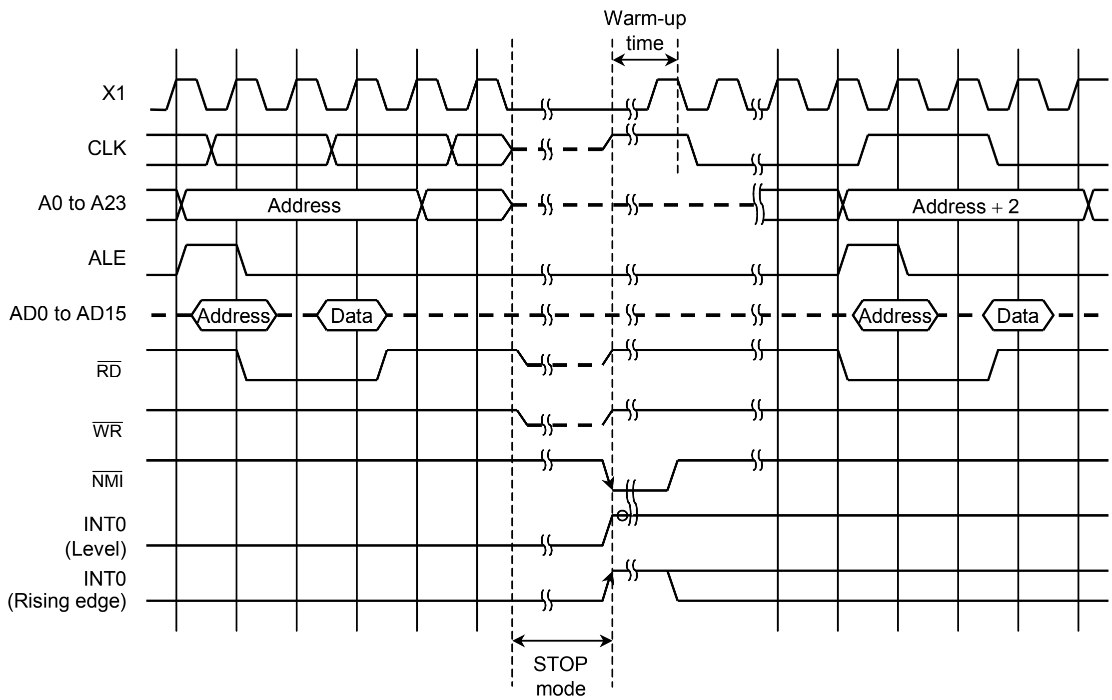



Figure 3.3.8 Timing Chart of Halt State Release by Interrupts in STOP Mode

Table 3.3.7 Example of Warm-up Time after Releasing the STOP Mode

| Clock Operation Frequency after the STOP Mode is Released | Warm-up Time [ms] | | Clock Frequency |
|---|---|---|---|
| | WDMOD<WARM> = 0 | WDMOD<WARM> = 1 | |
| fc | 0.8192 | 3.2768 | fc = 20 MHz |
| fc/2 | 1.6384 | 6.5536 | |
| fc/4 | 3.2768 | 13.1072 | |
| fc/8 | 6.5536 | 26.2144 | |
| fc/16 | 13.1072 | 52.4288 | |
| fs | 500 | 2000 | fs = 32.768 kHz |

How to calculate the warm-up time

WDMOD<WARM> = 0: Clock operation frequency after the $2^{14}$/STOP mode

WDMOD<WARM> = 1: Clock operation frequency after the $2^{16}$/STOP mode

The selection of NORMAL versus SLOW modes is possible after the STOP mode is released.

This selection is mode according to the contents of the SYSCR0<RSYSCK> register. Therefore, setting <RSYSCK>, <RXEN>, and <RXTEN> is necessary before the HALT instruction is executed.

Setting example:

In this illustrative case, the STOP mode is entered while the clock is operating at low frequency (fs). After the STOP mode is released by a NMI interrupt, the clock resumes operation at high frequency.

```
Address
         SYSCR0   EQU    006EH
         SYSCR1   EQU    006FH
         WDMOD    EQU    005CH
8FFDH             LD     (SYSCR1), 08H            ; fSYS = fs/2.
9000H             RES    4,(WDMOD)                ; Sets warm-up time to 2^14/fc.
9002H             LD     (SYSCR0), -11000 - - B   ; Operates at high frequency after STOP mode
                                                    is released.
9005H             HALT
         NMI                                        Clears and starts
                                                    warm-up timer
                                                    (High frequency)
                                                    End
                                                    NMI interrupt routine
9006H             LD     XX, XX                     RETI

         -: No change
```

Note: When different operation modes are used before and after the STOP mode, and halt release interrupt request is accepted during execution of the HALT instruction (8 states), it is possible to release the HALT mode without changing the operation mode. In a system which accepts interrupts during execution of the HALT instruction, set the same operation mode before and after the STOP mode.

Table 3.3.8 Pin States in STOP Mode

| Pin name | I/O | TMP93CS40 | | TMP93CS41 | |
|---|---|---|---|---|---|
| | | <DRVE> = 0 | <DRVE> = 1 | <DRVE> = 0 | <DRVE> = 1 |
| P00 to P07 | Input mode<br>Output mode<br>AD8 to AD15 | ▲<br>–<br>– | ▲<br>Output<br>– | ×<br>×<br>– | ×<br>×<br>– |
| P10 to P17 | Input mode<br>Output mode<br>AD0 to AD7 | ▲<br>–<br>– | ▲<br>Output<br>– | ×<br>×<br>– | ×<br>×<br>– |
| P20 to P27 | Input mode<br>Output mode, A0 to A7/A16 to A23 | ▲<br>▲ | ▲<br>Output | ▲<br>▲ | ▲<br>Output |
| P30 ($\overline{RD}$), P31 ($\overline{WR}$) | Output | – | Output | – | "H" |
| P32 to P37 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| P40, P41 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| P42 (CS2/CAS2) | Input mode<br>Output mode | PD*<br>PD* | Input<br>Output | | |
| P5 | Input mode | ▲ | ▲ | | |
| P6 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| P7 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| P80 to P86 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| P87 (INT0) | Input mode<br>Output mode<br>Input mode (INT0) | PU<br>PU<br>Input | Input<br>Output<br>Input | | |
| P90 to P95 | Input mode<br>Output mode | PU*<br>PU* | Input<br>Output | | |
| PA0 to PA6 | Input mode<br>Output mode | –<br>– | Input<br>Output | The same as for TMP93CS40 | |
| PA7 | Input mode<br>Output mode, SCOUT | –<br>– | Input<br>Output | | |
| $\overline{NMI}$ | Input | Input | Input | | |
| WDOUT | Output | Output | Output | | |
| ALE | Output (<ALEEN> = 1) | "L" | "L" | | |
| CLK | Output (<CLKEN> = 1) | – | "H" | | |
| $\overline{RESET}$ | Input | Input | Input | | |
| $\overline{EA}$ | Input | Input | Input | | |
| AM8/$\overline{AM16}$ | Input | Input | Input | | |
| X1 | Input | – | – | | |
| X2 | Output | "H" | "H" | | |
| P96 | Input mode<br>Output mode<br>XT1 | –<br>–<br>– | Input<br>Output*<br>– | | |
| P97 | Input mode<br>Output mode<br>XT2 | –<br>–<br>– | Input<br>Output*<br>– | | |

(Align)

–: Input is not accepted; output is at high impedance.

Input: Input gate in operation. Fix input voltage to "L" or "H" so that the input state pin stays constant.

Output: Output state.

Output*: Open-drain output state. Input gate in operation. Set output to "L" or attach pull up on pin so that the input gate stays constant.

PU: Programmable pull-up pin. When a pull-up resistor is not set, fix the pin to avoid through current because the input gate always operates.

PU*: Programmable pull-up pin in input gate disable state. No through current flows even if the pin is set to high impedance.

PD*: Programmable pull-down pin in input gate disable state. No through current flows even if the pin is set to high impedance.

▲: When a HALT instruction is executed and CPU stops at the address of the port register, an input gate operates. Fix the pin to avoid through current, and change the program. In all other cases, input is not accepted; output is at high impedance.

×: Cannot be set.

Note: Port registers are used for controlling programmable pull up/pull down. If a pin can be used for an output function (e.g., P71/TO1) and the output function is specified, whether pull up or pull down is selected depends on the output function data. If a pin can be used for an input function, whether pull up or pull down is selected depends on the port register setting value only.

## 3.4 Interrupts

Interrupts are controlled by the CPU interrupt mask register SR<IFF2:0> and the built-in interrupt controller.

Altogether the TMP93CS40 and TMP93CS41 have the following 29 interrupt sources:

- Interrupts from the CPU, 9 sources
  (Software interrupts, and illegal (Undefined) instruction execution)
- Interrupts from external pins ($\overline{\text{NMI}}$, INT0, and INT4 to INT7), 6 sources
- Interrupts from built-in I/Os, 14 sources

A fixed individual interrupt vector number is assigned to each interrupt source; any one of six levels of priority can also be assigned to each maskable interrupt. Non-maskable interrupts have a fixed priority of 7.

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

The CPU compares the value of the priority sent, with the value in the CPU interrupt mask register <IFF2:0>. If the value sent is greater than in that the CPU interrupt mask register, the interrupt is accepted. However, software interrupts and illegal instruction execution interrupts are not compared with the <IFF2:0> register. They are given top priority. The value in the CPU interrupt mask register <IFF2:0> can be changed using the EI instruction. Executing EI n changes the contents of <IFF2:0> to n. For example, programming EI 3 enables acceptance of maskable interrupts with a priority of 3 or greater, and non-maskable interrupts which are set in the interrupt controller. When EI or EI 0 is programmed, maskable interrupts with a priority of 1 or greater, and non-maskable interrupts are enabled in the interrupt instructions (In the same way as the EI 1).

The DI instruction operates in the same way as the EI 7 instruction, setting <IFF2:0> = 7. Since the priority values for maskable interrupts are 0 to 6, the DI instruction is used to disable acceptance of maskable interrupts. The EI instruction becomes effective immediately after execution. (With the TLCS-90, the EI instruction becomes effective after execution of the next following instruction.)

In addition to the general-purpose interrupt processing mode described above, there is also a micro DMA processing mode. Micro DMA is a mode used by the CPU to automatically transfer byte or word data. It enables the CPU to process interrupts such as data saves to built-in I/Os at high speed.

Figure 3.4.1 is a flowchart showing overall interrupt processing.

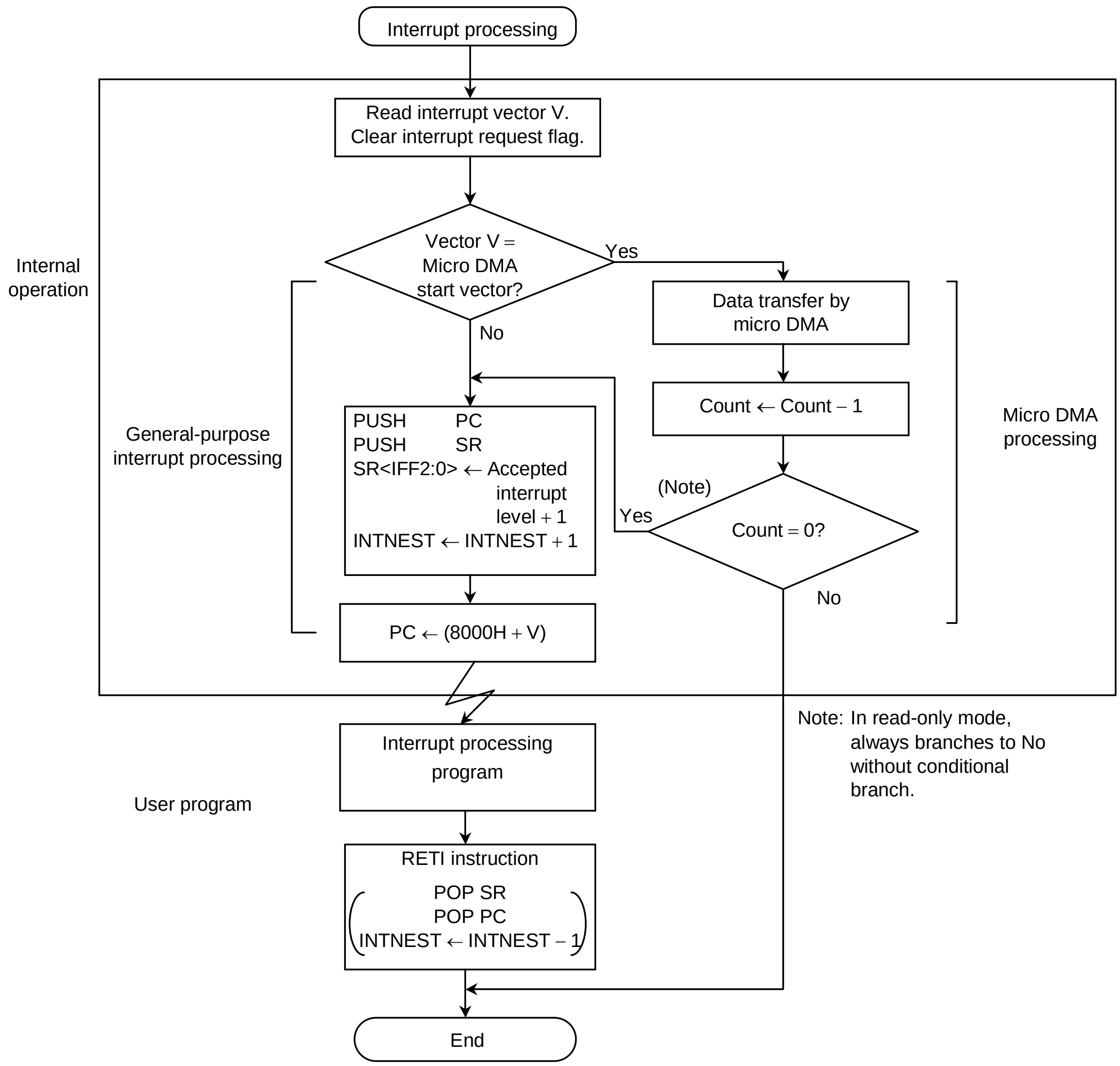



Figure 3.4.1 Interrupt Processing Flowchart

### 3.4.1 General-purpose Interrupt Processing

When accepting an interrupt, the CPU operates as follows. In the cases of software interrupts or interrupts generated by the CPU because of attempts to execute illegal instructions, the following steps (1) and (3) are not executed.

(1) The CPU reads the interrupt vector from the interrupt controller. When more than one interrupt with the same priority level is generated simultaneously, the interrupt controller generates interrupt vectors in accordance with the default priority, then clears the interrupt request. The default priority is fixed as follows: The smaller the vector value, the higher the priority.

(2) The CPU pushes the program counter and the status register to the system stack area (Area indicated by the system mode stack pointer (XSP)).

(3) The CPU sets a value in the CPU interrupt mask register <IFF2:0> that is higher by 1 than the priority level value of the accepted interrupt. However, if the accepted interrupt's priority value is 7, 7 is set without an increment.

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

(5) The CPU jumps to an address stored in the 8000H + interrupt vector, then starts the interrupt processing routine.

The following table shows the number of processing states corresponding to steps 1 to 5 above.

| Bus Width of Stack Area | Bus Width of Interrupt Vector Area | Number of Interrupt Processing States | |
|---|---|---|---|
| | | Max Mode | Min Mode |
| 8 bits | 8 bits | 35 | 31 |
| | 16 bits | 31 | 27 |
| 16 bits | 8 bits | 29 | 27 |
| | 16 bits | 25 | 23 |

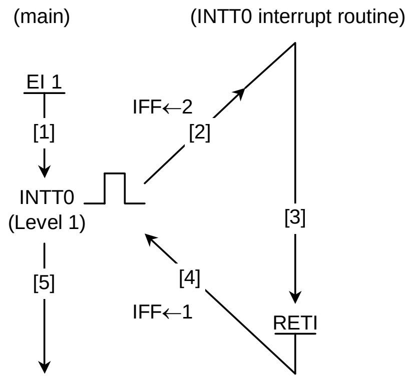
The RETI instruction is usually used to complete the interrupt processing. Executing this instruction restores the contents of the program counter and the status registers, and decrements the interrupt nesting counter (INTNEST).

Though acceptance of non-maskable interrupts cannot be disabled by programming, acceptance of maskable interrupts can. A priority can be set for each source of maskable interrupts. The CPU accepts any interrupt request with a priority higher than the current value in the CPU mask register <IFF2:0>. The CPU mask register <IFF2:0> is then set to a value higher by 1 than the priority of the accepted interrupt. Thus, if another interrupt is generated with a priority level higher than the interrupt currently being processed, the CPU accepts the interrupt with the higher level, causing interrupt processing to nest.

If an interrupt request with a priority higher than the currently-processed interrupt is generated during the time that CPU is processing the above steps (1) to (5), and is accepted before the first instruction in the interrupt processing routine is executed, this will cause interrupt processing to nest. (The nesting process is the same as in the case of overlapping each non-maskable interrupt (Level 7).) The CPU does not accept an interrupt request of the same priority level as that of the interrupt currently being processed.

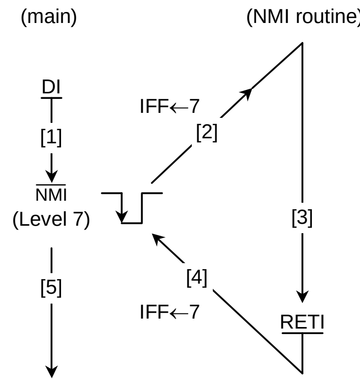
Resetting initializes the CPU mask registers <IFF2:0> to the value 7; therefore, acceptance of all maskable interrupts is disabled.

(1) Maskable interrupt



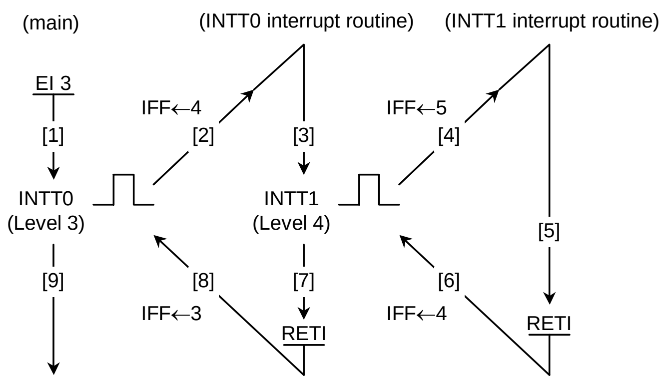

During execution of the main program, the CPU accepts an interrupt request. The CPU then increments IFF so that no new interrupts of priority level 1 will be accepted during processing of the interrupt routine.

(2) Non-maskable interrupt



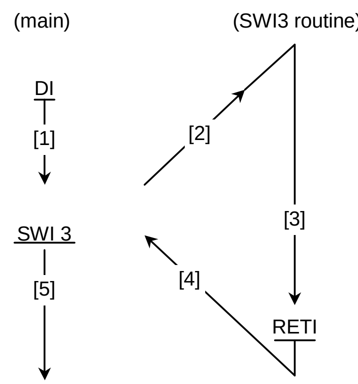

The DI instruction is executed in the main program, so that only interrupts of priority level 7 are accepted. In this state the CPU does not increment the IFF even if the CPU accepts an interrupt request of level 7.

(3) Interrupt nesting



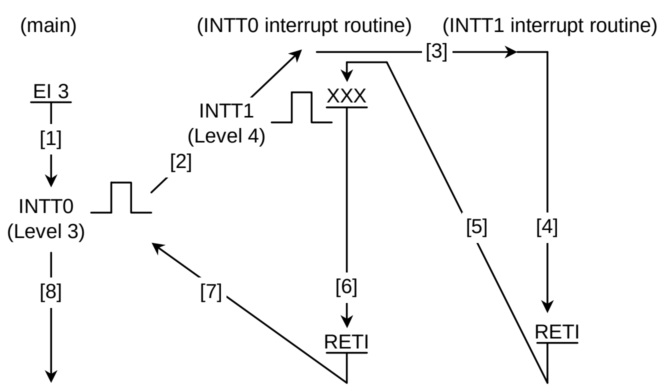

During processing an interrupt of priority level 3, the IFF is set to 4. When an interrupt with a level higher than 4 is generated, the CPU accepts the interrupt with the higher priority level, causing interrupt processing to nest.

(4) Software interrupt




The CPU accepts a software interrupt request during DI status (IFF = 7) because the request has a priority of level 7. The IFF is not changed by the software interrupt.

(5) Timing of interrupt acceptance




If an interrupt with a priority level higher than the interrupt currently being processed is generated, the CPU accepts the interrupt with the higher level. The program counter which returns at [5] is the state address of the INTT0 interrupt routine.

___ (underline): Instruction
[1], [2], …: Execution flow

The addresses 008000H to 0080FFH (256 bytes) of the TMP93CS40 and TMP93CS41 are assigned as interrupt vector areas.

Table 3.4.1 TMP93CS40/TMP93CS41 Interrupt Table

| Default Priority | Type | Interrupt Source | Vector Value "V" | Address Referring to Vector | Micro DMA Start Vector |
|---|---|---|---|---|---|
| 1 | Non-maskable | Reset, or SWI0 instruction | 0000H | 8000H | – |
| 2 | | SWI 1 instruction | 0004H | 8004H | – |
| 3 | | Illegal instruction, or SWI2 | 0008H | 8008H | – |
| 4 | | SWI 3 instruction | 000CH | 800CH | – |
| 5 | | SWI 4 instruction | 0010H | 8010H | – |
| 6 | | SWI 5 instruction | 0014H | 8014H | – |
| 7 | | SWI 6 instruction | 0018H | 8018H | – |
| 8 | | SWI 7 instruction | 001CH | 801CH | – |
| 9 | | NMI: $\overline{NMI}$ pin input | 0020H | 8020H | 08H |
| 10 | | INTWD: Watchdog timer | 0024H | 8024H | 09H |
| 11 | Maskable | INT0: INT0 pin input | 0028H | 8028H | 0AH |
| 12 | | INT4: INT4 pin input | 002CH | 802CH | 0BH |
| 13 | | INT5: INT5 pin input | 0030H | 8030H | 0CH |
| 14 | | INT6: INT6 pin input | 0034H | 8034H | 0DH |
| 15 | | INT7: INT7 pin input | 0038H | 8038H | 0EH |
| – | | (Reserved) | 003CH | 803CH | – |
| 16 | | INTT0: 8-bit timer 0 | 0040H | 8040H | 10H |
| 17 | | INTT1: 8-bit timer 1 | 0044H | 8044H | 11H |
| 18 | | INTT2: 8-bit timer 2/PWM 0 | 0048H | 8048H | 12H |
| 19 | | INTT3: 8-bit timer 3/PWM 1 | 004CH | 804CH | 13H |
| 20 | | INTTR4: 16-bit timer 4 (TREG4) | 0050H | 8050H | 14H |
| 21 | | INTTR5: 16-bit timer 4 (TREG5) | 0054H | 8054H | 15H |
| 22 | | INTTR6: 16-bit timer 5 (TREG6) | 0058H | 8058H | 16H |
| 23 | | INTTR7: 16-bit timer 5 (TREG7) | 005CH | 805CH | 17H |
| 24 | | INTRX0: Serial receive (Channel 0) | 0060H | 8060H | 18H |
| 25 | | INTTX0: Serial send (Channel 0) | 0064H | 8064H | 19H |
| 26 | | INTRX1: Serial receive (Channel 1) | 0068H | 8068H | 1AH |
| 27 | | INTTX1: Serial send (Channel 1) | 006CH | 806CH | 1BH |
| 28 | | INTAD: AD conversion completion | 0070H | 8070H | 1CH |
| – | | (Reserved) | 0074H | 8074H | – |
| | | | to | to | to |
| – | | (Reserved) | 00FCH | 80FCH | – |

Setting to reset and interrupt vectors

1. Reset vector

| Address | |
|---|---|
| 8000H | PC<7:0> |
| 8001H | PC<15:8> |
| 8002H | PC<23:16> |
| 8003H | XX |
| | |

The vector base addresses are dependent on the products.

| Type No. | Vector Base Address | PC Setting Sequence after Reset | Notes |
|---|---|---|---|
| TMP93CS40/CS41<br>TMP93CM40<br>TMP93PS40<br>TMP93CW40/CW41<br>TMP93PW40 | 008000H | PC (7:0) ← Data in location 8000H<br>PC (15:8) ← Data in location 8001H<br>PC (23:16) ← Data in location 8002H | P27 to P20 and A23 to A16 pins are defined as input ports and are pulled down in resetting. The logic data item is 00H. When port 2 is used for the A23 to A16 pins to access the program ROM, set PC (23 to 16) to 00H and set the reset vector to lie within the area 0000H to FFFFH. (This is applicable mainly to products without ROM.) |

2. Interrupt vector (except reset vector)

| Address refers to vector | |
|---|---|
| +0 | PC<7:0> |
| +1 | PC<15:8> |
| +2 | PC<23:16> |
| +3 | XX |
| | |

XX: Don't care

Setting example:

Set the reset vector to 8100H, NMI vector to 9ABCH and INTAD vector to 123456H.

```
ORG     8000H
DL      008100H     ; Reset = 8100H

ORG     8020H
DL      009ABCH     ; NMI = 9ABCH

ORG     8070H
DL      123456H     ; INTAD = 123456H

ORG     8100H
LD      A, B
   ⁞
ORG     9ABCH
LD      B, C
   ⁞
ORG     123456H
LD      C, A
   ⁞
```

Note:
ORG and DL are assembler directives.
ORG: Control location counter
DL: Defines long word (32-bit) data

### 3.4.2 Micro DMA

In addition to the conventional interrupt processing, the TMP93CS40 and TMP93CS41 also have a micro DMA function. When an interrupt is accepted, in addition to an interrupt vector, the CPU receives data indicating whether it is to be processed in micro DMA mode or in general-purpose interrupt mode. The CPU performs micro DMA processing only if that mode is requested.

The micro DMA of the TMP93CS40 and TMP93CS41 can process at very high speed compared with the TLCS-90 micro DMA because it has transfer parameters in dedicated registers in the CPU. Since those dedicated registers are assigned as CPU control registers, they can only be accessed by the LDC instruction.

(1) Micro DMA operation

Micro DMA operation starts when the accepted interrupt vector value matches the micro DMA start vector value set in the interrupt controller. The micro DMA has four channels, so that it can be set for up to four types of interrupt sources at the same time.

When a micro DMA interrupt is accepted, data are automatically transferred from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decremented. If the value in the counter after decrementing is other than “0”, micro DMA processing is completed; if the value in the counter after decrementing is “0”, general-purpose interrupt processing is performed. In read-only mode, which provides for DRAM refresh, the value in the counter is ignored and a dummy read operation is repeated.

32-bit control registers are used for setting transfer source and destination addresses. However, the TMP93CS40 and TMP93CS41 have only 24 address pins for output. A 16-Mbyte space is available for the micro DMA.

There are two data transfer modes: One-byte mode and one-word mode. Incrementing, decrementing, and fixing the transfer source and destination addresses after transfer can be done in both modes. Therefore data can easily be transferred between I/O and memory and between different I/Os. For details of transfer modes, see the description of transfer mode registers.

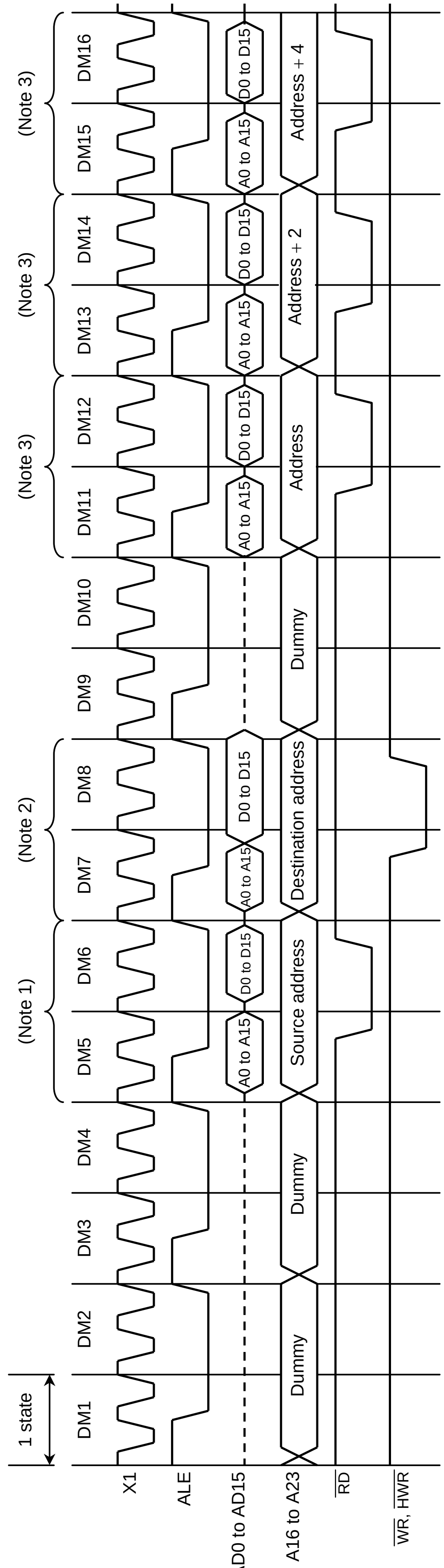
The transfer counter has 16 bits, so up to 65536 transfers (The maximum when the initial value of the transfer counter is 0000H) can be performed for one interrupt source by micro DMA processing.

When the transfer counter is decremented to “0” after data are transferred by the micro DMA, general-purpose interrupt processing is performed. After processing the general-purpose interrupt, restarting the interrupts of the same channel restarts the transfer counter from 65536. It is necessary to reset the transfer counter in the general-purpose interrupt processing routine.

Interrupt sources handled by micro DMA processing are 20 in total, and the micro DMA start vectors are listed in Table 3.4.1.

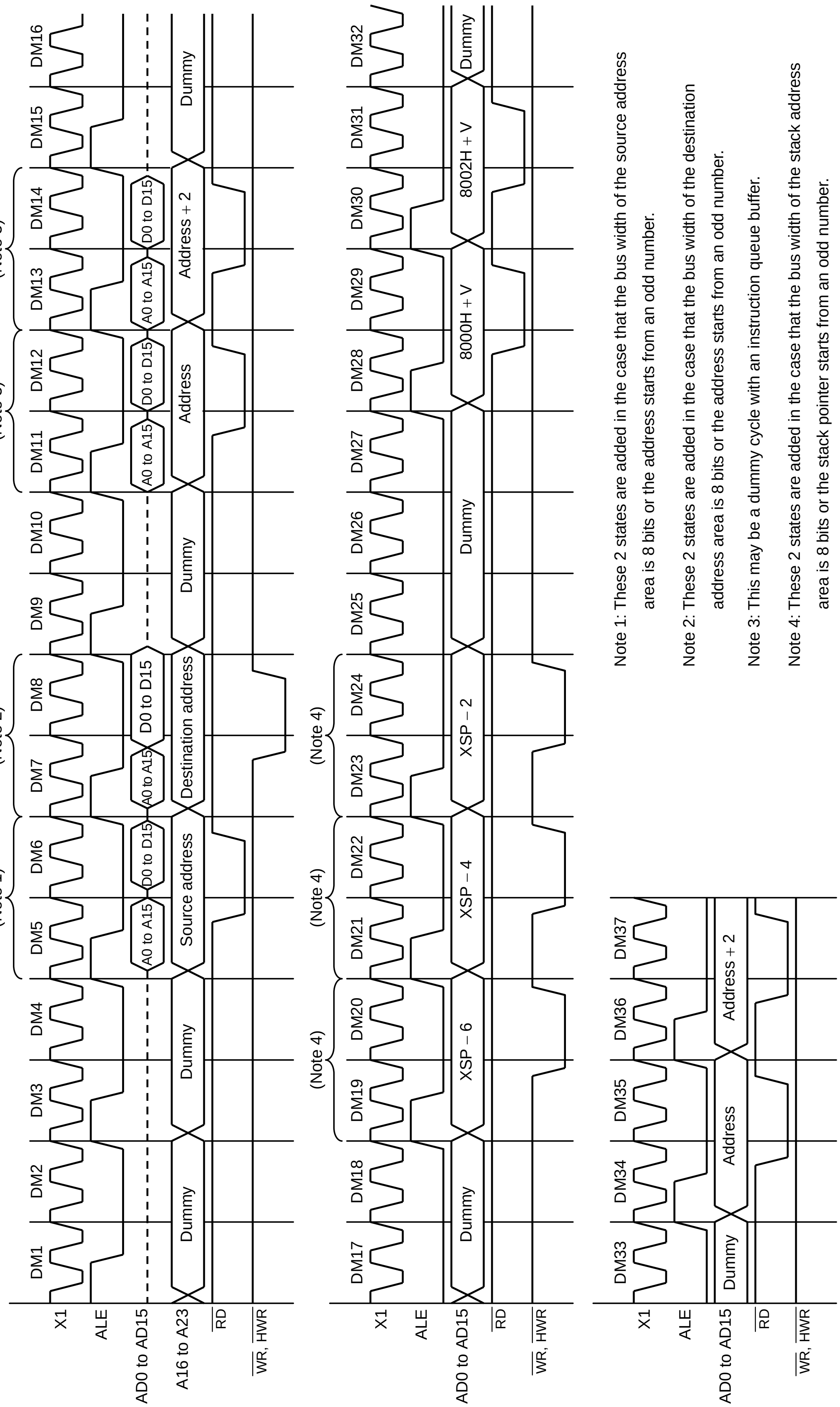
The following timing chart is a micro DMA cycle of the transfer address increment (INC) mode (The other modes are the same as this except for the read-only mode).

(Conditions: MAX mode, 16-bit bus width for 16 Mbytes, 0 waits.)




Note 1: These 2 states are added in the case that the bus width of the source address area is 8 bits.
Note 2: These 2 states are added in the case that the bus width of the destination address area is 8 bits.
Note 3: This may be a dummy cycle with an instruction queue buffer.

Figure 3.4.2 Micro DMA Cycle (Count ≠ 0)




Note 1: These 2 states are added in the case that the bus width of the source address area is 8 bits or the address starts from an odd number.

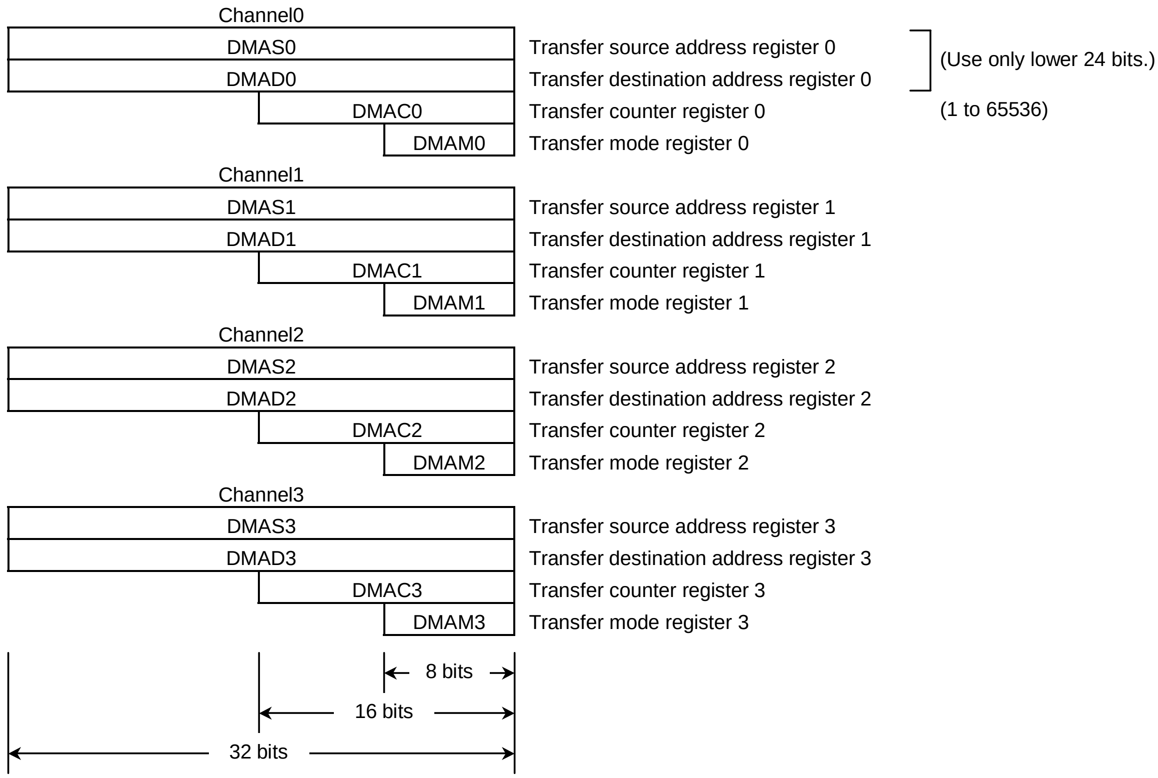
Note 2: These 2 states are added in the case that the bus width of the destination address area is 8 bits or the address starts from an odd number.

Note 3: This may be a dummy cycle with an instruction queue buffer.

Note 4: These 2 states are added in the case that the bus width of the stack address area is 8 bits or the stack pointer starts from an odd number.

Figure 3.4.3 Micro DMA Cycle (Count = 0)

(2) Register configuration (CPU control registers)




These control registers can only be set with the "LDC cr, r" instruction.

Example:

```
LD    XWA, 100H
LDC   DMAS0, XWA
LD    XWA, 50H
LDC   DMAD0, XWA
LD    WA, 40H
LDC   DMAC0, WA
LD    A, 05H
LDC   DMAM0, A
```

(3) Transfer mode register details

| 0 | 0 | 0 | 0 | Mode | | | |
|---|---|---|---|---|---|---|---|

Note: When setting values for this register, set the upper 4 bits to 0.

Z: 0 = byte transfer, 1 = word transfer

| | | | | | Execution time |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Z | Transfer destination address INC mode .................................... for I/O to memory<br>(DMADn+)←(DMASn)<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 16 states<br>(1.6 μs) |
| 0 | 0 | 1 | Z | Transfer destination address DEC mode .................................. for I/O to memory<br>(DMADn–)←(DMASn)<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 16 states<br>(1.6 μs) |
| 0 | 1 | 0 | Z | Transfer source address INC mode.......................................... for memory to I/O<br>(DMADn)←(DMASn+)<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 16 states<br>(1.6 μs) |
| 0 | 1 | 1 | Z | Transfer source address DEC mode ........................................ for memory to I/O<br>(DMADn)←(DMASn–)<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 16 states<br>(1.6 μs) |
| 1 | 0 | 0 | Z | Fixed address mode................................................................. I/O to I/O<br>(DMADn)←(DMASn)<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 16 states<br>(1.6 μs) |
| 1 | 0 | 1 | 0 | Read-only mode........................................................................ for DRAM refresh<br>Dummy←(DMASn) ; Reads 4 bytes.<br>DMASn←DMASn + 4 ; Increments lower word only.<br>DMACn←DMACn – 1 | 14 states<br>(1.4 μs) |
| 1 | 0 | 1 | 1 | Counter mode .......................................................................... for interrupt counter<br>DMASn←DMASn + 1<br>DMACn←DMACn – 1<br>if DMACn = 0 then INT. | 11 states<br>(1.1 μs) |

Note 1: n: corresponds to micro DMA channels 0 to 3.
DMADn+/DMASn+: Post-increment (Increments register value after transfer.)
DMADn–/DMASn–: Post-decrement (Decrements register value after transfer.)

Note 2: Execution time: When setting source address/destination address area to 16-bit bus, 0 waits. Clock condition: fc = 20 MHz, clock gear: 1 (fc)

Note 3: Do not use any codes for transfer mode registers other than those indicated above.

<Example for usage of read-only mode (DRAM refresh)>

* Clock condition

System clock: fc
Clock gear: 1 (fc)

When the hardware configuration is as follows:

DRAM mapping size: = 1 Mbyte
DRAM data bus size: = 8 bits
DRAM mapping address range: = 100000H to 1FFFFFH

Set the following registers first; refresh is performed automatically.

1. Register initial value setting

```
LD    XIX,   100000H
LDC   DMAS0, XIX       ...  Mapping start address
LD    A,     00001010B
LDC   DMAM0, A         ...  Read only mode (for DRAM refresh)
```

2. Timer setting

   Set the timers so that interrupts are generated at intervals of 62.5 μs or less.

3. Interrupt controller setting

   Set the timer interrupt mask higher than the mask for other interrupts. Write the above timer interrupt vector value into the micro DMA start vector register, DMA0V.

(Operation description)

The DRAM data bus is an 8-bit bus and the micro DMA is in read-only mode (4 bytes), so refresh is performed four times per interrupt.

When a 512-refresh per 8 ms DRAM is connected, DRAM refresh is performed sufficiently if the micro DMA is started every 15.625 μs × 4 = 62.5 μs or less, since the timing is 15.625 μs per refresh.

(Overhead)

Each processing time for read-only mode by the micro DMA is 1.8 μs (18 states) at 20 MHz with an 8-bit data bus.

In the above example, the micro DMA is started every 62.5 μs, and 1.8 μs ÷ 62.5 μs = 0.0288; thus, the overhead factor is 2.88%.

Note: When the bus which must wait to accept the interrupt is released ($\overline{\text{BUSAK}}$ = "0"), DRAM refresh is not performed because the micro DMA is generated by an interrupt.

### 3.4.3 Interrupt Controller

Figure 3.4.4 is a block diagram of the interrupt circuits. The left half of the diagram shows the interrupt controller; the right half includes the CPU interrupt request signal circuit and the halt release signal circuit.

Each interrupt channel (Total of 20 channels) in the interrupt controller has an interrupt request flag, interrupt priority setting register, and a register for storing the micro DMA start vector. The interrupt request flag is used to latch interrupt requests from peripheral devices.

The flag is cleared to 0 when any of the following conditions are met.

- Upon resetting
- When the CPU reads the interrupt vector after acceptance of an interrupt.
- When the CPU executes an instruction that clears the interrupt from that channel (Writes 0 in <IxxC> of the interrupt priority setting register).

For example, to clear the INT0 interrupt request, after the DI instruction set the register INTE0AD as follows.

INTE0AD ← − − − − 0 − − −     Clears the INT0 flip-flop.

The status of the interrupt request flag is detected by reading the corresponding clear bit. This also allows the interrupt to be identified by the software.

The interrupt priority can be set by writing the priority in the interrupt priority setting register (e.g., INTE0AD or INTE45) provided for each interrupt source. Interrupt priority levels to be set range from or 1 to 6. Except for NMIs (Non-maskable interrupts), writing 0 or 7 as the interrupt priority disables the corresponding interrupt request. The priority of non-maskable interrupt sources ($\overline{\text{NMI}}$ pin, watchdog timer, etc.) is fixed to 7. If interrupt requests with the same interrupt level are generated simultaneously, interrupts are accepted in accordance with the default ranking of priorities.

The interrupt controller selects the interrupt request with the highest priority among the simultaneous interrupts, and sends it and its vector address to the CPU. The CPU compares the priority value <IFF2:0> set in the status register, with the priority value sent by the interrupt request signal; if the latter is higher, the interrupt is accepted. Then the CPU sets in CPU SR<IFF2:0> a value equal to one plus the priority value of the interrupt request just received. Interrupt requests whose priority values equal or are higher than the value set in the register are accepted concurrently with execution of the previous interrupt routine. When interrupt processing is completed (after execution of the RETI instruction), the CPU restores to CPU SR<IFF2:0> the priority value saved in the stack before the interrupt was generated.

The interrupt controller also has four registers used to store the micro DMA start vector. Unlike other micro DMA registers (DMAS, DMAD, DMAM, and DMAC), these are I/O registers. Writing the start vector of the interrupt source for micro DMA processing (See Table 3.4.1), enables the corresponding interrupt to be processed by micro DMA. Please note that appropriate values must be set in the micro DMA parameter registers (e.g., DMAS and DMAD) prior to the beginning of micro DMA processing.

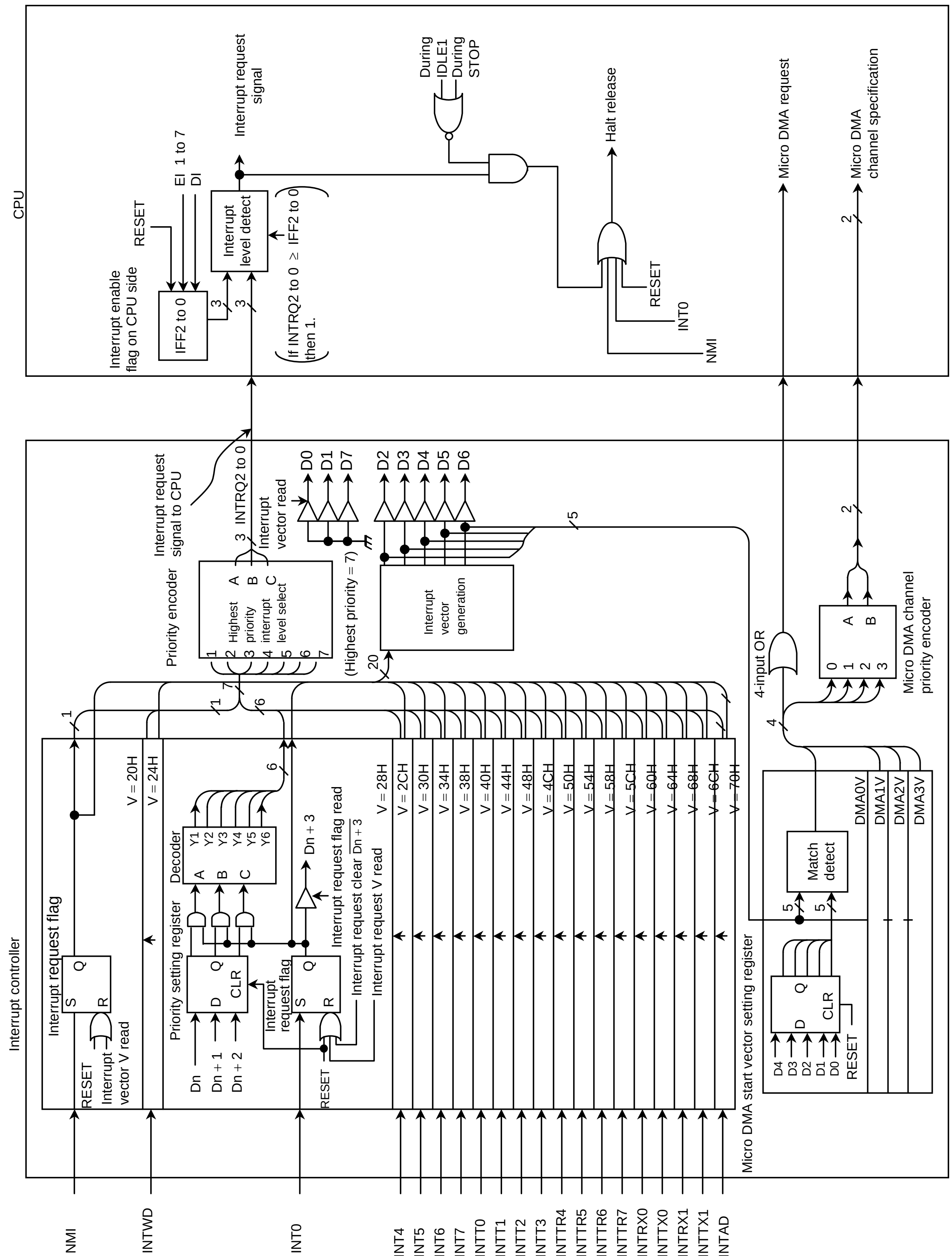



Figure 3.4.4 Block Diagram of Interrupt Controller

(1) Interrupt priority setting register

| Symbol | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| INTE0AD | 0070H | INTAD | | | | INT0 | | | | ←Interrupt source |
| | | IADC | IADM2 | IADM1 | IADM0 | I0C | I0M2 | I0M1 | I0M0 | ←Bit symbol |
| | | R/W | W | | | R/W | W | | | ←Read/Write |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ←After reset |
| INTE45 | 0071H | INT5 | | | | INT4 | | | | |
| | | I5C | I5M2 | I5M1 | I5M0 | I4C | I4M2 | I4M1 | I4M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTE67 | 0072H | INT7 | | | | INT6 | | | | |
| | | I7C | I7M2 | I7M1 | I7M0 | I6C | I6M2 | I6M1 | I6M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTET10 | 0073H | INTT1 (Timer1) | | | | INTT0 (Timer0) | | | | |
| | | IT1C | IT1M2 | IT1M1 | IT1M0 | IT0C | IT0M2 | IT0M1 | IT0M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTEPW10 | 0074H | INTT3 (Timer3/PWM1) | | | | INTT2 (Timer2/PWM0) | | | | |
| | | IPW1C | IPW1M2 | IPW1M1 | IPW1M0 | IPW0C | IPW0M2 | IPW0M1 | IPW0M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTET54 | 0075H | INTTR5 (TREG5) | | | | INTTR4 (TREG4) | | | | |
| | | IT5C | IT5M2 | IT5M1 | IT5M0 | IT4C | IT4M2 | IT4M1 | IT4M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTET76 | 0076H | INTTR7 (TREG7) | | | | INTTR6 (TREG6) | | | | |
| | | IT7C | IT7M2 | IT7M1 | IT7M0 | IT6C | IT6M2 | IT6M1 | IT6M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTES0 | 0077H | INTTX0 | | | | INTRX0 | | | | |
| | | ITX0C | ITX0M2 | ITX0M1 | ITX0M0 | IRX0C | IRX0M2 | IRX0M1 | IRX0M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTES1 | 0078H | INTTX1 | | | | INTRX1 | | | | |
| | | ITX1C | ITX1M2 | ITX1M1 | ITX1M0 | IRX1C | IRX1M2 | IRX1M1 | IRX1M0 | |
| | | R/W | W | | | R/W | W | | | |
| | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| IxxM2 | IxxM1 | IxxM0 | Function (Write) |
|---|---|---|---|
| 0 | 0 | 0 | Prohibits interrupt request. |
| 0 | 0 | 1 | Sets interrupt request level to “1”. |
| 0 | 1 | 0 | Sets interrupt request level to “2”. |
| 0 | 1 | 1 | Sets interrupt request level to “3”. |
| 1 | 0 | 0 | Sets interrupt request level to “4”. |
| 1 | 0 | 1 | Sets interrupt request level to “5”. |
| 1 | 1 | 0 | Sets interrupt request level to “6”. |
| 1 | 1 | 1 | Prohibits interrupt request. |

| IxxC | Function (Read) | Function (Write) |
|---|---|---|
| 0 | Indicates no interrupt request. | Clears interrupt request flag. |
| 1 | Indicates interrupt request. | Don’t care |

Note 1: Read-modify-write is prohibited.

Note 2: This note is about clearing interrupt request flags. The interrupt request flags of INTAD, INTRX0, and INTRX1 are not cleared by writing “0” to IxxC because they are level-sense interrupts.

Figure 3.4.5 Interrupt Priority Setting Register

(2) External interrupt control

Interrupt Input Mode Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| IIMC (007BH) | Bit symbol | | | | | | I0IE | I0LE | NMIREE |
| | Read/Write | | | | | | W | W | W |
| | After reset | | | | | | 0 | 0 | 0 |
| | Function | | | | | | 1: INT0 input enable | 0: INT0 edge-sense mode<br>1: INT0 level-sense mode | 1: Can be accepted in $\overline{\text{NMI}}$ rising edge. |

INT0 input enable (Note 1)

| | |
|---|---|
| 0 | INT0 disable (P87 function only) |
| 1 | Input enable |

$\overline{\text{NMI}}$ rising edge enable

| | |
|---|---|
| 0 | Interrupt request generation at falling edge |
| 1 | Interrupt request generation at rising and falling edge |

INT0 level enable (Note 2)

| | |
|---|---|
| 0 | Rising edge detect interrupt |
| 1 | High level interrupt |

Note 1: The INT0 pin can also be used for standby release as described in section 3.3.4. Even if the pin is not used for standby release, setting this register to "0" maintains the port function during standby mode.

Note 2: This is a case of changing from level-sence to edge-sence for INT0 pin mode. Execution example:

```
LD (INTE0AD)   ,xxxx0000B    ; INT0 disable, clear the request flag.
LD (IIMC)      ,xxxxx10xB    ; Change from level to edge.
LD (INTE0AD)   ,xxxx0nnnB    ; Set interrupt level "n" for INT0, clear
                               the request flag.
```

Note 3: Read-modify-write is prohibited.

Note 4: IMC<bit7:3> are always read as "1".

Note 5: See electrical characteristics in section 4 for external interrupt input pulse.

Figure 3.4.6 Interrupt Input Mode Control Register

Table 3.4.2 Setting of External Interrupt Pin Function

| Interrupt | Pin Name | Mode | Setting Method |
|---|---|---|---|
| $\overline{\text{NMI}}$ | – | Falling edge | IIMC<NMIREE> = 0 |
| | | Falling and rising edges | IIMC<NMIREE> = 1 |
| INT0 | P87 | Rising edge | IIMC<I0LE> = 0, <I0IE> = 1 |
| | | High level | IIMC<I0LE> = 1, <I0IE> = 1 |
| INT4 | P80 | Rising edge | T4MOD<CAP12M1:0> = 0, 0 or 0, 1 or 1, 1 |
| | | Falling edge | T4MOD<CAP12M1:0> = 1, 0 |
| INT5 | P81 | Rising edge | ――――― |
| INT6 | P84 | Rising edge | T5MOD<CAP34M1:0> = 0, 0 or 0, 1 or 1, 1 |
| | | Falling edge | T5MOD<CAP34M1:0> = 1, 0 |
| INT7 | P85 | Rising edge | ――――― |

(3) Micro DMA start vector

When the CPU reads the interrupt vector after accepting an interrupt, it simultaneously compares the bits 2 to 6 of the interrupt vector with each channel's micro DMA start vector. When the two match, the interrupt from the channel whose value matched is processed in micro DMA mode.

If the interrupt vector matches more than one channel, the channel with the lower channel number has a higher priority.

Micro DMA 0 State Vector

DMA0V (007CH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | DMA0V4 | DMA0V3 | DMA0V2 | DMA0V1 | DMA0V0 |
| Read/Write | | | | W | | | | |
| After reset | | | | 0 | 0 | 0 | 0 | 0 |
| Function | Micro DMA channel 0 processed by matching bits 2 to 6 of the interrupt vector. | | | | | | | |

Micro DMA 1 State Vector

DMA1V (007DH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | DMA1V4 | DMA1V3 | DMA1V2 | DMA1V1 | DMA1V0 |
| Read/Write | | | | W | | | | |
| After reset | | | | 0 | 0 | 0 | 0 | 0 |
| Function | Micro DMA channel 1 processed by matching bits 2 to 6 of the interrupt vector. | | | | | | | |

Micro DMA 2 State Vector

DMA2V (007EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | DMA2V4 | DMA2V3 | DMA2V2 | DMA2V1 | DMA2V0 |
| Read/Write | | | | W | | | | |
| After reset | | | | 0 | 0 | 0 | 0 | 0 |
| Function | Micro DMA channel 2 processed by matching bits 2 to 6 of the interrupt vector. | | | | | | | |

Micro DMA 3 State Vector

DMA3V (007FH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | DMA3V4 | DMA3V3 | DMA3V2 | DMA3V1 | DMA3V0 |
| Read/Write | | | | W | | | | |
| After reset | | | | 0 | 0 | 0 | 0 | 0 |
| Function | Micro DMA channel 3 processed by matching bits 2 to 6 of the interrupt vector. | | | | | | | |

Note: Read-modify-write is not possible for DMA0V to DMA3V.

Figure 3.4.7 Micro DMA State Vector Register

(4) Notes

The instruction execution unit and the bus interface unit of this CPU operate independently of each other. Therefore, if the instruction used to clear the interrupt request flag of an interrupt is fetched before the interrupt is generated, it is possible that the CPU might accept the interrupt and execute the fetched instruction to clear the interrupt request flag while reading the interrupt vector. If so, the CPU would start the interrupt processing from the address "8028H".

To avoid the above occurring, clear the interrupt request flag by entering the instruction to clear the flag after the DI instruction. In the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing instruction and following more than one instruction are executed. When EI instruction is placed immediately after clearing instruction, an interrupt becomes enable before interrupt request flags are cleared.

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

## 3.5 Functions of Ports

The TMP93CS40 has 79 bits for I/O ports. The TMP93CS41 has 61 bits for I/O ports because port 0, port 1, P30, and P31 are dedicated pins for AD0 to AD7, AD8 to AD15 or A8 to A15, $\overline{RD}$, and $\overline{WR}$.

These port pins have I/O functions for the built-in CPU and internal I/Os as well as general-purpose I/O port functions. Table 3.5.1 lists the function of each port pin. Table 3.5.2 lists I/O registers and their specifications.

Table 3.5.1 Functions of Ports

| Port No. | Pin No. | Number of Pins | Direction | R | Direction Setting Unit | Pin Name for Built-in Function |
|---|---|---|---|---|---|---|
| Port 0 | P00 to P07 | 8 | I/O | – | Bit | AD0 to AD7 |
| Port 1 | P10 to P17 | 8 | I/O | – | Bit | AD8 to AD15/A8 to A15 |
| Port 2 | P20 to P27 | 8 | I/O | ↓ | Bit | A0 to A7/A16 to A23 |
| Port 3 | P30 | 1 | Output | – | (Fixed) | $\overline{RD}$ |
| | P31 | 1 | Output | – | (Fixed) | $\overline{WR}$ |
| | P32 | 1 | I/O | ↑ | Bit | $\overline{HWR}$ |
| | P33 | 1 | I/O | ↑ | Bit | $\overline{WAIT}$ |
| | P34 | 1 | I/O | ↑ | Bit | $\overline{BUSRQ}$ |
| | P35 | 1 | I/O | ↑ | Bit | $\overline{BUSAK}$ |
| | P36 | 1 | I/O | ↑ | Bit | R/$\overline{W}$ |
| | P37 | 1 | I/O | ↑ | Bit | $\overline{RAS}$ |
| Port 4 | P40 | 1 | I/O | ↑ | Bit | $\overline{CS0}$ / $\overline{CAS0}$ |
| | P41 | 1 | I/O | ↑ | Bit | $\overline{CS1}$ / $\overline{CAS1}$ |
| | P42 | 1 | I/O | ↓ | Bit | $\overline{CS2}$ / $\overline{CAS2}$ |
| Port 5 | P50 to P57 | 8 | Input | – | (Fixed) | AN0 to AN7 |
| Port 6 | P60 to P67 | 8 | I/O | ↑ | Bit | PG00 to PG03, PG10 to PG13 |
| Port 7 | P70 | 1 | I/O | ↑ | Bit | TI0 |
| | P71 | 1 | I/O | ↑ | Bit | TO1 |
| | P72 | 1 | I/O | ↑ | Bit | TO2 |
| | P73 | 1 | I/O | ↑ | Bit | TO3 |
| Port 8 | P80 | 1 | I/O | ↑ | Bit | TI4/INT4 |
| | P81 | 1 | I/O | ↑ | Bit | TI5/INT5 |
| | P82 | 1 | I/O | ↑ | Bit | TO4 |
| | P83 | 1 | I/O | ↑ | Bit | TO5 |
| | P84 | 1 | I/O | ↑ | Bit | TI6/INT6 |
| | P85 | 1 | I/O | ↑ | Bit | TI7/INT7 |
| | P86 | 1 | I/O | ↑ | Bit | TO6 |
| | P87 | 1 | I/O | ↑ | Bit | INT0 |
| Port 9 | P90 | 1 | I/O | ↑ | Bit | TXD0 |
| | P91 | 1 | I/O | ↑ | Bit | RXD0 |
| | P92 | 1 | I/O | ↑ | Bit | $\overline{CTS0}$ /SCLK0 |
| | P93 | 1 | I/O | ↑ | Bit | TXD1 |
| | P94 | 1 | I/O | ↑ | Bit | RXD1 |
| | P95 | 1 | I/O | ↑ | Bit | SCLK1 |
| | P96 | 1 | I/O | – | Bit | XT1 |
| | P97 | 1 | I/O | – | Bit | XT2 |
| Port A | PA0 to PA6 | 7 | I/O | – | Bit | |
| | PA7 | 1 | I/O | – | Bit | SCOUT |

R: ↑ = With programmable pull-up resistor

↓ = With programmable pull-down resistor.

Table 3.5.2 I/O Registers and Specifications (1/2)

| Port No. | Pin No. | Function | I/O Register | | |
|---|---|---|---|---|---|
| | | | Pn | PnCR | PnFC |
| Port 0 | P00 to P07 | Input port (Note 1) | × | 0 | None |
| | | Output port (Note 1) | × | 1 | |
| | | AD0 to AD7 bus | × | × | |
| Port 1 | P10 to P17 | Input port (Note 1) | × | 0 | 0 |
| | | Output port (Note 1) | × | 1 | 0 |
| | | AD8 to AD15 bus (Note 2) | × | 0 | 1 |
| | | A8 to A15 output (Note 2) | × | 1 | 1 |
| Port 2 | P20 to P27 | Input port (without PD) | 1 | 0 | 0 |
| | | Input port (with PD) | 0 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | | A0 to A7 output (Note 1) | 1 | 0 | 1 |
| | | A16 to A23 output | 1 | 1 | 1 |
| Port 3 | P30 | Output port (Note 1) | × | None | 0 |
| | | Outputs $\overline{RD}$ only when accessing external space | 1 | | 1 |
| | | Always outputs $\overline{RD}$ | 0 | | 1 |
| | P31 | Output port (Note 1) | × | None | 0 |
| | | Outputs $\overline{WR}$ only when accessing external space | × | | 1 |
| | P32 to P37 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P32 | $\overline{HWR}$ output | × | 1 | 1 |
| | P33 | $\overline{WAIT}$ input (without PU) | 0 | 0 | None |
| | | $\overline{WAIT}$ input (with PU) | 1 | 0 | |
| | P34 | $\overline{BUSRQ}$ input (without PU) | 0 | 0 | 1 |
| | | $\overline{BUSRQ}$ input (with PU) | 1 | 0 | 1 |
| | P35 | $\overline{BUSAK}$ output | × | 1 | 1 |
| | P36 | R/$\overline{W}$ output | × | 1 | 1 |
| | P37 | $\overline{RAS}$ output | × | 1 | 1 |
| Port 4 | P40 to P41 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P42 | Input port (without PD) | 1 | 0 | 0 |
| | | Input port (with PD) | 0 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P40 | $\overline{CS0}$ output (Note 3) | × | 1 | 1 |
| | P41 | $\overline{CS1}$ output (Note 3) | × | 1 | 1 |
| | P42 | $\overline{CS2}$ output (Note 3) | × | 1 | 1 |
| Port 5 | P50 to P57 | Input port | × | None | |
| | | AN0 to AN7 input (Note 4) | × | | |
| Port 6 | P60 to P67 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | | PGn output | × | 1 | 1 |

×: Don't care

Note 1: In the case of the TMP93CS41F, this function is not available.

Note 2: In the case of the TMP93CS41F, this function is fixed by AM8/$\overline{AM16}$ pin.

Note 3: CS/WAIT control register BnCH<BnCAS> selects the wave form output from P40 to P42 pins, $\overline{CS0}$ to $\overline{CS2}$ or $\overline{CAS0}$ to $\overline{CAS2}$.

Note 4: The channel for AD input is selected by ADMOD2<ADCHn> for P50 to P57 pins.

Note 5: PU = pull-up resistor; PD = pull-down resistor.

Table 3.5.3 I/O Registers and Specifications (2/2)

| Port No. | Pin No. | Function | I/O Register | | |
|---|---|---|---|---|---|
| | | | Pn | PnCR | PnFC |
| Port 7 | P70 to P73 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P70 | TI0 input (without PU) | 0 | 0 | None |
| | | TI0 input (with PU) | 1 | 0 | |
| | P71 | TO1 output | × | 1 | 1 |
| | P72 | TO2 output | × | 1 | 1 |
| | P73 | TO3 output | × | 1 | 1 |
| Port 8 | P80 to P87 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P80 | TI4/INT4 input (without PU) | 0 | 0 | None |
| | | TI4/INT4 input (with PU) | 1 | 0 | |
| | P81 | TI5/INT5 input (without PU) | 0 | 0 | None |
| | | TI5/INT5 input (with PU) | 1 | 0 | |
| | P84 | TI6/INT6 input (without PU) | 0 | 0 | None |
| | | TI6/INT6 input (with PU) | 1 | 0 | |
| | P85 | TI7/INT7 input (without PU) | 0 | 0 | None |
| | | TI7/INT7 input (with PU) | 1 | 0 | |
| | P82 | TO4 output | × | 1 | 1 |
| | P83 | TO5 output | × | 1 | 1 |
| | P86 | TO6 output | × | 1 | 1 |
| | P87 (Note 5) | INT0 input (without PU) | 0 | 0 | None |
| | | INT0 input (with PU) | 1 | 0 | |
| Port 9 | P90 to P95 | Input port (without PU) | 0 | 0 | 0 |
| | | Input port (with PU) | 1 | 0 | 0 |
| | | Output port | × | 1 | 0 |
| | P90 | TXD0 output | × | 1 | 1 |
| | P93 | TXD1 output | × | 1 | 1 |
| | P91 | RXD0 input (without PU) | 0 | 0 | None |
| | | RXD0 input (with PU) | 1 | 0 | |
| | P94 | RXD1 input (without PU) | 0 | 0 | None |
| | | RXD1 input (with PU) | 1 | 0 | |
| | P92 | SCLK0 output | × | 1 | 1 |
| | | CTS0/SCLK0 input (without PU) | 0 | 0 | 0 |
| | | CTS0/SCLK0 input (with PU) | 1 | 0 | 0 |
| | P95 | SCLK1 output | × | 1 | 1 |
| | | SCLK1 input (without PU) | 0 | 0 | 0 |
| | | SCLK1 input (with PU) | 1 | 0 | 0 |
| | P96 to P97 | Input port | × | 0 | None |
| | | Output port (Note 6) | × | 1 | |
| | | XT1/2 (Note 7) | × | 0 | |
| Port A | PA0 to PA7 | Input port | × | 0 | None |
| | | Output port | × | 1 | |
| | PA7 | SCOUT output (Note 8) | × | 1 | |

×: Don’t care

Note 5: When the P87 pin is used as INT0, the IIMC register has to be set to enable interrupt.

Note 6: When using P96 to P97 as output ports, output goes through the open-drain buffer.

Note 7: When P96 to P97 are used as XT1 to XT2, the SYSCR0<XTEN> has to be set to “1”.

Note 8: When PA7 is used as SCOUT, the PACR and CKOCR must have the appropriate values written to them.

Resetting makes the port pins listed below function as general-purpose I/O ports.

I/O pins programmable for input or output are then set to function as input ports, except for P96/XT1 and P97/XT2.

A program is needed to set port pins for built-in functions.

Because the TMP93CS41 needs external ROMs, some ports are permanently assigned for memory interfacing.

- P00 to P07 → AD0 to AD7
- P10 to P17 → AD8 to AD15 (or A8 to A15)
- P30 → $\overline{RD}$
- P31 → $\overline{WR}$

* Note about the bus release and programmable pull-up/pull-down I/O ports.

When the bus is released ($\overline{BUSAK}$ = "0"), the output buffers of AD0 to AD15 and A0 to A23, as well as the control signals ($\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, R/$\overline{W}$, $\overline{RAS}$, $\overline{CS0}$/$\overline{CAS0}$ to $\overline{CS2}$/$\overline{CAS2}$) are all set to OFF and they go into a high-impedance state.

However, the states of the built-in programmable pull-up/pull-down resistors are retained when the bus is released. These programmable pull-up/pull-down resistors can be switched ON or OFF by programming when they are used as input ports.

When they are used as output ports, they cannot be switched by programming.

Table 3.5.4 shows the pin states when the bus is released ($\overline{BUSAK}$ = "0")

Table 3.5.4 Pin States (when the bus is released)

| Pin Name | Pin States (when the bus is released) | |
|---|---|---|
| | Used as a Port | Used for a Function |
| P00 to P07<br>(AD0 to AD7)<br>P10 to P17<br>(AD8 to AD15/A8 to A15) | The state is not changed.<br>(does not go to high-impedance (High-Z).) | Goes to high-impedance (High-Z). |
| P30 ($\overline{RD}$)<br>P31 ($\overline{WR}$) | Goes to high-impedance (High-Z). | Goes to high-impedance (High-Z). |
| P32 ($\overline{HWR}$)<br>P37 ($\overline{RAS}$) | The output buffer is OFF.<br>The programmable pull-up resistor is ON only in the case that the output latch is equal to "1". | The output buffer is OFF.<br>The programmable pull-up resistor is ON irrespective of the output latch. |
| P36 (R/$\overline{W}$)<br>P40 ($\overline{CS0}$/$\overline{CAS0}$)<br>P41 ($\overline{CS1}$/$\overline{CAS1}$) | The output buffer is OFF.<br>The programmable pull-up resistor is ON only in the case that the output latch is equal to "1". | The output buffer is OFF.<br>The state of the programmable pull-up resistor is retained when the bus is released. |
| P42 ($\overline{CS2}$/$\overline{CAS2}$) | The output buffer is OFF.<br>The programmable pull-down resistor is ON only in the case that the output latch is equal to "0". | The output buffer is OFF.<br>The state of the programmable pull-down resistor is retained when the bus is released. |
| P20 to P27 (A16 to A23) | The state is not changed.<br>(does not go to high-impedance (High-Z).) | The output buffer is OFF.<br>The programmable pull-down resistor is ON only in the case that the output latch is equal to "0". |

Figure 3.5.1 shows an example of an interface circuit using some of the pins described in Table 3.5.4, in a case when the bus releasing function is used.

When the bus is released, neither internal memory nor internal I/O can be accessed. However, the internal I/O continues to operate, so the watchdog timer (WDT) also continues to run. Therefore, be careful about bus releasing time and setting of the detection time of the WDT.

