TOSHIBA

TOSHIBA Original CMOS 16-Bit Microcontroller

TLCS-900/L1 Series

TMP91CW28

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.

Low-Voltage, Low-Power

CMOS 16-Bit Microcontroller

TMP91CW28FG

1. Outline

The TMP91CW28 is a high-speed, high-performance 16-bit microcontroller suitable for low-voltage, low-power applications.

The TMP91CW28FG comes in a 100-pin mini flat package. Features of the TMP91CW28FG include the following:

(1) High-speed 16-bit CPU (900/L1 CPU)
- Instruction set is upwardly assembly code compatible with the TLCS-90
- 16-Mbyte linear address space
- Architecture based on general-purpose registers and register banks
- 16-bit multiply/divide instructions and bit transfer/arithmetic instructions
- 4-channel micro DMA (1.6 μs/2 bytes at 10 MHz)

(2) Minimum instruction execution time: 400 ns (at 10 MHz)

(3) 8-Kbyte on-chip RAM
128-Kbyte on-chip ROM

(4) External memory expansion
- 16-Mbyte off-chip address space for code and data
- External bus interface with dynamic bus sizing for 8-bit and 16-bit data ports

(5) 4-channel 8-bit timer

(6) 2-channel 16-bit timer

(7) 1-channel general-purpose serial interface
- Both UART and synchronous transfer modes are supported

(8) 2-channel serial bus interface
Either I^{2}C bus mode or clock-synchronous mode can be selected

(9) 8-channel 10-bit AD converter (with internal sample/hold)

(10) Watchdog timer

(11) Key wakeup interrupt with 8-bit inputs

(12) WAKE output pin

(13) BCD adder/subtractor

(14) Program patch logic

- 6 banks of registers

(15) 4-channel chip select/wait controller

(16) 48 interrupt sources

- 9 CPU interrupts: Triggered by software interrupt instruction or upon the execution of an undefined instruction
- 21 internal interrupts: 7 priority levels
- 18 external interrupts: 7 priority levels (16 interrupts supporting selection of triggering edge)

(17) 80-pin input/output ports

(18) Standby modes

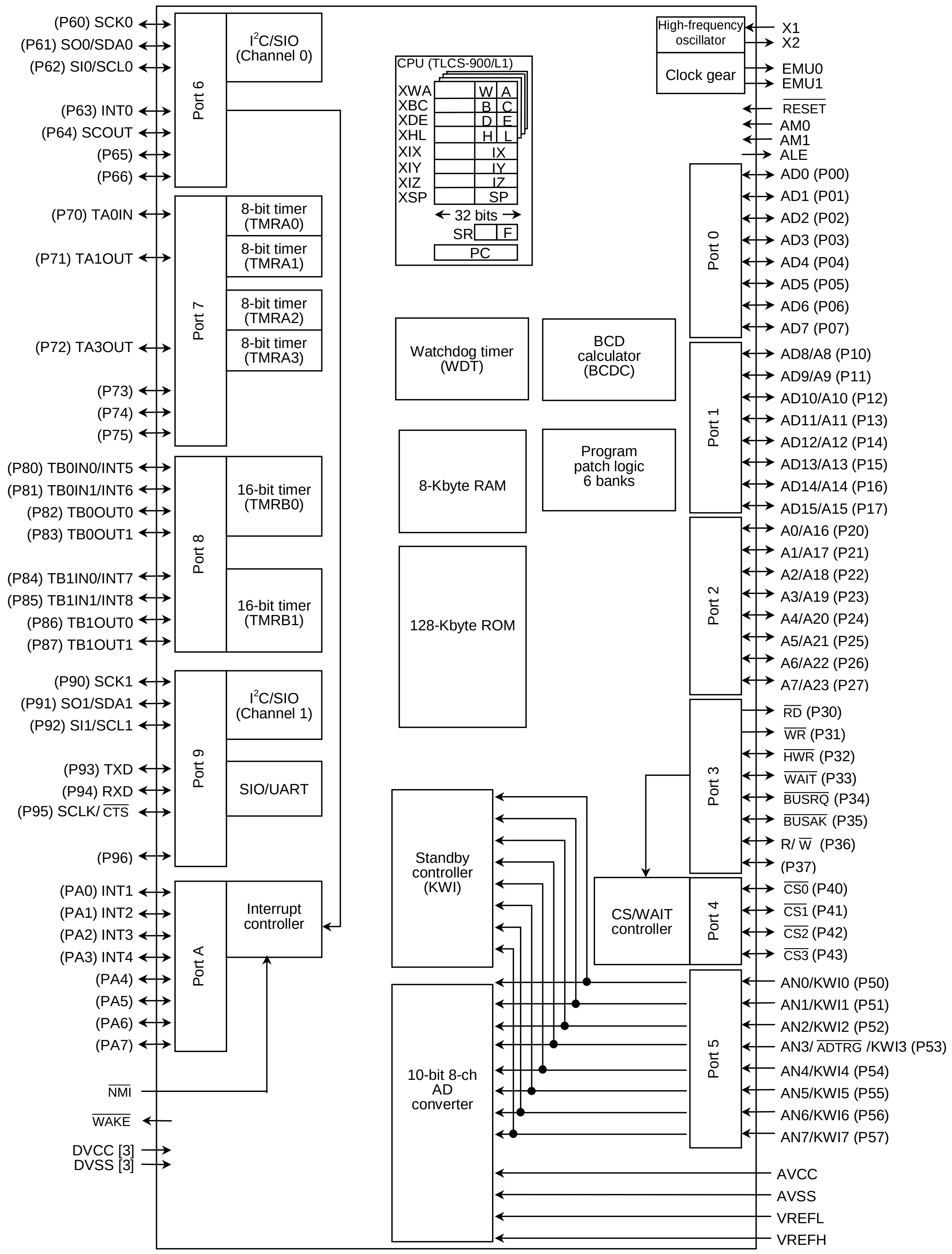
- Three HALT modes: Programmable IDLE2, IDLE1, STOP

(19) Clock control

- Clock gear: Changes the frequency of high-frequency clock within the range from fc to fc/16

(20) Operating voltage range: 1.8 to 2.6 V (fc max = 10 MHz)

(21) Package: P-LQFP100-1414-0.50F



( ): Initial pin function after reset

Figure 1.1 TMP91CW28 Block Diagram

2. Signal Descriptions

This section contains pin assignments for the TMP91CW28 as well as brief descriptions of the TMP91CW28 input and output signals.

2.1 Pin Assignment

The following illustrates the TMP91CW28FG pin assignment.

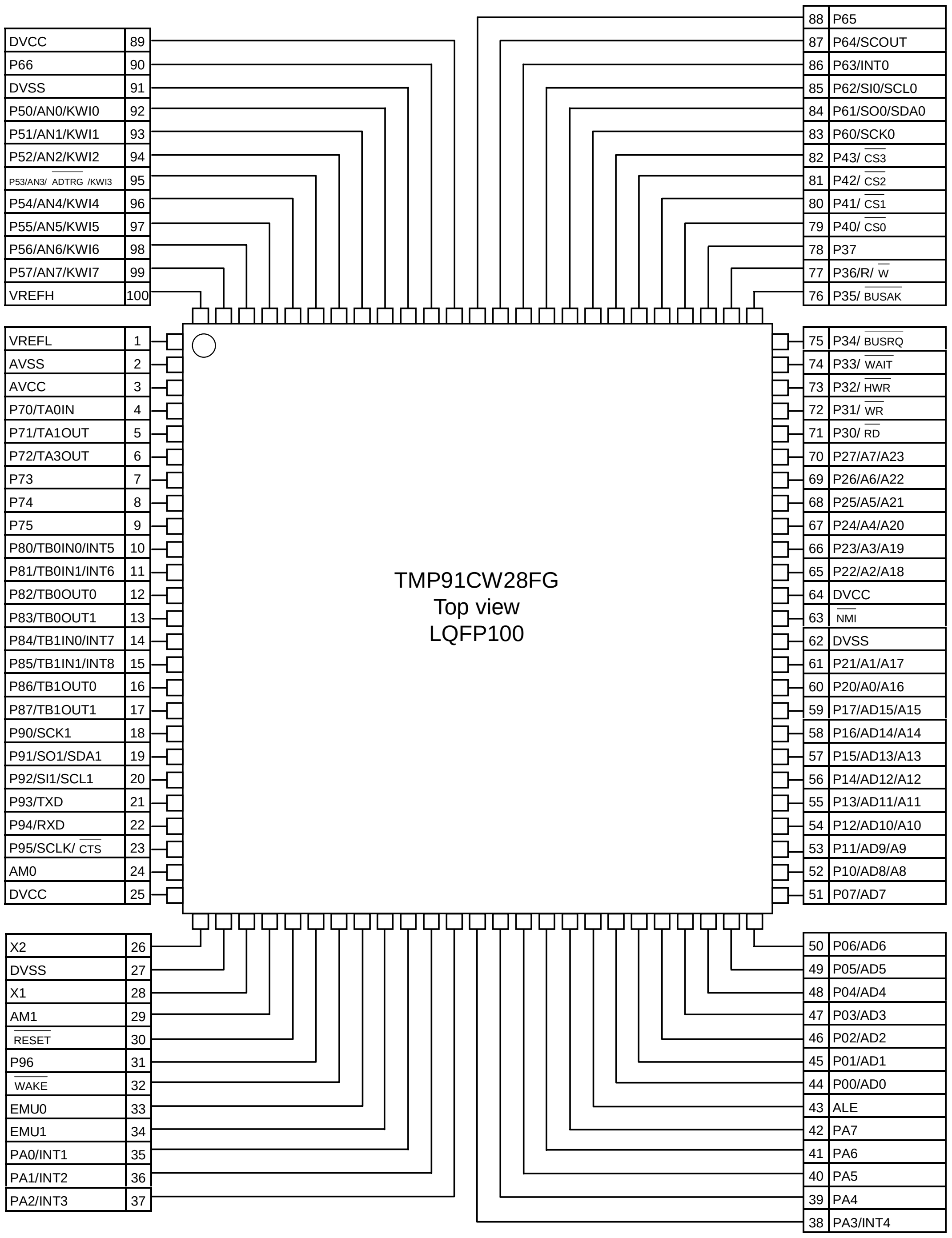


Figure 2.1.1 100-Pin LQFP Pin Assignment

2.2 Pin Usage Information

Table 2.2.1 to 2.2.4 list the input and output pins of the TMP91CW28, including alternate pin names and functions for multi-function pins.

Table 2.2.1 Pin Names and Functions (1/4)

Pin Name	Number of Pins	I/O	Functions
P00 to P07 AD0 to AD7	8	I/O I/O	Port 0: Individually programmable as input or output Address/data (Lower): Bits 0 to 7 of the address/data bus
P10 to P17 AD8 to AD15 A8 to A15	8	I/O I/O Output	Port 1: Individually programmable as input or output Address/data (Upper): Bits 8 to 15 of the address/data bus Address: Bits 8 to 15 of the address bus
P20 to P27 A0 to A7 A16 to A23	8	I/O Output Output	Port 2: Individually programmable as input or output Address: Bits 0 to 7 of the address bus Address: Bits 16 to 23 of the address bus
P30 $\overline{\text{RD}}$	1	Output Output	Port 30: Output only Read strobe: Asserted during a read operation from an external memory device Also asserted during a read from internal memory if P3.P30 = 0 and P3FC.P30F = 1
P31 $\overline{\text{WR}}$	1	Output Output	Port 31: Output only Write strobe: Asserted during a write operation on AD0 to AD7
P32 $\overline{\text{HWR}}$	1	I/O Output	Port 32: Programmable as input or output (with internal pull-up resistor) Higher write strobe: Asserted during a write operation on AD8 to AD15
P33 $\overline{\text{WAIT}}$	1	I/O Input	Port 33: Programmable as input or output (with internal pull-up resistor) Wait: Causes the CPU to suspend external bus activity ((1 + N) wait states)
P34 $\overline{\text{BUSRQ}}$	1	I/O Input	Port 34: Programmable as input or output (with internal pull-up resistor) Bus request: Asserted to request that the AD0 to AD15, A0 to A23, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{HWR}}$, R/$\overline{\text{W}}$, and $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ pins be placed in high-impedance state (for external DMAC)
P35 $\overline{\text{BUSAK}}$	1	I/O Output	Port 35: Programmable as input or output (with internal pull-up resistor) Bus acknowledge: Indicates that the AD0 to AD15, A0 to A23, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{HWR}}$, R/$\overline{\text{W}}$, and $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ pins have been placed in high-impedance state in response to $\overline{\text{BUSRQ}}$ (for external DMAC)
P36 R/$\overline{\text{W}}$	1	I/O Output	Port 36: Programmable as input or output (with internal pull-up resistor) Read/write: Indicates the direction of data transfer on the bus: 1 = Read or dummy cycle, 0 = Write cycle
P37	1	I/O	Port 37: Programmable as input or output (with internal pull-up resistor)
P40 $\overline{\text{CS0}}$	1	I/O Output	Port 40: Programmable as input or output (with internal pull-up resistor) Chip select 0: Asserted low to enable external devices at programmed addresses

Note: An external DMA controller configured with the $\overline{\text{BUSRQ}}$ and $\overline{\text{BUSAK}}$ pins cannot access the on-chip memory and peripheral functions of the TMP91CW28.

Table 2.2.2 Pin Names and Functions (2/4)

Pin Name	Number of Pins	I/O	Functions
P41 $\overline{\text{CS1}}$	1	I/O Output	Port 41: Programmable as input or output (with internal pull-up resistor) Chip select 1: Asserted low to enable external devices at programmed addresses
P42 $\overline{\text{CS2}}$	1	I/O Output	Port 42: Programmable as input or output (with internal pull-up resistor) Chip select 2: Asserted low to enable external devices at programmed addresses
P43 $\overline{\text{CS3}}$	1	I/O Output	Port 43: Programmable as input or output (with internal pull-up resistor) Chip select 3: Asserted low to enable external devices at programmed addresses
P50 to P57 AN0 to AN7 $\overline{\text{ADTRG}}$ KWI0 to KWI7	8	Input Input Input Input	Port 5: Input only Analog input: Input to the on-chip AD converter AD trigger: Starts an AD conversion (multiplexed with P53) Key wakeup input (multiplexed with P50 to P57)
P60 SCK0	1	I/O I/O	Port 60: Programmable as input or output Clock input/output pin when serial bus interface 0 is in SIO mode
P61 SO0 SDA0	1	I/O Output I/O	Port 61: Programmable as input or output (with internal pull-up resistor) Data transmit pin when serial bus interface 0 is in SIO mode Data transmit/receive pin when serial bus interface 0 is in I^2C mode; programmable as an open-drain output
P62 SI0 SCL0	1	I/O Input I/O	Port 62: Programmable as input or output (with internal pull-up resistor) Data receive pin when serial bus interface 0 is in SIO mode Clock input/output pin when serial bus interface 0 is in I^2C mode; programmable as an open-drain output
P63 INT0	1	I/O Input	Port 63: Programmable as input or output Interrupt request 0: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
P64 SCOUT	1	I/O Output	Port 64: Programmable as input or output System clock output: Drives out f_{FPH} clock
P65	1	I/O	Port 65: Programmable as input or output
P66	1	I/O	Port 66: Programmable as input or output
P70 TA0IN	1	I/O Input	Port 70: Programmable as input or output (with internal pull-up resistor) 8-bit timer 0 input: Input to timer 0
P71 TA1OUT	1	I/O Output	Port 71: Programmable as input or output (with internal pull-up resistor) 8-bit timer 1 output: Output from either timer 0 or timer 1
P72 TA3OUT	1	I/O Output	Port 72: Programmable as input or output (with internal pull-up resistor) 8-bit timer 3 output: Output from either timer 2 or timer 3

Table 2.2.3 Pin Names and Functions (3/4)

Pin Name	Number of Pins	I/O	Functions
P73	1	I/O	Port 73: Programmable as input or output (with internal pull-up resistor)
P74	1	I/O	Port 74: Programmable as input or output (with internal pull-up resistor)
P75	1	I/O	Port 75: Programmable as input or output (with internal pull-up resistor)
P80 TB0IN0 INT5	1	I/O Input Input	Port 80: Programmable as input or output (with internal pull-up resistor) 16-bit timer 0 input 0: Count/capture trigger input to 16-bit timer 0 Interrupt request 5: Programmable to be rising-edge or falling-edge sensitive
P81 TB0IN1 INT6	1	I/O Input Input	Port 81: Programmable as input or output (with internal pull-up resistor) 16-bit timer 0 input 1: Count/capture trigger input to 16-bit timer 0 Interrupt request 6: Rising-edge sensitive
P82 TB0OUT0	1	I/O Output	Port 82: Programmable as input or output (with internal pull-up resistor) 16-bit timer 0 output 0: Output from 16-bit timer 0
P83 TB0OUT1	1	I/O Output	Port 83: Programmable as input or output (with internal pull-up resistor) 16-bit timer 0 output 1: Output from 16-bit timer 0
P84 TB1IN0 INT7	1	I/O Input Input	Port 84: Programmable as input or output (with internal pull-up resistor) 16-bit timer 1 input 0: Count/capture trigger input to 16-bit timer 1 Interrupt request 7: Programmable to be rising-edge or falling-edge sensitive
P85 TB1IN1 INT8	1	I/O Input Input	Port 85: Programmable as input or output (with internal pull-up resistor) 16-bit timer 1 input 1: Count/capture trigger input to 16-bit timer 1 Interrupt request 8: Rising-edge sensitive
P86 TB1OUT0	1	I/O Output	Port 86: Programmable as input or output (with internal pull-up resistor) 16-bit timer 1 output 0: Output from 16-bit timer 1
P87 TB1OUT1	1	I/O Output	Port 87: Programmable as input or output (with internal pull-up resistor) 16-bit timer 1 output 1: Output from 16-bit timer 1
P90 SCK1	1	I/O I/O	Port 90: Programmable as input or output Clock input/output pin when serial bus interface 1 is in SIO mode
P91 SO1 SDA1	1	I/O Output I/O	Port 91: Programmable as input or output (with internal pull-up resistor) Data transmit pin when serial bus interface 1 is in SIO mode Data transmit/receive pin when serial bus interface 1 is in I^2C mode; programmable as an open-drain output
P92 SI1 SCL1	1	I/O Input I/O	Port 92: Programmable as input or output (with internal pull-up resistor) Data receive pin when serial bus interface 1 is in SIO mode Clock input/output pin when serial bus interface 1 is in I^2C mode; programmable as an open-drain output
P93 TXD	1	I/O Output	Port 93: Programmable as input or output Serial transmit data: Programmable as an open-drain output

Table 2.2.4 Pin Names and Functions (4/4)

Pin Name	Number of Pins	I/O	Functions
P94	1	I/O	Port 94: Programmable as input or output
RXD		Input	Serial receive data
P95	1	I/O	Port 95: Programmable as input or output
SCLK		I/O	Serial clock input/output
$\overline{\text{CTS}}$		Input	Serial clear-to-send
P96	1	I/O	Port 96: Programmable as input or output
PA0 to PA3	4	I/O	Ports A0 to A3: Individually programmable as input or output (with internal pull-up resistor)
INT1 to INT4		Input	Interrupt request 1 to 4: Individually programmable to be rising-edge or falling-edge sensitive
PA4 to PA7	4	I/O	Ports A4 to A7: Individually programmable as input or output (with internal pull-up resistor)
$\overline{\text{WAKE}}$	1	Output	STOP mode monitor output This pin drives low when the CPU is operating; the pin is in high-impedance state during reset or in STOP mode.
ALE	1	Output	Address latch enable (This pin can be disabled in order to reduce noise.)
$\overline{\text{NMI}}$	1	Input	Nonmaskable interrupt request: Causes an NMI interrupt on the falling edge; programmable to be rising-edge sensitive
AM0 to AM1	2	Input	Both AM0 and AM1 should be held at logic 1.
EMU0	1	Output	Test pin. This pin should be left open.
EMU1	1	Output	Test pin. This pin should be left open.
$\overline{\text{RESET}}$	1	Input	Reset (with internal pull-up resistor): Initializes the whole TMP91CW28.
VREFH	1	Input	Input pin for high reference voltage for the AD converter
VREFL	1	Input	Input pin for low reference voltage for the AD converter
AVCC	1		Power supply pin for the AD converter
AVSS	1		Ground pin for the AD converter
X1/X2	2	I/O	Connection pins for a crystal oscillator
DVCC	3		Power supply pins. The DVCC pins should be connected to power supply.
DVSS	3		Ground pins. The DVSS pins should be connected to ground.

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

3. Operation

This section describes the functions and basic operation of each block constituting the TMP91CW28.

See also section 7, "Points of Note and Restrictions" for an explanation of precautions and restrictions for individual blocks.

3.1 CPU

The TMP91CW28 contains a high-performance 16-bit CPU called the 900/L1. For a detailed description of the CPU, refer to "TLCS-900/L1 CPU" in the preceding chapter.

Functions unique to the TMP91CW28, which are not covered in "TLCS-900/L1 CPU", are described below.

3.1.1 Reset Operation

When resetting the TMP91CW28 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 (80 μs at 4 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).

The CPU performs the following operations as a result of a reset:

- Set the program counter (PC) according to the reset vectors stored at addresses FFFF00H to FFFF02H
  PC [7:0] ← Value at FFFF00H
  PC [15:8] ← Value at FFFF01H
  PC [23:16] ← Value at FFFF02H
- Set the stack pointer (XSP) to 100H.
- Set the IFF2 to IFF0 bits of the status register (SR) to 111 (Setting the interrupt level mask register to level 7).
- Set the MAX bit of the status register (SR) to 1 (Selecting maximum mode).
- Clear the RFP2 to RFP0 bits of the status register (SR) to 000 (Selecting register bank0).

After a reset, the CPU starts executing instructions according to the set PC. CPU internal registers other than the above are not modified.

The on-chip I/O peripherals, ports and other pins are initialized as follows upon a reset.

- All on-chip I/O peripheral registers are initialized.
- All port pins, including those multiplexed with on-chip peripheral functions, are configured as either general-purpose inputs or general-purpose outputs.
- The ALE pin is placed in high-impedance state.

Note: A reset operation does not affect the contents of the on-chip RAM or the CPU registers other than PC, SR and XSP.

Figure 3.1.1 shows TMP91CW28 reset timings.

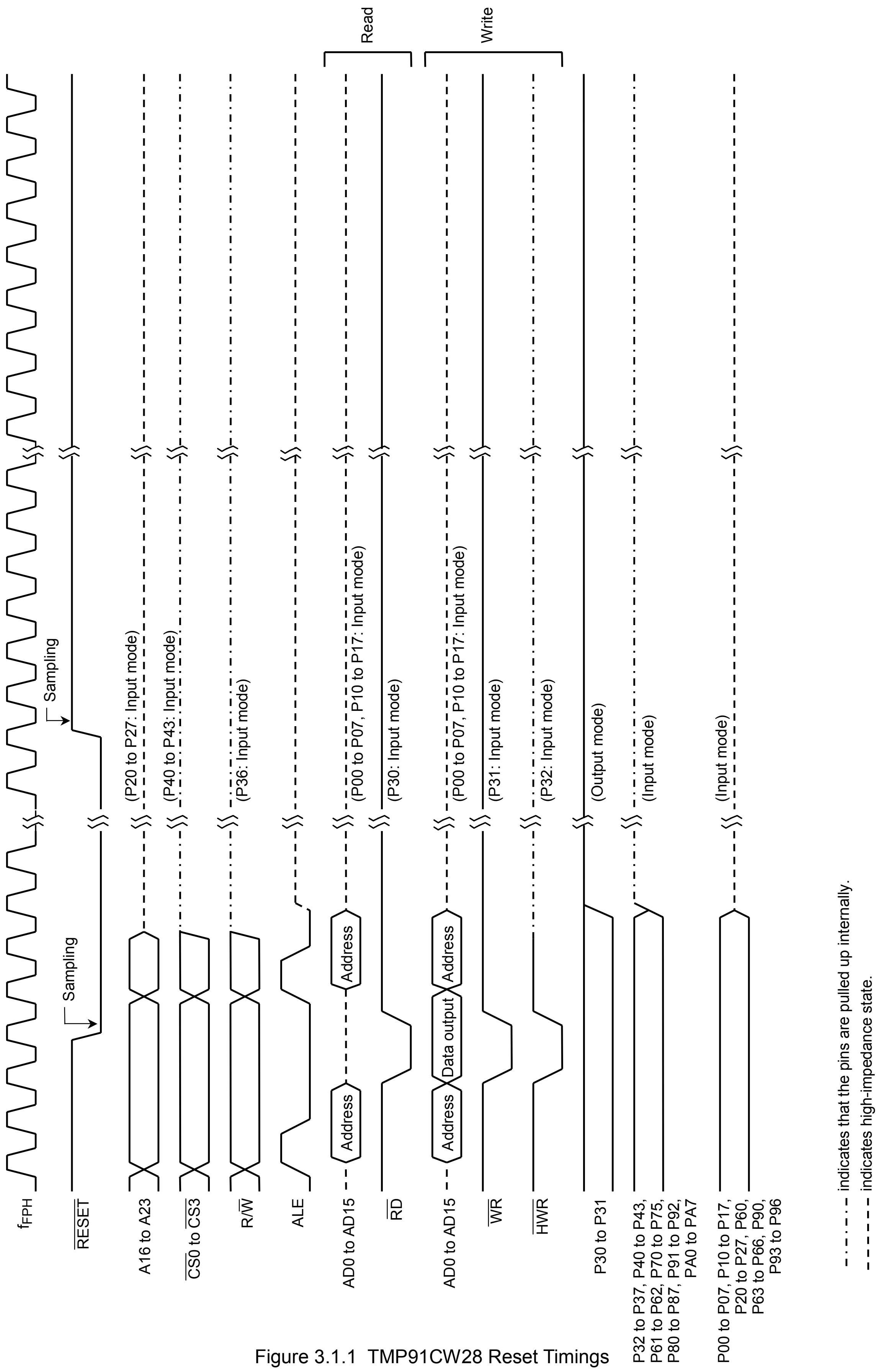

fFPH
RESET
A16 to A23
CS0 to CS3
R/W
ALE
AD0 to AD15
RD
AD0 to AD15
WR
HWR
P30 to P31
P32 to P37, P40 to P43, P61 to P62, P70 to P75, P80 to P87, P91 to P92, PA0 to PA7
P00 to P07, P10 to P17, P20 to P27, P60, P63 to P66, P90, P93 to P96
Read
Write
Sampling
Sampling
(P20 to P27: Input mode)
(P40 to P43: Input mode)
(P36: Input mode)
(P00 to P07, P10 to P17: Input mode)
(P30: Input mode)
(P00 to P07, P10 to P17: Input mode)
(P31: Input mode)
(P32: Input mode)
(Output mode)
(Input mode)
(Input mode)
Address
Address
Address
Data output
Address
indicates that the pins are pulled up internally.
indicates high-impedance state.

Figure 3.1.1 TMP91CW28 Reset Timings

3.2 Memory Map

Figure 3.2.1 shows memory assignment for the TMP91CW28.