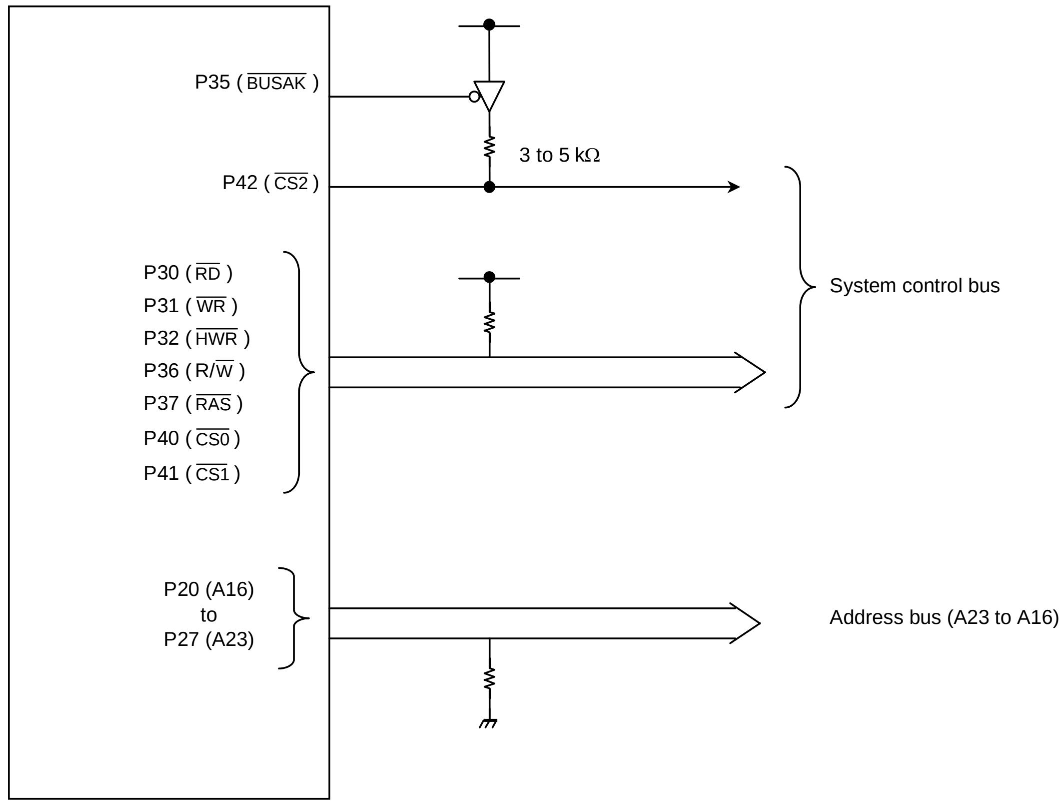



Figure 3.5.1 Example of an Interface Circuit using the Bus Releasing Function

A circuit like the one shown above is needed to fix the signal level in the case when the bus is released.

Resetting sets P30 ($\overline{RD}$) and P31 ($\overline{WR}$) to output; P40 ($\overline{CS0}$), P41 ($\overline{CS1}$), P32 ($\overline{HWR}$), P36 (R/$\overline{W}$), P37 ($\overline{RAS}$), and P35 ($\overline{BUSAK}$) all to input with pull-up resistor; as well as P42 ($\overline{CS2}$) and P20 to P27 (A16 to A23) to input with pull-down resistor.

A circuit like the one above is also needed to fix the signal level after resetting, because of the possibility of conflict between the external pull-up resistor and the internal pull-down resistor. The resistance of the external pull-up resistor must be 3 to 5 kΩ, and the resistance of the internal pull-down resistor is about 50 to 150 kΩ.

Using a pull-down resistor is recommended for P20 to P27 (A16 to A23); however, if this is not possible, a switching circuit like the one used for P42 ($\overline{CS2}$) may be used.

### 3.5.1 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using the control register P0CR. Resetting resets all bits of P0CR to “0”, and sets port 0 to input mode.

In addition to functioning as a general-purpose I/O port, port 0 also functions as an address data bus (AD0 to AD7). To access external memory, port 0 functions as an address data bus (AD0 to AD7), and all bits of the control register P0CR are cleared to “0”.

In the TMP93CS41, which needs external ROMs, port 0 always functions as an address data bus (AD0 to AD7) regardless of the value set in control register P0CR.

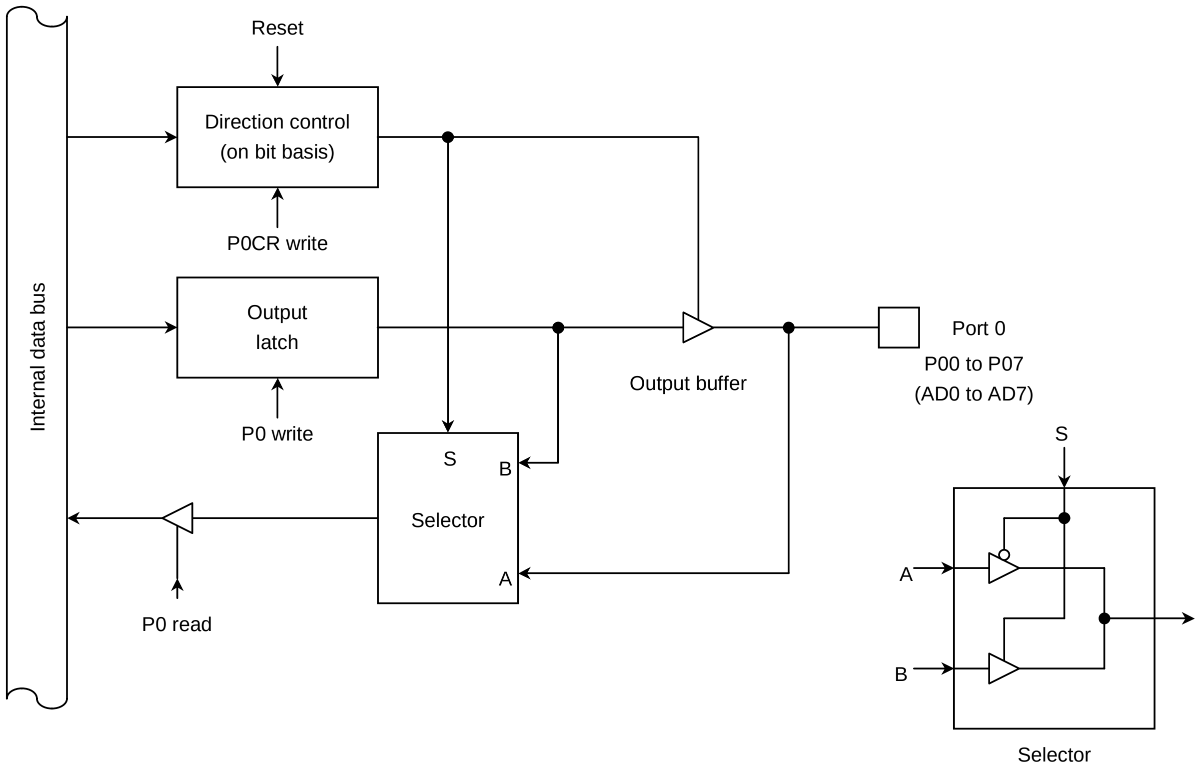



Figure 3.5.2 Port 0

### 3.5.2 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using control register P1CR and function register P1FC. Resetting resets all bits of output latch P1, control register P1CR, and function register P1FC to 0, and sets port 1 to input mode.

In addition to functioning as a general-purpose I/O port, port 1 also functions as an address data bus (AD8 to AD15) or an address bus (A8 to A15).

In the TMP93CS41, which needs external ROMs, port 1 always functions either as an address data bus (AD8 to AD15) when AM8/$\overline{\text{AM16}}$ = "0", or as an address bus (A8 to A15) when AM8/$\overline{\text{AM16}}$ = "1", regardless of the value set in control register P1CR.

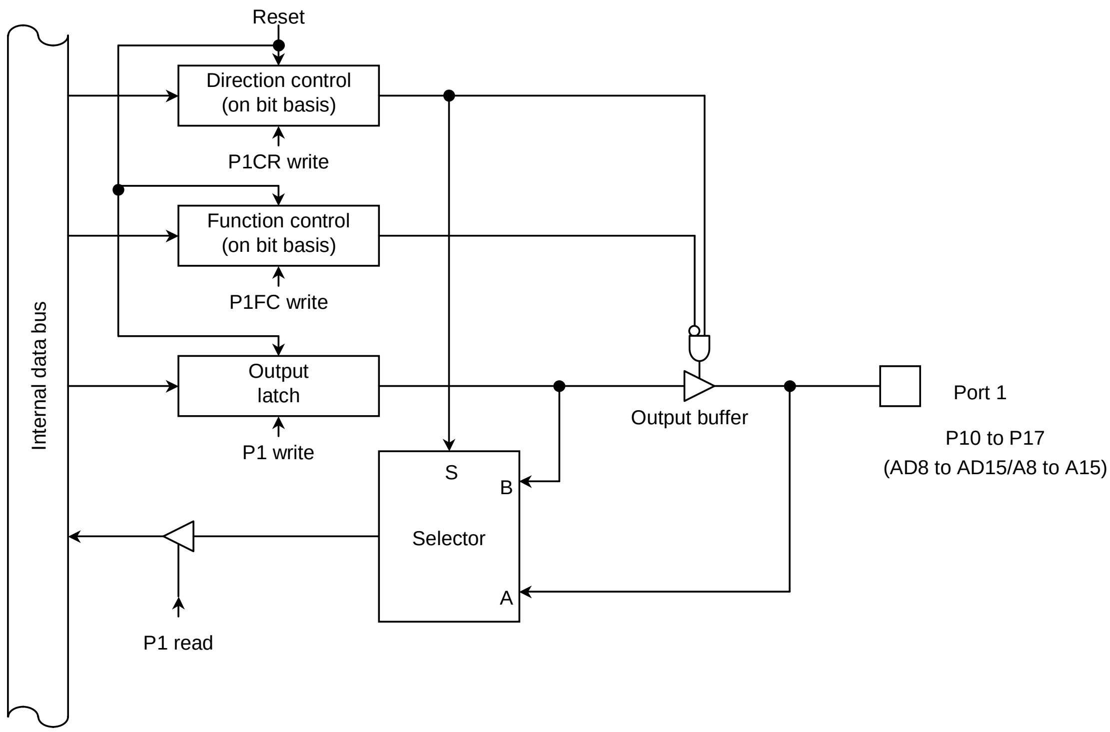



Figure 3.5.3 Port 1

Port 0 Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P0 | Bit symbol | P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |
| (0000H) | Read/Write | R/W | | | | | | | |
| | After reset | Input mode (Output latch register becomes undefined.) | | | | | | | |

Port 0 Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P0CR | Bit symbol | P07C | P06C | P05C | P04C | P03C | P02C | P01C | P00C |
| (0002H) | Read/Write | R/W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | 0: Input 1: Output (when externally accessed, port 0 becomes AD7 to AD0 and P0CR is cleared to 0.) | | | | | | | |

Port 0 I/O setting

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

Port 1 Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P1 | Bit symbol | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| (0001H) | Read/Write | R/W | | | | | | | |
| | After reset | Input mode (Output latch register is cleared to “0”.) | | | | | | | |

Port 1 Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P1CR | Bit symbol | P17C | P16C | P15C | P14C | P13C | P12C | P11C | P10C |
| (0004H) | Read/Write | R/W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | <<See table below.>> | | | | | | | |

Port 1 Function Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P1FC | Bit symbol | P17F | P16F | P15F | P14F | P13F | P12F | P11F | P10F |
| (0005H) | Read/Write | R/W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | <<See table below.>> | | | | | | | |

Note 1: Read-modify-write is prohibited for registers P0CR, P1CR, and P1FC.
Note 2: <P1XF> is bit X in register P1FC; <P1XC> is bit X in register P1CR.

Port 1 function setting

| P1CR <P1XC> \ P1FC<P1XF> | 0 | 1 |
|---|---|---|
| 0 | Input port | Address data bus (AD15 to AD8) |
| 1 | Output port | Address bus (A15 to A8) |

Figure 3.5.4 Registers for Ports 0 and 1

### 3.5.3 Port 2 (P20 to P27)

Port 2 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using the control register P2CR and function register P2FC. Resetting resets all bits of output latch P2, control register P2CR and function register P2FC to “0”. It also sets port 2 to input mode and connects a pull-down resistor.

In addition to functioning as a general-purpose I/O port, port 2 also functions as an address bus (A0 to A7) or (A16 to A23). To use port 2 as an address bus, write 1 to the output latches to turn off the programmable pull-down resistors.

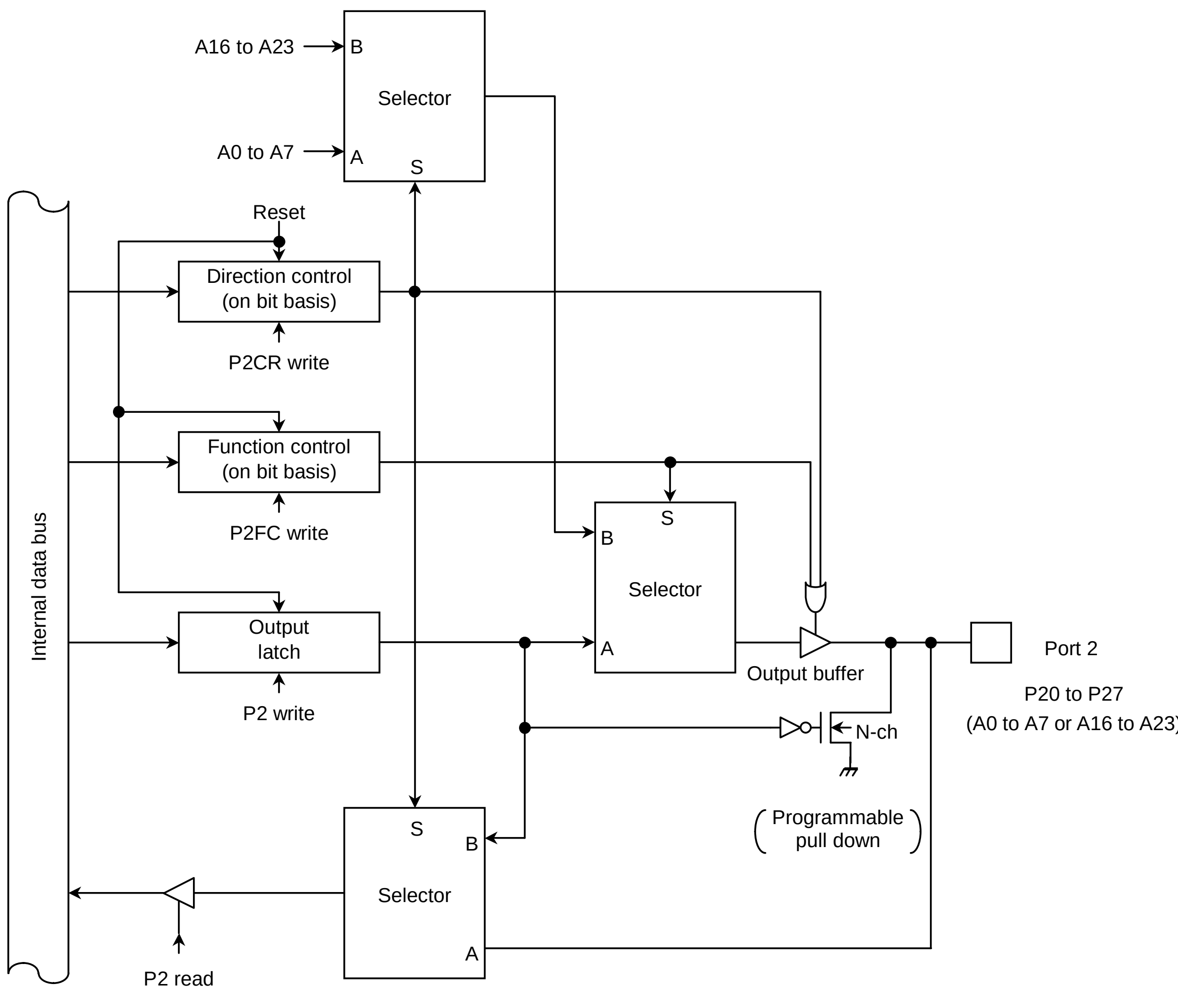



Figure 3.5.5 Port 2

Port 2 Register

P2 (0006H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| Read/Write | R/W | | | | | | | |
| After reset | Input mode (Output latch register is cleared to "0".) | | | | | | | |

Port 2 Control Register

P2CR (0008H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P27C | P26C | P25C | P24C | P23C | P22C | P21C | P20C |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | <<See table below.>> | | | | | | | |

Port 2 Function Register

P2FC (0009H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P27F | P26F | P25F | P24F | P23F | P22F | P21F | P20F |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | <<See table below.>> | | | | | | | |

Port 2 function setting

| P2CR <P2XC> \ P2FC<P2XF> | 0 | 1 |
|---|---|---|
| 0 | Input port | Address data bus (A7 to A0) |
| 1 | Output port | Address bus (A23 to A16) |

Note 1: Read-modify-write is prohibited for registers P2CR and P2FC.

Note 2: When port P2 is used in the input mode, P2 register controls the built-in pull-down resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: <P2XF> is bit X in register P2FC; <P2XC> is bit X in register P2CR.
To set as an address bus A23 to A16, set P2FC after setting P2CR.

Figure 3.5.6 Registers for Port 2

### 3.5.4 Port 3 (P30 to P37)

Port 3 is an 8-bit general-purpose I/O port.

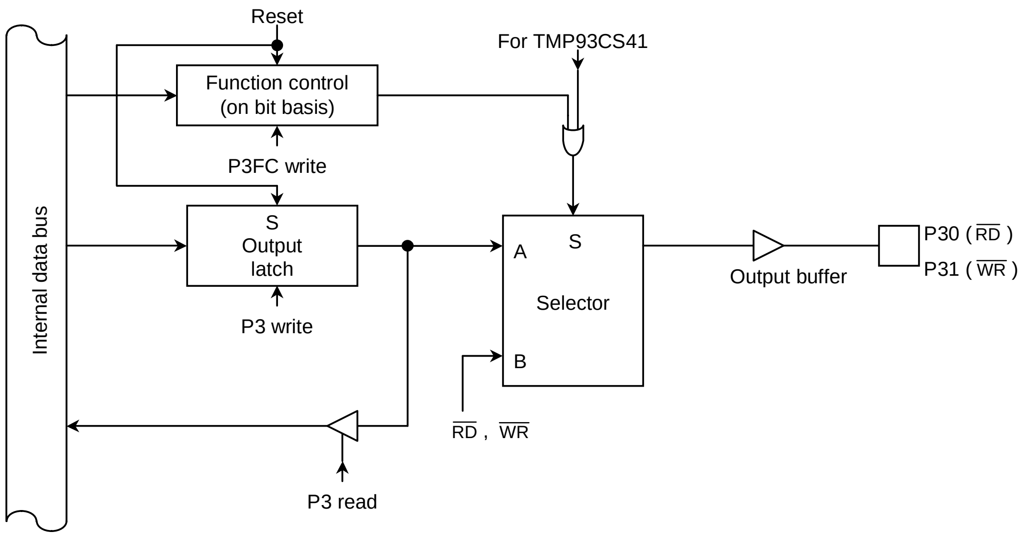
I/O can be set on a bit basis, but note that P30 and P31 are used for output only. I/O is set using control register P3CR and function register P3FC. Resetting sets all bits of output latch P3 to P1, and control register P3CR (bits 0 and 1 are unused) and function register P3FC to 0. Resetting also outputs 1 from P30 and P31, sets P32 to P37 to input mode, and connects a pull-up resistor.

In addition to functioning as a general-purpose I/O port, port 3 also functions as an I/O for the CPU’s control/status signal.

When the P30 pin is defined as $\overline{\text{RD}}$ signal output mode (<P30F> = 1), in the TMP93CS40 or the TMP93CS41 is used, clearing the output latch register <P30> to 0 outputs the $\overline{\text{RD}}$ strobe (used for the pseudo-static RAM) from the P30 pin even when the internal address area is accessed.

If the output latch register <P30> remains 1, the $\overline{\text{RD}}$ strobe signal is output only when the external address area is accessed.

In the TMP93CS41, which needs external ROMs, P30 outputs the $\overline{\text{RD}}$ signal and P31 outputs the $\overline{\text{WR}}$ signal, regardless of the values set in function registers P30F and P31F.



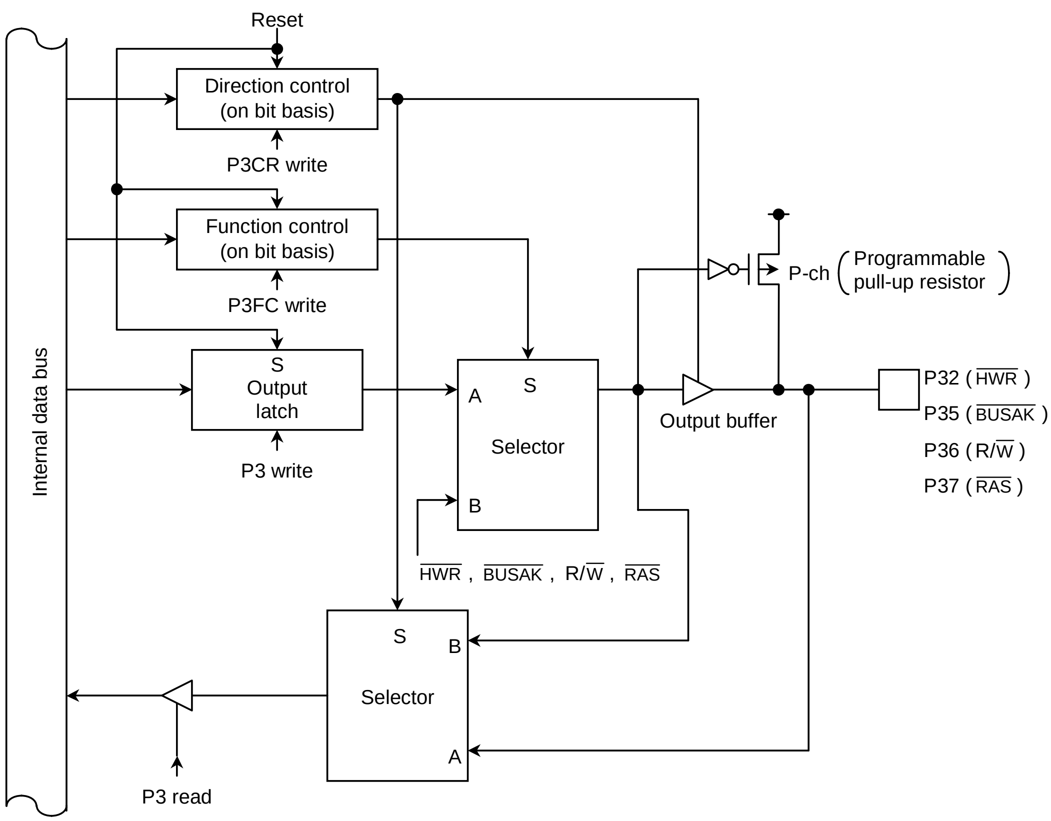
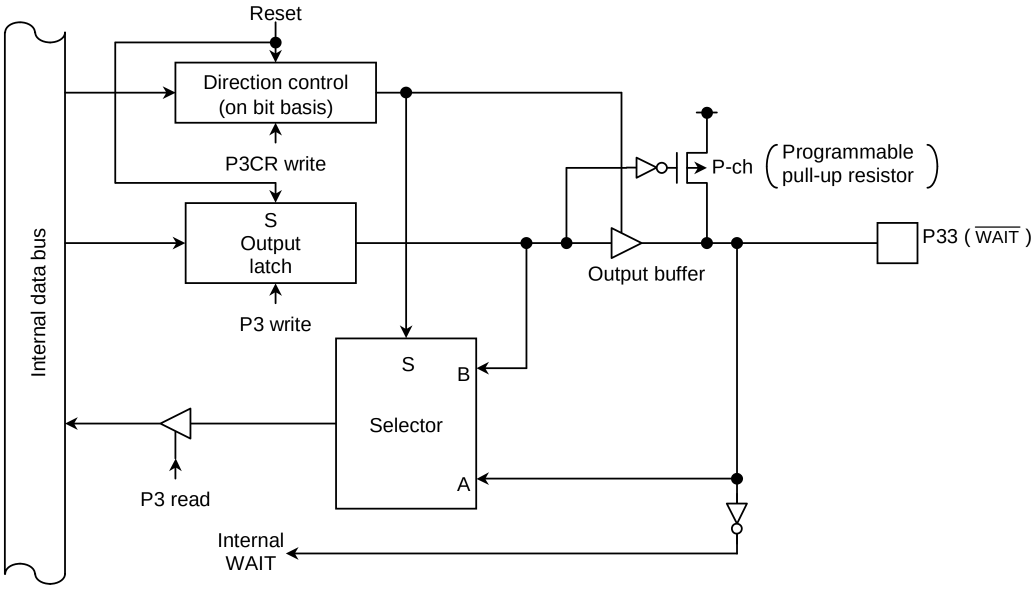




Figure 3.5.7 Port 3 (P30, P31, P32, P35, P36, P37)



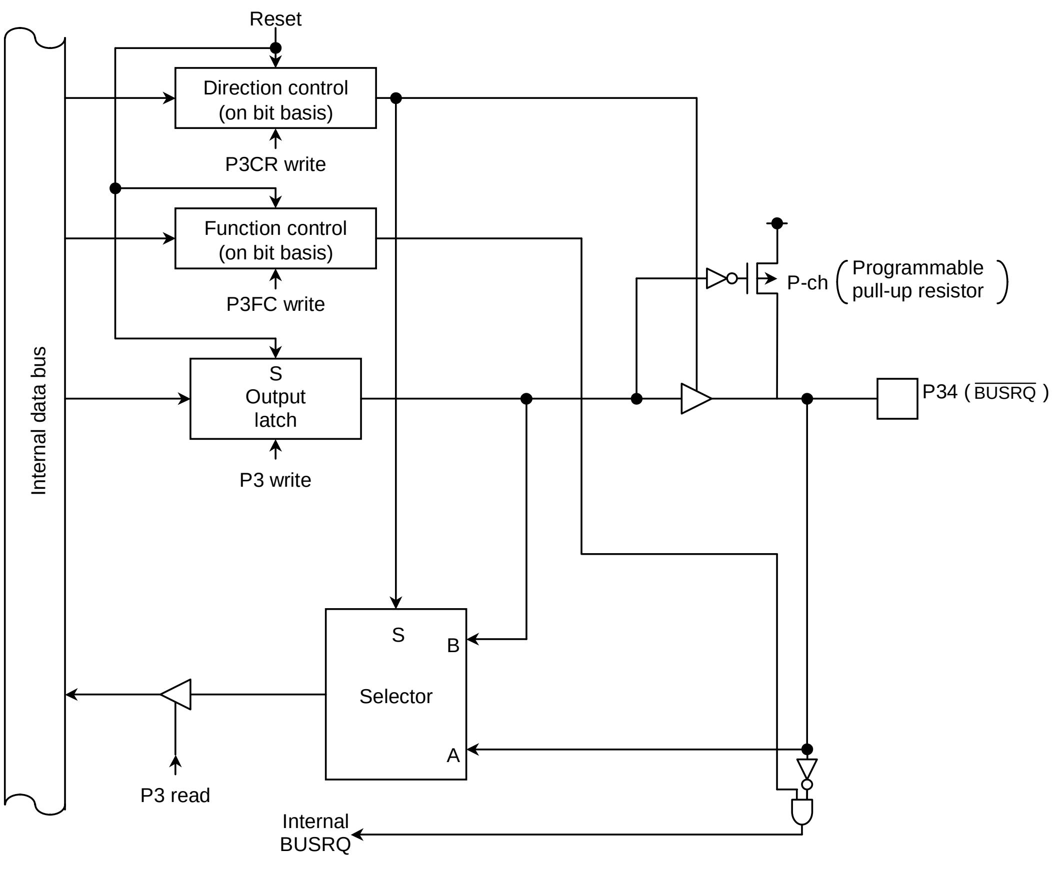




Figure 3.5.8 Port 3 (P33, P34)

Port 3 Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P3 (0007H) | Bit symbol | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 |
| | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Function | Input mode (Pull up) | | | | | | Output mode | |

Port 3 Control Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P3CR (000AH) | Bit symbol | P37C | P36C | P35C | P34C | P33C | P32C | | |
| | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | | |
| | Function | 0: Input 1: Output | | | | | | | |

I/O setting

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

Port 3 Function Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P3FC (000BH) | Bit symbol | P37F | P36F | P35F | P34F | | P32F | P31F | P30F |
| | Read/Write | W | | | | | W | | |
| | After reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| | Function | 0: Port<br>1: $\overline{\text{RAS}}$ | 0: Port<br>1: R/$\overline{\text{W}}$ | 0: Port<br>1: $\overline{\text{BUSAK}}$ | 0: Port<br>1: $\overline{\text{BUSRQ}}$ | | 0: Port<br>1: $\overline{\text{HWR}}$ | 0: Port<br>1: $\overline{\text{WR}}$ | 0: Port<br>1: $\overline{\text{RD}}$ |

P30 ($\overline{\text{RD}}$) function setting

| <P30F> \ <P30> | 0 | 1 |
|---|---|---|
| 0 | "0" output | "1" output |
| 1 | Always $\overline{\text{RD}}$ output (for pseudo SRAM) | $\overline{\text{RD}}$ output only for external access |

$\overline{\text{BUSRQ}}$ setting

| | |
|---|---|
| P3FC<P34F> | 1 |
| P3CR<P34C> | 0 |

$\overline{\text{BUSAK}}$ setting

| | |
|---|---|
| P3FC<P35F> | 1 |
| P3CR<P35C> | 1 |

P31 ($\overline{\text{WR}}$) function setting

| <P31F> \ <P31> | 0 | 1 |
|---|---|---|
| 0 | "0" output | "1" output |
| 1 | $\overline{\text{WR}}$ output only for external access | |

R/$\overline{\text{W}}$ setting

| | |
|---|---|
| P3FC<P36F> | 1 |
| P3CR<P36C> | 1 |

$\overline{\text{RAS}}$ setting

| | |
|---|---|
| P3FC<P37F> | 1 |
| P3CR<P37C> | 1 |

$\overline{\text{HWR}}$ setting

| | |
|---|---|
| P3FC<P32F> | 1 |
| P3CR<P32C> | 1 |

Note 1: Read-modify-write is prohibited for registers P3CR and P3FC.

Note 2: When port P3 is used in the input mode, the P3 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: When the P33/WAIT pin is used as a WAIT pin, set P3CR<P33C> to "0" and set the chip select/WAIT control register <BnW1:0> to "10".

Figure 3.5.9 Registers for Port 3

### 3.5.5 Port 4 (P40 to P42)

Port 4 is a 3-bit general-purpose I/O port. I/O can be set on a bit basis using control register P4CR and function register P4FC. Resetting does the following:

- Sets the P40 and P41 output latch registers to 1.
- Resets all bits of the P42 output latch register, the control register P4CR, and the function register P4FC to 0.
- Sets P40 and P41 to input mode and connects a pull-up resistor.
- Sets P42 to input mode and connects a pull-down resistor.

In addition to functioning as a general-purpose I/O port, port 4 also functions as a chip select output signal ( $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ or $\overline{\text{CAS0}}$ to $\overline{\text{CAS2}}$ ).

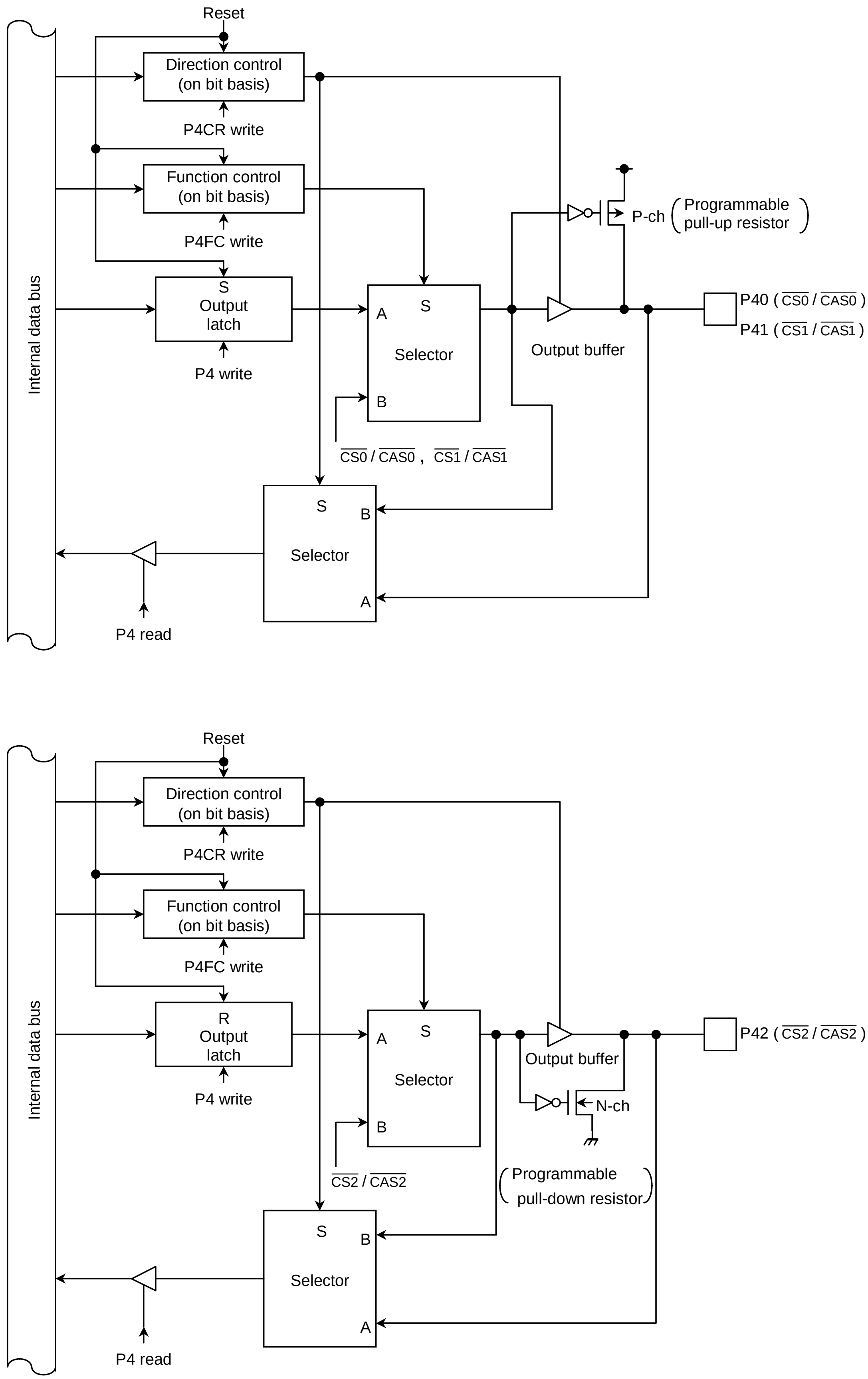



Figure 3.5.10 Port 4

Port 4 Register

| P4 (000CH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | | P42 | P41 | P40 |
| | Read/Write | | | | | | R/W | | |
| | After reset | | | | | | Input mode | | |
| | Function | | | | | | 0 (Pull down) | 1 (Pull up) | 1 (Pull up) |

Port 4 Control Register

| P4CR (000EH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | | P42C | P41C | P40C |
| | Read/Write | | | | | | W | | |
| | After reset | | | | | | 0 | 0 | 0 |
| | Function | | | | | | 0: Input 1: Output | | |

I/O setting

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

Port 4 Function Register

| P4FC (0010H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | | P42F | P41F | P40F |
| | Read/Write | | | | | | W | | |
| | After reset | | | | | | 0 | 0 | 0 |
| | Function | | | | | | 0: Port 1: $\overline{CS}$ / $\overline{CAS}$ | | |

| | |
|---|---|
| 0 | Port (P40) |
| 1 | $\overline{CS0}$ / $\overline{CAS0}$ |

| | |
|---|---|
| 0 | Port (P41) |
| 1 | $\overline{CS1}$ / $\overline{CAS1}$ |

| | |
|---|---|
| 0 | Port (P42) |
| 1 | $\overline{CS2}$ / $\overline{CAS2}$ |

Note 1: Read-modify-write is prohibited for registers P4CR and P4FC.

Note 2: When port P4 is used in the input mode, the P4 register controls the built-in pull-up/pull-down resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: To output chip select signal ( $\overline{CS0}$ / $\overline{CAS0}$ to $\overline{CS2}$ / $\overline{CAS2}$ ), set the corresponding bits of the control register P4CR and the function register P4FC to "1".
Chip select/wait controller (B0CS, B1CS, B2CS) registers select the function of $\overline{CS}$ / $\overline{CAS}$ .

Note 4: P4<bit7:3> are always read as "1".

Figure 3.5.11 Registers for Port 4

### 3.5.6 Port 5 (P50 to P57)

Port 5 is an 8-bit input port, also used as an analog input pin for the internal AD converter.

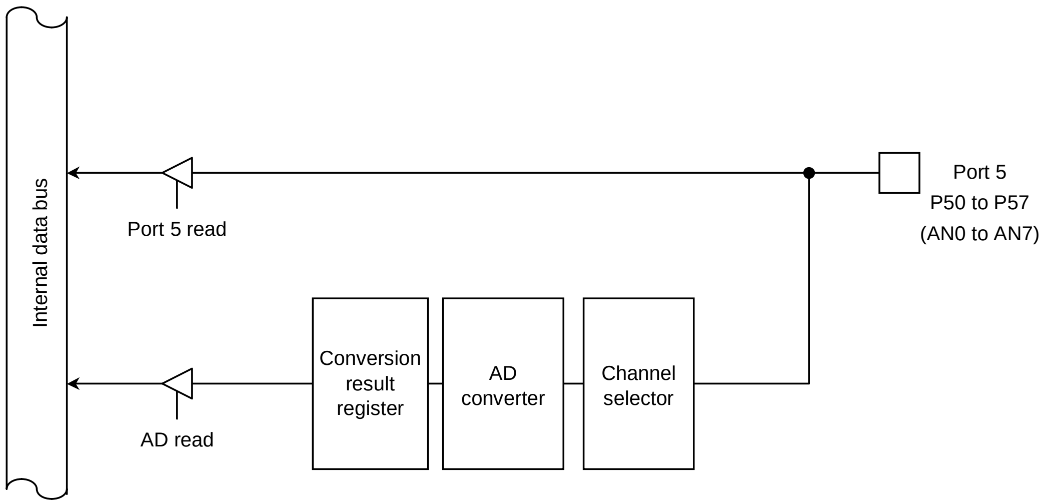



Figure 3.5.12 Port 5

Port 5 Register

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| P5 | Bit symbol | P57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 |
| (000DH) | Read/Write | R | | | | | | | |
| | After reset | Input mode | | | | | | | |

Note: The input channel selection of the AD converter is set by AD converter mode register ADMOD2.

Figure 3.5.13 Registers for Port 5

### 3.5.7 Port 6 (P60 to P67)

Port 6 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis. Resetting sets port 6 as an input port and connects a pull-up resistor. It also sets all bits of the output latch to 1. In addition to functioning as a general-purpose I/O port, port 6 also functions as a pattern generator of PG0 or PG1 output. PG0 is assigned to P60 to P63; PG1, to P64 to P67. Writing 1 in the appropriate bit of the port 6 function register (P6FC) enables PG output. Resetting resets the function register P6CR, P6FC value to "0", and sets all bits to input ports.

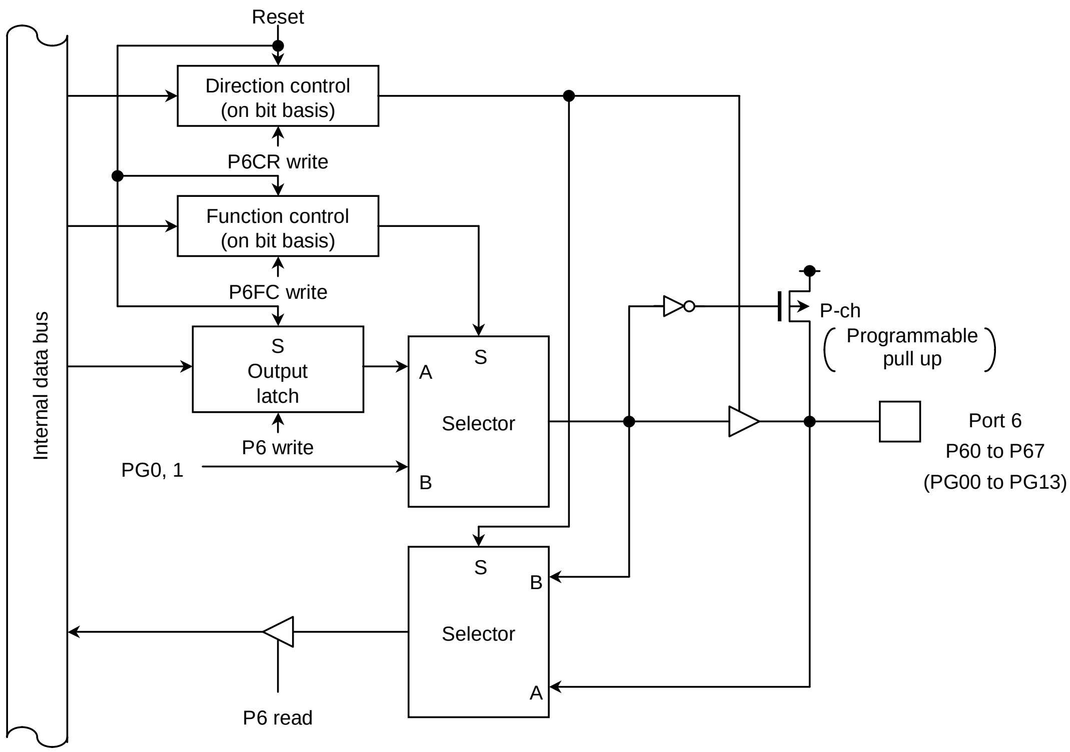



Figure 3.5.14 Port 6

Port 6 Register

P6 (0012H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Function | Input mode (Pull up) | | | | | | | |

Port 6 Control Register

P6CR (0014H)

| | 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 | | | | | | | |

Port 6 I/O setting

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

Port 6 Function Register

P6FC (0016H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | P67F | P66F | P65F | P64F | P63F | P62F | P61F | P60F |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | 0: Port 1: PG1-OUT | | | | 0: Port 1: PG0-OUT | | | |

Port 6 function setting

| | |
|---|---|
| 0 | General-purpose port |
| 1 | Stepping motor control/pattern generation port |

Note 1: Read-modify-write is prohibited for registers P6CR and P6FC.

Note 2: When port P6 is used in the input mode, the P6 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Figure 3.5.15 Registers for Port 6

### 3.5.8 Port 7 (P70 to P73)

Port 7 is a 4-bit general-purpose I/O port. I/O can be set on a bit basis. Resetting sets port 7 as an input port and connects a pull-up resistor. In addition to functioning as a general-purpose I/O port, port 70 also functions as an input clock pin TI0; port 71 as an 8-bit timer output (TO1), port 72 as a PWM0 output (TO2), and port 73 as a PWM1 output (TO3) pin. Writing 1 in the corresponding bit of the port 7 function register (P7FC) enables output of the timer. Resetting resets the function register P7CR, P7FC value to 0, and sets all bits to input ports.

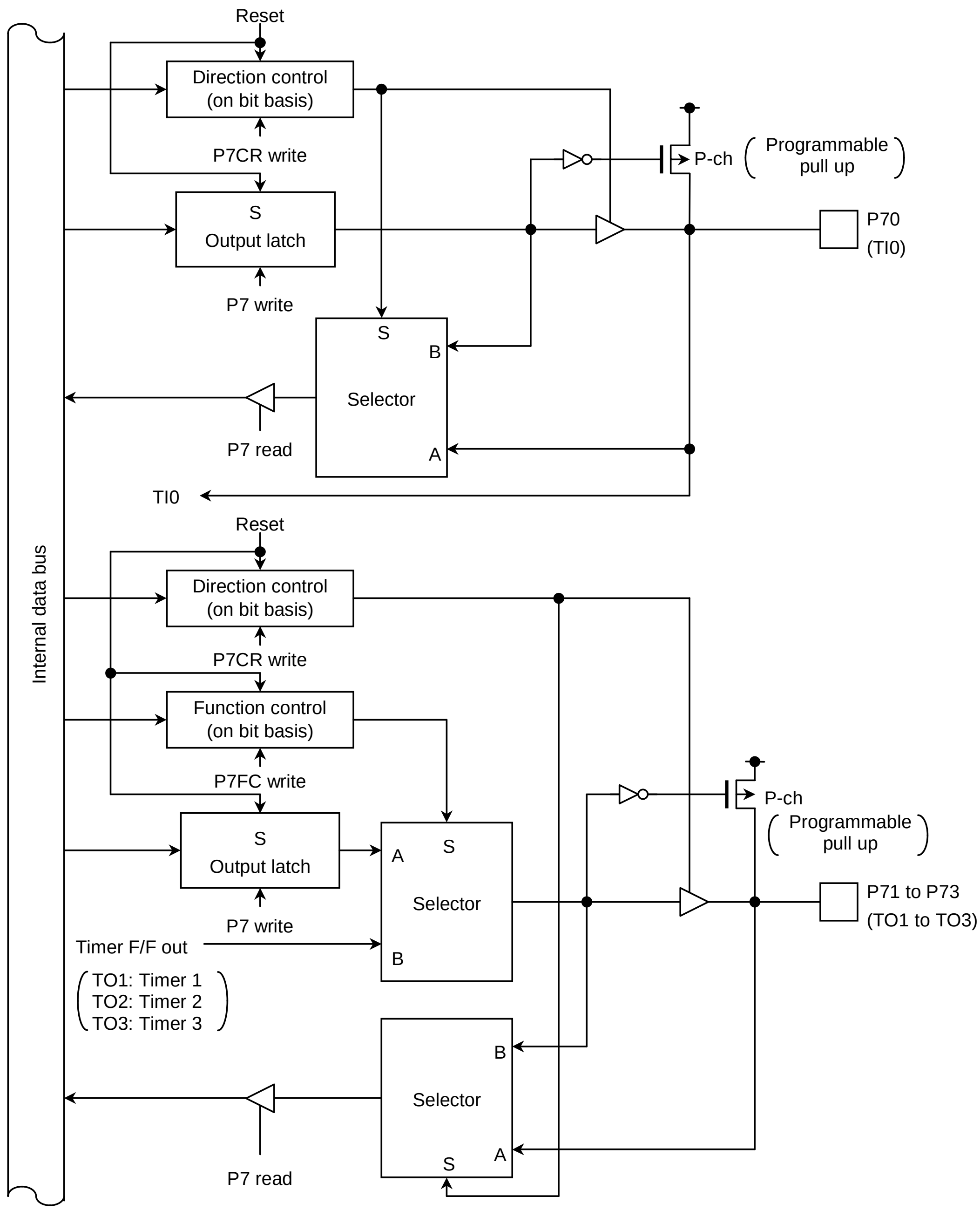



Figure 3.5.16 Port 7

Port 7 Register

| P7 (0013H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | P73 | P72 | P71 | P70 |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 1 | 1 | 1 | 1 |
| | Function | | | | | Input mode (Pull up) | | | |

Port 7 Control Register

| P7CR (0015H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | P73C | P72C | P71C | P70C |
| | Read/Write | | | | | W | | | |
| | After reset | | | | | 0 | 0 | 0 | 0 |
| | Function | | | | | 0: Input 1: Output | | | |

Port 7 I/O setting

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

Port 7 Function Register

| P7FC (0017H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | P73F | P72F | P71F | |
| | Read/Write | | | | | W | | | |
| | After reset | | | | | 0 | 0 | 0 | |
| | Function | | | | | 0: Port<br>1: TO3 | 0: Port<br>1: TO2 | 0: Port<br>1: TO1 | |

Setting P71 as TO1

| | |
|---|---|
| P7FC<P71F> | 1 |
| P7CR<P71C> | 1 |

Setting P72 as TO2

| | |
|---|---|
| P7FC<P72F> | 1 |
| P7CR<P72C> | 1 |

Setting P73 as TO3

| | |
|---|---|
| P7FC<P73F> | 1 |
| P7CR<P73C> | 1 |

Note 1: Read-modify-write is prohibited for registers P7CR and P7FC.

Note 2: When port P7 is used in the input mode, the P7 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: The P70/T10 pin does not have a register changing it from port to function. For example, when it is used as an input port, the incoming signal is input to 8-bit timer 0 as a timer input 0 (T10).

Note 4: P7<bit7:4> are always read as "1".

Figure 3.5.17 Registers for Port 7

### 3.5.9 Port 8 (P80 to P87)

Port 8 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis. Resetting sets port 8 as an input port and connects a pull-up resistor. It also sets all bits of the output latch register P8 to 1. In addition to functioning as a general-purpose I/O port, port 8 also functions as an input for 16-bit timer 4 and 5 clocks, an output for 16-bit timer F/F 4, 5, and 6 output, and an input for INT0. Writing "1" in the corresponding bit of the port 8 function register (P8FC) enables those functions. Resetting resets the function register P8CR, P8FC value to "0" and sets all bits to input ports.

(1) P80 to P86

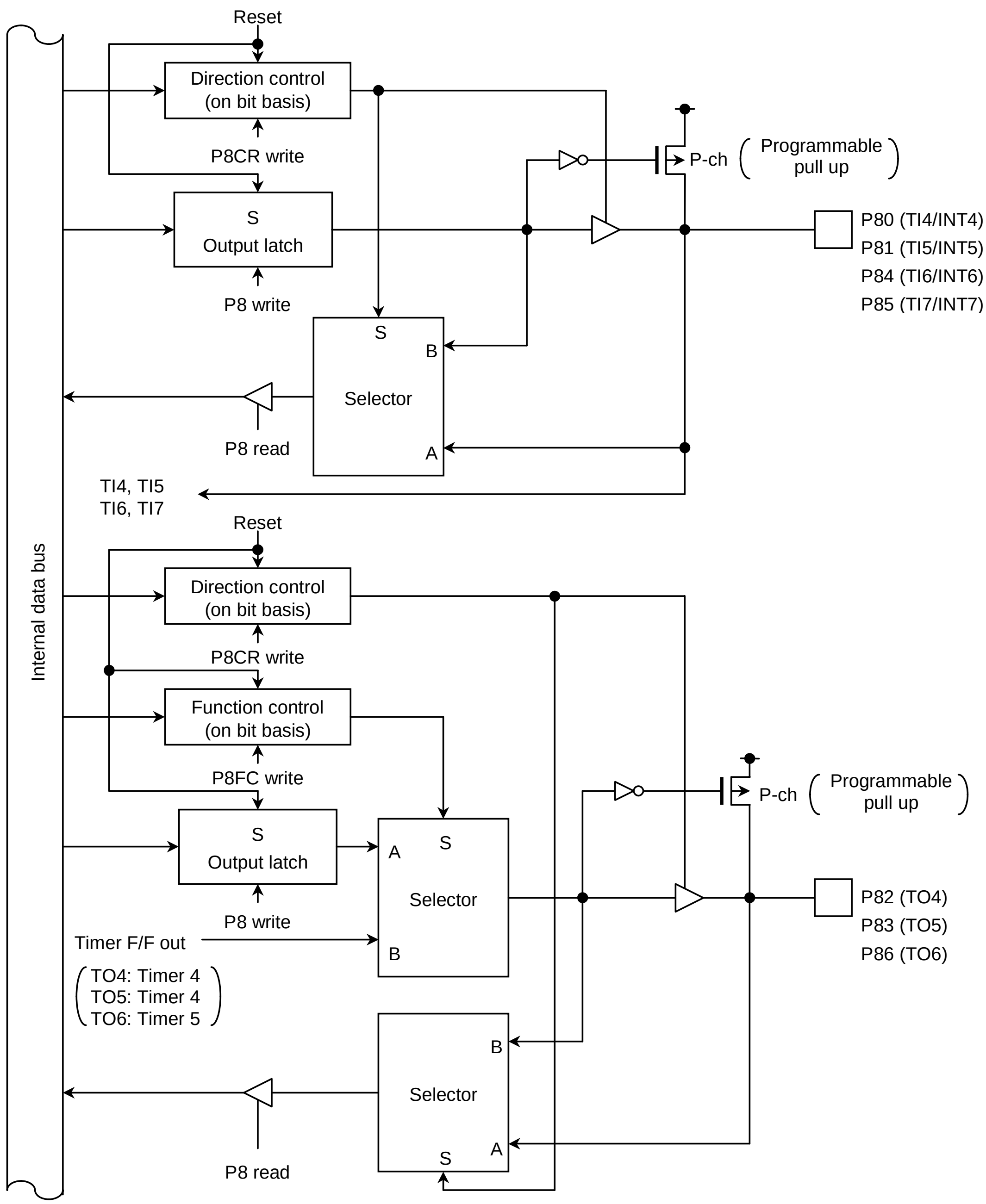



Figure 3.5.18 Port 8 (P80 to P86)

(2) P87 (INT0)

Port 87 is a general-purpose I/O port, and is also used as an INT0 pin for external interrupt request input.

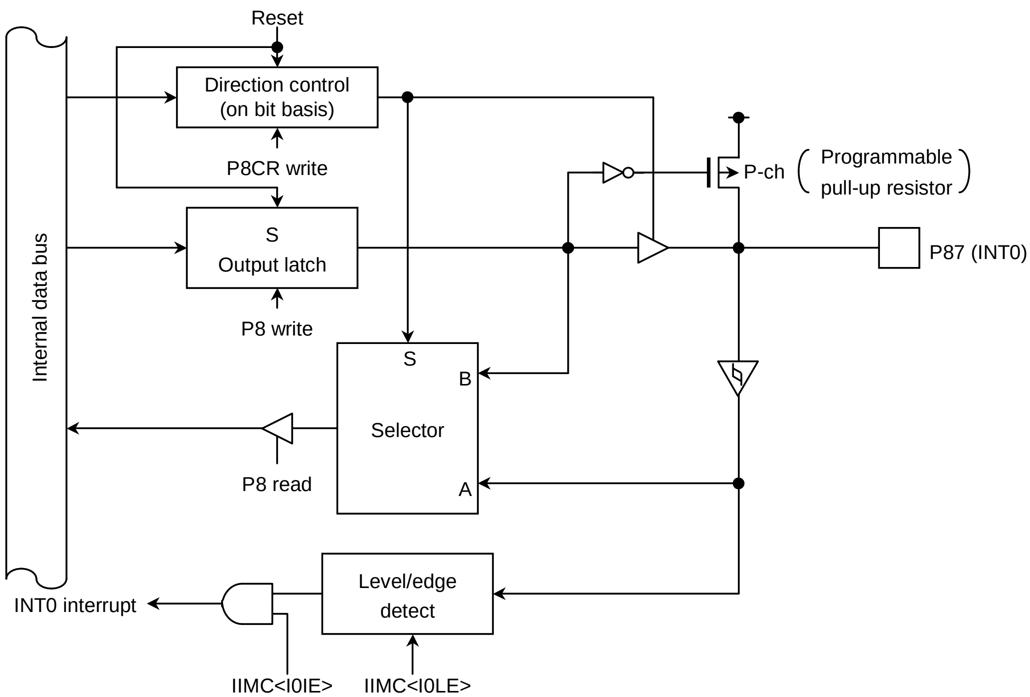



Figure 3.5.19 Port 87

Port 8 Register

| P8 (0018H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | P87 | P86 | P85 | P84 | P83 | P82 | P81 | P80 |
| | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Function | Input mode | | | | | | | |

Port 8 Control Register

| P8CR (001AH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | P87C | P86C | P85C | P84C | P83C | P82C | P81C | P80C |
| | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | 0: Input 1: Output | | | | | | | |

Port 8 I/O setting

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

Port 8 Function Register

| P8FC (001CH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | P86F | | | P83F | P82F | | |
| | Read/Write | | W | | | W | W | | |
| | After reset | | 0 | | | 0 | 0 | | |
| | Function | | 0: Port<br>1: TO6 | | | 0: Port<br>1: TO5 | 0: Port<br>1: TO4 | | |

Setting P82 as TO4

| | |
|---|---|
| P8FC<P82F> | 1 |
| P8CR<P82C> | 1 |

Setting P83 as TO5

| | |
|---|---|
| P8FC<P83F> | 1 |
| P8CR<P83C> | 1 |

Setting P86 as TO6

| | |
|---|---|
| P8FC<P86F> | 1 |
| P8CR<P86C> | 1 |

Note 1: Read-modify-write is prohibited for registers P8CR and P8FC.

Note 2: When port P8 is used in the input mode, the P8 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: The P80/TI4, P81/TI5, P84/TI6, and P85/TI7 pins do not have a register changing them from port to function. For example, when they are used as an input port, the incoming signal is input to the 16-bit timer as timer input.
When P87/INT0 pin is used as an INT0 pin, set P8CR<P87C> to “0” and IIMC<I0IE> to “1”.

Figure 3.5.20 Registers for Port 8

### 3.5.10 Port 9 (P90 to P97)

- Ports 90 to 95

  Ports 90 to 95 constitute a 6-bit general-purpose I/O ports. I/Os can be set on a bit basis.

  Resetting sets P90 to P95 to an input port and connects a pull-up resistor.

  It also sets all bits of the output latch register to 1.

  In addition to functioning as a general-purpose I/O port, P90 to P95 can also function as an I/O for serial channels 0 and 1. Writing "1" in the corresponding bit of the port 9 function register (P9FC) enables those functions.

  Resetting resets the function register P9CR, P9FC value to "0" and sets all bits to input ports.

- Ports 96 to 97

  Ports 96 to 97 form a 2-bit general-purpose I/O port. I/Os can be set on a bit basis.

  The output buffer for P96 to P97 is an open-drain type buffer.

  Resetting sets the output latch and control registers to "1" and outputs high-impedance (High-Z).

  In addition to functioning as a general-purpose I/O port, P96 to P97 can also function as a low-frequency oscillator connecting pin for dual clock mode. The dual clock function can be set by programming system clock control register SYSCR0 and 1.

(1) Ports 90, 93 (TXD0/TXD1)

Ports 90 and 93 also function as serial channel TXD output pins in addition to I/O ports.

They have a programmable open-drain function.

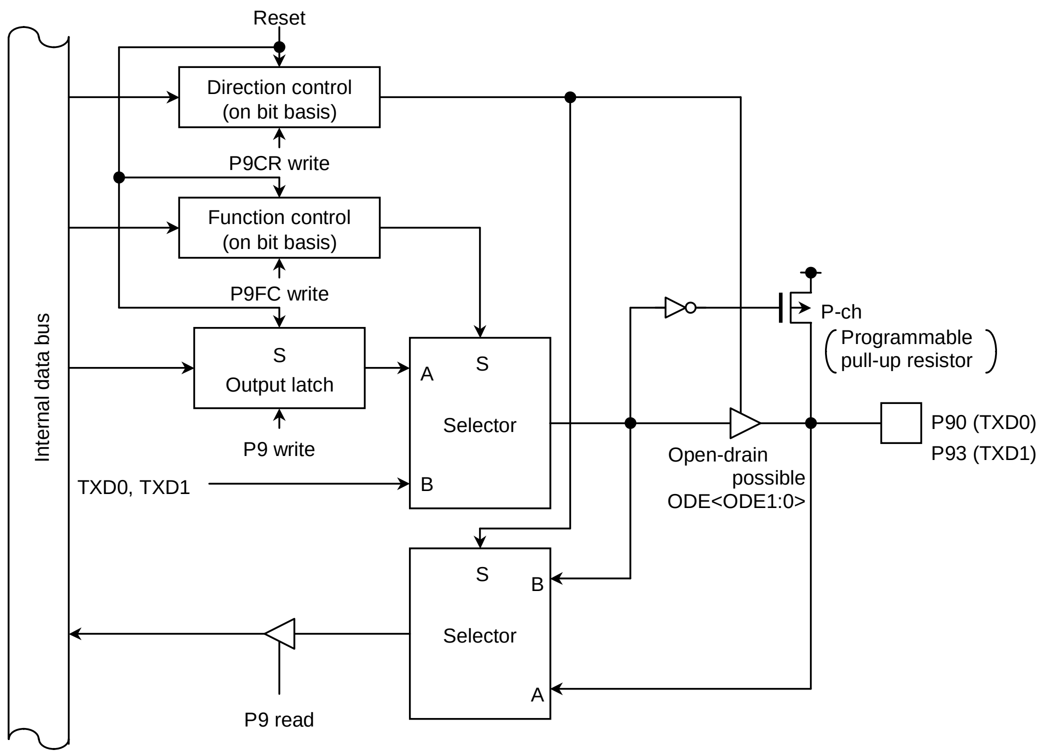



Figure 3.5.21 Ports 90 and 93

(2) Port 91, 94 (RXD0, RXD1)

Ports 91 and 94 are I/O ports, and are also used as RXD input pins for serial channels.

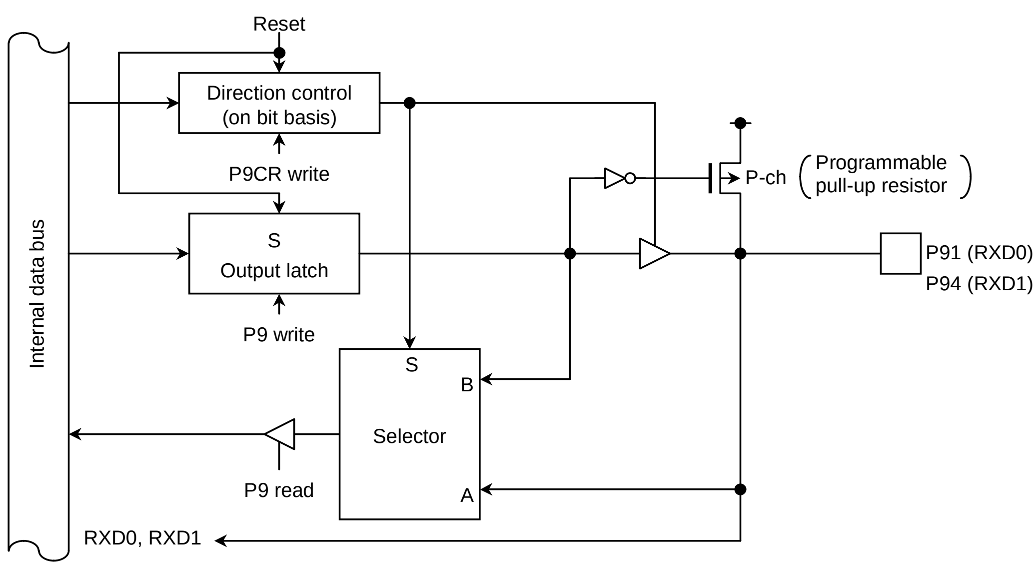



Figure 3.5.22 Ports 91 and 94

(3) Port 92 ($\overline{\text{CTS0}}$ /SCLK0)

Port 92 is an I/O port, and is also used as a $\overline{\text{CTS0}}$ input pin and as a SCLK0 I/O pin for serial channels.

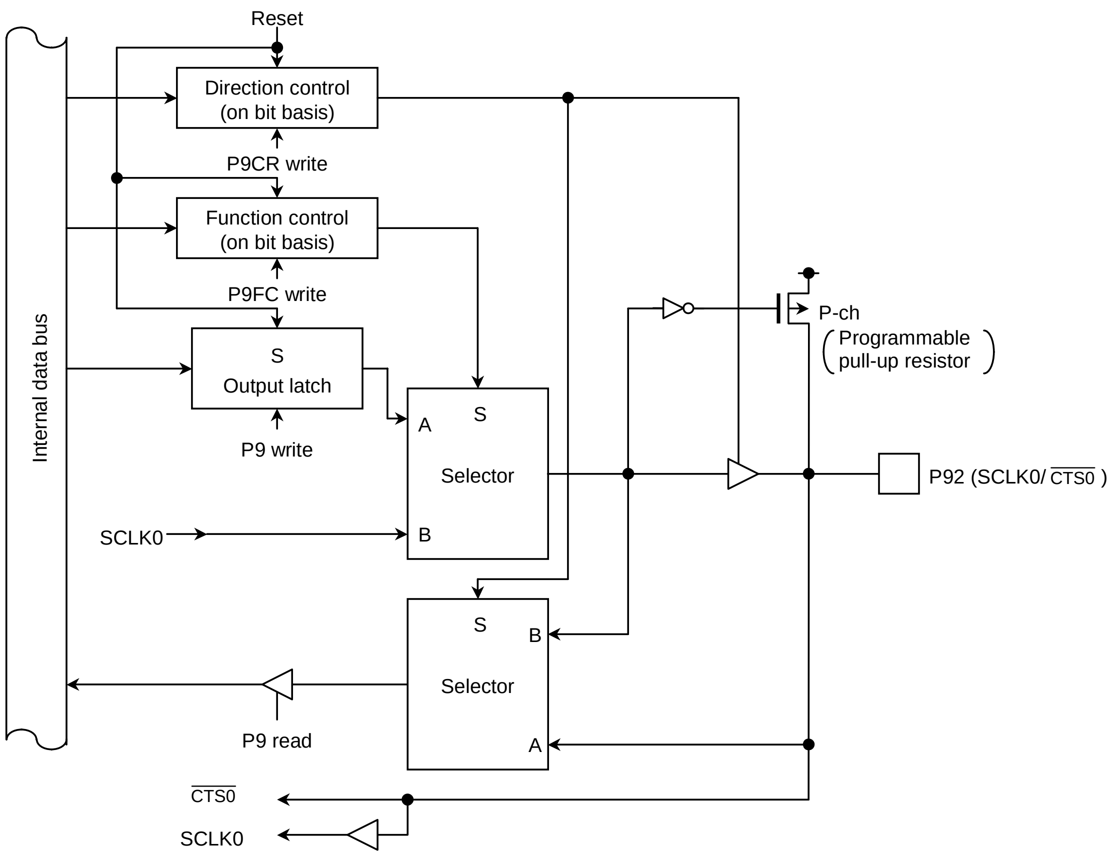



Figure 3.5.23 Port 92

(4) Port 95 (SCLK1)

Port 95 is a general-purpose I/O port. It is also used as a SCLK1 I/O pin for serial channel 1.

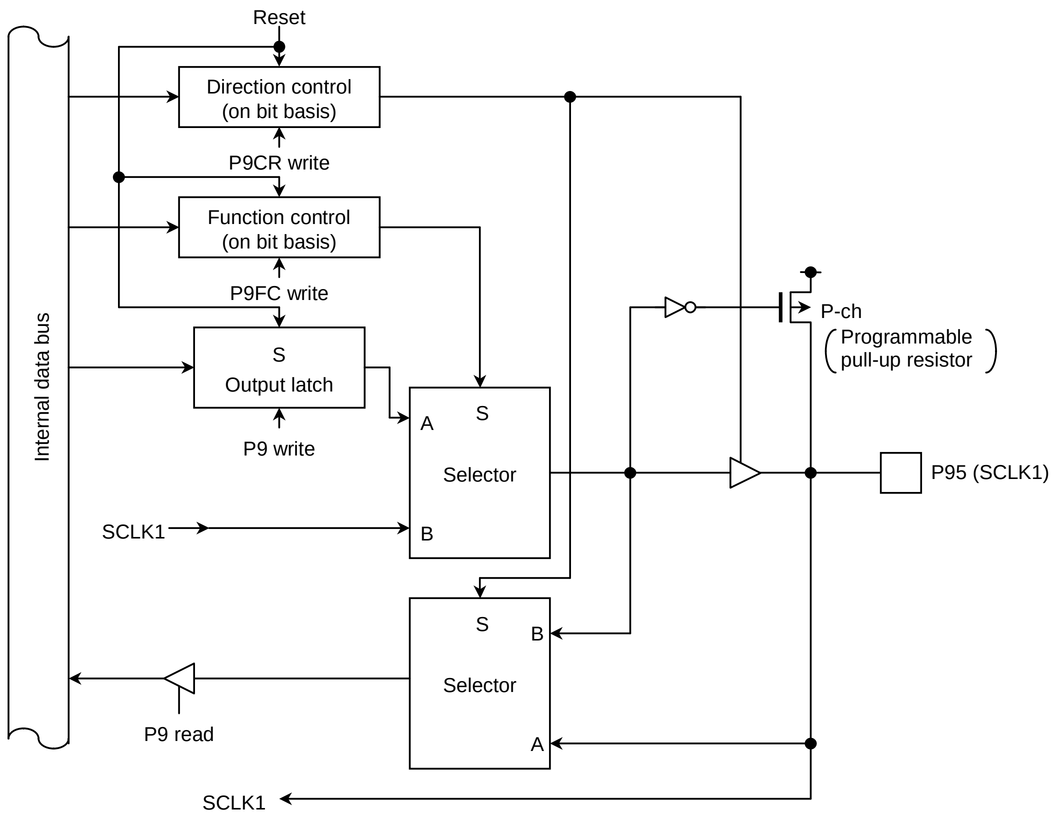



Figure 3.5.24 Port 95

(5) Port 96 (XT1), 97 (XT2)

Ports 96 and 97 are general purpose I/O ports. They are also used as low-frequency oscillator connecting pins.

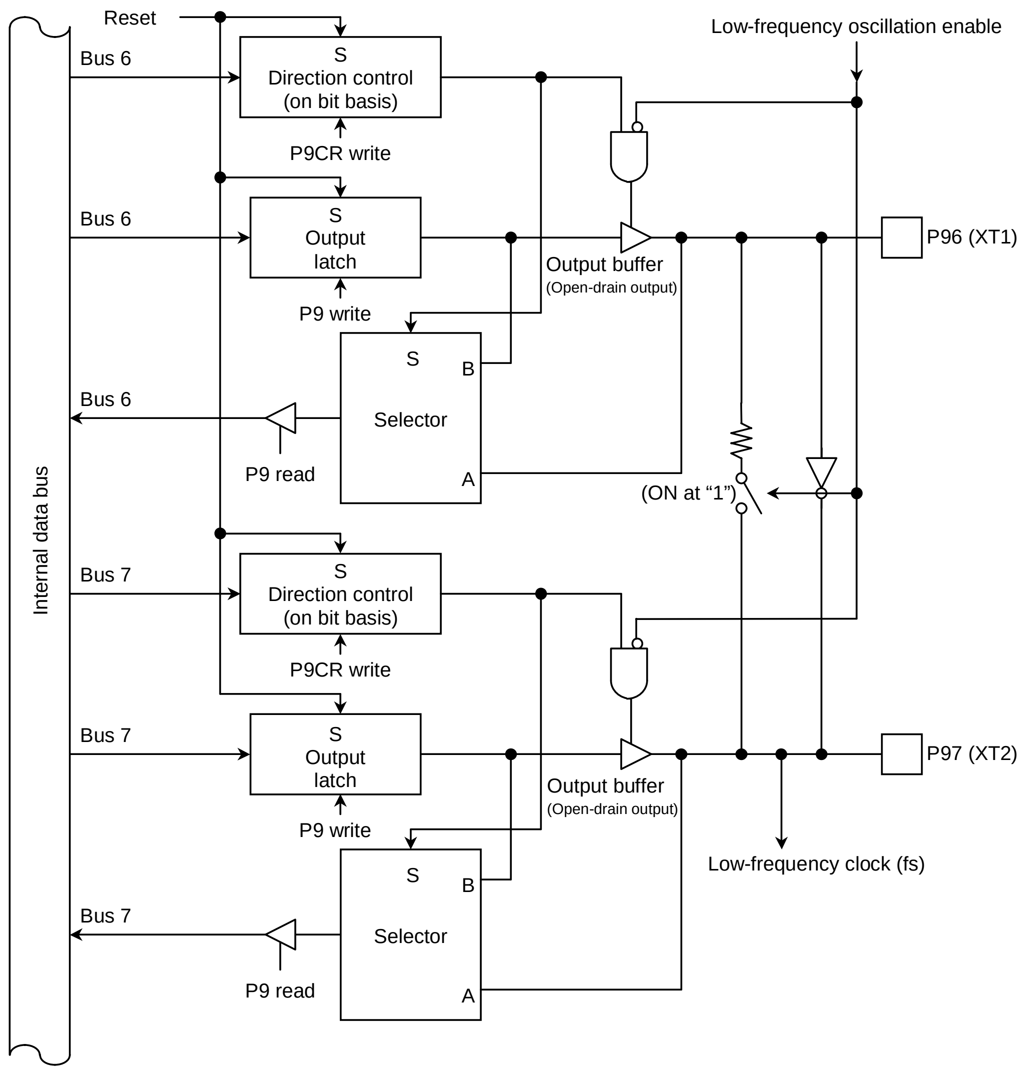



Figure 3.5.25 Ports 96 and 97

Port 9 Register

| P9 (0019H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | P97 | P96 | P95 | P94 | P93 | P92 | P91 | P90 |
| | Read/Write | R/W | | | | | | | |
| | After reset | Output mode | | Input mode | | | | | |
| | Function | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Port 9 Control Register

| P9CR (001BH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | P97C | P96C | P95C | P94C | P93C | P92C | P91C | P90C |
| | Read/Write | W | | | | | | | |
| | After reset | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | 0: Input 1: Output | | | | | | | |

Port 9 I/O setting

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

Note: Port 96 and 97's output buffer is an open-drain output type.

Port 9 Function Register

| P9FC (001DH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | P95F | | P93F | P92F | | P90F |
| | Read/Write | | | W | | W | W | | W |
| | After reset | | | 0 | | 0 | 0 | | 0 |
| | Function | | | 0: Port<br>1: SCLK1 | | 0: Port<br>1: TXD1 | 0: Port<br>1: SCLK0 | | 0: Port<br>1: TXD0 |

P90 TXD0 output setting (Note 3)

| | |
|---|---|
| P9FC<P90F> | 1 |
| P9CR<P90C> | 1 |

P92 SCLK0 output setting

| | |
|---|---|
| P9FC<P92F> | 1 |
| P9CR<P92C> | 1 |

P93 TXD1 output setting (Note 3)

| | |
|---|---|
| P9FC<P93F> | 1 |
| P9CR<P93C> | 1 |

P95 SCLK1 output setting

| | |
|---|---|
| P9FC<P95F> | 1 |
| P9CR<P95C> | 1 |

Note 1: Read-modify-write is prohibited for registers P9CR and P9FC.

Note 2: When port P9 is used in the input mode, the P9 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up/pull-down resistors.

Note 3: When the TXD pin is set to be an open-drain output type, set 1 to bit0 (for TXD0 pin) or bit1 (for TXD1 pin) of the ODE register.
The P91/RXD0 and P94/RXD1 pins do not have a register changing them from port to function. For example, when they are used as an input port, the incoming signal is input to the SIO as serial receiving data.

Note 4: Notes on using low-frequency oscillation circuit.
To connect a low-frequency resonator to ports 96 and 97,
it is necessary to set the following procedures to reduce the consumption of power.
(Connecting to a resonator)
Set P9CR<P96C, P97C> = "11", P9<P96:97> = "00"
(Connecting to an oscillator)
Set P9CR<P96C, P97C> = "11", P9<P96:97> = "10"

Note5: When ports 96 and 97 is used in the output mode, input gate in operation.
Set output to "L" or attach pull-up on pin to reduce the consumption of power, before the HALT instruction is executed.

Figure 3.5.26 Registers for Port 9

### 3.5.11 Port A (PA0 to PA7)

Port A is an 8-bit general-purpose I/O port. I/Os can be set on a bit basis by control register PACR.

Resetting sets port A to an input port by resetting PACR.

It also sets all bits of the output latch register to "1".

In addition to functioning as a general-purpose I/O port (Only PA7), PA7 can also function as an internal clock output pin.

The output clock is $f_{FPH}$ or $f_{SYS}$ that is selected as oscillator output clock. It is selected by CKOCR<SCOSEL>.

The SCOUT function is enabled by setting PACR<PA7C> and CKOCR<SCOEN>.

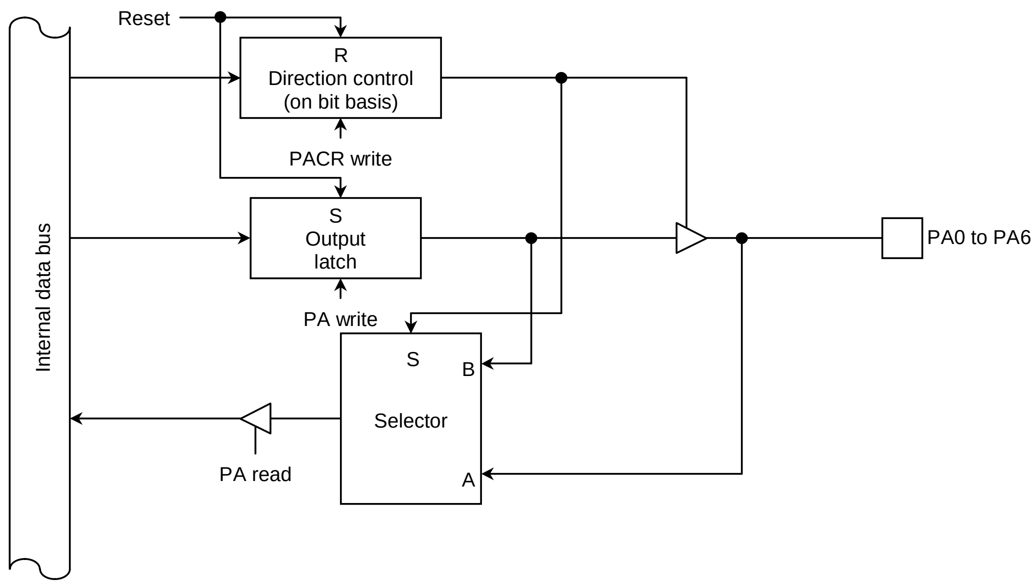



Figure 3.5.27 Port A0 to A6

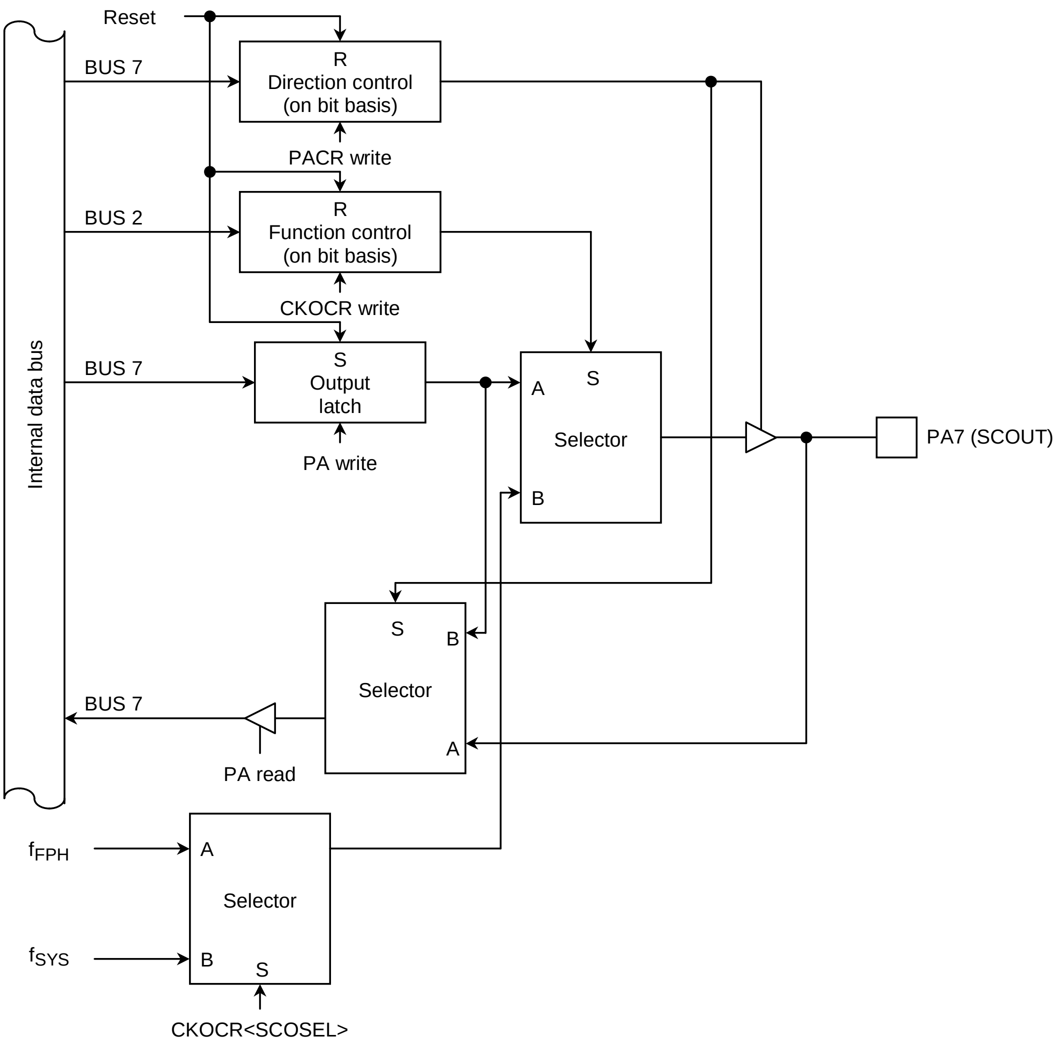



Figure 3.5.28 Port A7

Port A Register

| PA (001EH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| | Read/Write | R/W | | | | | | | |
| | After reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Function | Input mode | | | | | | | |

Port A Control Register

| PACR (001FH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | PA7C | PA6C | PA5C | PA4C | PA3C | PA2C | PA1C | PA0C |
| | Read/Write | W | | | | | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | 0: Input 1: Output | | | | | | | |

Clock Output Control Register

| CKOCR (006DH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | | | SCOSEL | SCOEN | ALEEN | CLKEN |
| | Read/Write | | | | | R/W | | | |
| | After reset | | | | | 0 | 0 | 0/1 | 0/1 |
| | Function | | | | | Clock select | Clock enable | ALE enable | CLK enable |

CLK pin output control (Note 1)

| | |
|---|---|
| 0 | High impedance |
| 1 | CLK output |

ALE pin output control (Note 1)

| | |
|---|---|
| 0 | High impedance |
| 1 | ALE output |

Note 1: Read-modify-write is prohibited for registers PACR.
Note 2: The value after reset of <CLKEN>, <ALEEN> is following:
TMP93CS40: "0" (High-impedance output)
TMP93CS41: "1" (CLK or ALE output)
But during reset, the CLK pin is pulled up internally regardless of which of these two microcontrollers is in use.
Note 3: The output clock from the SCOUT pin is as $f_{FPH}$ or $f_{SYS}$ clock signal.
e.g.
The case of a 20 MHz oscillator connected to the X1 and X2 pins.
<SCOSEL> = "0" → 20 MHz clock
<SCOSEL> = "1" → 10 MHz clock

SCOUT/PA7 pin control

| <SCOEN> | <SCOSEL> / PACR<PA7C> | 0 | 1 |
|---|---|---|---|
| 0 | 0 | Input Port | Output mode |
| 0 | 1 | Input Port | Output mode |
| 1 | 0 | Input Port | $f_{FPH}$ clock output (Note 3) |
| 1 | 1 | Input Port | $f_{SYS}$ clock output (Note 3) |

Figure 3.5.29 Registers for Port A

## 3.6 Chip Select/Wait Controller, AM8/$\overline{\text{AM16}}$ pin

The TMP93CS40 and TMP93CS41 have a built-in chip select/wait controller used to control chip select ($\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ pins), wait ($\overline{\text{WAIT}}$ pin), and data bus size (8 or 16 bits) for any of the three block address areas.

In addition, the AM8/$\overline{\text{AM16}}$ pin selects external data bus width.

### 3.6.1 AM8/$\overline{\text{AM16}}$ pin

(1) Usage in the TMP93CS40

Set this pin to “1”. After resetting, the CPU accesses the internal ROM with 16-bit bus width. The bus width when the CPU accesses an external area is set by the chip select/wait control register (described in section 3.6.3), P1CR and P1FC. (If this pin is set to 1, that value will be ignored and the value set by register will be active.)

(2) Usage in the TMP93CS41

(2-1) When 16-bit bus width and 8-bit bus width are both used, or when 16-bit bus width only is used:

Set this pin to “0”. Then the AD8 to AD15 or A8 to A15 pins of port 1 are fixed to their A8 to A15 function compulsorily, and the values of P1CR and P1FC are ignored.

The bus width when the CPU accesses an external area is set by the chip select/wait control register described in section 3.6.2.

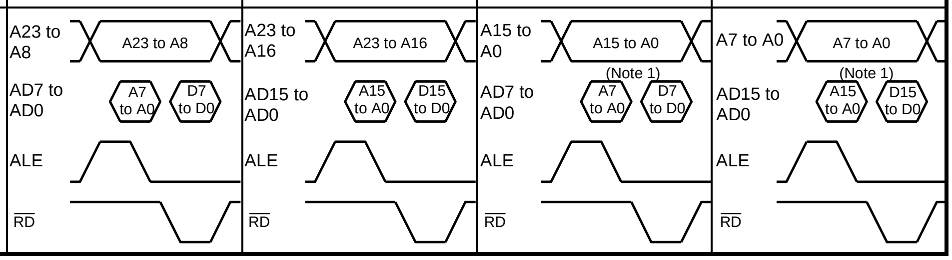
After a reset, 16-bit external program memory must be accessed before any other memory is accessed.

(2-2) When 8-bit bus width only is used:

Set this pin to “1”. Then the AD8 to AD15 or A8 to A15 pins of port 1 are fixed to their A8 to A15 function compulsorily, and the values of P1CR and P1FC are ignored. The values of bit4 in <B0BUS>, <B1BUS>, or <B2BUS>, described in section 3.6.3, are ignored and the bus width is fixed to 8 bits.

### 3.6.2 Address/Data Bus Pins

Port 0, port 1 and port 2 function as an address and data bus for connecting the microcontroller to the external memories and I/O peripherals.

| | | 1. | 2. | 3. | 4. |
|---|---|---|---|---|---|
| Products | | TMP93CS41F (Note 4) | | TMP93CS40F (Note 2), (Note 3) | |
| Number of Address Bus Pins | | max24 (to 16 Mbytes) | max24 (to 16 Mbytes) | max16 (to 64 Kbytes) | max8 (to 256 Kbytes) |
| Number of Data Bus Pins | | 8 | 16 | 8 | 16 |
| Number of Multiplexed Pins | | 8 | 16 | 0 | 0 |
| Mode Pins | $\overline{EA}$ | $V_{IL}$ | | $V_{IH}$ | |
| | AM8/$\overline{AM16}$ | $V_{IH}$ | $V_{IL}$ | $V_{IH}$ | |
| Port Function | Port 0 | AD0 to AD7 | AD0 to AD7 | AD0 to AD7 | AD0 to AD7 |
| | Port 1 | A8 to A15 | AD8 to AD15 | A8 to A15 | AD8 to AD15 |
| | Port 2 | A16 to A23 | A16 to A23 | A0 to A7 | A0 to A7 |
| Timing Chart | |  | | | |



Note 1: In the cases of 3. and 4., the data bus signals output the addresses because the signals are also used as the address bus. By writing “0” to bit CKOCR<ALEEN>, the ALE signal can be prevented from outputting.

Note 2: After a reset operation, port 0, port 1, and port 2 of the TMP93CS40F function as input ports.

Note 3: In the case of the TMP93CS40F, all the options a. to d. can be made available using the P1CR, P1FC, P2CR, and P2FC registers. ($\overline{EA}$ = VIH, AM8/$\overline{AM16}$ = VIH)

Note 4: In the case of the TMP93CS41F, options 3. and 4. cannot be made available.

### 3.6.3 Control Registers

Table 3.6.1 shows control registers.

One block of the address areas is controlled by each of the 1-byte CS/WAIT control registers B0CS, B1CS, and B2CS.

(1) Master enable bits

Bit7 of the control registers (B0E, B1E, and B2E) are master bits used to specify setting enable (1) or disable (0).

Resetting sets B0E and B1E to disable (0) and B2E to enable (1).

(2) CS/CAS waveform select

Bit5 of the control registers (B0CAS, B1CAS, and B2CAS) are used to specify the waveform mode output from the chip select pin (from $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$, or from $\overline{\text{CAS0}}$ to $\overline{\text{CAS2}}$). Setting these bits to 0 specifies $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ waveforms; setting them to 1 specifies $\overline{\text{CAS0}}$ to $\overline{\text{CAS2}}$ waveforms.

Resetting clears bits 5 to 0.

(3) Data bus size select

Bit4 (B0BUS, B1BUS, and B2BUS) of the control register is used to specify data bus size. Setting this bit to 0 accesses the memory in 16-bit data bus mode; setting it to 1 accesses the memory in 8-bit data bus mode.

Changing data bus size depending on the access address is called dynamic bus sizing. Table 3.6.2 shows the details of the bus operation.

This bit is changed by changing the state of the AM8/$\overline{\text{AM16}}$ pin.

(4) Wait control

Control register bits 3 and 2 <B0W1:0, B1W1:0, and B2W1:0> are used to specify the number of waits. Setting these bits to 00 inserts a 2-state wait regardless of the $\overline{\text{WAIT}}$ pin status. Setting them to 01 inserts a 1-state wait regardless of the $\overline{\text{WAIT}}$ status. Setting them to 10 inserts a 1-state wait and samples the $\overline{\text{WAIT}}$ pin status. If the pin is Low, inserting the wait maintains the bus cycle until the pin goes high. Setting them to 11 completes the bus cycle without a wait, regardless of the $\overline{\text{WAIT}}$ pin status.

Resetting sets these bits to 00 (2-state wait mode).

(5) Address area specification

Control register bits 1 and 0 <B0C1:0, B1C1:0, and B2C1:0> are used to specify the target address area. Setting these bits to 00 enables settings (e.g., $\overline{\text{CS}}$ output, wait state, and bus size) as follows:

* The CS0 setting is enabled when the address space 7F00H to 7FFFH is accessed.
* The CS1 setting is enabled when the address space 880H to 7FFFH is accessed.
* The CS2 setting is enabled when the address space 8000H to 3FFFFFH is accessed in the TMP93CS41, which does not have a built-in ROM.

The CS2 setting is enabled when the address space 18000H to 3FFFFFH is accessed in the TMP93CS40, which has built-in ROM.

Setting these bits to 01 enables settings ($\overline{\text{CS}}$ output, wait state...) for all CS's blocks and outputs a low strobe signal (from $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$, or from $\overline{\text{CAS0}}$ to $\overline{\text{CAS2}}$) from the chip select pins when the address space 400000H to 7FFFFFH is accessed. Setting these bits to 10 enables the address space 800000H to BFFFFFH to be accessed. Setting these bits to 11 enables the address space C00000H to FFFFFFH to be accessed.

Table 3.6.1 Chip Select/Wait Control Register

| Code | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| B0CS | Block0 CS/WAIT control register | 0068H | B0E | | B0CAS | B0BUS | B0W1 | B0W0 | B0C1 | B0C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: Master bit of bit0 to 6 | | 0: $\overline{\text{CS0}}$<br>1: $\overline{\text{CAS0}}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 7F00H to 7FFFH<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |
| B1CS | Block1 CS/WAIT control register | 0069H | B1E | | B1CAS | B1BUS | B1W1 | B1W0 | B1C1 | B1C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: Master bit of bit0 to 6 | | 0: $\overline{\text{CS1}}$<br>1: $\overline{\text{CAS1}}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 880H to 7FFFH<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |
| B2CS | Block2 CS/WAIT control register | 006AH | B2E | | B2CAS | B2BUS | B2W1 | B2W0 | B2C1 | B2C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 1 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: Master bit of bit0 to 6 | | 0: $\overline{\text{CS2}}$<br>1: $\overline{\text{CAS2}}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 8000H to<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |

Note: Read-modify-write is prohibited for B0CS, B1CS, and B2CS.

Table 3.6.2 Dynamic Bus Sizing

| Operand Data Size | Operand Start Address | Memory Data Size | CPU Address | CPU Data | |
|---|---|---|---|---|---|
| | | | | D15 to D8 | D7 to D0 |
| 8 bits | 2n + 0 | 8 bits | 2n + 0 | xxxxx | b7 to b0 |
| | (Even number) | 16 bits | 2n + 0 | xxxxx | b7 to b0 |
| | 2n + 1 | 8 bits | 2n + 1 | xxxxx | b7 to b0 |
| | (Odd number) | 16 bits | 2n + 1 | b7 to b0 | xxxxx |
| 16 bits | 2n + 0 | 8 bits | 2n + 0 | xxxxx | b7 to b0 |
| | (Even number) | | 2n + 1 | xxxxx | b15 to b8 |
| | | 16 bits | 2n + 0 | b15 to b8 | b7 to b0 |
| | 2n + 1 | 8 bits | 2n + 1 | xxxxx | b7 to b0 |
| | (Odd number) | | 2n + 2 | xxxxx | b15 to b8 |
| | | 16 bits | 2n + 1 | b7 to b0 | xxxxx |
| | | | 2n + 2 | xxxxx | b15 to b8 |
| 32 bits | 2n + 0 | 8 bits | 2n + 0 | xxxxx | b7 to b0 |
| | (Even number) | | 2n + 1 | xxxxx | b15 to b8 |
| | | | 2n + 2 | xxxxx | b23 to b16 |
| | | | 2n + 3 | xxxxx | b31 to b24 |
| | | 16 bits | 2n + 0 | b15 to b8 | b7 to b0 |
| | | | 2n + 2 | b31 to b24 | b23 to b16 |
| | 2n + 1 | 8 bits | 2n + 1 | xxxxx | b7 to b0 |
| | (Odd number) | | 2n + 2 | xxxxx | b15 to b8 |
| | | | 2n + 3 | xxxxx | b23 to b16 |
| | | | 2n + 4 | xxxxx | b31 to b24 |
| | | 16 bits | 2n + 1 | b7 to b0 | xxxxx |
| | | | 2n + 2 | b23 to b16 | b15 to b8 |
| | | | 2n + 4 | xxxxx | b31 to b24 |

xxxxx: During a read, data input to the bus is ignored. While writing, the bus is at high impedance and the write strobe signal remains in active.

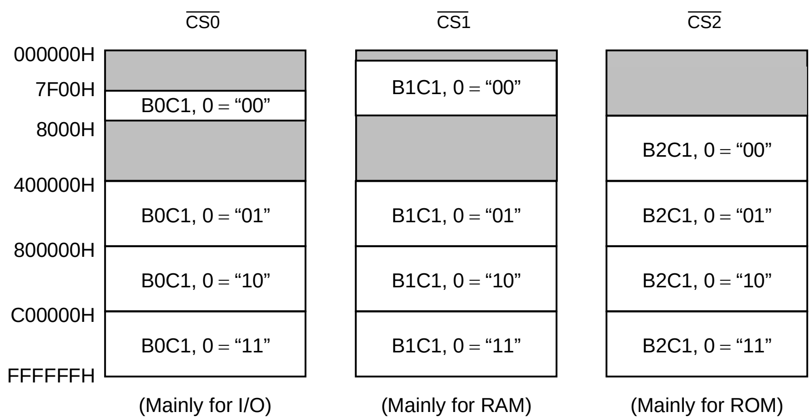
### 3.6.4 Chip Select Addresses Image

An image of the actual addresses which can be specified by chip select is shown below. Out of the whole memory area, address areas that can be specified are divided into four parts. Addresses from 000000H to 3FFFFFH are further divided as follows: 7F00H to 7FFFH is specified for CS0; 880H to 7FFFH, for CS1; and 8000H to 3FFFFFH, for CS2. The reason is that a device other than ROM (e.g., RAM or I/O) might be connected externally.

7F00 to 7FFFH (256 bytes) designated as CS0 are mapped mainly for possible expansions to external I/O.

880H to 7FFFH (approx. 31 Kbytes) designated as CS1 are mapped mainly for possible extensions to external RAM.

8000H to 3FFFFFH (approx. 4 Mbytes) designated as CS2 are mapped mainly for possible extensions to external ROM. After resetting, CS2 is enabled in a 16-bit bus and 2-wait configuration. In the case of the TMP93CS41, which does not have a built-in ROM, the program is externally read at address 8000H with these settings (16-bit bus, 2 waits). With the TMP93CS40, which does have a built-in ROM, addresses from 8000H to 17FFFH are used as the internal ROM area; CS2 is disabled in this area. After resetting, the CPU reads the program from the built-in ROM in 16-bit bus, 0-wait mode.




Note 1: Access priority is highest for built-in I/O, then built-in memory, and lowest for the chip select/wait controller.

Note 2: External areas other than $\overline{CS0}$ to $\overline{CS2}$ are accessed in 0 wait mode.
For the TMP93CS40, the data bus width is fixed at 16 bits. For the TMP93CS41, the data bus width is 16 bits when AM8/$\overline{AM16}$ = 0, and 8 bits when AM8/$\overline{AM16}$ = 1.

When using the chip select/wait controller, do not specify the same address area more than once. (However, when specifications overlap, only one of them will be utilized. For example, when addresses 7F00H to 7FFFH for CS0 are specified at the same time as 880H to 7FFFH for CS1, only the CS0 setting and pin will be active.)

Note 3: When the bus is released ($\overline{BUSAK}$ = "0"), the $\overline{CS0}$ to $\overline{CS2}$ pins are also released (the output buffer is OFF). For further information about the state of pins, refer to the note about the bus release in section 3.5 "Functions of Ports".

### 3.6.5 Example of Usage

(1) Example of usage -1

Figure 3.6.1 is an example in which an external memory is connected to the TMP93CS41. In this example, a ROM is connected using a 16-bit bus; a RAM is connected using an 8-bit bus.

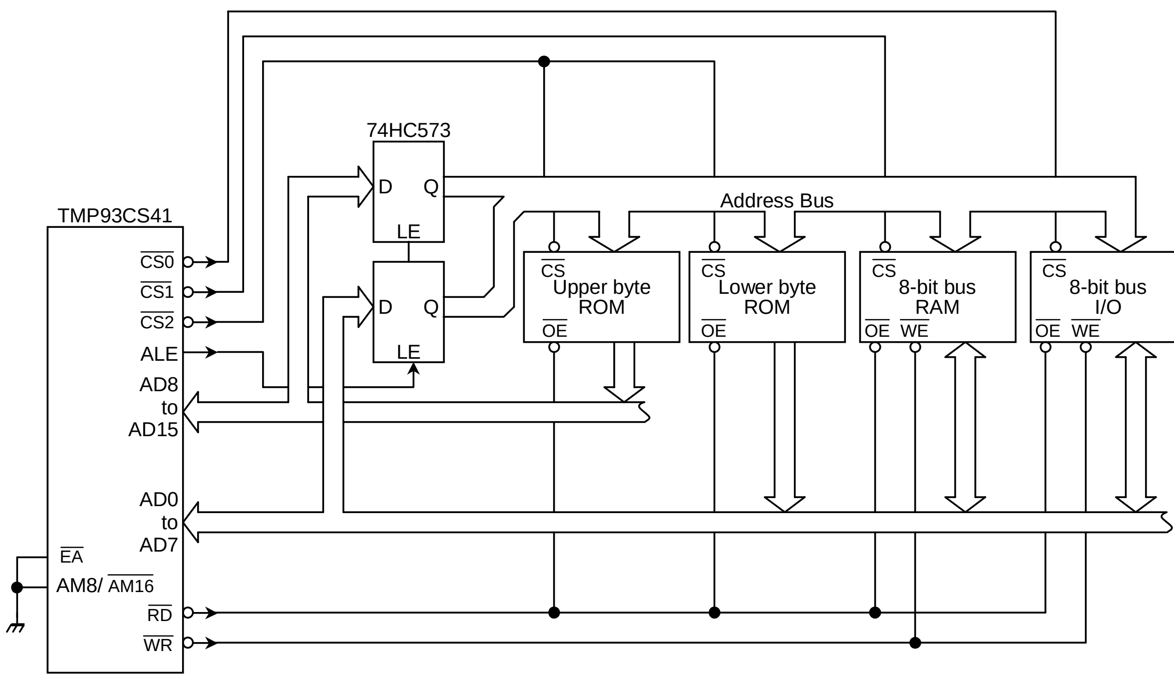



Figure 3.6.1 Example of External Memory Connection (ROM = 16 bits, RAM and I/O = 8 bits)

Resetting sets pins $\overline{CS0}$ to $\overline{CS2}$ to input port mode. $\overline{CS0}$ and $\overline{CS1}$ are set high due to an internal pull-up resistor; $\overline{CS2}$ is set low due to an internal pull-down resistor. The program used to set these pins is as follows.

```
P4CR    EQU     0EH
P4FC    EQU     10H
B0CS    EQU     68H
B1CS    EQU     69H
B2CS    EQU     6AH
LD      (B0CS), 1X010000B      ; CS0 = 8 bits, 2 waits, 7F00H to 7FFFH
LD      (B1CS), 1X011100B      ; CS1 = 8 bits, 0 waits, 880H to 7EFFH
LD      (B2CS), 1X000100B      ; CS2 = 16 bits, 1 wait, 8000H to 3FFFFFH
LD      (P4CR), XXXXX111B      } CS0 , CS1 , CS2 output mode setting
LD      (P4FC), XXXXX111B
```

X: Don’t care

(2) Example of usage -2

Figure 3.6.2 is an example in which an external memory is connected to the TMP93CS41. In this example, a ROM, RAM, and I/O are each connected using an 8-bit bus.

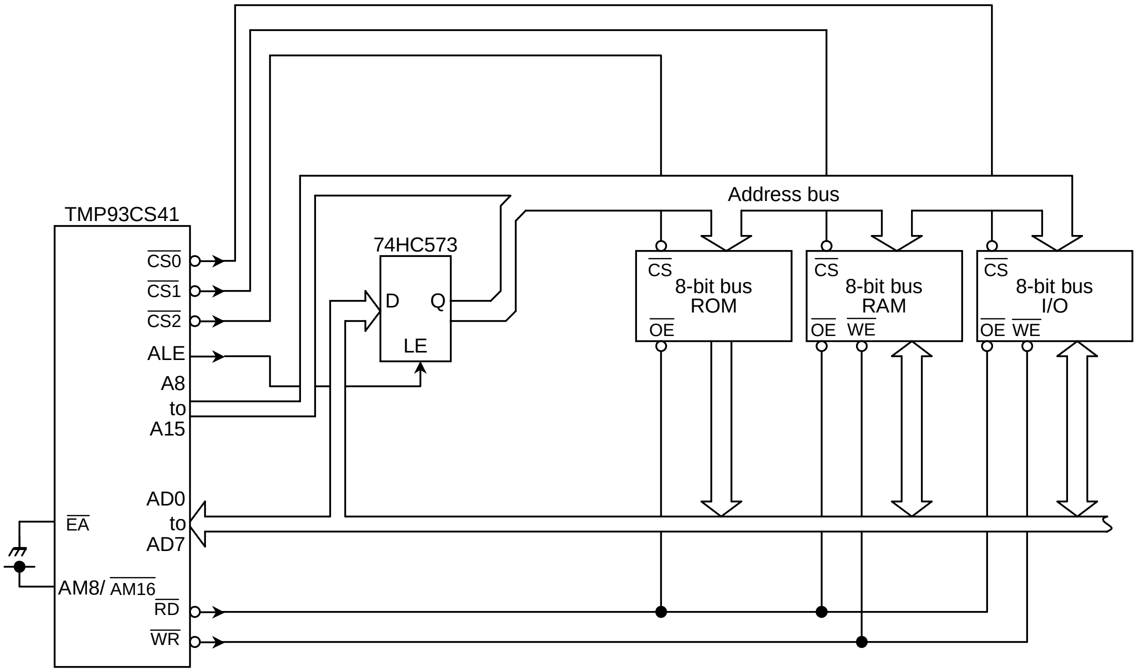



Figure 3.6.2 Example of External Memory Connection (ROM, RAM and I/O = 8 Bits)

Resetting sets pins $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ to input port mode. $\overline{\text{CS0}}$ and $\overline{\text{CS1}}$ are set high due to an internal pull-up resistor; $\overline{\text{CS2}}$ is set low due to an internal pull-down resistor. The program used to set these pins is as follows.

```
P4CR   EQU    0EH
P4FC   EQU    10H
B0CS   EQU    68H
B1CS   EQU    69H
B2CS   EQU    6AH
LD     (B0CS), 1X010000B       ; CS0 = 8 bits, 2 waits, 7F00H to 7FFFH
LD     (B1CS), 1X011100B       ; CS1 = 8 bits, 0 waits, 880H to 7EFFH
LD     (B2CS), 1X000100B       ; CS2 = 8 bits, 1 wait, 8000H to 3FFFFFH
LD     (P4CR), XXXXX111B     }
LD     (P4FC), XXXXX111B     }  CS0, CS1, CS2 output mode setting
```

X: Don’t care

(3) Example of usage -3

Figure 3.6.3 is an example in which an external memory is connected to the TMP93CS40. In this example, a 128-Kbyte ROM is connected using a 16-bit bus, and a 256-Kbyte RAM is connected using a 16-bit bus.

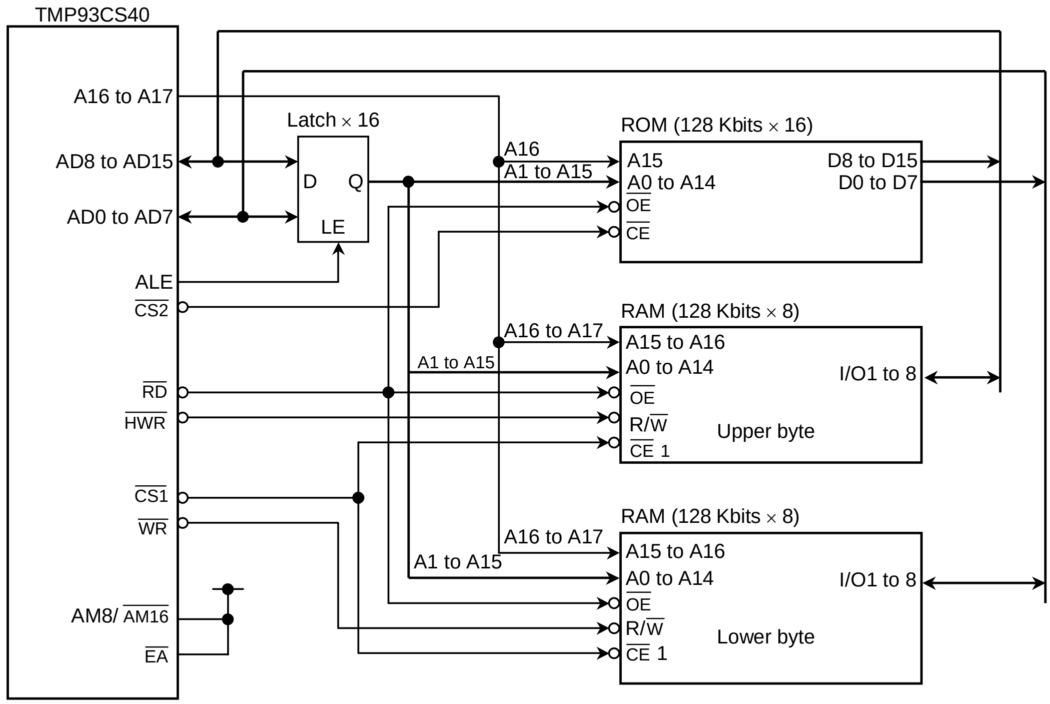



Figure 3.6.3 Example of External Memory Connection (ROM & RAM = 16 bits)

The TMP93CS40 has built-in ROM and RAM. When ROM and RAM have insufficient capacity, it is possible to connect an external memory following the usage examples for this purpose. In this example, the memory configuration is as follows.

| Memory | | Memory Size | Address | $\overline{CS}$ Pin | Data Bus |
|---|---|---|---|---|---|
| ROM | Internal | 64 Kbytes | 008000H to 017FFFH | – | 16 bits |
| | External | 128 Kbytes | 400000H to 41FFFFH | $\overline{CS2}$ | 16 bits |
| SRAM | Internal | 2 Kbytes | 000080H to 00087FH | – | 16 bits |
| | External | 256 Kbytes | 800000H to 83FFFFH | $\overline{CS2}$ | 16 bits |

## 3.7 8-Bit Timers

The TMP93CS40 and S41 contain two 8-bit timers (Timers 0 and 1), each of which can be operated independently. The cascade connection also allows these timers to be used together as a 16-bit timer. The following four operating modes are supported for the 8-bit timers:

- 8-bit interval timer mode (2 timers)
- 16-bit interval timer mode (1 timer)
- 8-bit programmable square wave pulse generation (PPG: Variable duty with variable cycle) output mode (1 timer)
- 8-bit pulse width modulation (PWM: Variable duty with constant cycle) output mode (1 timer)

Figure 3.7.1 shows the block diagram of the 8-bit timers (Timer 0 and timer 1).

Each timer consists of an 8-bit up counter, 8-bit comparator, and 8-bit timer register. Besides, one timer flip-flop (TFF1) is provided for the pair consisting of timer 0 and timer 1.

Among the input clock sources for the timers, the internal clocks of ϕT1, ϕT4, ϕT16, and ϕT256 are obtained from the 9-bit prescaler shown in Figure 3.7.2.

The operation modes and timer flip-flops of the 8-bit timers are controlled by the three control registers TMOD, TFFCR, and TRUN.

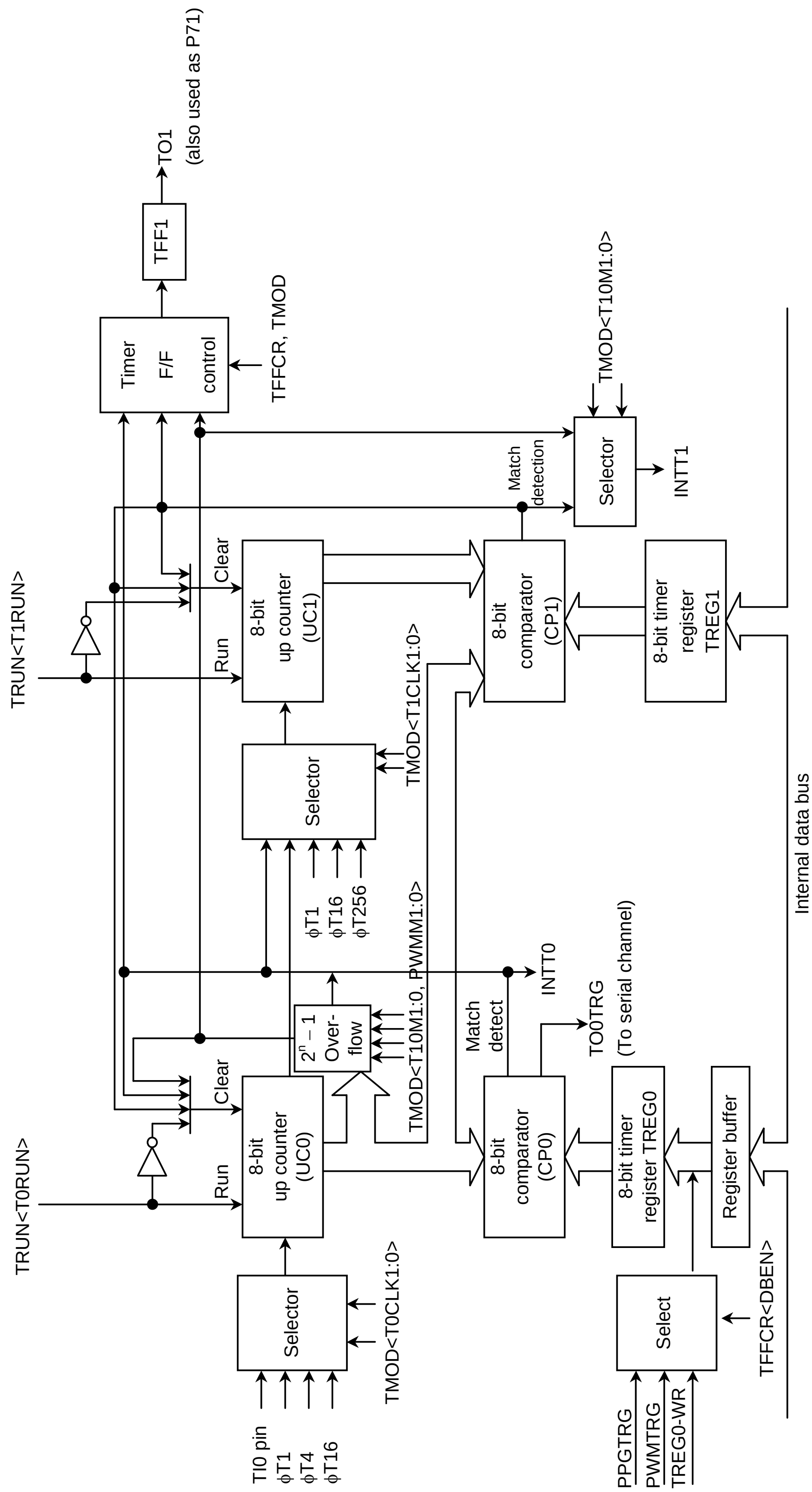



Figure 3.7.1 Block Diagram of 8-Bit Timers (Timers 0 and 1)

1. Prescaler

There are 9-bit prescaler and prescaler clock selection registers to generate input clock signals for the 8-bit timers 0 and 1, the 16-bit timers 4 and 5, and the serial interfaces 0 and 1.

Figure 3.7.2 shows the corresponding block diagram, and Table 3.7.1 shows prescaler clock signal resolution into 8 and 16-bit timers.

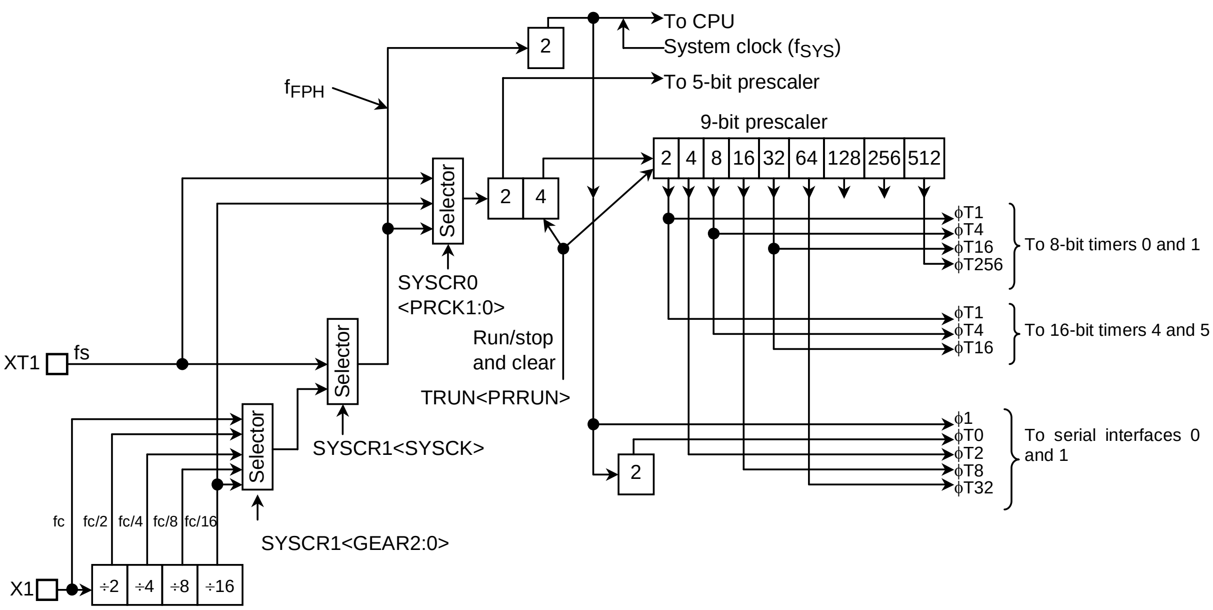



Figure 3.7.2 Block Diagram of the Prescaler

Table 3.7.1 Prescaler Clock Resolution to 8 and 16-Bit Timers

at fc = 20 MHz, fs = 32.768 kHz

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | Prescaler Clock Resolution | | | |
|---|---|---|---|---|---|---|
| | | | φT1 | φT4 | φT16 | φT256 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | $fs/2^3$ (244 µs) | $fs/2^5$ (977 µs) | $fs/2^7$ (3.9 ms) | $fs/2^{11}$ (62.5 ms) |
| 0 (fc) | | 000 (fc) | $fc/2^3$ (0.4 µs) | $fc/2^5$ (1.6 µs) | $fc/2^7$ (6.4 µs) | $fc/2^{11}$ (102.4 µs) |
| | | 001 (fc/2) | $fc/2^4$ (0.8 µs) | $fc/2^6$ (3.2 µs) | $fc/2^8$ (12.8 µs) | $fc/2^{12}$ (204.8 µs) |
| | | 010 (fc/4) | $fc/2^5$ (1.6 µs) | $fc/2^7$ (6.4 µs) | $fc/2^9$ (25.6 µs) | $fc/2^{13}$ (409.6 µs) |
| | | 011(fc/8) | $fc/2^6$ (3.2 µs) | $fc/2^8$ (12.8 µs) | $fc/2^{10}$ (51.2 µs) | $fc/2^{14}$ (0.82 ms) |
| | | 100 (fc/16) | $fc/2^7$ (6.4 µs) | $fc/2^9$ (25.6 µs) | $fc/2^{11}$ (102.4 µs) | $fc/2^{15}$ (1.64 ms) |
| XXX | 01 (Low-frequency clock) | XXX | $fs/2^3$ (244 µs) | $fs/2^5$ (977 µs) | $fs/2^7$ (3.9 ms) | $fs/2^{11}$ (62.5 ms) |
| XXX | 10 (Note) (fc/16 clock) | XXX | $fc/2^7$ (6.4 µs) | $fc/2^9$ (25.6 µs) | $fc/2^{11}$ (102.4 µs) | $fc/2^{15}$ (1.64 ms) |

16-bit timer (φT1 to φT16)

8-bit timer (φT1 to φT256)

XXX: Don't care

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

The timer clock selected among $f_{FPH}$, fc/16, and fs is divided by 4 and input to this prescaler. The selection is made by system clock control register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to 00, which selects the $f_{FPH}$ clock input to be divided by 4.

The 8-bit timers 0 and 1 select among 4 clock inputs: ϕT1, ϕT4, ϕT16, and ϕT256 of the prescaler output.

This prescaler can be run or stopped by the timer control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to "1". The prescaler is cleared to zero and stops operation when <PRRUN> is set to "0".

Resetting clears <PRRUN> to "0" and stops the prescaler.

When the IDLE1 mode (in which only the oscillator operates) is used, set TRUN <PRRUN> to "0" to reduce the power consumption of the prescaler before the "HALT" instruction is executed.

2. Up counter

This is an 8-bit binary counter which counts up by the input clock pulse specified by TMOD.

The input clock of timer 0 is selected from among the external clock from the TI0 pin, and the three internal clocks ϕT1, ϕT4, and ϕT16, according to the value set in the TMOD register.

The input clock used by timer 1 depends on the operation mode. When 16-bit timer mode is set, the overflow output of timer 0 is used as the input clock. When any mode other than 16-bit timer mode is set, the input clock is selected from among the internal clocks ϕT1, ϕT16, and ϕT256 as well as the comparator output (Match detection signal) of timer 0 according to the set value of the TMOD register.

Example: When TMOD<T10M1:0> = 01, the overflow output of timer 0 becomes the input clock of timer 1 (16-bit timer mode).

When TMOD<T10M1:0> = 00 and TMOD<T1CLK1:0> = 01, ϕT1 becomes the input of timer 1 (8-bit timer mode).

Similarly, operation mode is also set by the TMOD register. When reset, it is initialized to TMOD<T01M1:0> = 00 whereby the up counter is placed in the 8-bit timer mode.

The counting and stop and clear of the up counter can be controlled for each interval timer by the timer operation control register TRUN. When reset, all up counters will be cleared to stop the timers.

3. Timer registers

These are 8-bit registers for setting a time interval. When the values of the timer registers match the values of the corresponding up counters, the comparator match detect signal becomes active. If the set value is 00H, this signal becomes active when the up counter overflows.

Timer register TREG0 has a double buffer.

The timer flip-flop control register TFFCR<DBEN> bit controls whether the double buffer structure should be enabled or disabled. It is disabled when <DBEN> = "0" and enabled when it is set to "1".

In the double buffer enable state, the data set in the register buffer are transferred to the timer register when the $2^n - 1$ overflow occurs in PWM mode, or at the PPG cycle in PPG mode. Therefore, during timer mode, the double buffer cannot be used.

Upon resetting, TFFCR will be initialized to <DBEN> = "0", disabling the double buffer. To use the double buffer, write data in the timer register, set <DBEN> to "1", and write the following data in the register buffer.

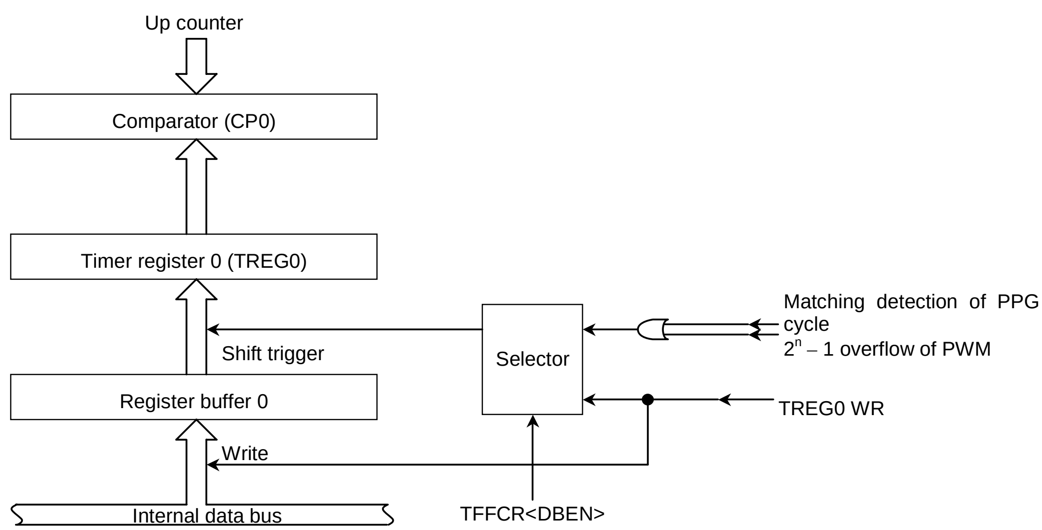



Figure 3.7.3 Configuration of Timer Register 0

Note: The timer register and the register buffer are allocated at the same memory address. When <DBEN> = 0, the same value is written in the register buffer and in the timer register, while when <DBEN> = 1 the value is written only into the register buffer.

The memory address of each timer register is as follows.

TREG0: 000022H

TREG1: 000023H

Both of these registers are write-only and cannot be read.

4. Comparator

   A comparator compares the value in the up counter with the values to which the timer register is set. When they match, the up counter is cleared to zero and an interrupt signal (INTT0 and INTT1) is generated. If the timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

5. Timer flip-flop

   The timer flip-flop (TFF1) is a flip-flop inverted by the match detect signal (8-bit comparator output).

   Inverting is enabled or disabled by the timer flip-flop control register TFFCR<TFF1IE>.

   After a reset operation, the value of TFF1 is undefined. Writing “01” or “10” to TFFCR<TFF1C1:0> sets “0” or “1” to TFF1. Additionally, writing 00 to this bit inverts the value of TFF1. (Software inversion.)

   The value in TFF1 can be output to the TO1 pin (also used as P71). When using the TFF1 contents as the timer output, the timer flip-flop should be set by the port 7 function register P7FC beforehand.

TRUN (0020H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PRRUN | | T5RUN | T4RUN | P1RUN | P0RUN | T1RUN | T0RUN |
| Read/Write | R/W | | R/W | | | | | |
| After reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Prescaler and timer run/stop control<br>0: Stop and clear<br>1: Run (Count up) | | | | | | | |

Count operation

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

PRRUN: Operation of prescaler
T5RUN: Operation of 16-bit timer (Timer 5)
T4RUN: Operation of 16-bit timer (Timer 4)
P1RUN: Operation of PWM timer (PWM1/timer 3)
P0RUN: Operation of PWM timer (PWM0/timer 2)
T1RUN: Operation of 8-bit timer (Timer 1)
T0RUN: Operation of 8-bit timer (Timer 0)

Note: TRUN<bit6> is read as “1”.

SYSCR0 (006EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | XEN | XTEN | RXEN | RXTEN | RSYSCK | WUEF | PRCK1 | PRCK0 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | High-frequency oscillator (fc)<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs)<br>0: Stop<br>1: Oscillation | High-frequency oscillator (fc) after release of the STOP mode<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs) after release of the STOP mode<br>0: Stop<br>1: Oscillation | Clock selection after release of the STOP mode<br>0: fc<br>1: fs | Warm-up timer<br>(Write)<br>0: Don’t care<br>1: Start timer<br>(Read)<br>0: End warm up<br>1: Continue warm up | Select prescaler clock<br>00: $f_{FPH}$<br>01: fs<br>10: fc/16<br>11: (Reserved) | |

Select prescaler clock setting

| | |
|---|---|
| 00 | $f_{FPH}$ |
| 01 | fs |
| 10 | fc/16 |
| 11 | (Reserved) |

Clock divided by 4

Figure 3.7.4 Timer Operation Control Register/System Clock Control Register

TMOD (0024H)

Prohibit read-modify-write

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | T10M1 | T10M0 | PWMM1 | PWMM0 | T1CLK1 | T1CLK0 | T0CLK1 | T0CLK0 |
| Read/Write | W | | | | | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Operation mode<br>00: 8-bit timer<br>01: 16-bit timer<br>10: 8-bit PPG<br>11: 8-bit PWM | | PWM cycle<br>00: –<br>01: $2^6 - 1$<br>10: $2^7 - 1$<br>11: $2^8 - 1$ | | Source clock of timer 1<br>00: TO0TRG<br>01: $\phi$T1<br>10: $\phi$T16<br>11: $\phi$T256 | | Source clock of timer 0<br>00: TI0 pin input<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |

Input clock signal of timer 0

| | |
|---|---|
| 00 | External input (TI0 pin input) |
| 01 | $\phi$T1 (Prescaler) |
| 10 | $\phi$T4 (Prescaler) |
| 11 | $\phi$T16 (Prescaler) |

Input clock of timer 1

| | TMOD<T10M1:0> ≠ 01 | TMOD <T10M1:0> = 01 |
|---|---|---|
| 00 | Comparator output of timer 0 | Overflow output of timer 0<br><br>(16-bit timer mode) |
| 01 | Internal clock $\phi$T1 | |
| 10 | Internal clock $\phi$T16 | |
| 11 | Internal clock $\phi$T256 | |

Select PWM cycle

| | |
|---|---|
| 00 | – |
| 01 | $2^6 - 1$ |
| 10 | $2^7 - 1$ |
| 11 | $2^8 - 1$ |

Set the operation modes of timers 0 and 1.

| | |
|---|---|
| 00 | Two 8-bit timers<br>(Timer 0 and timer 1) |
| 01 | 16-bit timer |
| 10 | 8-bit PPG output |
| 11 | 8-bit PWM output (Timer 0)<br>8-bit timer (Timer 1) |

Figure 3.7.5 Timer Mode Control Register (TMOD)

TFFCR (0025H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | DBEN | TFF1C1 | TFF1C0 | TFF1IE | TFF1IS |
| Read/Write | | | | R/W | W | | R/W | |
| After reset | | | | 0 | 1 | 1 | 0 | 0 |
| Function | | | | Double buffer<br>0: Disable<br>1: Enable | 00: Invert TFF1<br>01: Set TFF1<br>10: Clear TFF1<br>11: Don't care | | TFF1 inversion trigger<br>0: Disable<br>1: Enable | TFF1 inversion source<br>0: Timer 0<br>1: Timer 1 |

Select inverse signal of timer F/F1
("Don't care" except in 8-bit timer mode)

| <TFF1IS> \ TMOD<T10M1:0> | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 0 | Inversion by timer 0 match signal | 16-bit timer mode Inversion by match signal | PPG mode Inversion by matching signals of both timer 0 and timer 1 | PWM mode Inversion by matching and overflow signals of timer 0 |
| 1 | Inversion by timer 1 match signal | | | |

Inversion of timer flip-flop 1 (TFF1)

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

Control of timer flip-flop 1 (TFF1)

| | |
|---|---|
| 00 | Invert the value of TFF1 (Software inversion) |
| 01 | Set TFF1 to "1". |
| 10 | Clear TFF1 to "0". |
| 11 | Don't care |

Double buffer control of TREG0

| | |
|---|---|
| 0 | Disable double buffer |
| 1 | Enable double buffer |

Note: TFFCR<bit7:5>, <bit3:2> are read as "1".

Figure 3.7.6 Timer Flip-Flop Control Register (TFFCR)

The operation of 8-bit timers will be described below:

(1) 8-bit timer mode

Two interval timers, designated “0” and “1”, can be used independently as 8-bit interval timers. All interval timers operate in the same manner, and thus only the operation of timer 1 will be explained below.

1. Generating interrupts in a fixed cycle

To generate timer 1 interrupts at constant intervals using timer 1 (INTT1), first stop timer 1. Set the operation mode and input clock speed by setting TMOD, and the cycle time by setting TREG1. Then, enable interrupt INTT1 and start the counting of timer 1.

Example: To generate timer 1 interrupt every 1 second at fs = 32 kHz, set each register in the following manner.

Clock condition
System clock: Low frequency (fs)
Prescaler clock: Low frequency (fs)

| | | 7 (MSB) | 6 | 5 | 4 | 3 | 2 | 1 | 0 (LSB) | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | 0 | – | Stop timer 1, and clear it to “0”. |
| TMOD | ← | 0 | 0 | X | X | 1 | 0 | – | – | Set the 8-bit timer mode, and select $\phi$T16 (4 ms at fs = 32 kHz) as the input clock. |
| TREG1 | ← | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | Set the timer register 1, 1 s ÷ $\phi$T16 = 250 = FAH (H signifies hexadecimal). |
| INTET10 | ← | 1 | 1 | 0 | 1 | – | – | – | – | Enable INTT1, and set it to “Level 5”. |
| TRUN | ← | 1 | X | – | – | – | – | 1 | – | Start timer 1 counting. |

X: Don’t care, –: No change

Use Table 3.7.1 for selecting the input clock.

Note: The input clock choices available for timer 0 and timer 1 differ from each other as follows.

Timer 0: TI0 input, $\phi$T1, $\phi$T4, $\phi$T16

Timer 1: Match detect signal of timer 0, $\phi$T1, $\phi$T16, $\phi$T256

2. Generating a 50% duty, square-wave pulse

The timer flip-flop (TFF1) is inverted at constant intervals, and its status is output to timer output pin (TO1).

Example: To output a 2.4 μs square wave pulse from the TO1 pin at fc = 20 MHz, set each register by the following procedures. Either timer 0 or timer 1 may be used, but this example uses timer 1.

* Clock condition

| | |
|---|---|
| System clock: | High frequency (fc) |
| Clock gear: | 1 (fc) |
| Prescaler clock: | $f_{FPH}$ |

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | 0 | – | Stop timer 1, and clear it to "0". |
| TMOD | ← | 0 | 0 | X | X | 0 | 1 | – | – | Set the 8-bit timer mode, and select φT1 (0.4 μs at fc = 20 MHz) as the input clock cycle time. |
| TREG1 | ← | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | Set the timer register at 2.4 μs ÷ φT1 ÷ 2 = 3. |
| TFFCR | ← | – | – | – | – | 1 | 0 | 1 | 1 | Clear TFF1 to "0", and set to invert by the match detect signal from timer 1. |
| P7CR | ← | X | X | X | X | – | – | 1 | – | Select P71 as TO1 pin. |
| P7FC | ← | X | X | X | X | – | – | 1 | X | |
| TRUN | ← | 1 | X | – | – | – | – | 1 | – | Start timer 1 counting. |

X: Don't care, –: No change

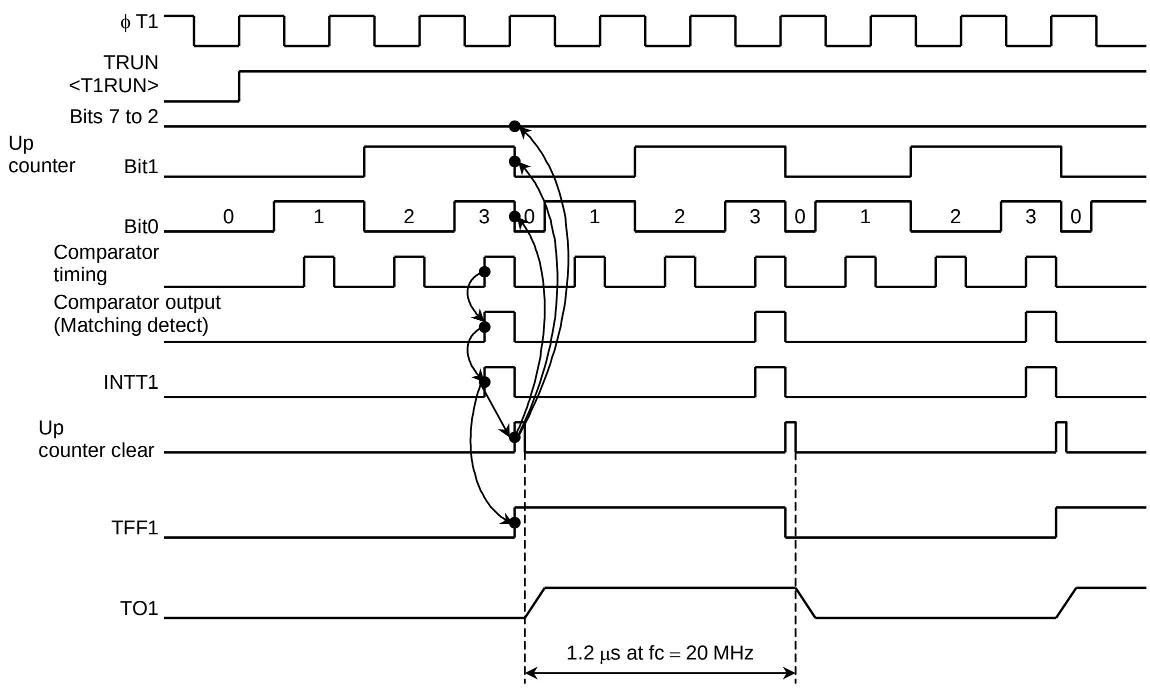



Figure 3.7.7 Square Wave (50% duty) Output Timing Chart

3. Making timer 1 count up by matching the signal from the timer 0 comparator

Set the 8-bit timer mode, and set the comparator output of timer 0 as the input clock to timer 1.

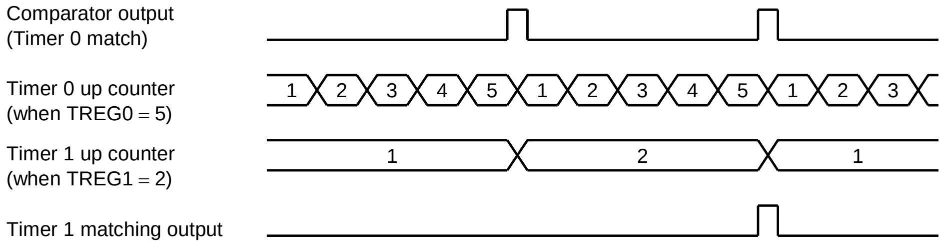



Figure 3.7.8 Timer 1 Count Up Regulated by Timer 0

(2) 16-bit timer mode

A 16-bit interval timer is configured by using timer 0 and timer 1 as a pair.

Setting timer mode register TMOD<T10M1:0> to "01" establishes a 16-bit timer mode.

When set in 16-bit timer mode, the overflow output of timer 0 will become the input clock of timer 1, regardless of the set value of TMOD<T1CLK1:0>. Table 3.7.1 shows the selection of timer 0 input clock.

The lower 8 bits of the timer (interrupt) cycle are set by the timer register TREG0, and the upper 8 bits are set by TREG1. Note that TREG0 always must be set first. (Writing data into TREG0 disables the comparator temporarily, and the comparator is restarted by writing data into TREG1.)

Setting example: To generate an interrupt INTT1 every 0.4 seconds at fc = 20 MHz, set the following values for timer registers TREG0 and TREG1.

Clock condition
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: $f_{FPH}$

When counting with the $\phi$T16 input clock (6.4 μs at 20 MHz)

$$0.4 \text{ s} \div 6.4\ \mu\text{s} = 62500 = \text{F424H}$$

Therefore, set TREG1 = F4H and TREG0 = 24H, respectively.

The comparator signal is output from timer 0 each time the up counter UC0 matches TREG0, when the up counter UC0 is not to be cleared.

With the timer 1 comparator, the match detect signal is output at each comparator check when the up counter UC1 and TREG1 values are found to match. When the match detect signal is output simultaneously from the comparators of both timer 0 and timer 1, the up counters UC0 and UC1 are cleared to 0, and the interrupt INTT1 is generated. If inversion is enabled, the value of the timer flip-flop TFF1 is inverted.

Example: When TREG1 = 04H and TREG0 = 80H

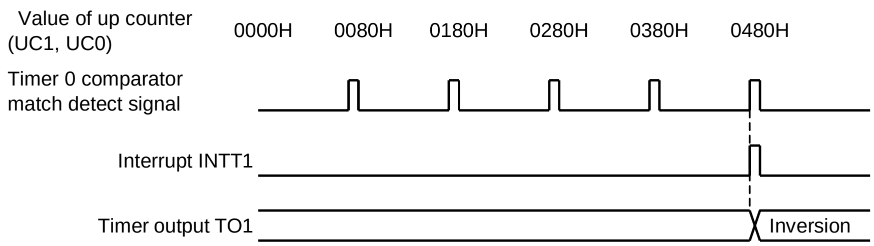



Figure 3.7.9 Timer Output by 16-Bit Timer Mode

(3) 8-bit PPG (Programmable pulse generation) output mode

Square wave pulse can be generated at any frequency and duty by timer 0. The output pulse may be either low-active or high-active. In this mode, timer 1 cannot be used.

Timer 0 outputs a pulse to the TO1 pin (also used as P71).

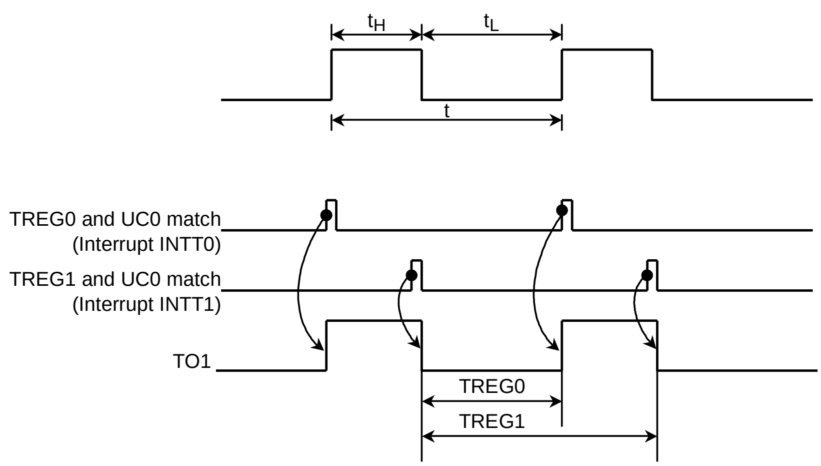



Figure 3.7.10 8-Bit PPG Output Waveforms

In this mode, a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the timer registers TREG0 and TREG1.

However, it is necessary for the set value of TREG0 to be smaller than that of TREG1.

Though the up counter (UC1) of timer 1 is not used in this mode, UC1 should be set for counting by setting TRUN<T1RUN> to "1".

Figure 3.7.11 shows the block diagram for this mode.

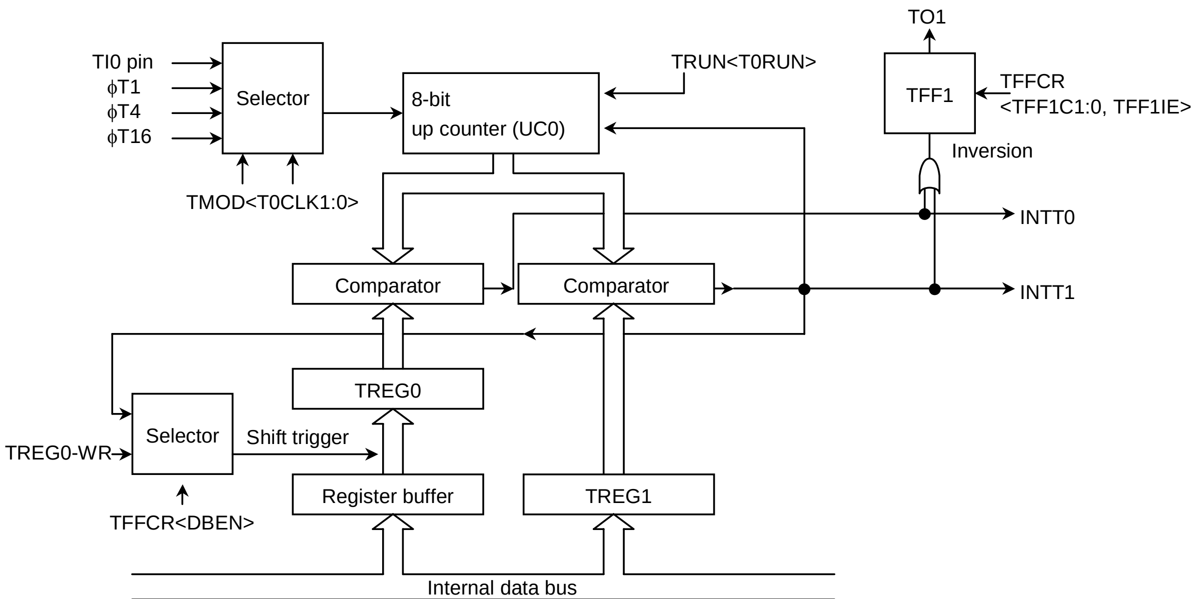



Figure 3.7.11 Block Diagram of 8-Bit PPG Output Mode

When the double buffer of TREG0 is enabled in this mode, the value of the register buffer will be shifted in TREG0 each time TREG1 matches UC0.