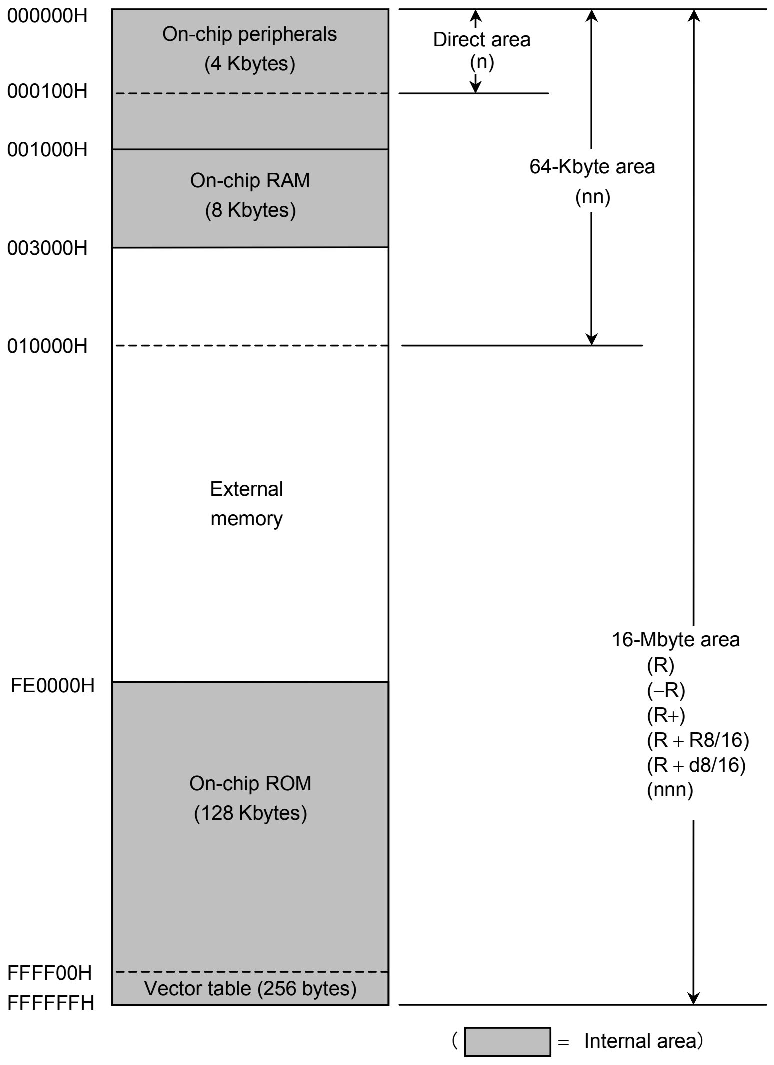


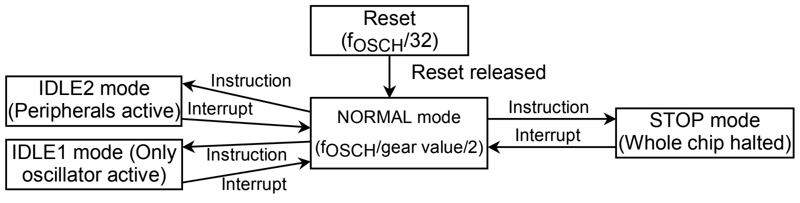
Figure 3.2.1 Memory Map

3.3 Standby Control and Noise Reduction

The TMP91CW28 incorporates clock gear, standby control and noise reduction circuits to minimize power consumption as well as noise.

The TMP91CW28 only supports single-clock mode, in which it operates off of the clock supplied from the X1 and X2 pins.

Figure 3.3.1 shows state transitions in single-clock mode.


Reset
(f_{OSCH}/32)
Reset released
IDLE2 mode (Peripherals active)
Instruction
Interrupt
NORMAL mode (f_{OSCH}/gear value/2)
Instruction
Interrupt
STOP mode (Whole chip halted)
IDLE1 mode (Only oscillator active)
Instruction
Interrupt

State transitions in single-clock mode

Figure 3.3.1 State Transitions in Single-clock Mode

f_{OSCH}:	Clock frequency supplied via the X1 and X2 pins
f_{FPH}:	Clock frequency selected by the GEAR[2:0] bit in the SYSCR1
f_{SYS}:	System clock frequency, created by dividing f_{FPH} by two
1 state:	One period of f_{SYS}

3.3.1 Clock Source Block Diagram

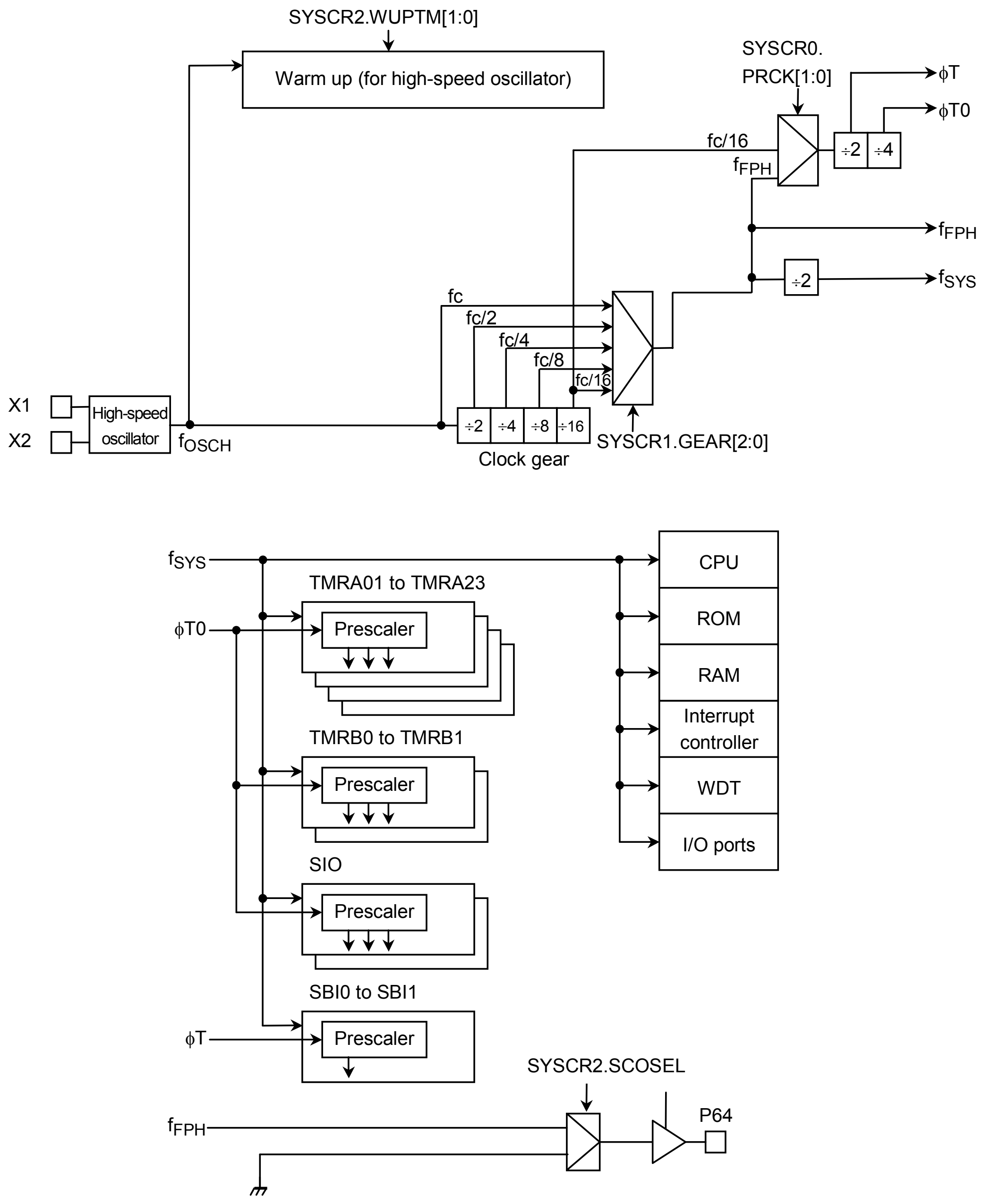


Figure 3.3.2 Clock and Standby Block Diagram

3.3.2 SFR Descriptions

		7	6	5	4	3	2	1	0
SYSCR0 (00E0H)	Bit symbol	–	–	–	–	–	–	PRCK1	PRCK0
	Read/Write	W						R/W	
	Reset value	1	0	1	0	0	0	0	0
	Function	Must be written as “1”.	Must be written as “0”.	Must be written as “1”.	Must be written as “0”.	Must be written as “0”.	Must be written as “0”.	Prescaler clock select 00: f_{FPH} 01: Reserved 10: fc/16 11: Reserved	
SYSCR1 (00E1H)	Bit symbol					–	GEAR2	GEAR1	GEAR0
	Read/Write					W	R/W		
	Reset value					0	1	0	0
	Function					Must be written as “0”.	High-speed clock gear select 000: High-speed clock 001: High-speed clock /2 010: High-speed clock /4 011: High-speed clock /8 100: High-speed clock /16 101: Reserved 110: Reserved 111: Reserved		
SYSCR2 (00E2H)	Bit symbol		SCOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
	Read/Write		R/W	R/W	R/W	R/W	R/W		R/W
	Reset value		0	1	0	1	1		0
	Function		0: Low level 1: f_{FPH}	Oscillator warm-up time 00: Reserved 01: 2^8/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		HALT mode select 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			1: Pins are driven in STOP mode.

Note 1: Bits7 to 2 of the SYSCR0, bits7 to 4 of the SYSCR1 and bits7 and 1 of the SYSCR2 are read as undefined.

Note 2: When the on-chip SBI is used, the prescaler select register, SYSCR0.PRCK[1:0], must be set to 00 (f_{FPH}).

Figure 3.3.3 Clock-related SFRs

		7	6	5	4	3	2	1	0
EMCCR0 (00E3H)	Bit symbol	PROTECT	–	–	–	ALEEN	EXTIN	–	–
	Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Reset value	0	0	1	0	0	0	1	1
	Function	Protection flag 0: Disabled 1: Enabled	Must be set to “0”.	Must be set to “1”.	Must be set to “0”.	1: ALE output enabled	1: External clock used as fc	Must be set to “1”.	Must be set to “1”.
EMCCR1 (00E4H)	Bit symbol	–							
	Read/Write	W							
	Reset value	–							
	Function	On writes: 1FH: Protection disabled Other than 1FH: Protection enabled							

Figure 3.3.4 Noise-related SFRs

3.3.3 System Clock Control Section

The system clock control section generates system clock pulses (f_{SYS}) that are supplied to the CPU core and on-chip peripherals. It accepts the fc clock pulses, output from the high-speed oscillator, and uses the SYSCR1.GEAR[2:0] bits to gear down the high-speed clock frequency to fc, fc/2, fc/4, fc/8, or fc/16, thus enabling reduction in power consumption.

A system reset initializes the SYSCR1.GEAR[2:0] bits to 100, putting the TMP91CW28 in single-clock mode. The system clock frequency (f_{SYS}) is geared down to fc/32 (= fc/16 × 1/2). For example, if a 10 MHz crystal is connected between the X1 and X2 pins, the f_{SYS} clock operates at 0.3125 MHz.

(1) Changing the clock gear

The clock gear select register SYSCR1.GEAR[2:0] can be used to set f_{FPH} to fc, fc/2, fc/4, fc/8 or fc/16. Gearing down f_{FPH} results in smaller power consumption.

The following shows an example of changing the clock gear:

Example:

Gearing down the high-speed clock frequency

```
SYSCR1   EQU   00E1H

         LD    (SYSCR1), XXXX0000B   ;   Changes system clock fSYS to fc/2.
```

X: Don’t care

There is one thing to remember when changing the clock gear value.

The clock gear can be changed by the programming of the GEAR[2:0] bits of the SYSCR1, as shown in the above example. It takes a few clock cycles for a gear change to take effect. Therefore, one or more instructions following the instruction that changed the clock gear value may be executed using the old clock gear value. If subsequent instructions need be executed with a new clock gear value, a dummy instruction (one that executes a write cycle, as shown below) should be inserted after the instruction that modifies the clock gear value.

Example:

```
SYSCR1   EQU   00E1H
         LD    (SYSCR1), XXXX0001B   ;   Changes fSYS to fc/4.
         LD    (DUMMY), 00H          ;   Dummy instruction.
               Instructions that need be
               executed with a new clock
               gear value
```

(2) Internal clock output

The f_{FPH} internal clock can be driven out from the P64/SCOUT pin.

The P64/SCOUT pin is configured as SCOUT (System clock output) by programming the port 6 registers as follows: P6CR.P64C = 1 and P6FC.P64F = 1. The output clock is selected through the SYSCR2.SCOSEL bit.

Table 3.3.1 shows the pin states in each clocking mode when the P64/SCOUT pin is configured as SCOUT.

Table 3.3.1 SCOUT Output States

Mode / SCOUT Select	NORMAL	HALT Modes		
		IDLE2	IDLE1	STOP
SCOSEL = 0	A low level is driven out.			
SCOSEL = 1	The f_{FPH} clock is driven out.		Held at either 1 or 0.	

3.3.4 Prescaler Clock Control Section

The on-chip peripherals (TMRA01 to TMRA23, TMRB0, TMRB1, SIO, SBI0 and SBI1) have a clock prescaler.

The prescaler clock source (ϕT, ϕT0) can be selected from either f_{FPH} or fc/16 through the PRCK[1:0] bits of the SYSCR0. The selected clock frequency (f_{FPH} or fc/16) is divided by two or four before being supplied to the prescaler.

When the on-chip SBI is used, PRCK[1:0] must be cleared to 00.

3.3.5 Noise Cancellers

The TMP91CW28 incorporates circuits providing the following features in order to reduce electromagnetic interference (EMI) and improve electromagnetic susceptibility (EMS):

(1) Canceling double-drive operation of the high-speed oscillator

(2) Disabling output from the ALE pin

(3) Preventing software or system lockups

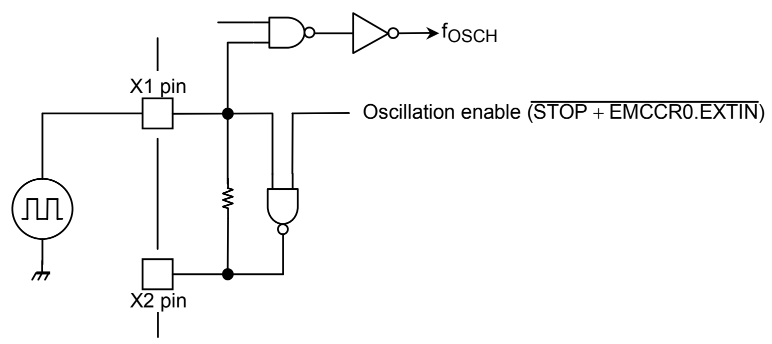
These features can be selected using the EMCCR0 and EMCCR1 registers.

(1) Canceling double-drive operation of the high-speed oscillator

Purpose:

To prevent malfunction due to noise coming through the X2 pin that is open when an external oscillator is used, with double-drive operation not required.

Block diagram:



f_{OSCH}
X1 pin
Oscillation enable ($\overline{\text{STOP + EMCCR0.EXTIN}}$)
X2 pin

Description:

Setting the EXTIN bit of the EMCCR0 to 1 causes the high-speed oscillator to stop oscillation and operate as a buffer, with the X2 pin driven high.

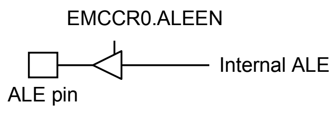
A system reset initializes the EXTIN bit to 0.

(2) Disabling output from the ALE pin

Purpose:

To prevent unwanted clock noise from being driven out when no external area is accessed.

Block diagram:



Description:

Clearing the ALEEN bit of the EMCCR0 to 0 disables the output buffer of the ALE pin, placing the pin into high-impedance state.

A system reset initializes the ALEEN bit to 0.

When accessing an external area, set ALEEN to 1 before attempting to access the area.

(3) Preventing software or system lockups using a protection register

Purpose:

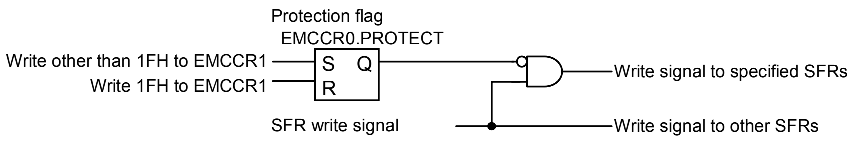
To prevent software or system lockups that may occur due to incoming noise.

Applying protection causes specified SFRs to be write-protected, thus preventing the system recovery routine from becoming unfetchable, for example, if the system clock stops or a memory control register (CS/WAIT controller) is modified.

Applicable SFRs

1. CS/WAIT controller
   B0CS, B1CS, B2CS, B3CS, BEXCS,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3
2. Clock gear (Only EMCCR1 can be written.)
   SYSCR0, SYSCR1, SYSCR2, EMCCR0

Block diagram:


Protection flag
EMCCR0.PROTECT
Write other than 1FH to EMCCR1
Write 1FH to EMCCR1
S
Q
R
SFR write signal
Write signal to specified SFRs
Write signal to other SFRs

Description:

Writing any code other than 1FH to the EMCCR1 register enables protection, thus preventing specified SFRs from being written.

Writing 1FH to the EMCCR1 register cancels protection. The state of protection can be determined by reading the PROTECT bit of the EMCCR0.

A system reset cancels protection.

3.3.6 Standby Control Section

(1) HALT mode

Executing the HALT instruction causes the TMP91CW28 to enter one of the HALT modes – IDLE2, IDLE1 or STOP – as specified by the SYSCR2.HALTM[1:0] bits.

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

a. IDLE2: The CPU stops.
On-chip peripherals can be selectively enabled and disabled through use of a register bit in an SFR, as shown in Table 3.3.2.

Table 3.3.2 IDLE2 Mode Register Settings

Peripheral	SFR
TMRA01	TA01RUN.I2TA01
TMRA23	TA23RUN.I2TA23
TMRB0	TB0RUN.I2TB0
TMRB1	TB1RUN.I2TB1
SIO	SC0MOD1.I2S0
SBI0	SBI0BR0.I2SBI0
SBI1	SBI1BR0.I2SBI1
ADC	ADMOD1.I2AD
WDT	WDMOD.I2WDT

b. IDLE1: Only the on-chip oscillator is operational.

c. STOP: The whole TMP91CW28 stops.

Table 3.3.3 shows the operation of each circuit block in HALT modes.

Table 3.3.3 TMP91CW28 Circuit Blocks in HALT Modes

HALT Mode		IDLE2	IDLE1	STOP
SYSCR2.HALTM[1:0]		11	10	01
Circuit block	CPU	OFF		
	I/O ports	Holding the states when the HALT instruction was executed		See Table 3.3.6 to Table 3.3.9
	TMRA, TMRB	Selectable programmatically on a block-by-block basis	OFF	
	SIO, SBI			
	ADC			
	WDT			
	Interrupt controller	ON		

(2) Wakeup signaling

There are two ways to exit a HALT mode: An interrupt request or reset signal. Availability of wakeup signaling depends on the settings of the interrupt mask level bits, IFF[2:0], of the CPU status register (SR) and the current HALT mode (See Table 3.3.4).

- Wakeup via interrupt signaling

  The operation upon return from a HALT mode varies, depending on the interrupt priority level programmed before executing the HALT instruction. If the interrupt priority level is greater than or equal to the processor's interrupt mask level, execution resumes with the interrupt service routine. Upon completion of the interrupt service routine, program execution resumes with the instruction immediately following the HALT instruction. If the interrupt priority level is less than the processor's interrupt mask level, the HALT mode is not terminated. (Nonmaskable interrupts are always serviced upon return from a HALT mode, regardless of the current interrupt mask level.)

  Only interrupts INT0 to INT4 can, however, terminate a HALT mode even if the interrupt priority level is less than the processor's interrupt mask level. In that case, program execution resumes with the instruction immediately following the HALT instruction, without executing the interrupt service routine. The interrupt request flag remains set.

- Wakeup via reset signaling

  Reset signaling always brings the TMP91CW28 out of any HALT mode. A wakeup from STOP mode must allow sufficient time for the oscillator to restart and stabilize (See Table 3.3.5).

  A reset does not affect the contents of the on-chip RAM, but initializes everything else, whereas an interrupt preserves all internal states that were in effect before the HALT mode was entered.

Table 3.3.4 Wakeup Signaling Sources and Wakeup Operations

Interrupt Masking			Unmasked Interrupt (Request_level ≥ mask_level)			Masked Interrupt (Request_level < mask_level)		
HALT Mode			Programmable IDLE2	IDLE1	STOP	Programmable IDLE2	IDLE1	STOP
Wakeup signaling sources	Interrupts	NMI	♦	♦	♦*1	−	−	−
		INTWDT	♦	×	×	−	−	−
		INT0 to 4 (Note 1)	♦	♦	♦*1	○	○	○*1
		INT5 to 8	♦ (Note 2)	×	×	×	×	×
		INTTA0 to 3	♦	×	×	×	×	×
		INTTB to 00, 01, 10, 11, OF0, OF1	♦	×	×	×	×	×
		INTRX, TX	♦	×	×	×	×	×
		INTSBI0 to 1	♦	×	×	×	×	×
		INTAD	♦	×	×	×	×	×
		INTBCD	♦	×	×	×	×	×
	RESET		Initializes the whole TMP91CW28					

♦: Execution resumes with the interrupt service routine. (RESET initializes the whole TMP91CW28.)

○: Execution resumes with the instruction immediately following the HALT instruction. The interrupt is left pending.

×: Cannot be used to exit a HALT mode.

−: These combinations are not possible because nonmaskable interrupts are assigned a highest priority level (7).

*1: The TMP91CW28 exits the HALT mode after the warm-up period timer expires.

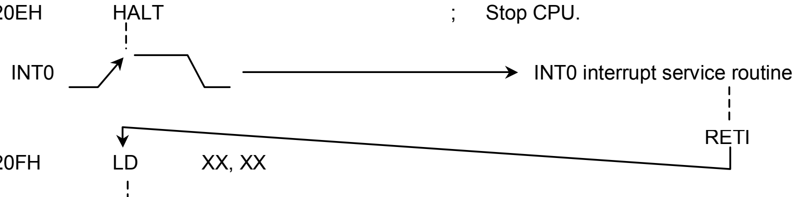
Note 1: If the interrupt request level is greater than the mask level, an INT0 interrupt signal which is programmed as level-sensitive must be held high until interrupt processing begins. Otherwise, the interrupt will not be serviced successfully.

Note 2: When external INT5 to INT8 interrupts are used in programmable IDLE2 mode, 16-bit timer run register bits TB0RUN.I2TB0 and TB1RUN.I2TB1 must be set to 1.

Example of exiting a HALT mode:

When using an edge-sensitive INT0 interrupt to exit IDLE1 mode

```
Address
8203H     LD      (IIMC), 00H          ;   Set INT0 interrupt to rising-edge sensitive.
8206H     LD      (INTE0AD), 06H       ;   Set INT0 interrupt priority level to 6.
8209H     EI      5                    ;   Set CPU interrupt priority level to 5.
820BH     LD      (SYSCR2), 28H        ;   Select IDLE1 mode.
820EH     HALT                         ;   Stop CPU.
```



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

(3) Operation in HALT modes

a. IDLE2 mode

In IDLE2 mode, the CPU stops executing instructions and only the on-chip peripherals enabled with the IDLE2 setting bits in respective SFRs are operational.

Figure 3.3.5 shows example timings for exiting IDLE2 mode with an interrupt.

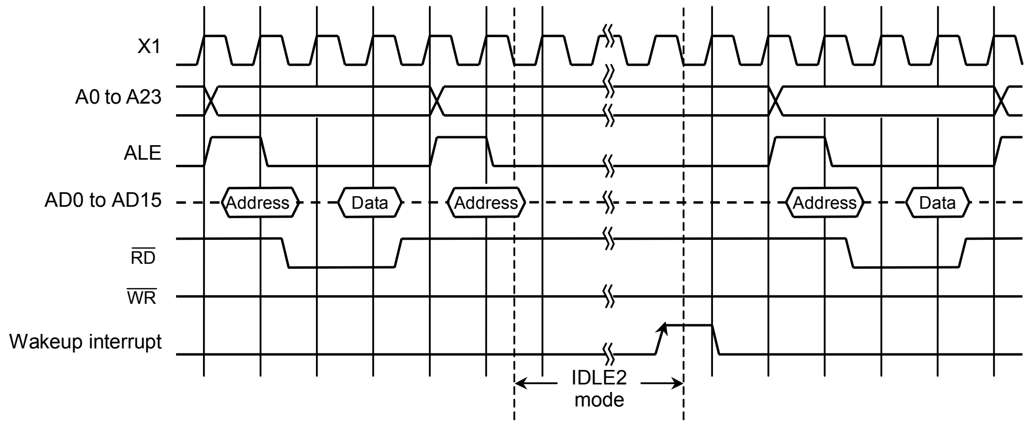


Figure 3.3.5 Example Timings for Exiting a HALT Mode with an Interrupt (in IDLE2 mode)

b. IDLE1 mode

In IDLE1 mode, the system clock stops while only the on-chip oscillator is active. Interrupt requests are sampled asynchronously with the system clock in a halt state but the HALT mode is exited in synchronization with the system clock.

Figure 3.3.6 shows example timings for exiting IDLE1 mode with an interrupt.

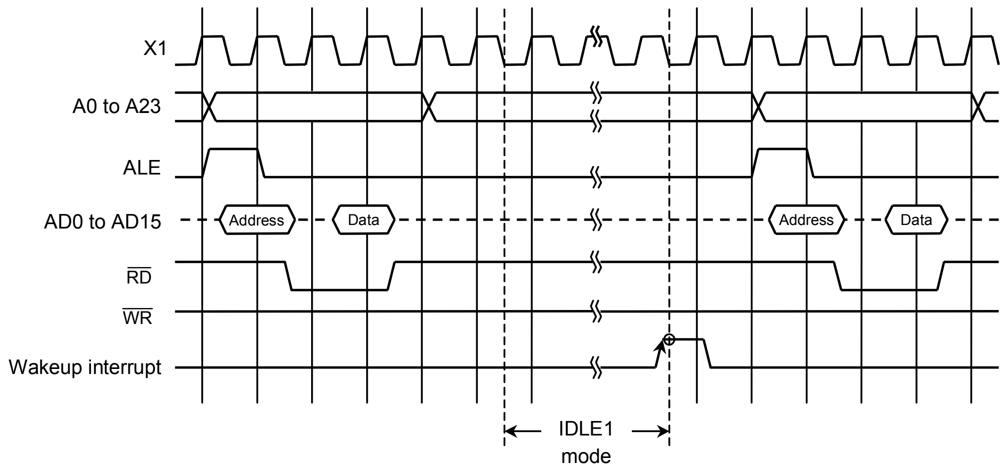


Figure 3.3.6 Example Timings for Exiting a HALT Mode with an Interrupt (in IDLE1 mode)

c. STOP mode

In STOP mode, the whole TMP91CW28 stops, including the on-chip oscillator. Pin states in STOP mode depend on the setting of the SYSCR2.DRVE bit, as shown in Table 3.3.6 to Table 3.3.9.

Upon detection of wakeup signaling, the warm-up period timer should be activated to allow sufficient time for the oscillator to restart and stabilize before exiting STOP mode. After that, the system clock output can restart. The warm-up period is chosen through the SYSCR2.WUPTM[1:0] bits, as shown in Table 3.3.5.

Figure 3.3.7 shows example timings for exiting STOP mode with an interrupt.