Use of the double buffer makes the handling of low duty waves easy (when duty is varied).

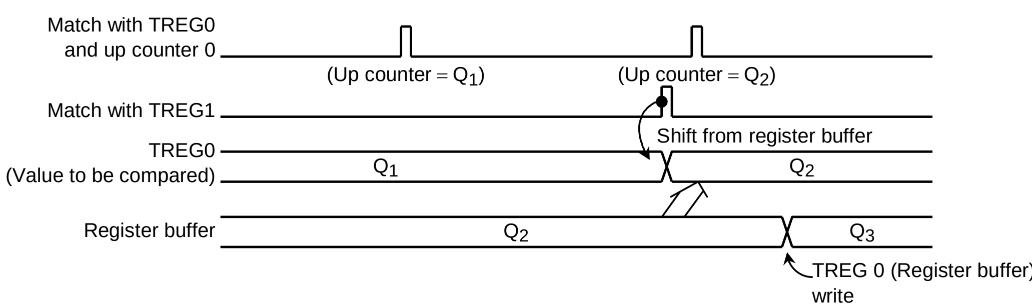
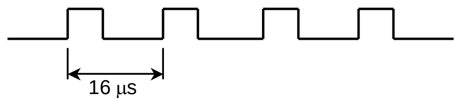



Figure 3.7.12 Operation of Register Buffer

Example: Generating 1/4 duty 62.5 kHz pulse (at fc = 20 MHz)




Clock condition
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: $f_{FPH}$

- Calculate the value to be set in the timer register.

To obtain a frequency of 62.5 kHz, the pulse cycle time t should be: t = 1/62.5 kHz = 16 μs.
Given $\phi$T1 = 0.4 μs (at 20 MHz),

16 μs ÷ 0.4 μs = 40

Consequently, set the timer register 1 (TREG1) to TREG1 = 40 = 28H
and then to obtain a duty of 1/4, t × 1/4 = 16 μs × 1/4 = 4 μs

4 μs ÷ 0.4 μs = 10

Therefore, set timer register 0 (TREG0) to TREG0 = 10 = 0AH.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | 0 | 0 | Stop timers 0 and 1, and clear then. |
| TMOD | ← | 1 | 0 | X | X | X | X | 0 | 1 | Set the 8-bit PPG mode, and select $\phi$T1 as input clock. |
| TREG0 | ← | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | Write “0AH”. |
| TREG1 | ← | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | Write “28H”. |
| TFFCR | ← | – | – | – | X | 0 | 1 | 1 | X | Set TFF1 and enable the inversion. |
| | | | | | | | | | | Writing “10” provides negative logic pulse. |
| P7CR | ← | X | X | X | X | – | – | 1 | – | Set P71 as the TO1 pin. |
| P7FC | ← | X | X | X | X | – | – | 1 | X | |
| TRUN | ← | 1 | X | – | – | – | – | 1 | 1 | Start timer 0 and timer 1 counting. |

X: Don’t care, –: No change

(4) 8-bit PWM output mode

This mode is valid only for timer 0. In this mode, the maximum 8-bit resolution of the PWM pulse can be output.

The PWM pulse is output to the TO1 pin (also used as P71) when using timer 0. Timer 1 can also be used as an 8-bit timer.

Timer output is inverted when the up counter (UC0) matches the set value of timer register TREG0, or when $2^n - 1$ (n = 6, 7, or 8; specified by TMOD<PWMM1:0>) counter overflow occurs. Up counter UC0 is cleared when $2^n - 1$ counter overflow occurs.

To use this PWM mode, the following conditions must be satisfied.

(Set value of timer register) < (Set value of $2^n - 1$ counter overflow)

(Set value of timer register) ≠ 0

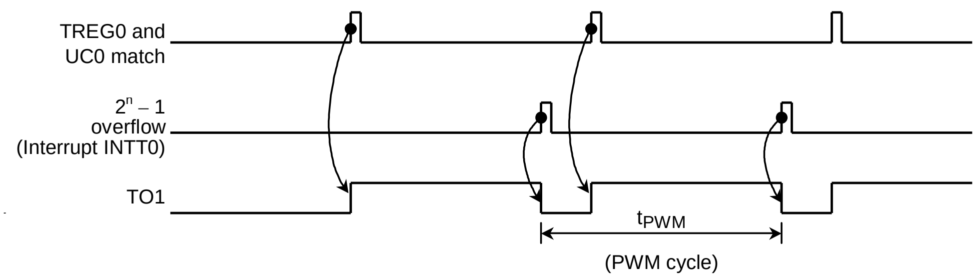



Figure 3.7.13 8-Bit PWM Waveforms

Figure 3.7.14 shows the block diagram of operations in this mode.

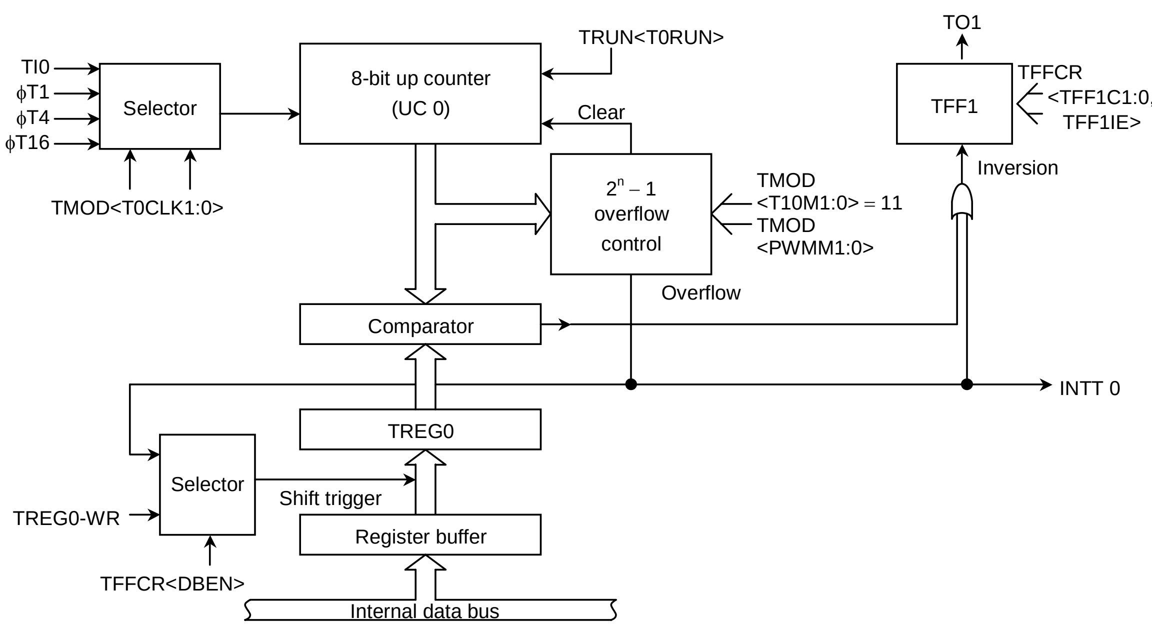



Figure 3.7.14 Block Diagram of Operations in 8-Bit PWM Mode

In this mode, the value of the register buffer will be shifted into TREG0 if a $2^n - 1$ overflow is detected while the double buffer of TREG0 is enabled.

Use of the double buffer makes easy the handling of small-duty waves.

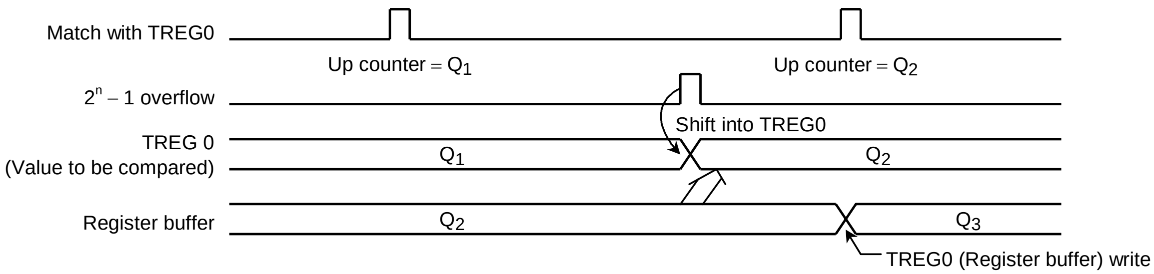
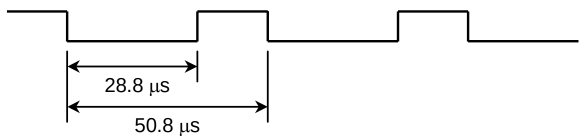



Figure 3.7.15 Operation of Register Buffer

Example: To output the following PWM waves to the TO1 pin at fc = 20 MHz.




Clock conditions
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: $f_{FPH}$

To implement a PWM cycle of 50.8 μs by utilizing $\phi$T1 = 0.4 μs (at fc = 20 MHz),

$$50.8\ \mu s \div 0.4\ \mu s = 127 = 2^n - 1$$

Consequently, n should be set to 7.

As the period of low level is 28.8 μs, for $\phi$T1 = 0.4 μs,

set the following value for TREG0.

$$28.8\ \mu s \div 0.4\ \mu s = 72 = 48H$$

| | | MSB 7 | 6 | 5 | 4 | 3 | 2 | 1 | LSB 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | – | 0 | Stop timer 0 and clear it. |
| TMOD | ← | 1 | 1 | 1 | 0 | – | – | 0 | 1 | Set 8-bit PWM mode (cycle: $2^7 - 1$) and select $\phi$T1 as the input clock. |
| TREG0 | ← | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | Writes “48H”. |
| TFFCR | ← | X | X | X | X | 1 | 0 | 1 | X | Clear TFF1, enable the inversion and double buffer. |
| P7CR | ← | X | X | X | X | – | – | 1 | – | Set P71 as the TO1 pin. |
| P7FC | ← | X | X | X | X | – | – | 1 | X | |
| TRUN | ← | 1 | X | – | – | – | – | – | 1 | Start timer 0 counting. |

X: Don’t care, –: No change

Table 3.7.2 PWM Cycle

at fc = 20 MHz, fs = 32.768 kHz

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | PWM Cycle | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $2^6 - 1$ | | | $2^7 - 1$ | | | $2^8 - 1$ | | |
| | | | $\phi$T1 | $\phi$T4 | $\phi$T16 | $\phi$T1 | $\phi$T4 | $\phi$T16 | $\phi$T1 | $\phi$T4 | $\phi$T16 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | 15.4 ms | 61.5 ms | 246 ms | 31.0 ms | 124 ms | 496 ms | 62.3 ms | 249 ms | 996 ms |
| 0 (fc) | | 000 (fc) | 25.2 μs | 100.8 μs | 403.2 μs | 50.8 μs | 203.2 μs | 812.8 μs | 102.0 μs | 408.0 μs | 1.63 ms |
| | | 001 (fc/2) | 50.4 μs | 201.6 μs | 806.4 μs | 101.6 μs | 406.4 μs | 1.63 ms | 204.0 μs | 816.0 μs | 3.26 ms |
| | | 010 (fc/4) | 100.8 μs | 403.2 μs | 1.61 ms | 203.2 μs | 812.8 μs | 3.26 ms | 408.0 μs | 1.63 ms | 6.53 ms |
| | | 011 (fc/8) | 201.6 μs | 806.4 μs | 3.23 ms | 406.4 μs | 1.63 ms | 6.52 ms | 816.0 μs | 3.26 ms | 13.06 ms |
| | | 100 (fc/16) | 403.2 μs | 1.61 ms | 6.45 ms | 812.8 μs | 3.25 ms | 13.04 ms | 1.63 ms | 6.53 ms | 26.11 ms |
| XXX | 01 (Low-frequency clock) | XXX | 15.4 ms | 61.5 ms | 246 ms | 31.0 ms | 124 ms | 496 ms | 62.3 ms | 249 ms | 996 ms |
| XXX | 10 (fc/16 clock) | XXX | 403.2 μs | 1.61 ms | 6.45 ms | 812.8 μs | 3.25 ms | 13.04 ms | 1.63 ms | 6.53 ms | 26.11 ms |

XXX: Don’t care

(5) The list of 8-bit timer modes

Table 3.7.3 shows the list of 8-bit timer modes.

Table 3.7.3 Timer Mode Setting Registers

| Register Name | TMOD | | | | TFFCR |
|---|---|---|---|---|---|
| Bit Name | T10M | PWMM | T1CLK | T0CLK | TFF1IS |
| Function | Timer Mode | PWM Cycle | Upper Timer Input Clock | Lower Timer Input Clock | Timer F/F Invert Signal Select |
| 16-bit timer mode | 01 | – | – | External clock, $\phi$T1, $\phi$T4, $\phi$T16 (00, 01, 10, 11) | – |
| 8-bit timer × 2 channels | 00 | – | Lower timer match: $\phi$T1, 16, 256 (00, 01, 10, 11) | External clock, $\phi$T1, $\phi$T4, $\phi$T16 (00, 01, 10, 11) | 0: Lower timer output<br>1: Upper timer output |
| 8-bit PPG × 1 channel | 10 | – | – | External clock, $\phi$T1, $\phi$T4, $\phi$T16 (00, 01, 10, 11) | – |
| 8-bit PWM × 1 channel | 11 | $2^6 - 1$, $2^7 - 1$, $2^8 - 1$ (01, 10, 11) | – | External clock, $\phi$T1, $\phi$T4, $\phi$T16 (00, 01, 10, 11) | – |
| 8-bit timer × 1 channel | 11 | – | $\phi$T1, $\phi$T16, $\phi$T256 (01, 10, 11) | – | Output disabled |

–: Don’t care

## 3.8 8-Bit PWM Timers

The TMP93CS40 and TMP93CS41 have two built-in 8-bit PWM timers (Timers 2 and 3).

Each of these timers has two operating modes.

- 8-bit PWM output mode
- 8-bit interval timer mode

Figure 3.8.1, Figure 3.8.2 are block diagram of 8-bit PWM timers 0 and 1 (Timers 2 and 3).

PWM timers consist of an 8-bit up counter, 8-bit comparator, and 8-bit timer register. Two timer flip-flops (TFF2 for timer 2 and TFF3 for timer 3) are provided.

Input clocks $\phi$P1, $\phi$P4, and $\phi$P16 for the PWM timers can be obtained using the 5-bit built-in prescaler (PWM dedicated prescaler).

PWM timer operating mode and timer flip-flops are controlled by four control registers (P0MOD, P1MOD, PFFCR, and TRUN).

PWM timers 0 and 1 can be used independently.

All PWM timers operate in the same manner, and thus only the operation of PWM timer 0 will be explained below.

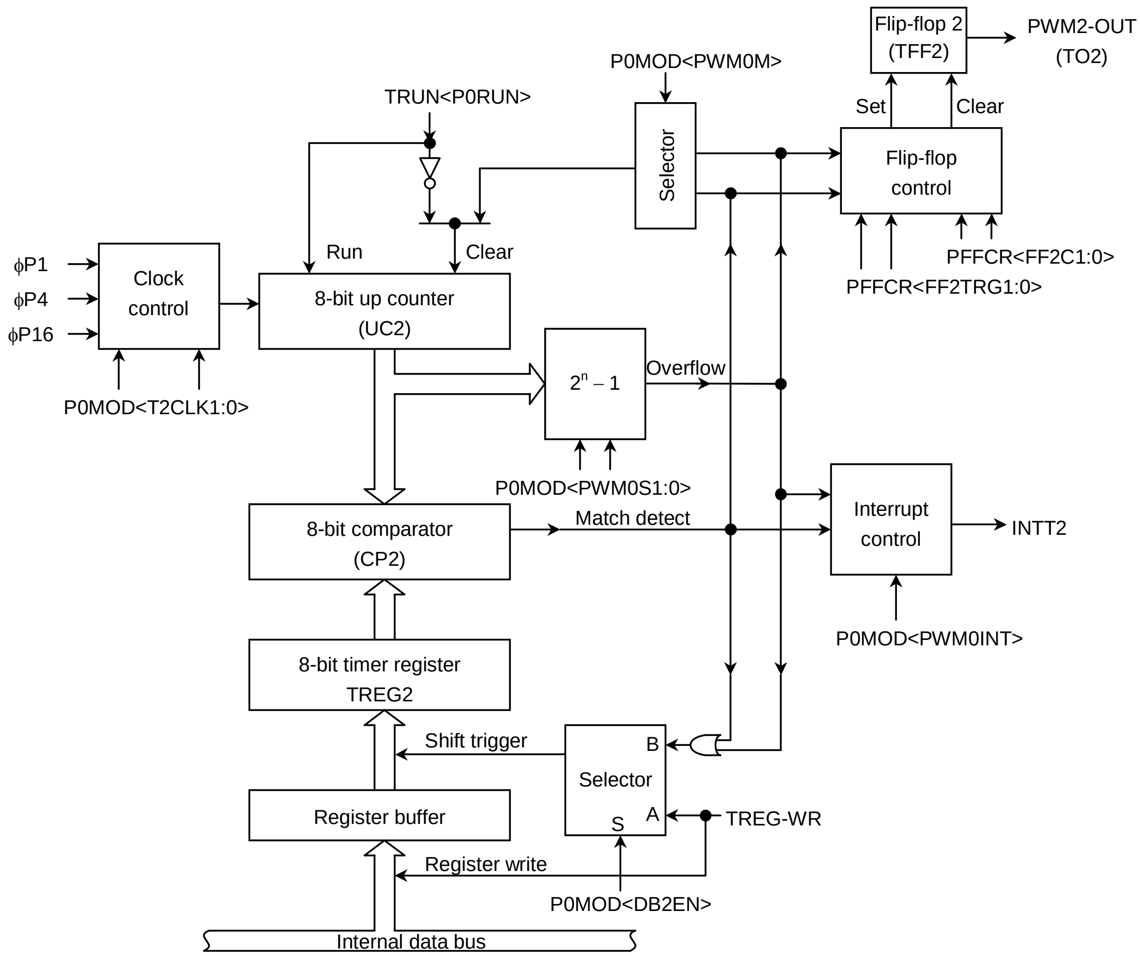



Figure 3.8.1 Block Diagram of 8-Bit PWM Timer 0 (Timer 2)

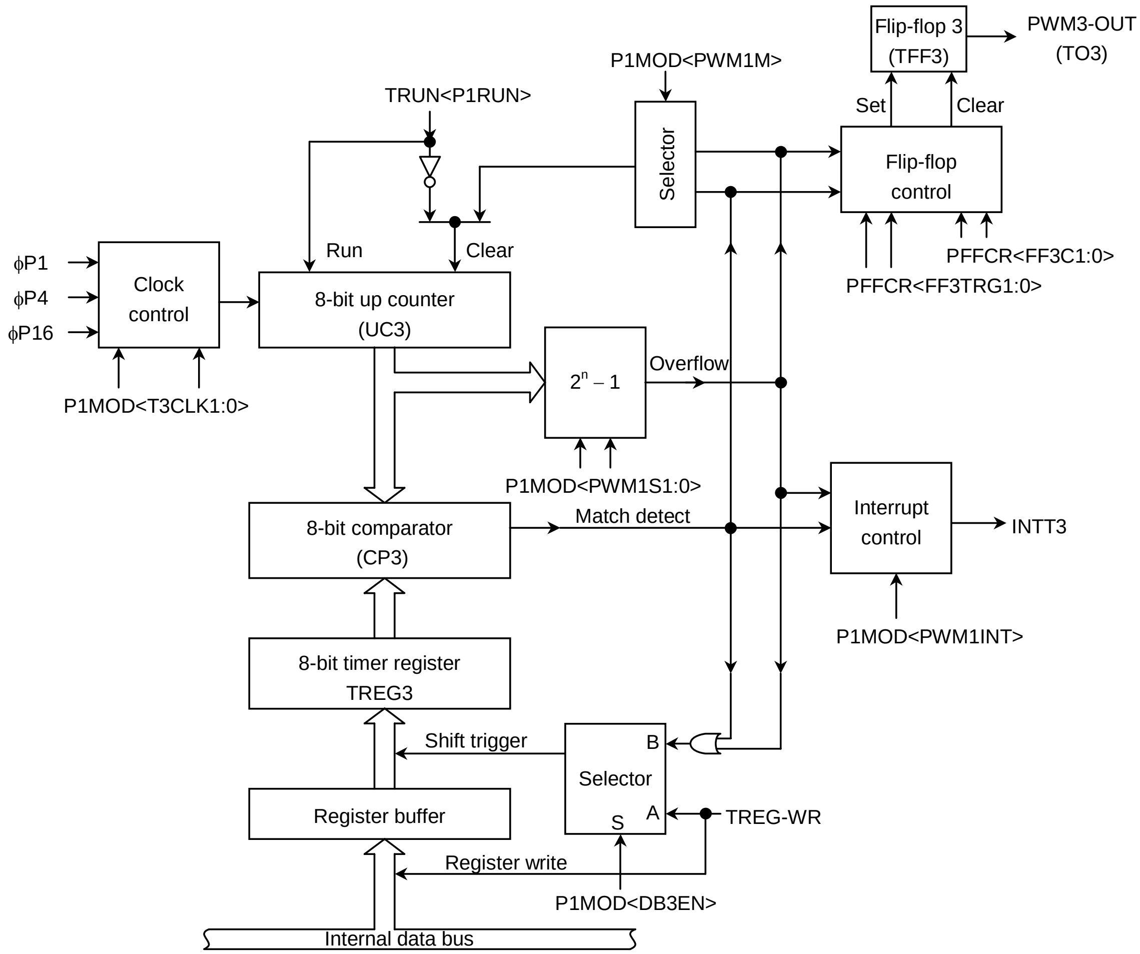



Figure 3.8.2 Block Diagram of 8-Bit PWM Timer 1 (Timer 3)

1. Prescaler

There are 5-bit prescaler and prescaler clock selection registers to generate clock inputs for 8-bit PWM timers 0 and 1.

Figure 3.8.3 shows the block diagram. Table 3.8.1 shows prescaler clock resolution into 8-bit PWM timers 0 and 1.

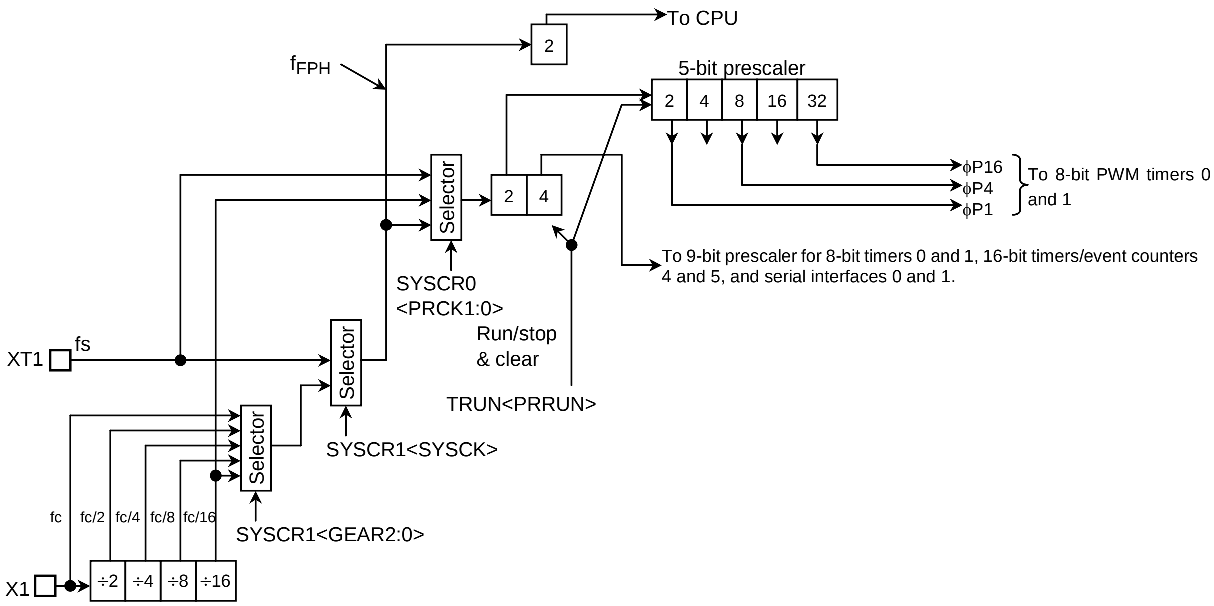



Figure 3.8.3 Block Diagram of the Prescaler

Table 3.8.1 Prescaler Clock Resolution to 8-Bit PWM Timers 0 and 1

at fc = 20 MHz, fs = 32.768 kHz

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | Prescaler Clock Resolution | | |
|---|---|---|---|---|---|
| | | | φP1 | φP4 | φP16 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | fs/$2^2$ (122 μs) | fs/$2^4$ (488 μs) | fs/$2^6$ (1.95 ms) |
| 0 (fc) | | 000 (fc) | fc/$2^2$ (0.2 μs) | fc/$2^4$ (0.8 μs) | fc/$2^6$ (3.2 μs) |
| | | 001 (fc/2) | fc/$2^3$ (0.4 μs) | fc/$2^5$ (1.6 μs) | fc/$2^7$ (6.4 μs) |
| | | 010 (fc/4) | fc/$2^4$ (0.8 μs) | fc/$2^6$ (3.2 μs) | fc/$2^8$ (12.8 μs) |
| | | 011 (fc/8) | fc/$2^5$ (1.6 μs) | fc/$2^7$ (6.4 μs) | fc/$2^9$ (25.6 μs) |
| | | 100 (fc/16) | fc/$2^6$ (3.2 μs) | fc/$2^8$ (12.8 μs) | fc/$2^{10}$ (51.2 μs) |
| XXX | 01 (Low-frequency clock) | XXX | fs/$2^2$ (122 μs) | fs/$2^4$ (488 μs) | fs/$2^6$ (1.95 ms) |
| XXX | 10 (fc/16 clock) | XXX | fc/$2^6$ (3.2 μs) | fc/$2^8$ (12.8 μs) | fc/$2^{10}$ (51.2 μs) |

XXX: Don't care

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

The clock selected among $f_{FPH}$, fc/16, and fs is divided by 2 and input to this prescaler. The selection is made by the system clock control register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to 00, selecting the $f_{FPH}$ clock input to be divided by 2. The TRUN<PRRUN> register which controls this prescaler, is also used for the other timers. So, this prescaler cannot be operated independently.

The 8-bit PWM timers 0 and 1 select one of the clock inputs: φP1, φP4, and φP16, from among the three prescaler outputs.

This prescaler also can be run or stopped by TRUN<PRRUN> as described in discussion of the 8-bit timer.

Counting starts when <PRRUN> is set to 1. The prescaler is cleared and stops operation when <PRRUN> is set to 0.

Resetting clears <PRRUN> and stops the prescaler.

When the IDLE1 mode (with only the oscillator operating) is used, set TRUN<PRRUN> to "0" to reduce the power consumption of the prescaler before the "HALT" instruction is executed.

2. Up counter

The up counter is an 8-bit binary counter which counts up using the input clock specified by PWM0 mode register P0MOD<T2CLK1:0>.

The input clock for the up counter is selected from among the internal clocks φP1, φP4, and φP16 (PWM dedicated prescaler output) depending on the value in <T2CLK1:0>.

Operating mode is set by P0MOD<PWM0M>. At reset, <PWM0M> is initialized to "0", and thus the up counter is placed in PWM mode. In PWM mode, the up counter is cleared when a $2^n - 1$ overflow occurs; in timer mode, the up counter is cleared at compare and match.

Count/stop and clear of the up counter can be controlled for each PWM timer using the timer operation control register TRUN. Resetting clears all up counters and stops all PWM timers.

3. Timer register

The 8-bit register is used for setting a time interval. When the value set in the timer register (TREG2) matches the value in the up counter, the match detect signal of the comparator becomes active.

Timer register TREG2 is paired with a register buffer to make a double buffer structure.

TREG2 controls double buffer enable/disable by P0MOD<DB2EN>. The double buffer is disabled when <DB2EN> = 0, and enabled when <DB2EN> = 1.

In double buffer enable state, data are transferred from register buffer to timer register when a $2^n - 1$ overflow occurs in PWM mode, or when compare and match occur in 8-bit timer mode. A PWM timer can be operated in timer mode in double buffer enable state, whereas timers 0 and 1 cannot.

At reset, <DB2EN> is initialized to 0 to disable the double buffer. The same value is set in both the register buffer and the timer register. To use the double buffer, write the data in the timer register first, then set <DB2EN> to 1 and write the following data in the register buffer.

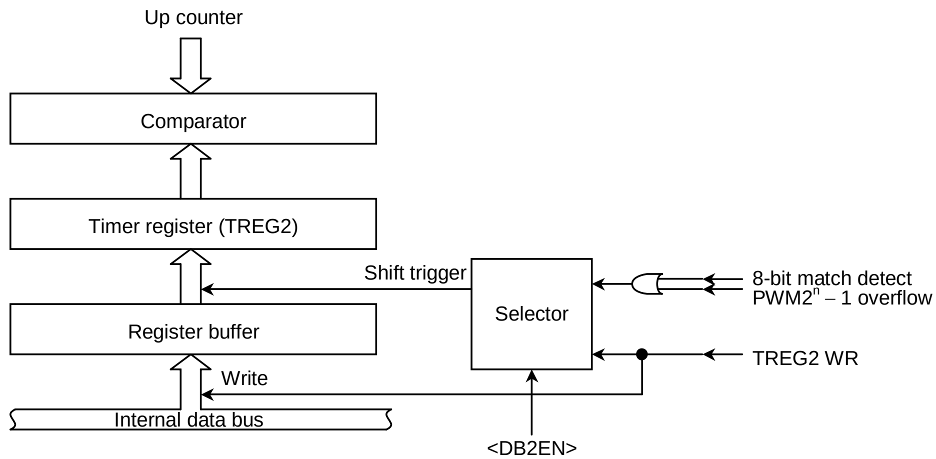



Figure 3.8.4 Structure of Timer Registers 2

Memory addresses of the timer registers are as follows:

TREG2: 000026H (PWM timer 0)
TREG3: 000027H (PWM timer 1)

The timer register and register buffer are allocated to the same memory address. When <DB2EN> = 0, the same value is written to both the register buffer and timer register. When <DB2EN> = 1, a value is written to the register buffer only.

Register buffer values can be read when reading the above addresses. The timer register is write-only and it cannot be read.

4. Comparator

This element compares the value in the up counter with the value in the timer register (TREG2). When they match, the comparator outputs the match detect signal. A timer interrupt (INTT2) is generated at compare and match if the interrupt select bit <PWM0INT> of the mode register (P0MOD) is set to 1. In timer mode, the comparator clears the up counter at compare and match. It also inverts the value of the timer flip-flop if timer flip-flop invert is enabled.

5. Timer flip-flop

The value of the timer flip-flop is inverted by the match detect signal (comparator output) of each interval timer or by $2^n - 1$ overflow. The flip-flop value can be output to the timer output pin TO2 (also used as P72).

P0MOD (0028H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | FF2RD | DB2EN | PWM0INT | PWM0M | T2CLK1 | T2CLK0 | PWM0S1 | PWM0S0 |
| Read/Write | R | W | | | | | | |
| After reset | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Flip-flop (Flip-flop 2) output data | 1: Double buffer 2 enable | 0: $2^n - 1$ overflow interrupt<br>1: Compare and match interrupt | 0: PWM mode<br>1: Timer mode | 00: $\phi$P1<br>01: $\phi$P4<br>10: $\phi$P16<br>11: Don't care | | 00: $2^6 - 1$<br>01: $2^7 - 1$<br>10: $2^8 - 1$<br>11: Don't care | |

Read-modify-write is prohibited.

Select PWM0 cycle

| | |
|---|---|
| 00 | $2^6 - 1$ |
| 01 | $2^7 - 1$ |
| 10 | $2^8 - 1$ |
| 11 | Don't care |

Select PWM0 input clock

| | |
|---|---|
| 00 | $\phi$P1 |
| 01 | $\phi$P4 |
| 10 | $\phi$P16 |
| 11 | Don't care |

Select PWM0 mode

| | |
|---|---|
| 0 | PWM mode |
| 1 | 8-bit timer mode |

Select PWM0 interrupt

| | |
|---|---|
| 0 | Overflow interrupt |
| 1 | Compare and match interrupt |

Control double buffer

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

PWM timer flip-flop 2 (TFF2) output value (TO2)

Figure 3.8.5 8-bit PWM0 Mode Control Register

P1MOD (0029H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | FF3RD | DB3EN | PWM1INT | PWM1M | T3CLK1 | T3CLK0 | PWM1S1 | PWM1S0 |
| Read/Write | R | W | | | | | | |
| After reset | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Flip-flop (Flip-flop 3) output data | 1: Double buffer 3 enable | 0: $2^n - 1$ overflow interrupt<br>1: Compare and match interrupt | 0: PWM mode<br>1: Timer mode | 00: $\phi$P1<br>01: $\phi$P4<br>10: $\phi$P16<br>11: Don't care | | 00: $2^6 - 1$<br>01: $2^7 - 1$<br>10: $2^8 - 1$<br>11: Don't care | |

Read-modify-write is prohibited.

Select PWM1 cycle

| | |
|---|---|
| 00 | $2^6 - 1$ |
| 01 | $2^7 - 1$ |
| 10 | $2^8 - 1$ |
| 11 | Don't care |

Select PWM1 input clock

| | |
|---|---|
| 00 | $\phi$P1 |
| 01 | $\phi$P4 |
| 10 | $\phi$P16 |
| 11 | Don't care |

Select PWM1 mode

| | |
|---|---|
| 0 | PWM mode |
| 1 | 8-bit timer mode |

Select PWM1 interrupt

| | |
|---|---|
| 0 | Overflow interrupt |
| 1 | Compare and match interrupt |

Control double buffer

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

PWM timer flip-flop 3 (TFF3) output value (TO3)

Figure 3.8.6 8-Bit PWM 1 Mode Control Register

| PFFCR (002AH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | FF3C1 | FF3C0 | FF3TRG1 | FF3TRG0 | FF2C1 | FF2C0 | FF2TRG1 | FF2TRG0 |
| | Read/Write | W | | R/W | | W | | R/W | |
| | After reset | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| | Function | 00: Don’t care<br>01: Set TFF3<br>10: Clear TFF3<br>11: Don’t care | | 00: Disable TFF3 inverted.<br>01: Invert by match.<br>10: Set by match; clear by overflow<br>11: Clear by match; set by overflow. | | 00: Don’t care<br>01: Set TFF2<br>10: Clear TFF2<br>11: Don’t care | | 00: Disable TFF2 inverted.<br>01: Invert by match.<br>10: Set by match; clear by overflow<br>11: Clear by match; set by overflow. | |

Select PWM timer flip-flop 2 (TFF2) trigger

| | |
|---|---|
| 00 | Disable TFF2 trigger. |
| 01 | Invert by compare and match. |
| 10 | Set by compare and match.<br>Clear by $2^n - 1$ overflow. |
| 11 | Clear by compare and match.<br>Set by $2^n - 1$ overflow. |

Control PWM timer flip-flop 2 (TFF2)

| | |
|---|---|
| 00 | Don’t care |
| 01 | Set TFF2 to “1”. |
| 10 | Clear TFF2 to “0”. |
| 11 | Don’t care |

Select PWM timer flip-flop 3 (TFF3) trigger

| | |
|---|---|
| 00 | Disable TFF3 trigger. |
| 01 | Invert by compare and match. |
| 10 | Set by compare and match.<br>Clear by $2^n - 1$ overflow. |
| 11 | Clear by compare and match.<br>Set by $2^n - 1$ overflow. |

Control PWM timer flip-flop 3 (TFF3)

| | |
|---|---|
| 00 | Don’t care |
| 01 | Set TFF3 to “1”. |
| 10 | Clear TFF3. |
| 11 | Don’t care |

Figure 3.8.7 8-Bit PWM Flip-flop Control Register

TRUN (0020H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PRRUN | | T5RUN | T4RUN | P1RUN | P0RUN | T1RUN | T0RUN |
| Read/Write | R/W | | R/W | | | | | |
| After reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Prescaler and timer run/stop control<br>0: Stop and clear<br>1: Run (Count up) | | | | | | | |

Count operation

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

PRRUN: Operation of prescaler
T5RUN: Operation of 16-bit timer (Timer 5)
T4RUN: Operation of 16-bit timer (Timer 4)
P1RUN: Operation of PWM timer 1
P0RUN: Operation of PWM timer 0
T1RUN: Operation of 8-bit timer (Timer 1)
T0RUN: Operation of 8-bit timer (Timer 0)

Note: TRUN<bit6> is read as "1".

SYSCR0 (006EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | XEN | XTEN | RXEN | RXTEN | RSYSCK | WUEF | PRCK1 | PRCK0 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | High-frequency oscillator (fc)<br>0: Stop<br>1: Oscillator operates | Low-frequency oscillator (fc)<br>0: Stop<br>1: Oscillator operates | High-frequency oscillator (fc) after release of STOP mode<br>0: Stop<br>1: Oscillator operates | Low-frequency oscillator (fc) after release of STOP mode<br>0: Stop<br>1: Oscillator operates | Select clock after release of STOP mode<br>0: fc<br>1: fs | Warm-up timer<br>(Write)<br>0: Don't care<br>1: Start timer<br>(Read)<br>0: End warm up<br>1: Continue warm up | Select prescaler clock<br>00: $f_{FPH}$<br>01: fs<br>10: fc/16<br>11: (Reserved) | |

Select gear value of high frequency

| | |
|---|---|
| 00 | $f_{FPH}$ |
| 01 | fs |
| 10 | fc/16 |
| 11 | (Reserved) |

Clock divided by 2

Figure 3.8.8 Timer Operation Control Register/System Clock Control Register

The following explains PWM timer operations.

(1) PWM timer mode

PWM output changes under the following two conditions.

Condition 1:

- TFF2 is cleared when the value in the up counter (UC2) matches the value set in the TREG2.
- TFF2 is set to 1 when a $2^n - 1$ counter overflow (n = 6, 7, or 8) occurs.

Condition 2:

- TFF2 is set to 1 when the value in the up counter (UC2) matches the value set in TREG2.
- TFF2 is cleared when a $2^n - 1$ counter overflow (n = 6, 7, or 8) occurs.

The up counter (UC2) is cleared by a $2^n - 1$ counter overflow.

The PWM timer can output 0% to 100% duty pulses because a $2^n - 1$ counter overflow has a higher priority. That is, to obtain 0% output (Always low), the mode used to set TFF2 to 0 due to overflow (PFFCR<FF2TRG1:0> = 10) must be set and $2^n - 1$ (A value indicating an overflow) must be set in TREG2. To obtain 100% output (Always high), the mode must be changed by setting PFFCR<FF2TRG1:0> = 11 and then TREG2 must be set to $2^n - 1$.

PWM timing

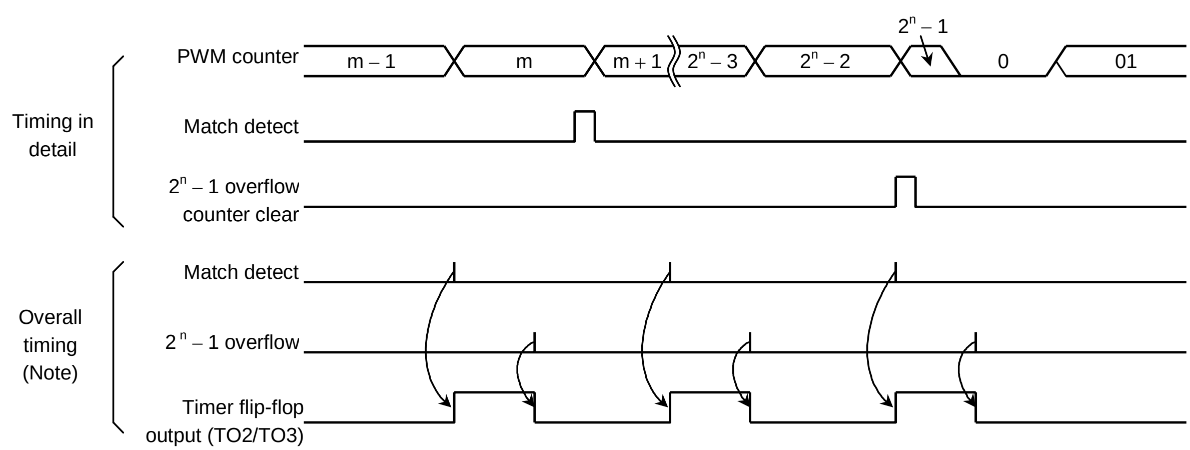



Figure 3.8.9 Output Waves in PWM Timer Mode

Note: The waves pictured above are obtained in the mode in which the flip-flop is set by a match with the timer register (TREG), and is reset by an overflow.

Figure 3.8.10 is a block diagram of this mode.

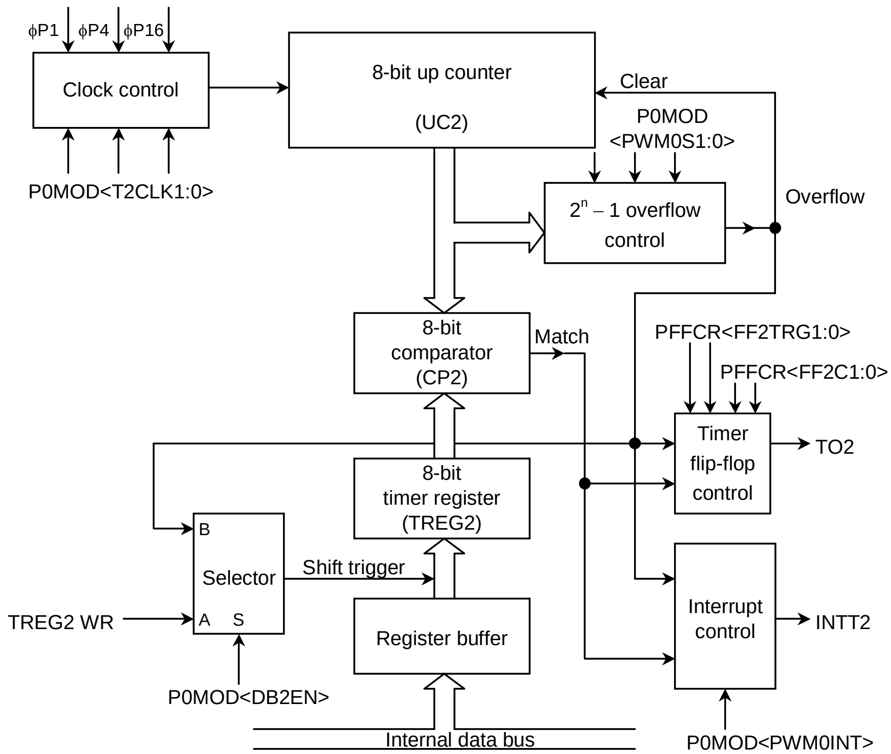



Figure 3.8.10 Block Diagram of PWM Timer Mode (PWM0)

In this mode, enabling double buffer is very useful. The register buffer value shifts into TREG2 when a $2^n - 1$ overflow is detected, when double buffer is enabled.

Use of the double buffer makes the handling of low duty waves easy.

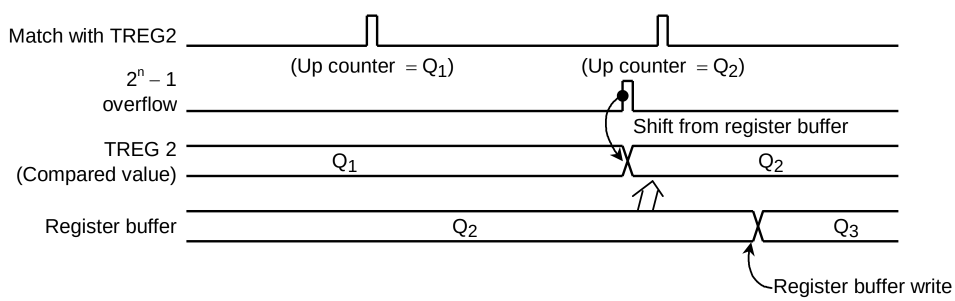
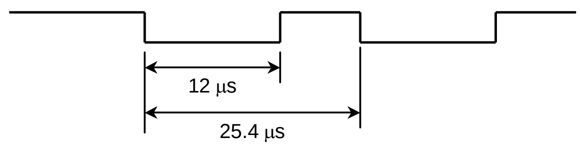



Figure 3.8.11 Register Buffer Operation

Example: To output the following PWM waves to the TO2 pin using PWM0 at
fc = 20 MHz

Clock condition
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: $f_{FPH}$




To implement 25.4 μs as the PWM cycle regulated by ϕP1 = 0.2 μs (at fc = 20 MHz)

$2^n - 1 = 25.4\ \mu s \div 0.2\ \mu s = 127.$

Consequently, set n to 7.

Since the low level cycle = 12 μs; for ϕP1 = 0.2 μs

3CH = 12 μs ÷ 0.2 = 60

set the 3CH in TREG2.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | 0 | – | – | Stops PWM0 and clears it. |
| P0MOD | ← | – | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Sets PWM ($2^7$ – 1) mode, input clock ϕP1, overflow interrupt, and disables double buffer. |
| TREG2 | ← | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | Writes 3CH. |
| P0MOD | ← | – | 1 | 0 | 0 | 0 | 0 | 0 | 1 | Enables double buffer. |
| PFFCR | ← | – | – | – | – | 0 | 1 | 1 | 0 | Sets TFF2 and the mode in which TFF2 is set by compare and match, and cleared by overflow. |
| P7CR | ← | X | X | X | X | – | 1 | – | – | Sets P72 as TO2 pin. |
| P7FC | ← | X | X | X | X | – | 1 | – | X | |
| TRUN | ← | 1 | X | – | – | – | 1 | – | – | Starts PWM0 counting. |

X: Don't care, –: No change

Table 3.8.2 PWM Cycle

at fc = 20 MHz, fs = 32.768 kHz

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | PWM cycle | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $2^6 - 1$ | | | $2^7 - 1$ | | | $2^8 - 1$ | | |
| | | | φP1 | φP4 | φP16 | φP1 | φP4 | φP16 | φP1 | φP4 | φP16 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | 7.69 ms | 30.8 ms | 123 ms | 15.5 ms | 62.0 ms | 248 ms | 31.1 ms | 125 ms | 498 ms |
| 0 (fc) | | 000 (fc) | 12.6 μs | 50.4 μs | 201.6 μs | 25.4 μs | 101.6 μs | 406.4 μs | 51.0 μs | 204.0 μs | 816.0 μs |
| | | 001 (fc/2) | 25.2 μs | 100.8 μs | 403.2 μs | 50.8 μs | 203.2 μs | 812.8 μs | 102.0 μs | 408.0 μs | 1.63 ms |
| | | 010 (fc/4) | 50.4 μs | 201.6 μs | 806.4 μs | 101.6 μs | 406.4 μs | 1.63 ms | 204.0 μs | 816.0 μs | 3.26 ms |
| | | 011 (fc/8) | 100.8 μs | 403.2 μs | 1.61 ms | 203.2 μs | 812.8 μs | 3.25 ms | 408.0 μs | 1.63 ms | 6.53 ms |
| | | 100 (fc/16) | 201.6 μs | 806.4 μs | 3.23 ms | 406.4 μs | 1.63 ms | 6.50 ms | 816.0 μs | 3.26 ms | 13.06 ms |
| XXX | 01 (Low frequency) | XXX | 7.69 ms | 30.8 ms | 123 ms | 15.5 ms | 62.0 ms | 248 ms | 31.1 ms | 125 ms | 498 ms |
| XXX | 10 (fc/16 clock) | XXX | 201.6 μs | 806.4 μs | 3.23 ms | 406.4 μs | 1.63 ms | 6.50 ms | 816.0 μs | 3.26 ms | 13.06 ms |

XXX: Don't care

(2) 8-bit timer mode

Both PWM timers can be used independently as 8-bit interval timers. Since both timers operate in exactly the same way, PWM0 (timer 2) is used for the purposes of explanation.

1. Generating interrupts at a fixed interval

To generate timer 2 interrupt (INTT2) at a fixed interval using the PWM0 timer, first stop PWM0, then set the operating mode, input clock, and interval in the P0MOD and TREG2 registers. Next, enable INTT2 and start counting PWM0.

Example: To generate a timer 2 interrupt every 32 μs at fc = 20 MHz, set registers as follows:

Clock condition
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: $f_{FPH}$

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | 0 | – | – | Stops PWM timer 0 and clears it. |
| P0MOD | ← | X | 0 | 1 | 1 | 0 | 0 | X | X | Sets 8-bit timer mode and selects φP1 (0.2 μs) and compare interrupt. |
| TREG2 | ← | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Sets A0H (= 32 μs ÷ 0.2 μs) in the timer register. |
| INTEPW10 | ← | – | – | – | – | 1 | 1 | 0 | 0 | Enables INTT2 and sets interrupt level 4. |
| TRUN | ← | 1 | X | – | – | – | 1 | – | – | Starts counting PWM0. |

X: Don't care, –: No change

Select an input clock using Table 3.8.1.

Note: To generate interrupts in 8-bit timer mode, bit5 (Interrupt control bit <PWM0INT> of P0MOD) must be set to 1.

2. Generating a 50% square wave

To generate a 50% square wave, invert the timer flip-flop at a fixed interval and output the timer flip-flop value to the timer output pin (TO2).

Example: To output a 2.4 μs square wave at fc = 20 MHz from the TO2 pin, set the registers as follows.

Clock condition
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: $f_{FPH}$

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | 0 | – | – | Stops PWM0 and clears it. |
| P0MOD | ← | X | 0 | 1 | 1 | 0 | 0 | X | X | Sets 8-bit timer mode and selects φP1 (0.2 μs) as the input clock. |
| TREG2 | ← | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | Sets 2.4 μs ÷ 0.2 μs ÷ 2 = 6 in the timer register. |
| PFFCR | ← | – | – | – | – | 1 | 0 | 0 | 1 | Clears TFF2 and inverts using comparator output. |
| P7CR | ← | X | X | X | X | – | 1 | – | – | Sets P72 as the TO2 pin. |
| P7FC | ← | X | X | X | X | – | 1 | – | X | |
| TRUN | ← | 1 | X | – | – | – | 1 | – | – | Starts counting PWM0. |

X: Don't care, –: No change

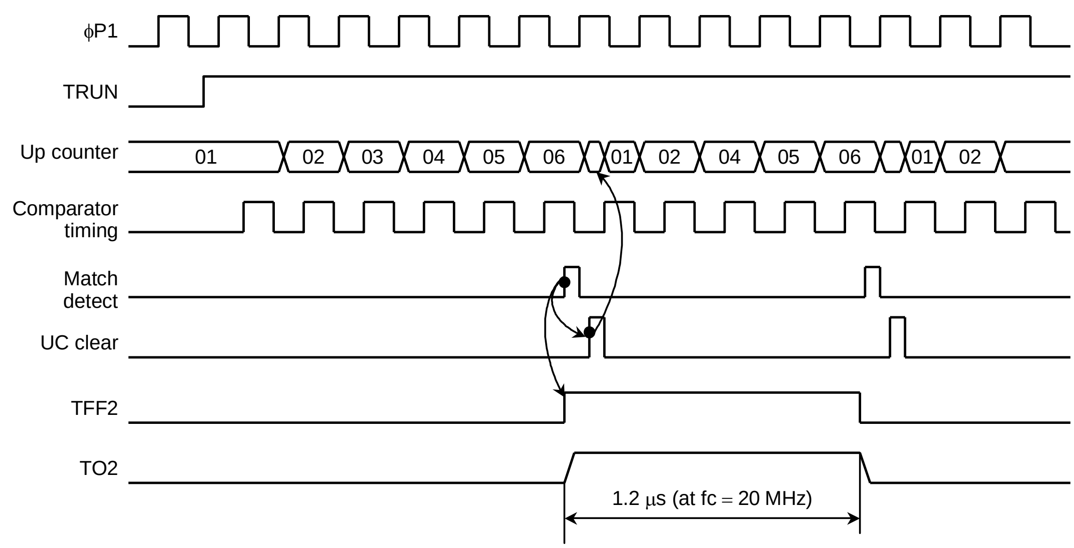



Figure 3.8.12 Square Wave (50% duty) Output Timing Chart

This mode is as shown in Figure 3.8.13 below.

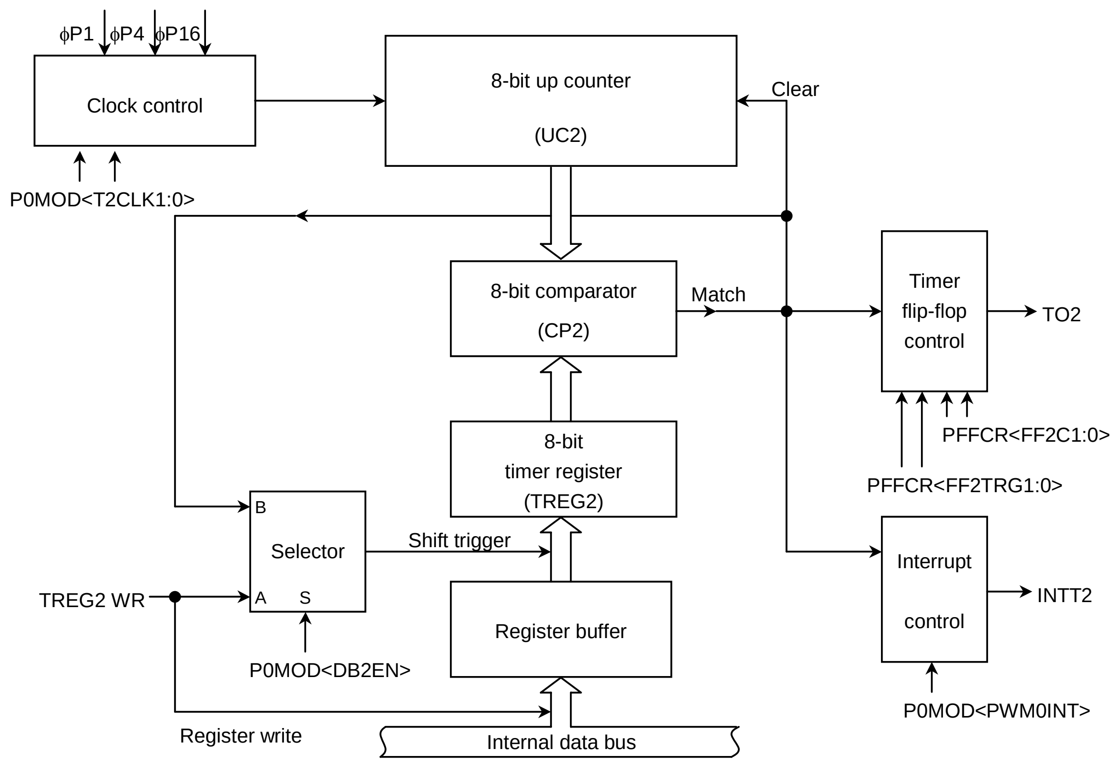



Figure 3.8.13 Block Diagram of 8-Bit Timer Mode

## 3.9 16-Bit Timers

The TMP93CS40 and TMP93CS41 contain two multifunctional 16-bit timer/event counters (Timer 4 and timer 5) which support the following operation modes.

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation (PPG) mode
- Frequency measurement mode
- Pulse width measurement mode
- Time differential measurement mode

Each timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (One with a double-buffer), two 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and the control circuit.

Timer/event counters are controlled by 4 control registers: T4MOD/T5MOD, T4FFCR/T5FFCR, TRUN and T45CR.

Figure 3.9.1, Figure 3.9.2 show the block diagram of the 16-bit timer/event counters (Timer 4 and timer 5).

Timers 4 and 5 can be used independently.

All timers operate in the same manner except for the following points, and thus only the operation of timer 4 will be explained below.

Points Differing between Timers 4 and 5

| | 16-Bit Timer 4 | 16-Bit Timer 5 |
|---|---|---|
| Timer out pin | TO4 pin (TFF4)<br>TO5 pin (TFF5) | TO6 pin (TFF6) |
| Different phased pulse output mode | Yes | No<br>(No TO7 pin) |

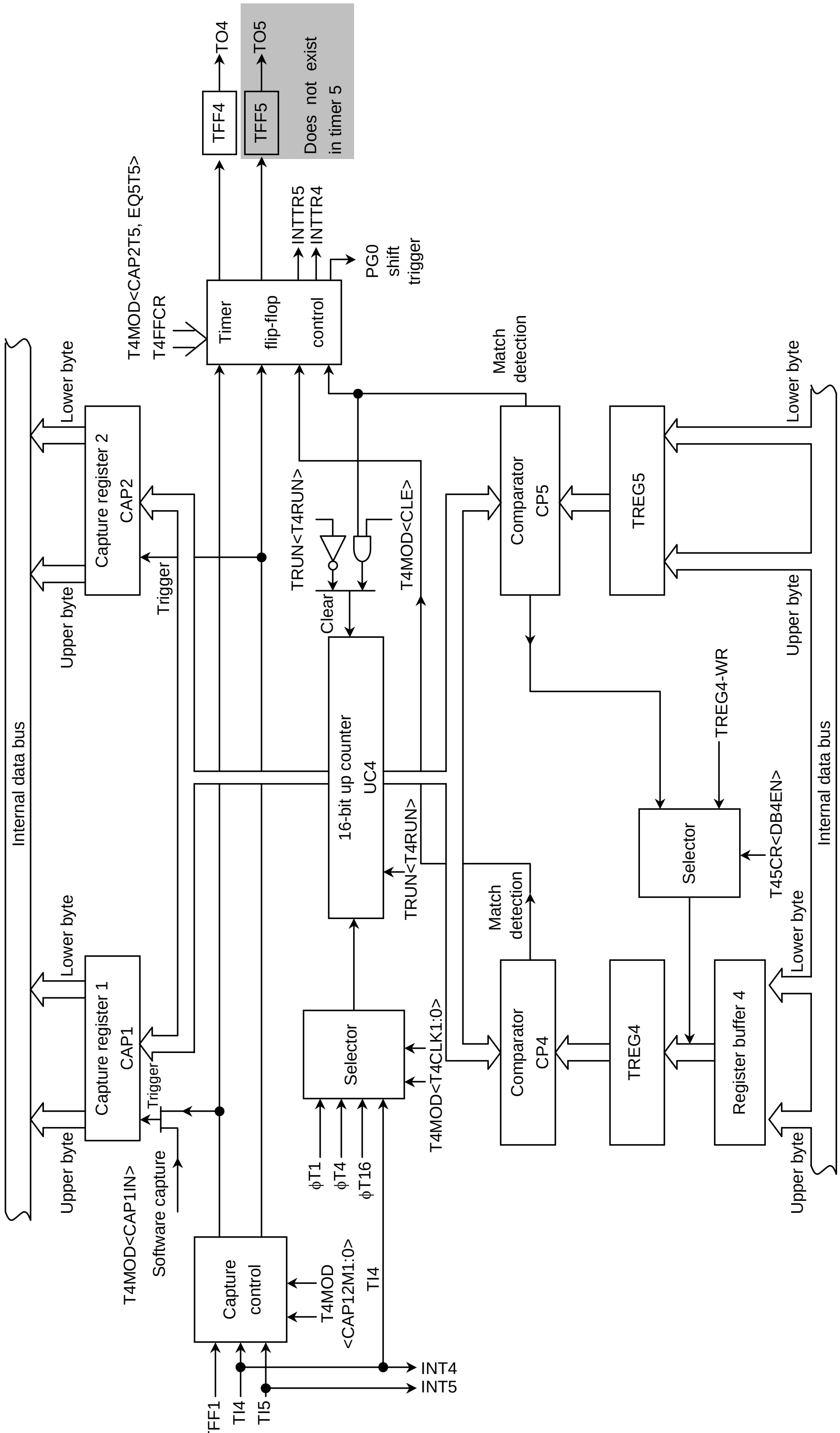



Figure 3.9.1 Block Diagram of 16-Bit Timer (Timer 4)

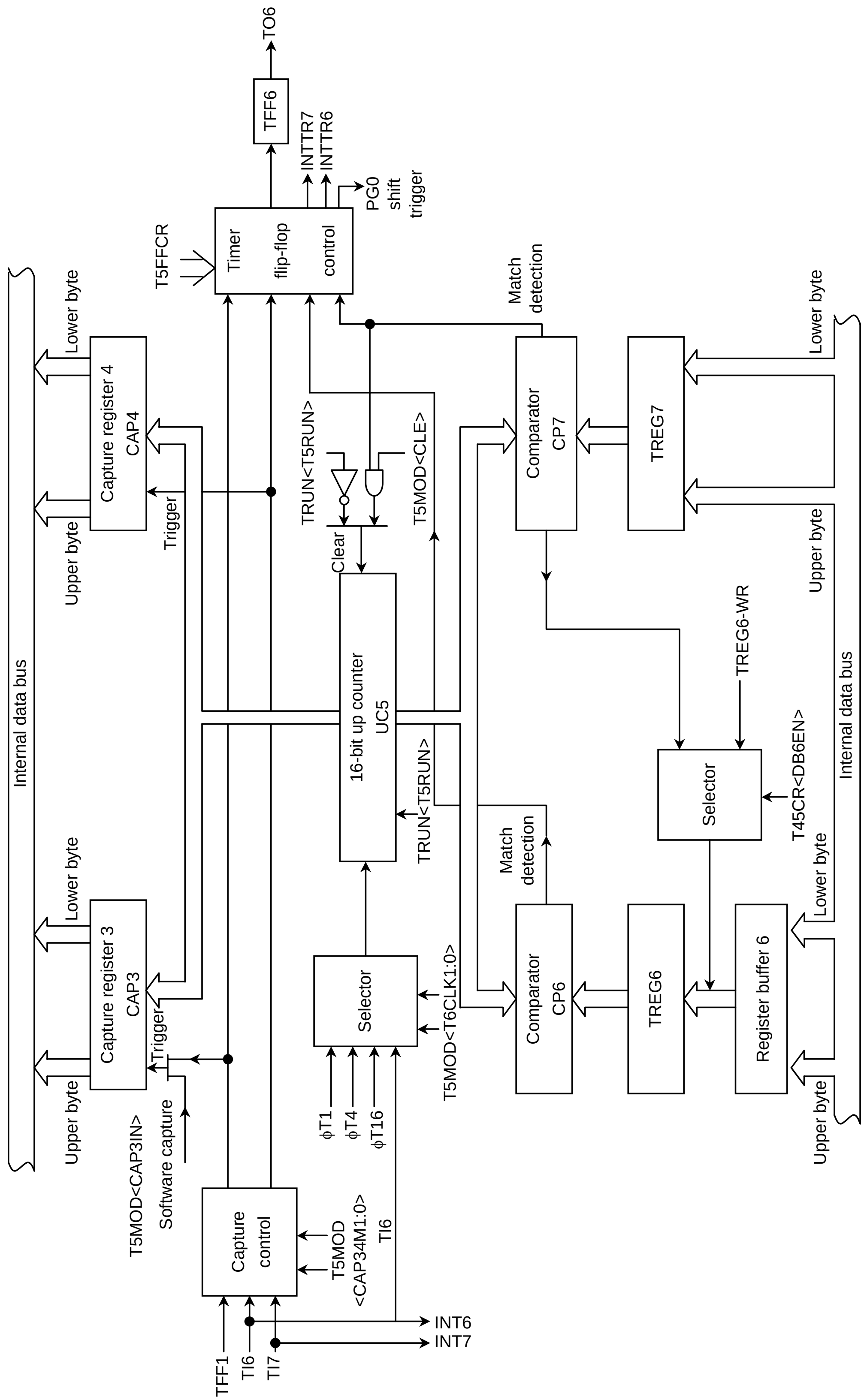



Figure 3.9.2 Block Diagram of 16-Bit Timer (Timer 5)

T4MOD (0038H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | CAP2T5 | EQ5T5 | CAP1IN | CAP12M1 | CAP12M0 | CLE | T4CLK1 | T4CLK0 |
| Read/Write | R/W | | W | R/W | | R/W | R/W | |
| After reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | TFF5 invert trigger<br>0: Disable trigger<br>1: Enable trigger | | 0: Software capture<br>1: Don’t care | Capture timing<br>00: Disable<br>INT4 occurs at rising edge.<br>01: TI4 $\uparrow$ TI5 $\uparrow$<br>INT4 occurs at rising edge.<br>10: TI4 $\uparrow$ TI4 $\downarrow$<br>INT4 occurs at falling edge.<br>11: TFF1 $\uparrow$ TFF1 $\downarrow$<br>INT4 occurs at rising edge. | | 1: UC4 clear enable | Timer 4 source clock<br>00: TI4<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |
| | Inverted when the UC value is latched to CAP2 | Inverted when the up counter matches TREG5 | | | | | | |

Timer 4 input clock

| | |
|---|---|
| 00 | External clock (TI4) |
| 01 | $\phi$T1 |
| 10 | $\phi$T4 |
| 11 | $\phi$T16 |

Clearing the up counter UC4

| | |
|---|---|
| 0 | Clear disable |
| 1 | Clear by match with TREG5 |

Figure 3.9.3 16-Bit Timer Mode Control Register (T4MOD) (1/2)

T4MOD (0038H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | CAP2T5 | EQ5T5 | CAP1IN | CAP12M1 | CAP12M0 | CLE | T4CLK1 | T4CLK0 |
| Read/Write | R/W | | W | R/W | | R/W | R/W | |
| After reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | TFF5 invert trigger<br>0: Disable trigger<br>1: Enable trigger | | 0: Software capture<br>1: Don't care | Capture timing<br>00: Disable<br>INT4 occurs at rising edge.<br>01: TI4 ↑ TI5 ↑<br>INT4 occurs at rising edge.<br>10: TI4 ↑ TI4 ↓<br>INT4 occurs at falling edge.<br>11: TFF1 ↑ TFF1 ↓<br>INT4 occurs at rising edge. | | 1: UC4 clear enable | Timer 4 source clock<br>00: TI4<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |
| | Inverted when the UC value is latched to CAP2 | Inverted when the up counter matches TREG5 | | | | | | |

Capture timing of timer4

| | Capture control | INT4 control |
|---|---|---|
| 00 | Capture disable | Interrupt occurs at the rising edge of TI4 (INT4) input. |
| 01 | CAP1 at TI4 rise<br>CAP2 at TI5 rise | |
| 10 | CAP1 at TI4 rise<br>CAP2 at TI4 fall | Interrupt occurs at the falling edge of TI4 (INT4) input. |
| 11 | CAP1 at TFF1 rise<br>CAP2 at TFF1 fall | Interrupt occurs at the rising edge of TI4 (INT4) input. |

Software capture

| | |
|---|---|
| 0 | The up counter 4 value is loaded to CAP1 (Software capture) |
| 1 | Always read as "1" |

Timer flip-flop 5 (TFF5) invert trigger

| | |
|---|---|
| 0 | Trigger disable (Invert prohibition) |
| 1 | Trigger enable (Invert permission) |

CAP2T5: Inverted when the up counter value is latched to CAP2
EQ5T5: Inverted when the up counter matches TREG5

Figure 3.9.4 16-Bit Timer Controller Register (T4MOD) (2/2)

T4FFCR (0039H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TFF5C1 | TFF5C0 | CAP2T4 | CAP1T4 | EQ5T4 | EQ4T4 | TFF4C1 | TFF4C0 |
| Read/Write | W | | R/W | R/W | R/W | R/W | W | |
| After reset | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| Function | 00: Invert TFF5<br>01: Set TFF5<br>10: Clear TFF5<br>11: Don't care<br>* Always read as "11" | | TFF4 invert trigger<br>0: Disable trigger<br>1: Enable trigger<br>Inverted when the UC value is latched to CAP2 | Inverted when the UC value is latched to CAP1 | Inverted when the UC value matches TREG5 | Inverted when the UC value matches TREG4 | 00: Invert TFF4<br>01: Set TFF4<br>10: Clear TFF4<br>11: Don't care<br>* Always read as "11" | |

Timer flip-flop 4 (TFF4) control

| | |
|---|---|
| 00 | Inverts the TFF4 value (Software inversion) |
| 01 | Sets TFF4 to "1" |
| 10 | Clear TFF4 to "0" |
| 11 | Don't care (Always read as "11") |

Timer flip-flop 4 (TFF4) invert trigger

| | |
|---|---|
| 0 | Trigger disable (Invert prohibition) |
| 1 | Trigger enable (Invert permission) |

CAP2T4: Inverted when the up counter value is latched to CAP2
CAP1T4: Inverted when the up counter value is latched to CAP1
EQ5T4: Inverted when up counter matches TREG5
EQ4T4: Inverted when up counter matches TREG4

Timer flip-flop 5 (TFF5) control

| | |
|---|---|
| 00 | Inverts the TFF5 value (Software inversion) |
| 01 | Set TFF5 to "1" |
| 10 | Clear TFF5 to "0" |
| 11 | Don't care (Always read as "11") |

Figure 3.9.5 16-Bit Timer 4 F/F Control (T4FFCR)

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| T5MOD (0048H) | Bit symbol | | | CAP3IN | CAP34M1 | CAP34M0 | CLE | T5CLK1 | T5CLK0 |
| | Read/Write | | | W | R/W | | R/W | R/W | |
| | After reset | | | 1 | 0 | 0 | 0 | 0 | 0 |
| | Function | | | 0: Software capture<br>1: Don't care | Capture timing<br>00: Disable<br>INT6 occurs at rising edge<br>01: TI6 ↑ TI7 ↑<br>INT6 occurs at rising edge.<br>10: TI6 ↑ TI6 ↓<br>INT6 occurs at falling edge.<br>11: TFF1 ↑ TFF1 ↓<br>INT6 occurs at rising edge. | | 1: UC5 clear enable | Timer 5 source clock<br>00: TI6<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |

Timer 5 input clock

| | |
|---|---|
| 00 | External clock (TI6) |
| 01 | $\phi$T1 |
| 10 | $\phi$T4 |
| 11 | $\phi$T16 |

Clearing the up counter UC5

| | |
|---|---|
| 0 | Clear disable |
| 1 | Clear by match with TREG7 |

Figure 3.9.6 16-Bit Timer Mode Control Register (T5MOD) (1/2)

| T5MOD (0048H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | | | CAP3IN | CAP34M1 | CAP34M0 | CLE | T5CLK1 | T5CLK0 |
| | Read/Write | | | W | R/W | | R/W | R/W | |
| | After reset | | | 1 | 0 | 0 | 0 | 0 | 0 |
| | Function | | | 0: Software capture<br>1: Don't care | Capture timing<br>00: Disable<br>INT6 occurs at rising edge.<br>01: TI6↑ TI7↑<br>INT6 occurs at rising edge.<br>10: TI6↑ TI6↓<br>INT6 occurs at falling edge.<br>11: TFF1↑ TFF1↓<br>INT6 occurs at rising edge. | | 1: UC5 clear enable | Timer 5 source clock<br>00: TI6<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |

Timer 5 capture timing

| | Capture control | INT6 Control |
|---|---|---|
| 00 | Capture disable | Interrupt occurs at the rising edge of TI6 (INT6) input. |
| 01 | CAP3 at TI6 rise<br>CAP4 at TI7 rise | |
| 10 | CAP3 at TI6 rise<br>CAP4 at TI6 fall | Interrupt occurs at the falling edge of TI6 (INT6) input. |
| 11 | CAP3 at TFF1 rise<br>CAP4 at TFF1 fall | Interrupt occurs at the rising edge of TI6 (INT6) input. |

Software capture

| | |
|---|---|
| 0 | The up counter 5 value is loaded to CAP3 |
| 1 | Always read as "1" |

Figure 3.9.7 16-Bit Timer Control Register (T5MOD) (2/2)

T5FFCR (0049H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | CAP4T6 | CAP3T6 | EQ7T6 | EQ6T6 | TFF6C1 | TFF6C0 |
| Read/Write | | | R/W | R/W | R/W | R/W | W | |
| After reset | | | 0 | 0 | 0 | 0 | 1 | 1 |
| Function | | | TFF6 invert trigger<br>0: Disable trigger<br>1: Enable trigger<br>Inverted when the UC value is latched to CAP4 | Inverted when the UC value is latched to CAP3 | Inverted when the UC value matches TREG7 | Inverted when the UC value matches TREG6 | 00: Invert TFF6<br>01: Set TFF6<br>10: Clear TFF6<br>11: Don't care<br>* Always read as "11" | |

Timer flip-flop 6 (TFF6) control

| | |
|---|---|
| 00 | Inverts the TFF6 value (Software inversion) |
| 01 | Sets TFF6 to "1" |
| 10 | Clear TFF6 to "0" |
| 11 | Don't care (Always read as "11") |

Timer flip-flop 6 (TFF6) invert trigger

| | |
|---|---|
| 0 | Trigger disable (Invert prohibition) |
| 1 | Trigger enable (Invert permission) |

CAP4T6: Inverted when the up counter value is latched to CAP4
CAP3T6: Inverted when the up counter value is latched to CAP3
EQ7T6: Inverted when up counter matches TREG7
EQ6T6: Inverted when up counter matches TREG6

Figure 3.9.8 16-Bit Timer 5 F/F Control (T5FFCR)

T45CR (003AH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | QCU | | | | PG1T | PG0T | DB6EN | DB4EN |
| Read/Write | R/W | | | | R/W | | | |
| After reset | 0 | | | | 0 | 0 | 0 | 0 |
| Function | Warm-up timer control | | | | PG1 shift trigger<br>0: 8-bit timer trigger (Timers 0, 1)<br>1: 16-bit timer trigger (Timer 5) | PG0 shift trigger<br>0: 8-bit timer trigger (Timers 0, 1)<br>1: 16-bit timer trigger (Timer 4) | Double buffer<br>0: Disable<br>1: Enable<br>Double buffer of TREG6 | Double buffer of TREG4 |

Double buffer control

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

DB6EN: Double buffer of TREG6
DB4EN: Double buffer of TREG4

Selecting PG0 shift trigger

| | |
|---|---|
| 0 | 8-bit timer trigger (Timers 0 and 1) |
| 1 | 16-bit timer trigger (Timer 4) |

Selecting PG1 shift trigger

| | |
|---|---|
| 0 | 8-bit timer trigger (Timers 0 and 1) |
| 1 | 16-bit timer trigger (Timer 5) |

Warm-up timer input control

| | |
|---|---|
| 0 | Uses 7 stage binary counter |
| 1 | Does not use 7 stage binary counter (Note 1) |

Note 1: In case of not using the 7 stage binary counter as a warm-up timer, a stable clock signal must be input from an external circuit.

Note 2: T45CR<bit6:4> are always read as “1”.

Figure 3.9.9 16-Bit Timer Trigger Control Register (T45CR)

TRUN (0020H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PRRUN | | T5RUN | T4RUN | P1RUN | P0RUN | T1RUN | T0RUN |
| Read/Write | R/W | | R/W | | | | | |
| After reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Prescaler and timer run/stop control<br>0: Stop and clear<br>1: Run (Count up) | | | | | | | |

Count operation

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

PRRUN: Operation of prescaler
T5RUN: Operation of 16-bit timer (Timer 5)
T4RUN: Operation of 16-bit timer (Timer 4)
P1RUN: Operation of PWM timer (PWM1/timer 3)
P0RUN: Operation of PWM timer (PWM0/timer 2)
T1RUN: Operation of 8-bit timer (Timer 1)
T0RUN: Operation of 8-bit timer (Timer 0)

Note: TRUN<bit6> is always read as “1”.

SYSCR0 (006EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | XEN | XTEN | RXEN | RXTEN | RSYSCK | WUEF | PRCK1 | PRCK0 |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Function | High-frequency oscillator (fc)<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs)<br>0: Stop<br>1: Oscillation | High-frequency oscillator (fc) after release STOP mode<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs) after release STOP mode<br>0: Stop<br>1: Oscillation | Select clock after release STOP mode<br>0: fc<br>1: fs | Warm-up timer (Write)<br>0: Don’t care<br>1: Start timer<br>(Read)<br>0: End warm up<br>1: Continue warm up | Select prescaler clock input<br>00: $f_{FPH}$<br>01: fs<br>10: fc/16<br>11: (Reserved) | |

Select prescaler clock input

| | | |
|---|---|---|
| 00 | $f_{FPH}$ | Clock divided by 4 |
| 01 | fs | |
| 10 | fc/16 | |
| 11 | (Reserved) | |

Figure 3.9.10 Timer Operation Control Register/System Clock Control Register

1. Prescaler

There are 9-bit prescaler and prescaler clock-selection registers to generate input clock signals for 8-bit timers 0 and 1, 16-bit timers 4 and 5, and serial interfaces 0 and 1.

Figure 3.9.11 shows the block diagram. Table 3.9.1 shows prescaler clock resolution into 8-/16-bit timers.

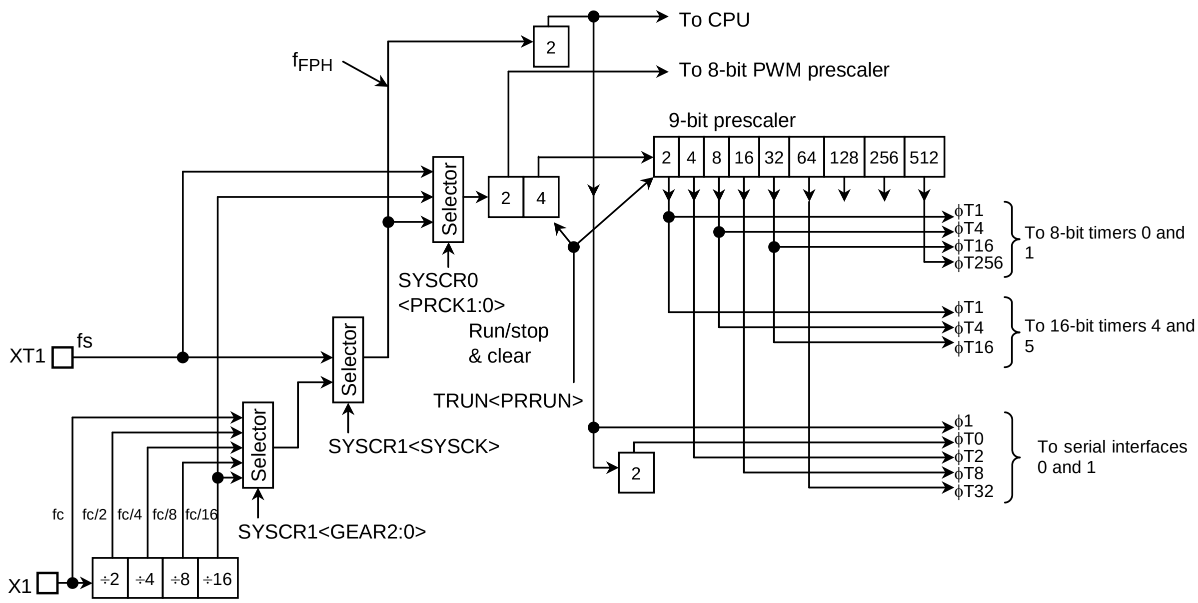



Figure 3.9.11 Block Diagram of Prescaler

Table 3.9.1 Prescaler Clock Resolution into 8-/16-Bit Timers

at fc = 20 MHz, fs = 32.768 kHz

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | Prescaler Clock Resolution | | | |
|---|---|---|---|---|---|---|
| | | | φT1 | φT4 | φT16 | φT256 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | fs/$2^3$ (244 μs) | fs/$2^5$ (977 μs) | fs/$2^7$ (3.9 ms) | fs/$2^{11}$ (62.5 ms) |
| 0 (fc) | | 000 (fc) | fc/$2^3$ (0.4 μs) | fc/$2^5$ (1.6 μs) | fc/$2^7$ (6.4 μs) | fc/$2^{11}$ (102.4 μs) |
| | | 001 (fc/2) | fc/$2^4$ (0.8 μs) | fc/$2^6$ (3.2 μs) | fc/$2^8$ (12.8 μs) | fc/$2^{12}$ (204.8 μs) |
| | | 010 (fc/4) | fc/$2^5$ (1.6 μs) | fc/$2^7$ (6.4 μs) | fc/$2^9$ (25.6 μs) | fc/$2^{13}$ (409.6 μs) |
| | | 011 (fc/8) | fc/$2^6$ (3.2 μs) | fc/$2^8$ (12.8 μs) | fc/$2^{10}$ (51.2 μs) | fc/$2^{14}$ (819.2 ms) |
| | | 100 (fc/16) | fc/$2^7$ (6.4 μs) | fc/$2^9$ (25.6 μs) | fc/$2^{11}$ (102.4 μs) | fc/$2^{15}$ (1.64 ms) |
| XXX | 01 (Low-frequency clock) | XXX | fs/$2^3$ (244 μs) | fs/$2^5$ (977 μs) | fs/$2^7$ (3.9 ms) | fs/$2^{11}$ (62.5 ms) |
| XXX | 10 (fc/16 clock) | XXX | fc/$2^7$ (6.4 μs) | fc/$2^9$ (25.6 μs) | fc/$2^{11}$ (102.4 μs) | fc/$2^{15}$ (1.64 ms) |

XXX: Don’t care

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

16-bit timer (φT1 to φT16)

8-bit timer (φT1 to φT256)

The clock selected from among $f_{FPH}$, fc/16, and fs is divided by 4 and input to this prescaler. This selection is made by prescaler clock selection register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to 00, selecting the $f_{FPH}$ clock input divided by 4.

The 16-bit timers 4 and 5 select among 3 clock inputs: ϕT1, ϕT4, and ϕT16 among the prescaler outputs.

This prescaler can be run or stopped by the timer operation control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to "1". The prescaler is cleared to zero and stops operation when <PRRUN> is set to "0".

Resetting clears <PRRUN> to "0" and stops the prescaler.

When the IDLE1 mode (In which only the oscillator operates) is used, set TRUN<PRRUN> to "0" to reduce the power consumption of the prescaler before the "HALT" instruction is executed.

2. Up counter

The up counter is a 16-bit binary counter which counts up according to the input clock specified by T4MOD<T4CLK1:0> register.

The alternatives for selecting the input clock include any one of the internal clocks ϕT1, ϕT4, and ϕT16 from the 9-bit prescaler (which is also used as an 8-bit timer), as well as the external clock from the TI4 pin (which itself can also be used as the P80 or INT4 pin). When reset, <T4CLK1:0> will be initialized to 00; this selects TI4 input mode. Counting or stop and clear of the counter are controlled by timer operation control register TRUN<T4RUN>.

When clearing is enabled, up counter UC4 will be cleared each time it matches the timer register TREG5. The "clear enable/disable" setting is made by T4MOD<CLE>.

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

3. Timer registers (TREG4 to TREG7)

These two 16-bit registers are used to set the interval time. When the value of up counter UC4 matches the value set in this timer register, the comparator match detect signal will be activated.

Setting data in both lower and upper registers are always needed. For example, either by using a 2-byte data transfer instruction, or by using 1-byte data transfer instructions twice: once for the lower 8 bits and once for the upper 8 bits in that order.

| TREG4 | | TREG5 | | |
|---|---|---|---|---|
| Upper 8 bits (TREG4H) | Lower 8 bits (TREG4L) | Upper 8 bits (TREG5H) | Lower 8 bits (TREG5L) | ⇒ Timer 4 |
| 000031H | 000030H | 000033H | 000032H | |

| TREG6 | | TREG7 | | |
|---|---|---|---|---|
| Upper 8 bits (TREG6H) | Lower 8 bits (TREG6L) | Upper 8 bits (TREG7H) | Lower 8 bits (TREG7L) | ⇒ Timer 5 |
| 000041H | 000040H | 000043H | 000042H | |

The TREG4 timer register is a double buffer structure, which is paired with a register buffer. The timer control register T45CR<DB4EN> controls whether the double buffer structure should be enabled or disabled. It is disabled when <DB4EN> = 0, and enabled when <DB4EN> = 1.

When the double buffer is enabled, the timing of data transfer from the register buffer to the timer register is at the match between the up counter (UC4) and timer register TREG5.

When reset, <DB4EN> will be initialized to "0"; this disables the double buffer. To use the double buffer, write data in the timer register, set <DB4EN> = 1, and then write the data which follows to the register buffer.

TREG4 and the register buffer are allocated to the same memory addresses 000030H/000031H. When <DB4EN> = 0, the same value will be written in both the timer register and in the register buffer. When <DB4EN> = 1, the value is written only into the register buffer. Therefore, to write the initial value into the timer register, the register buffer should be disabled.

4. Capture registers

These 16-bit registers are used to hold the values of the up counter.

Data in the capture registers should be read all 16 bits. For example, by a 2-byte data load instruction or by two 1-byte data load instructions, starting from the lower 8 bits followed by the upper 8 bits.

| CAP1 | | CAP2 | | |
|---|---|---|---|---|
| Upper 8 bits (CAP1H) | Lower 8 bits (CAP1L) | Upper 8 bits (CAP2H) | Lower 8 bits (CAP2L) | ⇒ Timer 4 |
| 000035H | 000034H | 000037H | 000036H | |

| CAP3 | | CAP4 | | |
|---|---|---|---|---|
| Upper 8 bits (CAP3H) | Lower 8 bits (CAP3L) | Upper 8 bits (CAP4H) | Lower 8 bits (CAP4L) | ⇒ Timer 5 |
| 000045H | 000044H | 000047H | 000046H | |

5. Capture input control

This circuit controls the timing of latching the value of up counter UC4 into CAP1 and CAP2. The latch timing of the capture register is controlled by register T4MOD<CAP12M1:0>. There are four possible settings:

- When T4MOD<CAP12M1:0> = 00

  Capture function is disabled. Disable is the default on resetting.

- When T4MOD<CAP12M1:0> = 01

  Data are loaded to CAP1 at the rising edge of the TI4 pin (which is also used as P80 or INT4) input, while data are loaded to CAP2 at the rising edge of the TI5 pin (also used as P81 or INT5) input. (Time difference measurement.)

- When T4MOD<CAP12M1:0> = 10

  Data are loaded to CAP1 at the rising edge of the TI4 pin input, and to CAP2 at the falling edge. Only in this setting, interrupt INT4 occurs at the falling edge. This results in pulse width measurement.

- When T4MOD<CAP12M1:0> = 11

  Data are loaded to CAP1 at the rising edge of timer flip-flop TFF1, and to CAP2 at the falling edge.

  Besides, the value of the up counter can be loaded to capture registers by software. Whenever "0" is written in T4MOD<CAP1IN>, the current value of the up counter will be loaded to capture register CAP1. It is necessary to keep the prescaler in RUN mode (TRUN<PRRUN> has to be 1).

6. Comparator

There are 16-bit comparators which compare the up counter UC4 value with the values set in TREG4 and TREG5 to detect matches. When a match is detected, these comparators generate interrupts INTTR4 or INTTR5 respectively. The up counter UC4 is cleared only when UC4 matches TREG5. The clearing of up counter UC4 can be disabled by setting T4MOD<CLE> = 0.

7. Timer flip-flop (TFF4)

This flip-flop is inverted by the match detect signal from the comparators or the latch signals to the capture registers. Disable or enable of inversion can be set for each element by T4FFCR<CAP2T4, CAP1T4, EQ5T4 and EQ4T4>. TFF4 will be inverted when "00" is written in T4FFCR<TFF4C1:0>. Also it is set to "1" when "01" is written, and cleared when "10"S is written. The value of TFF4 can be output to the timer output pin TO4 (which is also used as P82). TFF4 is undefined on resetting.

8. Timer flip-flop (TFF5)

This flip-flop is inverted when the comparator detects that the up counter signal (UC4) and the contents of the timer register TREG5 match, or when the contents of the up counter are latched to the capture register CAP2. Disable or enable of inversion can be set for each element by T4MOD<CAP2T5 and EQ575>. TFF5 will be inverted when "00" is written in T4FFCR<TFF5C1:0>. Also it is set to "1" when "01" is written, and cleared when "10" is written. The value of TFF5 can be output to the timer output pin TO5 (also used as P83). TFF5 is undefined on resetting.

Note: This flip-flop (TFF5) is contained only in the 16-bit timer 4.

(1) 16-bit timer mode

Generating interrupts at fixed intervals:

In this example, the interval time is set in the timer register TREG5 to generate the interrupt INTTR5.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | 0 | – | – | – | – | Stop timer 4. |
| INTET54 | ← | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | Enable INTTR5 and set interrupt level 4. Disable INTTR4. |
| T4FFCR | ← | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | Disable trigger. |
| T4MOD | ← | 0 | 0 | 1 | 0 | 0 | 1 | * | * | Select internal clock for input and |
| | | | | | (** = 01, 10, 11) | | | | | disable the capture function. |
| TREG5 | ← | * | * | * | * | * | * | * | * | Set the interval time (16 bits). |
| | | * | * | * | * | * | * | * | * | |
| TRUN | ← | 1 | X | – | 1 | – | – | – | – | Start timer 4. |

X: Don't care, –: No change

(2) 16-bit event counter mode

In 16-bit timer mode as described in (1) above, the timer can be used as an event counter by selecting the external clock (TI4 pin input) as the input clock. To read the value of the counter, first perform “software capture” once and read the captured value.

The counter counts at the rising edge of the TI4 pin input.

The TI4 pin can also be used as P80 or INT4.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | 0 | – | – | – | – | Stop timer 4. |
| P8CR | ← | – | – | – | – | – | – | – | 0 | Set P80 to input mode. |
| INTET54 | ← | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | Enable INTTR5 and sets interrupt level 4, while disables INTTR4. |
| T4FFCR | ← | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | Disable trigger. |
| T4MOD | ← | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | Select TI4 as the input clock. |
| TREG5 | ← | * | * | * | * | * | * | * | * | Set the number of counts (16 bits). |
| | | * | * | * | * | * | * | * | * | |
| TRUN | ← | 1 | X | – | 1 | – | – | – | – | Start timer 4. |

Note: When using this set up as an event counter, set the prescaler in RUN mode.

(3) 16-bit programmable pulse generation (PPG) output mode

The PPG mode is obtained by inversion of the timer flip-flop TFF4 that is enabled by the match of the up counter UC4 with either of the timer registers TREG4 or TREG5. TFF4 is also output to TO4 (which can be alternatively used as P82). In this mode, the following conditions must be satisfied:

(Set value of TREG4) < (Set value of TREG5)

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| T45CR | ← | 0 | X | X | X | – | – | – | 0 | Disable double buffer of TREG4. |
| TRUN | ← | – | X | – | 0 | – | – | – | – | Stop timer 4. |
| TREG4 | ← | * | * | * | * | * | * | * | * | Set the duty. (16 bits) |
| | | * | * | * | * | * | * | * | * | |
| TREG5 | ← | * | * | * | * | * | * | * | * | Set the cycle. (16 bits) |
| | | * | * | * | * | * | * | * | * | |
| T45CR | ← | 0 | X | X | X | – | – | – | 1 | Enable double buffer of TREG4. (Change the duty and cycle at the interrupt INTTR5.) |
| T4FFCR | ← | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | Set the mode to invert TFF4 at the match with TREG4 or TREG5, and also set TFF4 to “0”. |
| T4MOD | ← | 0 | 0 | 1 | 0 | 0 | 1 | * | * | Select the internal clock for the input, and disable the capture function. |
| | | | | (** = 01, 10, 11) | | | | | | |
| P8CR | ← | – | – | – | – | – | 1 | – | – | Assign P82 as TO4. |
| P8FC | ← | X | – | X | X | – | 1 | X | X | |
| TRUN | ← | 1 | X | – | 1 | – | – | – | – | Start timer 4. |

X: Don’t care, –: No change

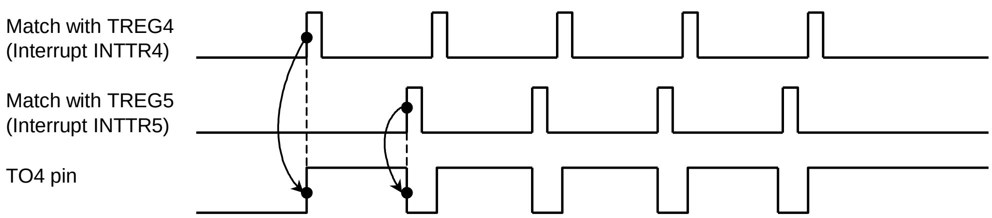



Figure 3.9.12 Programmable Pulse Generation (PPG) Output Waveforms

When the double buffer of TREG4 is enabled in this mode, the value of register buffer 4 will be shifted into TREG4 on finding a match with TREG5. This feature makes the handling of low duty waves easy.

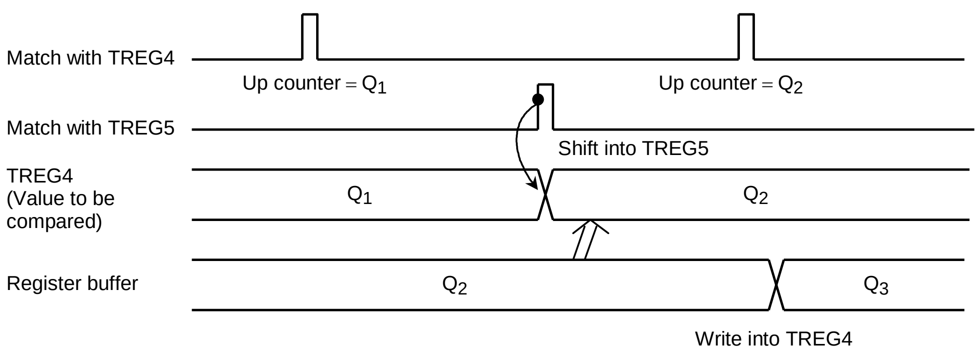



Figure 3.9.13 Operation of Register Buffer

Figure 3.9.14 shows the block diagram of this mode.

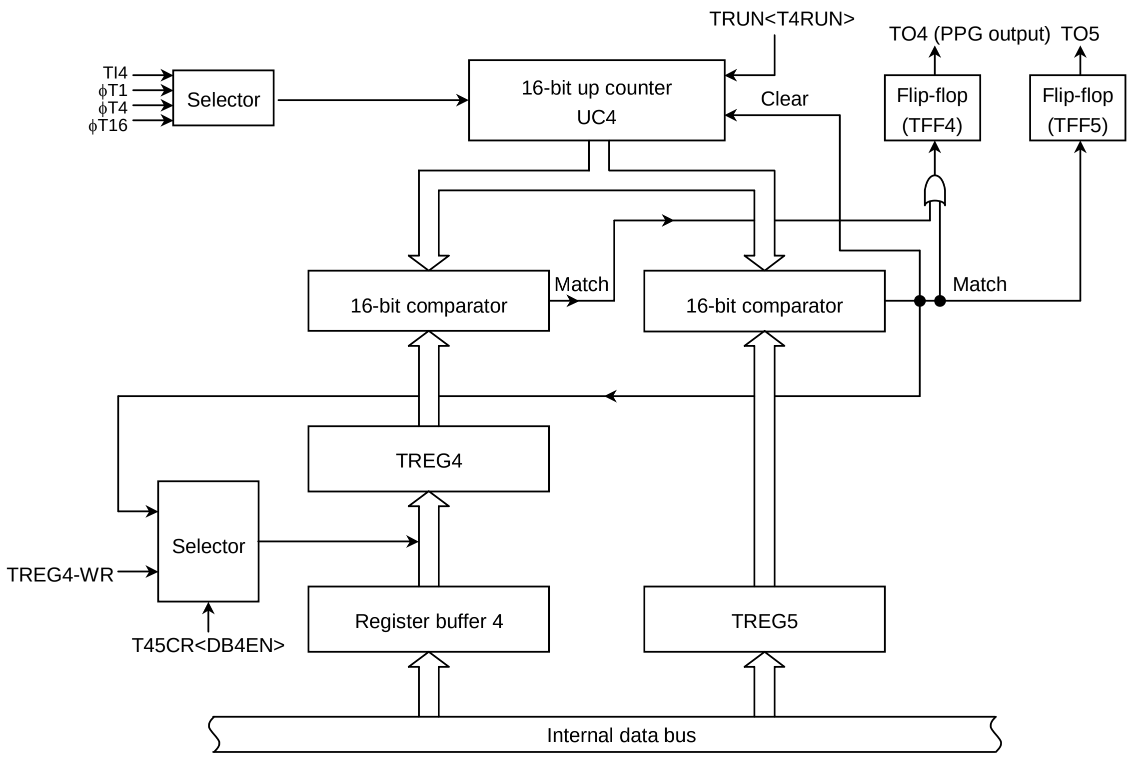



Figure 3.9.14  Block Diagram of 16-Bit PPG Mode

(4) Application examples of the capture function

It is possible to enable or disable the loading of up counter (UC4) values into the capture registers CAP1 and CAP2, the timer flip-flop TFF4 inversion due to match detection by comparators CP4 and CP5, and the output of the TFF4 status to the TO4 pin. Combined with the interrupt function, these options can be applied in many ways, for example:

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

These four application examples are described in detail below.

1. One-shot pulse output from external trigger pulse

To program this application, set the up counter UC4 in free-running mode with the internal input clock, input the external trigger pulse from the TI4 pin, and load the value of the up counter into capture register CAP1 at the rising edge of the TI4 pin. Then set T4MOD<CAP12M1:0> = 01.

When the interrupt INT4 is generated at the rising edge of the TI4 input, set the CAP1 value (c) plus a delay time (d) to TREG4 (= c + d), and set the above set value (c + d) plus a one-shot pulse width (p) to TREG5 (= c + d + p). When the interrupt INT4 occurs, the T4FFCR<EQ5T4 and EQ4T4>register should be set so that the TFF4 inversion is enabled only when the up counter value matches TREG4 or TREG5. When interrupt INTTR5 occurs, this inversion will be disabled.

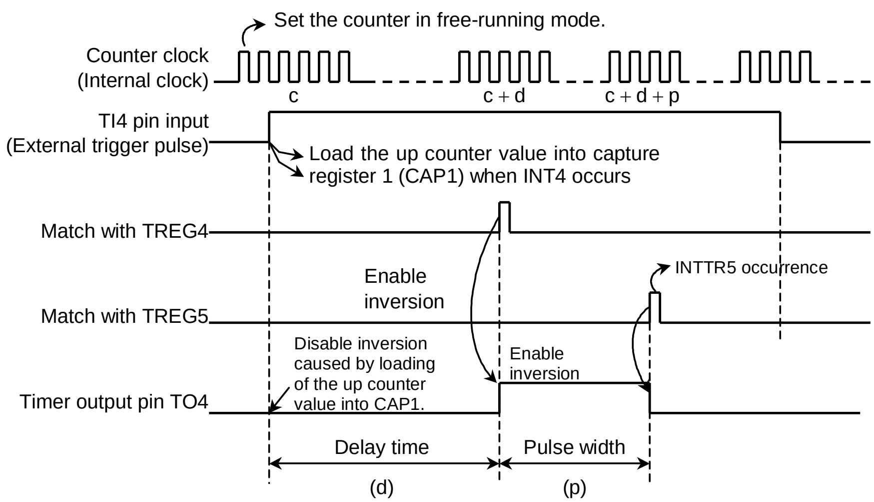



Figure 3.9.15 One-shot Pulse Output (with delay)

Setting example: To output a 2 ms one-shot pulse to the external trigger pulse to TI4 pin, with 3 ms delay

Clock condition
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: $f_{FPH}$

Main setting

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| T4MOD | ← | – | – | 1 | 0 | 1 | 0 | 0 | 1 | Keep counting (Free running)<br>Count with φT1.<br>Load the up counter value into CAP1 at the rising edge of TI4 pin input. |
| T4FFCR | ← | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | Clear TFF4 to zero.<br>Disable TFF4 inversion. |
| P8CR | ← | – | – | – | – | – | 1 | – | – | Select P82 as the TO4 pin. |
| P8FC | ← | X | – | X | X | – | 1 | X | X | |
| INTE45 | ← | – | – | – | – | 1 | 1 | 0 | 0 | Enable INT4, and disable INTTR4 and INTTR5. |
| INTET54 | ← | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| TRUN | ← | 1 | X | – | 1 | – | – | – | – | Start timer 4. |

Setting of INT4

| | | | |
|---|---|---|---|
| TREG4 | ← | CAP1 + 3 ms/φT1 | |
| TREG5 | ← | TREG4 + 2 ms/φT1 | |
| T4FFCR | ← | – – – – – 1 1 – – | Enable TFF4 inversion when the up counter value matches TREG4 or TREG5. |
| INTET54 | ← | 1 1 0 0 – – – – | Enable INTTR5. |

Setting of INTTR5

| | | | |
|---|---|---|---|
| T4FFCR | ← | – – – – – 0 0 – – | Disable TFF4 inversion when the up counter value matches TREG4 or TREG5. |
| INTET54 | ← | 1 0 0 0 – – – – | Disable INTTR5. |

X: Don't care, –: No change

When a delay time is unnecessary, invert the timer flip-flop TFF4 by loading the up counter value into capture register 1 (CAP1). Then set TREG5 to the CAP1 value (c) plus the one-shot pulse width (p) when an INT4 interrupt occurs. The TFF4 inversion should be enabled when the up counter (UC4) value matches TREG5, and disabled when generating the interrupt INTTR5.

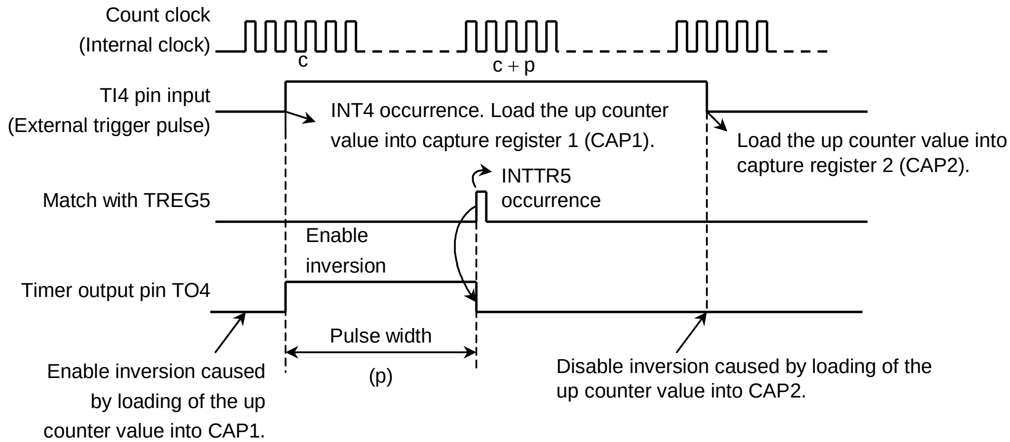



Figure 3.9.16 One-shot Pulse Output (without delay)

2. Frequency measurement

The frequency of the external clock can be measured in this mode. The clock signal is input through the TI4 pin, and its frequency is measured by the 8-bit timers (Timer 0 and timer 1) and the 16-bit timer/event counter (Timer 4).

The TI4 pin input should be selected for the clock input of timer 4. The value of the up counter is loaded into the capture register CAP1 at the rising edge of the timer flip-flop TFF1 of the 8-bit timers (Timer 0 and timer 1). Similarly, the up counter value is loaded into CAP2 at the falling edge of the TFF1 flip-flop.

The frequency is calculated by the difference between the values loaded in CAP1 and CAP2, at the moment when an interrupt (INTT0 or INTT1) is generated by either 8-bit timer.

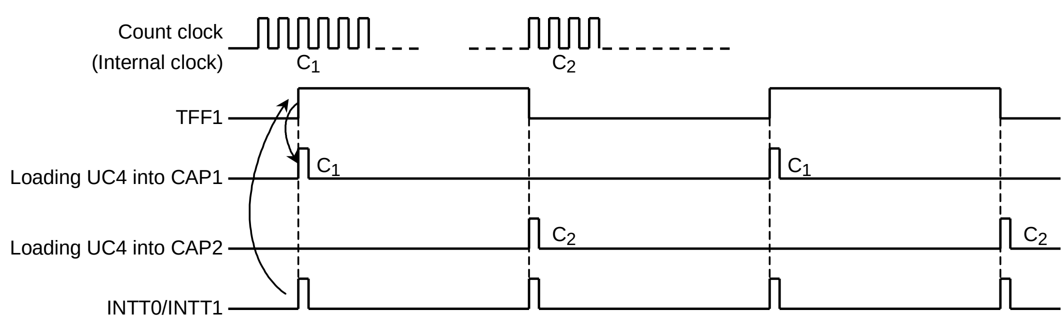



Figure 3.9.17 Frequency Measurement

For example, if the value for the level "1" width of TFF1 of the 8-bit timer is set to 0.5 s and the difference between CAP1 and CAP2 is 100, the frequency will be 100 ÷ 0.5 s = 200 Hz.

3. Pulse width measurement

This mode allows measurement of the H level width of an external pulse. While keeping the 16-bit timer/event counter counting (Free running) with the internal clock input, the external pulse is input via the TI4 pin. Then the capture function is used to load the UC4 values into CAP1 and CAP2 on the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT4 occurs at the falling edge of TI4.

The pulse width is obtained from the difference between the values of CAP1 and CAP2 and the internal clock cycle.

For example, if the internal clock is 0.8 μs and the difference between CAP1 and CAP2 is 100, the pulse width will be 100 × 0.8 μs = 80 μs.

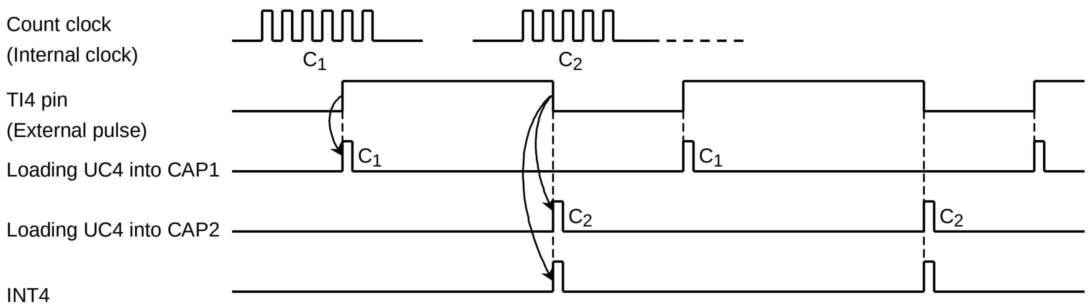



Figure 3.9.18 Pulse Width Measurement

Note: External interrupt INT4 occurs at the falling edge of the TI4 pin input only in this pulse width measuring mode (T4MOD<CAP12M1:0> = 10). In other modes, it occurs at the rising edge.

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

4. Time difference measurement

This mode is used to measure the difference in time between the rising edges of external pulses input via TI4 and TI5.

While keeping the 16-bit timer/event counter (Timer 4) counting (Free running) with the internal clock, the UC4 value is loaded into CAP1 on the rising edge of the input pulse to TI4. Then the interrupt INT4 is generated.

Similarly, the UC4 value is loaded into CAP2 on the rising edge of the input pulse to TI5, generating the interrupt INT5.

The time difference between these pulses can be obtained from the difference between the time counts at which loading the up counter value into CAP1 and CAP2 was performed.

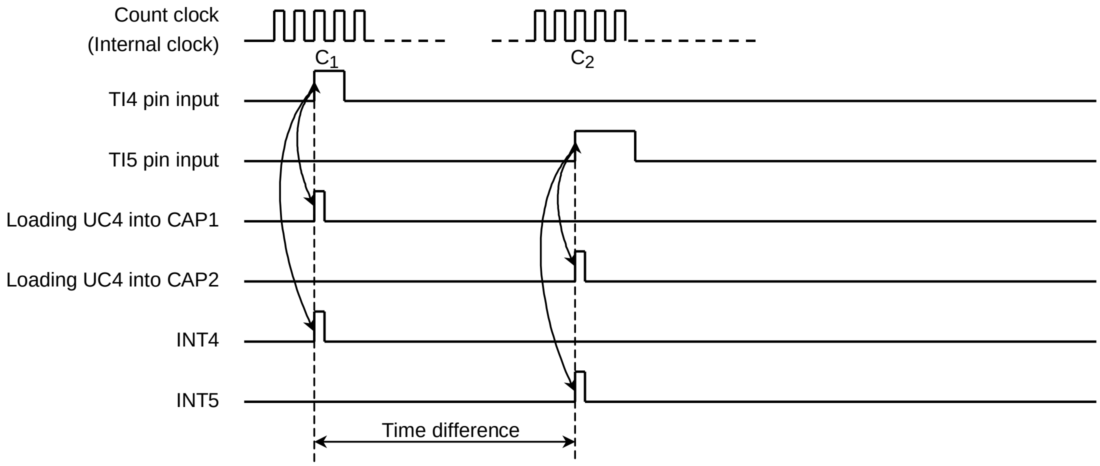



Figure 3.9.19 Time Difference Measurement

(5) Different phased pulses output mode (This mode can be used only with timer 4.)

In this mode, signals with any phase can be output by the free-running up counter UC4.

When the value in up counter UC4 and the value in TREG4 (TREG5) match, the value in TFF4 (TFF5) is inverted and output to TO4 (TO5).

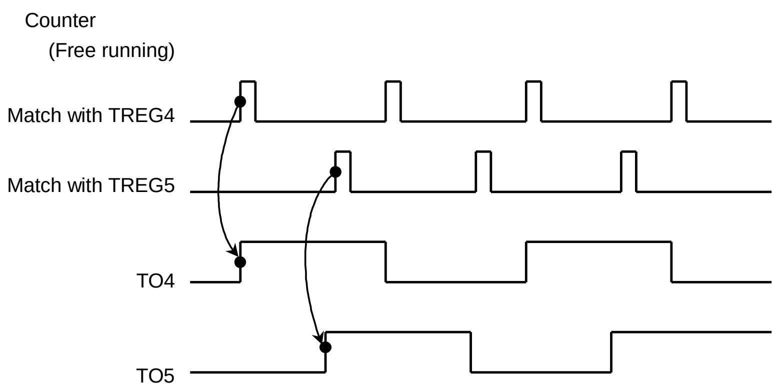



Figure 3.9.20 Phase Output

The periods of the output waveforms above (Expressed as counter overflow time) are listed in Table 3.9.2.

Table 3.9.2 Timer Output Periods in Different Phased Pulse Output Mode
(Expressed as counter overflow time)

at fc = 20 MHz, fs = 32.768 kHz

| System Clock Selected <SYSCK> | Prescaler Clock Selected <PRCK1:0> | Gear Value <GEAR2:0> | Counter Overflow Time | | |
|---|---|---|---|---|---|
| | | | φT1 | φT4 | φT16 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | 16.0s | 64.0 s | 256.0 s |
| 0 (fc) | | 000 (fc) | 26.21 ms | 104.86 ms | 419.43 ms |
| | | 001 (fc/2) | 52.43 ms | 209.72 ms | 838.86 ms |
| | | 010 (fc/4) | 104.86 ms | 419.43 ms | 1.68 s |
| | | 011 (fc/8) | 209.72 ms | 838.86 ms | 3.36 s |
| | | 100 (fc/16) | 419.43 ms | 1.68 s | 6.71 s |
| XXX | 01 (Low-frequency clock) | XXX | 16.0 s | 64.0 s | 256.0 s |
| XXX | 10 (fc/16 clock) | XXX | 419.43 ms | 1.68 s | 6.71 s |

XXX: Don’t care

## 3.10 Stepping Motor Control/Pattern Generation Port

The TMP93CS40/TMP93CS41 contains two 4-bit hardware stepping motor control/pattern generation channels, PG0 and PG1, (hereinafter called PG) which actuate in synchronization with the (8-bit/16-bit) timers. PG (PG0 and PG1) shares the 8-bit input/output port with P6.

The output on channel 0 (PG0) is updated in synchronization with the 8-bit timer 0, 8-bit timer 1, or 16-bit timer 4. The output on channel 1 (PG1) is updated in synchronization with the 8-bit timer 0, the 8-bit timer 1, or the 16-bit timer 5.

The PG ports are controlled by the control register (PG01CR) and can select either stepping motor control mode or pattern generation mode. Each bit of P6 can be used for a PG port.

PG0 and PG1 can be used independently.

Since the two PG channels operate in the same manner, except for the following points, only the operation of PG0 will be explained below.

Differences between PG0 and PG1

| | PG0 | PG1 |
|---|---|---|
| Trigger signal | from timer 4 | from timer 5 |

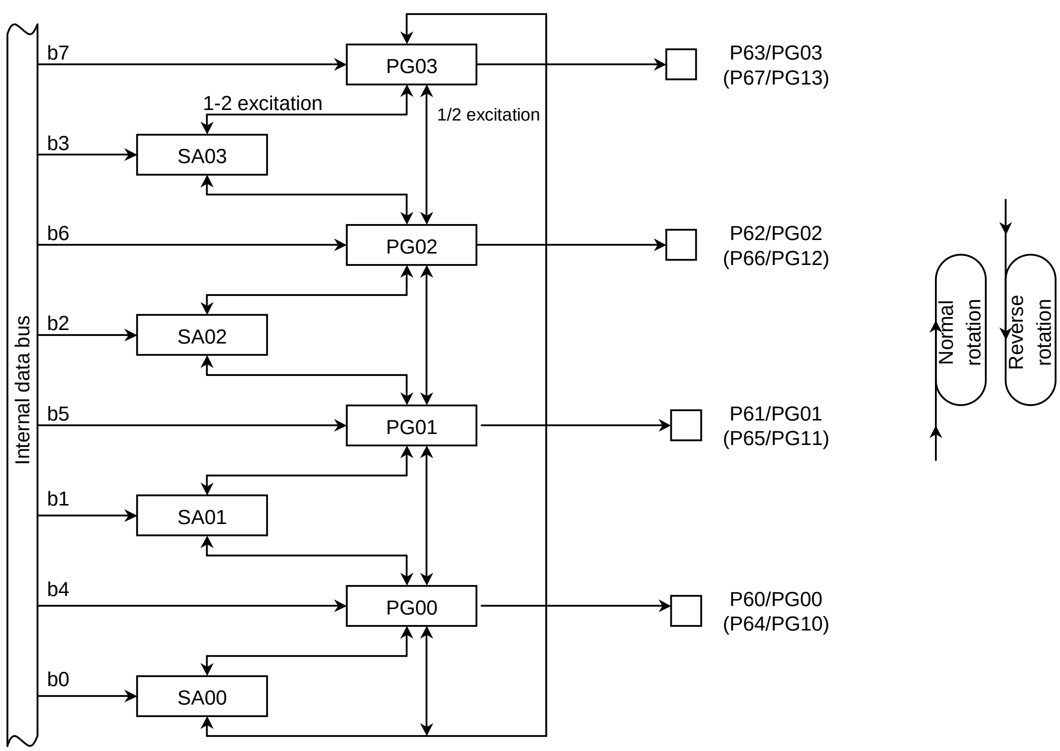



Figure 3.10.1 PG Block Diagram

| PG01CR (004EH) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | PAT1 | CCW1 | PG1M | PG1TE | PAT0 | CCW0 | PG0M | PG0TE |
| | Read/Write | R/W | | | | R/W | | | |
| | After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | PG1 write mode<br>0: 8-bit write<br>1: 4-bit write | PG1 rotation direction<br>0: Normal rotation<br>1: Reverse rotation | PG1 mode (Excitation)<br>0: 1-step excitation or 2-step excitation<br>1: 1-to-2 step excitation | PG1 trigger input enable<br>0: Disable<br>1: Enable | PG0 write mode<br>0: 8-bit write<br>1: 4-bit write | PG0 rotation direction<br>0: Normal rotation<br>1: Reverse rotation | PG0 mode (Excitation)<br>0: 1-step excitation or 2-step excitation<br>1: 1-to-2 step excitation | PG0 trigger input enable<br>0: Disable<br>1: Enable |

PG0 trigger input enable

| | |
|---|---|
| 0 | Trigger input disable for PG0 |
| 1 | Trigger input enable for PG0 |

Set the operation mode for PG0

| | |
|---|---|
| 0 | 1- or 2-step excitation (Full step) |
| 1 | 1-to-2 step excitation (Half step)/PG mode |

PG0 (Stepping motor control) rotation direction control

| | |
|---|---|
| 0 | Normal rotation/PG mode |
| 1 | Reverse rotation |

Selecting PG0 write mode

| | |
|---|---|
| 0 | 8-bit write |
| 1 | 4-bit write/PG mode<br>(Only the shift alternate register can be written.) |

Figure 3.10.2 Pattern Generation Control Register (PG01CR) (1/2)

PG01CR (004EH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PAT1 | CCW1 | PG1M | PG1TE | PAT0 | CCW0 | PG0M | PG0TE |
| Read/Write | R/W | | | | R/W | | | |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | PG1 write mode<br>0: 8-bit write<br>1: 4-bit write | PG1 rotation direction<br>0: Normal rotation<br>1: Reverse rotation | PG1 mode (Excitation)<br>0: 1-step excitation or 2-step excitation<br>1: 1 to 2 step excitation | PG1 trigger input enable<br>0: Disable<br>1: Enable | PG0 write mode<br>0: 8-bit write<br>1: 4-bit write | PG0 rotation direction<br>0: Normal rotation<br>1: Reverse rotation | PG0 mode (Excitation)<br>0: 1-step excitation or 2-step excitation<br>1: 1 to 2 step excitation | PG0 trigger input enable<br>0: Disable<br>1: Enable |

PG1 trigger input enable

| | |
|---|---|
| 0 | Trigger input disable for PG1 |
| 1 | Trigger input enable for PG1 |

Set the operation mode for PG1

| | |
|---|---|
| 0 | 1- or 2-step excitation (Full step) |
| 1 | 1-to-2 step excitation (Half step)/PG mode |

PG1 (Stepping motor control)
Rotating direction control

| | |
|---|---|
| 0 | Normal rotation/PG mode |
| 1 | Reverse rotation |

Selecting PG1 write mode

| | |
|---|---|
| 0 | 8-bit write |
| 1 | 4-bit write/PG mode<br>(Only the shift alternate register can be written.) |

Figure 3.10.3 Pattern Generation Control Register (PG01CR) (2/2)

PG0REG (004CH)

Prohibit read-modify-write

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PG03 | PG02 | PG01 | PG00 | SA03 | SA02 | SA01 | SA00 |
| Read/Write | W | | | | R/W | | | |
| After reset | 0 | 0 | 0 | 0 | Undefined | | | |
| Function | Pattern generation 0 (PG0) output latch register (PG0 can be read by reading the port (P6) that is assigned to PG) | | | | Shift alternate register 0 for the PG mode (4-bit write) register | | | |

Figure 3.10.4 Pattern generation 0 register (PG0REG)

PG1REG (004DH)

Prohibit read-modify-write

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | PG13 | PG12 | PG11 | PG10 | SA13 | SA12 | SA11 | SA10 |
| Read/Write | W | | | | R/W | | | |
| After reset | 0 | 0 | 0 | 0 | Undefined | | | |
| Function | Pattern generation 1 (PG1) output latch register (PG1 can be read by reading the port (P6) that is assigned to PG) | | | | Shift alternate register 1 for the PG mode (4-bit write) register | | | |

Figure 3.10.5 Pattern Generation 1 Register (PG1REG)

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| T45CR (003AH) | Bit symbol | QCU | | | | PG1T | PG0T | DB6EN | DB4EN |
| | Read/Write | R/W | | | | R/W | | | |
| | After reset | 0 | | | | 0 | 0 | 0 | 0 |
| | Function | Watchdog/ warm-up timer control | | | | PG1 shift trigger 0: 8-bit timer trigger (Timers 0 and 1) 1: 16-bit timer trigger (Timer 5) | PG0 shift trigger 0: 8-bit timer trigger (Timers 0 and 1) 1: 16-bit timer trigger (Timer 4) | Double buffer 0: Disable 1: Enable<br>Double buffer for TREG6 | Double buffer for TREG4 |

Double buffer control

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

DB6EN: Double buffer for TREG6
DB4EN: Double buffer for TREG4

Selecting PG0 shift trigger

| | |
|---|---|
| 0 | 8-bit timer trigger (Timers 0 and 1) |
| 1 | 16-bit timer trigger (Timer 4) |

Selecting PG1 shift trigger

| | |
|---|---|
| 0 | 8-bit timer trigger (Timers 0 and 1) |
| 1 | 16-bit timer trigger (Timer 5) |

Watchdog timer/warm-up timer input control

| | |
|---|---|
| 0 | 7-stage binary counter used |
| 1 | 7-stage binary counter not used (Note 1) |

Note 1: When the 7-stage binary counter is not used as a warm-up timer, a stable clock must be input from an external circuit.

Note 2: T45CR<bit6:4> are always 1.

Figure 3.10.6 16-Bit Timer Trigger Control Register (T45CR)

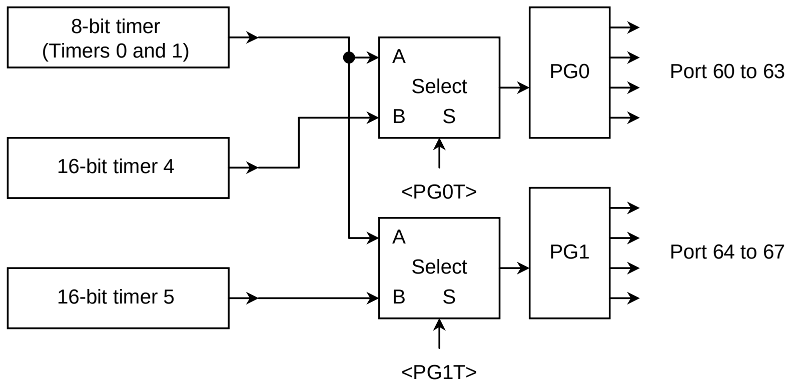



Figure 3.10.7 Connection between Timer and Pattern Generator

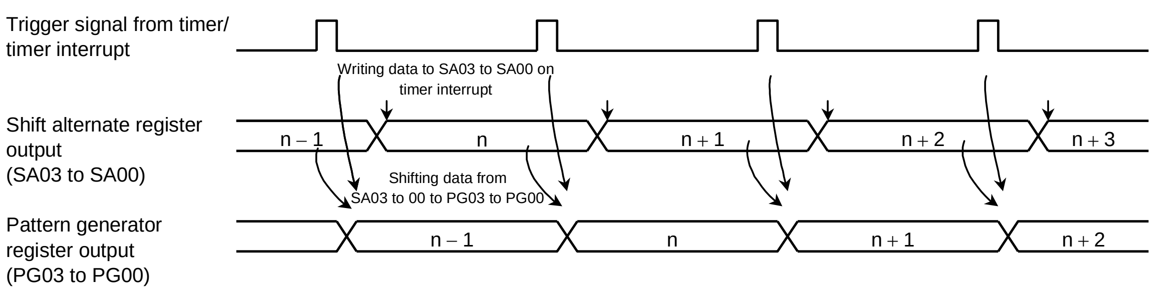
(1) Pattern generation mode

When PG01CR<PAT0> = "1", PG functions as a pattern generator. In this mode data is written from the CPU to the shift alternate register only. The pattern data is then written from the shift alternate register to the pattern generator register synchronized to the shift trigger interrupt from the timer.

In this mode, PG01CR<PG0M> should be set to "1", PG01CR<CCW0> to "0", and PG01CR<PG0TE> to "1".

The output from the pattern generator goes to port 6; since port or functions can be switched by the bit settings in the port function control register, P6FC, any port pin can be assigned to pattern generator output.

Figure 3.10.8 shows the block diagram for this mode.




Example of Pattern Generation Mode

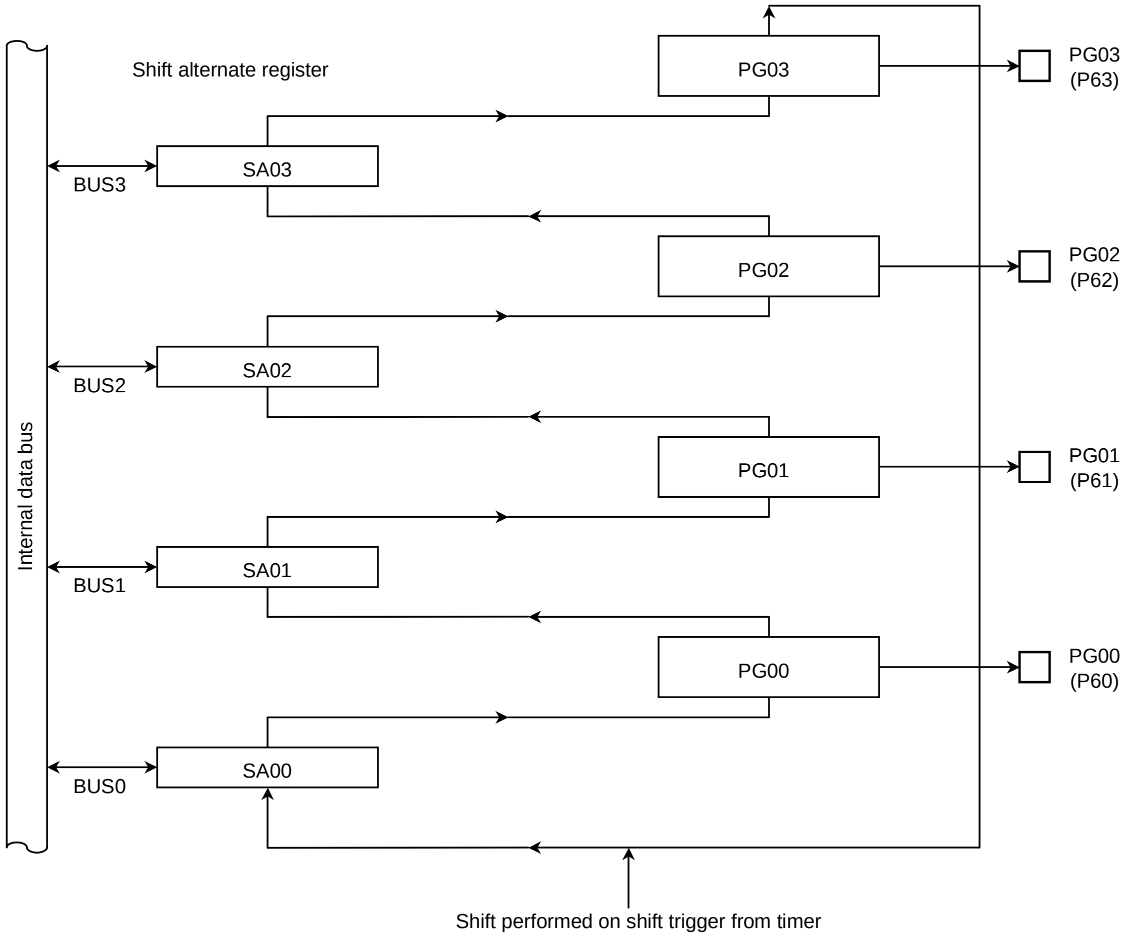



Figure 3.10.8 Pattern Generation Mode Block Diagram (PG0)

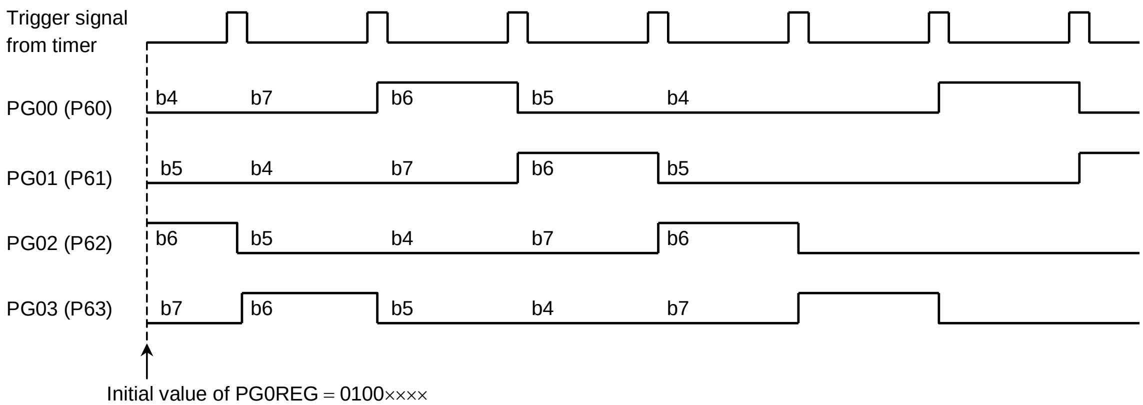
In pattern generation mode, only writing to the output latch can be disabled by hardware. All other functions behave in the same way as 1 to 2 step excitation in stepping motor control port mode. Hence, data shifted on the trigger signal from a timer must be written before the next trigger signal is output.

(2) Stepping motor control mode

1. 4-phase 1-step/2-step excitation

Figure 3.10.9 and Figure 3.10.10 show the output waveforms for 4-phase 1 excitation and 4-phase 2 excitation respectively when channel 0 (PG0) is selected.

a. Normal rotation




Note: bn indicates the initial value of PG0REG ← b7 b6 b5 b4××××

b. Reverse rotation

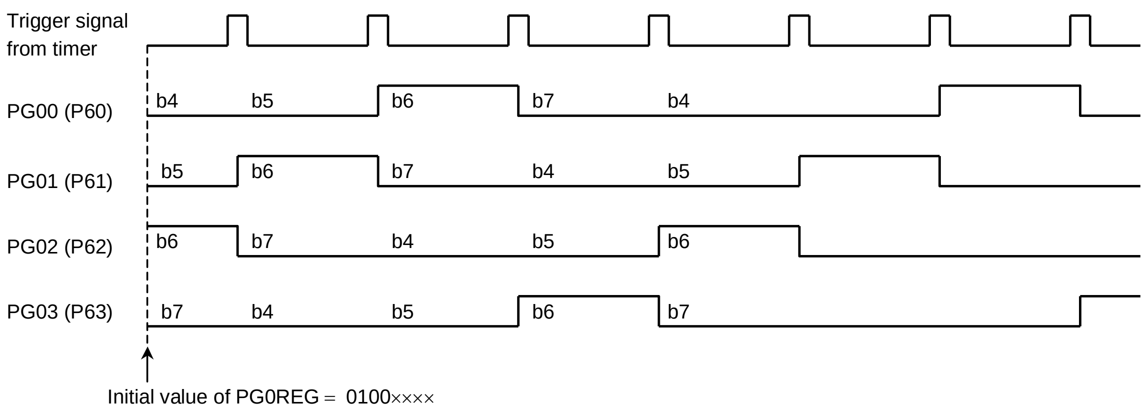



Figure 3.10.9 Output Waveforms for 4-Phase 1-Step Excitation

(Normal rotation and reverse rotation)

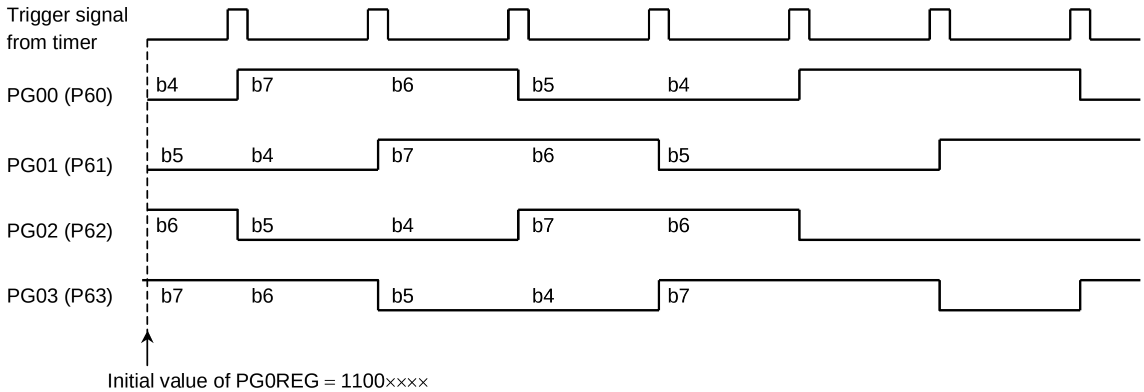



Figure 3.10.10 Output Waveforms for 4-phase 2-step Excitation (Normal rotation)

The output from PG0 (P6) is latched on the rising edge of the trigger signal from the timer.

The direction of shift is specified by the setting of PG01CR<CCW0>: Normal rotation (PG00→PG01→PG02→PG03) is selected when <CCW0> is set to "0"; reverse rotation (PG00←PG01←PG02←PG03) is selected when <CCW0> is set to "1". 4-phase 1-step excitation will be selected when only one bit is set to "1" during the initialization of PG, while 4-phase 2-step excitation will be selected when two consecutive bits are set to "1".

The value in the shift alternate registers are ignored when 4-phase 1-step/2-step excitation mode is selected.

Figure 3.10.11 shows the block diagram.

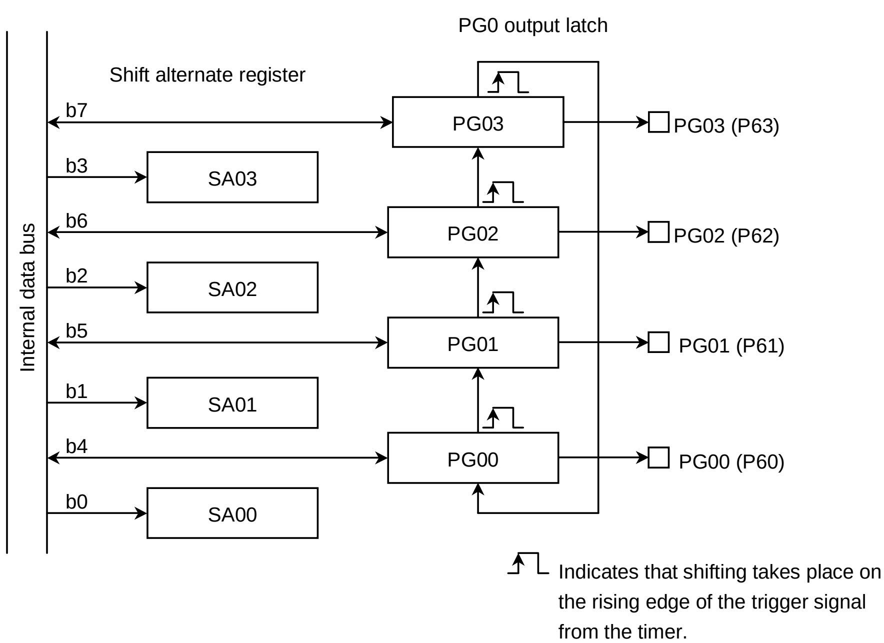
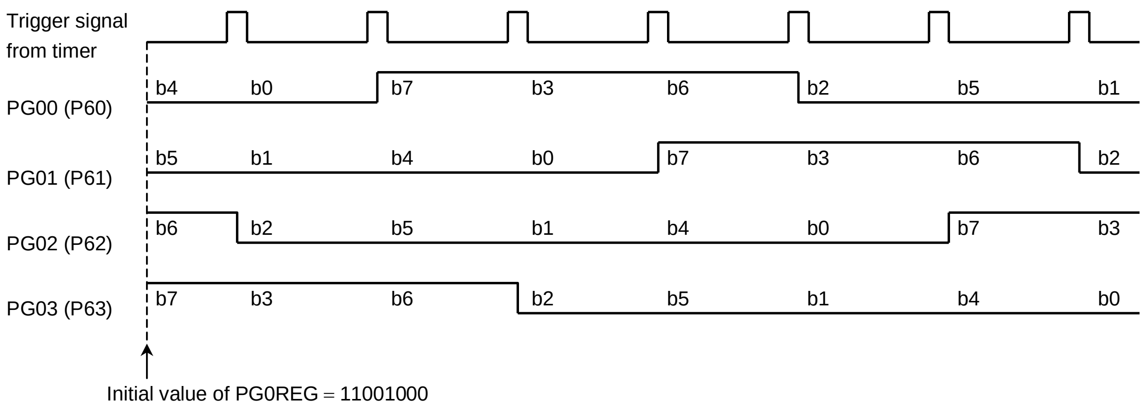



Figure 3.10.11 Block Diagram for 4-phase 1-step Excitation/2-step Excitation (Normal rotation)

2. 4-phase 1 to 2 step excitation

Figure 3.10.12 shows the output waveforms for 4-phase 1-2 step excitation when channel 0 is selected.

a. Normal rotation




Note: bn denotes the initial value PG0REG ← b7 b6 b5 b4 b3 b2 b1 b0

b. Reverse rotation

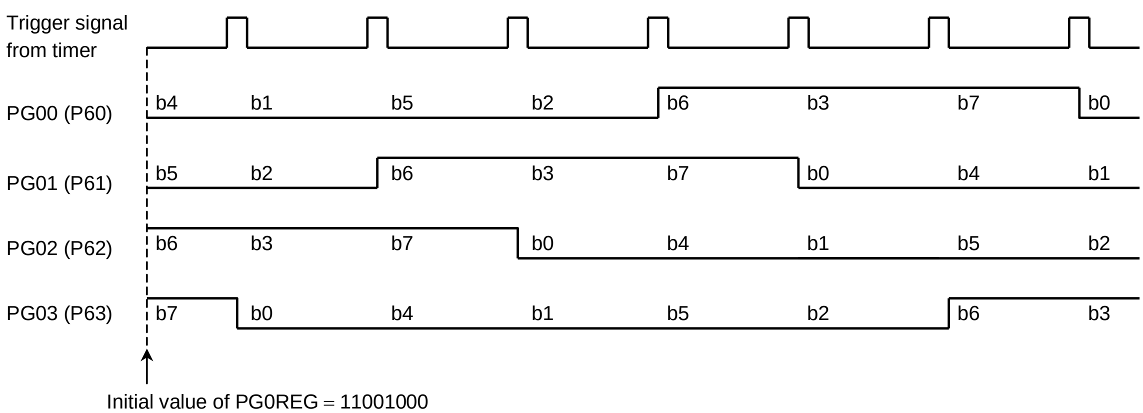



Figure 3.10.12 Output Waveforms for 4-phase 1- to 2-step Excitation
(Normal rotation and reverse rotation)

The initialization sequence for 4-phase 1-2 step excitation is as follows.

By rearranging the initial value b7 b6 b5 b4 b3 b2 b1 b0 to b7 b3 b6 b2 b5 b1 b4 b0, three consecutive bits are set to 1 and the other bits are set to 0 (Positive logic).

For example, if b7, b3, and b6 are set to 1, the initial value becomes 11001000, producing the output waveforms shown in Figure 3.10.12.

To generate a negative logic output waveform, the 1's and 0's in the initial value must be inverted. For example, to change the output waveform shown in Figure 3.10.12 negative logic, change the initial value to 00110111.

The operation will be explained below for channel 0.

The output from PG0 (P6) and from the shift alternate register (SA0) for pattern generation is latched on the rising edge of the trigger signal from the timer. The shift direction is set by PG01CR<CCW0>.

Figure 3.10.13 shows the block diagram.