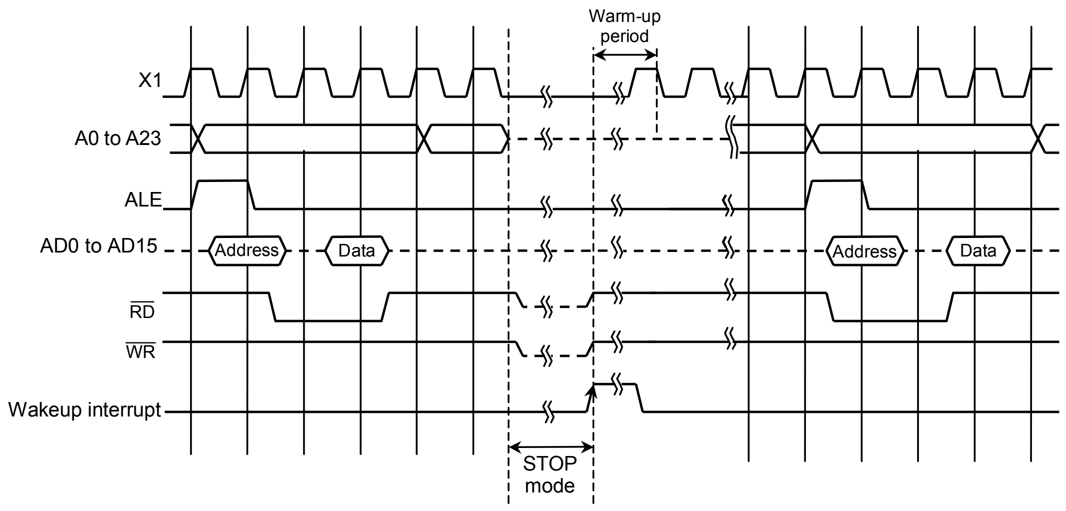


Figure 3.3.7 Example Timings for Exiting a HALT Mode with an Interrupt (in STOP mode)

Table 3.3.5 Example Warm-up Period Settings (when exiting STOP mode)

at f_{OSCH} = 10 MHz

SYSCR2.WUPTM[1:0]		
01 (2^8)	10 (2^{14})	11 (2^{16})
25.6 μs	1.6384 ms	6.5536 ms

Table 3.3.6 Input Buffer State Table (1/2)

Port Name	Input Function Name	Input Buffer State								
		During Reset	When the CPU is Operating		In HALT Mode (IDLE2/IDLE1)		In HALT Mode (STOP)			
							<DRVE> = 1		<DRVE> = 0	
			When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port
P00 to P07	AD0 to AD7	OFF	ON	ON (*3)	OFF	OFF	OFF	OFF	OFF	OFF
P10 to P17	AD8 to AD15	OFF	ON	ON (*3)	OFF	OFF	OFF	OFF	OFF	OFF
	A8 to A15	OFF	ON	ON (*3)	OFF	OFF	OFF	OFF	OFF	OFF
P20 to P27	A0 to A7	OFF	ON	ON (*3)	OFF	OFF	OFF	OFF	OFF	OFF
	A16 to A23	OFF	ON	ON (*3)	OFF	OFF	OFF	OFF	OFF	OFF
P32 (*1)	–	OFF	–	ON (*3)	–	OFF	–	OFF	–	OFF
P33 (*1)	$\overline{\text{WAIT}}$	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
P34 (*1)	$\overline{\text{BUSRQ}}$	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P35 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P36 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P37 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P40 to P43 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P50 (*2)	AN0	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI0	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P51 (*2)	AN1	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI1	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P52 (*2)	AN2	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI2	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P53 (*2)	AN3	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	$\overline{\text{ADTRG}}$	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI3	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P54 (*2)	AN4	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI4	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P55 (*2)	AN5	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI5	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P56 (*2)	AN6	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI6	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P57 (*2)	AN7	ON	ON	ON (*3)	ON	ON	ON	ON	ON	ON
	KWI7	OFF	ON	ON (*3)	ON	ON	ON	ON	ON	ON
P60	SCK0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P61 (*1)	SDA0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P62 (*1)	SI0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	SCL0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P63	INT0	ON	ON	ON	ON	ON	ON	ON	ON	ON
P64	–	ON	–	ON	–	ON	–	ON	–	OFF
P65	–	ON	–	ON	–	ON	–	ON	–	OFF
P66	–	ON	–	ON	–	ON	–	ON	–	OFF
P70 (*1)	TA0IN	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P71 to P75 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF

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

OFF: The buffer is always turned off.

–: Not applicable.

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

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

*3: The buffer is turned on if read port.

Table 3.3.7 Input Buffer State Table (2/2)

Port Name	Input Function Name	Input Buffer State								
		During Reset	When the CPU is Operating		In HALT Mode (IDLE2/IDLE1)		In HALT Mode (STOP)			
							<DRVE> = 1		<DRVE> = 0	
			When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port	When Used as Function Pin	When Used as Input Port
P80 (*1)	TB0IN0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	INT5	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P81 (*1)	TB0IN1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	INT6	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P82 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P83 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P84 (*1)	TB1IN0	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	INT7	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P85 (*1)	TB1IN1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	INT8	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P86 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P87 (*1)	–	ON	–	ON	–	ON	–	ON	–	OFF
P90	SCK1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P91 (*1)	SDA1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P92 (*1)	SI1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	SCL1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P93	–	ON	–	ON	–	ON	–	ON	–	OFF
P94	RXD1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P95	SCLK1	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
	$\overline{\text{CTS1}}$	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
P96	–	ON	–	ON	–	ON	–	ON	–	OFF
PA0 (*1)	INT1	OFF	ON	ON (*3)	ON	OFF	ON	OFF	ON	OFF
PA1 (*1)	INT2	OFF	ON	ON (*3)	ON	OFF	ON	OFF	ON	OFF
PA2 (*1)	INT3	OFF	ON	ON (*3)	ON	OFF	ON	OFF	ON	OFF
PA3 (*1)	INT4	OFF	ON	ON (*3)	ON	OFF	ON	OFF	ON	OFF
PA4 (*1)	–	OFF	–	ON (*3)	–	OFF	–	OFF	–	OFF
PA5 (*1)	–	OFF	–	ON (*3)	–	OFF	–	OFF	–	OFF
PA6 (*1)	–	OFF	–	ON (*3)	–	OFF	–	OFF	–	OFF
PA7 (*1)	–	OFF	–	ON (*3)	–	OFF	–	OFF	–	OFF
$\overline{\text{NMI}}$ (*1)	–	ON	ON	ON	ON	ON	ON	ON	ON	ON
$\overline{\text{RESET}}$ (*1)	–	ON	ON	ON	ON	ON	ON	ON	ON	ON
AM0, AM1	–	ON	ON	ON	ON	ON	ON	ON	ON	ON
X1	–	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF

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

OFF: The buffer is always turned off.

–: Not applicable.

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

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

*3: The buffer is turned on if read port.

Table 3.3.8 Output Buffer State Table (1/2)

Port Name	Output Function Name	Output Buffer State								
		During Reset	When the CPU is Operating		In HALT Mode (IDLE2/IDLE1)		In HALT Mode (STOP)			
							<DRVE> = 1		<DRVE> = 0	
			When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port
P00 to P07	AD0 to AD7	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P10 to P17	AD8 to AD15	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	A8 to A15	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P20 to P27	A0 to A7	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	A16 to A21	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P30	$\overline{RD}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P31	$\overline{WR}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P32 (*1)	$\overline{HWR}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P33 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P34 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P35 (*1)	$\overline{BUSAK}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P36 (*1)	R/$\overline{W}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P37 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P40 to P43 (*1)	$\overline{CS0}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	$\overline{CS1}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	$\overline{CS2}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	$\overline{CS3}$	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P60	SCK0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P61	SDA0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	SO0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P62	SCL0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P63	–	OFF	–	ON	–	ON	–	ON	–	OFF
P64	SCOUT	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P65	–	OFF	–	ON	–	ON	–	ON	–	OFF
P66	–	OFF	–	ON	–	ON	–	ON	–	OFF
P70 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P71(*1)	TA1OUT	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P72 (*1)	TA3OUT	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P73 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P74(*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P75(*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P80 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P81 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P82 (*1)	TB0OUT0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P83 (*1)	TB0OUT1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P84 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P85 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
P86 (*1)	TB1OUT0	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P87 (*1)	TB1OUT1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF

ON:	The buffer is always turned on.
OFF:	The buffer is always turned off.
–:	Not applicable.
*1:	Port having a pull-up/pull-down resistor.
*2:	AIN input does not cause a current to flow through the buffer.

Table 3.3.9 Output Buffer State Table (2/2)

Port Name	Output Function Name	Output Buffer State								
		During Reset	When the CPU is Operating		In HALT Mode (IDLE2/IDLE1)		In HALT Mode (STOP)			
							<DRVE> = 1		<DRVE> = 0	
			When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port	When Used as Function Pin	When Used as Output Port
P90	SCK1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P91 (*1)	SDA1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
	SO1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P92 (*1)	SCLK1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P93	TXD1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P94	–	OFF	–	ON	–	ON	–	ON	–	OFF
P95	SCLK1	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF
P96	–	OFF	–	ON	–	ON	–	ON	–	OFF
PA0 to PA7 (*1)	–	OFF	–	ON	–	ON	–	ON	–	OFF
WAKE	–	OFF	ON	ON	ON	ON	ON	ON	OFF	ON
ALE	–	OFF	ON	ON	ON	ON	ON	ON	OFF	ON
X2	–	OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF

ON:	The buffer is always turned on.
OFF:	The buffer is always turned off.
–:	Not applicable.
*1:	Port having a pull-up/pull-down resistor.
*2:	AIN input does not cause a current to flow through the buffer.

3.4 Interrupts

Interrupt processing is coordinated between the CPU interrupt mask register SR.IFF[2:0] and the on-chip interrupt controller.

The TMP91CW28 supports the following 48 interrupt sources:

- 9 CPU internal interrupts
  (Software interrupts and interrupts triggered when an undefined instruction is executed.)
- 18 external interrupt pins ($\overline{\text{NMI}}$, INT0 to INT8, KWI0 to KWI7)
- 21 on-chip peripheral interrupts

Each interrupt source has a unique interrupt vector number (Fixed). Each maskable interrupt is assigned one of six priority levels (Variable) while nonmaskable interrupts have the highest priority level of 7 (Fixed).

When an interrupt occurs, the interrupt controller sends the priority level of that interrupt source to the CPU. If two or more interrupts occur simultaneously, it sends the highest of their priority levels (7 if a nonmaskable interrupt occurs) to the CPU.

The CPU compares the sent priority level with the contents of the CPU interrupt mask register IFF[2:0]. If the sent priority level is greater than or equal to the interrupt mask level, the CPU accepts the interrupt. The contents of the IFF[2:0] bits can be modified using the EI instruction in the format of EI num, where num is the value to be set in IFF[2:0]. For example, EI 3 causes the CPU to accept maskable interrupts having a priority level of 3 or greater, as specified with the interrupt controller, as well as all nonmaskable interrupts. The DI instruction, which sets IFF[2:0] to 7, has the same effect as EI 7. It is used to prevent the CPU from accepting maskable interrupts because maskable interrupts can have priority levels of only up to 6. The EI instruction takes effect immediately after it is executed.

In addition to general interrupt servicing, as described above, the TMP91CW28 supports micro DMA mode, where the CPU automatically transfers data (1 byte, 2 bytes or 4 bytes). This mode enables faster data transfer to on-chip/external memory and on-chip peripherals.

A micro DMA request can be issued either using an interrupt source or programmatically with the soft start feature.

Figure 3.4.1 shows the overall flow of interrupt servicing.

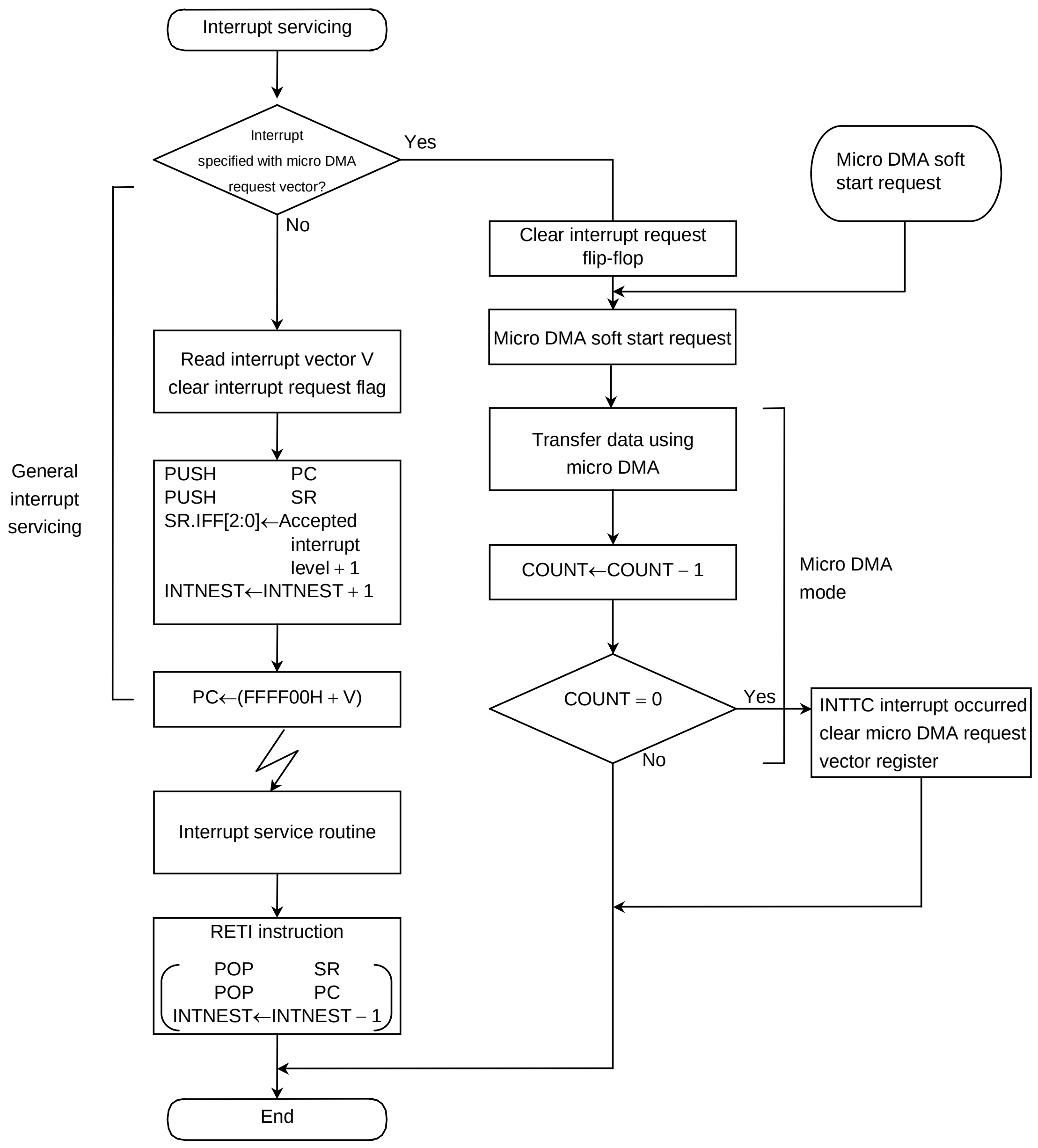

Interrupt servicing
Interrupt specified with micro DMA request vector?
Yes
No
Micro DMA soft start request
Clear interrupt request flip-flop
Micro DMA soft start request
Read interrupt vector V clear interrupt request flag
General interrupt servicing
PUSH PC
PUSH SR
SR.IFF[2:0]←Accepted interrupt level + 1
INTNEST←INTNEST + 1
Transfer data using micro DMA
COUNT←COUNT − 1
Micro DMA mode
PC←(FFFF00H + V)
COUNT = 0
Yes
No
INTTC interrupt occurred clear micro DMA request vector register
Interrupt service routine
RETI instruction
POP SR
POP PC
INTNEST←INTNEST − 1
End

Figure 3.4.1 Overall Interrupt Servicing Flow

3.4.1 General Interrupt Servicing

The CPU performs the following operations once it accepts an interrupt. However, when the CPU itself generates an interrupt, as triggered by a software interrupt instruction or upon the execution of an undefined instruction, it only performs steps 2, 4 and 5. The operations are the same as those performed by the TLCS-900/L and TLCS-900/H.

(1) Reads an interrupt vector from the interrupt controller.
If two or more interrupts having the same priority level occur simultaneously, the interrupt controller generates an interrupt vector according to default priorities (Fixed; higher priorities assigned to smaller vector values) and clears the interrupt request.

(2) Pushes the contents of the program counter (PC) and status register (SR) to the stack area, indicated by the XSP.

(3) Sets the interrupt mask register bits IFF[2:0] to one higher than the accepted interrupt level. If the level is 7, however, it sets the bits to 7 without incrementing the value.

(4) Increments the interrupt nesting counter INTNEST by one.

(5) Makes a branch to the address specified with the data stored at address (FFFF00H + interrupt vector) and then starts the interrupt service routine.

The above procedure requires 18 states (3.6 μs at 10 MHz) in the best case (with 16-bit data bus and 0-wait cycles).

Upon the completion of interrupt servicing, the RETI instruction is usually used to return to the main routine. The RETI instruction restores the contents of the PC and SR from the stack and decrements the INTNEST by one.

Nonmaskable interrupts cannot be disabled programmatically. Maskable interrupts can be disabled or enabled programmatically and a priority level can be specified for each interrupt source. The CPU accepts an interrupt if its priority level is greater than or equal to the value stored in the CPU's IFF[2:0] bits. The CPU then sets the IFF[2:0] bits to the accepted priority level plus one. This enables the CPU to accept any higher-priority interrupt that occurs while servicing the current interrupt, so that interrupts are nested.

If another interrupt request is issued while the CPU is performing the above steps, the request is sampled immediately after the first instruction of the current interrupt service routine is executed. The DI instruction can be used as the first instruction to prohibit nesting of maskable interrupts.

Upon a system reset, the IFF[2:0] bits are initialized to 7 so that maskable interrupts are disabled.

Addresses FFFF00H through FFFFFFH (256 bytes) are assigned to the interrupt vector area. Table 3.4.1 shows the interrupt vector table.

Table 3.4.1 TMP91CW28 Interrupt Vector Table

Default Priority	Type	Interrupt Source	Vector Value	Vector Reference Address	Micro DMA Request Vector
1	Nonmaskable	Reset or SWI 0 instruction	0000H	FFFF00H	–
2		SWI1 instruction	0004H	FFFF04H	–
3		INTUNDEF: Execution of an undefined instruction; or SWI2 instruction	0008H	FFFF08H	–
4		SWI3 instruction	000CH	FFFF0CH	–
5		SWI4 instruction	0010H	FFFF10H	–
6		SWI5 instruction	0014H	FFFF14H	–
7		SWI6 instruction	0018H	FFFF18H	–
8		SWI7 instruction	001CH	FFFF1CH	–
9		$\overline{\text{NMI}}$ pin	0020H	FFFF20H	–
10		INTWD: Watchdog timer	0024H	FFFF24H	–
–	Maskable	(Micro DMA)	–	–	–
11		INT0 pin	0028H	FFFF28H	0AH
12		INT1 pin	002CH	FFFF2CH	0BH
13		INT2 pin	0030H	FFFF30H	0CH
14		INT3 pin	0034H	FFFF34H	0DH
15		INT4 pin, KWI0 to KWI7 pins	0038H	FFFF38H	0EH
16		INT5 pin	003CH	FFFF3CH	0FH
17		INT6 pin	0040H	FFFF40H	10H
18		INT7 pin	0044H	FFFF44H	11H
19		INT8 pin	0048H	FFFF48H	12H
20		INTTA0: 8-bit timer 0	004CH	FFFF4CH	13H
21		INTTA1: 8-bit timer 1	0050H	FFFF50H	14H
22		INTTA2: 8-bit timer 2	0054H	FFFF54H	15H
23		INTTA3: 8-bit timer 3	0058H	FFFF58H	16H
–		–	–	–	–
–		–	–	–	–
–		–	–	–	–
–		–	–	–	–
24		INTTB00: 16-bit timer 0 (TB0RG0)	006CH	FFFF6CH	1BH
25		INTTB01: 16-bit timer 0 (TB0RG1)	0070H	FFFF70H	1CH
26		INTTB10: 16-bit timer 1 (TB1RG0)	0074H	FFFF74H	1DH
27		INTTB11: 16-bit timer 1 (TB1RG1)	0078H	FFFF78H	1EH
28		INTTBOF0: 16-bit timer 0 (Overflow)	007CH	FFFF7CH	1FH
29		INTTBOF1: 16-bit timer 1 (Overflow)	0080H	FFFF80H	20H
–		–	–	–	–
–		–	–	–	–
30		INTRX: UART receive	008CH	FFFF8CH	23H
31		INTTX: UART transmit	0090H	FFFF90H	24H
32		INTSBI0: Serial bus interface interrupt	0094H	FFFF94H	25H
33		INTSBI1: Serial bus interface interrupt	0098H	FFFF98H	26H
34		INTAD: AD conversion complete	009CH	FFFF9CH	27H
35		INTTC0: Micro DMA complete (Channel 0)	00A0H	FFFFA0H	–
36		INTTC1: Micro DMA complete (Channel 1)	00A4H	FFFFA4H	–
37		INTTC2: Micro DMA complete (Channel 2)	00A8H	FFFFA8H	–
38		INTTC3: Micro DMA complete (Channel 3)	00ACH	FFFFACH	–
39		INTBCD: BCD computation complete	00B0H	FFFFB0H	2CH
		(Reserved) : (Reserved)	00B4H : 00FCH	FFFFB4H : FFFFFCH	– : –

Note: Micro DMA default priority.
If an interrupt request is generated by a source specified by micro DMA, the interrupt has the highest priority of the maskable interrupts (Irrespective of the default priority allocated to all channels).

3.4.2 Micro DMA

In addition to general interrupt servicing, the TMP91CW28 supports a micro DMA feature. Interrupt requests specified with the micro DMA are assigned highest priority levels among maskable interrupts regardless of the priority levels actually set.

The micro DMA consists of four channels so that continuous transfer can be performed using burst specification, described later.

Because the micro DMA feature is provided in combination with the CPU, micro DMA requests are ignored and remain pending if the CPU executes the HALT instruction and enters a standby state (STOP, IDLE1 or IDLE2). A DMA transfer is started upon the release from the standby state.

(1) Micro DMA operation

If an interrupt specified with the micro DMA request vector register is requested, the micro DMA transfers data to the CPU assuming the highest priority level for a maskable interrupt regardless of the priority level assigned to the interrupt source. Micro DMA requests are not, however, accepted when IFF[2:0] = 7.

The micro DMA has four channels so that it can be specified for up to four interrupt sources simultaneously.

When the CPU accepts a micro DMA request, it clears the interrupt request flag assigned to that channel, performs a single data transfer (1 byte, 2 bytes or 4 bytes) from the source address to destination address, as specified with the control register, and then decrements the transfer counter. If the decremented counter reaches zero, the interrupt controller receives a request from the CPU and generates a micro DMA transfer complete interrupt (INTTCn). Then the CPU clears the micro DMA request vector register (DMAnV) to 0, thus disabling subsequent start of the micro DMA and terminating micro DMA servicing. If the decremented counter does not reach zero, the CPU terminates micro DMA servicing unless burst is specified. In that case, the interrupt controller does not generate a micro DMA transfer complete interrupt (INTTCn).

When using an interrupt source only to start the micro DMA, set the priority level for that interrupt to 0. If another interrupt request with a priority level of 1 to 6 is issued before the current interrupt is set for the micro DMA request vector, the CPU performs general interrupt servicing for the new interrupt.

When using an interrupt source for both the micro DMA and general interrupt servicing, set the priority level for that interrupt to a level less than those of all other interrupt sources. Note that only edge-triggered interrupts can be used in such a way.

A micro DMA transfer complete interrupt is serviced according to its priority level and default priorities, in the same way as other maskable interrupts.

If two or more micro DMA channels issue requests simultaneously, channels having smaller numbers have higher priorities, regardless of the respective interrupt priority levels.

The transfer source and destination addresses are specified using a 32-bit control register. The micro DMA can, however, handle only 16-Mbyte space because there are only 24 address output lines.

Assume the INT0 and INTSBI interrupts have the priority levels as shown below. When an INT0 interrupt occurs, the micro DMA determines that it is an interrupt specified with a micro DMA request vector, as shown in the interrupt servicing flow (Figure 3.4.2). If an INTSBI interrupt then occurs before the CPU reads the interrupt vector V, however, the vector changes to indicate INTSBI because its priority level is higher. Because the CPU has already determined a micro DMA request in the interrupt servicing flow, it reads the vector V for INTSBI, so that the INTSBI interrupt takes effect regardless of the micro DMA transfer counter.

INT0:	Level 1 (Not set)
INTSBI:	Level 6 (Set)

The micro DMA supports three transfer modes: 1 byte, 2 bytes or 4 bytes. For each transfer mode, the transfer source and destination addresses can be incremented, decremented or fixed after the transfer of a single unit of data. This ability to select various modes facilitates data transfer from memory to memory, peripheral to memory, memory to peripheral and peripheral to peripheral. For details of transfer modes, see (4) "Transfer mode registers".

The transfer counter consists of 16 bits, so that up to 65536 micro DMA transfers (if the counter defaults to 0000H) can be performed for a single interrupt source.

The micro DMA supports 30 interrupt sources, for which micro DMA request vectors are shown in Table 3.4.1, as well as a soft start.

Figure 3.4.2 shows micro DMA cycles for 2-byte transfer with the transfer destination address incremented, where all address areas are accessed with a 16-bit bus, no wait cycles are inserted, and both the source and destination addresses are even numbers. Cycles for other counter modes are also similar to the following.

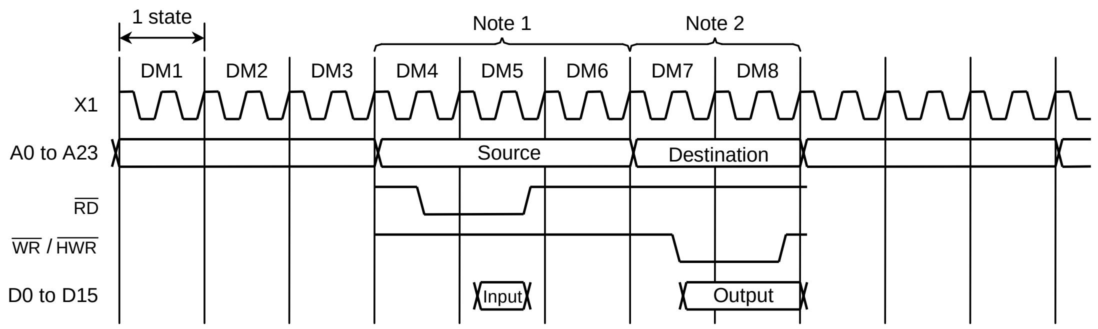


Figure 3.4.2 Micro DMA Cycles

1st to 3rd states:	Instruction fetch cycles (Prefetching next instruction code). These cycles are dummy cycles if three or more bytes of instruction code are stored in the instruction queue buffer.
4th and 5th states:	Micro DMA read cycles.
6th states:	Dummy cycle (Address bus left in 5th state).
7th and 8th states:	Micro DMA write cycles.

Note 1: Additional two states are involved if the source address area uses an 8-bit bus. If the source address area uses a 16-bit bus but starts with an odd address, additional two states are involved.

Note 2: Additional two states are involved if the destination address area uses an 8-bit bus. If the destination address area uses a 16-bit bus but starts with an odd address, additional two states are involved.

(2) Soft start

The micro DMA is usually started by an interrupt source but it also supports a soft start feature, that enables it to start upon the detection of a write cycle to the DMAR register.

Writing 1 to a bit of the DMAR register can start the corresponding micro DMA channel once (Writing 0 has no effect). Upon the completion of transfer, the DMAR register bit for that channel is automatically cleared to 0. Only a single channel can be started simultaneously (More than one bit cannot be set to 1 simultaneously) due to a restriction imposed by the specification.

A DMAR register bit must be determined to be 0 before it can be set to 1 again. If the bit is read as 1, a micro DMA transfer has not yet started.

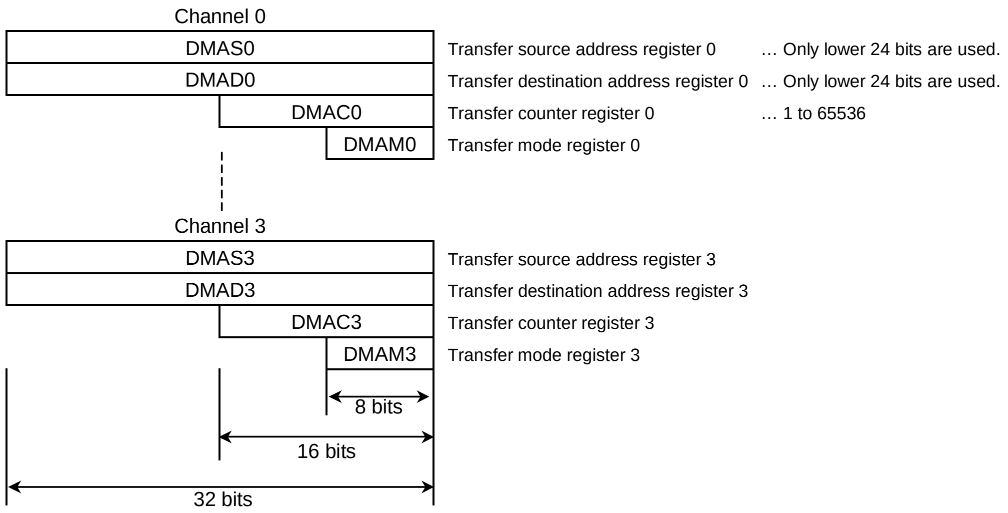
If the DMAB register specifies burst, the started micro DMA channel transfers data continuously until the micro DMA transfer counter reaches 0.

Any soft start attempted between interrupt-triggered micro DMA transfers does not cause the micro DMA transfer counter to change. To prevent other bits from being written unintentionally, no read-modify-write instruction should be used.

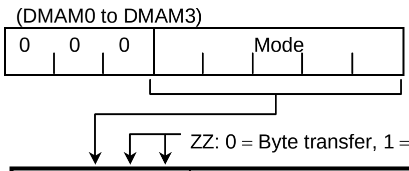
Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA request register	89H					DMA request			
							DMAR3	DMAR2	DMAR1	DMAR0
							R/W			
							0	0	0	0

(3) Transfer control registers

The following registers in the CPU are used to control the transfer source and destination addresses. Use the “LDC cr, r” instruction to set data in these registers.


Channel 0
DMAS0
DMAD0
DMAC0
DMAM0
Transfer source address register 0 … Only lower 24 bits are used.
Transfer destination address register 0 … Only lower 24 bits are used.
Transfer counter register 0 … 1 to 65536
Transfer mode register 0
Channel 3
DMAS3
DMAD3
DMAC3
DMAM3
Transfer source address register 3
Transfer destination address register 3
Transfer counter register 3
Transfer mode register 3
8 bits
16 bits
32 bits

(4) Transfer mode registers: DMAM0 to DMAM3



ZZ: 0 = Byte transfer, 1 = Word transfer, 2 = 4-byte transfer, 3 = Reserved

Note: The upper three bits of data written to these registers must always be 0.

Execution time

0 0 0 Z Z	Destination address increment mode .... Peripheral to memory
	(DMADn+) ← (DMASn)
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		8 states (1600 ns)
		Byte/word transfer
		12 states (2400 ns)
		4-byte transfer

0 0 1 Z Z	Destination address decrement mode .... Peripheral to memory
	(DMADn−) ← (DMASn)
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		8 states (1600 ns)
		Byte/word transfer
		12 states (2400 ns)
		4-byte transfer

0 1 0 Z Z	Source address increment mode.................. Memory to peripheral
	(DMADn) ← (DMASn+)
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		8 states (1600 ns)
		Byte/word transfer
		12 states (2400 ns)
		4-byte transfer

0 1 1 Z Z	Source address decrement mode.................. Memory to peripheral
	(DMADn) ← (DMASn−)
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		8 states (1600 ns)
		Byte/word transfer
		12 states (2400 ns)
		4-byte transfer

1 0 0 Z Z	Fixed address mode.................................... Peripheral to peripheral
	(DMADn) ← (DMASn)
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		8 states (1600 ns)
		Byte/word transfer
		12 states (2400 ns)
		4-byte transfer

1 0 1 0 0	Counter mode … Counting the number of interrupts that have occurred
	DMASn ← DMASn + 1
	DMACn ← DMACn − 1
	if DMACn = 0 then INTTC occurs
		5 states
		(1000 ns)

Note 1 n: Corresponding micro DMA channel (0 to 3)
DMADn+/DMASn+: Post-increment (Incrementing the register value after transfer)
DMADn−/DMASn−: Post-decrement (Decrementing the register value after transfer)
In the table, "peripheral" means a fixed address while "memory" means an address that can be incremented or decremented.

Note 2: Execution time: The time required to complete transferring a single unit of data when a 16-bit bus is used for the source and destination address areas and no wait cycles are inserted.
Clock settings: fc = 10 MHz, clock gear: 1 (fc)

Note 3: Any code other than those listed above must not be written to transfer mode registers.

3.4.3 Interrupt Controller

Figure 3.4.3 shows a block diagram of the interrupt circuit. The left-hand side of the diagram shows the interrupt controller while the right-hand side shows the CPU's interrupt request signal circuit and halt wakeup circuit.

The interrupt controller has an interrupt request flag, interrupt priority register and micro DMA request vector. The interrupt request flag is used to latch an interrupt request issued by peripherals.

This flag is cleared in the following cases:

- The device is reset.
- The CPU accepts the interrupt and reads the vector for the interrupt.
- An instruction that clears the interrupt is executed (A DMA request vector is written to the INTCLR register).
- The CPU accepts a micro DMA request for the interrupt.
- Micro DMA burst transfer for the interrupt completes.

Priority levels for individual interrupts can be specified using interrupt priority registers (such as INTE0AD and INTE12) provided for each interrupt source. Six levels of priority (1 to 6) can be set. An interrupt request is disabled when its priority level is set to 0 or 7. Nonmaskable interrupts ($\overline{\text{NMI}}$ pin and watchdog timer) have a fixed level of 7. If two or more interrupts having the same priority level occur simultaneously, the CPU accepts interrupts according to default priorities. Reading bits 3 and 7 of the interrupt priority register obtains the status of the interrupt request flag, indicating whether an interrupt request is present for a channel.

The interrupt controller determines the interrupt, and sends its priority level and vector address to the CPU. The CPU compares that priority level with the contents of the interrupt mask register, that is, the IFF[2:0] bits of the status register (SR). The CPU accepts the interrupt if its priority level is greater than the register value. It then sets the SR.IFF[2:0] bits to the accepted interrupt level plus one, so that only interrupt requests having a priority level greater than or equal to the register value can be accepted while the current interrupt is handled. Upon the completion of interrupt servicing (with the execution of the RETI instruction), the SR.IFF[2:0] bits restore the values existing before the interrupt occurred from the stack.

The interrupt controller has registers for storing micro DMA request vectors for four channels. Writing a request vector (See Table 3.4.1) to these registers enables the micro DMA to start when the corresponding interrupt occurs. Note that the micro DMA parameter registers (such as DMAS and DMAD) must be set beforehand.

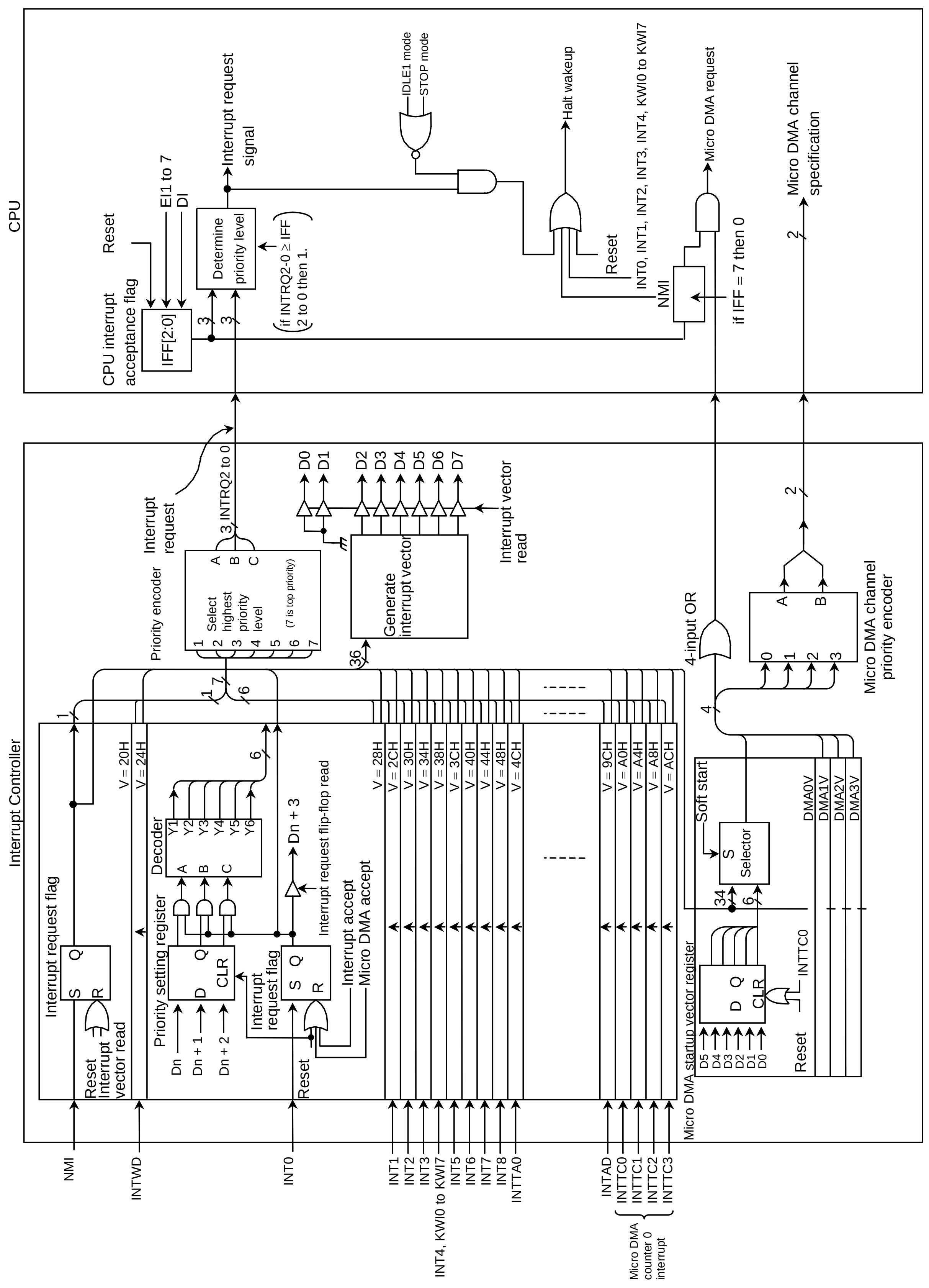
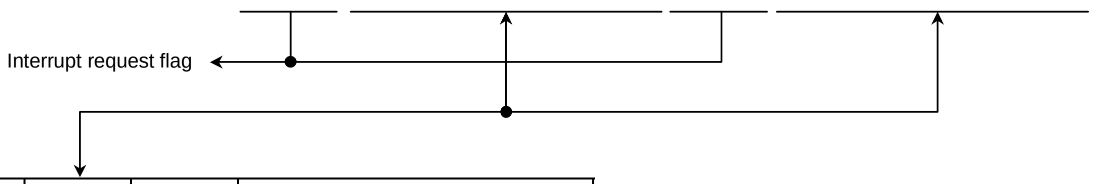


Figure 3.4.3 Interrupt Controller Block Diagram

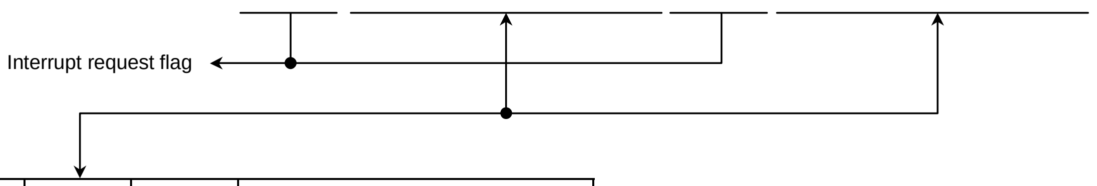
(1) Interrupt priority registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	INT0 & INTAD enable	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE12	INT1 & INT2 enable	91H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	92H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	93H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE78	INT7 & INT8 enable	94H	INT8				INT7			
			I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	95H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	96H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB0	Interrupt enabel TMRB0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	Interrupt enabel TMRB1	9AH	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0



IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disable interrupt requests.
0	0	1	Set the priority level to 1.
0	1	0	Set the priority level to 2.
0	1	1	Set the priority level to 3.
1	0	0	Set the priority level to 4.
1	0	1	Set the priority level to 5.
1	1	0	Set the priority level to 6.
1	1	1	Disable interrupt requests.

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB01V	Interrupt enable TMRB0/1 (Overflow)	9BH	INTTBOF1 (TMRB1 overflow)				INTTBOF0 (TMRB0 overflow)			
			ITF1C	ITF1M2	ITF1M1	ITF1M0	ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEBCD	Interrupt enable BCD	9CH					INTBCD			
							IBCDC	IBCD1M2	IBCD1M1	IBCD1M0
							R	R/W		
							0	0	0	0
INTES1	Interrupt enable serial 1	9DH	INTTX				INTRX			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES2	Interrupt enable SBI0/1	9EH	INTSBI1				INTSBI0			
			IS1C	IS1M2	IS1M1	IS1M0	IS0C	IS0M2	IS0M1	IS0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC01	INTTC0 & INTTC1 enable	A0H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	INTTC2 & INTTC3 enable	A1H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0



IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disable interrupt requests.
0	0	1	Set the priority level to 1.
0	1	0	Set the priority level to 2.
0	1	1	Set the priority level to 3.
1	0	0	Set the priority level to 4.
1	0	1	Set the priority level to 5.
1	1	0	Set the priority level to 6.
1	1	1	Disable interrupt requests.

Note: Bits7 to 4 of the INTEBCD are read as undefined.

(2) Controlling external interrupts

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt input mode control	8CH (RMW prohibited)	–	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	NMIREE
			W							
			0	0	0	0	0	0	0	0
			Must be written as “0”.	INT4 edge polarity 0: Rising 1: Falling	INT3 edge polarity 0: Rising 1: Falling	INT2 edge polarity 0: Rising 1: Falling	INT1 edge polarity 0: Rising 1: Falling	INT0 edge polarity 0: Rising 1: Falling	INT0 sensitivity 0: Edge-triggered 1: Level-sensitive	1: Also triggered by $\overline{\text{NMI}}$ rising edge

INT0 level detection enable

0	Edge sensitive INT
1	Active high level-sensitive INT

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

0	INT request occurs at falling edge
1	INT request occurs at rising/falling edge

(3) Interrupt request flag clear register

An interrupt request flag can be cleared by writing a micro DMA request vector (See Table 3.4.1) to the INTCLR register.

For example, the INT0 interrupt flag can be cleared by the following register operation after executing the DI instruction.

INTCLR ← 0AH: Clear the INT0 interrupt request flag

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTCLR	Interrupt clear control	88H (RMW prohibited)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					0	0	0	0	0	0
					Interrupt vector					

(4) Micro DMA request vector registers

A micro DMA request vector register specifies which interrupt source is targeted for a micro DMA request. The interrupt source having the micro DMA request vector specified in the register is assigned as the micro DMA request source.

When the micro DMA transfer counter reaches 0, the interrupt controller receives a request from the CPU and generates a micro DMA transfer complete interrupt for the relevant channel. Then, the CPU clears the micro DMA request vector register, thus clearing the micro DMA request source for the channel. If it is necessary to continue micro DMA processing for the same interrupt source, the interrupt controller must reload the micro DMA request vector into the register during the service routine for the micro DMA transfer completion interrupt.

If the same vector is set in micro DMA request vector registers for two or more channels simultaneously, the channel having the smallest number takes precedence.

When the same vector is set in micro DMA request vector registers for two channels simultaneously, micro DMA transfer is first performed with the channel having the smaller number. Once transfer completes, micro DMA transfer for the channel having the larger number starts (Micro DMA chaining), unless the interrupt controller reloads the micro DMA request vector for the first channel.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 request vector	80H			DMA0 request vector					
					DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
DMA1V	DMA1 request vector	81H			DMA1 request vector					
					DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
DMA2V	DMA2 request vector	82H			DMA2 request vector					
					DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
DMA3V	DMA3 request vector	83H			DMA3 request vector					
					DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0

(5) Micro DMA burst specification

The micro DMA supports burst specification, with which a single micro DMA startup can cause transfer to continue until the transfer counter register reaches zero. Burst transfer can be specified by setting the DMAB register bit corresponding to a micro DMA channel to 1.

If another interrupt request (Maskable or nonmaskable) is issued during a burst transfer, the CPU first completes the burst transfer before servicing the interrupt.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA software request register	89H					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA soft request			
DMAB	DMA burst register	8AH					DMAB3	DMAB2	DMAB1	DMAB0
							R/W			
							0	0	0	0
							1: DMA burst request			

(6) Precautions

The CPU consists of a separate instruction execution unit and bus interface unit. It may fetch an instruction that clears the interrupt request flag for an interrupt (Note) immediately before that instruction is issued. Once the CPU accepts an interrupt, it may execute such an instruction before reading the interrupt vector. In such a case, the CPU reads 0008H (Interrupt vector cleared) and reads the interrupt vector from address FFFF08H.

To prevent the above situation from arising, the DI instruction should be executed before an instruction for clearing an interrupt request flag. After the clear instruction is executed, at least one instruction should be executed before the EI instruction is executed to re-enable interrupts. If the EI instruction immediately follows the clear instruction, interrupts may be enabled before the interrupt flag is cleared.

When the POP SR instruction is used to modify the interrupt mask level (SR.IFF[2:0]), the DI instruction must be executed to disable interrupts before executing the POP SR instruction.

Also note the following two exceptional circuits:

INT0 level detection mode	When INT0 is used as a level-sensitive interrupt pin, rather than edge-triggered, the interrupt request flip-flop is disabled so that a peripheral interrupt request directly passes through the S input of the flip-flop to appear at the S output. Modifying the mode (Edge to level) causes the previous interrupt request flag to be cleared automatically.
	If INT0 is driven from low to high, causing the CPU to start an interrupt response sequence, INT0 must be held high until the interrupt response sequence is completed. When INT0 in level-sensitive mode is used to exit a HALT mode, INT0 must also be held high once it is driven from low to high. Ensure that it is not temporarily driven low due to noise during that period. When the INT0 detection mode is changed from level to edge, any interrupt request flag accepted in level-sensitive mode is not cleared. Use the following sequence to clear the interrupt request flag: DI LD (IIMC), 00H ; Change from level to edge. LD (INTCLR), 0AH ; Clear INT0 interrupt request flag. NOP ; Wait EI instruction. EI
INTRXn	Clearing the interrupt request flip-flop requires a system reset or reading the serial channel receive buffer. It cannot be cleared using an instruction.

Note: The following instructions and pin state transition are also equivalent to this type of instruction:

INT0: Instruction that changes the pin mode to level detection after an interrupt occurs in edge-triggered mode.
Change in the pin input (from high to low) after an interrupt occurs level-sensitive mode.

INTRXn: Instruction that reads the receive buffer.

3.5 I/O Ports

The TMP91CW28 has 80 I/O port pins. All the port pins except a few share pins with alternate functions. They can be individually programmed as general-purpose I/O or dedicated I/O for the on-chip CPU or peripherals. Table 3.5.1 shows all the I/O port pins available on the TMP91CW28 and their shared functions. Table 3.5.2 to Table 3.5.4 give a summary of register settings used to control the port pins.

Table 3.5.1 Programmable I/O Ports

Port	Pin Name	# of Pins	Direction	Pull Resistor	Direction Programmability	Alternate Functions
Port 0	P00 to P07	8	Input/output	–	Bitwise	AD0 to AD7
Port 1	P10 to P17	8	Input/output	–	Bitwise	AD8 to AD15/A8 to A15
Port 2	P20 to P27	8	Input/output	–	Bitwise	A16 to A23/A0 to A7
Port 3	P30	1	Output	–	(Fixed)	$\overline{RD}$
	P31	1	Output	–	(Fixed)	$\overline{WR}$
	P32	1	Input/output	Pull up	Bitwise	$\overline{HWR}$
	P33	1	Input/output	Pull up	Bitwise	$\overline{WAIT}$
	P34	1	Input/output	Pull up	Bitwise	$\overline{BUSRQ}$
	P35	1	Input/output	Pull up	Bitwise	$\overline{BUSAK}$
	P36	1	Input/output	Pull up	Bitwise	R/ $\overline{W}$
	P37	1	Input/output	Pull up	Bitwise	
Port 4	P40	1	Input/output	Pull up	Bitwise	$\overline{CS0}$
	P41	1	Input/output	Pull up	Bitwise	$\overline{CS1}$
	P42	1	Input/output	Pull up	Bitwise	$\overline{CS2}$
	P43	1	Input/output	Pull up	Bitwise	$\overline{CS3}$
Port 5	P50 to P57	8	Input	–	(Fixed)	AN0 to AN7, $\overline{ADTRG}$ (P53) KWI0 to KWI7
Port 6	P60	1	Input/output	–	Bitwise	SCK0
	P61	1	Input/output	Pull up	Bitwise	SO0/SDA0
	P62	1	Input/output	Pull up	Bitwise	SI0/SCL0
	P63	1	Input/output	–	Bitwise	INT0
	P64	1	Input/output	–	Bitwise	SCOUT
	P65	1	Input/output	–	Bitwise	
	P66	1	Input/output	–	Bitwise	
Port 7	P70	1	Input/output	Pull up	Bitwise	TA0IN
	P71	1	Input/output	Pull up	Bitwise	TA1OUT
	P72	1	Input/output	Pull up	Bitwise	TA3OUT
	P73	1	Input/output	Pull up	Bitwise	
	P74	1	Input/output	Pull up	Bitwise	
	P75	1	Input/output	Pull up	Bitwise	
Port 8	P80	1	Input/output	Pull up	Bitwise	TB0IN0/INT5
	P81	1	Input/output	Pull up	Bitwise	TB0IN1/INT6
	P82	1	Input/output	Pull up	Bitwise	TB0OUT0
	P83	1	Input/output	Pull up	Bitwise	TB0OUT1
	P84	1	Input/output	Pull up	Bitwise	TB1IN0/INT7
	P85	1	Input/output	Pull up	Bitwise	TB1IN1/INT8
	P86	1	Input/output	Pull up	Bitwise	TB1OUT0
	P87	1	Input/output	Pull up	Bitwise	TB1OUT1
Port 9	P90	1	Input/output	–	Bitwise	SCK1
	P91	1	Input/output	Pull up	Bitwise	SO1/SDA1
	P92	1	Input/output	Pull up	Bitwise	SI1/SCL1
	P93	1	Input/output	–	Bitwise	TXD
	P94	1	Input/output	–	Bitwise	RXD
	P95	1	Input/output	–	Bitwise	SCLK/ $\overline{CTS}$
	P96	1	Input/output	–	Bitwise	
Port A	PA0 to PA3	4	Input/output	Pull up	Bitwise	INT1 to INT4
	PA4 to PA7	4	Input/output	Pull up	Bitwise	

Table 3.5.2 I/O Port Programmability (1/3)

Port	Pin Name	Direction/Function	I/O Register Settings			
			Pn	PnCR	PnFC	PUPn
Port 0	P00 to P07	Input port	×	0	N/A	N/A
		Output port	×	1		
		AD0 to AD7 bus lines	×	×		
Port 1	P10 to P17	Input port	×	0	0	N/A
		Output port	×	1	0	
		AD8 to AD15 bus lines	×	0	1	
		A8 to A15 outputs	×	1	1	
Port 2	P20 to P27	Input port	×	0	0	N/A
		Output port	×	1	0	
		A0 to A7 outputs	×	0	1	
		A16 to A23 outputs	×	1	1	
Port 3	P30	Output port	×	N/A	1	N/A
		$\overline{\text{RD}}$ output during external accesses	1		0	
		$\overline{\text{RD}}$ always output	0		1	
	P31	Output port	×	N/A	0	N/A
		$\overline{\text{WR}}$ output during external accesses	×		1	
	P32 to P37	Input port (with pull-up disabled)	0	0	0	N/A
		Input port (with pull-up enabled)	1	0	0	
		Output port	×	1	0	
	P32	$\overline{\text{HWR}}$ output	×	1	1	N/A
	P33	$\overline{\text{WAIT}}$ input (with pull-up disabled)	0	0	N/A	N/A
		$\overline{\text{WAIT}}$ input (with pull-up enabled)	1	0		
	P34	$\overline{\text{BUSRQ}}$ input (with pull-up disabled)	0	0	1	N/A
		$\overline{\text{BUSRQ}}$ input (with pull-up enabled)	1	0	1	
	P35	$\overline{\text{BUSAK}}$ output	×	1	1	N/A
	P36	R/$\overline{\text{W}}$ output	×	1	1	N/A
Port 4	P40 to P43	Input port (with pull-up disabled)	0	0	0	N/A
		Input port (with pull-up enabled)	1	0	0	
		Output port	×	1	0	
	P40	$\overline{\text{CS0}}$ output	×	1	1	N/A
	P41	$\overline{\text{CS1}}$ output	×	1	1	N/A
	P42	$\overline{\text{CS2}}$ output	×	1	1	N/A
	P43	$\overline{\text{CS3}}$ output	×	1	1	N/A
Port 5	P50 to P57	Input port	×	N/A		
		AN[0:7] inputs (Note 1)	×			
		KWI[0:7] inputs	×			
	P53	$\overline{\text{ADTRG}}$ input (Note 2)	×			

×: Don't care

Note 1: When P50 to P57 are configured as analog channels of the ADC, the ADCH[2:0] field in the ADMOD1 register is used to select a channel(s).

Note 2: When P53 is configured as $\overline{\text{ADTRG}}$, the ADTRGE bit in the ADMOD1 register is used to enable and disable the external trigger input to the ADC.