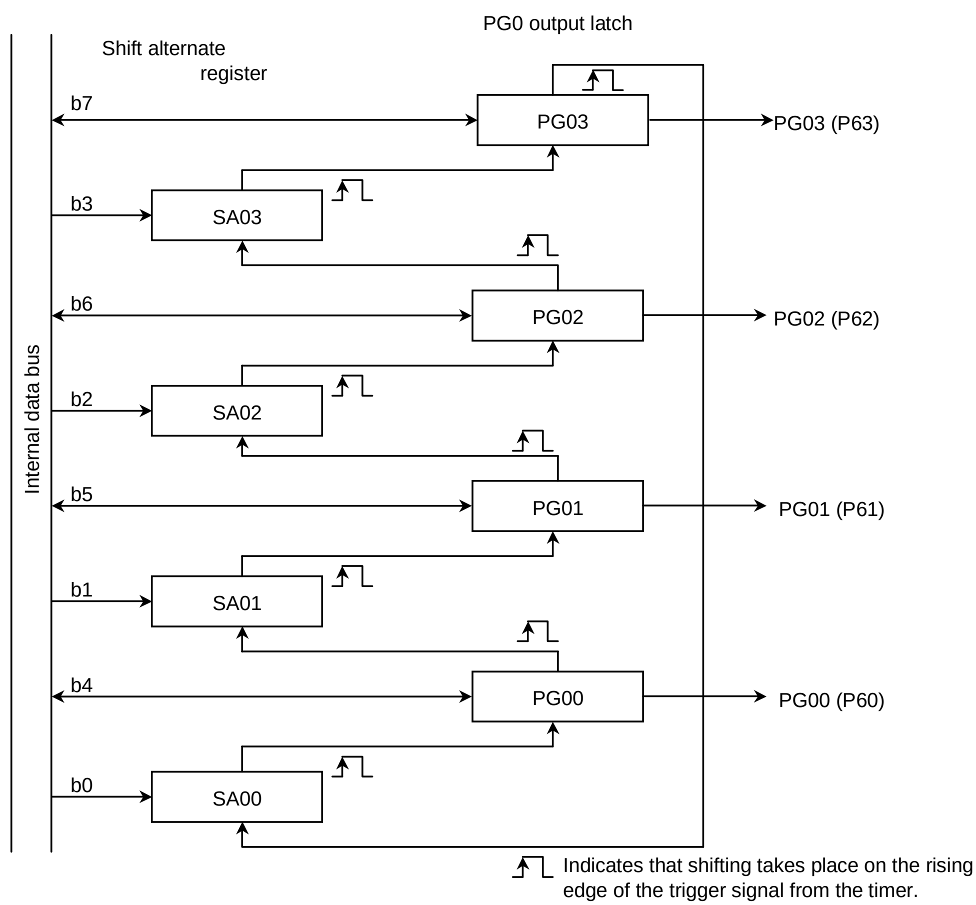



Figure 3.10.13 Block Diagram for 4-phase 1- to 2-step Excitation (Normal rotation)

Setting example: To drive channel 0 (PG0) using 4-phase 1-2 step excitation (Normal rotation) when timer 0 is selected, set each register as follows.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | – | 0 | Stop timer 0, and clear it to zero. |
| TMOD | ← | 0 | 0 | X | X | – | – | 0 | 1 | Set 8-bit timer mode and select $\phi$T1 as the input clock for timer 0. |
| TFFCR | ← | X | X | X | 0 | 1 | 0 | 1 | 0 | Clear TFF1 to zero and enable the inversion trigger using timer 0. |
| TREG0 | ← | * | * | * | * | * | * | * | * | Set the cycle in the timer register. |
| P6CR | ← | – | – | – | – | 1 | 1 | 1 | 1 | Set bits P60 to P63 to output mode. |
| P6FC | ← | – | – | – | – | 1 | 1 | 1 | 1 | Set bits P60 to P63 to PG output. |
| PG01CR | ← | – | – | – | – | 0 | 0 | 1 | 1 | Select PG0 4-phase 1 to 2 step excitation mode and normal rotation. |
| PG0REG | ← | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | Set an initial value. |
| TRUN | ← | 1 | X | – | – | – | – | – | 1 | Start timer 0. |

X: Don't care, –: No change

(3) Trigger signal from timer

The trigger signal from the timer used by PG is not the same as the trigger signal for the timer flip-flop (TFF1, TFF4, TFF5, and TFF6); they differ as shown in Table 3.10.1 depending on the operation mode of the timer.

Table 3.10.1 Trigger Signal Selection

| | TFF1 Inversion | PG Shift |
|---|---|---|
| 8-bit timer mode | Selected by TFFCR<TFF1IS> when the up counter value matches TREG0 or TREG1 value | Selected by TFFCR<TFF1IS> when the up counter value matches TREG0 or TREG1 value |
| 16-bit timer mode | When the up counter value matches both TREG0 and TREG1 values (the value of up counter = TREG1 $\times 2^8 +$ TREG0) | When the up counter value matches both TREG0 and TREG1 values (the value of up counter = TREG1 $\times 2^8 +$ TREG0) |
| PPG output mode | When the up counter value matches both TREG0 and TREG1 | When the up counter value matches TREG1 value (PPG cycle) |
| PWM output mode | When the up counter value matches TREG0 value and PWM cycle | Trigger signal for PG is not generated |

Note: To shift PG, TFFCR<TFF1IE> must be set to 1 to enable TFF1 inversion.

Channel 1 of PG can be synchronized with the 16-bit timer timer 4/timer 5. In this case, the PG shift trigger signal from the 16-bit timer is output only when the up counter UC4/UC5 value matches TREG5/TREG7.

When using a trigger signal from timer 4, set either T4FFCR<EQ5T4> or T4MOD<EQ5T5> to “1”; a trigger is generated when the value in UC4 and the value in TREG5 match. When using a trigger signal from timer 5, set T5FFCR<EQ7T6> to “1”; a trigger is generated when the value in UC5 and the value in TREG7 match.

(4) Application of PG and timer output

As explained in the previous section trigger signal from timer, the timings for shifting PG and inverting TFF differ depending on the timer mode. An application which operates PG while operating an 8-bit timer in PPG mode is explained below.

To drive a stepping motor, a synchronizing signal is required for the excitation timing, in addition to the value of each phase (PG output). In this application, port 6 is used as a stepping motor control port to output a synchronizing signal to the TO1 pin (shared with P71).

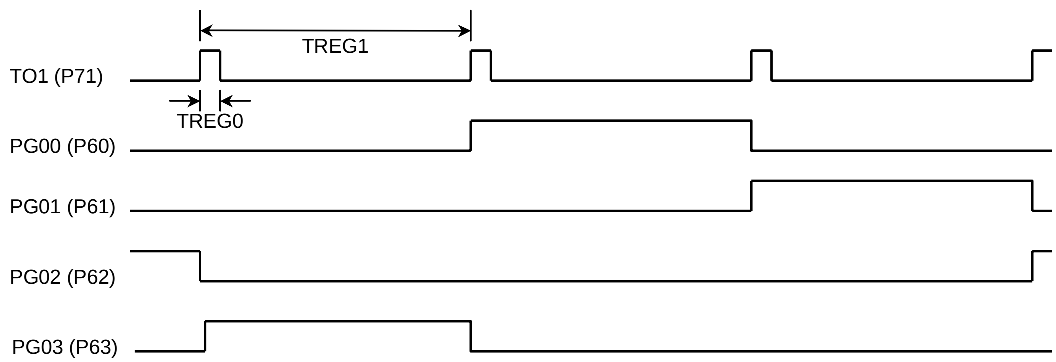



Figure 3.10.14 Output Waveforms for 4-Phase 1-step Excitation

Setting example:

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | ← | – | X | – | – | – | – | 0 | 0 | Stop timer 0, and clear it to zero. |
| TMOD | ← | 1 | 0 | X | X | X | X | 0 | 1 | Set timer 0 and timer 1 to PPG output mode and select $\phi$T1 as the input clock. |
| TFFCR | ← | X | X | X | 0 | 0 | 1 | 1 | X | Enable TFF1 inversion and set TFF1 to “1”. |
| TREG0 | ← | * | * | * | * | * | * | * | * | Set the duty of TO1 to TREG0. |
| TREG1 | ← | * | * | * | * | * | * | * | * | Set the cycle of TO1 to TREG1. |
| P7CR | ← | X | X | X | X | – | – | 1 | – | Assign P71 as TO1. |
| P7FC | ← | X | X | X | X | – | – | 1 | X | |
| P6CR | ← | – | – | – | – | 1 | 1 | 1 | 1 | Assign P60 to P63 as PG0. |
| P6FC | ← | – | – | – | – | 1 | 1 | 1 | 1 | |
| PG01CR | ← | – | – | – | – | 0 | 0 | 0 | 1 | Set PG0 to 4-phase 1-step excitation mode. |
| PG0REG | ← | * | * | * | * | * | * | * | * | Set an initial value. |
| TRUN | ← | 1 | X | – | – | – | – | 1 | 1 | Start timer 0 and timer 1. |

X: Don’t care, –: No change

## 3.11 Serial Channel

The TMP93CS40/TMP93CS41 includes two serial I/O channels. Channel 0 and channel 1 select UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission).

The serial channel has the following operation modes:

- I/O interface mode (Channel 0 and 1) — Mode 0: Transmission and reception of extended I/O data and synchronizing signal (SCLK).
- UART mode (Channel 0 and 1)
  - 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 wake-up function for making the master controller start slave controllers in a serial link (multi-controller) system.

Figure 3.11.1 shows the data format for each mode.

Serial channels 0 and 1 can be used independently.

All channels operate in the same manner except for the following points. Hence only the operation of channel 0 is explained below.

Differences between channel 0 and 1

| | Channel 0 | Channel 1 |
|---|---|---|
| Pin name | TXD0 (P90), RXD0 (P91)<br>$\overline{\text{CTS0}}$ /SCLK0 (P92) | TXD1 (P93), RXD1 (P94)<br>SCLK1 (P95) |
| Handshake function | Yes | No<br>(No $\overline{\text{CTS}}$ pin) |

Note: Using the handshake function allows transmission in the units of one data format. Thus overrun error is prevented. See the section entitled handshake function for details.

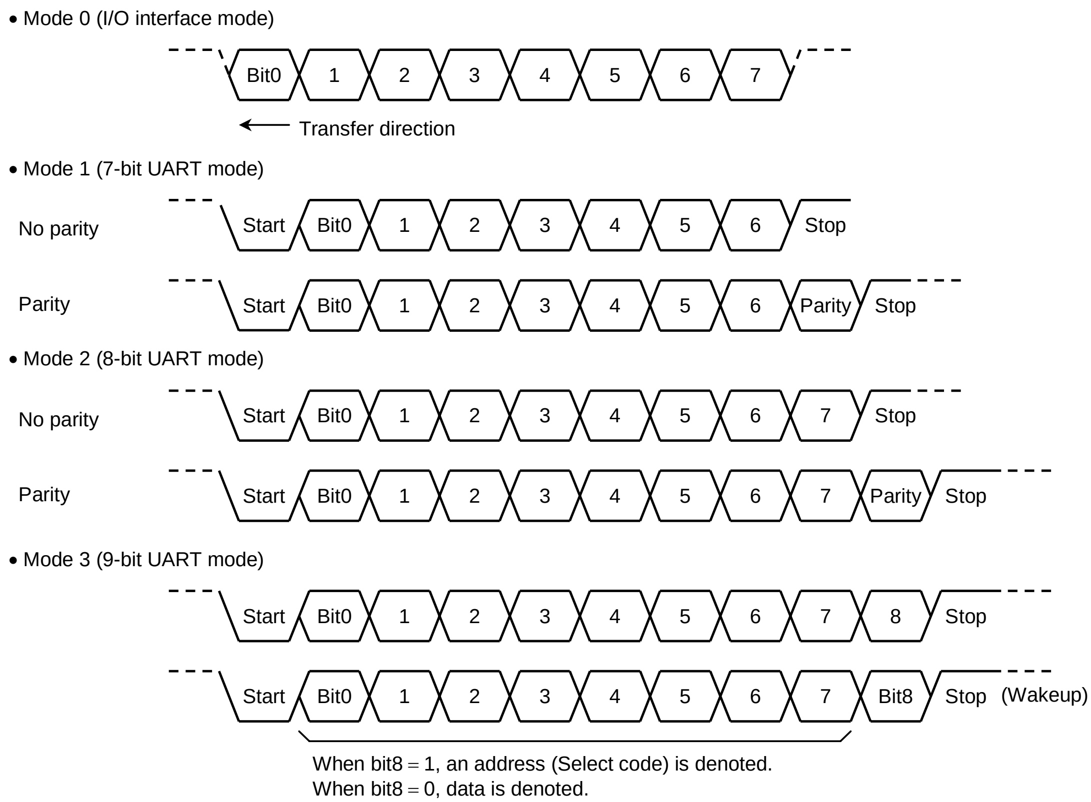



Figure 3.11.1 Data Formats

The serial channel has buffer registers for temporarily storing transmitted or received data during transmitting and receiving operations. This is so that transmitting and receiving operations can be performed independently (Full duplex).

However, in I/O interface mode, the SCLK (Serial clock) pin is used for both transmitting and receiving and the channel becomes half duplex.

The receiving data register is a double buffer structure that prevents the occurrence of overrun errors and provides a margin of one frame before the CPU reads the received data. The receiving data register stores the previously received data while the buffer register receives the next frame.

By using $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ (There is no $\overline{\text{RTS}}$ pin, so any single port must be controlled by software), it is possible to halt data transmission until the CPU finishes reading received data wherever a frame is received (The handshake function).

In UART mode, an additional check function ensures that erroneous state bits caused by noise do not cause receiving operations to start. The channel starts receiving data only when the start bit is detected properly at least twice out of three samplings of the start bit.

When the transmission buffer becomes empty and requests the CPU to send the next transmission data, or when data is stored in the receiving data register and the CPU is requested to read the data, an INTTX (Transmit interrupt) or INTRX (Receive interrupt) interrupt occurs. If an overrun error, parity error or framing error occurs during a receiving operation, the flag SC0CR<OERR, PERR, FERR> is set.

The serial channel 0/1 has a special baud rate generator, which can set any baud rate by dividing the frequency of the four clocks ($\phi$T0, $\phi$T2, $\phi$T8, and $\phi$T32) from the 9-bit prescaler (shared by the 8-bit/16-bit timers) by a value from 2 to 16.

In I/O interface mode, it is possible to input synchronous signals, as well as transmit or receive data using an external clock.

### 3.11.1 Control Registers

The serial channels are controlled by three control registers: SC0CR, SC0MOD and BR0CR. Transmitted and received data are stored in the register SC0BUF.

| SC0MOD (0052H) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | Bit symbol | TB8 | CTSE | RXE | WU | SM1 | SM0 | SC1 | SC0 |
| | Read/Write | R/W | | | | | | | |
| | After reset | Undefind | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Function | Transfer data bit8 | Handshake<br>0: CTS disable<br>1: CTS enable | Receiving function<br>0:Receive disable<br>1:Receive enable | Wake-up function<br>0: Disable<br>1: Enable | Serial transmission mode<br>00: I/O interface mode<br>01: 7-bit UART<br>10: 8-bit UART<br>11: 9-bit UART | | Serial transmission clock (UART)<br>00: TO0 trigger<br>01: Baud rate generator<br>10: Internal clock ϕ1<br>11: Don’t care | |

Serial transmission clock source (UART)

| | |
|---|---|
| 00 | Timer 0 match detect signal |
| 01 | Baud rate generator |
| 10 | Internal clock ϕ1 |
| 11 | Don’t care |

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 length |
| 10 | | 8-bit length |
| 11 | | 9-bit length |

Wake-up function

| | 9-bit UART | Other modes |
|---|---|---|
| 0 | Interrupt when data is received | Don’t care |
| 1 | Interrupt only when RB8 = 1 | |

Receiving function

| | |
|---|---|
| 0 | Receive disable |
| 1 | Receive enable |

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

| | |
|---|---|
| 0 | Disable (Always transferable) |
| 1 | Enable |

Transmission data bit8

Note: SC1MOD (56H) is on channel 1.

Figure 3.11.2 Serial Mode Control Register (Channel 0, SC0MOD)

SC0CR
(0051H)

| | 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 zero when read) | | | R/W | |
| After reset | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Received data bit8 | Parity<br>0: Odd<br>1: Even | Parity addition<br>0: Disable<br>1: Enable | 1: Error<br>Overrun | Parity | Framing | 0: SCLK0 ( ↑ )<br>1: SCLK0 ( ↓ ) | 0: Baud rate generator<br>1: SCLK0 pin input |

Select I/O interface input clock

| | |
|---|---|
| 0 | Baud rate generator |
| 1 | SCLK0 pin input |

Edge selection for SCLK pin (Input mode only) (Note 1)

| | |
|---|---|
| 0 | Transmits and receives ( ↑ ) data on rising edge of SCLK0 |
| 1 | Transmits and receives ( ↓ ) data on falling edge of SCLK0 |

Framing error flag
Parity error flag
Overrun error flag
} Cleared to zero when read.

Enable parity addition

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

Addition/check of even parity

| | |
|---|---|
| 0 | Odd parity |
| 1 | Even parity |

Receiving data bit8

Note 1: Serial control register for channel 1 is SC1CR (55H).

Note 2: As all error flags are cleared after reading, do not test a single bit only with a bit-testing instruction.

Figure 3.11.3 Serial Control Register (Channel 0, SC0CR)

BR0CR (0053H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | | BR0CK1 | BR0CK0 | BR0S3 | BR0S2 | BR0S1 | BR0S0 |
| Read/Write | R/W | | R/W | | | | | |
| After reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Always fixed to 0 | | 00: φT0<br>01: φT2<br>10: φT8<br>11: φT32 | | Divided frequency setting | | | |

Baud rate generator divided frequency setting

| | |
|---|---|
| 0000 | 16 divisions |
| 0001 | Invalid |
| 0010 to 1111 | 2 to 15 divisions |

Baud rate generator input clock selection

| | |
|---|---|
| 00 | Internal clock φT0 |
| 01 | Internal clock φT2 |
| 10 | Internal clock φT8 |
| 11 | Internal clock φT32 |

Note 1: Serial control register for channel 1 is BR1CR (57H).

Note 2: Set TRUN<PRRUN> to 1 when the baud rate generator is used.

Note 3: BR0CR<bit6> is always 1.

Note 4: Don’t read from or write to BR0CR register during sending or receiving.

Figure 3.11.4 Serial Channel Control (Channel 0, BR0CR)

SC0BUF (0050H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| TB7 | TB6 | TB5 | TB4 | TB3 | TB2 | TB1 | TB0 | (Transmission) |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | (Receiving) |

Note: Read-modify-write is prohibited for SC0BUF.

Figure 3.11.5 Serial Transmission/Receiving Buffer Registers (Channel 0, SC0BUF)

SC1MOD (0056H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | TB8 | – | RXE | WU | SM1 | SM0 | SC1 | SC0 |
| Read/Write | R/W | | | | | | | |
| After reset | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Transferred data bit8 | Always fixed to 0 | Receiving function<br>0: Receive disable<br>1: Receive enable | Wake-up function<br>0: Disable<br>1: Enable | Serial transmission mode<br>00: I/O interface mode<br>01: 7-bit UART<br>10: 8-bit UART<br>11: 9-bit UART | | Serial transmission clock (UART)<br>00: TO0 trigger<br>01: Baud rate generator<br>10: Internal clock $\phi$1<br>11: Don't care | |

Serial transmission clock source (UART)

| | |
|---|---|
| 00 | Timer 0 match detect signal |
| 01 | Baud rate generator |
| 10 | Internal clock $\phi$1 |
| 11 | Don't care |

Note: 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 length |
| 10 | | 8-bit length |
| 11 | | 9-bit length |

Wakeup function

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

Receiving control

| | |
|---|---|
| 0 | Receive disable |
| 1 | Receive enable |

Transmission data bit8

Figure 3.11.6 Serial Mode Control Register (Channel 1, SC1MOD)

SC1CR (0055H)

| | 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 zero when read) | | | R/W | |
| After reset | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Received data bit8 | Parity<br>0: Odd<br>1: Even | Parity addition<br>0: Disable<br>1: Enable | 1: Error | | | 0: SCLK1<br>1: SCLK1 | 0: Baud rate generator<br>1: SCLK1 pin input |
| | | | | Overrun | Parity | Framing | | |

Select I/O interface input clock

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

Edge selection for SCLK pin (Input mode only)

| | |
|---|---|
| 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 zero when read

Enable parity addition

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

Addition/check of even parity

| | |
|---|---|
| 0 | Odd parity |
| 1 | Even parity |

Receiving data bit8

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

Figure 3.11.7 Serial Control Register (Channel 1, SC1CR)

BR1CR (0057H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | – | | BR1CK1 | BR1CK0 | BR1S3 | BR1S2 | BR1S1 | BR1S0 |
| Read/Write | R/W | | R/W | | | | | |
| After reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | Always fixed to 0 | | 00: $\phi$T0<br>01: $\phi$T2<br>10: $\phi$T8<br>11: $\phi$T32 | | Divided frequency setting | | | |

Baud rate generator divided frequency setting

| | |
|---|---|
| 0000 | 16 divided |
| 0001 | Invalid |
| 0010 to 1111 | 2 to 15 divisions |

Baud rate generator input clock selection

| | |
|---|---|
| 00 | Internal clock $\phi$T0 |
| 01 | Internal clock $\phi$T2 |
| 10 | Internal clock $\phi$T8 |
| 11 | Internal clock $\phi$T32 |

Note 1: To use baud rate generator, set TRUN<PRRUN> to “1”, putting the prescaler in RUN.

Note 2: BR1CR<bit6> is always “1”.

Note 3: Don’t read from or write to BR1CR register during sending or receiving.

Figure 3.11.8 Baud Rate Generator Control Register (Channel 1, BR1CR)

SC1BUF (0054H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| TB7 | TB6 | TB5 | TB4 | TB3 | TB2 | TB1 | TB0 | (Transmission) |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | (Receiving) |

Note: Read-modify-write is prohibited for SC1BUF.

Figure 3.11.9 Serial Transmission/Receiving Buffer Registers (Channel 1, SC1BUF)

P9FC (001DH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | P95F | | P93F | P92F | | P90F |
| Read/Write | | | W | | W | W | | W |
| After reset | | | 0 | | 0 | 0 | | 0 |
| Function | | | 0: Port<br>1: SCLK1 | | 0: Port<br>1: TXD1 | 0: Port<br>1: SCLK0 | | 0: Port<br>1: TXD0 |

Read-modify-write is prohibited

Setting P90 as TXD0 output

| | |
|---|---|
| 0 | Port |
| 1 | TXD0 (Channel 0) output |

Setting P92 as SCLK0 output

| | |
|---|---|
| 0 | Port |
| 1 | SCLK0 (Channel 0) output |

Setting P93 as TXD1 output

| | |
|---|---|
| 0 | Port |
| 1 | TXD1 (Channel 1) output |

Setting P95 as SCLK1 output

| | |
|---|---|
| 0 | Port |
| 1 | SCLK1 (Channel 1) output |

Figure 3.11.10 Port 9 Function Register (P9FC)

ODE (0058H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | | | | | | | ODE1 | ODE0 |
| Read/Write | | | | | | | R/W | |
| After reset | | | | | | | 0 | 0 |
| Function | | | | | | | P93<br>0: CMOS<br>1: Open drain | P90<br>0: CMOS<br>1: Open drain |

Setting P90 as open-drain output

| | |
|---|---|
| 0 | CMOS output |
| 1 | Open-drain output |

Setting P93 as open-drain output

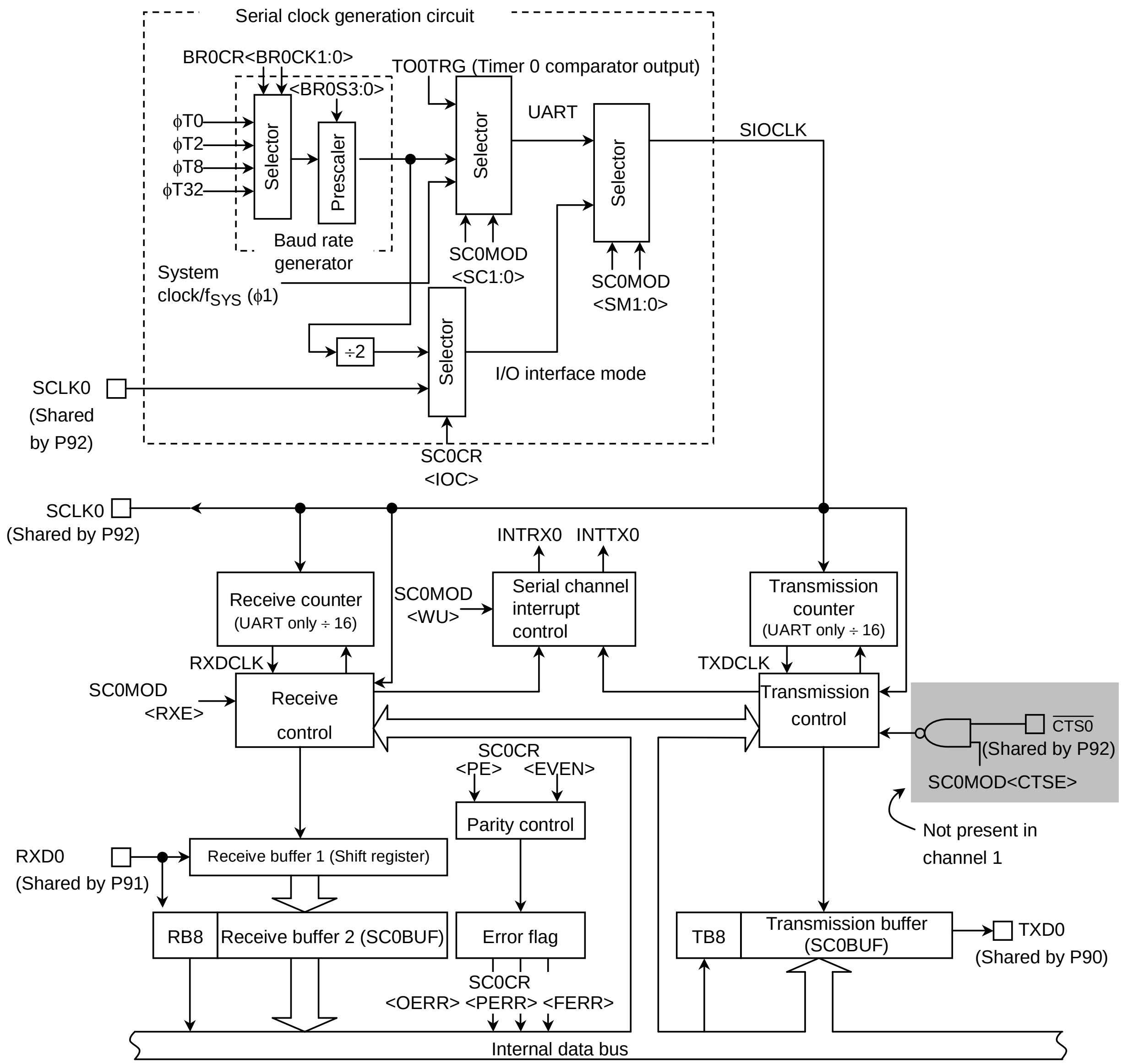
| | |
|---|---|
| 0 | CMOS output |
| 1 | Open-drain output |

Note: ODE<bit7:2> is always “1”.

Figure 3.11.11 Port 9 Open-drain Enable Register (ODE)

### 3.11.2 Configuration

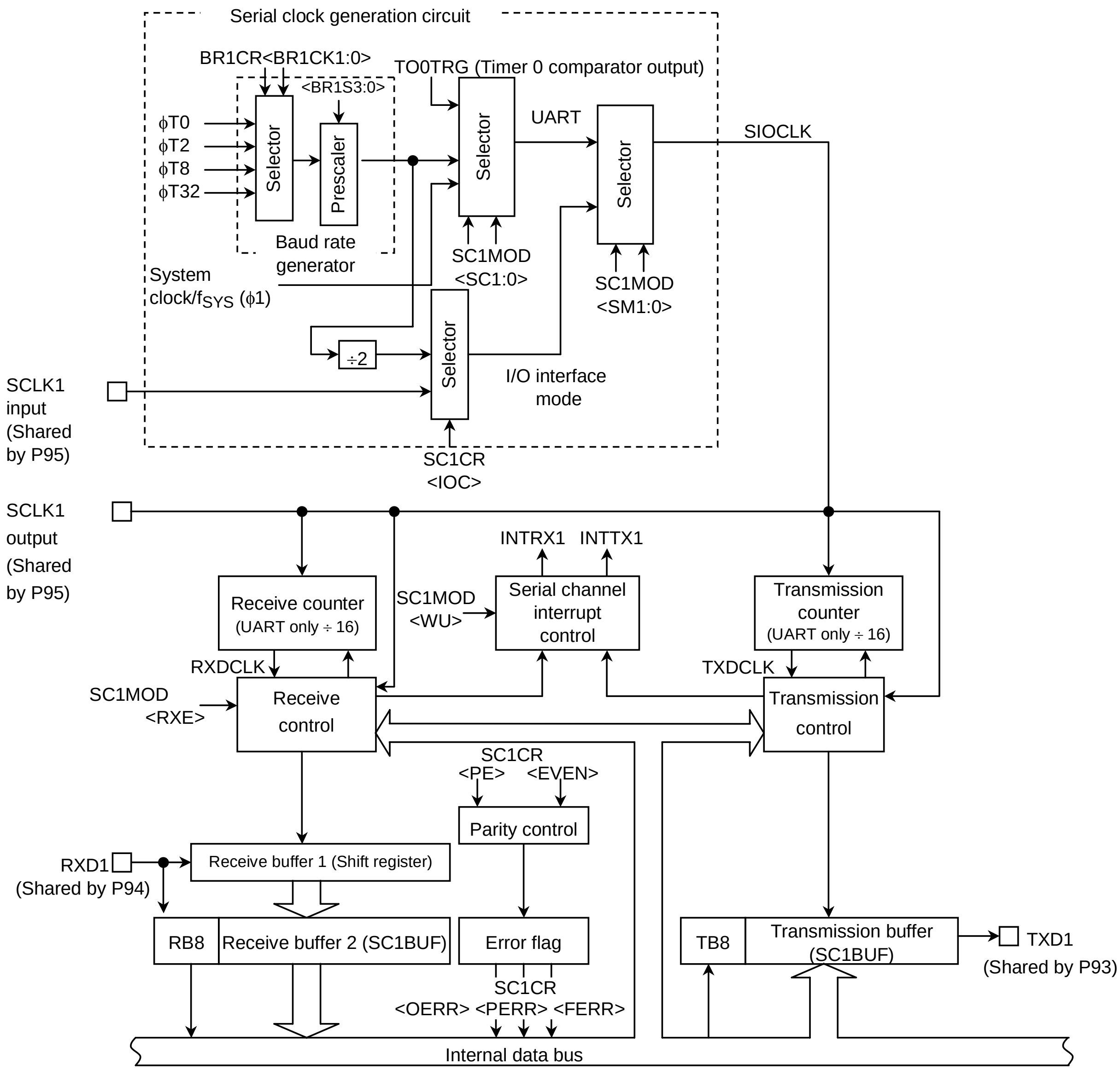
Figure 3.11.12 shows the block diagram for serial channel 0.




Note: SLCK0 pin can only be used for I/O in I/O interface mode.

Figure 3.11.12 Block Diagram of Serial Channel 0

Figure 3.11.13 shows the block diagram for serial channel 1.




Note: SLCK1 pin can only be used for I/O in I/O interface mode.

Figure 3.11.13 Block Diagram for Serial Channel 1

1. Prescaler

9-bit prescaler and prescaler clock selection registers generate input clocks for 8-bit timer 0, 1, 16-bit timer 4, 5, and serial interface 0, 1.

Figure 3.11.15 shows the block diagram. Table 3.11.1 shows how the prescaler clock is resolved into the baud rate generator.

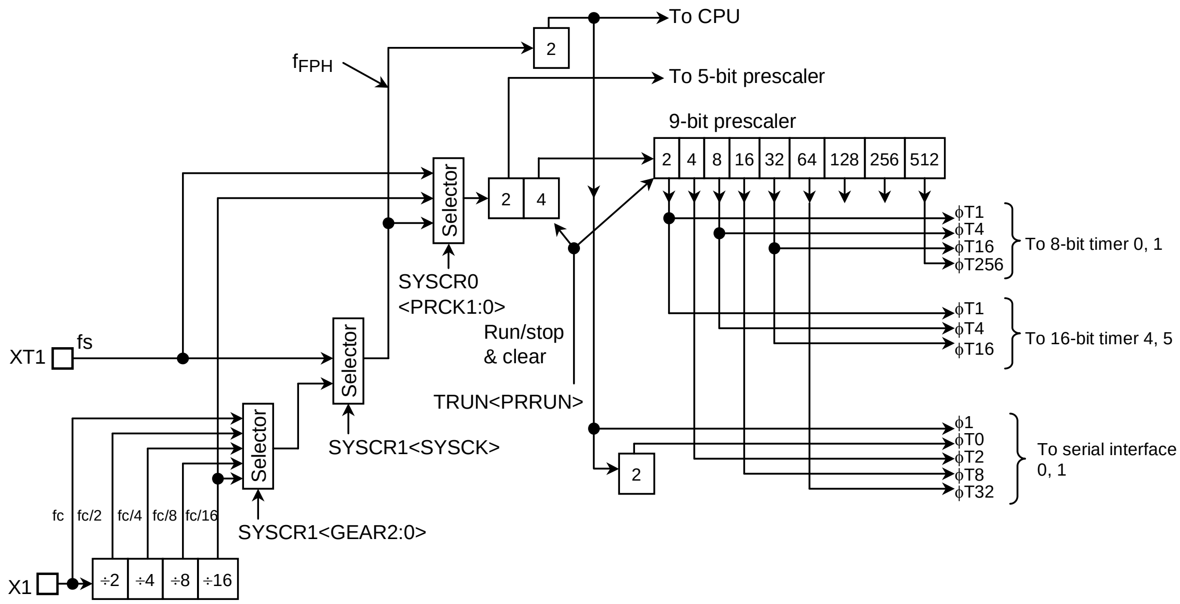



Figure 3.11.14 Block Diagram of Prescaler

Table 3.11.1 Prescaler Clock Resolution for Baud Rate Generator

| Select System Clock <SYSCK> | Select Prescaler Clock <PRCK1:0> | Gear Value <GEAR2:0> | Prescaler Output Clock Resolution | | | |
|---|---|---|---|---|---|---|
| | | | φT0 | φT2 | φT8 | φT32 |
| 1 (fs) | 00 ($f_{FPH}$) | XXX | fs/$2^2$ | fs/$2^4$ | fs/$2^6$ | fs/$2^8$ |
| 0 (fc) | | 000 (fc) | fc/$2^2$ | fc/$2^4$ | fc/$2^6$ | fc/$2^8$ |
| | | 001 (fc/2) | fc/$2^3$ | fc/$2^5$ | fc/$2^7$ | fc/$2^9$ |
| | | 010 (fc/4) | fc/$2^4$ | fc/$2^6$ | fc/$2^8$ | fc/$2^{10}$ |
| | | 011 (fc/8) | fc/$2^5$ | fc/$2^7$ | fc/$2^9$ | fc/$2^{11}$ |
| | | 100 (fc/16) | fc/$2^6$ | fc/$2^8$ | fc/$2^{10}$ | fc/$2^{12}$ |
| XXX | 01 (Low-frequency clock) | XXX | – | fs/$2^4$ | fs/$2^6$ | fs/$2^8$ |
| XXX | 10 (fc/16 clock) | XXX | – | fc/$2^8$ | fc/$2^{10}$ | fc/$2^{12}$ |

XXX: Don’t care, –: Invalid

Note: The fc/16 clock cannot be used as a prescaler clock when the fs clock is used as a system clock.

The selected clock ($f_{FPH}$ clock, fc/16 clock or fs clock) is divided by 4 and input to the prescaler. This selection is made by the prescaler clock selection register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to "00" and selects the $f_{FPH}$ clock input divided by 4.

The baud rate generator selects between 4 clock inputs: The prescaler outputs φT0, φT2, φT8, and φT32.

The prescaler can be run or stopped by the timer operation control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to "1". The prescaler is cleared to zero and stops operation when <PRRUN> is set to "0".

Resetting clears <PRRUN> to "0", clearing the prescaler and stopping operation.

When IDLE1 mode (in which only the oscillator operates) is used, set TRUN<PRRUN> to "0" to stop the prescaler before a HALT instruction is executed.

2. Baud rate generator

The baud rate generator is a circuit that generates transmission and receiving clocks to determine the transfer rate of the serial channel.

The input clock to the baud rate generator, $\phi$T0, $\phi$T2, $\phi$T8, or $\phi$T32, is generated by the 9-bit prescaler which is shared by the timers. One of these input clocks is selected by the baud rate generator control register BR0CR<BR0CK1:0>.

The baud rate generator includes a 4-bit frequency divider, which divides frequency by from 2 to 16 to determine the transfer rate.

The method for calculating a transfer rate when the baud rate generator is used is explained below.

- UART mode

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

- I/O interface mode

$$\text{Baud rate} = \frac{\text{Input clock for baud rate generator}}{\text{Frequency divisor for baud rate generator}} \div 2$$

For example, when the source clock (fc) is 12.288 MHz, the input clock is $\phi$T2 (fc/16), and the frequency divisor is 5, the transfer rate in UART mode is as follows:

* Clock configuration

$$\left\{\begin{array}{ll} \text{System clock:} & \text{High frequency (fc)} \\ \text{Clock gear:} & \text{1 (fc)} \\ \text{Prescaler clock:} & f_{FPH} \end{array}\right.$$

$$\text{Baud rate} = \frac{\text{fc/16}}{5} \div 16$$

$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \text{ (bps)}$$

Table 3.11.2 shows an example of the transfer rate in UART mode.

Also, using the 8-bit timer 0, the serial channel can generate a transfer rate. Table 3.11.3 shows examples of baud rate settings using timer 0.

Table 3.11.2 Selection of Transfer Rate (1) (when baud rate generator is used)

Unit (kbps)

| fc [MHz] | Input clock / Frequency divisor | $\phi$T0 | $\phi$T2 | $\phi$T8 | $\phi$T32 |
|---|---|---|---|---|---|
| 9.830400 | 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 |
| | 0 | 9.600 | 2.400 | 0.600 | 0.150 |
| 12.288000 | 5 | 38.400 | 9.600 | 2.400 | 0.600 |
| | A | 19.200 | 4.800 | 1.200 | 0.300 |
| 14.745600 | 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 |

Note 1: Transfer rates in I/O interface mode are 8 times faster than the values given in the above table.

Note 2: This table is calculated for when fc is selected as the system clock, the clock gear is set to fc, and the system clock is selected as the prescaler clock input.

Table 3.11.3 Selection of Transfer Rate (2) (when timer 0 (Input clock $\phi$T1) is used)

Unit (kbps)

| TREG0 / fc | 12.288 MHz | 12 MHz | 9.8304 MHz | 8 MHz | 6.144 MHz |
|---|---|---|---|---|---|
| 1H | 96 | | 76.8 | 62.5 | 48 |
| 2H | 48 | | 38.4 | 31.25 | 24 |
| 3H | 32 | 31.25 | | | 16 |
| 4H | 24 | | 19.2 | | 12 |
| 5H | 19.2 | | | | 9.6 |
| 8H | 12 | | 9.6 | | 6 |
| AH | 9.6 | | | | 4.8 |
| 10H | 6 | | 4.8 | | 3 |
| 14H | 4.8 | | | | 2.4 |

How to calculate the transfer rate (when timer 0 is used):

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by SYSCR0<PRCK1:0>}}{\text{TREG0} \times 8 \times 16}$$

(when timer 0 (Input clock $\phi$T1) is used)

Note 1: Timer 0 match detect signal cannot be used as the transfer clock in I/O interface mode.

Note 2: This table is calculated for when fc is selected as the system clock, the clock gear is set to fc, and $f_{FPH}$ is selected as the prescaler clock input.

3. Serial clock generation circuit

   This circuit generates the basic clock for transmitting and receiving data.

   - I/O interface mode

     In SCLK output mode with a setting of 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 setting of SC0CR<IOC> = “1”, the rising edge or falling edge is detected, according to the setting of the SC0CR<SCLKS> register, and used to generate the basic clock.

   - UART mode

     The setting of SC0MOD<SC1:0> selects the baud rate generator clock, internal clock $\phi$1 (Max 625 kbps at fc = 20 MHz) or the match detect signal from timer 0 as the signal from which to generate the basic clock SIOCLK.

4. Receiving counter

   The receiving counter is a 4-bit binary counter used in asynchronous communication (UART) mode and counts up with the SIOCLK clock. A duration of 16 pulses of SIOCLK are used for receiving 1 bit of data, and the data bit is sampled three times, at the 7th, 8th, and 9th clock ticks.

   Using these three samples, the received data is evaluated using the majority rule.

   For example, if the sampled data bits are respectively “1”, “0” and “1” at the 7th, 8th, and 9th clock ticks, the received data is evaluated as “1”. Sampled data “0”, “0” and 1 is evaluated to be “0”.

5. Receiving control

   - I/O interface mode

     In SCLK0 output mode with a setting of SC0CR<IOC> = “0”, the RXD0 signal will be sampled at the rising edge of the shift clock which is output to the SCLK0 pin.

     In SCLK0 input mode with a setting of SC0CR<IOC> = “1”, the RXD0 signal will be sampled at the rising edge or falling edge of the SCLK0 input, according to the setting of the SC0CR<SCLKS> register.

   - Asynchronous communication (UART) mode

     The receiving control block has a circuit for detecting the start bit using the majority rule. When two or more 0’s are detected out of 3 samples, it is recognized as a start bit and the receiving operation is started.

     The data being received is also evaluated using the majority rule.

6. Receiving buffer

To prevent overrun errors, the receiving buffer has a double buffer structure.

Received data is stored bit by bit in receiving buffer 1 (Shift register type). When 7 bits or 8 bits of data are stored in receiving buffer 1, the stored data is transferred to receiving buffer 2 (SC0BUF), generating an interrupt INTRX0. The CPU reads only receiving buffer 2 (SC0BUF). Even before the CPU has read receiving buffer 2 (SC0BUF), more received data can be stored in receiving buffer 1. However, unless receiving buffer 2 (SC0BUF) is read before all the 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> are still preserved.

The parity bit added in 8-bit UART mode and the most significant bit (MSB) in 9-bit UART mode are stored in SC0CR<RB8>.

In 9-bit UART mode, the wakeup function for the slave controller is enabled by setting SC0MOD<WU> to "1"; and interrupt INTRX0 occurs only when SC0CR<RB8> is set to 1.

7. Transmission counter

The transmission counter is a 4-bit binary counter which is used in Asynchronous communication (UART) mode and, like a receiving counter, counts using the SIOCLK clock. This generates a TXDCLK pulse every 16 clock pulses.

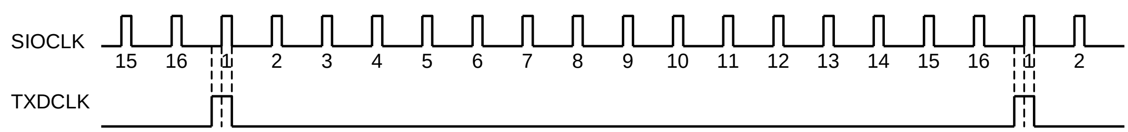



Figure 3.11.15 Generation of Transmission Clock

8. Transmission controller

- I/O interface mode

  In SCLK0 output mode with a setting of SC0CR<IOC> = 0, the data in the transmission buffer is output bit by bit to the TXD0 pin at the rising edge of the shift clock which is output from the SCLK0 pin.

  In SCLK0 Input mode with a setting of SC0CR<IOC> = 1, the data in the transmission buffer is output bit by bit to the TXD0 pin at the rising edge or falling edge of the SCLK0 input, according to the setting of the SC0CR<SCLKS> register.

- Asynchronous communication (UART) mode

  When transmission data is written to the transmission buffer from the CPU, transmission starts at the rising edge of the next TXDCLK pulse.

Handshake function

Serial channel 0 has a $\overline{\text{CTS0}}$ pin. Using this pin, data can be sent in units of one frame; thus, overrun errors can be avoided. The handshake function is enabled/disabled by SC0MOD<CTSE>.

When the $\overline{\text{CTS0}}$ pin goes high, after completion of the current data transmission, data transmission is halted until the $\overline{\text{CTS0}}$ pin goes low again. When the INTTX0 interrupt is generated, it requests the next data transmission to the CPU.

Although there is no $\overline{\text{RTS}}$ pin, a handshake function can easily be configured by assigning any port to the $\overline{\text{RTS}}$ function. The $\overline{\text{RTS}}$ output should be high to request a halt to data transmission after data receive is completed by software in the RXD interrupt routine.

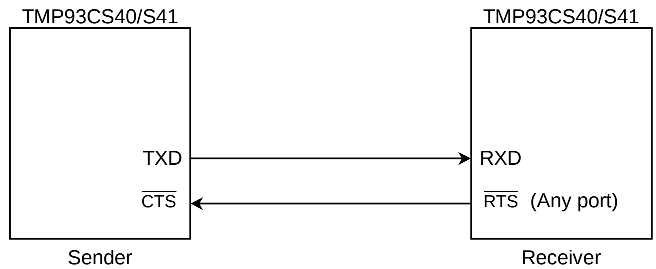
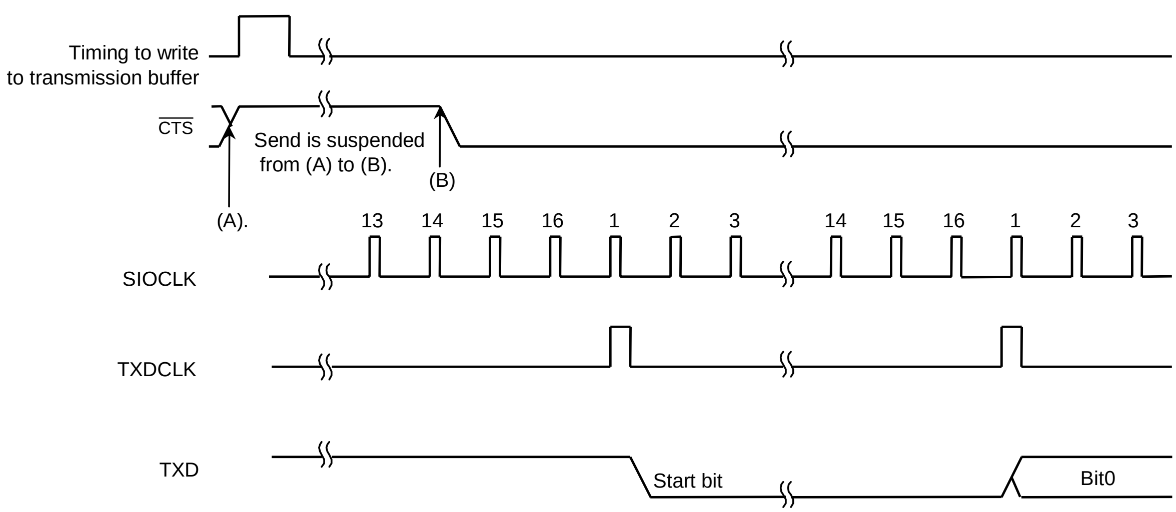



Figure 3.11.16 Handshake Function




Note 1: If the $\overline{\text{CTS}}$ signal rises during transmission, the next datum is not sent after completion of the current transmission.

Note 2: Transmission starts at the first falling edge of the TXDCLK clock after the $\overline{\text{CTS}}$ signal falls.

Figure 3.11.17 Timing of $\overline{\text{CTS}}$ (Clear to send)

9. Transmission buffer

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

10. Parity control circuit

   When the serial channel control register SC0CR<PE> is set to 1, it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART or 8-bit UART modes. The SC0CR<EVEN> register can select even or odd parity.

   During transmission, parity is automatically generated according to the data written to the transmission buffer SC0BUF. The data are transmitted after the parity bit is stored in SC0BUF<TB7> in 7-bit UART mode, or in SC0MOD<TB8> in 8-bit UART mode. <PE> and <EVEN> must be set before the transmission data is written to the transmission buffer.

   During receiving, data are shifted to the receiving buffer 1 and the parity is automatically set after the data are transferred to the receiving buffer 2 (SC0BUF). The parity bit is then compared with SC0BUF<RB7> in 7-bit UART mode, or with SC0MOD<RB8> in 8-bit UART mode. If the bits do not match, a parity error occurs and the SC0CR<PERR> flag is set.

11. Error flag

   Three error flags are provided to increase the reliability of data reception.

   1) Overrun error <OERR>

      If all bits of the next datum are received in receiving buffer 1 while valid data is stored in receiving buffer 2 (SC0BUF), an overrun error occurs.

   2) Parity error <PERR>

      The parity generated for the data shifted into receiving buffer 2 (SC0BUF) is compared with the parity bit received from the RXD pin. If they are not equal, a parity error occurs.

   3) Framing error <FERR>

      The stop bit of the received data is sampled three times in the center of the pulse. If the majority of the sampled values are “0”, a framing error occurs.

12. Signal generating timing

1) In UART mode

Receive

| Mode | 9 Bits | 8 Bits + Parity | 8 Bits, 7 Bits + Parity, 7 Bits |
|---|---|---|---|
| Timing for interrupt generation | Around center of bit8 | Around center of parity bit | Around center of stop bit |
| Timing for framing generation | Around center of stop bit | Around center of stop bit | Around center of stop bit |
| Timing for parity error generation | − | Around center of parity bit | ← |
| Timing for overrun error timing | Around center of bit8 | Around center of parity bit | Around center of stop bit |

Note: In 9-Bit and 8-Bit + 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.

Send

| Mode | 9 Bits | 8 Bits + Parity | 8 Bits, 7 Bits + Parity, 7 Bits |
|---|---|---|---|
| Timing for interrupt generation | Immediately before stop bit sent | ← | ← |

2) In I/O interface mode

| | | |
|---|---|---|
| Timing for send interrupt generation | SCLK0 output mode | Immediately after rise of last SCLK0 signal (See Figure 3.9.20) |
| | SCLK0 input mode | Immediately after rise (Rising mode) or fall (Falling mode) of last SCLK0 signal (See Figure 3.9.21) |
| Timing for receive interrupt generation | SCLK0 output mode | Immediately after final SCLK0 (when received data are transferred to receive buffer 2 (SC0BUF)) (See Figure 3.9.22) |
| | SCLK0 input mode | Immediately after final SCLK0 (when received data are transferred to receive buffer 2 (SC0BUF)) (See Figure 3.9.23) |

### 3.11.3 Operational Description

(1) Mode 0 (I/O interface mode)

This mode is used to increase the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode encompasses the SCLK output mode for outputting a synchronous clock SCLK and the SCLK input mode for inputting an external synchronous clock SCLK.

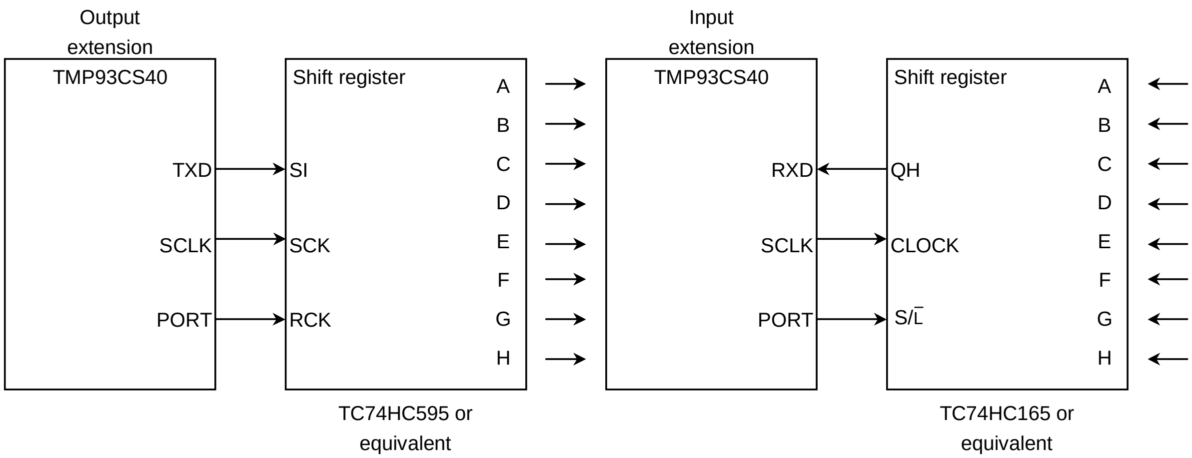



Figure 3.11.18 Example of SCLK Output Mode Connection

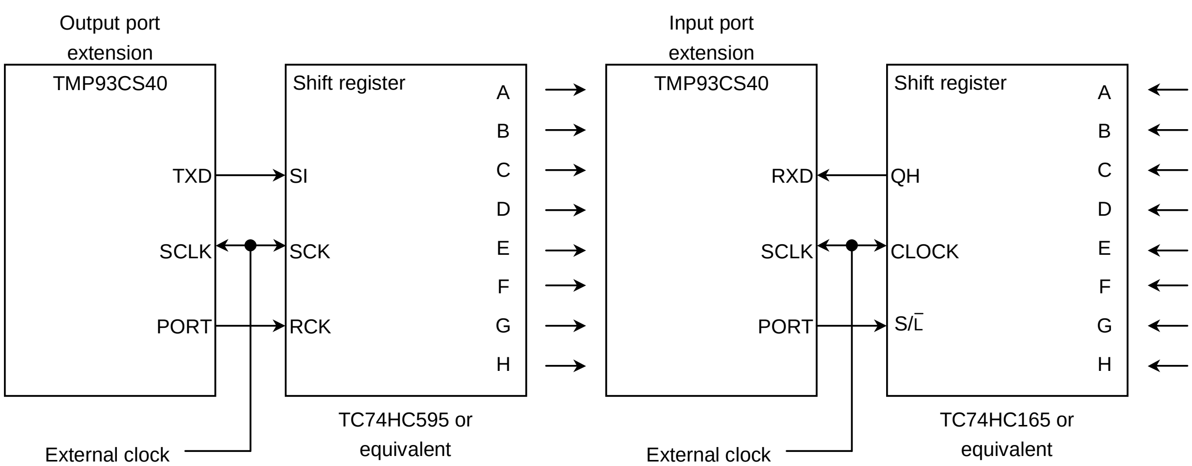



Figure 3.11.19 Example of SCLK Input Mode Connection

1. Transmission

In SCLK output mode, 8-bit data and the synchronous clock are output from the TXD0 pin and SCLK0 pin respectively, each time the CPU writes data to the transmission buffer. When all data has been output, INTES0<ITX0C> will be set, generating the INTTX0 interrupt.

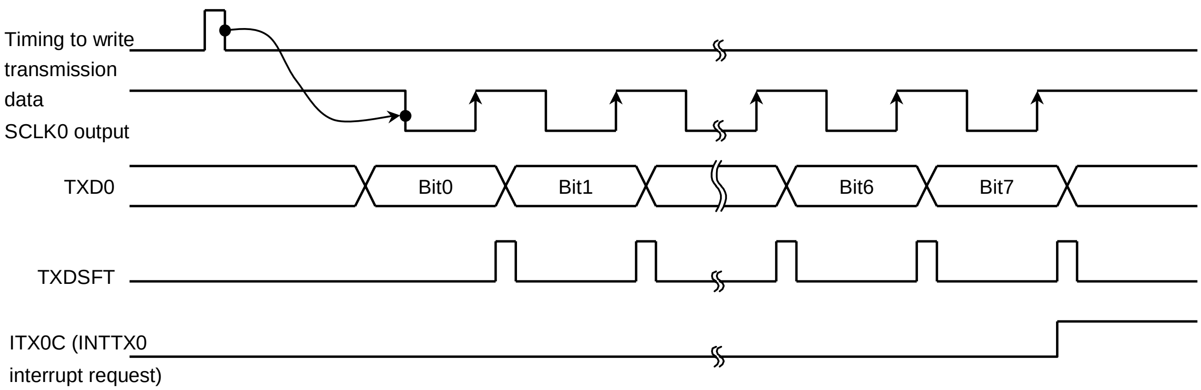



Figure 3.11.20 Transmitting Operation in I/O Interface Mode (SCLK output mode) (Channel 0)

In SCLK input mode, 8-bit data are output from the TXD0 pin when SCLK0 input becomes active after data are written to the transmission buffer by the CPU.

When all data have been output, INTES0<ITX0C> will be set, generating the INTTX0 interrupt.

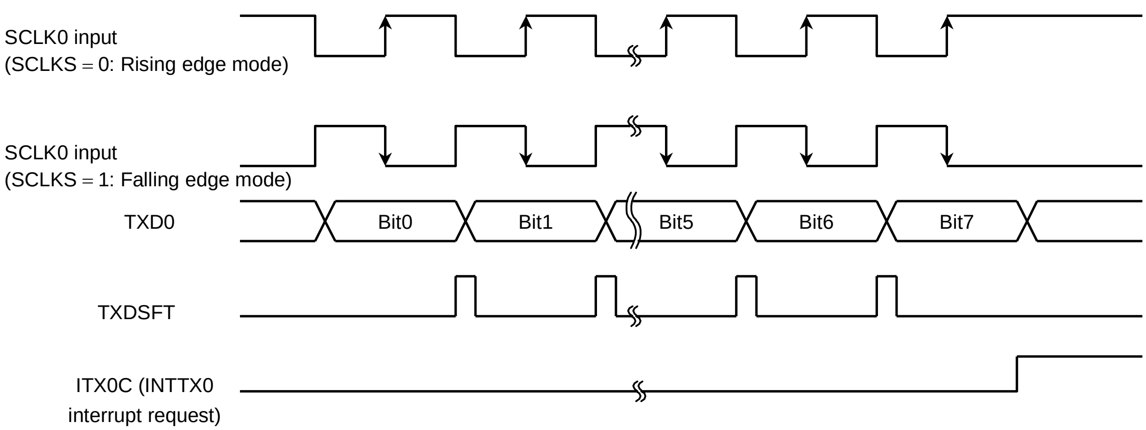



Figure 3.11.21 Transmitting Operation in I/O Interface Mode (SCLK input mode) (Channel 0)

2. Receiving

In SCLK output mode, the synchronous clock is output from the SCLK0 pin and data are shifted into the receiving buffer 1 whenever the receive interrupt flag INTES0<IRX0C> is cleared by a read of the received data. When 8-bit data is received, the data will be transferred to receiving buffer 2 (SC0BUF) according to the timing shown below and INTES0<IRX0C> will be set again, generating an INTRX0 interrupt.

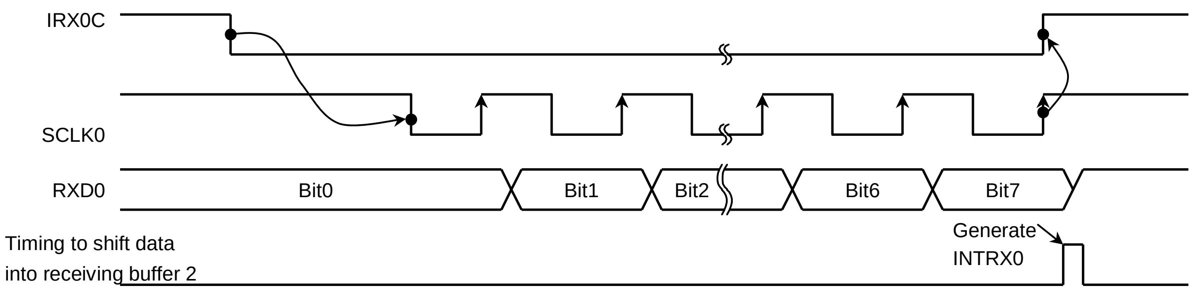



Figure 3.11.22 Receiving Operation in I/O Interface Mode (SCLK0 output mode) (Channel 0)

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

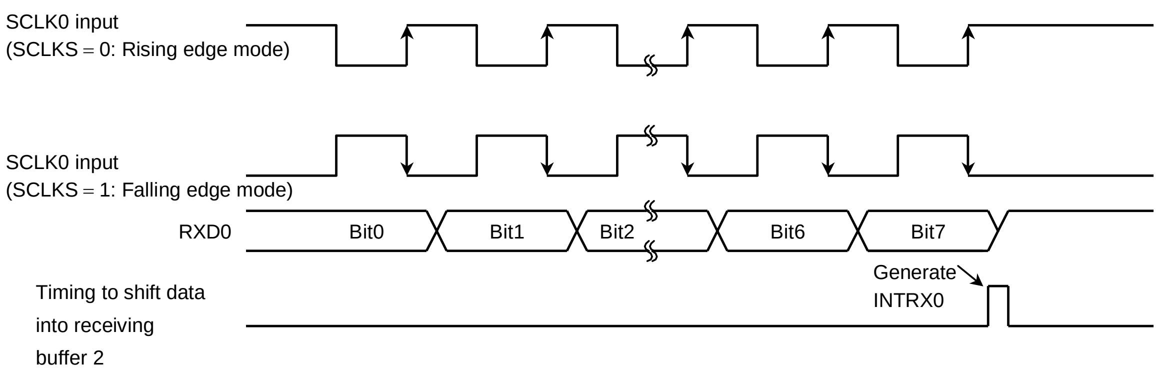



Figure 3.11.23 Receiving Operation in I/O Interface Mode (SCLK input mode) (Channel 0)

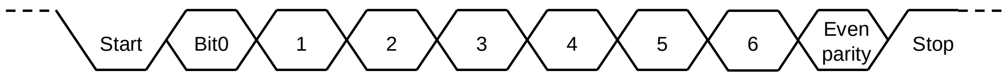
Note: When receiving data, the system must be put in receive enable state (SC0MOD<RXE> = 1).

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

7-bit mode can be selected by setting the serial channel mode register SC0MOD<SM1:0> to 01.

In this mode, a parity bit can be added. The parity bit is enabled or disabled by serial channel control register SC0CR<PE>. Even parity or odd parity is selected using SC0CR<EVEN> when <PE> is set to 1 (Enable).

Setting example: When transmitting data with the following format, the control registers should be set as described below. Channel 0 is explained here.




Direction of transmission (Transmission rate: 2400 bps at fc = 12.288 MHz)

* Clock configuration
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: System clock

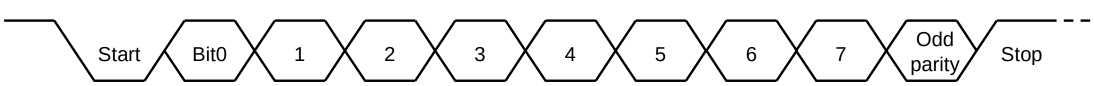
| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|
| P9CR | ← | X | X | – | – | – | – | – | 1 | Select P90 as the TXD0 pin. |
| P9FC | ← | X | X | – | X | – | X | X | 1 | |
| SC0MOD | ← | X | 0 | – | X | 0 | 1 | 0 | 1 | Set 7-bit UART mode. |
| SC0CR | ← | X | 1 | 1 | X | X | X | 0 | 0 | Set even parity. |
| BR0CR | ← | 0 | X | 1 | 0 | 0 | 1 | 0 | 1 | Set transfer rate at 2400 bps. |
| TRUN | ← | 1 | X | – | – | – | – | – | – | Start the prescaler for the baud rate generator. |
| INTES0 | ← | 1 | 1 | 0 | 0 | – | – | – | – | Enable INTTX0 interrupt and set interrupt level 4. |
| SC0BUF | ← | * | * | * | * | * | * | * | * | Set data for transmission. |

X: Don't care, –: No change

(3) Mode 2 (8-bit UART mode)

8-bit UART mode can be selected by setting SC0MOD<SM1:0> to 10. In this mode, a parity bit can be added. The parity bit is enabled or disabled by SC0CR<PE>. Even parity or odd parity is selected by using SC0CR<EVEN> when <PE> is set to 10 (Enable).

Setting example: When receiving data with the following format, the control registers should be set as described below.




Direction of transmission (Transmission rate: 9600 bps at fc = 12.288 MHz)

* Clock configuration
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: System clock

Main setting

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P9CR ← | X | X | – | – | – | – | 0 | – | Select P91 (RXD0) as the input pin. |
| SC0MOD ← | – | 0 | 1 | X | 1 | 0 | 0 | 1 | Enable receiving in 8-bit UART mode. |
| SC0CR ← | X | 0 | 1 | X | X | X | 0 | 0 | Set odd parity. |
| BR0CR ← | 0 | X | 0 | 1 | 0 | 1 | 0 | 1 | Set transfer rate at 9600 bps. |
| TRUN ← | 1 | X | – | – | – | – | – | – | Start the prescaler for the baud rate generator. |
| INTES0 ← | – | – | – | – | 1 | 1 | 0 | 0 | Enable INTTX0 interrupt and set interrupt level 4. |

Interrupt processing

Acc ← SC0CR AND 00011100
if Acc ≠ 0 then ERROR } Check for error.
Acc ← SC0BUF — Read the received data.

X: Don't care, –: No change

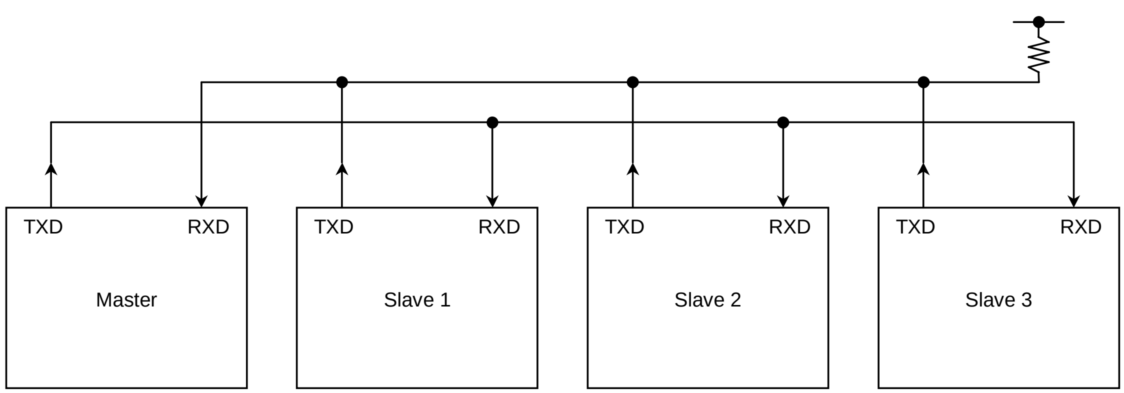
(4) Mode 3 (9-bit UART mode)

9-bit UART mode can be selected by setting SC0MOD<SM1:0> to 11. In this mode, a parity bit cannot be set.

During transmission, the MSB (9th bit) is written to the serial channel mode register <TB8>. During receiving it is stored in the serial channel control register <RB8>. When the buffer is written or read, the MSB is read or written first, then the rest of the data is read from or written to SC0BUF.

Wakeup function

In 9-bit UART mode, the wakeup function for slave controllers is enabled by setting SC0MOD<WU> to 1. The interrupt INTRX0 occurs only when<RB8> = 1.

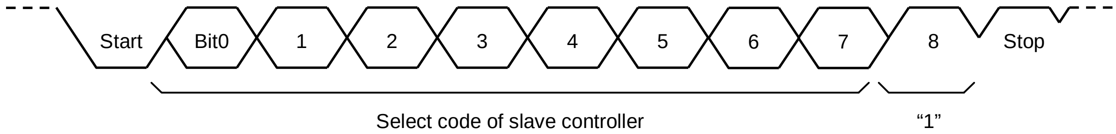



Note: The TXD pins of the slave controllers must be in open-drain output mode.

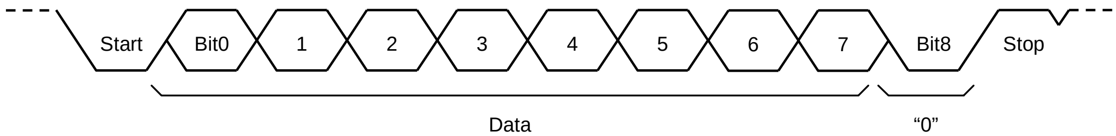
Figure 3.11.24 Serial Link Using Wakeup Function

Protocol

1. Select 9-bit UART mode for the master and slave controllers.
2. Set the SC0MOD<WU> bit of 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".



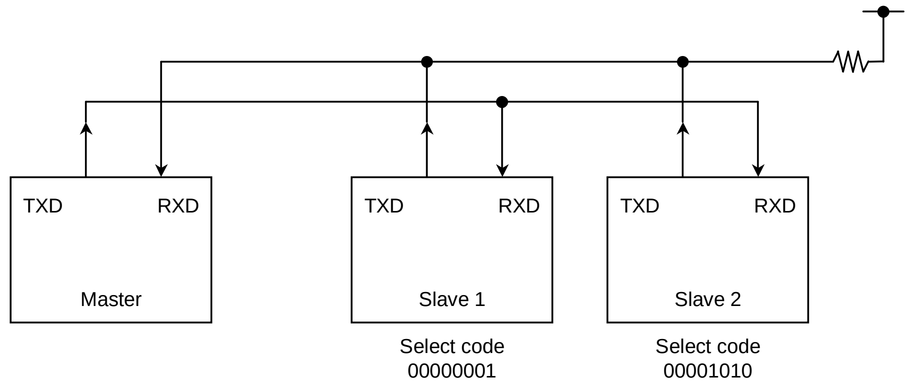

4. Each slave controller receives the above frame and clears its WU bit to "0" if the above select code matches its own select code.
5. The master controller transmits data to the specified slave controller (the one whose SC0MOD<WU> bit is cleared to "0"). The MSB (bit8)<TB8> is cleared to "0".




6. The other slave controllers (those with their <WU> bits remaining at 1) ignore the receiving data because their MSBs (bit8 or <RB8>) are set to "0" to disable the INTRX0 interrupt.

   Slave controllers (with WU = 0) can transmit data to the master controller. In this way they can indicate the end of data reception to the master controller.

Setting example: To link two slave controllers serially with the master controller, and use the internal clock $\phi1$ as the transfer clock.




Since serial channels 0 and 1 operate in exactly the same way, channel 0 only is used for the purposes of this explanation.

- Setting the master controller

Main

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| P9CR | ← | X | X | – | – | – | – | 0 | 1 | Select P90 as TXD0 pin and P91 as RXD0 pin. |
| P9FC | ← | X | X | – | X | – | X | X | 1 | |
| INTES0 | ← | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | Enable INTTX0 and set interrupt level 4.<br>Enable INTRX0 and set interrupt level 5. |
| SC0MOD | ← | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | Set $\phi1$ as the transmission clock in 9-bit UART mode. |
| SC0BUF | ← | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Set the select code for slave controller 1. |

INTTX0 interrupt

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| SC0MOD | ← | 0 | – | – | – | – | – | – | – | Sets TB8 to 0. |
| SC0BUF | ← | * | * | * | * | * | * | * | * | Set data for transmission. |

- Setting the slave controller

Main

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| P9CR | ← | X | X | – | – | – | – | 0 | 1 | Select P91 as RXD0 pin and P90 as TXD0 pin (Open-drain output). |
| P9FC | ← | X | X | – | X | – | X | X | 1 | |
| ODE | ← | X | X | X | X | X | X | – | 1 | |
| INTES0 | ← | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | Enable INTRX0 and INTTX0. |
| SC0MOD | ← | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | Set <WU> to 1 in 9-bit UART transmission mode with transfer clock $\phi1$. |

INTRX0 interrupt

Acc ← SC0BUF
if Acc = Select code
Then SC0MOD ← – – – 0 – – – – Clear <WU> to 0.

## 3.12 Analog/Digital Converter

The TMP93CS40/S41 contains an analog/digital converter (AD converter) with 8-channel analog input that features 10-bit successive approximation.

Figure 3.12.1 shows the block diagram for the AD converter. 8-channel analog input pins (AN7 to AN0) are shared by the input only port P5 which can also be used as a general-purpose input port.