Table 3.5.3 I/O Port Programmability (2/3)

Port	Pin Name	Direction/Function	I/O Register Settings			
			Pn	PnCR	PnFC	PUPn
Port 6	P60, P63 to P67	Input port	×	0	0	N/A
		Output port	×	1	0	
	P61, P62	Input port (with pull-up disabled)	×	×	×	0
		Input port (with pull-up disabled)	0	0	0	1
		Input port (with pull-up enabled)	1	0	0	1
		Output port	×	1	0	×
	P60	SCK0 input	×	0	0	N/A
		SCK0 output	×	1	1	
	P61	SDA0 input (with pull-up disabled)	×	×	×	0
		SDA0 input (with pull-up disabled)	0	0	0	1
		SDA0 input (with pull-up enabled)	1	0	0	1
		SDA0 output (Note 3)	×	1	1	×
		SO0 output	×	1	1	×
	P62	SI0 input (with pull-up disabled)	×	×	×	0
		SI0 input (with pull-up disabled)	0	0	0	1
		SI0 input (with pull-up enabled)	1	0	0	1
		SCL0 input (with pull-up disabled)	×	×	×	0
		SCL0 input (with pull-up disabled)	0	0	0	1
		SCL0 input (with pull-up enabled)	1	0	0	1
		SCL0 output (Note 3)	×	1	1	×
	P63	INT0 input	×	0	1	N/A
	P64	SCOUT output	×	1	1	N/A
Port 7	P70 to P75	Input port (with pull-up disabled)	0	0	0	N/A
		Input port (with pull-up enabled)	1	0	0	
		Output port	×	1	0	
	P70	TA0IN input	×	0	N/A	
	P71	TA1OUT output	×	1	1	N/A
	P72	TA3OUT output	×	1	1	N/A
Port 8	P80 to P87	Input port (with pull-up disabled)	0	0	0	N/A
		Input port (with pull-up enabled)	1	0	0	
		Output port	×	1	0	
	P80	TB0IN0, INT5 input (with pull-up disabled)	0	0	1	N/A
		TB0IN0, INT5 input (with pull-up enabled)	1	0	1	
	P81	TB0IN1, INT6 input (with pull-up disabled)	0	0	1	N/A
		TB0IN1, INT6 input (with pull-up enabled)	1	0	1	
	P82	TB0OUT0 output	×	1	1	N/A
	P83	TB0OUT1 output	×	1	1	N/A
	P84	TB1IN0, INT7 input (with pull-up disabled)	0	0	1	N/A
		TB1IN0, INT7 input (with pull-up enabled)	1	0	1	
	P85	TB1IN1, INT8 input (with pull-up disabled)	0	0	1	N/A
		TB1IN1, INT8 input (with pull-up enabled)	1	0	1	
	P86	TB1OUT0 output	×	1	1	N/A
	P87	TB1OUT1 output	×	1	1	N/A

×: Don't care

Note 3: When P61 and P62 are configured as SDA0 and SCL0 outputs for the SBI, the ODE6[2:1] field in the ODE register can be used to configure them as either push-pull or open-drain outputs. Upon reset, the default is push-pull.

Table 3.5.4 I/O Port Programmability (3/3)

Port	Pin Name	Direction/Function	I/O Register Settings			
			Pn	PnCR	PnFC	PUPn
Port 9	P90, P93 to P96	Input port	×	0	0	N/A
		Output port	×	1	0	
	P91, P92	Input port (with pull-up disabled)	×	×	×	0
		Input port (with pull-up disabled)	0	0	0	1
		Input port (with pull-up enabled)	1	0	0	1
		Output port	×	1	0	×
	P90	SCK1 input	×	0	0	N/A
		SCK1 output	×	1	1	
	P91	SDA1 input (with pull-up disabled)	×	×	×	0
		SDA1 input (with pull-up disabled)	0	0	0	1
		SDA1 input (with pull-up enabled)	1	0	0	1
		SDA1 output (Note 4)	×	1	1	×
		SO1 output	×	1	1	×
	P92	SI1 input (with pull-up disabled)	×	×	×	0
		SI1 input (with pull-up disabled)	0	0	0	1
		SI1 input (with pull-up enabled)	1	0	0	1
		SCL1 input (with pull-up disabled)	×	×	×	0
		SCL1 input (with pull-up disabled)	0	0	0	1
		SCL1 input (with pull-up enabled)	1	0	0	1
		SCL1 output (Note 4)	×	1	1	×
	P93	TXD output	×	1	1	N/A
	P94	RXD input	×	0	N/A	
	P95	SCLK input	×	0	0	N/A
		SCLK output	×	1	1	
		$\overline{\text{CTS}}$ input	×	0	0	
Port A	PA0 to PA7	Input port (with pull-up disabled)	0	0	0	N/A
		Input port (with pull-up enabled)	1	0	0	
		Output port	×	1	0	
	PA0	INT1 input (with pull-up disabled)	0	0	1	N/A
		INT1 input (with pull-up enabled)	1	0	1	
	PA1	INT2 input (with pull-up disabled)	0	0	1	N/A
		INT2 input (with pull-up enabled)	1	0	1	
	PA2	INT3 input (with pull-up disabled)	0	0	1	N/A
		INT3 input (with pull-up enabled)	1	0	1	
	PA3	INT4 input (with pull-up disabled)	0	0	1	N/A
		INT4 input (with pull-up enabled)	1	0	1	

×: Don't care

Note 4: When P91 and P92 are configured as SDA1 and SCL1 outputs for the SBI, the ODE9[2:1] field in the ODE register can be used to configure them as either push-pull or open-drain outputs. Upon reset, the default is push-pull.

Upon reset, the port pins function as general-purpose input/output ports. Pins that can be programmed for either input or output are input ports by default. Programming is necessary to use port pins for alternate functions.

Notes on using the programmable pull-up function when the bus is relinquished

When the bus is relinquished ($\overline{\text{BUSAK}} = 0$), the output buffers for AD0 to AD15, A0 to A23 and bus control signals ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{HWR}}$, R/$\overline{\text{W}}$, $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) are disabled and placed in high-impedance state. On-chip programmable pull-up resistors are, however, still active. Whether these resistors are enabled can be selected only when the pin is used in input mode.

Table 3.5.5 shows the status of pins when the bus is relinquished.

Table 3.5.5 Pin Status when the Bus is Relinquished

Pin Name	Status	
	Port Mode	Function Mode
P00 to P07 (AD0 to AD7) P10 to P17 (AD8 to AD15/ A8 to A15)	Not changed (Not placed in high-impedance state)	Placed in high-impedance state
P20 to P27 (A16 to A23)	Not changed (Not placed in high-impedance state)	Output buffer disabled (after the pin is driven high)
P30 ($\overline{\text{RD}}$) P31 ($\overline{\text{WR}}$)	Not changed (Not placed in high-impedance state)	Output buffer disabled (after the pin is driven high)
P32 ($\overline{\text{HWR}}$) P37	Not changed (Not placed in high-impedance state)	Output buffer disabled. The on-chip pull-up resistor is enabled regardless of the value contained in the output latch.
P36 (R/$\overline{\text{W}}$) P40 ($\overline{\text{CS0}}$) P41 ($\overline{\text{CS1}}$) P42 ($\overline{\text{CS2}}$) P43 ($\overline{\text{CS3}}$)	Not changed (Not placed in high-impedance state)	Output buffer disabled. The on-chip pull-up resistor is enabled regardless of the value contained in the output latch.

Figure 3.5.1 shows an example external interface for the above signals when the bus relinquish function is used.

When the bus is relinquished, the on-chip memory and peripherals cannot be accessed but the on-chip peripherals continue operation, so that the watchdog timer (WDT) keeps counting. The period for which the bus is relinquished must be taken into account to set the WDT time-out period when using the bus relinquish function.

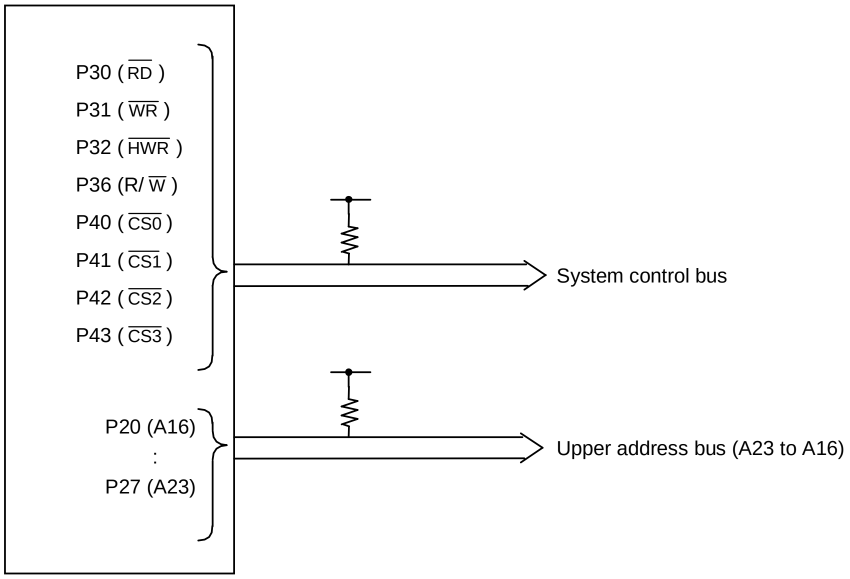


Figure 3.5.1 Example External Bus Interface when the Bus Relinquish Function is Used

A circuit as shown above is required when an external pull-up resistor is connected to fix a signal level with the bus relinquished.

Upon reset, P30 ($\overline{RD}$) and P31 ($\overline{WR}$) are output pins while P40 to P43 ($\overline{CS0}$ to $\overline{CS3}$), P32 ($\overline{HWR}$), P36 (R/$\overline{W}$) and P35 ($\overline{BUSAK}$) are input pins with pull-up resistors enabled.

3.5.1 Port 0 (P00 to P07)

Eight port 0 pins function as either discrete general-purpose I/O pins or the AD[0:7] bits of the address/data bus. The P0CR register controls the direction of the port 0 pins. Upon reset, the P0CR register bits are cleared, configuring all port 0 pins as inputs.

During external memory accesses, port 0 pins are automatically configured as AD[0:7], with the P0CR register bits all cleared.

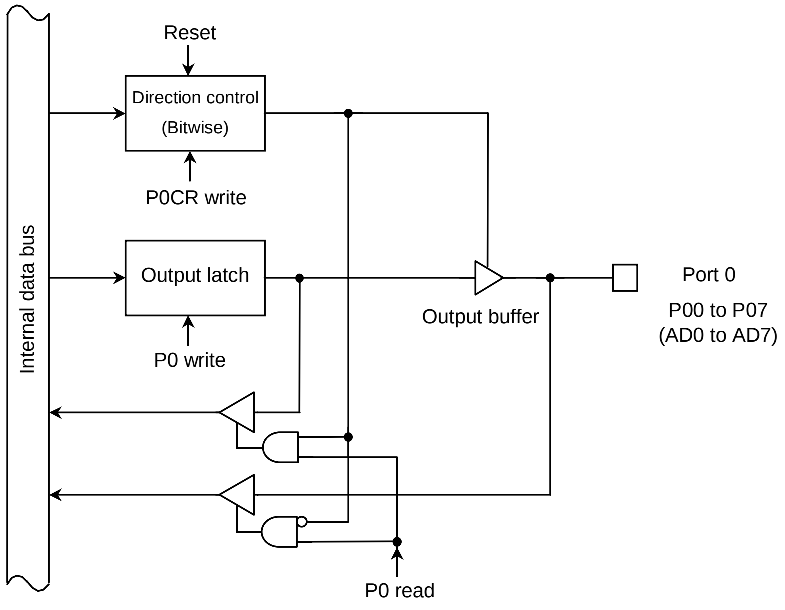


Figure 3.5.2 Port 0

3.5.2 Port 1 (P10 to P17)

Eight port 1 pins can be individually programmed to function as discrete general-purpose I/O pins, the AD[8:15] bits of the address/data bus or the A[8:15] bits of the address bus. The P1CR and P1FC registers select the direction and function of the port 1 pins. Upon reset, the output latch (P1) is cleared, and the P1CR and P1FC register bits are cleared to all 0s, configuring all port 1 pins as input port pins.

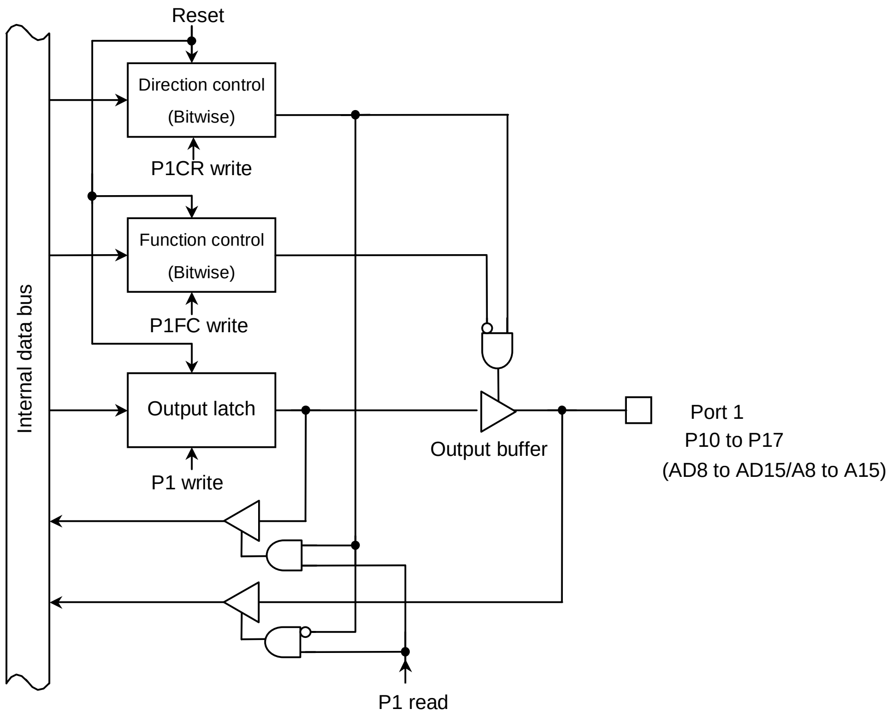


Figure 3.5.3 Port 1

Port 0 Register

P0 (0000H)

	7	6	5	4	3	2	1	0
Bit symbol	P07	P06	P05	P04	P03	P02	P01	P00
Read/Write	R/W							
Reset value	Data from external port (Output latch register is undefined)							

Port 0 Control Register

P0CR (0002H)

	7	6	5	4	3	2	1	0
Bit symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	0: Input 1: Output (Functions as AD7 to AD0 during external memory accesses, with all bits cleared.)							

Port 0 direction settings

0	Input port
1	Output port

Port 1 Register

P1 (0001H)

	7	6	5	4	3	2	1	0
Bit symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
Reset value	Data from external port (Output latch register is cleared to 0)							

Port 1 Control Register

P1CR (0004H)

	7	6	5	4	3	2	1	0
Bit symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	<<Refer to P1FC.>>							

Port 1 Function Register

P1FC (0005H)

	7	6	5	4	3	2	1	0
Bit symbol	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	P1FC/P1CR = 00: Input, 01: Output, 10: AD15 to AD8, 11: A15 to A8							

Note: P0CR, P1CR, and P1FC do not support read-modify-write operation.

Port 1 function settings

P1CR.P1XC \ P1FC.P1XF	0	1
0	Input port	Address/Data bus (AD15 to AD8)
1	Output port	Address bus (A15 to A8)

Note: P1XF and P1XC mean bit X in the P1FC and P1CR registers respectively.

Figure 3.5.4 Port 0 and 1 Registers

3.5.3 Port 2 (P20 to P27)

Eight port 2 pins can be individually programmed to function as discrete general-purpose I/O pins, the A[0:7] bits of the address bus or the A[16:23] bits of the address bus. The P2CR and P2FC registers select the direction and function of the port 2 pins. Upon reset, the output latch (P2) is cleared, and the P2CR and P2FC register bits are cleared to all 0s, configuring all port 2 pins as input port pins.

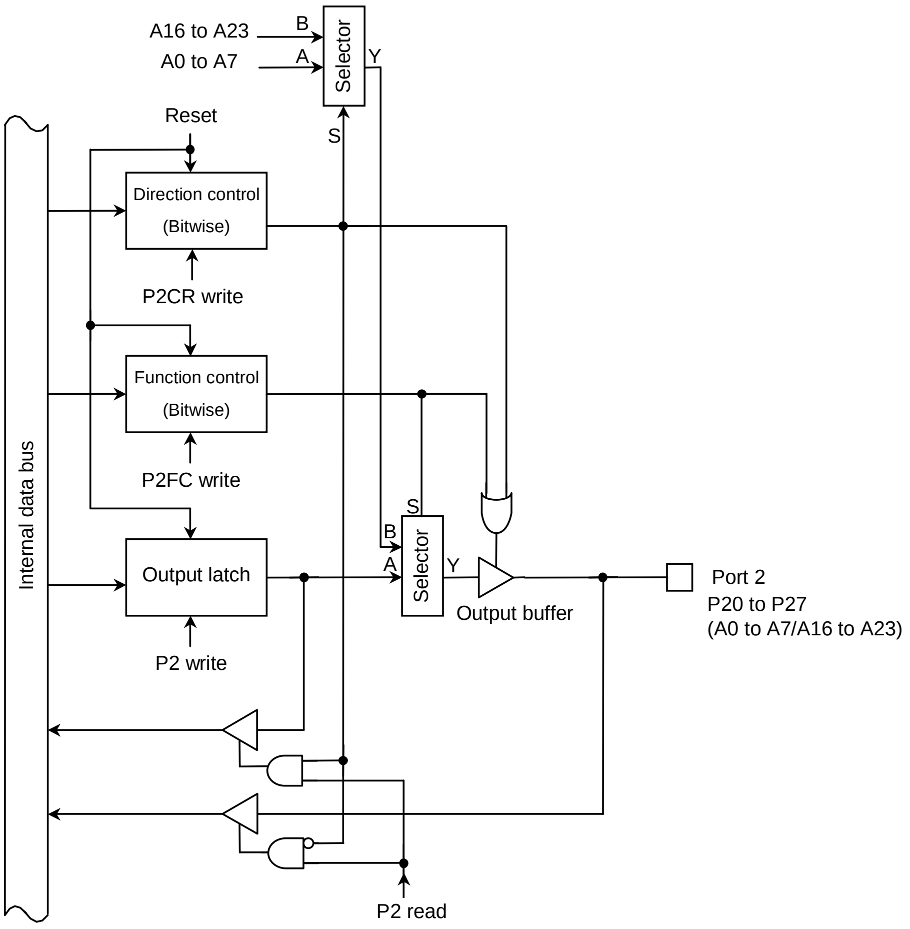


Figure 3.5.5 Port 2

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							
Reset value	Data from external port (Output latch register is set to 1)							

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							
Reset value	0	0	0	0	0	0	0	0
Function	Refer to P2FC							

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							
Reset value	0	0	0	0	0	0	0	0
Function	P2FC/P2CR = 00: Input, 01: Output, 10: A7 to A0, 11: A23 to A16							

Note: P2CR and P2FC do not support read-modify-write operation.

→ Port 2 function settings

P2CR.P2XC \ P2FC.P2XF	0	1
0	Input port	Address bus (A7 to A0)
1	Output port	Address bus (A23 to A16)

Note: P2XF and P2XC mean bit X in the P2FC and P2CR registers respectively.
When using the pin as address bus A23 to A16, set the P2CR before setting the P2FC.

Figure 3.5.6 Port 2 Registers

3.5.4 Port 3 (P30 to P37)

Eight port 3 pins can be individually programmed to function as either discrete general-purpose I/O pins or CPU control/status pins. In either case, P30 and P31 are output-only pins.

The P3CR and P3FC registers select the direction and function of the port 3 pins. Upon reset, the P3CR and P3FC register bits are cleared, configuring P30 and P31 as output port pins and P32 to P37 as input port pins with pull-up enabled. (Bits 0 and 1 in the P3CR and bits 3 and 7 in the P3FC are unused.) Upon reset, the output latch (P3) is set to all 1s; so a logic 1 appears on P30 and P31.

When P30 is configured as $\overline{\text{RD}}$ (P3FC.P30F = 1), the read strobe signal is activated only when external address space is accessed, if the P30 bit of the output latch is set to 1 (Default). Clearing P30 to 0 enables the read strobe signal to be also activated when on-chip address space is accessed. This feature is provided for accessing pseudo static RAM.

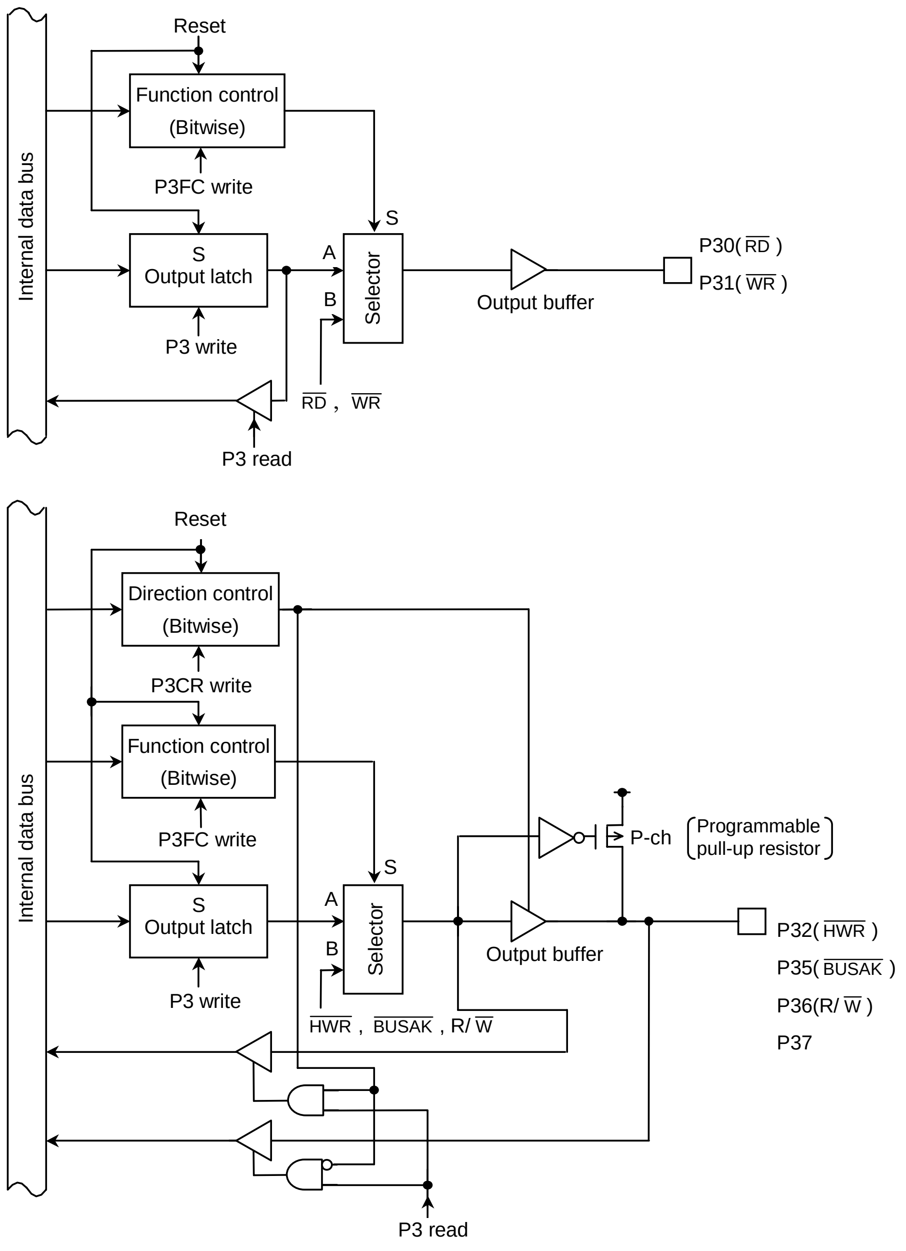


Figure 3.5.7 Port 3 (P30, P31, P32, P35, P36, P37)

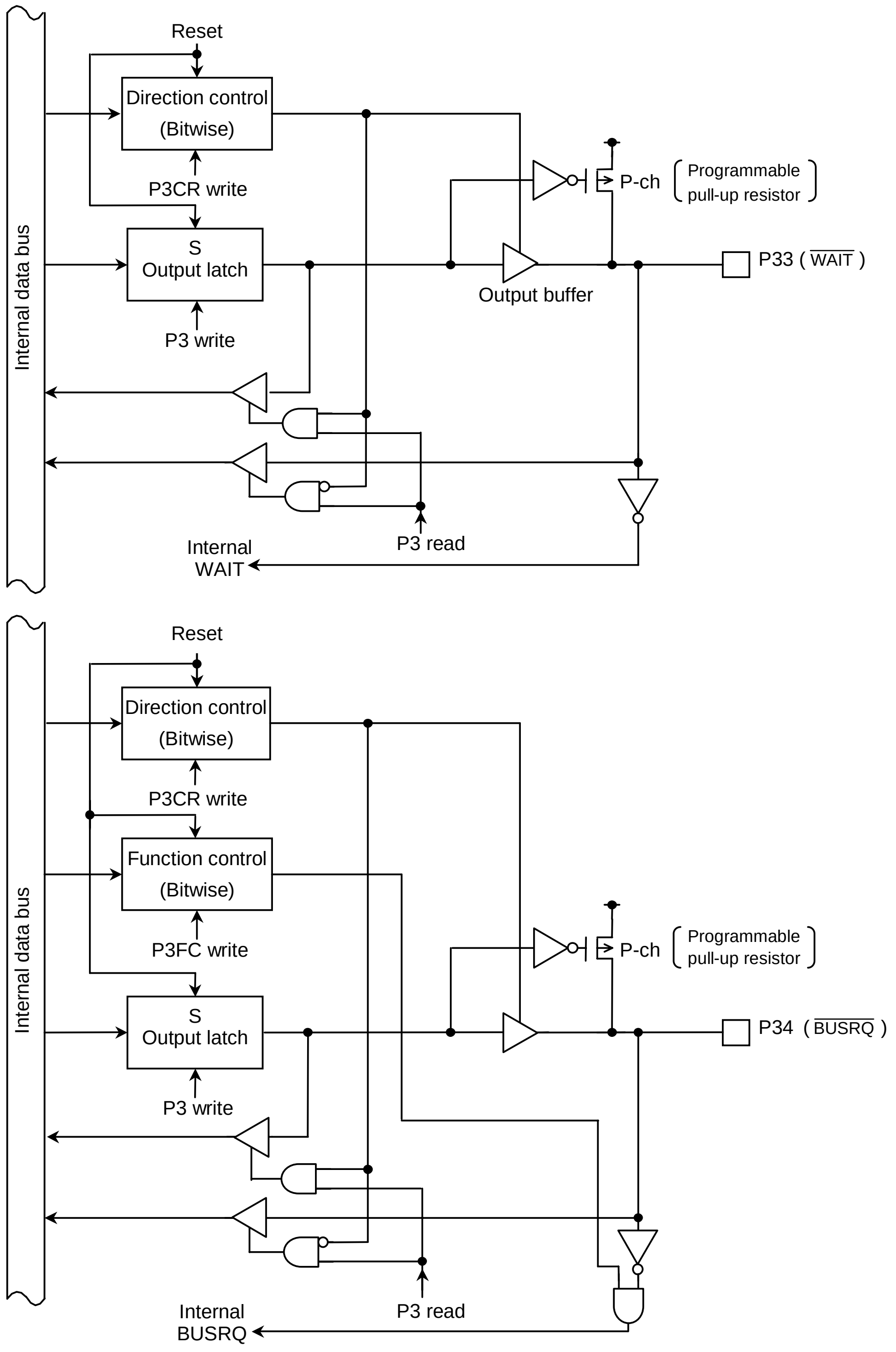

Reset
Direction control (Bitwise)
P3CR write
S Output latch
P3 write
P-ch
Programmable pull-up resistor
P33 ($\overline{\text{WAIT}}$)
Output buffer
Internal data bus
Internal WAIT
P3 read
Reset
Direction control (Bitwise)
P3CR write
Function control (Bitwise)
P3FC write
S Output latch
P3 write
P-ch
Programmable pull-up resistor
P34 ($\overline{\text{BUSRQ}}$)
Internal data bus
Internal BUSRQ
P3 read

Figure 3.5.8 Port 3 (P33, P34)

Port 3 Register

P3 (0007H)		7	6	5	4	3	2	1	0
	Bit symbol	P37	P36	P35	P34	P33	P32	P31	P30
	Read/Write	R/W							
	Reset value	Data from external port (Output latch register is set to 1)						1	1
	Function	0: Pull-up resistor disabled 1: Pull-up resistor enabled						–	

Port 3 Control Register

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

Port 3 direction settings

0	Input port
1	Output port

Port 3 Function Register

P3FC (000BH)		7	6	5	4	3	2	1	0
	Bit symbol	–	P36F	P35F	P34F		P32F	P31F	P30F
	Read/Write	W					W		
	Reset value	0	0	0	0		0	0	0
	Function	Must be written as “0”.	0: Port 1: R/ $\overline{\text{W}}$	0: Port 1: $\overline{\text{BUSAK}}$	0: Port 1: $\overline{\text{BUSRQ}}$		0: Port 1: $\overline{\text{HWR}}$	0: Port 1: $\overline{\text{WR}}$	0: Port 1: $\overline{\text{RD}}$

$\overline{\text{BUSRQ}}$ settings

P3FC.P34F	1
P3CR.P34C	0

$\overline{\text{BUSAK}}$ settings

P3FC.P35F	1
P3CR.P35C	1

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

P3FC.P36F	1
P3CR.P36C	1

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

P30F \ P30	0	1
0	Output a “0”.	Output a “1”.
1	Always assert $\overline{\text{RD}}$ (for pseudo SRAM).	Assert $\overline{\text{RD}}$ only during external accesses.