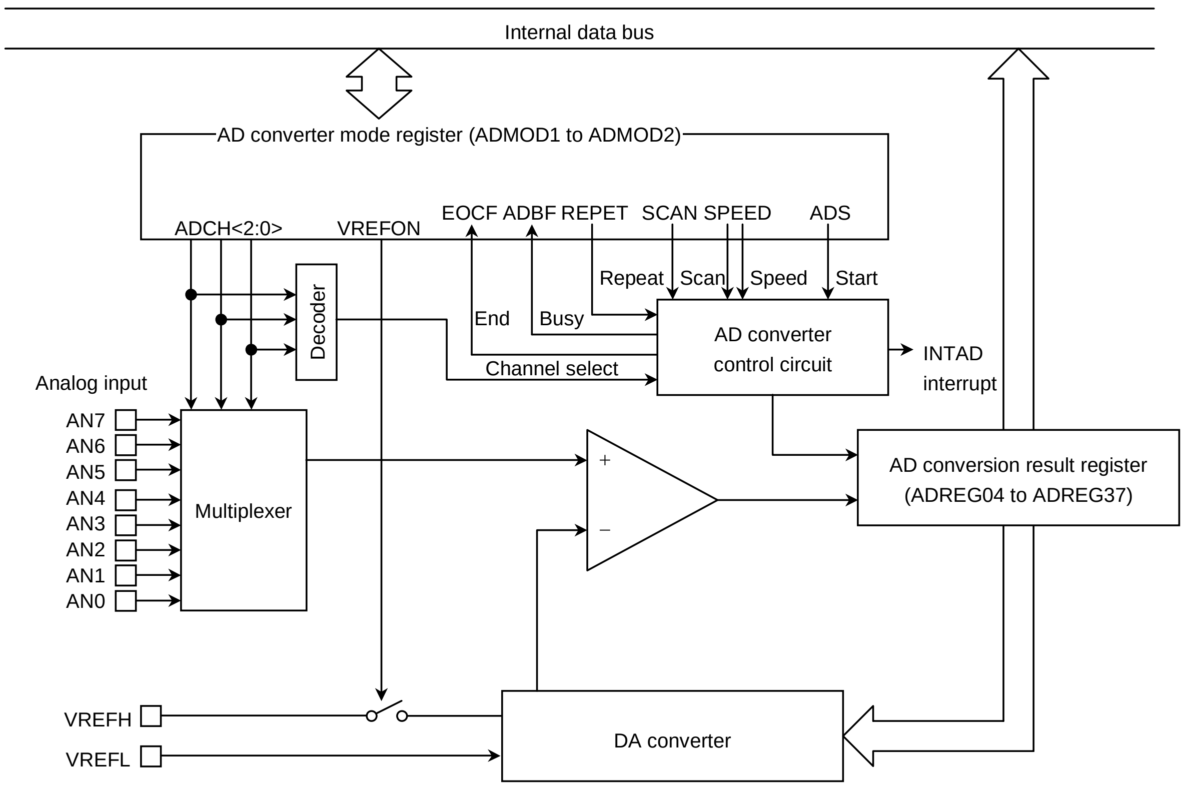



Figure 3.12.1 Block Diagram for AD Converter

Note 1: This AD converter does not have a built-in sample and hold circuit.

Note 2: When the power supply current is reduced in IDLE2, IDLE1 or STOP mode, stop operation of the AD converter before the HALT instruction is executed. And set ADMOD2<SPEED1:0> = “00”.

Note 3: The operation of the AD converter is guaranteed only when fc (The high-frequency oscillator) is used (It is not guaranteed when fs is used). It is guaranteed when $f_{FPH} \geq 4$ MHz.

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ADMOD1 (005EH) | Bit symbol | EOCF | ADBF | REPET | SCAN | | ADS | | |
| | Read/Write | R | | R/W | | | R/W | | |
| | After reset | 0 | 0 | 0 | 0 | | 0 | | |
| | Function | AD conversion END flag 1: End | AD conversion BUSY flag 1: Busy | Repeat mode 0: Single mode 1: Repeat mode | Scan mode 0: Fixed channel mode 1: Channel scan mode | | AD conversion start 1: START Note: Always "0" is read. | | |

AD conversion start

| | |
|---|---|
| 0 | – |
| 1 | Start conversion |

Note: Always "0" when data is read

AD scan mode

| | |
|---|---|
| 0 | Fixed channel mode |
| 1 | Channel scan mode |

AD repeat mode

| | |
|---|---|
| 0 | Single mode |
| 1 | Repeat mode |

AD BUSY flag

| | |
|---|---|
| 0 | Not busy |
| 1 | Busy |

AD END flag

| | |
|---|---|
| 0 | Not end |
| 1 | End |

Figure 3.12.2 AD Control Register (1/2)

ADMOD2 (005FH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | VREFON | | SPEED1 | SPEED0 | | ADCH2 | ADCH1 | ADCH0 |
| Read/Write | R/W | | R/W | | | R/W | | |
| After reset | 1 | | 0 | 0 | | 0 | 0 | 0 |
| Function | String resistance ON/OFF | | Conversion speed<br>00: 160 states<br>01: 320 states<br>10: 640 states<br>11: 1280 states | | | Analog input channel selection | | |

Analog input channel selection

| <ADCH2:0> \ <SCAN> | 0 | 1 |
|---|---|---|
| 000 | AN0 | AN0 |
| 001 | AN1 | AN0 → AN1 |
| 010 | AN2 | AN0 → AN1 → AN2 |
| 011 | AN3 | AN0 → AN1 → AN2 → AN3 |
| 100 | AN4 | AN4 |
| 101 | AN5 | AN4 → AN5 |
| 110 | AN6 | AN4 → AN5 → AN6 |
| 111 | AN7 | AN4 → AN5 → AN6 → AN7 |

Conversion speed

| | |
|---|---|
| 00 | 160 states (16 μs at 20 MHz) |
| 01 | 320 states (32 μs at 20 MHz) |
| 10 | 640 states (64 μs at 20 MHz) |
| 11 | 1280 states (128 μs at 20 MHz) |

String resistance ON/OFF selection

| | |
|---|---|
| 0 | String resistance OFF |
| 1 | String resistance ON |

Note: Set the <VREFON> register to "1" before starting conversion (setting <ADS> to "1").

Figure 3.12.3 AD Control Register (2/2)

ADREG04L (0060H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR01 | ADR00 | | | | | | |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Lower 2 bits of AD result for AN0 or AN4 are stored. | | | | | | | |

ADREG04H (0061H)

| | 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 | Upper 8 bits of AD result for AN0 or AN4 are stored. | | | | | | | |

ADREG15L (0062H)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | ADR11 | ADR10 | | | | | | |
| Read/Write | R | | | | | | | |
| After reset | Undefined | | | | | | | |
| Function | Lower 2 bits of AD result for AN1 or AN5 are stored. | | | | | | | |

ADREG15H (0063H)

| | 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 | Upper 8 bits of AD result for AN1 or AN5 are stored. | | | | | | | |

Note: The result registers are used both as AN0 and AN4, AN1 and AN5, AN2 and AN6, and AN3 and AN7. They are stored in ADREG04, 15, 26, and 37.

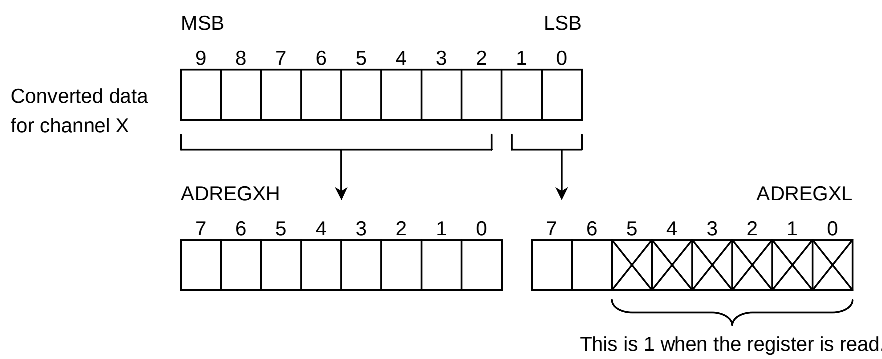



Figure 3.12.4 AD Conversion Result Register (ADREG04, ADREG15) (1/2)

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ADREG26L | Bit symbol | ADR21 | ADR20 | | | | | | |
| (0064H) | Read/Write | R | | | | | | | |
| | After reset | Undefined | | | | | | | |
| | Function | Lower 2 bits of AD result for AN2 or AN6 are stored. | | | | | | | |

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ADREG26H | Bit symbol | ADR29 | ADR28 | ADR27 | ADR26 | ADR25 | ADR24 | ADR23 | ADR22 |
| (0065H) | Read/Write | R | | | | | | | |
| | After reset | Undefined | | | | | | | |
| | Function | Upper 8 bits of AD result for AN2 or AN6 are stored. | | | | | | | |

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ADREG37L | Bit symbol | ADR31 | ADR30 | | | | | | |
| (0066H) | Read/Write | R | | | | | | | |
| | After reset | Undefined | | | | | | | |
| | Function | Lower 2 bits of AD result for AN3 or AN7 are stored. | | | | | | | |

| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| ADREG37H | Bit symbol | ADR39 | ADR38 | ADR37 | ADR36 | ADR35 | ADR34 | ADR33 | ADR32 |
| (0067H) | Read/Write | R | | | | | | | |
| | After reset | Undefined | | | | | | | |
| | Function | Upper 8 bits of AD result for AN3 or AN7 are stored. | | | | | | | |

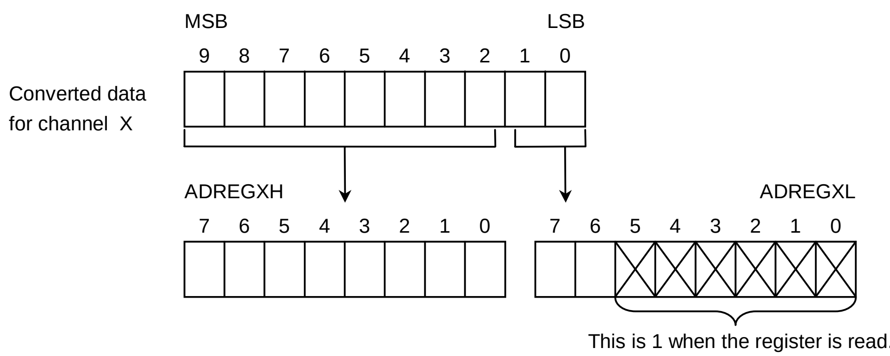



Figure 3.12.5 AD Conversion Result Register (ADREG26, ADREG37) (2/2)

### 3.12.1 Operation

(1) Analog reference voltage

The high analog reference voltage is applied to the VREFH pin, and the low analog reference voltage is applied to the VREFL pin.

The reference voltage between VREFH and VREFL is divided by 1024 (using string resistance) and compared with the analog input voltage for AD conversion.

The switch between VREFH and VREFL can be turned off by writing “0” to ADMOD2<VREFON>.

When <VRFFON> = “0”, before the conversion can start, must be written to <VREFON> and a 3 μs period must be allowed so that the internal reference voltage can stabilize (regardless of fc) before “1” is written to ADMOD1<ADS>.

(2) Analog input channels

The analog input channel is selected by ADMOD2<ADCH2:0>. However, the channel which should be selected depends on the operation mode of the AD converter.

In fixed analog input mode, one channel is selected out of eight pins, AN0 to AN7, by <ADCH2:0>.

In analog input channel scan mode, the number of channels to be scanned is specified by ADMOD2<ADCH2:0>. For example, AN0 only, AN0 → AN1, AN0 → AN1 → AN2, AN0 → AN1 → AN2 → AN3, AN4 only, AN4 → AN5, AN4 → AN5 → AN6, or AN4 → AN5 → AN6 → AN7.

When reset the AD conversion channel register will be initialized to ADMOD2<ADCH2:0> = “000”, so that the AN0 pin is selected.

The pins which are not used as analog input channels can be used as ordinary input port pins for port P5.

(3) Starting AD conversion

AD conversion starts when “1” is written to the AD conversion register ADMOD1<ADS>. When conversion starts, the conversion busy flag ADMOD1<ADBF>, which indicates that conversion is in progress, is set to “1”.

(4) AD conversion mode

Both fixed AD conversion channel mode and conversion channel scan mode include two conversion modes; single and repeat conversion mode.

In fixed channel repeat mode, conversion of the specified single channel is executed repeatedly.

In scan repeat mode, scanning is executed repeatedly.

The AD conversion mode is selected by ADMOD1<REPET, SCAN>.

(5) AD conversion speed selection

There are four AD conversion speed modes. The selection is made by the ADMOD2<SPEED1:0> register.

When reset, <SPEED1:0> is initialized to “00”, selecting 160-state conversion mode (16 μs at 20 MHz).

(6) AD conversion end and interrupt

- AD conversion single mode

When AD conversion of the specified channel has finished (in fixed channel conversion mode) or when AD conversion of the last channel has finished (in channel scan mode), ADMOD<EOCF> is set to "1", the ADMOD<ADBF> flag is reset to "0", and the INTAD interrupt is generated.

- AD conversion repeat mode

For both fixed conversion channel mode and conversion channel scan mode, INTAD should be disabled in repeat mode. Always set INTE0AD to "000", to disable the interrupt request.

Write 0 to ADMOD2<REPET> to terminate repeat mode. Repeat mode will be exited as soon as the conversion in progress is completed.

(7) Storing the AD conversion result

The results of AD conversion are stored in the registers ADREG04, ADREG15, ADREG26, ADREG37 for each channel. The result registers are used as AN0 and AN4, AN1 and AN5, AN2 and AN6, and AN3 and AN7.

However, the contents of the registers do not indicate which channel's data has been converted.

In repeat mode, the registers are updated as soon as conversion ends.

ADREG04 to ADREG37 are read only registers.

(8) Reading the AD conversion result

The results of AD conversion are stored in the registers ADREG04 to ADREG37.

When the lowest two bits of one of the registers ADREG04L, ADREG15L, ADREG26L, or ADREG37L are read, ADMOD1<EOCF> is cleared to "0".

<EOCF> is not cleared to "0" when the upper eight bits of one of the registers ADREG04H, ADREG15H, ADREG26H, or ADREG37H are read.

Setting example: 1. When the analog input voltage on the AN3 pin is AD converted at 160-state speed and the result is transferred to the memory address 0100H by the AD interrupt INTAD routine.

Main setting

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| INTE0AD | ← 1 | 1 | 0 | 0 | – | – | – | – | | Enable INTAD and set interrupt level 4. |
| ADMOD2 | ← 1 | X | 0 | 0 | X | 0 | 1 | 1 | | Specify AN3 pin as an analog input channel and start |
| ADMOD1 | ← X | X | 0 | 0 | X | 1 | X | X | | AD conversion in 160-state speed mode. |

INTAD routine

| | | | |
|---|---|---|---|
| WA | ← | ADREG37 | Read ADREG37L and ADREG37H values and write to WA (16 bits). |
| WA | >> | 6 | Right-shift WA six times and write "0" in upper bits. |
| (000100H) | ← | WA | Write contents of WA in memory at 0100H. |

2. When the analog input voltage of the four pins AN4 to AN7 are AD converted at 320-state speed and the channel is set to scan and repeat mode.

Main setting

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| INTE0AD | ← 1 | 0 | 0 | 0 | – | – | – | – | | Disable INTAD. |
| ADMOD2 | ← 1 | X | 0 | 1 | 0 | 1 | 1 | 1 | | Specify AN4 to AN7 pins as input channel, select scan |
| ADMOD1 | ← X | X | 1 | 1 | X | 1 | 0 | 0 | | and repeat mode and start AD conversion. |

X: Don't care, –: No change

## 3.13 Watchdog Timer (Runaway Detection Timer) and Warm-up Timer

The TMP93CS40/S41 contains a watchdog timer for runaway detection.

The watchdog timer (WDT) is used to return the CPU to a 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 to notify the CPU of the malfunction, and outputs "0" from the watchdog timer out pin $\overline{\text{WDTOUT}}$ to notify peripheral devices of the malfunction.

Connecting the watchdog timer output to the reset pin internally forces a reset.

The watchdog timer consists of 7-stage and 15-stage binary counters.

These binary counters are also used as a warm-up timer for internal oscillator stabilization. This is used for releasing stop and also before changing the system clock.

### 3.13.1 Configuration

Figure 3.13.1 shows the block diagram for the watchdog timer (WDT).

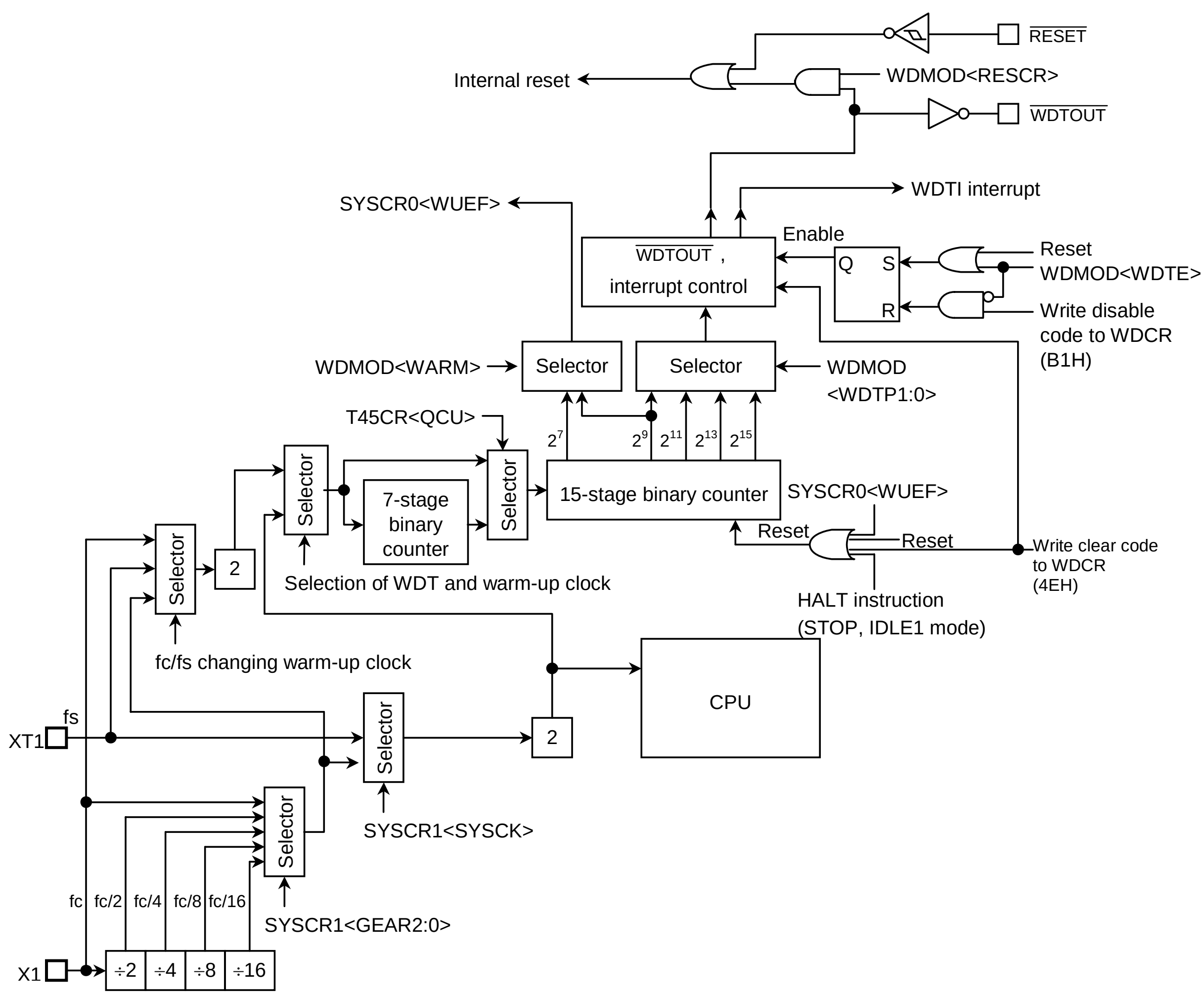



Figure 3.13.1 Block Diagram for Watchdog Timer/Warm-up Timer

The watchdog timer consists of 7-stage and 15-stage binary counters which use the system clock ($f_{SYS}$) as the input clock. The 15-stage binary counter has $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/2^{19}$ and $f_{SYS}/2^{21}$ outputs. Selecting one of the outputs with the WDMOD<WDTP1:0> register generates a watchdog interrupt and outputs watchdog timer out when an overflow occurs.

For the warm-up counter, either a $2^7$ or a $2^9$ output from the 15-stage binary counter can be selected using the WDMOD<WARM> register. When a stable-external oscillator is used, a shorter warm-up time is available using the T45CR<QCU> register. When <QCU> = "1", a counting value $2^7$ is selected.

When the watchdog timer is in operation, this shorter warm-up time function cannot be selected. The warm-up counter function can be made available by setting <QCU> = "0".

Since the watchdog timer out pin ($\overline{\text{WDTOUT}}$) outputs "0" when there is a watchdog timer overflow, the peripheral devices can be reset. The watchdog timer out pin is set to "1" after WDT has been disabled and the watchdog timer cleared (by a clear code 4EH being written into the WDCR register).

Example:

```
LDW   (WDMOD), B100H    ; Disable
LD    (WDCR), 4EH       ; Write clear code
SET   7, (WDMOD)        ; Enable again
```

In other words, $\overline{\text{WDTOUT}}$ keeps outputting "0" until the clear code is written.

The watchdog timer out pin can also be connected to the reset pin internally. In this case, the watchdog timer out pin ($\overline{\text{WDTOUT}}$) outputs "0" for 8 to 20 states (12.8 to 32 μs at fc = 20 MHz), and then resets itself.

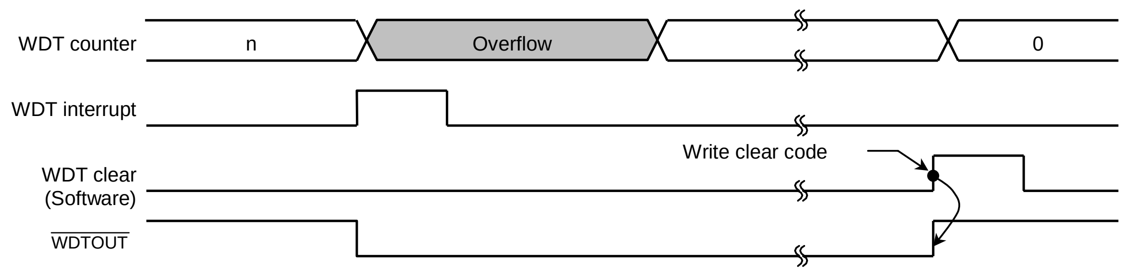



Figure 3.13.2 Normal Mode

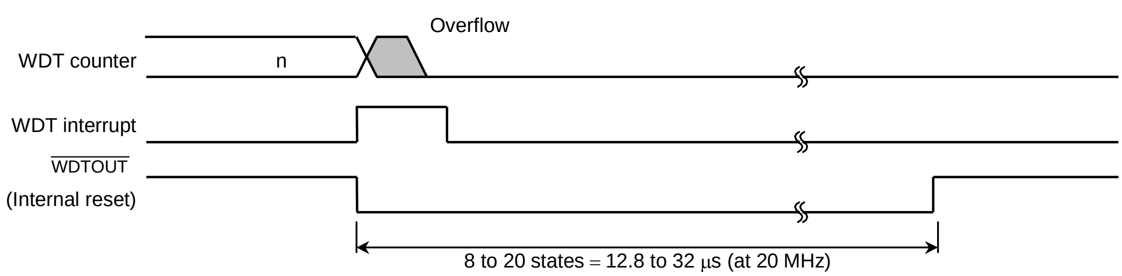



Figure 3.13.3 Reset Mode

### 3.13.2 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 <WDTP>

   This 2-bit register is used to set the watchdog timer interrupt time for detecting runaway. This register is initialized to WDMOD<WDTP1:0> = “00” when reset.

   The detection time for WDT is shown in Figure 3.13.6.

2. Watchdog timer enable/disable control <WDTE>

   When reset, WDMOD<WDTE> is initialized to “1” to enable the watchdog timer.

   To disable the timer, it is necessary to clear this bit to “0” and write the disable code (B1H) into 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 from the disabled state to the enabled state simply by setting <WDTE> to “1”.

3. Watchdog timer out reset connection<RESCR>

   This bit is used to connect the output of the watchdog timer with $\overline{\text{RESET}}$ internally. Since WDMOD<RESCR> is initialized to “0” at reset, a watchdog timer reset is not performed.

(2) Watchdog timer control register (WDCR)

This register is used to disable and clear the watchdog timer’s binary counter.

- Disable control

The watchdog timer can be disabled by writing the disable code (B1H) to the WDCR register after clearing WDMOD<WDTE> to “0”.

```
┌ WDMOD   ← 0  –  –  –  –  –  X  X    Clear WDMOD<WDTE> to "0".
└ WDCR    ← 1  0  1  1  0  0  0  1    Write the disable code (B1H).
```

- Enable control

This sets WDMOD<WDTE> to “1”.

- Watchdog timer clear control

The binary counter can be cleared and made to resume counting by writing the clear code (4EH) into the WDCR register.

```
WDCR    ← 0  1  0  0  1  1  1  0    Write the clear code (4EH).
```

WDMOD (005CH)

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit symbol | WDTE | WDTP1 | WDTP0 | WARM | HALTM1 | HALTM0 | RESCR | DRVE |
| Read/Write | R/W | | | | | | | |
| After reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Function | WDT control<br>1: Enable | Select detection time<br>00: $2^{15}/f_{SYS}$<br>01: $2^{17}/f_{SYS}$<br>10: $2^{19}/f_{SYS}$<br>11: $2^{21}/f_{SYS}$<br>See Figure 3.13.6 | | Warm-up time | Standby mode<br>00: RUN mode<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE2 mode | | 1: Internally connects WDT out to the reset pin | 1: Drives the pin even in STOP mode |

DRVE (Explanation in STOP mode section)

Watchdog timer out control

| | |
|---|---|
| 0 | – |
| 1 | Connects WDT out to reset |

Select the standby mode HALT instruction

| | |
|---|---|
| 00 | RUN mode (only the CPU stops) |
| 01 | STOP mode (All circuits stop) |
| 10 | IDLE1 mode (only the oscillator operates) |
| 11 | IDLE2 mode (CPU and AD stop) |

Selection of warm-up time at fc = 20 MHz, fs = 32.768 kHz

| System Clock Selection <SYSCK> | Gear Value <GEAR2:0> | Warm-up Time | | |
|---|---|---|---|---|
| | | T45CR<QCU> = 0 | | T45CR<QCU> = 1 |
| | | WARM = 0 | WARM = 1 | WARM = X |
| 1 (fs) | XXX (Don’t care) | 0.50 s | 2.00 s | 3.9 ms |
| 0 (fc) | 000 (fc) | 0.82 ms | 3.28 ms | 6.4 μs |
| | 001 (fc/2) | 1.64 ms | 6.55 ms | 12.8 μs |
| | 010 (fc/4) | 3.28 ms | 13.11 ms | 25.6 μs |
| | 011 (fc/8) | 6.55 ms | 26.21 ms | 51.2 μs |
| | 100 (fc/16) | 13.11 ms | 52.43 ms | 102.4 μs |

Watchdog timer enable/disable control

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

Figure 3.13.4 Watchdog Timer Mode Register

WDCR (005DH)

| | 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 | | | | | | | |

Disable/clear WDT

| | |
|---|---|
| B1H | Disable code |
| 4EH | Clear code |
| Others | Don't set |

Figure 3.13.5 Watchdog Timer Control Register

at fc = 20 MHz, fs = 32.768 kHz

| System Clock Selection <SYSCK> | Gear Value <GEAR2:0> | Watchdog Timer Detecting Time WDMOD<WDTP1:0> 00 | 01 | 10 | 11 |
|---|---|---|---|---|---|
| 1 (fs) | XXX (Don't care) | 2.00 s | 8.00 s | 32.00 s | 128.0 s |
| 0 (fc) | 000 (fc) | 3.28 ms | 13.11 ms | 52.43 ms | 209.72 ms |
| | 001 (fc/2) | 6.55 ms | 26.24 ms | 104.86 ms | 419.43 ms |
| | 010 (fc/4) | 13.11 ms | 52.43 ms | 209.72 ms | 838.86 ms |
| | 011 (fc/8) | 26.21 ms | 104.86 ms | 419.43 ms | 1.68 s |
| | 100 (fc/16) | 52.43 ms | 209.72 ms | 838.86 ms | 3.36 s |

Note: When using the register as the watchdog timer, write 0 to the T45CR<QCU> bit.

Figure 3.13.6 Watchdog Timer Detection Time

### 3.13.3 Operation

The watchdog timer generates interrupt INTWD after the detection time set in the WDMOD<WDTP1:0> and T45CR<QCU> registers has elapsed, and outputs a low level signal. The watchdog timer must be cleared to zero by software before an INTWD interrupt is generated. If the CPU malfunctions (Undergoes runaway) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter overflows and an INTWD interrupt is generated. The CPU detects malfunction (Runaway) from the INTWD interrupt and it is possible to return to normal operation using a recovery program. If the watchdog timer out pin is connected to the reset pins of peripheral devices, a CPU malfunction can also be acknowledged by these other devices.

The watchdog timer restarts operation immediately after reset is released.

The watchdog timer does not operate in IDLE1 or STOP mode. In RUN mode, the watchdog timer is operational. When the bus is released ( $\overline{\text{BUSAK}}$ = L), WDT continues counting.

However, the function can be disabled when entering RUN or IDLE2 mode. The watchdog timer is enabled in IDLE2 mode, but the overflow interrupt is disabled. Disable the watchdog timer before entering IDLE2 mode.

Example: 1. Clear the binary counter.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| WDCR | ← | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | Write the clear code (4EH). |

2. Set the watchdog timer detecting time to $2^{18}/f_{SYS}$.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| WDMOD | ← | 1 | 0 | 1 | – | – | – | X | X | |

3. Disable the watchdog timer.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| WDMOD | ← | 0 | – | – | – | – | – | X | X | Clear WDTE to “0”. |
| WDCR | ← | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | Write disable code (B1H). |

4. Set IDLE1 mode.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| WDMOD | ← | 0 | – | – | – | 1 | 0 | X | X | Disable WDT and set IDLE1 mode. |
| WDCR | ← | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | |
| Executes halt command | | | | | | | | | | Set standby mode. |

5. Set the STOP mode (Warm-up time: $2^{16}/f_{SYS}$)

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| WDMOD | ← | – | – | – | 1 | 0 | 1 | X | X | Set the STOP mode. |
| Executes halt command | | | | | | | | | | Execute HALT instruction. |
| | | | | | | | | | | Set standby mode. |

# 4. Electrical Characteristics

"X" used in an expression shows a frequency for the clock $f_{FPH}$ selected by SYSCR1<SYSCK>. The value of X changes according to whether a clock gear or a low speed oscillator is selected. An example value is calculated for fc, with gear = 1/fc (SYSCR1<SYSCK, GEAR2:0> = 0000).

## 4.1 Maximum Ratings

(TMP93CS40F, TMP93CS41F
TMP93CS40DF, TMP93CS41DF)

| Parameter | Symbol | Rating | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$ | −0.5 to 6.5 | V |
| Input voltage | $V_{IN}$ | −0.5 to $V_{CC}$ + 0.5 | V |
| Output current (Total) | $\Sigma I_{OL}$ | 120 | mA |
| Output current (Total) | $\Sigma I_{OH}$ | −80 | mA |
| Power dissipation (Ta = 85°C) | $P_D$ | 600 | mW |
| Soldering temperature (10 s) | $T_{SOLDER}$ | 260 | °C |
| Storage temperature | $T_{STG}$ | −65 to 150 | °C |
| Operating temperature | $T_{OPR}$ | −40 to 85 | °C |

Note: The maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no maximum rating value will ever be exceeded.

## 4.2 DC Characteristics (1/2)

Ta = −40 to 85°C

| Parameter | | Symbol | Condition | | Min | Typ. (Note) | Max | Unit |
|---|---|---|---|---|---|---|---|---|
| Power supply voltage ($AV_{CC} = V_{CC}$, $AV_{CC} = V_{SS}$ = 0 V) | | $V_{CC}$ | fc = 4 to 20 MHz | fs = 30 to 34 kHz | 4.5 | | 5.5 | V |
| | | | fc = 4 to 12.5 MHz | | 2.7 | | | |
| Input low voltage | AD0 to AD15 | $V_{IL}$ | $V_{CC}$ ≥ 4.5 V | | −0.3 | | 0.8 | V |
| | | | $V_{CC}$ < 4.5 V | | | | 0.6 | |
| | Port 2 to port A (except P87) | $V_{IL1}$ | $V_{CC}$ = 2.7 to 5.5 V | | | | 0.3 $V_{CC}$ | |
| | $\overline{RESET}$, $\overline{NMI}$, INT0 | $V_{IL2}$ | | | | | 0.25 $V_{CC}$ | |
| | $\overline{EA}$, AM8/$\overline{AM16}$ | $V_{IL3}$ | | | | | 0.3 | |
| | X1 | $V_{IL4}$ | | | | | 0.2 $V_{CC}$ | |
| Input high voltage | AD0 to AD15 | $V_{IH}$ | $V_{CC}$ ≥ 4.5 V | | 2.2 | | $V_{CC}$ + 0.3 | |
| | | | $V_{CC}$ < 4.5 V | | 2.0 | | | |
| | Port 2 to port A (except P87) | $V_{IH1}$ | $V_{CC}$ = 2.7 to 5.5 V | | 0.7 $V_{CC}$ | | | |
| | $\overline{RESET}$, $\overline{NMI}$, INT0 | $V_{IH2}$ | | | 0.75 $V_{CC}$ | | | |
| | $\overline{EA}$, AM8/$\overline{AM16}$ | $V_{IH3}$ | | | $V_{CC}$ − 0.3 | | | |
| | X1 | $V_{IH4}$ | | | 0.8 $V_{CC}$ | | | |
| Output low voltage | | $V_{OL}$ | $I_{OL}$ = 1.6 mA ($V_{CC}$ = 2.7 to 5.5 V) | | | | 0.45 | V |
| Output high voltage | | $V_{OH1}$ | $I_{OH}$ = −400 μA ($V_{CC}$ = 3 V ± 10%) | | 2.4 | | | |
| | | $V_{OH2}$ | $I_{OH}$ = −400 μA ($V_{CC}$ = 5 V ± 10%) | | 4.2 | | | |

Note: Typical values are for Ta = 25°C and $V_{CC}$ = 5 V unless otherwise noted.

## 4.2 DC Characteristics (2/2)

| Parameter | Symbol | Condition | Min | Typ. (Note 1) | Max | Unit |
|---|---|---|---|---|---|---|
| Darlington drive current (8 output pins max) | $I_{DAR}$ (Note 2) | $V_{EXT}$ = 1.5 V<br>$R_{EXT}$ = 1.1 kΩ<br>(when $V_{CC}$ = 5 V ± 10%) | −1.0 | | −3.5 | mA |
| Input leakage current | $I_{LI}$ | 0.0 ≤ $V_{IN}$ ≤ $V_{CC}$ | | 0.02 | ±5 | μA |
| Output leakage current | $I_{LO}$ | 0.2 ≤ $V_{IN}$ ≤ $V_{CC}$ − 0.2 | | 0.05 | ±10 | μA |
| Powerdown voltage (at stop, RAM backup) | $V_{STOP}$ | $V_{IL2}$ = 0.2 $V_{CC}$,<br>$V_{IH2}$ = 0.8 $V_{CC}$ | 2.0 | | 6.0 | V |
| $\overline{RESET}$ pull-up resistor | $R_{RST}$ | $V_{CC}$ = 5 V ± 10% | 50 | | 150 | kΩ |
| | | $V_{CC}$ = 3 V ± 10% | 80 | | 200 | |
| Pin capacitance | $C_{IO}$ | fc = 1 MHz | | | 10 | pF |
| Schmitt width $\overline{RESET}$, $\overline{NMI}$, INT0 | $V_{TH}$ | | 0.4 | 1.0 | | V |
| Programmable pull-down resistor | $R_{KL}$ | $V_{CC}$ = 5 V ± 10% | 10 | | 80 | kΩ |
| | | $V_{CC}$ = 3 V ± 10% | 30 | | 150 | |
| Programmable pull-up resistor | $R_{KH}$ | $V_{CC}$ = 5 V ± 10% | 50 | | 150 | |
| | | $V_{CC}$ = 3 V ± 10% | 100 | | 300 | |
| NORMAL (Note 3) | $I_{CC}$ | $V_{CC}$ = 5 V ± 10%<br>fc = 20 MHz | | 19 | 25 | mA |
| NORMAL2 (Note 4) | | | | 24 | 30 | |
| RUN | | | | 17 | 25 | |
| IDLE2 | | | | 10 | 15 | |
| IDLE1 | | | | 3.5 | 5 | |
| NORMAL (Note 3) | | $V_{CC}$ = 3 V ± 10%<br>fc = 12.5 MHz<br>(Typ: $V_{CC}$ = 3.0 V) | | 6.5 | 10 | mA |
| NORMAL2 (Note 4) | | | | 9.5 | 13 | |
| RUN | | | | 5.0 | 9 | |
| IDLE2 | | | | 3.0 | 5 | |
| IDLE1 | | | | 0.8 | 1.5 | |
| SLOW (Note 3) | | $V_{CC}$ = 3 V ± 10%<br>fs = 32.768 kHz<br>(Typ: Vcc = 3.0 V) | | 20 | 35 | μA |
| RUN | | | | 16 | 30 | |
| IDLE2 | | | | 10 | 20 | |
| IDLE1 | | | | 5 | 15 | |
| STOP | | $V_{CC}$ = 2.7 to 5.5 V | | 0.2 | 10 | μA |

Note 1: Typical values are for Ta = 25°C and $V_{CC}$ = 5 V unless otherwise noted.

Note 2: $I_{DAR}$ is guaranteed for up to eight ports.

Note 3: $I_{CC}$ measurement conditions (NORMAL, SLOW):
Only CPU is operational; output pins are open and input pins are fixed.

Note 4: $I_{CC}$ measurement conditions (NORMAL2):
All functions are operational; output pins are open and input pins are fixed.

## 4.3 AC Characteristics

(1) $V_{CC} = 5\ V \pm 10\%$

| No. | Parameter | Symbol | Variable | | 16 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | Min | Max | |
| 1 | Osc. period (= X) | $t_{OSC}$ | 50 | 31250 | 62.5 | | 50 | | ns |
| 2 | CLK pulse width | $t_{CLK}$ | 2x – 40 | | 85 | | 60 | | ns |
| 3 | A0 to A23 valid → CLK hold | $t_{AK}$ | 0.5x – 20 | | 11 | | 5 | | ns |
| 4 | CLK valid → A0 to A23 hold | $t_{KA}$ | 1.5x – 70 | | 24 | | 5 | | ns |
| 5 | A0 to A15 valid → ALE fall | $t_{AL}$ | 0.5x – 15 | | 16 | | 10 | | ns |
| 6 | ALE fall → A0 to A15 hold | $t_{LA}$ | 0.5x – 20 | | 11 | | 5 | | ns |
| 7 | ALE high pulse width | $t_{LL}$ | x – 40 | | 23 | | 10 | | ns |
| 8 | ALE fall → $\overline{RD}$ / $\overline{WR}$ fall | $t_{LC}$ | 0.5x – 25 | | 6 | | 0 | | ns |
| 9 | $\overline{RD}$ / $\overline{WR}$ rise → ALE rise | $t_{CL}$ | 0.5x – 20 | | 11 | | 5 | | ns |
| 10 | A0 to A15 valid → $\overline{RD}$ / $\overline{WR}$ fall | $t_{ACL}$ | x – 25 | | 38 | | 25 | | ns |
| 11 | A0 to A23 valid → $\overline{RD}$ / $\overline{WR}$ fall | $t_{ACH}$ | 1.5x – 50 | | 44 | | 25 | | ns |
| 12 | $\overline{RD}$ / $\overline{WR}$ rise → A0 to A23 hold | $t_{CA}$ | 0.5x – 25 | | 6 | | 0 | | ns |
| 13 | A0 to A15 valid → D0 to D15 input | $t_{ADL}$ | | 3.0x – 55 | | 133 | | 95 | ns |
| 14 | A0 to A23 valid → D0 to D15 input | $t_{ADH}$ | | 3.5x – 65 | | 154 | | 110 | ns |
| 15 | $\overline{RD}$ fall → D0 to D15 input | $t_{RD}$ | | 2.0x – 60 | | 65 | | 40 | ns |
| 16 | $\overline{RD}$ low pulse width | $t_{RR}$ | 2.0x – 40 | | 85 | | 60 | | ns |
| 17 | $\overline{RD}$ rise → D0 to D15 hold | $t_{HR}$ | 0 | | 0 | | 0 | | ns |
| 18 | $\overline{RD}$ rise → A0 to A15 output | $t_{RAE}$ | x – 15 | | 48 | | 35 | | ns |
| 19 | $\overline{WR}$ low pulse width | $t_{WW}$ | 2.0x – 40 | | 85 | | 60 | | ns |
| 20 | D0 to D15 valid → $\overline{WR}$ rise | $t_{DW}$ | 2.0x – 55 | | 70 | | 45 | | ns |
| 21 | $\overline{WR}$ rise → D0 to D15 hold | $t_{WD}$ | 0.5x – 15 | | 16 | | 10 | | ns |
| 22 | A0 to A23 valid → $\overline{WAIT}$ input (1 + N) WAIT mode | $t_{AWH}$ | | 3.5x – 90 | | 129 | | 85 | ns |
| 23 | A0 to A15 valid → $\overline{WAIT}$ input (1 + N) WAIT mode | $t_{AWL}$ | | 3.0x – 80 | | 108 | | 70 | ns |
| 24 | $\overline{RD}$ / $\overline{WR}$ fall → $\overline{WAIT}$ hold (1 + N) WAIT mode | $t_{CW}$ | 2.0x + 0 | | 125 | | 100 | | ns |
| 25 | A0 to A23 valid → Port input | $t_{APH}$ | | 2.5x – 120 | | 36 | | 5 | ns |
| 26 | A0 to A23 valid → Port hold | $t_{APH2}$ | 2.5x + 50 | | 206 | | 175 | | ns |
| 27 | $\overline{WR}$ rise → Port valid | $t_{CP}$ | | 200 | | 200 | | 200 | ns |
| 28 | A0 to A23 valid → $\overline{RAS}$ fall | $t_{ASRH}$ | 1.0x – 40 | | 23 | | 10 | | ns |
| 29 | A0 to A15 valid → $\overline{RAS}$ fall | $t_{ASRL}$ | 0.5x – 15 | | 16 | | 10 | | ns |
| 30 | $\overline{RAS}$ fall → D0 to D15 input | $t_{RAC}$ | | 2.5x – 70 | | 86 | | 55 | ns |
| 31 | $\overline{RAS}$ fall → A0 to A15 hold | $t_{RAH}$ | 0.5x – 15 | | 16 | | 10 | | ns |
| 32 | $\overline{RAS}$ low pulse width | $t_{RAS}$ | 2.0x – 40 | | 85 | | 60 | | ns |
| 33 | $\overline{RAS}$ high pulse width | $t_{RP}$ | 2.0x – 40 | | 85 | | 60 | | ns |
| 34 | $\overline{CAS}$ fall → $\overline{RAS}$ rise | $t_{RSH}$ | 1.0x – 40 | | 23 | | 10 | | ns |
| 35 | $\overline{RAS}$ rise → $\overline{CAS}$ rise | $t_{RSC}$ | 0.5x – 25 | | 6 | | 0 | | ns |
| 36 | $\overline{RAS}$ fall → $\overline{CAS}$ fall | $t_{RCD}$ | 1.0x – 40 | | 23 | | 10 | | ns |
| 37 | $\overline{CAS}$ fall → D0 to D15 input | $t_{CAC}$ | | 1.5x – 65 | | 29 | | 10 | ns |
| 38 | $\overline{CAS}$ low pulse width | $t_{CAS}$ | 1.5x – 30 | | 64 | | 40 | | ns |

AC measuring conditions

- Output level: High 2.2 V/Low 0.8 V, $C_L = 50$ pF
  (However, $C_L = 100$ pF for AD0 to AD15, A0 to A23, ALE, $\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, R/$\overline{W}$, CLK, $\overline{RAS}$, $\overline{CAS0}$ to $\overline{CAS2}$)
- Input level: High 2.4 V/Low 0.45 V (AD0 to AD15)
  High $0.8 \times V_{CC}$/Low $0.2 \times V_{CC}$ (except for AD0 to AD15)

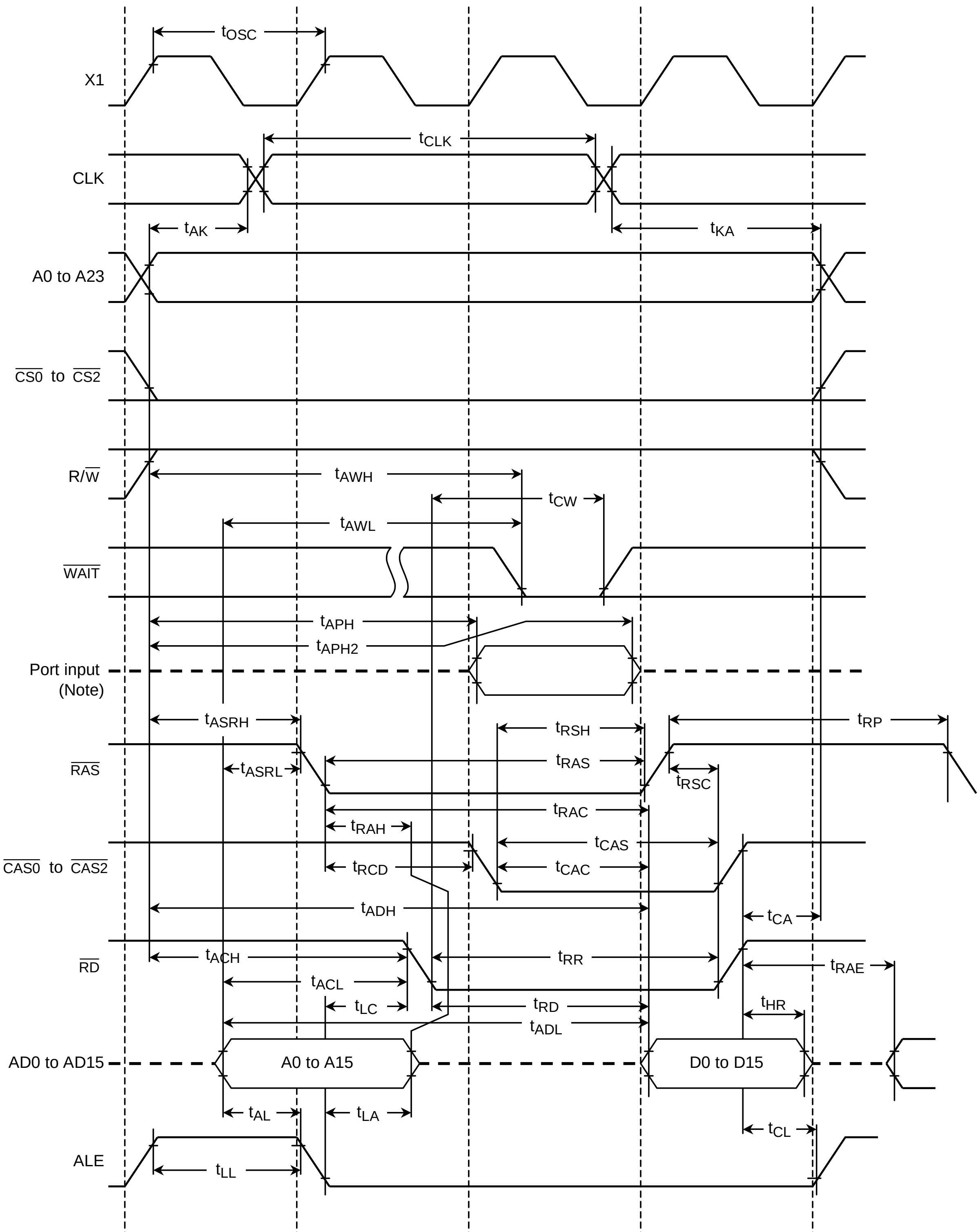
(2) $V_{CC} = 3\ V \pm 10\%$

| No. | Parameter | Symbol | Variable | | 12.5 MHz | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| 1 | Osc. period (= X) | $t_{OSC}$ | 80 | 31250 | 80 | | ns |
| 2 | CLK pulse width | $t_{CLK}$ | 2x – 40 | | 120 | | ns |
| 3 | A0 to A23 valid → CLK hold | $t_{AK}$ | 0.5x – 30 | | 10 | | ns |
| 4 | CLK valid → A0 to A23 hold | $t_{KA}$ | 1.5x – 80 | | 40 | | ns |
| 5 | A0 to A15 valid → ALE fall | $t_{AL}$ | 0.5x – 35 | | 5 | | ns |
| 6 | ALE fall → A0 to A15 hold | $t_{LA}$ | 0.5x – 35 | | 5 | | ns |
| 7 | ALE high pulse width | $t_{LL}$ | x – 60 | | 20 | | ns |
| 8 | ALE fall → $\overline{RD}$ / $\overline{WR}$ fall | $t_{LC}$ | 0.5x – 35 | | 5 | | ns |
| 9 | $\overline{RD}$ / $\overline{WR}$ rise → ALE rise | $t_{CL}$ | 0.5x – 40 | | 0 | | ns |
| 10 | A0 to A15 valid → $\overline{RD}$ / $\overline{WR}$ fall | $t_{ACL}$ | x – 50 | | 30 | | ns |
| 11 | A0 to A23 valid → $\overline{RD}$ / $\overline{WR}$ fall | $t_{ACH}$ | 1.5x – 50 | | 70 | | ns |
| 12 | $\overline{RD}$ / $\overline{WR}$ rise → A0 to A23 hold | $t_{CA}$ | 0.5x – 40 | | 0 | | ns |
| 13 | A0 to A15 valid → D0 to D15 input | $t_{ADL}$ | | 3.0x – 110 | | 130 | ns |
| 14 | A0 to A23 valid → D0 to D15 input | $t_{ADH}$ | | 3.5x – 125 | | 155 | ns |
| 15 | $\overline{RD}$ fall → D0 to D15 input | $t_{RD}$ | | 2.0x – 115 | | 45 | ns |
| 16 | $\overline{RD}$ low pulse width | $t_{RR}$ | 2.0x – 40 | | 120 | | ns |
| 17 | $\overline{RD}$ rise → D0 to D15 hold | $t_{HR}$ | 0 | | 0 | | ns |
| 18 | $\overline{RD}$ rise → A0 to A15 output | $t_{RAE}$ | x – 25 | | 55 | | ns |
| 19 | $\overline{WR}$ low pulse width | $t_{WW}$ | 2.0x – 40 | | 120 | | ns |
| 20 | D0 to D15 valid → $\overline{WR}$ rise | $t_{DW}$ | 2.0x – 120 | | 40 | | ns |
| 21 | $\overline{WR}$ rise → D0 to D15 hold | $t_{WD}$ | 0.5x – 40 | | 0 | | ns |
| 22 | A0 to A23 valid → $\overline{WAIT}$ input (1 + N) WAIT mode | $t_{AWH}$ | | 3.5x – 130 | | 150 | ns |
| 23 | A0 to A15 valid → $\overline{WAIT}$ input (1 + N) WAIT mode | $t_{AWL}$ | | 3.0x – 100 | | 140 | ns |
| 24 | $\overline{RD}$ / $\overline{WR}$ fall → $\overline{WAIT}$ hold (1 + N) WAIT mode | $t_{CW}$ | 2.0x + 0 | | 160 | | ns |
| 25 | A0 to A23 valid → Port input | $t_{APH}$ | | 2.5x – 195 | | 5 | ns |
| 26 | A0 to A23 valid → Port hold | $t_{APH2}$ | 2.5x + 50 | | 250 | | ns |
| 27 | $\overline{WR}$ rise → Port valid | $t_{CP}$ | | 200 | | 200 | ns |
| 28 | A0 to A23 valid → $\overline{RAS}$ fall | $t_{ASRH}$ | 1.0x – 60 | | 20 | | ns |
| 29 | A0 to A15 valid → $\overline{RAS}$ fall | $t_{ASRL}$ | 0.5x – 40 | | 0 | | ns |
| 30 | $\overline{RAS}$ fall → D0 to D15 input | $t_{RAC}$ | | 2.5x – 90 | | 110 | ns |
| 31 | $\overline{RAS}$ fall → A0 to A15 hold | $t_{RAH}$ | 0.5x – 25 | | 15 | | ns |
| 32 | $\overline{RAS}$ low pulse width | $t_{RAS}$ | 2.0x – 40 | | 120 | | ns |
| 33 | $\overline{RAS}$ high pulse width | $t_{RP}$ | 2.0x – 40 | | 120 | | ns |
| 34 | $\overline{CAS}$ fall → $\overline{RAS}$ rise | $t_{RSH}$ | 1.0x – 55 | | 25 | | ns |
| 35 | $\overline{RAS}$ rise → $\overline{CAS}$ rise | $t_{RSC}$ | 0.5x – 25 | | 15 | | ns |
| 36 | $\overline{RAS}$ fall → $\overline{CAS}$ fall | $t_{RCD}$ | 1.0x – 40 | | 40 | | ns |
| 37 | $\overline{CAS}$ fall → D0 to D15 input | $t_{CAC}$ | | 1.5x – 120 | | 0 | ns |
| 38 | $\overline{CAS}$ low pulse width | $t_{CAS}$ | 1.5x – 40 | | 80 | | ns |

AC measuring conditions

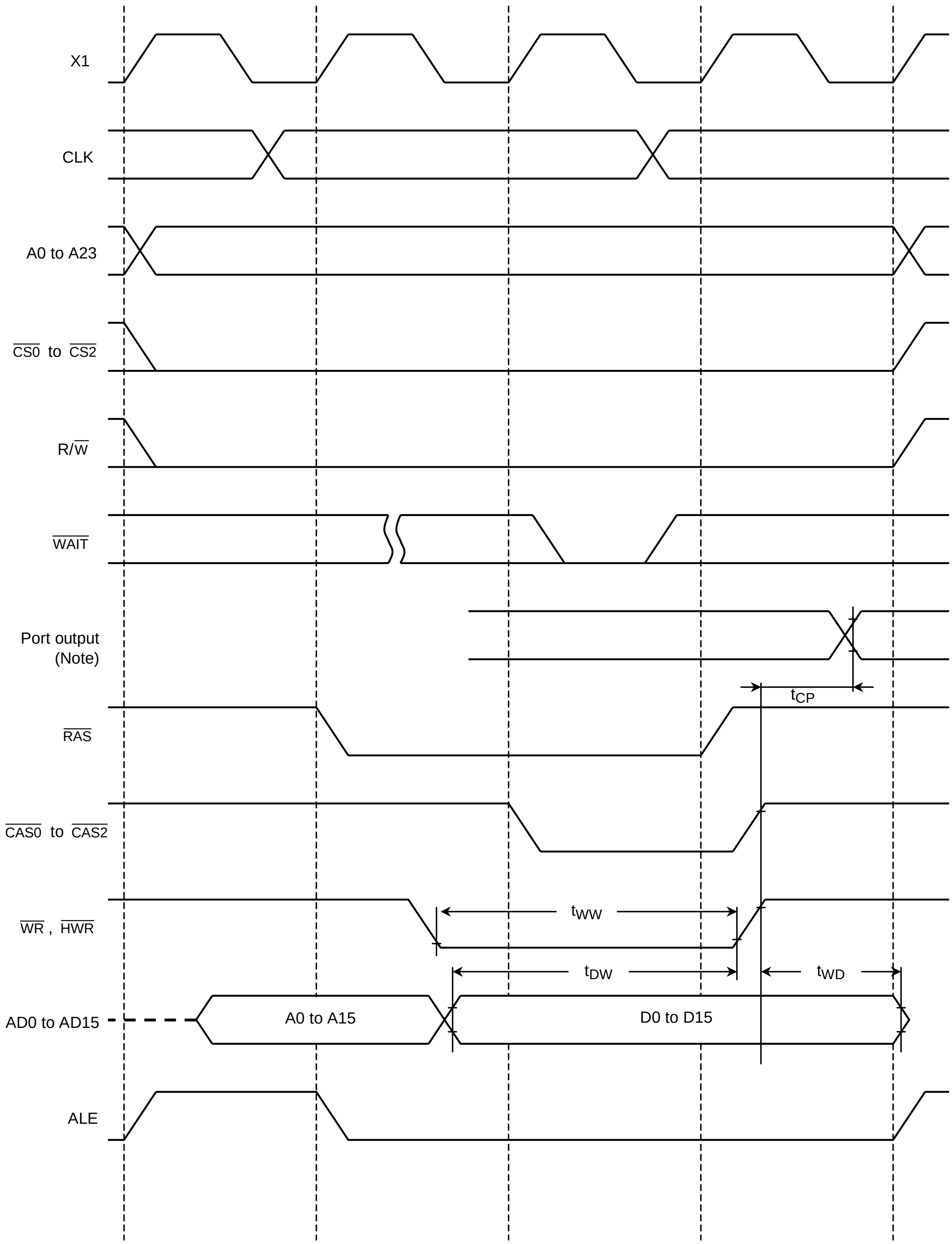
- Output level: High $0.7 \times V_{CC}$/Low $0.3 \times V_{CC}$, $C_L = 50\ pF$
- Input level: High $0.9 \times V_{CC}$/Low $0.1 \times V_{CC}$

(1) Read cycle




Note: Since the CPU accesses the internal area to read data from a port, the control signals of external pins such as $\overline{RD}$ and $\overline{CS}$ are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

(2) Write cycle




Note: Since the CPU accesses the internal area to write data to a port, the control signals of external pins such as WR and CS are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

## 4.4 AD Conversion Characteristics

$AV_{CC} = V_{CC}$, $AV_{SS} = V_{SS}$

| Parameter | Symbol | Power Supply | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Analog reference voltage (+) | $V_{REFH}$ | $V_{CC} = 5$ V ± 10% | $V_{CC} - 1.5$ V | $V_{CC}$ | $V_{CC}$ | V |
| | | $V_{CC} = 3$ V ± 10% | $V_{CC} - 0.2$ V | $V_{CC}$ | $V_{CC}$ | |
| Analog reference voltage (–) | $V_{REFL}$ | $V_{CC} = 5$ V ± 10% | $V_{SS}$ | $V_{SS}$ | $V_{SS} + 0.2$ V | |
| | | $V_{CC} = 3$ V ± 10% | $V_{SS}$ | $V_{SS}$ | $V_{SS} + 0.2$ V | |
| Analog input voltage range | $V_{AIN}$ | | $V_{REFL}$ | | $V_{REFH}$ | |
| Analog current for analog reference voltage <VREFON> = 1 | $I_{REF}$ ($V_{REFL} = 0$ V) | $V_{CC} = 5$ V ± 10% | | 0.5 | 1.5 | mA |
| | | $V_{CC} = 3$ V ± 10% | | 0.3 | 0.9 | |
| <VREFON> = 0 | | $V_{CC} = 2.7$ to $5.5$ V | | 0.02 | 5.0 | μA |
| Error (not including quantizing errors) | – | $V_{CC} = 5$ V ± 10% | | ±1.0 | ±3.0 | LSB |
| | | $V_{CC} = 3$ V ± 10% | | ±1.0 | ±3.0 | |

Note 1: $1LSB = (V_{REFH} - V_{REFL})/2^{10}$ [V]

Note 2: The operation of this AD converter is guaranteed only using fc (The high-frequency oscillator). It is not guaranteed for fs.
The operation above is guaranteed for $f_{FPH} \geq 4$ MHz.

Note 3: The value $I_{CC}$ includes the current which flows through the AVCC pin.

## 4.5 Serial Channel Timing

(1) I/O interface mode

1. SCLK input mode

| Parameter | Symbol | Variable | | 32.768 MHz (Note) | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | Min | Max | |
| SCLK cycle | $t_{SCY}$ | 16X | | 488 | | 1.28 | | 0.8 | | μs |
| Output data → Rising edge or falling edge* of SCLK | $t_{OSS}$ | $t_{SCY}/2 - 5X - 50$ | | 91.5 μs | | 190 | | 100 | | ns |
| SCLK rising edge or falling edge* → Output data hold | $t_{OHS}$ | 5X – 100 | | 152 μs | | 300 | | 150 | | ns |
| SCLK rising edge or falling edge* → Input data hold | $t_{HSR}$ | 0 | | 0 | | 0 | | 0 | | ns |
| SCLK rising edge or falling edge* → Effective data input | $t_{SRD}$ | | $t_{SCY} - 5X - 100$ | | 336 μs | | 780 | | 450 | ns |

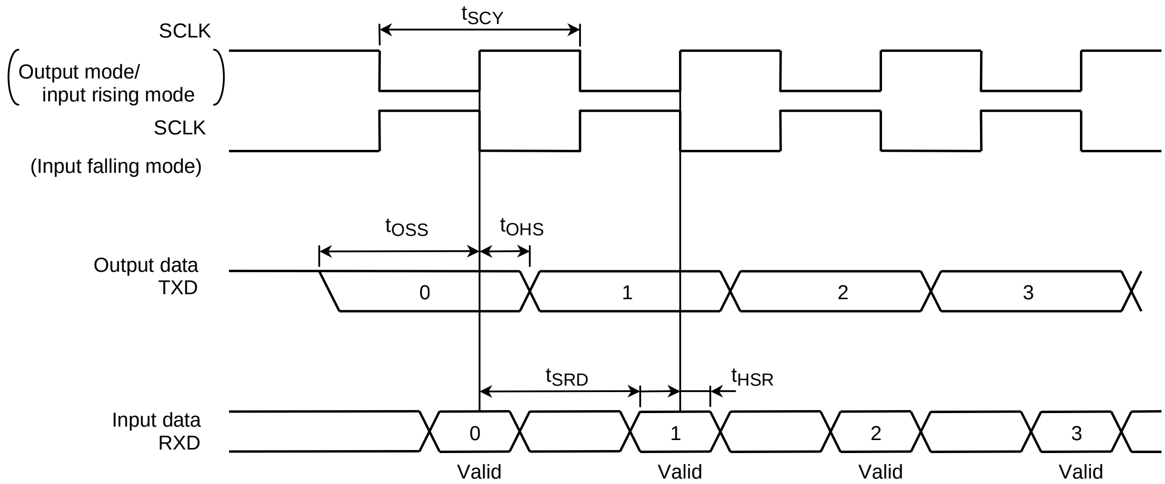
Note: System clock is fs, or input clock to prescaler is divisor clock of fs.

* The rising edge is used in SCLK rising mode.
The falling edge is used SCLK falling mode.

2. SCLK output mode

| Parameter | Symbol | Variable | | 32.768 MHz (Note) | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | Min | Max | |
| SCLK cycle (Programmable) | $t_{SCY}$ | 16X | 8192X | 488 | 250 ms | 1.28 | 655.36 | 0.8 | 409.6 | μs |
| Output data → SCLK rising edge | $t_{OSS}$ | $t_{SCY} - 2X - 150$ | | 427 μs | | 970 | | 550 | | ns |
| SCLK rising edge → Output data hold | $t_{OHS}$ | 2X – 80 | | 60 μs | | 80 | | 20 | | ns |
| SCLK rising edge → Input data hold | $t_{HSR}$ | 0 | | 0 | | 0 | | 0 | | ns |
| SCLK rising edge → Effective data input | $t_{SRD}$ | | $t_{SCY} - 2X - 150$ | | 428 μs | | 970 | | 550 | ns |

Note: System clock is fs, or input clock to prescaler is divisor clock of fs.




## 4.6 Timer/Counter Input Clock (TI0, TI4, TI5, TI6 and TI7)

| Parameter | Symbol | Variable | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| Clock cycle | $t_{VCK}$ | 8X + 100 | | 740 | | 500 | | ns |
| Low level clock pulse width | $t_{VCKL}$ | 4X + 40 | | 360 | | 240 | | ns |
| High level clock pulse width | $t_{VCKH}$ | 4X + 40 | | 360 | | 240 | | ns |

## 4.7 Interrupt and Capture

(1) $\overline{\text{NMI}}$ and INT0 interrupts

| Parameter | Symbol | Variable | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| $\overline{\text{NMI}}$, INT0 low level pulse width | $t_{INTAL}$ | 4X | | 320 | | 200 | | ns |
| $\overline{\text{NMI}}$, INT0 high level pulse width | $t_{INTAH}$ | 4X | | 320 | | 200 | | ns |

(2) INT4 to INT7 interrupts, capture

The INT4 to INT7 input pulse width depends on the CPU operation clock and timer (9-bit prescaler). The following shows the pulse width for each clock.

| System Clock Selected <SYSCK> | Prescaler Clock Selected <PRCK1:0> | $t_{INTBL}$ (INT4 to INT7 low level pulse width) | | $t_{INTBH}$ (INT4 to INT7 high level pulse width) | | Unit |
|---|---|---|---|---|---|---|
| | | Variable | 20 MHz | Variable | 20 MHz | |
| | | Min | Min | Min | Min | |
| 0 (fc) | 00 ($f_{FPH}$) | 8X + 100 | 500 | 8X + 100 | 500 | ns |
| | 01 (fs) | 8XT + 0.1 | 244.3 | 8XT + 0.1 | 244.3 | μs |
| | 10 (fc/16) | 128x + 0.1 | 6.5 | 128X + 0.1 | 6.5 | |
| 1 (fs) (Note 2) | 00 ($f_{FPH}$) | 8XT + 0.1 | 244.3 | 8XT + 0.1 | 244.3 | |
| | 01 (fs) | | | | | |

Note 1: XT represents the frequency of the low-frequency clock fs. Calculated at fs = 32.768 kHz.

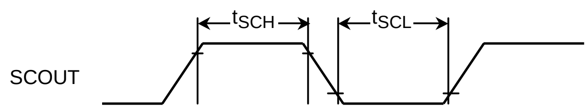
Note 2: When using fs as the system clock, fc/16 cannot be selected as the prescaler clock.

## 4.8 SCOUT Pin AC Characteristics

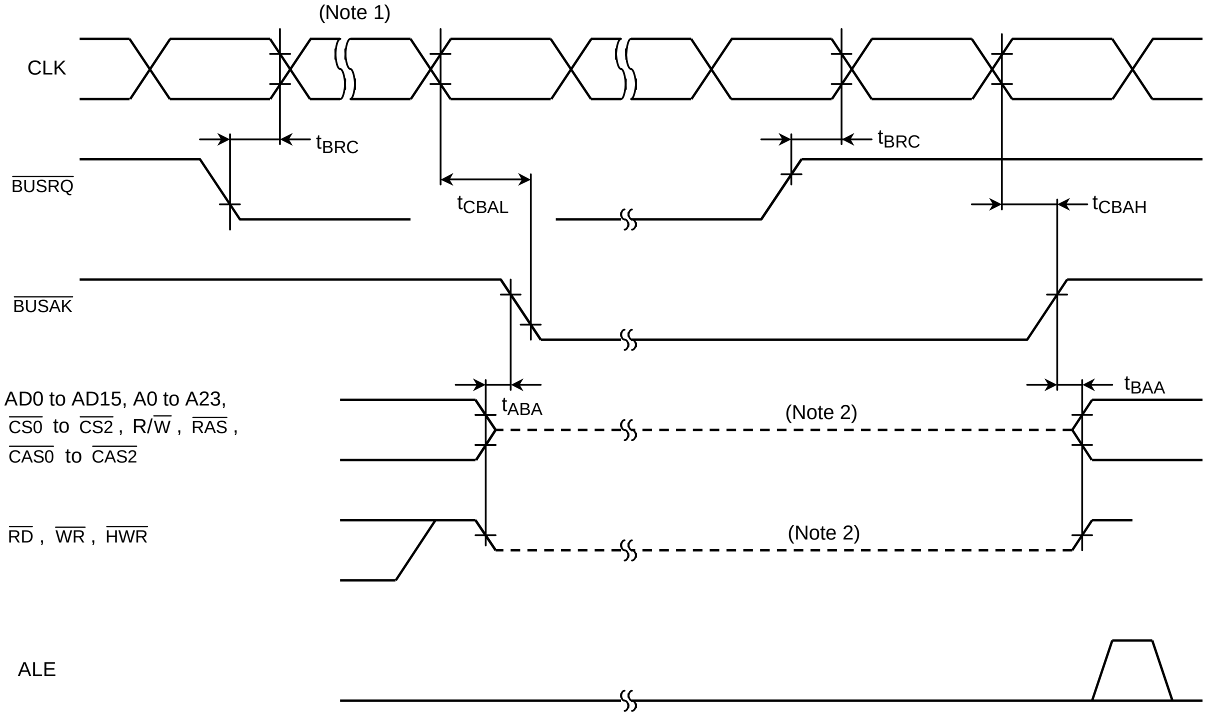
| Parameter | Symbol | Variable | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| High-level pulse width $V_{CC}$ = 5 V ± 10% | $t_{SCH}$ | 0.5X − 10 | | 40 | | 15 | | ns |
| High-level pulse width $V_{CC}$ = 3 V ± 10% | | 0.5X − 20 | | 30 | | – | – | |
| Low-level pulse width $V_{CC}$ = 5 V ± 10% | $t_{SCL}$ | 0.5X − 10 | | 40 | | 15 | | ns |
| Low-level pulse width $V_{CC}$ = 3 V ± 10% | | 0.5X − 20 | | 30 | | – | – | |

Measurement condition

- Output level: High 2.2 V/Low 0.8 V, $C_L$ = 10 pF



## 4.9 Timing Chart for Bus Request ($\overline{\text{BUSRQ}}$)/Bus Acknowledge ($\overline{\text{BUSAK}}$)




| Parameter | Symbol | Variable | | 12.5 MHz | | 20 MHz | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| $\overline{\text{BUSRQ}}$ setup time to CLK | $t_{BRC}$ | 120 | | 120 | | 120 | | ns |
| CLK→$\overline{\text{BUSAK}}$ falling edge | $t_{CBAL}$ | | 1.5X + 120 | | 240 | | 195 | ns |
| CLK→$\overline{\text{BUSAK}}$ rising edge | $t_{CBAH}$ | | 0.5x + 40 | | 80 | | 65 | ns |
| Output buffer off to $\overline{\text{BUSAK}}$ | $t_{ABA}$ | 0 | 80 | 0 | 80 | 0 | 80 | ns |
| $\overline{\text{BUSAK}}$ to output buffer on | $t_{BAA}$ | 0 | 80 | 0 | 80 | 0 | 80 | ns |

Note 1: Even if the $\overline{\text{BUSRQ}}$ signal goes low, the bus will not be released while the $\overline{\text{WAIT}}$ signal is low. The bus will only be released when $\overline{\text{BUSRQ}}$ goes low while $\overline{\text{WAIT}}$ is high.