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

P31F \ P31	0	1
0	Output a “0”.	Output a “1”.
1	Assert $\overline{\text{WR}}$ only during external accesses.	

$\overline{\text{HWR}}$ settings

P3FC.P32F	1
P3CR.P32C	1

Note 1: P3CR and P3FC do not support read-modify-write operation.
Note 2: When a port 3 pin is used in input mode, the P3 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 3 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.
Note 3: When the P33/ $\overline{\text{WAIT}}$ pin is used as the $\overline{\text{WAIT}}$ pin, the P3CR.P33C bit must be cleared and the BnW[2:0] bits (Bits 3 to 1 of the chip select/wait control register) must be set to 010.

Figure 3.5.9 Port 3 Registers

3.5.5 Port 4 (P40 to P43)

Four port 4 pins can be individually programmed to function as either discrete general-purpose I/O pins or programmable chip select ($\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) pins. The P4CR and P4FC registers select the direction and function of the port 4 pins. Upon reset, the output latch (P4) is set to all 1s and the P4CR and P4FC register bits are cleared, configuring all the port 4 pins as input port pins having internal pull-up resistors.

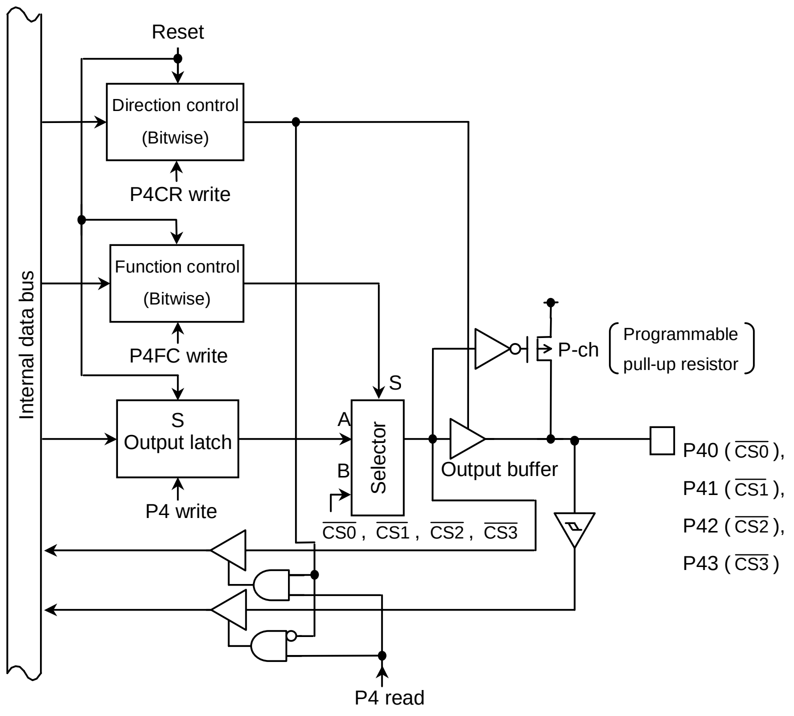


Figure 3.5.10 Port 4

Port 4 Register

P4 (000CH)		7	6	5	4	3	2	1	0
	Bit symbol					P43	P42	P41	P40
	Read/Write					R/W			
	Reset value					Data from external port (Output latch register set to 1)			
	Function					0: Pull-up resistor disabled 1: Pull-up resistor enabled			

Port 4 Control Register

P4CR (000EH)		7	6	5	4	3	2	1	0
	Bit symbol					P43C	P42C	P41C	P40C
	Read/Write					W			
	Reset value					0	0	0	0
	Function					0: Input 1: Output			

Port 4 direction settings

0	Input port
1	Output port

Port 4 Function Register

P4FC (000FH)		7	6	5	4	3	2	1	0
	Bit symbol					P43F	P42F	P41F	P40F
	Read/Write					W			
	Reset value					0	0	0	0
	Function					0: Port 1: $\overline{CS}$			

0	Port (P40)
1	$\overline{CS0}$

0	Port (P41)
1	$\overline{CS1}$

0	Port (P42)
1	$\overline{CS2}$

0	Port (P43)
1	$\overline{CS3}$

Note 1: P4CR and P4FC do not support read-modify-write operation.

Note 2: When a port 4 pin is used in input mode, the P4 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 4 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Note 3: To output chip select signals ($\overline{CS0}$ to $\overline{CS3}$), first set the corresponding bits in the function register (P4FC) to 1 and then set the corresponding bits in the control register (P4CR) to 1.

Figure 3.5.11 Port 4 Registers

3.5.6 Port 5 (P50 to P57)

Eight port 5 pins are input-only pins shared with the analog input pins of the AD converter (ADC) as well as KWI input pins. P53 is also shared with the AD trigger input pin.

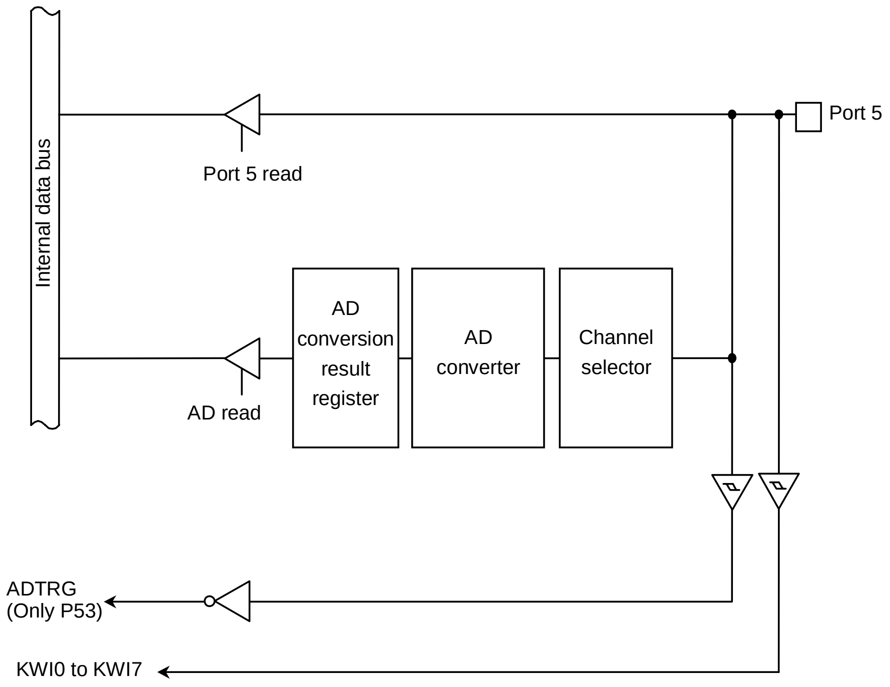


Figure 3.5.12 Port 5

Port 5 Register

		7	6	5	4	3	2	1	0
P5 (000DH)	Bit symbol	P57	P56	P55	P54	P53	P52	P51	P50
	Read/Write	R							
	Reset value	Data from external port							

Figure 3.5.13 Port 5 Register

Note 1: The AD converter mode register (ADMOD1) is used to select an AD converter input channel(s) and to enable the AD trigger input for P53.

Note 2: The KWI control register (KWIC) is used to select a KWI input channel(s) and to enable the KWI input.

3.5.7 Port 6 (P60 to P66)

Seven port 6 pins can be individually programmed to function as either discrete general-purpose I/O pins or serial bus interface (SBI) pins. Upon reset, the output latch (P6) is set to all 1s and all port 6 pins are configured as input port pins.

Setting the P6FC register bits configures the corresponding port 6 pins for dedicated functions.

A reset clears all the P6CR and P6FC register bits, configuring all port 6 pins as input port pins; P61 and P62 have an internal pull-up resistor.

(1) P60 (SCK0)

P60 can be programmed to function as a general-purpose I/O pin or an SCK0 clock input or output pin for the serial bus interface in SIO mode.

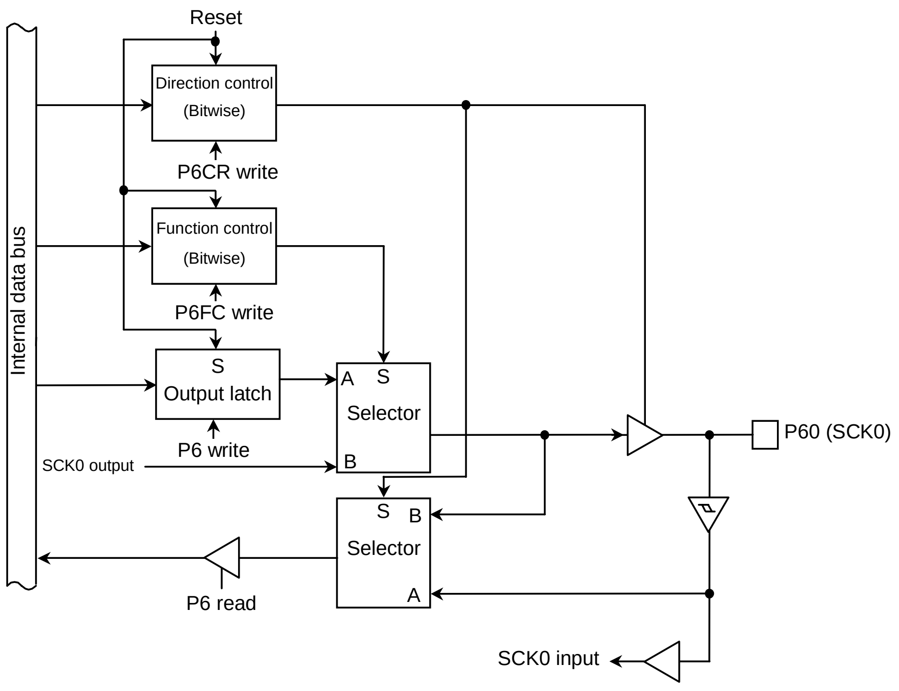


Figure 3.5.14 Port 6 (P60)

(2) P61 (SO0/SDA0)

P61 can be programmed to function as a general-purpose I/O pin, an SDA0 data input pin for the serial bus interface in I²C bus mode, or an SO0 data output pin for the serial bus interface in SIO mode. P61 is configurable as an open-drain output and has a programmable pull-up resistor.

When P61 is used as an open-drain output and does not require a pull-up resistor, the pull-up resistor can be disconnected by clearing the PUP.PUP61 register bit to 0.

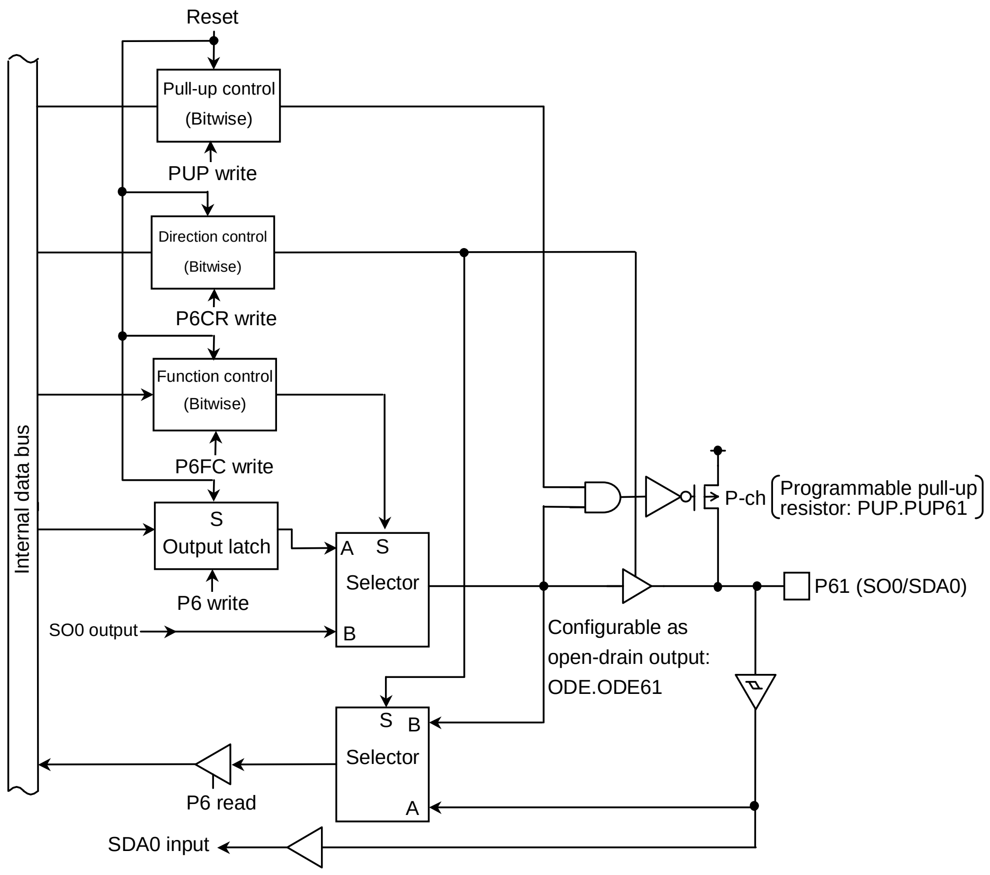

Reset
Pull-up control (Bitwise)
PUP write
Direction control (Bitwise)
P6CR write
Function control (Bitwise)
P6FC write
S
Output latch
P6 write
Internal data bus
SO0 output
A S Selector B
P-ch
Programmable pull-up resistor: PUP.PUP61
P61 (SO0/SDA0)
Configurable as open-drain output: ODE.ODE61
S B Selector A
P6 read
SDA0 input

Figure 3.5.15 Port 6 (P61)

(3) P62 (SI0/SCL0)

P62 can be programmed to function as a general-purpose I/O pin, an SI0 data input pin for the serial bus interface in SIO mode, or an SCL0 clock input or output pin for the serial bus interface in I²C bus mode. P62 is configurable as an open-drain output and has a programmable pull-up resistor.

When P62 is used as an open-drain output and does not require a pull-up resistor, the pull-up resistor can be disabled by clearing the PUP.PUP62 register bit to 0.

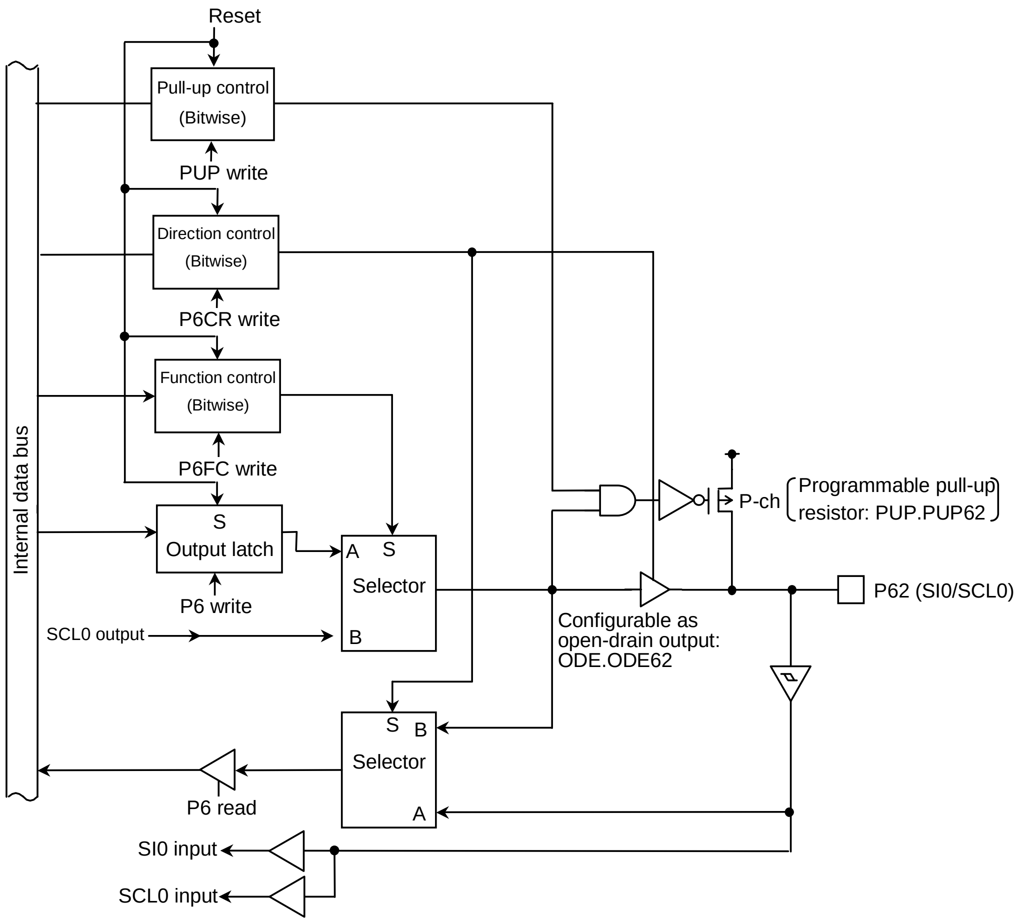

Reset
Pull-up control (Bitwise)
PUP write
Direction control (Bitwise)
P6CR write
Function control (Bitwise)
P6FC write
S
Output latch
P6 write
Internal data bus
SCL0 output
A S
Selector
B
P-ch
Programmable pull-up resistor: PUP.PUP62
P62 (SI0/SCL0)
Configurable as open-drain output: ODE.ODE62
S B
Selector
A
P6 read
SI0 input
SCL0 input

Figure 3.5.16 Port 6 (P62)

(4) P63 (INT0)

P63 can be programmed to function as a general-purpose I/O pin or an external interrupt request pin (INT0).

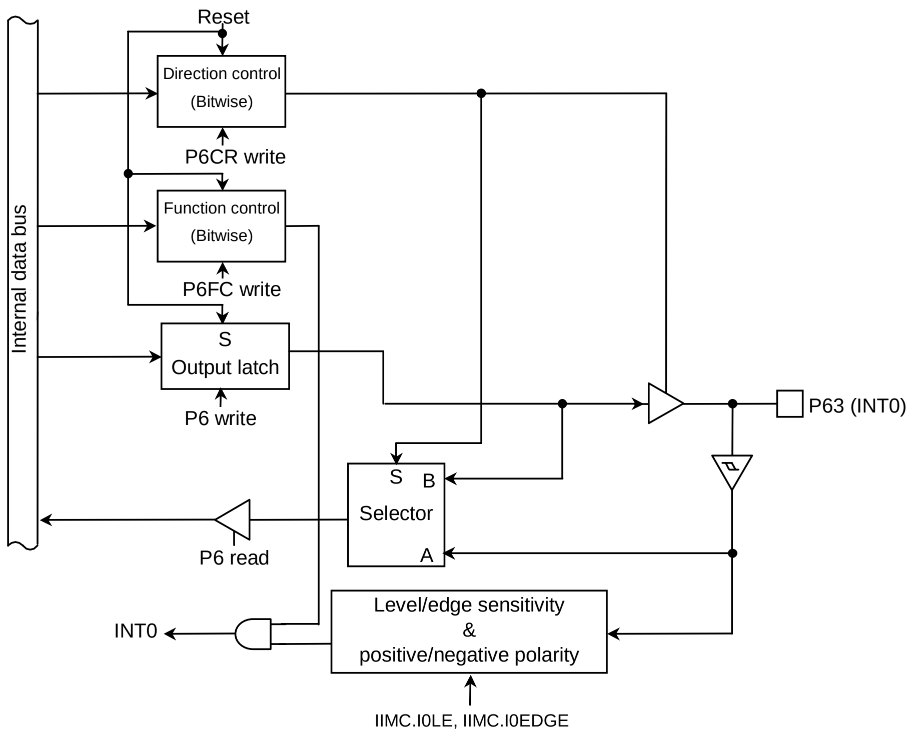


Figure 3.5.17 Port 6 (P63)

(5) P64 (SCOUT)

P64 can be programmed to function as a general-purpose I/O pin or a system clock output (SCOUT) pin.

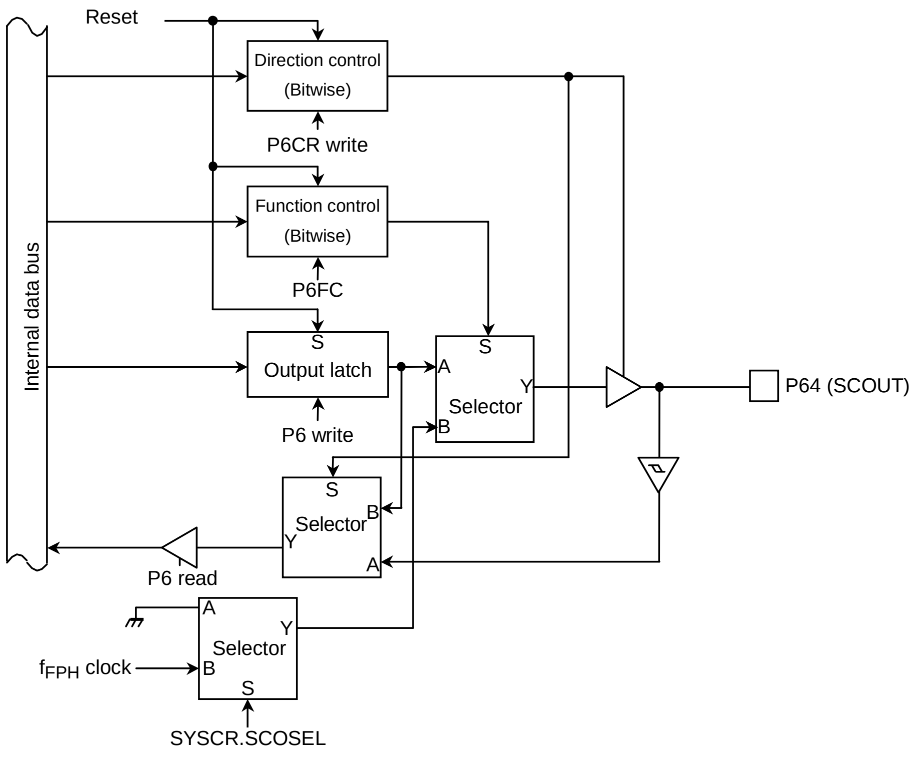


Figure 3.5.18 Port 6 (P64)

(6) P65 and P66

P65 and P66 function as general-purpose I/O pins.

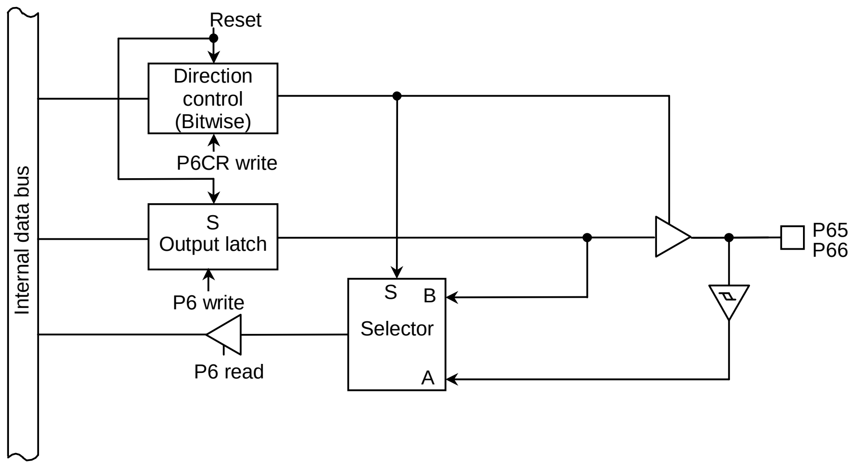


Figure 3.5.19 Port 6 (P65, P66)

Port 6 Register

P6 (0012H)		7	6	5	4	3	2	1	0
	Bit symbol		P66	P65	P64	P63	P62	P61	P60
	Read/Write		R/W						
	Reset value		Data from external port (Output latch register is set to 1)						
	Function		–				0: Pull-up resistor disabled 1: Pull-up resistor enabled		–

Port 6 Control Register

P6CR (0014H)		7	6	5	4	3	2	1	0
	Bit symbol		P66C	P65C	P64C	P63C	P62C	P61C	P60C
	Read/Write		W						
	Reset value		0	0	0	0	0	0	0
	Function		0: Input 1: Output						

Port 6 direction settings

0	Input port
1	Output port

Port 6 Function Register

P6FC (0015H)		7	6	5	4	3	2	1	0
	Bit symbol				P64F	P63F	P62F	P61F	P60F
	Read/Write				W	W	W	W	W
	Reset value				0	0	0	0	0
	Function				0: Port 1: SCOUT output	0: Port 1: INT0 input	0: Port 1: SCL0 output	0: Port 1: SDA0/SO0 output	0: Port 1: SCK0 output

SCK0 output settings for P60

P6FC.P60F	1
P6CR.P60C	1

SDA0/SO0 output settings for P61

P6FC.P61F	1
P6CR.P61C	1

SCL0 output settings for P62

P6FC.P62F	1
P6CR.KP62C	1

INT0 input settings for P63

P6FC.P63F	1
P6CR.P63C	0

SCOUT output settings for P64

P6FC.P64F	1
P6CR.P64C	1

Note 1: P6CR and P6FC do not support read-modify-write operation.

Note 2: When a port 6 pin is used in input mode, the P6 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 6 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Figure 3.5.20 Port 6 Registers

3.5.8 Port 7 (P70 to P75)

Six port 7 pins can be individually programmed to function as discrete general-purpose or dedicated I/O pins. Upon reset, all port 7 pins are configured as input port pins. Alternatively, P70 can be programmed as the clock input (TA0IN) to 8-bit timer 0. P71 and P72 can each be programmed as the timer output (TA1OUT or TA3OUT) from an 8-bit timer. Setting the P7FC register bits configures the corresponding port 7 pins as timer output pins. A reset clears the P7CR and P7FC register bits, configuring all port 7 pins as input port pins having internal pull-up resistors.

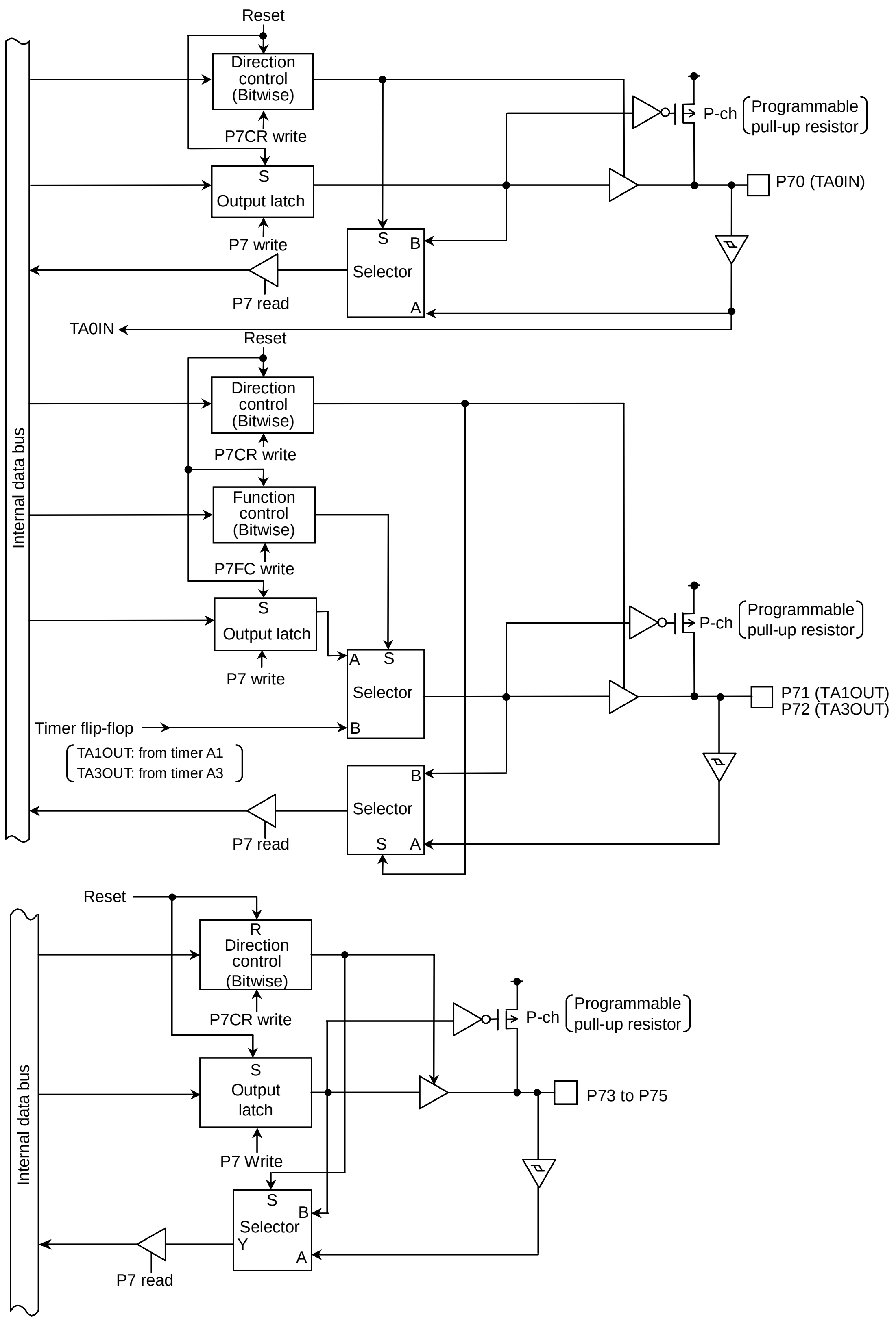


Figure 3.5.21 Port 7

Port 7 Register

		7	6	5	4	3	2	1	0
P7 (0013H)	Bit symbol			P75	P74	P73	P72	P71	P70
	Read/Write			R/W					
	Reset value			Data from external port (Output latch register is set to 1)					
	Function			0: Pull-up resistor disabled 1: Pull-up resistor enabled					

Port 7 Control Register

		7	6	5	4	3	2	1	0
P7CR (0016H)	Bit symbol			P75C	P74C	P73C	P72C	P71C	P70C
	Read/Write			W					
	Reset value			0	0	0	0	0	0
	Function			0: Input 1: Output					

Port 7 direction settings

0	Input port
1	Output port

Port 7 Function Register

		7	6	5	4	3	2	1	0
P7FC (0017H)	Bit symbol						P72F	P71F	
	Read/Write						W		
	Reset value						0	0	
	Function						0: Port 1: TA3OUT	0: Port 1: TA1OUT	

Timer output 1 settings for P71

P7FC.P71F	1
P7CR.P71C	1

Timer output 3 settings for P72

P7FC.P72F	1
P7CR.P72C	1

Note 1: P7CR and P7FC do not support read-modify-write operation.

Note 2: When a port 7 pin is used in input mode, the P7 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 7 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Note 3: The P70/TA0IN pin does not have a register bit for selecting the port or timer function. The input to the P70/TA0IN pin is always directed to 8-bit timer 0 even when the pin is used as a general-purpose input pin.

Figure 3.5.22 Port 7 Registers

3.5.9 Port 8 (P80 to P87)

Eight port 8 pins can be individually programmed to function as discrete general-purpose or dedicated I/O pins. Upon reset, all port 8 pins are configured as input port pins, and the output latch (P8) is set to all 1s. port 8 pins can be programmed as clock inputs to 16-bit timers, timer flip-flop outputs from 16-bit timers, or external interrupt request pins (INT5 through INT8). Setting the P8FC register bits configures the corresponding port 8 pins for dedicated functions. A reset clears the P8CR and P8FC register bits, configuring all port 8 pins as input port pins having internal pull-up resistors.

(1) P80 to P87

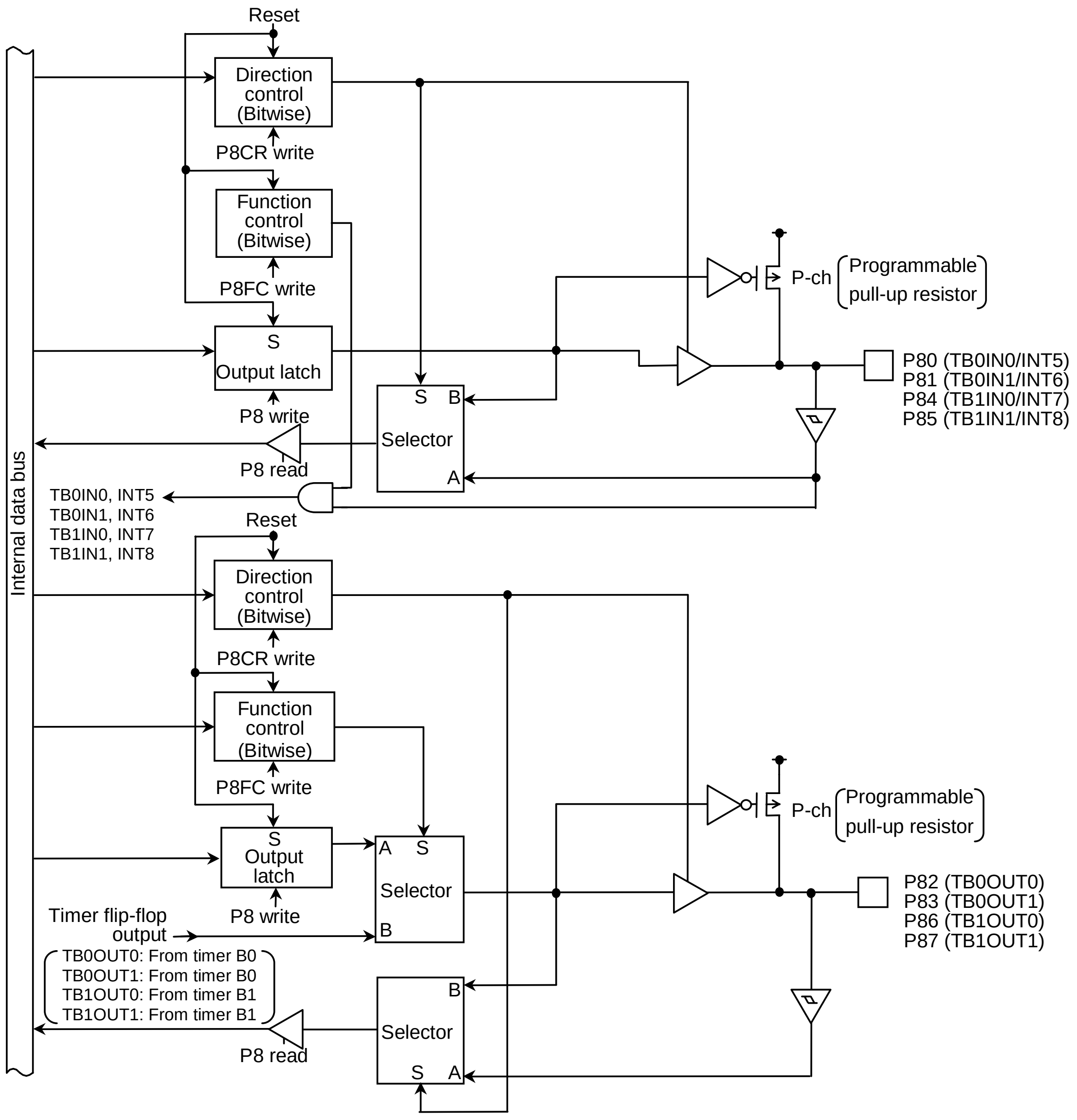


Figure 3.5.23 Port 8 (P80 to P87)

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							
Reset value	Data from external port (Output latch register is set to 1)							
Function	0: Pull-up resistor disabled 1: Pull-up resistor enabled							

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							
Reset value	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 8 direction settings

0	Input port
1	Output port

Port 8 Function Register

P8FC (001BH)

	7	6	5	4	3	2	1	0
Bit symbol	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
Read/Write	W	W	W	W	W	W	W	W
Reset value	0	0	0	0	0	0	0	0
Function	0: Port 1: TB1OUT1	0: Port 1: TB1OUT0	0: Port 1: TB1IN1, INT8 input	0: Port 1: TB1IN1, INT7 input	0: Port 1: TB0OUT1	0: Port 1: TB0OUT0	0: Port 1: TB0IN1 INT6 input	0: Port 1: TB0IN0 INT5 input

Timer output 4 settings for P82

P8FC.P82F	1
P8CR.P82C	1

Timer output 5 settings for P83

P8FC.P83F	1
P8CR.P83C	1

Timer output 6 settings for P86

P8FC.P86F	1
P8CR.P86C	1

Timer output 7 settings for P87

P8FC.P87F	1
P8CR.P87C	1

Note 1: P8CR and P8FC do not support read-modify-write operation.

Note 2: When a port 8 pin is used in input mode, the P8 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 8 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Figure 3.5.24 Port 8 Registers

3.5.10 Port 9 (P90 to P96)

Seven port 9 pins can be individually programmed to function as discrete general-purpose or dedicated I/O pins. Upon reset, all port 9 pins are configured as input port pins, and the output latch (P9) is set to all 1s. port 9 pins can be programmed as SIO input or output pins.

Setting the P9FC register bits configures the corresponding port 9 pins for dedicated functions.

A reset clears all the P9CR and P9FC register bits, configuring all port 9 pins as input port pins; P91 and P92 have an internal pull-up resistor.

(1) P90 (SCK1)

P90 can be programmed to function as a general-purpose I/O pin or an SCK1 clock input or output pin for the serial bus interface in SIO mode.

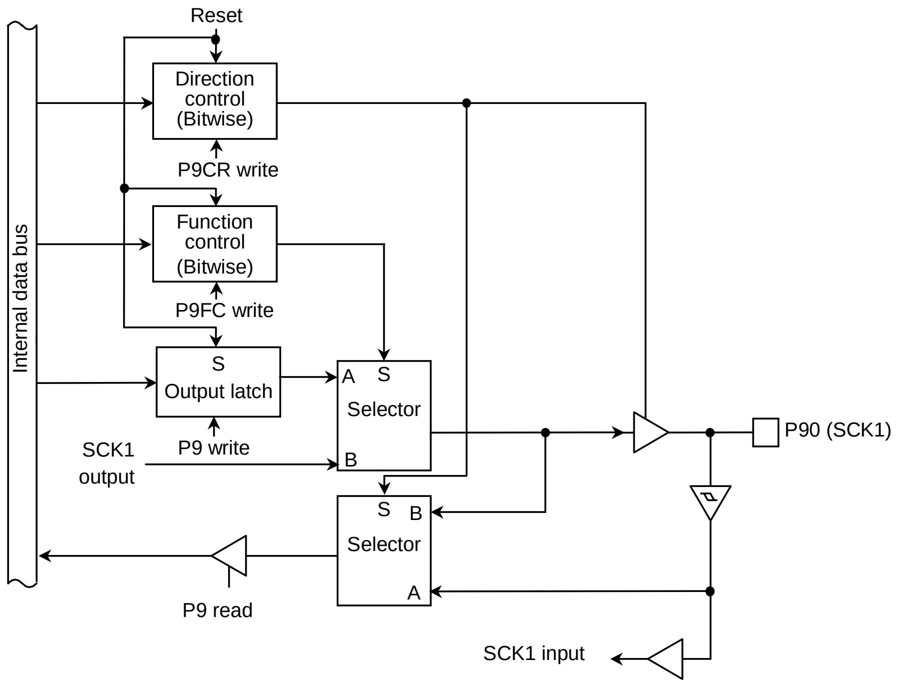
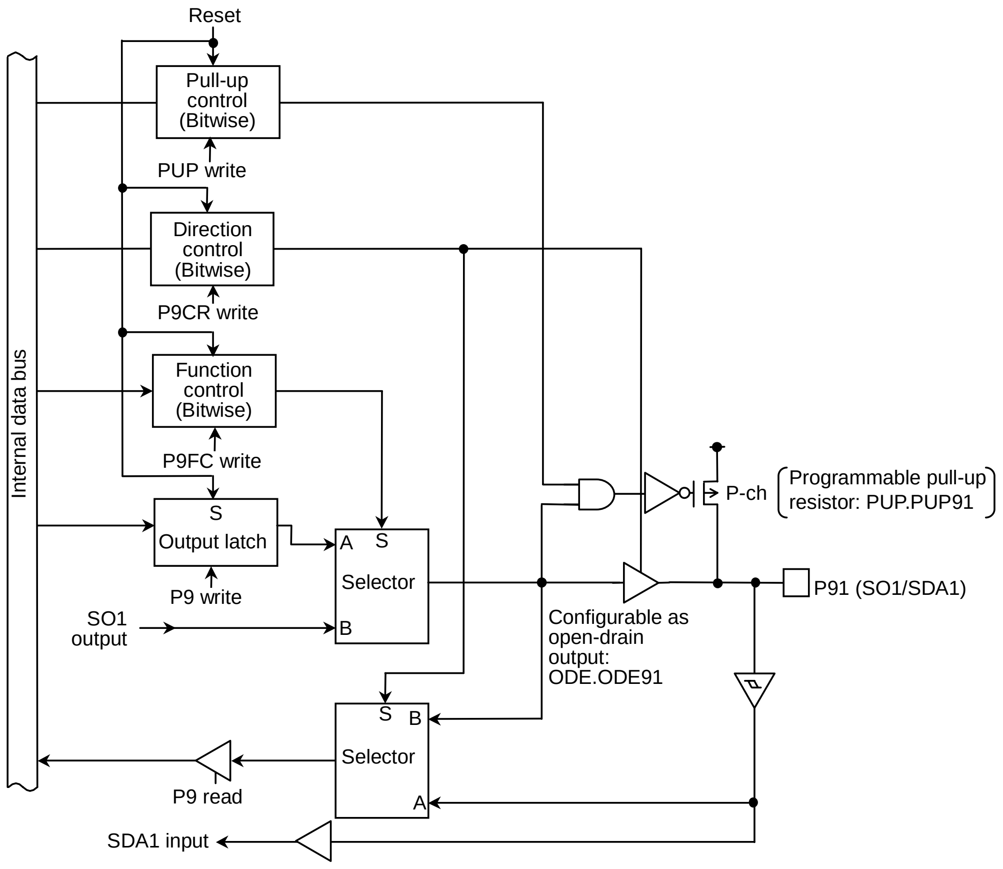


Figure 3.5.25 Port 9 (P90)

(2) P91 (SO1/SDA1)

P91 can be programmed to function as a general-purpose I/O pin, an SDA1 data input pin for the serial bus interface in I²C bus mode, or an SO1 data output pin for the serial bus interface in SIO mode. P91 is configurable as an open-drain output and has a programmable pull-up resistor.

When P91 is used as an open-drain output and does not require a pull-up resistor, the pull-up resistor can be disabled by clearing the PUP.PUP91 register bit to 0.



Reset
Internal data bus
Pull-up control (Bitwise)
PUP write
Direction control (Bitwise)
P9CR write
Function control (Bitwise)
P9FC write
S
Output latch
P9 write
SO1 output
A S
Selector
B
P-ch
Programmable pull-up resistor: PUP.PUP91
P91 (SO1/SDA1)
Configurable as open-drain output: ODE.ODE91
S B
Selector
A
P9 read
SDA1 input

Figure 3.5.26 Port 9 (P91)

(3) P92 (SI1/SCL1)

P92 can be programmed to function as a general-purpose I/O pin, an SI1 data input pin for the serial bus interface in SIO mode, or an SCL1 clock input or output pin for the serial bus interface in I²C bus mode. P92 is configurable as an open-drain output and has a programmable pull-up resistor.

When P92 is used as an open-drain output and does not require a pull-up resistor, the pull-up resistor can be disabled by clearing the PUP.PUP92 register bit to 0.

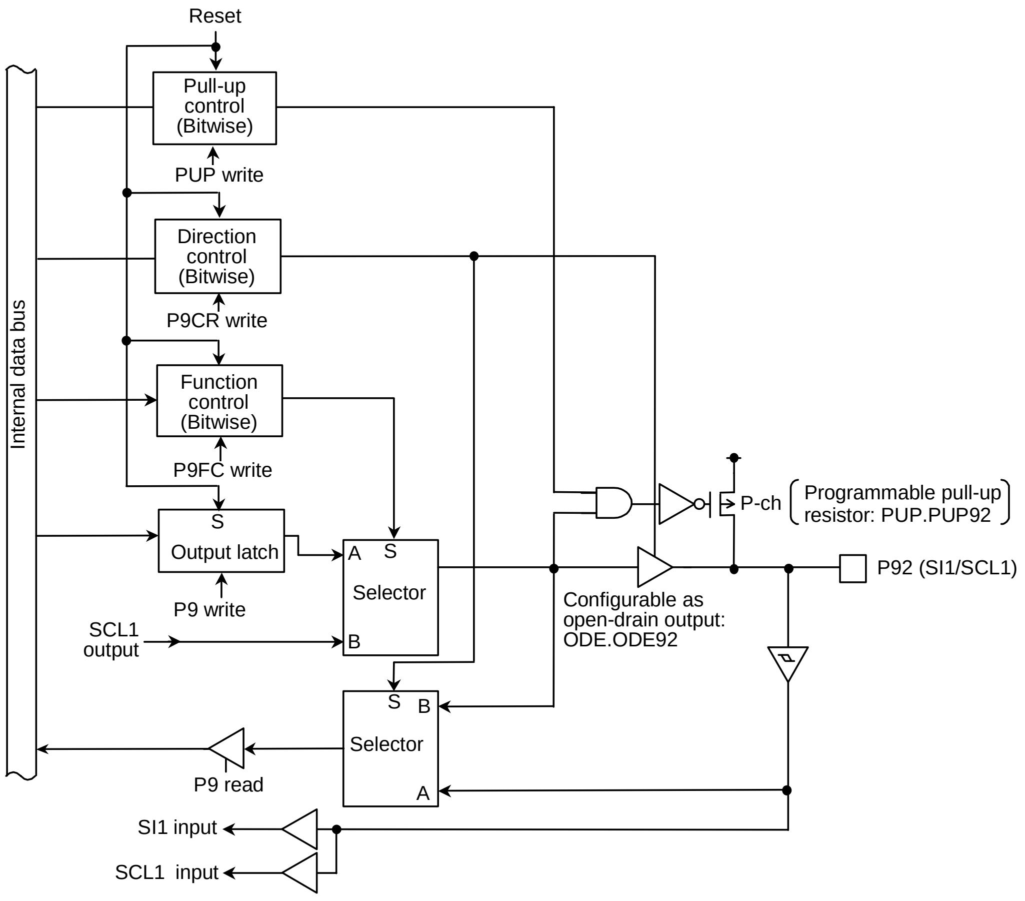

Reset
Pull-up control (Bitwise)
PUP write
Direction control (Bitwise)
P9CR write
Function control (Bitwise)
P9FC write
S
Output latch
P9 write
SCL1 output
A
S
Selector
B
Internal data bus
P-ch
Programmable pull-up resistor: PUP.PUP92
P92 (SI1/SCL1)
Configurable as open-drain output: ODE.ODE92
S
B
Selector
A
P9 read
SI1 input
SCL1 input

Figure 3.5.27 Port 9 (P92)

(4) P93 (TXD)

P93 can be programmed to function as a general-purpose I/O pin or a TXD output pin for the SIO channel. P93 is configurable as an open-drain output.

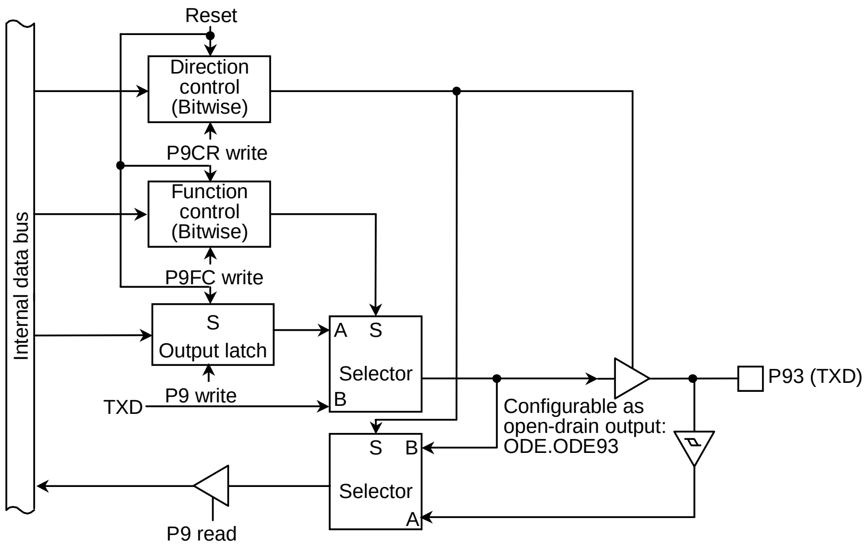


Figure 3.5.28 Port 9 (P93)

(5) P94 (RXD)

P94 can be programmed to function as a general-purpose I/O pin or an RXD input pin for the SIO channel.

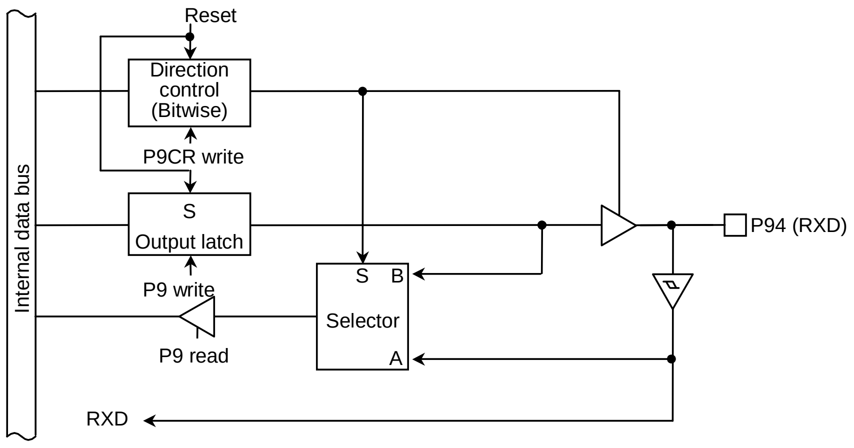


Figure 3.5.29 Port 9 (P94)

(6) P95 ($\overline{\text{CTS}}$ /SCLK)

P95 can be programmed to function as a general-purpose I/O pin, or an SCLK clock input/output pin or $\overline{\text{CTS}}$ input pin for the SIO channel.

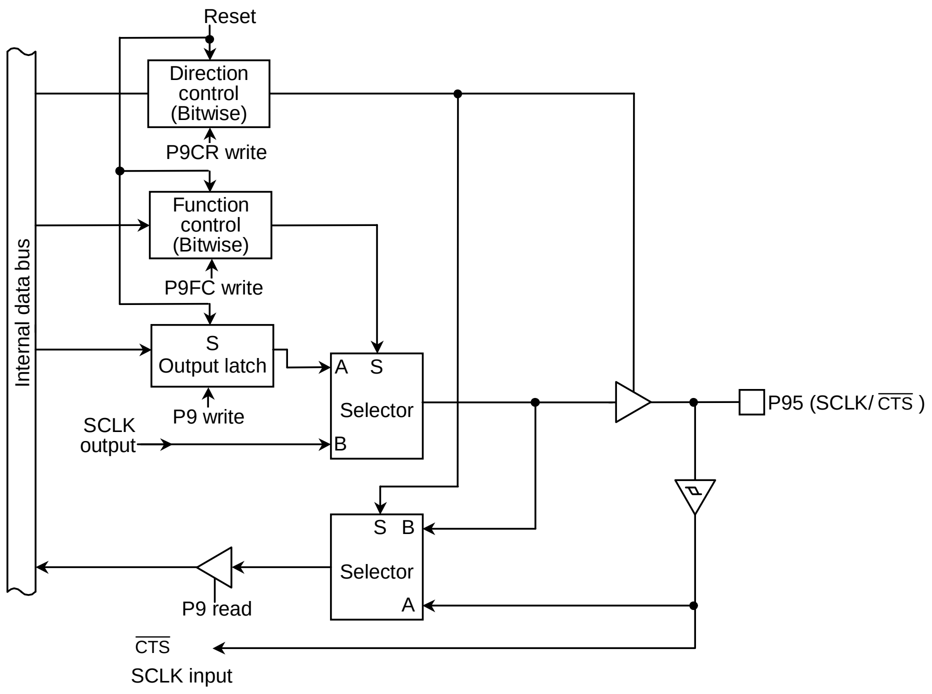


Figure 3.5.30 Port 9 (P95)

(7) P96

P96 functions as a general-purpose I/O pin.