Note 2: This line shows only that the output buffer is in the off state.
It does not indicate that the signal level is fixed.
Just after the bus is released, the signal level set before the bus was released is maintained dynamically by the external capacitance. Therefore, to fix the signal level using an external resistor during bus release, careful design is necessary, as fixing of the level is delayed.
The internal programmable pull-up/pull-down resistor is switched between the active and non-active states by the internal signal.

## 4.10 Recommended Oscillator

The TMP93CS40/S41 are evaluated with various resonators. The evaluation results are displayed below to enable appropriate selection for any given application.

Note: The load capacitance of the resonator consists of the load capacitors C1 and C2 which are to be connected and the floating capacitance of the target board.
Even if the specified values of C1 and C2 are used, there is a possibility that the oscillator will malfunction due to varying load capacitance on the target boards. Hence the oscillator’s wiring patterns on the board should be designed to be as short as possible.
It is recommended that evaluation of the resonators be conducted on the target board.

(1) Examples of resonator connection

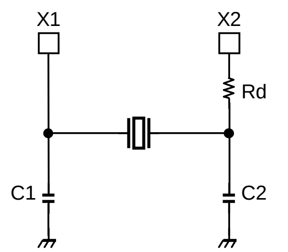



Figure1: Example of High-frequency Resonator Connection

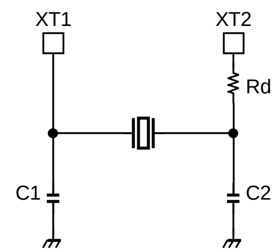



Figure2: Example of Low-frequency Resonator Connection

(2) Ceramic resonator: Murata Manufacturing. Co., Ltd (Note 1)

Ta = −20 to 80°C

| Parameter | Frequency (MHz) | Recommended Resonator | Recommended Value | | | $V_{CC}$ [V] |
|---|---|---|---|---|---|---|
| | | | C1 [pF] | C2 [pF] | Rd [kΩ] | |
| High-frequency oscillation | 4.00 | CSA4.00MG | 30 | 30 | 0 | 2.7 to 5.5 |
| | | CST4.00MGW | (30) (Note 2) | (30) (Note 2) | | |
| | 10.00 | CSA10.0MTZ093 | 30 | 30 | | |
| | | CST10.0MTW093 | (30) (Note 2) | (30) (Note 2) | | |
| | 12.50 | CSA12.5MTZ093 | 30 | 30 | | |
| | | CST12.5MTW093 | (30) (Note 2) | (30) (Note 2) | | |
| | 16.00 | CSA16.00MXZ040 | 5 | 5 | | 4.5 to 5.5 |
| | | CST16.00MXW0C1 | (5) (Note 2) | (5) (Note 2) | | |
| | 20.00 | CSA20.00MXZ040 | 3 | 3 | | |

Note 1: Murata Manufacturing. Co., Ltd. (Japan)
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.com/

Note 2: For built-in condenser type

(3) Crystal resonator: Nihon Denpa Kogyo (Note 1)

Ta = −10 to 60°C

| Parameter | Frequency (MHz) | Recommended Resonator | Recommended Value | | | Vcc [V] |
|---|---|---|---|---|---|---|
| | | | C1 [pF] | C2 [pF] | Rd [kΩ] | |
| High-frequency oscillation | 4.00 | NT040016A | 12 | 12 | 0 | 2.7 to 5.5 |
| | 10.00 | NT100016A | 10 | 10 | | |
| | 12.50 | NT125016A | 10 | 10 | | |
| | 16.00 | NT160016A | 10 | 10 | | 4.5 to 5.5 |
| | 20.00 | NT200016A | 7 | 7 | | |

Note 1:

| | | |
|---|---|---|
| NDK AMERICA, INC.: U.S.A | Phone: +1-510-623-6512, | Fax: +1-510-623-6590 |
| NDK ELECTRONICS SINGAPORE PTE, LTD. | Phone: +65-6298-9878, | Fax: +65-6293-1150 |
| NDK ELECTRONICS (HK) LIMITED: HONG KONG | Phone: +852-2956-3181, | Fax: +852-2956-1567 |
| NDK EUROPE LIMITED: ENGLAND | Phone: +44-20-8390-8344, | Fax: +44-20-8390-6926 |
| NDK FRANCE SARL: FRANCE | Phone: +33-1-60-95-0000, | Fax: +33-1-60-95-8200 |
| NDK ITALY SRL: ITALY | Phone: +39-02-96702920, | Fax: +39-02-96703284 |

Note 2: High-frequency resonator

| | |
|---|---|
| NR-18: | Lead mount type |
| AT-51: | Lead mount type |
| CP12A: | Surface mount type |

# 5. Table of Special Function Registers (SFRs)

(SFR: Special function register)

The special function registers (SFRs) include the I/O ports and peripheral control registers allocated to the 128 bytes whose addresses run from 000000H to 00007FH.

(1) I/O ports

(2) I/O port control

(3) Timer control

(4) Pattern generator control

(5) Watchdog timer control

(6) Serial channel control

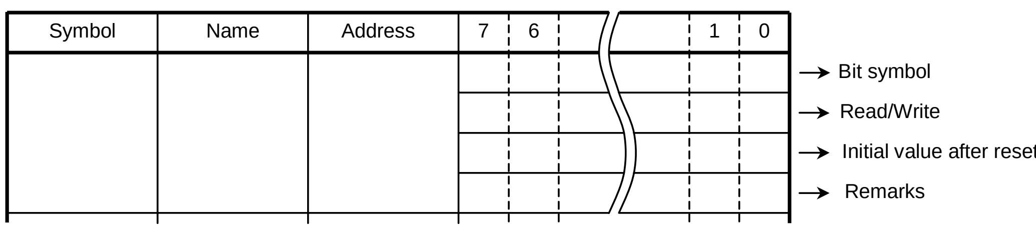
(7) AD converter control

(8) Interrupt control

(9) Chip select/wait control

(10) Clock control

Configuration of the table




Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these registers.

(Example) When setting only bit0 of register P0CR, "SET 0, (0002H)" cannot be used.
The LD (Transfer) instruction must be used to write all eight bits.

Table 5.1 I/O Register Address Map

| Address | Name | Address | Name | Address | Name | Address | Name |
|---|---|---|---|---|---|---|---|
| 000000H | P0 | 20H | TRUN | 40H | TREG6L | 60H | ADREG04L |
| 1H | P1 | 21H | | 41H | TREG6H | 61H | ADREG04H |
| 2H | P0CR | 22H | TREG0 | 42H | TREG7L | 62H | ADREG15L |
| 3H | | 23H | TREG1 | 43H | TREG7H | 63H | ADREG15H |
| 4H | P1CR | 24H | TMOD | 44H | CAP3L | 64H | ADREG26L |
| 5H | P1FC | 25H | TFFCR | 45H | CAP3H | 65H | ADREG26H |
| 6H | P2 | 26H | TREG2 | 46H | CAP4L | 66H | ADREG37L |
| 7H | P3 | 27H | TREG3 | 47H | CAP4H | 67H | ADREG37H |
| 8H | P2CR | 28H | P0MOD | 48H | T5MOD | 68H | B0CS |
| 9H | P2FC | 29H | P1MOD | 49H | T5FFCR | 69H | B1CS |
| AH | P3CR | 2AH | PFFCR | 4AH | | 6AH | B2CS |
| BH | P3FC | 2BH | | 4BH | | 6BH | |
| CH | P4 | 2CH | | 4CH | PG0REG | 6CH | |
| DH | P5 | 2DH | | 4DH | PG1REG | 6DH | CKOCR |
| EH | P4CR | 2EH | | 4EH | PG01CR | 6EH | SYSCR0 |
| FH | | 2FH | | 4FH | | 6FH | SYSCR1 |
| 10H | P4FC | 30H | TREG4L | 50H | SC0BUF | 70H | INTE0AD |
| 11H | | 31H | TREG4H | 51H | SC0CR | 71H | INTE45 |
| 12H | P6 | 32H | TREG5L | 52H | SC0MOD | 72H | INTE67 |
| 13H | P7 | 33H | TREG5H | 53H | BR0CR | 73H | INTET10 |
| 14H | P6CR | 34H | CAP1L | 54H | SC1BUF | 74H | INTEPW10 |
| 15H | P7CR | 35H | CAP1H | 55H | SC1CR | 75H | INTET54 |
| 16H | P6FC | 36H | CAP2L | 56H | SC1MOD | 76H | INTET76 |
| 17H | P7FC | 37H | CAP2H | 57H | BR1CR | 77H | INTES0 |
| 18H | P8 | 38H | T4MOD | 58H | ODE | 78H | INTES1 |
| 19H | P9 | 39H | T4FFCR | 59H | | 79H | |
| 1AH | P8CR | 3AH | T45CR | 5AH | | 7AH | |
| 1BH | P9CR | 3BH | | 5BH | | 7BH | IIMC |
| 1CH | P8FC | 3CH | | 5CH | WDMOD | 7CH | DMA0V |
| 1DH | P9FC | 3DH | | 5DH | WDCR | 7DH | DMA1V |
| 1EH | PA | 3EH | | 5EH | ADMOD1 | 7EH | DMA2V |
| 1FH | PACR | 3FH | | 5FH | ADMOD2 | 7FH | DMA3V |

Note: Do not access addresses which do not have register names allocated.

(1) I/O port

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P0 | Port0 | 00H | P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |
| | | | R/W | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | Undefined | | | | | | | |
| P1 | Port1 | 01H | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| | | | R/W | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| P2 | Port2 | 06H | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 |
| | | | *R/W (Note 3) | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| P3 | Port 3 | 07H | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 (Note 1) |
| | | | *R/W (Note 3) | | | | | | | |
| | | | Input mode | | | | | | Output mode | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| P4 | Port4 | 0CH | | | | | | P42 | P41 | P40 |
| | | | | | | | | *R/W (Note 3) | | |
| | | | | | | | | Input mode | | |
| | | | | | | | | 0 | 1 | 1 |
| P5 | Port5 | 0DH | P57 | P56 | P55 | P54 | P53 | P52 | P51 | P50 |
| | | | R | | | | | | | |
| | | | Input mode | | | | | | | |
| P6 | Port6 | 12H | P67 | P66 | P65 | P64 | P63 | P62 | P61 | P60 |
| | | | *R/W (Note 3) | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| P7 | Port7 | 13H | | | | | P73 | P72 | P71 | P70 |
| | | | | | | | *R/W (Note 3) | | | |
| | | | | | | | Input mode | | | |
| | | | | | | | 1 | 1 | 1 | 1 |
| P8 | Port8 | 18H | P87 | P86 | P85 | P84 | P83 | P82 | P81 | P80 |
| | | | *R/W (Note 3) | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| P9 | Port9 (Note2) | 19H | P97 | P96 | P95 | P94 | P93 | P92 | P91 | P90 |
| | | | R/W | R/W | *R/W (Note 3) | | | | | |
| | | | Output mode | Output mode | Input mode | | | | | |
| | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| PA | PortA | 1EH | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| | | | R/W | | | | | | | |
| | | | Input mode | | | | | | | |
| | | | 1 | | | | | | | |

Note 1: When P30 pin is defined as $\overline{\text{RD}}$ signal output mode (P30F = 1), clearing the output latch register P30 to “0” outputs the $\overline{\text{RD}}$ strobe from P30 pin for PSRAM, even when the internal address is accessed. If the output latch register P30 remains “1”, the $\overline{\text{RD}}$ strobe is output only when the external address is accessed.

Note 2: Port 96, 97 is also used as XT1, XT2. Therefore these pins are open-drain output type.

Read/Write

- R/W: Either read or write is possible
- R: Only read is possible
- W: Only write is possible
- Prohibit RMW: Prohibit read-modify-write. (Prohibit RES/SET/TSET/CHG/STCF/ANDCF/ORCF/XORCF instruction.)

Note 3: *R/W: Read-modify-write is prohibited when controlling the PU/PD resistors.

(2) I/O port control (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P0CR | Port 0 control | 02H (Prohibit RMW) | P07C | P06C | P05C | P04C | P03C | P02C | P01C | P00C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output (When external access, set as AD7 to AD0 and cleared to 0.) | | | | | | | |
| P1CR | Port 1 control | 04H (Prohibit RMW) | P17C | P16C | P15C | P14C | P13C | P12C | P11C | P10C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | <<Refer to the P1FC>> | | | | | | | |
| P1FC | Port 1 function | 05H (Prohibit RMW) | P17F | P16F | P15F | P14F | P13F | P12F | P11F | P10F |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | P1FC/P1CR = 00: Input 01: Output 10: AD15 to AD8 11: A15 to A8 | | | | | | | |
| P2CR | Port 2 control | 08H (Prohibit RMW) | P27C | P26C | P25C | P24C | P23C | P22C | P21C | P20C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | <<Refer to the P2FC>> | | | | | | | |
| P2FC | Port 2 function | 09H (Prohibit RMW) | P27F | P26F | P25F | P24F | P23F | P22F | P21F | P20F |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | P2FC/P2CR = 00: Input 01: Output 10: A7 to A0 11: A23 to A16 | | | | | | | |
| P3CR | Port 3 control | 0AH (Prohibit RMW) | P37C | P36C | P35C | P34C | P33C | P32C | | |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | | |
| | | | 0: Input 1: Output | | | | | | | |
| P3FC | Port 3 function | 0BH (Prohibit RMW) | P37F | P36F | P35F | P34F | | P32F | P31F | P30F |
| | | | W | | | | | W | | |
| | | | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| | | | 0: Port<br>1: $\overline{RAS}$ | 0: Port<br>1: R/$\overline{W}$ | 0: Port<br>1: $\overline{BUSAK}$ | 0: Port<br>1: $\overline{BUSRQ}$ | | 0: Port<br>1: $\overline{HWR}$ | 0: Port<br>1: $\overline{WR}$ | 0: Port<br>1: $\overline{RD}$ |
| P4CR | Port 4 control | 0EH (Prohibit RMW) | | | | | | P42C | P41C | P40C |
| | | | | | | | | W | | |
| | | | | | | | | 0 | 0 | 0 |
| | | | | | | | | 0: Input 1: Output | | |
| P4FC | Port 4 function | 10H (Prohibit RMW) | | | | | | P42F | P41F | P40F |
| | | | | | | | | W | | |
| | | | | | | | | 0 | 0 | 0 |
| | | | | | | | | 0: Port 1: $\overline{CS}$ / $\overline{CAS}$ | | |

Note: With the TMP93CS41, which requires an external ROM, port 0 functions as AD0 to AD7; port 1, AD8 to AD15 or A8 to A15; P30, the $\overline{RD}$ signal; P31, the $\overline{WR}$ signal, regardless of the values set in P0CR, P1CR, P1FC, P30F, and P31F.

I/O port control (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| P6CR | Port 6 control | 14H (Prohibit RMW) | P67C | P66C | P65C | P64C | P63C | P62C | P61C | P60C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| P7CR | Port 7 control | 15H (Prohibit RMW) | | | | | P73C | P72C | P71C | P70C |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | 0: Input 1: Output | | | |
| P6FC | Port 6 function | 16H (Prohibit RMW) | P67F | P66F | P65F | P64F | P63F | P62F | P61F | P60F |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Port 1: PG1 output | | | | 0: Port 1: PG0 output | | | |
| P7FC | Port 7 function | 17H (Prohibit RMW) | | | | | P73F | P72F | P71F | |
| | | | | | | | W | | | |
| | | | | | | | 0 | 0 | 0 | |
| | | | | | | | 0: Port<br>1: TO3 | 0: Port<br>1: TO2 | 0: Port<br>1: TO1 | |
| P8CR | Port 8 control | 1AH (Prohibit RMW) | P87C | P86C | P85C | P84C | P83C | P82C | P81C | P80C |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| P9CR | Port 9 control | 1BH (Prohibit RMW) | P97C | P96C | P95C | P94C | P93C | P92C | P91C | P90C |
| | | | W | W | W | | | | | |
| | | | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: Input 1: Output | | | | | | | |
| P8FC | Port 8 function | 1CH (Prohibit RMW) | | P86F | | | P83F | P82F | | |
| | | | | W | | | W | W | | |
| | | | | 0 | | | 0 | 0 | | |
| | | | | 0: Port<br>1: TO6 | | | 0: Port<br>1: TO5 | 0: Port<br>1: TO4 | | |
| P9FC | Port 9 function | 1DH (Prohibit RMW) | | | P95F | | P93F | P92F | | P90F |
| | | | | | W | | W | W | | W |
| | | | | | 0 | | 0 | 0 | | 0 |
| | | | | | 0: Port<br>1: SCLK1 | | 0: Port<br>1: TXD1 | 0: Port<br>1: SCLK0 | | 0: Port<br>1: TXD0 |
| PACR | Port A control | 1FH (Prohibit RMW) | PA7C | PA6C | PA5C | PA4C | PA3C | PA2C | PA1C | PA0C |
| | | | W | | | | | | | |
| | | | 0 | | | | | | | |
| | | | 0: Input 1: Output | | | | | | | |

(3) Timer control (1/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| TRUN | Timer control | 20H | PRRUN | | T5RUN | T4RUN | P1RUN | P0RUN | T1RUN | T0RUN |
| | | | R/W | | R/W | | | | | |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Prescaler and timer run/stop control<br>0: Stop and clear<br>1: Run (Count up) | | | | | | | |
| TREG0 | 8-bit timer register 0 | 22H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG1 | 8-bit timer register 1 | 23H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TMOD | 8-bit timer source CLK & mode | 24H (Prohibit RMW) | T10M1 | T10M0 | PWMM1 | PWMM0 | T1CLK1 | T1CLK0 | T0CLK1 | T0CLK0 |
| | | | W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 00: 8-bit timer<br>01: 16-bit timer<br>10: 8-bit PPG<br>11: 8-bit PWM | | 00: –<br>01: $2^6 - 1$<br>10: $2^7 - 1$<br>11: $2^8 - 1$<br>PWM | | 00: TO0TRG<br>01: $\phi$T1<br>10: $\phi$T16<br>11: $\phi$T256 | | 00: TI0 input<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |
| TFFCR | 8-bit timer flip-flop control | 25H | | | | DBEN | TFF1C1 | TFF1C0 | TFF1IE | TFF1IS |
| | | | | | | R/W | W | | R/W | |
| | | | | | | 0 | 1 | 1 | 0 | 0 |
| | | | | | | 1: Double buffer enable | 00: Invert TFF1<br>01: Set TFF1<br>10: Clear TFF1<br>11: Don't care | | 1: TFF1 invert enable | 0: Inverted by timer 0 |
| TREG2 | PWM timer register 2 | 26H | – | | | | | | | |
| | | | (R)/W (Can read double buffer values.) | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG3 | PWM timer register 3 | 27H | – | | | | | | | |
| | | | (R)/W (Can read double buffer values.) | | | | | | | |
| | | | Undefined | | | | | | | |
| P0MOD | PWM0 mode | 28H (Prohibit RMW) | FF2RD | DB2EN | PWM0INT | PWM0M | T2CLK1 | T2CLK0 | PWM0S1 | PWM0S0 |
| | | | R | W | | | | | | |
| | | | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | TFF2 output value | 1: Double buffer enable | 0: Overflow interrupt<br>1: Compare/match interrupt | 0: PWM mode<br>1: Timer mode | 00: $\phi$P1<br>01: $\phi$P4<br>10: $\phi$P16<br>11: Don't care | | 00: $2^6 - 1$<br>01: $2^7 - 1$<br>10: $2^8 - 1$<br>11: Don't care | |
| P1MOD | PWM1 mode | 29H (Prohibit RMW) | FF3RD | DB3EN | PWM1INT | PWM1M | T3CLK1 | T3CLK0 | PWM1S1 | PWM1S0 |
| | | | R | W | | | | | | |
| | | | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | TFF3 output value | 1: Double buffer enable | 0: Overflow interrupt<br>1: Compare/match interrupt | 0: PWM mode<br>1: Timer mode | 00: $\phi$P1<br>01: $\phi$P4<br>10: $\phi$P16<br>11: Don't care | | 00: $2^6 - 1$<br>01: $2^7 - 1$<br>10: $2^8 - 1$<br>11: Don't care | |

Timer control (2/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| PFFCR | PWM flip-flop control | 2AH | FF3C1 | FF3C0 | FF3TRG1 | FF3TRG0 | FF2C1 | FF2C0 | FF2TRG1 | FF2TRG0 |
| | | | W | | R/W | | W | | R/W | |
| | | | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| | | | 00: Don't care<br>01: Set TFF3<br>10: Clear TFF3<br>11: Don't care | | 00: Prohibit TFF3 invert<br>01: Invert if matched<br>10: Set if matched; clear if overflow<br>11: Clear if matched; set if overflow | | 00: Don't care<br>01: Set TFF2<br>10: Clear TFF2<br>11: Don't care | | 00: Prohibit TFF2 invert<br>01: Invert if matched<br>10: Set if matched; clear if overflow<br>11: Clear if matched; set if overflow | |
| TREG4L | 16-bit timer register 4 low | 30H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG4H | 16-bit timer register 4 high | 31H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG5L | 16-bit timer register 5 low | 32H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG5H | 16-bit timer register 5 high | 33H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP1L | Capture register 1 low | 34H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP1H | Capture register 1 high | 35H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP2L | Capture register 2 low | 36H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP2H | Capture register 2 high | 37H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| T4MOD | 16-bit timer 4 source CLK and mode | 38H | CAP2T5 | EQ5T5 | CAP1IN | CAP12M1 | CAP12M0 | CLE | T4CLK1 | T4CLK0 |
| | | | R/W | | W | R/W | | | | |
| | | | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | TFF5 INV TRG<br>0: TRG disable<br>1: TRG enable | | 0: Software capture<br>1: Don't care | Capture timing<br>00: Disable<br>01: T14 ↑ T15 ↑<br>10: T14 ↑ T14 ↓<br>11: TFF1 ↑ TFF1 ↓ | | 1: UC4 clear enable | Source clock<br>00: T14<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |
| | | | Inverted when the UC value is latched to CAP2 | Inverted when the up counter matches TREG5 | | | | | | |
| T4FFCR | 16-bit timer 4 flip-flop control | 39H | TFF5C1 | TFF5C0 | CAP2T4 | CAP1T4 | EQ5T4 | EQ4T4 | TFF4C1 | TFF4C0 |
| | | | W | | R/W | | | | W | |
| | | | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| | | | 00: Invert TFF5<br>01: Set TFF5<br>10: Clear TFF5<br>11: Don't care | | TFF4 invert trigger<br>0: Trigger disable<br>1: Trigger enable | | | | 00: Invert TFF4<br>01: Set TFF4<br>10: Clear TFF4<br>11: Don't care | |
| | | | | | Inverted when the UC value is latched to CAP2 | Inverted when the UC value is latched to CAP1 | Inverted when the UC value matches TREG5 | Inverted when the UC value matches TREG4 | | |

Timer control (3/3)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| T45CR | T4, T5 control | 3AH | QCU | | | | PG1T | PG0T | DB6EN | DB4EN |
| | | | R/W | | | | R/W | | | |
| | | | 0 | | | | 0 | 0 | 0 | 0 |
| | | | Watchdog /warm-up timer control | | | | PG1 shift trigger<br>0: Timer 0, 1<br>1: Timer 5 | PG0 shift trigger<br>0: Timer 0, 1<br>1: Timer 4 | 1: Double buffer enable | |
| | | | | | | | | | Double buffer of TREG6 | Double buffer of TREG4 |
| TREG6L | 16-bit timer register 6 low | 40H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG6H | 16-bit timer register 6 high | 41H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG7L | 16-bit timer register 7 low | 42H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| TREG7H | 16-bit timer register 7 high | 43H (Prohibit RMW) | – | | | | | | | |
| | | | W | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP3L | Capture register 3 low | 44H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP3H | Capture register 3 high | 45H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP4L | Capture register 4 low | 46H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| CAP4H | Capture register 4 high | 47H | – | | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| T5MOD | 16-bit timer 5 source CLK and mode | 48H | | | CAP3IN | CAP34M1 | CAP34M0 | CLE | T5CLK1 | T5CLK0 |
| | | | | | W | R/W | | | | |
| | | | | | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | | | 0: Soft capture<br>1: Don't care | Capture timing<br>00: Disable<br>01: T16 ↑ T17 ↑<br>10: T16 ↑ T16 ↓<br>11: TFF1 ↑ TFF1 ↓ | | 1: UC5 clear enable | Source clock<br>00: T16<br>01: $\phi$T1<br>10: $\phi$T4<br>11: $\phi$T16 | |
| T5FFCR | 16-bit timer 5 flip-flop control | 49H | | | CAP4T6 | CAP3T6 | EQ7T6 | EQ6T6 | TFF6C1 | TFF6C0 |
| | | | | | R/W | | | | W | |
| | | | | | 0 | 0 | 0 | 0 | 1 | 1 |
| | | | | | TFF6 invert trigger<br>0: Trigger disable<br>1: Trigger enable | | | | 00: Invert TFF6<br>01: Set TFF6<br>10: Clear TFF6<br>11: Don't care | |
| | | | | | Inverted when the UC value is latched to CAP4 | Inverted when the UC value is latched to CAP3 | Inverted when the UC value matches TREG7 | Inverted when the UC value matches TREG6 | | |

(4) Pattern generator

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| PG0REG | PG0 register | 4CH (Prohibit RMW) | PG03 | PG02 | PG01 | PG00 | SA03 | SA02 | SA01 | SA00 |
| | | | W | | | | R/W | | | |
| | | | 0 | 0 | 0 | 0 | Undefined | | | |
| PG1REG | PG1 register | 4DH (Prohibit RMW) | PG13 | PG12 | PG11 | PG10 | SA13 | SA12 | SA11 | SA10 |
| | | | W | | | | R/W | | | |
| | | | 0 | 0 | 0 | 0 | Undefined | | | |
| PG01CR | PG0, 1 control | 4EH | PAT1 | CCW1 | PG1M | PG1TE | PAT0 | CCW0 | PG0M | PG0TE |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0: 8-bit write<br>1: 4-bit write | 0: Normal rotation<br>1: Reverse rotation | 0: 4-bit step<br>1: 8-bit step | PG1 trigger input enable<br>1: Enable | 0: 8-bit write<br>1: 4-bit write | 0: Normal rotation<br>1: Reverse rotation | 0: 4-bit step<br>1: 8-bit step | PG0 trigger input enable<br>1: Enable |

(5) Watchdog timer

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| WDMOD | Watchdog timer mode | 5CH | WDTE | WDTP1 | WDTP0 | WARM | HALTM1 | HALTM0 | RESCR | DRVE |
| | | | R/W | | | | | | | |
| | | | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: WDT enable | 00: $2^{15}/f_{SYS}$<br>01: $2^{17}/f_{SYS}$<br>10: $2^{19}/f_{SYS}$<br>11: $2^{21}/f_{SYS}$ | | Warm-up time<br>0: $2^{14}$/inputted frequency<br>1: $2^{16}$/inputted frequency | Standby mode<br>00: RUN mode<br>01: STOP mode<br>10: IDLE1 mode<br>11: IDLE2 mode | | 1: Connect internally WDT out to reset pin | 1: Drive the pin in STOP mode |
| WDCR | Watchdog timer control register | 5DH | – | | | | | | | |
| | | | W | | | | | | | |
| | | | – | | | | | | | |
| | | | B1H: WDT disable code 4EH: WDT clear code | | | | | | | |

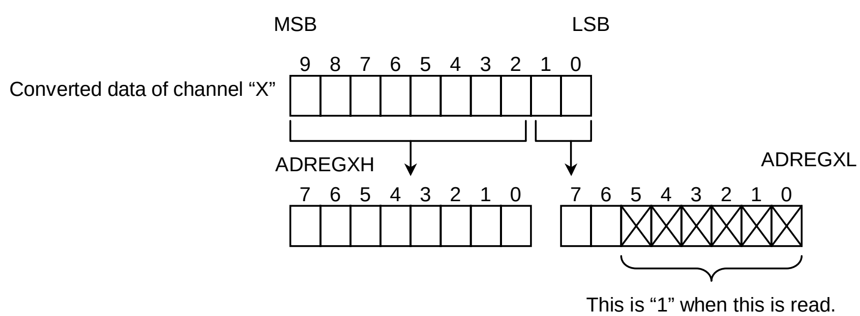
(6) Serial channel

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| SC0BUF | Serial channel 0 buffer | 50H | RB7<br>TB7 | RB6<br>TB6 | RB5<br>TB5 | RB4<br>TB4 | RB3<br>TB3 | RB2<br>TB2 | RB1<br>RB1 | RB0<br>TB0 |
| | | | R (Receiving)/W (Transmission) | | | | | | | |
| | | | Undefined | | | | | | | |
| SC0CR | Serial channel 0 control | 51H | RB8 | EVEN | PE | OERR | PERR | FERR | SCLKS | IOC |
| | | | R | R/W | | R (Cleared to 0 by reading) | | | R/W | |
| | | | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Receiving data bit8 | Parity<br>0: Odd<br>1: Even | 1: Parity enable | 1: Error<br>Overrun | Parity | Framing | 0: SCLK0 ↑<br>1: SCLK0 ↓ | 1: Input SCLK0 pin |
| SC0MOD | Serial channel 0 mode | 52H | TB8 | CTSE | RXE | WU | SM1 | SM0 | SC1 | SC0 |
| | | | R/W | | | | | | | |
| | | | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Transmission data bit8 | 1: CTS enable | 1: Receive enable | 1: Wake up enable | 00: I/O Interface<br>01: UART 7 bits<br>10: UART 8 bits<br>11: UART 9 bits | | 00: TO0 trigger<br>01: Baud rate generator<br>10: Internal clock $\phi$1<br>11: Don't care | |
| BR0CR | Baud rate control | 53H | – | | BR0CK1 | BR0CK0 | BR0S3 | BR0S2 | BR0S1 | BR0S0 |
| | | | R/W | | R/W | | | | | |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Always fixed to 0 | | 00: $\phi$T0<br>01: $\phi$T2<br>10: $\phi$T8<br>11: $\phi$T32 | | Set frequency divisor 0 to F<br>("1" prohibited) | | | |
| SC1BUF | Serial channel 1 buffer | 54H | RB7<br>TB7 | RB6<br>TB6 | RB5<br>TB5 | RB4<br>TB4 | RB3<br>TB3 | RB2<br>TB2 | RB1<br>RB1 | RB0<br>TB0 |
| | | | R (Receiving)/W (Transmission) | | | | | | | |
| | | | Undefined | | | | | | | |
| SC1CR | Serial channel 1 control | 55H | RB8 | EVEN | PE | OERR | PERR | FERR | SCLKS | IOC |
| | | | R | R/W | | R (Cleared to 0 by reading) | | | R/W | |
| | | | Undefined | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Receiving data bit8 | Parity<br>0: Odd<br>1: Even | 1: Parity enable | 1: Error<br>Overrun | Parity | Framing | 0:SCLK1 (↑)<br>1: SCLK1 (↓) | 1: Input SCLK1 pin |
| SC1MOD | Serial channel 1 mode | 56H | TB8 | – | RXE | WU | SM1 | SM0 | SC1 | SC0 |
| | | | R/W | | | | | | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Transmission data bit8 | Always fixed to 0 | 1: Receive enable | 1: Wake up enable | 00: I/O interface<br>01: UART 7 bits<br>10: UART 8 bits<br>11: UART 9 bits | | 00: TO0 Trigger<br>01: Baud rate generator<br>10: Internal clock $\phi$1<br>11: Don't care | |
| BR1CR | Baud rate control | 57H | – | | BR1CK1 | BR1CK0 | BR1S3 | BR1S2 | BR1S1 | BR1S0 |
| | | | R/W | | R/W | | | | | |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | Always fixed to 0 | | 00: $\phi$T0<br>01: $\phi$T2<br>10: $\phi$T8<br>11: $\phi$T32 | | Set frequency divisor 0 to F<br>("1" prohibited) | | | |
| ODE | Serial open-drain enable | 58H | | | | | | | ODE1 | ODE0 |
| | | | | | | | | | R/W | |
| | | | | | | | | | 0 | 0 |
| | | | | | | | | | 1: P93 Open drain | 1: P90 Open drain |

(7) AD converter control

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| ADMOD1 | AD mode register 1 | 5EH | EOCF | ADBF | REPET | SCAN | | ADS | | |
| | | | R | | R/W | | | R/W | | |
| | | | 0 | 0 | 0 | 0 | | 0 | | |
| | | | 1: End | 1: Busy | 1: Repeat | 1: Scan | | 1: Start | | |
| ADMOD2 | AD mode register 2 | 5FH | VREFON | | SPEED1 | SPEED0 | | ADCH2 | ADCH1 | ADCH0 |
| | | | R/W | | R/W | | | R/W | | |
| | | | 1 | | 0 | 0 | | 0 | 0 | 0 |
| | | | Ladder resistance switch ON/OFF | | Speed select | | | Analog input channel select | | |
| (Note 1) AD REG04L | AD result register 0/4 low | 60H | ADR01 | ADR00 | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| AD REG04H | AD result register 0/4 high | 61H | ADR09 | ADR08 | ADR07 | ADR06 | ADR05 | ADR04 | ADR03 | ADR02 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| (Note 1) AD REG15L | AD result register 1/5 low | 62H | ADR11 | ADR10 | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| AD REG15H | AD result register 1/5 high | 63H | ADR19 | ADR18 | ADR17 | ADR16 | ADR15 | ADR14 | ADR13 | ADR12 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| (Note 1) AD REG26L | AD result register 2/6 low | 64H | ADR21 | ADR20 | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| AD REG26H | AD result register 2/6 high | 65H | ADR29 | ADR28 | ADR27 | ADR26 | ADR25 | ADR24 | ADR23 | ADR22 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| (Note 1) AD REG37L | AD result register 3/7 low | 66H | ADR31 | ADR30 | | | | | | |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |
| AD REG37H | AD result register 3/7 high | 67H | ADR39 | ADR38 | ADR37 | ADR36 | ADR35 | ADR34 | ADR33 | ADR32 |
| | | | R | | | | | | | |
| | | | Undefined | | | | | | | |

Note 1: Data to be stored in AD conversion result register low are the lower 2 bits of the conversion result. The contents of the lower 6 bits of this register are always read as “1”.



(8) Interrupt control (1/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| INTE0AD | Interrupt enable 0 & AD | 70H (Prohibit RMW) | INTAD | | | | INT0 | | | |
| | | | IADC | IADM2 | IADM1 | IADM0 | I0C | I0M2 | I0M1 | I0M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTE45 | Interrupt enable 4/5 | 71H (Prohibit RMW) | INT5 | | | | INT4 | | | |
| | | | I5C | I5M2 | I5M1 | I5M0 | I4C | I4M2 | I4M1 | I4M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTE67 | Interrupt enable 6/7 | 72H (Prohibit RMW) | INT7 | | | | INT6 | | | |
| | | | I7C | I7M2 | I7M1 | I7M0 | I6C | I6M2 | I6M1 | I6M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTET10 | Interrupt enable timer 1/0 | 73H (Prohibit RMW) | INTT1 (Timer 1) | | | | INTT0 (Timer 0) | | | |
| | | | IT1C | IT1M2 | IT1M1 | IT1M0 | IT0C | IT0M2 | IT0M1 | IT0M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTEPW10 | Interrupt enable PWM 1/0 | 74H (Prohibit RMW) | INTT3 (Timer 3/PWM1) | | | | INTT2 (Timer 2/PWM0) | | | |
| | | | IPW1C | IPW1M2 | IPW1M1 | IPW1M0 | IPW0C | IPW0M2 | IPW0M1 | IPW0M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTET54 | Interrupt enable T register 5/4 | 75H (Prohibit RMW) | INTTR5 (TREG5) | | | | INTTR4 (TREG4) | | | |
| | | | IT5C | IT5M2 | IT5M1 | IT5M0 | IT4C | IT4M2 | IT4M1 | IT4M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTET76 | Interrupt enable T register 7/6 | 76H (Prohibit RMW) | INTTR7 (TREG7) | | | | INTTR6 (TREG6) | | | |
| | | | IT7C | IT7M2 | IT7M1 | IT7M0 | IT6C | IT6M2 | IT6M1 | IT6M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| INTES0 | Interrupt enable serial 0 | 77H (Prohibit RMW) | INTTX0 | | | | INTRX0 | | | |
| | | | ITX0C | ITX0M2 | ITX0M1 | ITX0M0 | IRX0C | IRX0M2 | IRX0M1 | IRX0M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| INTES1 | Interrupt enable serial 1 | 78H (Prohibit RMW) | INTTX1 | | | | INTRX1 | | | |
| | | | ITX1C | ITX1M2 | ITX1M1 | ITX1M0 | IRX1C | IRX1M2 | IRX1M1 | IRX1M0 |
| | | | R/W | W | | | R/W | W | | |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| IxxM2 | IxxM1 | IxxM0 | Function (Write) |
|---|---|---|---|
| 0 | 0 | 0 | Prohibit interrupt request. |
| 0 | 0 | 1 | Set interrupt request level to “1”. |
| 0 | 1 | 0 | Set interrupt request level to “2”. |
| 0 | 1 | 1 | Set interrupt request level to “3”. |
| 1 | 0 | 0 | Set interrupt request level to “4”. |
| 1 | 0 | 1 | Set interrupt request level to “5”. |
| 1 | 1 | 0 | Set interrupt request level to “6”. |
| 1 | 1 | 1 | Prohibit interrupt request. |

| IxxC | Function (Read) | Function (Write) |
|---|---|---|
| 0 | Indicate no interrupt request. | Clear interrupt request flag. |
| 1 | Indicate interrupt request. | – – – – – Don’t care – – – – – |

Interrupt control (2/2)

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| DMA0V | DMA 0 request vector | 7CH (Prohibit RMW) | | | | Micro DMA0 start vector | | | | |
| | | | | | | DMA0V4 | DMA0V3 | DMA0V2 | DMA0V1 | DMA0V0 |
| | | | | | | W | | | | |
| | | | | | | 0 | 0 | 0 | 0 | 0 |
| DMA1V | DMA 1 request vector | 7DH (Prohibit RMW) | | | | Micro DMA1 start vector | | | | |
| | | | | | | DMA1V4 | DMA1V3 | DMA1V2 | DMA1V1 | DMA1V0 |
| | | | | | | W | | | | |
| | | | | | | 0 | 0 | 0 | 0 | 0 |
| DMA2V | DMA 2 request vector | 7EH (Prohibit RMW) | | | | Micro DMA2 start vector | | | | |
| | | | | | | DMA2V4 | DMA2V3 | DMA2V2 | DMA2V1 | DMA2V0 |
| | | | | | | W | | | | |
| | | | | | | 0 | 0 | 0 | 0 | 0 |
| DMA3V | DMA 3 request vector | 7FH (Prohibit RMW) | | | | Micro DMA3 start vector | | | | |
| | | | | | | DMA3V4 | DMA3V3 | DMA3V2 | DMA3V1 | DMA3V0 |
| | | | | | | W | | | | |
| | | | | | | 0 | 0 | 0 | 0 | 0 |
| IIMC | Interrupt input mode control | 7BH (Prohibit RMW) | | | | | | I0IE | I0LE | NMIREE |
| | | | | | | | | W | W | W |
| | | | | | | | | 0 | 0 | 0 |
| | | | | | | | | 0: INT0 input enable | 0: INT0 edge mode<br>1: INT0 level mode | 0: Operation even at $\overline{\text{NMI}}$ rising edge |

(9) Chip select/wait controller

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| B0CS | Block 0 CS/WAIT control register | 68H (Prohibit RMW) | B0E | | B0CAS | B0BUS | B0W1 | B0W0 | B0C1 | B0C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: B0CS master bit | | 0: $\overline{CS0}$<br>1: $\overline{CAS0}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 7F00H to 7FFFH<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |
| B1CS | Block 1 CS/WAIT control register | 69H (Prohibit RMW) | B1E | | B1CAS | B1BUS | B1W1 | B1W0 | B1C1 | B1C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: B1CS master bit | | 0: $\overline{CS1}$<br>1: $\overline{CAS1}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 880H to 7FFFH<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |
| B2CS | Block 2 CS/WAIT control register | 6AH (Prohibit RMW) | B2E | | B2CAS | B2BUS | B2W1 | B2W0 | B2C1 | B2C0 |
| | | | W | | W | W | W | W | W | W |
| | | | 1 | | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 1: B2CS master bit | | 0: $\overline{CS2}$<br>1: $\overline{CAS2}$ | 0: 16-bit bus<br>1: 8-bit bus | 00: 2 waits<br>01: 1 wait<br>10: (1 + n) waits<br>11: 0 waits | | 00: 8000H to<br>01: 400000H to<br>10: 800000H to<br>11: C00000H to | |

Note: After reset, only “Block 2” is set to enable.

(10) Clock control

| Symbol | Name | Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| CKOCR | Clock output control register | 006DH | | | | | SCOSEL | SCOEN | ALEEN | CLKEN |
| | | | | | | | R/W | | | |
| | | | | | | | 0 | 0 | 0/1 (Note 1) | 0/1 (Note 1) |
| | | | | | | | SCOUT select<br>0: $f_{FPH}$<br>1: $f_{SYS}$ | SCOUT output control<br>0: I/O port<br>1: SCOUT output | ALE pin control<br>0:High-Z output<br>1: ALE output | CLK pin control<br>0:High-Z output<br>1: CLK output |
| SYSCR0 | System clock control register 0 | 006EH | XEN | XTEN | RXEN | RXTEN | RSYSCK | WUEF | PRCK1 | PRCK0 |
| | | | R/W | | | | | | | |
| | | | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | High-frequency oscillator (fc)<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs)<br>0: Stop<br>1: Oscillation | High-frequency oscillator (fc) after release of STOP mode<br>0: Stop<br>1: Oscillation | Low-frequency oscillator (fs) after release of STOP mode<br>0: Stop<br>1: Oscillation | Selected clock after Release of Stop mode<br>0: fc<br>1: fs | Warm-up timer<br>0 Write: Don't care<br>1 Write: Start timer<br>0 Read: End warm up<br>1 Read: Continue warm up | Select prescaler clock<br>00: $f_{FPH}$<br>01: fs<br>10: fc/16<br>11: (Reserved) | |
| SYSCR1 | System clock control register 1 | 006FH | | | | | SYSCK | GEAR2 | GEAR1 | GEAR0 |
| | | | | | | | R/W | | | |
| | | | | | | | 0 | 1 | 0 | 0 |
| | | | | | | | Select system clock<br>0: fc<br>1: fs (Note2) | 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) | | |

Note 1: The value after reset of <CLKEN>, <ALEEN> is as follows:
TMP93CS40: 0 (High impedance output)
TMP93CS41: 1 (CLK or ALE output)
However, during reset the CLK pin is pulled up internally on both models.

Note 2: The high-frequency oscillator will be enabled when SYSCR1<SYSCK> is set to 0 regardless of the value of SYSCR0<XEN>.
The low-frequency oscillator will be enabled when SYSCR1<SYSCK> is set to 1 regardless of the value of SYSCR0<XTEN>.

# 6. Port Section Equivalent Circuit Diagram

- Reading the circuit diagram

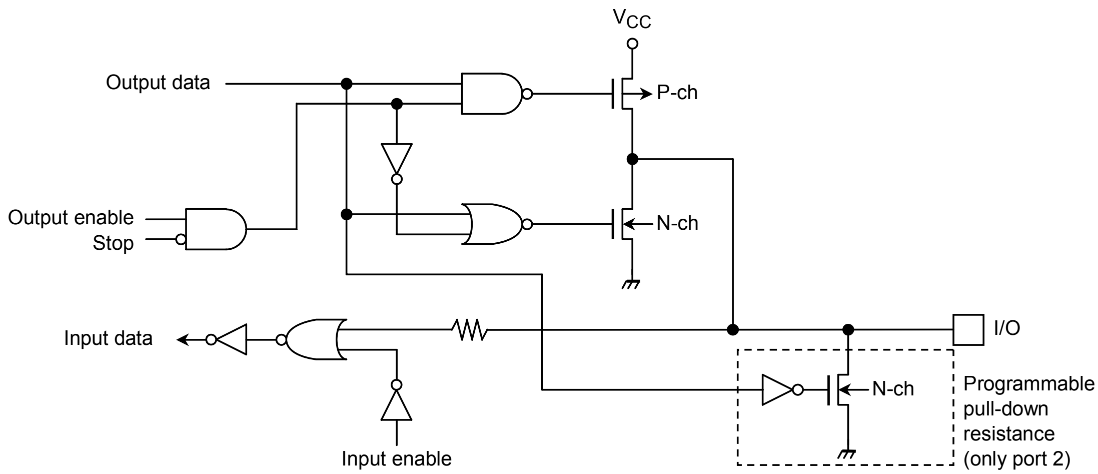
  Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

  The dedicated signal is described below.

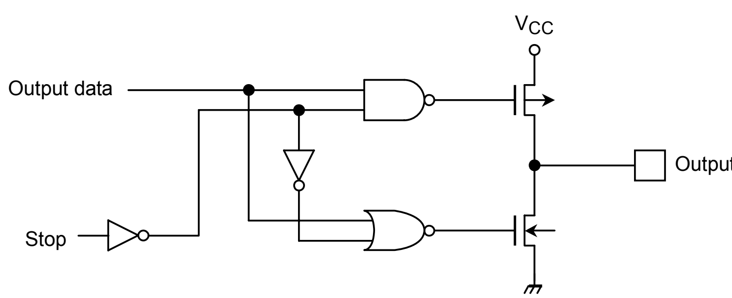
  Stop: This signal becomes active "1" when the HALT mode setting register is set to the STOP mode and the CPU executes the HALT instruction. When the drive enable bit [DRVE] is set to "1", however, stop remains at "0".

- The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

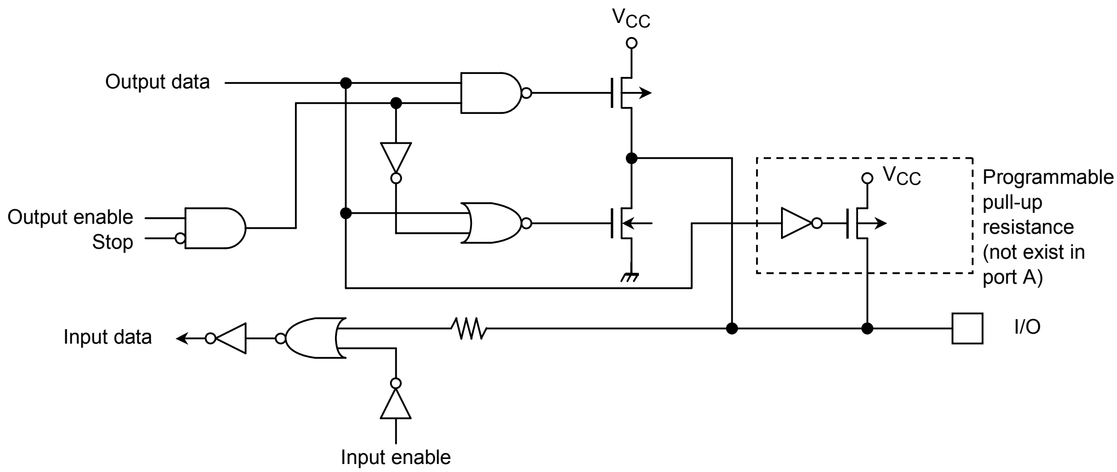
■ P0 (AD0 to AD7), P1 (AD8 to AD15, A8 to A15), P2 (A16 to A23, A0 to A7)



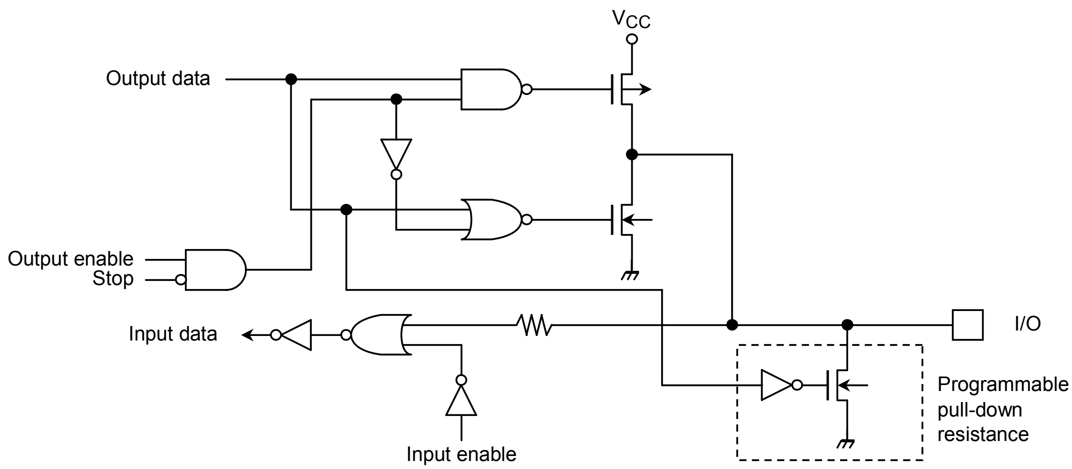

■ P30 ($\overline{RD}$), P31 ($\overline{WR}$)



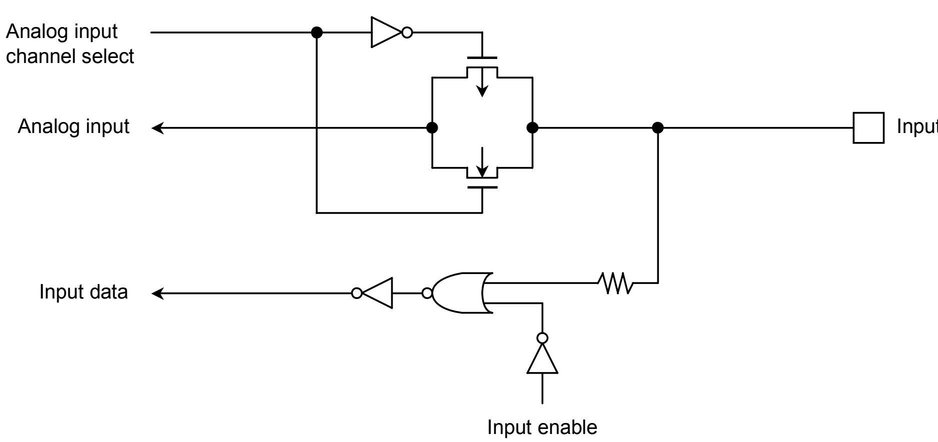
■ P32 to P37, P40 to P41, P6, P7, P80 to P86, P91 to P92, P94 to P95, PA



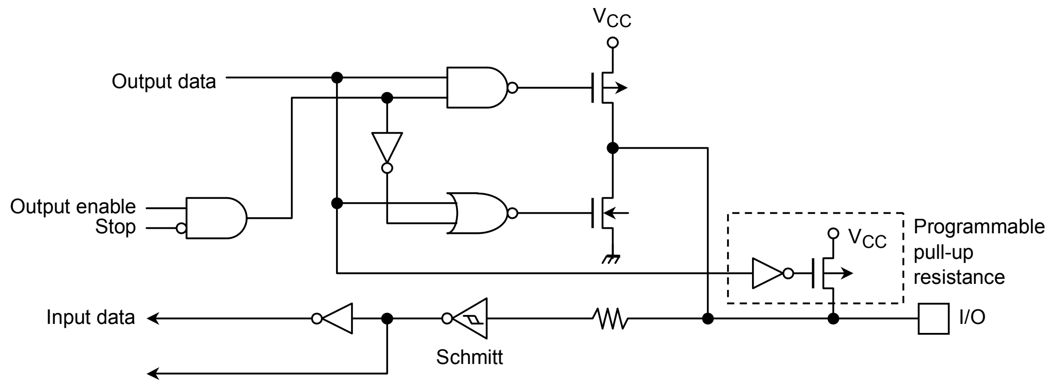

■ P42 ( $\overline{CS2}$ , $\overline{CAS2}$ )



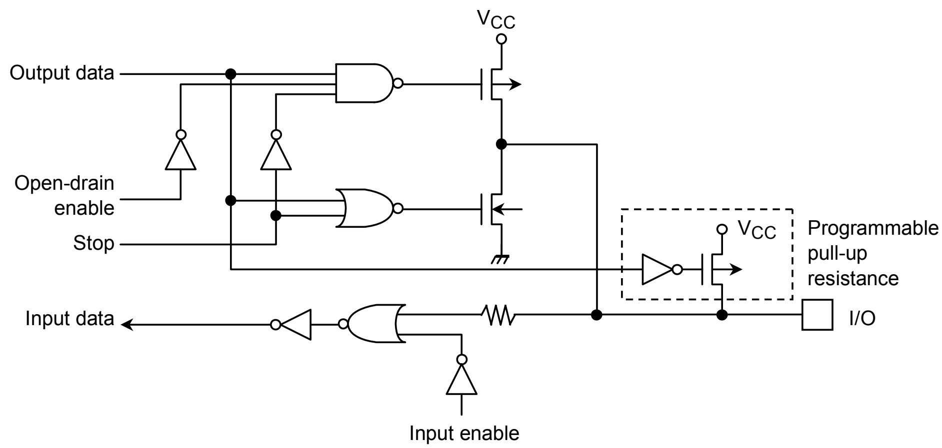

■ P5 (AN0 to AN7)



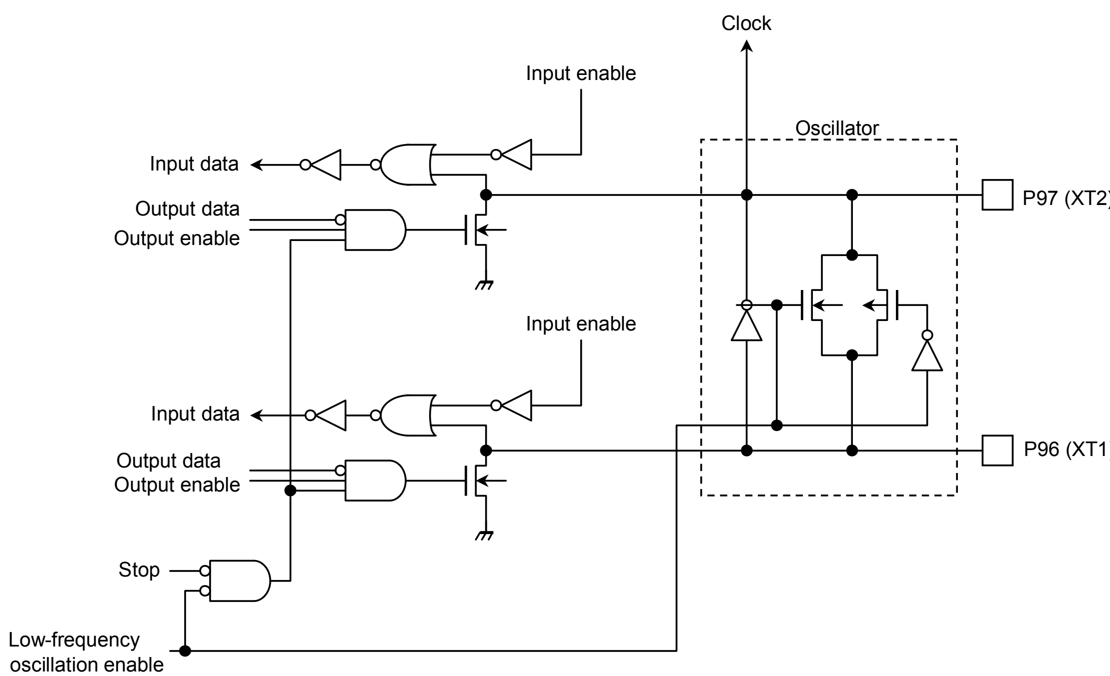
■ P87 (INT0)



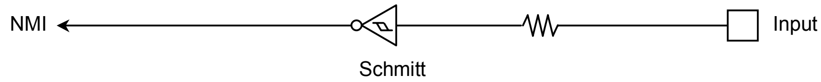

■ P90 (TXD0), P93 (TXD1)



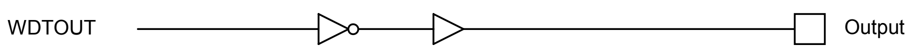

■ P96 (XT1), P97 (XT2)



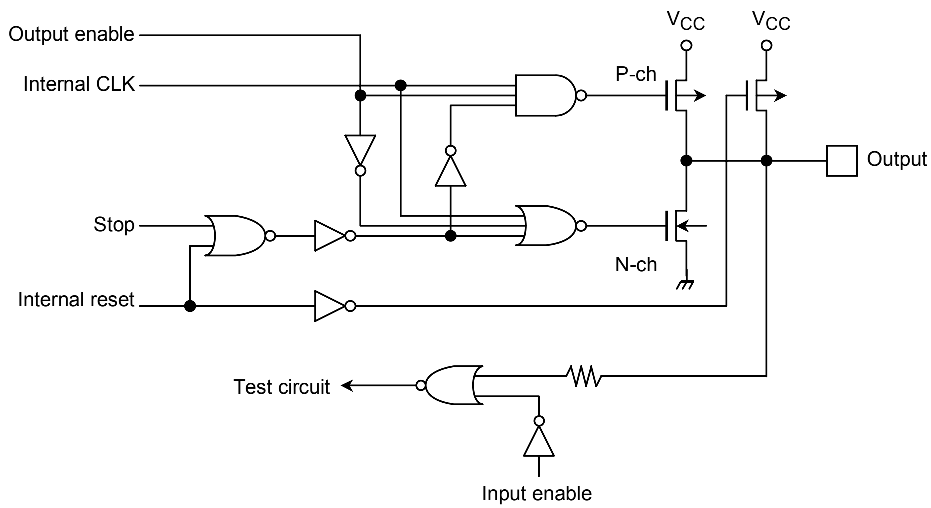
- $\overline{\text{NMI}}$



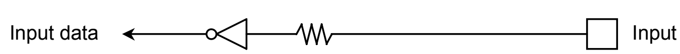

- $\overline{\text{WDTOUT}}$



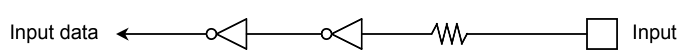

- CLK



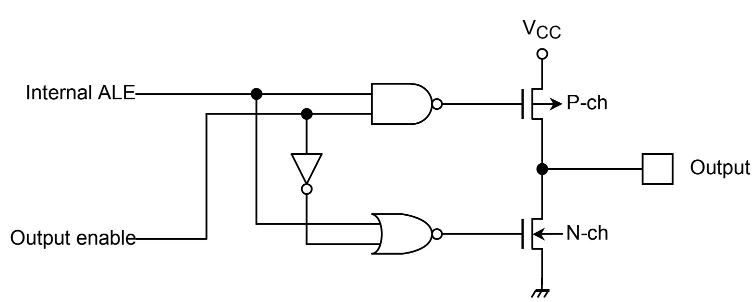

- $\overline{\text{EA}}$



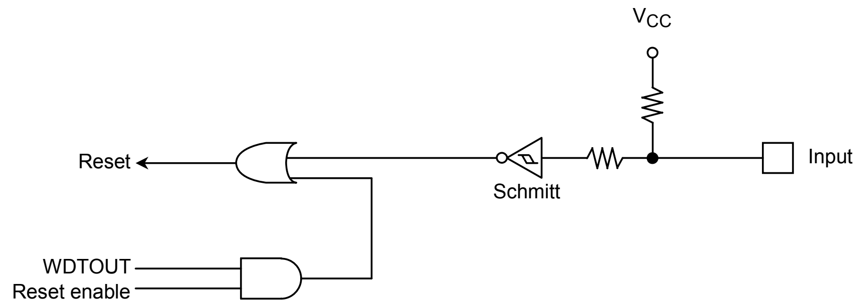

- AM8/ $\overline{\text{AM16}}$



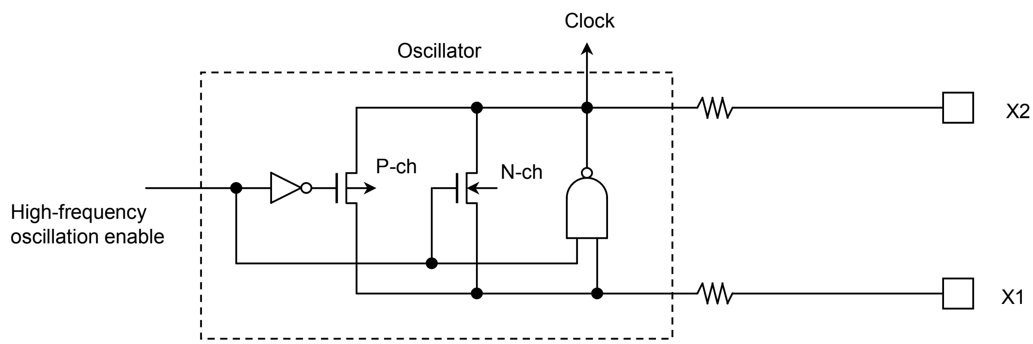

- ALE



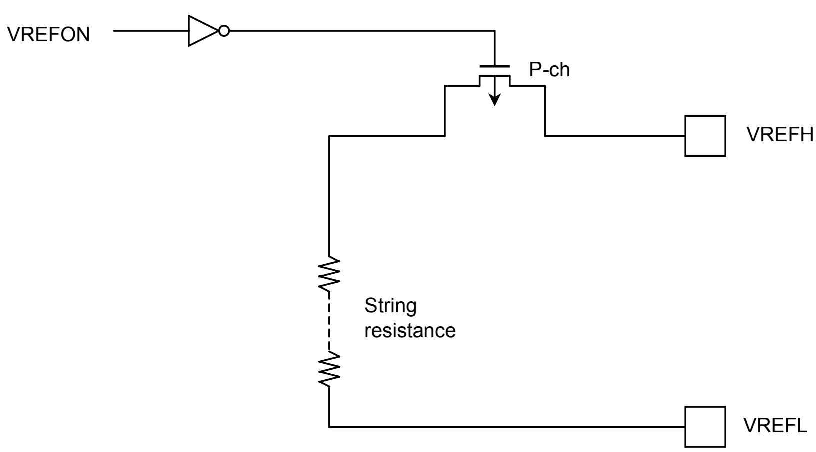
■ $\overline{\text{RESET}}$




■ X1, X2




■ VREFH, VREFL



# 7. Points of Note and Restrictions

(1) Notation

1. How a built-in I/O register is denoted: Register symbol<Bit symbol>

   e.g.) TRUN<T0RUN> … Bit T0RUN of register TRUN

2. Read-modify-write instruction

   An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

   Example 1: SET 3, (TRUN) … Set bit3 of TRUN

   Example 2: INC 1, (100H) … Increment the data at 100H

- A sample read-modify-write instructions using the TLCS-900

```
Exchange instruction
    EX    (mem), R
Arithmetic operation
    ADD   (mem), R/#        ADC  (mem), R/#
    SUB   (mem), R/#        SBC  (mem), R/#
    INC   #3, (mem)         DEC  #3, (mem)
Logic operation
    AND   (mem), R/#        OR   (mem), R/#
    XOR   (mem), R/#
Bit manipulation
    STCF  #3/A, (mem)       RES  #3, (mem)
    SET   #3, (mem)         CHG  #3, (mem)
    TSET  #3, (mem)
Rotate, Shift
    RLC   (mem)             RRC  (mem)
    RL    (mem)             RR   (mem)
    SLA   (mem)             SRA  (mem)
    SLL   (mem)             SRL  (mem)
    RLD   (mem)             RRD  (mem)
```

3. fc, $f_{FPH}$, $f_{SYS}$, one state

   The clock frequency input from pins X1 and X2 is called fc, the clock selected by SYSCR<SYSCK> is called $f_{FPH}$, and the clock frequency given by $f_{FPH}$ divided by 2 is called $f_{SYS}$. One cycle of $f_{SYS}$ is called one state.

(2) Points to note

1. $\overline{\text{EA}}$, AM8/$\overline{\text{AM16}}$ pin

   Fix these pins to $V_{CC}$ or GND unless changing voltage.

2. TEST1, TEST2 pin

   Connect the TEST1 pin with the TEST2 pin. Do not connect to any other pins.

3. Reserved area in memory space

   The 256 bytes of memory between FFFF00H and FFFFFFH cannot be used because they are reserved.

4. Standby mode (IDLE1)

   When IDLE1 mode (Oscillator operation only) is used, set TRUN<PRRUN> to 0 to stop the prescaler before the HALT instruction is executed.

5. 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 warm-up time elapses from input of the release request to output of the system clock.

6. Micro DMA (DRAM refresh mode)

   When the bus is released ($\overline{\text{BUSAK}}$ = 0). DRAM refresh cannot be performed because the micro DMA cannot access the bus.

7. Programmable pull-up/pull-down resistance

   The programmable pull-up/pull-down resistors can be turned ON/OFF by the program when the ports are used as input ports. When the ports are used as outputs, they cannot be turned ON/OFF by the program.

   The data registers (e.g., P2, P3, …) are used for the pull-up/pull-down resistors ON/OFF. Consequently read-modify-write instructions are prohibited.

8. Bus release function

   Refer to the note about the bus release in 3.5 “Functions of Ports” as it describes the state of the pins when the bus is released.

9. Watchdog timer

   The watchdog timer starts operation immediately after the reset is released. When the watch dog timer will not be used, disable it.

10. Watchdog timer

    When the bus is released, neither internal memory nor internal I/O can be accessed. However, the internal I/O continues to operate and hence, the watchdog timer continues to run. Thus, take care when setting the bus release time and the detection timer for the watchdog timer.

11. AD converter

    The ladder resistor between the VREFH and VREFL pins can be cut by a program to reduce power consumption. When standby mode is used, disable the resistor using the program before the HALT instruction is executed. And set ADMOD2<SPEED1:0> = “00”.

12. CPU (Micro DMA)

    Only the “LDC cr, r” and “LDC r, cr” instructions can be used to access the control registers in the CPU (like the transfer source address register (DMASn)).

13. POP SR instruction

    Please execute POP SR instruction during DI condition.

14. Pin states in STOP mode

    Open-drain output state. Input gate in operation. Set output to “L” or attach pull up on pin so that the input gate stays constant.

15. Releasing the HALT mode by requesting an interruption

Usually, interrupts can release all halts status. However, the interrupts ($\overline{\text{NMI}}$, INT0) which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of $f_{FPH}$) with IDLE1 or STOP mode (IDLE2/RUN are not applicable to this case). (In this case, an interrupt request is kept on hold internally.)

If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

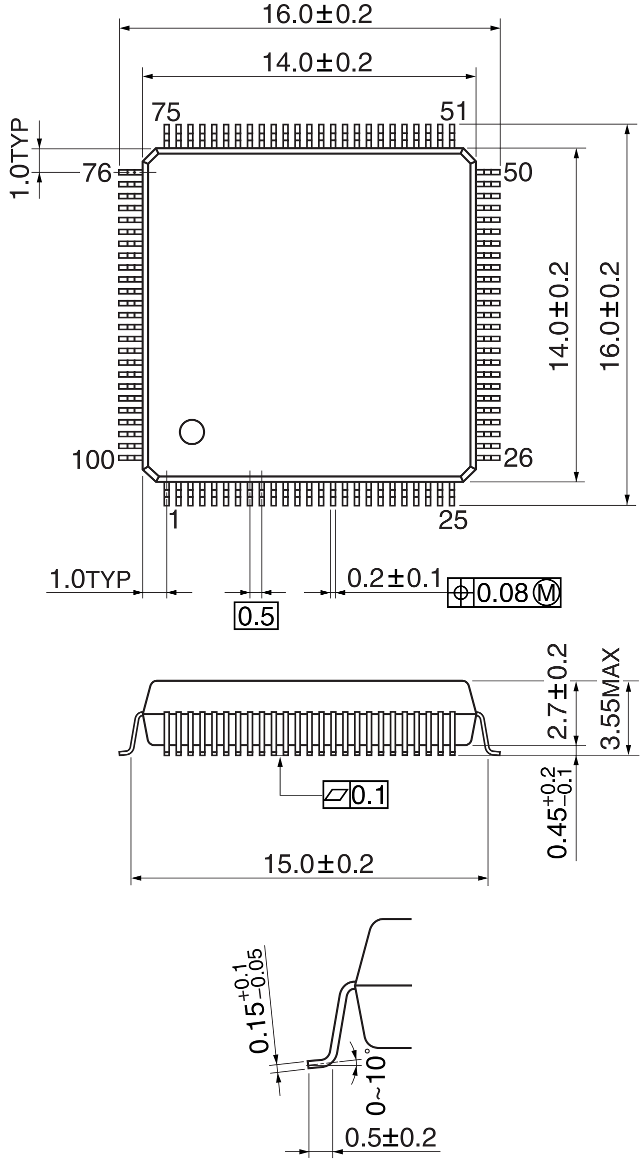
# 8. TMP93XX40/41 Different Points

| Item | TMP93CM40F | TMP93CS40F | TMP93CS41F | TMP93PS40F | TMP93CW40F | TMP93CW41F | TMP93PW40F |
|---|---|---|---|---|---|---|---|
| Built-in ROM | 32-Kbyte mask ROM (8000H to FFFFH) | 64-Kbyte mask ROM (8000H to 17FFFH) | None | 64-Kbyte OTP (8000H to 17FFFH) | 128-Kbyte mask ROM (8000H to 27FFFH) | None | 128-Kbyte OTP (8000H to 27FFFH) |
| Built-in RAM | 2-Kbyte (0080H to 087FH) | | | | 4-Kbyte (0080H to 107FH) | | |
| Operation frequency: fc (3 V ± 10%) | 4 to 10 MHz | 4 to 12.5 MHz | | | | | |
| ADC operation voltage range | 5 V ± 10% (4 to 20 MHz) | 5 V ± 10% (4 to 20 MHz)<br>3 V ± 10% (4 to 12.5 MHz) | | | | | |
| Port 5 input level (VIL) | 0.2 $V_{CC}$ | 0.3 $V_{CC}$ | | | | | |
| CS2 Mapping area (B2CS < B2C1 to 0 ≥ 00) | from 10000H | from 18000H | from 08000H | from 18000H | from 28000H | from 08000H | from 28000H |
| CS1 Mapping area (B1CS < B1C1 to 0 ≥ 00) | 880H to 7FFFH | | | | 1080H to 7FFFH | | |

# 9. Package Dimensions

P-QFP100-1414-0.50

Unit: mm



P-LQFP100-1414-0.50F

Unit: mm