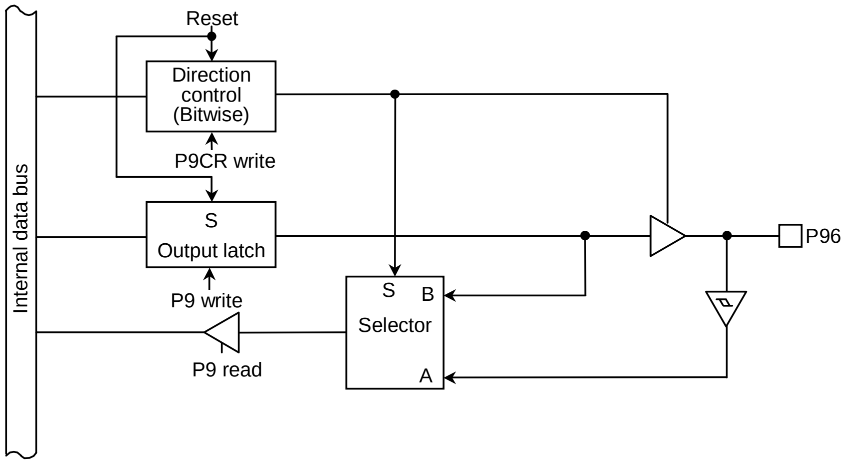


Figure 3.5.31 Port 9 (P96)

Port 9 Register

P9 (0019H)

	7	6	5	4	3	2	1	0
Bit symbol		P96	P95	P94	P93	P92	P91	P90
Read/Write		R/W						
Reset value		Data from external port (Output latch register is set to 1)						
Function		–				0: Pull-up resistor disabled 1: Pull-up resistor enabled		–

Port 9 Control Register

P9CR (001CH)

	7	6	5	4	3	2	1	0
Bit symbol		P96C	P95C	P94C	P93C	P92C	P91C	P90C
Read/Write		W						
Reset value		0	0	0	0	0	0	0
Function		0: Input 1: Output						

Port 9 direction settings

0	Input port
1	Output port

Port 9 Function Register

P9FC (001DH)

	7	6	5	4	3	2	1	0
Bit symbol			P95F		P93F	P92F	P91F	P90F
Read/Write			W		W	W	W	W
Reset value			0		0	0	0	0
Function			0: Port 1: SCLK output		0: Port 1: TXD	0: Port 1: SCL1 output	0: Port 1: SDA1/SO1 output	0: Port 1: SCK1 output

Note 1: P9CR and P9FC do not support read-modify-write operation.

Note 2: When a port 9 pin is used in input mode, the P9 register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port 9 is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Note 3: To specify the TXD pin as an open-drain output, write a 1 to bit1 (for the TXD pin) of the ODE register. The P94/RXD pin does not have a register bit for selecting the port or timer function. The input to the P94/RXD pin is always directed to the SIO as serial receive data even when the pin is used as a general-purpose input pin.

SCK1 output settings for P90

P9FC.P90F	1
P9CR.P90C	1

SDA1/SO1 output settings for P91

P9FC.P91F	1
P9CR.P91C	1

SCL1 output settings for P92

P9FC.P92F	1
P9CR.P92C	1

TXD output settings for P93

P9FC.P93F	1
P9CR.P93C	1

SCLK output settings for P95

P9FC.P95F	1
P9CR.P95C	1

Figure 3.5.32 Port 9 Registers

3.5.11 Port A (PA0 to PA7)

Eight port A pins can be individually programmed to function as discrete general-purpose or dedicated I/O pins. A reset clears all the PACR register bits, configuring all port A pins as input port pins. Alternatively, PA0 to PA3 can be programmed as external interrupt request pins (INT1 to INT4). PA0 to PA3 have programmable pull-up resistors.

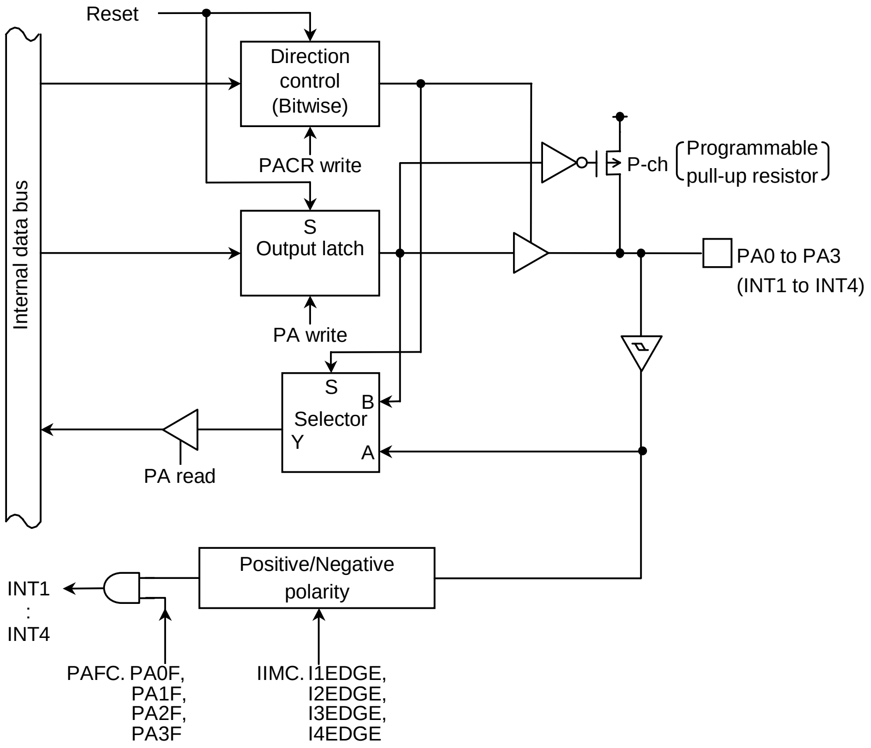

Reset
Internal data bus
Direction control (Bitwise)
PACR write
Output latch
S
PA write
Selector
S
B
Y
A
PA read
P-ch
Programmable pull-up resistor
PA0 to PA3 (INT1 to INT4)
Positive/Negative polarity
INT1 : INT4
PAFC. PA0F, PA1F, PA2F, PA3F
IIMC. I1EDGE, I2EDGE, I3EDGE, I4EDGE

Figure 3.5.33 Port A (PA0 to PA3)

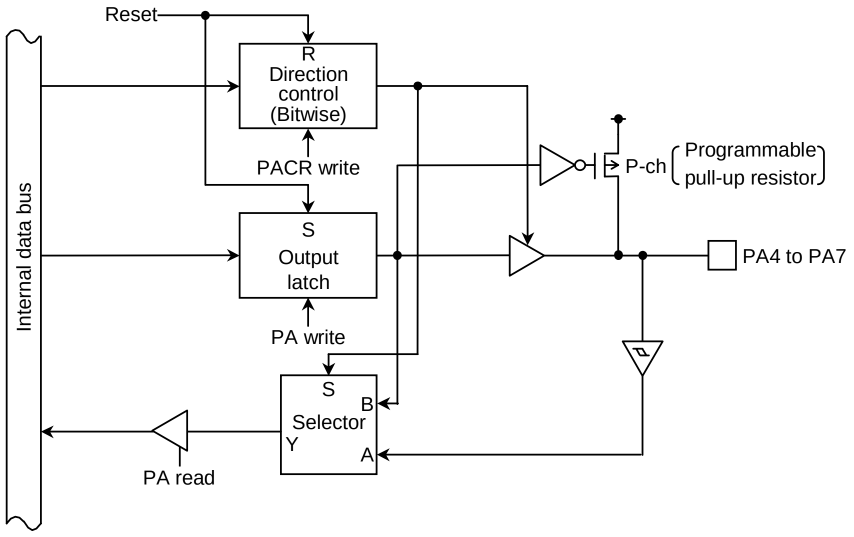


Figure 3.5.34 Port A (PA4 to PA7)

Port A Register

		7	6	5	4	3	2	1	0
PA (001EH)	Bit symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
	Read/Write	R/W							
	Reset value	Data from external port (Output latch register is set to 1)							
	Function	0: Pull-up resistor disabled 1: Pull-up resistor enabled							

Port A Control Register

		7	6	5	4	3	2	1	0
PACR (0020H)	Bit symbol	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

Port A Function Register

		7	6	5	4	3	2	1	0
PAFC (0021H)	Bit symbol					PA3F	PA2F	PA2F	PA0F
	Read/Write					W	W	W	W
	Reset value					0	0	0	0
	Function					0: Port 1: INT4 input	0: Port 1: INT3 input	0: Port 1: INT2 input	0: Port 1: INT1 input

INT1 input settings for PA0

PAFC.PA0F	1
PACR.PA0C	0

INT2 input settings for PA1

PAFC.PA1F	1
PACR.PA1C	0

INT3 input settings for PA2

PAFC.PA2F	1
PACR.PA2C	0

INT4 input settings for PA3

PAFC.PA3F	1
PACR.PA3C	0

Note 1: PACR and PAFC do not support read-modify-write operation.

Note 2: When a port A pin is used in input mode, the PA register controls the on-chip pull-up resistor. A read-modify-write instruction cannot be executed if at least one pin of port A is placed in input mode. A read-modify-write instruction may modify the on-chip pull-up setting for an input pin depending on the current pin status.

Figure 3.5.35 Port A Registers

3.5.12 Input Pull-up Resistor and Open-drain Output Control

The SO0/SDA0 (P61) and SO1/SDA1 (P91) data transmit or data transmit/receive pins of the serial bus interface and the SI0/SCL0 (P62) and SI1/SCL1 (P92) data receive or clock input/output pins of the serial bus interface can be configured as inputs with pull-up resistors enabled.

Pull-up Enable Register

		7	6	5	4	3	2	1	0
PUP (002EH)	Bit symbol			PUP92	PUP91	PUP62	PUP61		
	Read/Write			R/W					
	Reset value			1	1	1	1		
	Function			0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable		

The TXD (P93) output pin of the SIO, the SO0/SDA0 (P61) and SO1/SDA1 (P91) data transmit or data transmit/receive pins of the serial bus interface, and the SI0/SCL0 (P62) and SI1/SCL1 (P92) data receive or clock input/output pins of the serial bus interface can be configured as open-drain outputs.

Serial Open-drain Enable Register

		7	6	5	4	3	2	1	0
ODE (002FH)	Bit symbol			ODE92	ODE91	ODE62	ODE61	ODE93	
	Read/Write			R/W					
	Reset value			0	0	0	0	0	
	Function			1: P92ODE	1: P91ODE	1: P62ODE	1: P61ODE	1: P93ODE	

3.6 Chip Select/Wait Controller

The TMP91CW28 provides four programmable chip select signals. Programmable features include variable block sizes, data bus width, and wait state insertion.

$\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ (Multiplexed with P40 to P43) are the chip select output pins for the CS0 to CS3 address ranges. These chip select signals are generated when the CPU issues an address within the programmed ranges. The P40 to P43 pins must be configured as CS0 to CS3 by programming the port 4 control (P4CR) register and the port 4 function (P4FC) register.

The TMP91CW28 supports direct connections to ROM and SRAM devices.

Chip select address ranges are defined in terms of a memory start address and an address mask. There is a memory start address (MSAR0 to MSAR3) register and memory address mask (MAMR0 to MAMR3) register for each of the four chip select signals.

There is also a set of chip select/wait control registers, B0CS to B3CS and BEXCS, each of which consists of a master enable bit, a data bus width bit, and a wait state field.

External memory devices can also use the $\overline{\text{WAIT}}$ pin to insert wait states and consequently prolong read and write bus cycles.

3.6.1 Programming Chip Select Ranges

Each of the four chip select address ranges is defined in the memory start address (MSAR0 to MSAR3) register and memory address mask (MAMR0 to MAMR3) register. The basic chip select model allows one of the chip select output signals ($\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) to assert when an address on the address bus falls within a particular programmed range. The B0CS to B3CS registers define specific operations for $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, respectively (See section 3.6.2).

(1) Memory start address register

Figure 3.6.1 shows the organization of a memory start address register. The memory start address register (MSAR0 to MSAR3) specifies the start address for a chip select. The S[23:16] bits specify the upper 8 bits (A23 to A16) of the start address. The lower 16 bits (A15 to A0) are assumed to be 0. Thus, the start address is any multiple of 64 Kbytes starting at 000000H. Figure 3.6.2 shows the relationships between start addresses and the contents of the memory start address register.

Memory Start Address Register (for CS0 to CS3)

MSAR0 (00C8H) / MSAR1 (00CAH)
MSAR2 (00CCH) / MSAR3 (00CEH)

	7	6	5	4	3	2	1	0
Bit symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
Reset value	1	1	1	1	1	1	1	1
Function	A23 to A16 of the start address							

→ Start address settings for CS0 to CS3 address ranges

Figure 3.6.1 Memory Start Address Register

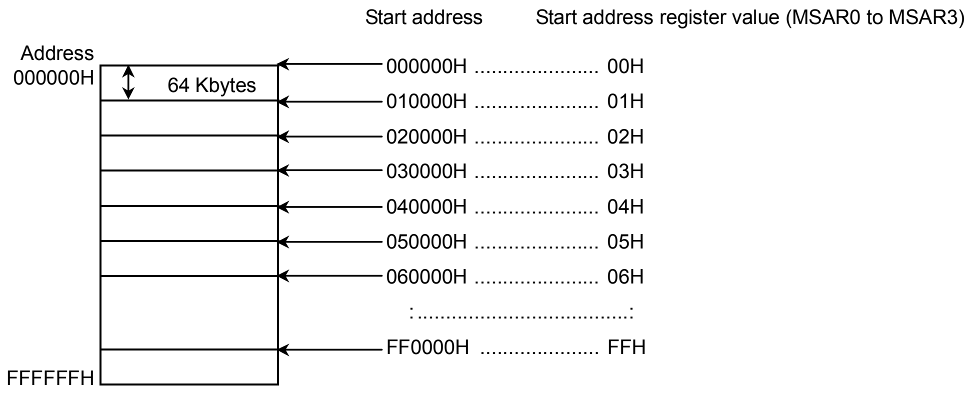

Start address
Start address register value (MSAR0 to MSAR3)
Address 000000H
64 Kbytes
000000H 00H
010000H 01H
020000H 02H
030000H 03H
040000H 04H
050000H 05H
060000H 06H
:.....................................:
FF0000H .... FFH
FFFFFFH

Figure 3.6.2 Relationships between Start Addresses and Start Address Register Values

(2) Memory address mask register

Figure 3.6.3 shows the memory address mask register. The memory address mask register (MAMR0 to MAMR3) controls the size of a chip select address range (CS0 to CS3) by specifying a mask for each bit of the start address specified with the memory start address register (MSAR0 to MSAR3). Any set bit masks the corresponding start address bit. The address compare logic uses only the address bits that are not masked (e.g., mask bit cleared to 0) to detect an address match.

Address bits that can be masked (e.g., supported block sizes) differ for the four chip select spaces.

Memory Address Mask Register (for CS0)

MAMR0 (00C9H)

	7	6	5	4	3	2	1	0
Bit symbol	V20	V19	V18	V17	V16	V15	V14 to V9	V8
Read/Write	R/W							
Reset value	1	1	1	1	1	1	1	1
Function	CS0 block size 0: The address compare logic uses this address bit.							

The CS0 block size can be set in the range from 256 bytes to 2 Mbytes.

Memory Address Mask Register (for CS1)

MAMR1 (00CBH)

	7	6	5	4	3	2	1	0
Bit symbol	V21	V20	V19	V18	V17	V16	V15 to V9	V8
Read/Write	R/W							
Reset value	1	1	1	1	1	1	1	1
Function	CS1 block size 0: The address compare logic uses this address bit.							

The CS1 block size can be set in the range from 256 bytes to 4 Mbytes.

Memory Address Mask Register (for CS2, CS3)

MAMR2 (00CDH) / MAMR3 (00CFH)

	7	6	5	4	3	2	1	0
Bit symbol	V22	V21	V20	V19	V18	V17	V16	V15
Read/Write	R/W							
Reset value	1	1	1	1	1	1	1	1
Function	CS2/CS3 block size 0: The address compare logic uses this address bit.							

The CS2 and CS3 block size can be set in the range from 32 Kbytes to 8 Mbytes.

Figure 3.6.3 Memory Address Mask Register

(3) Memory start address and address mask value calculations

Figure 3.6.4 shows example register settings, causing CS0 to be asserted in the 64 Kbytes of address space starting at 010000H.

The S[23:16] bits in the MSAR0 register specify the upper eight bits of the start address, or 01H. Calculate the difference between the start address and the end address (01FFFFH). Bits 20 to 8 of the result specify a mask value to be used when the CS0 space is specified. Set this value in the V[20:8] bits in the memory address mask register MAMR0 to specify the space size.

This example sets 07H in the MAMR0 to specify 64 Kbytes of address space.

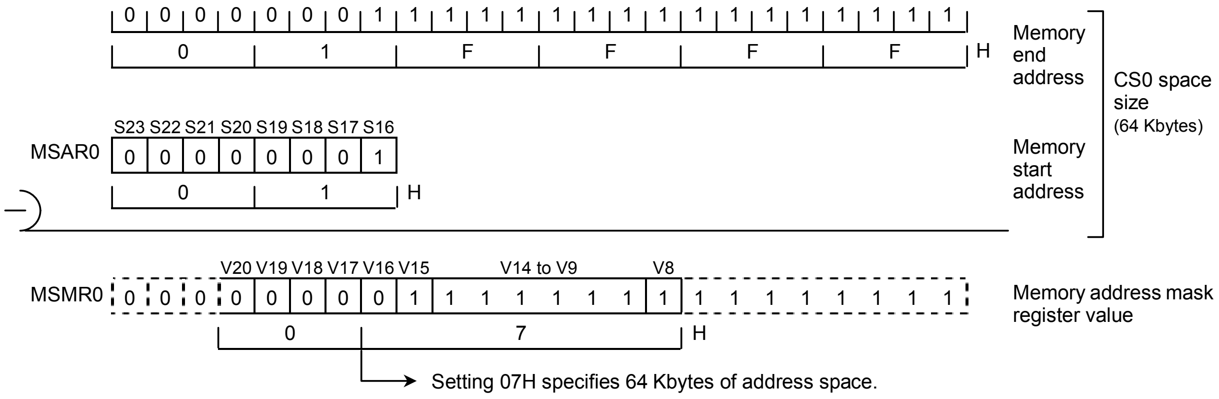


Figure 3.6.4 Example CS0 Space Settings

Upon reset, the MSAR0 to MSAR3 and MAMR0 to MAMR3 are set to FFH while the B0CS.B0E, B1CS.B1E and B3CS.B3E bits are cleared to 0, so that the CS0, CS1 and CS3 spaces are disabled. The TMP91CW28, however, enables the CS2 space in the address range of 003000H through FDFFFFH because B2CS.B2M is cleared to 0 and B2CS.B2E is set to 1. The BEXCS register controls the bus width and wait insertion for addresses other than those included in the CS0 to CS3 spaces. See section 3.6.2.

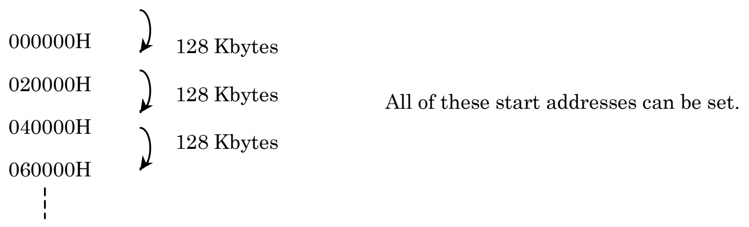
(4) Specifying an address space size

Table 3.6.1 shows the programmable block sizes for CS0 to CS3. "Δ" indicates a combination which may not be possible depending on the memory start address and address mask register values. When using a size marked "Δ", specify start addresses using desired increments from 000000H.

Even if the user has accidentally programmed more than one chip select line to the same area, or if the CS2 space is specified to be 16 Mbytes, only one chip select line is driven because of internal line priorities. CS0 has the highest priority, and CS3 the lowest.

Example: Specifying 128 Kbytes of CS0 space.

a Possible start addresses



b. Impossible start addresses

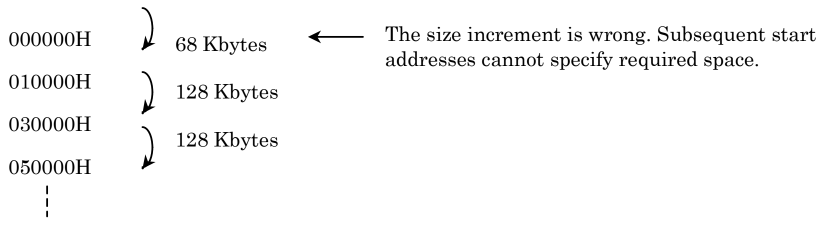

000000H
010000H
030000H
050000H
68 Kbytes
128 Kbytes
128 Kbytes
The size increment is wrong. Subsequent start addresses cannot specify required space.

Table 3.6.1 Supported Block Sizes

Size (bytes) / CS Space	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	Δ	Δ	Δ	Δ	Δ		
CS1	○	○		○	Δ	Δ	Δ	Δ	Δ	Δ	
CS2			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ
CS3			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ

Note: "Δ" indicates a combination which may not be possible depending on the memory start address and address mask register values.

3.6.2 Chip Select/Wait Control Registers

The organizations of the chip select/wait control registers are shown in Figure 3.6.5. Each of these registers consists of a chip select type field, a master enable bit, a data bus width bit, and a wait state field. The B0CS to B3CS registers define the CS0 to CS3 lines, respectively. The BEXCS register defines the access characteristics for the rest of the address locations.

Chip Select/Wait Control Registers

		7	6	5	4	3	2	1	0
B0CS (00C0H)	Bit symbol	B0E		B0OM1	B0OM0	B0BUS	B0W2	B0W1	B0W0
	Read/Write	W		W					
	Reset value	0		0	0	0	0	0	0
Read-modify-write operation is not supported.	Function	0: Disable 1: Enable		Chip select output waveform 00: ROM/SRAM 01: 10: 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of wait-state cycles 000: 2 wait states 001: 1 wait state 010: (1 + N) wait states 011: No wait state 100 101 110 111 Setting prohibited		
B1CS (00C1H)	Bit symbol	B1E		B1OM1	B1OM0	B1BUS	B1W2	B1W1	B1W0
	Read/Write	W		W					
	Reset value	0		0	0	0	0	0	0
Read-modify-write operation is not supported.	Function	0: Disable 1: Enable		Chip select output waveform 00: ROM/SRAM 01: 10: 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of wait-state cycles 000: 2 wait states 001: 1 wait state 010: (1 + N) wait states 011: No wait state 100 101 110 111 Setting prohibited		
B2CS (00C2H)	Bit symbol	B2E	B2M	B2OM1	B2OM0	B2BUS	B2W2	B2W1	B2W0
	Read/Write	W							
	Reset value	1	0	0	0	0	0	0	0
Read-modify-write operation is not supported.	Function	0: Disable 1: Enable	CS2 space select 0: Whole 16-Mbyte space 1: CS space	Chip select output waveform 00: ROM/SRAM 01: 10: 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of wait-state cycles 000: 2 wait states 001: 1 wait state 010: (1 + N) wait states 011: No wait state 100 101 110 111 Setting prohibited		
B3CS (00C3H)	Bit symbol	B3E		B3OM1	B3OM0	B3BUS	B3W2	B3W1	B3W0
	Read/Write	W		W					
	Reset value	0		0	0	0	0	0	0
Read-modify-write operation is not supported.	Function	0: Disable 1: Enable		Chip select output waveform 00: ROM/SRAM 01: 10: 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of wait-state cycles 000: 2 wait states 001: 1 wait state 010: (1 + N) wait states 011: No wait state 100 101 110 111 Setting prohibited		
BEXCS (00C7H)	Bit symbol					BEXBUS	BEXW2	BEXW1	BEXW0
	Read/Write					W			
	Reset value					0	0	0	0
Read-modify-write operation is not supported.	Function					Data bus width 0: 16 bits 1: 8 bits	Number of wait-state cycles 000: 2 wait states 001: 1 wait state 010: (1 + N) wait states 011: No wait state 100 101 110 111 Setting prohibited		

Master enable bit

0	CS space disabled
1	CS space enabled

CS2 space select

Chip select output waveform

00	ROM/SRAM
01	Don't care
10	
11	

Wait state settings
(See "Wait control" in section 3.6.2 (3).)

Data bus width settings

0	16-bit data bus
1	8-bit data bus

Figure 3.6.5 Chip Select/Wait Control Registers

(1) Master enable bit

Bit7 (B0E, B1E, B2E and B3E) in each chip select/wait control register is a master enable bit, which enables or disables the settings in the register for the corresponding address space. Writing a 1 to the bit enables the settings. A reset results in B0E, B1E and B3E being cleared to 0 and B2E being set to 1, so that only the register settings for the CS2 space are enabled.

(2) Data bus width

The TMP91CW28 supports dynamic bus sizing, or modifying the data bus width according to the address space it accesses. Bit3 (B0BUS, B1BUS, B2BUS, B3BUS and BEXBUS) in each chip select/wait control register specifies the data bus width. Writing a 0 to the bit causes the TMP91CW28 to access the corresponding memory space using a 16-bit data bus. Writing a 1 to the bit causes the TMP91CW28 to use an 8-bit data bus. Table 3.6.2 shows details of dynamic bus sizing.

Table 3.6.2 Dynamic Bus Sizing

Data Bus Width for Operand	Operand Start Address	Data Bus Width on Memory	CPU Address	CPU Data	
				D15 to D8	D7 to D0
8 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
		16 bits	2n + 0	xxxxx	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
		16 bits	2n + 1	b7 to b0	xxxxx
16 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	xxxxx	b15 to b8
32 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			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 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			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: For a read, input data on the bus is ignored. For a write, the bus is placed in the high-impedance state and the write strobe for the bus remains inactive.

(3) Wait control

Bits 2 to 0 (B0W[2:0], B1W[2:0], B2W[2:0], B3W[2:0] and BEXW[2:0]) in each chip select/wait control register specifies the number of wait states to be inserted. The following table shows how wait states are inserted according to the combination of these bits. Any combinations other than those listed in the table cannot be used.

Table 3.6.3 Wait Settings

<BxW2:0>	Number of Waits	Operation
000	2	Two wait states are inserted regardless of the state of the $\overline{\text{WAIT}}$ pin.
001	1	One wait state is inserted regardless of the state of the $\overline{\text{WAIT}}$ pin.
010	(1 + N)	One wait state is inserted, after which the state of the $\overline{\text{WAIT}}$ pin is sampled and, if it is low, another wait state is inserted so that the bus cycle is elongated until the pin goes high.
011	0	The bus cycle is completed without wait states inserted, regardless of the state of the $\overline{\text{WAIT}}$ pin.

A reset clears the bits to 000 (2 waits).

(4) Bus width and wait control for addresses outside the CS0 to CS3 spaces

The BEXCS register controls the data bus width and wait states when an address that does not belong to any of the CS0 to CS3 blocks is accessed. The settings in this register are always enabled.

(5) 16-Mbyte space selection

Clearing the B2M bit in the B2CS register to 0 results in 16 Mbytes of address space (003000H to FDFFFFH) being assigned to CS2. Setting the bit to 1 results in the CS2 space being assigned in the same way as CS0, CS1 and CS3, according to the settings in the MSAR2 and MAMR2 registers. A reset clears the bit to 0 so that 16-Mbyte space is assigned.

(6) Procedure for setting the chip select/wait controller

When using the chip select/wait controller, set the following registers in the stated order:

1. Memory start address registers: MSAR0 to MSAR3

   Set the start addresses of the CS0 to CS3 spaces.
2. Memory address mask registers: MAMR0 to MAMR3

   Set the sizes of the CS0 to CS3 spaces.
3. Control registers: B0CS to B3CS

   Set the chip select type, data bus width, number of wait states and master enable/disable for the CS0 to CS3 spaces.

The $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ pins are shared with the P40 to P43 pins. To drive a chip select signal from these pins, the appropriate bits in the port 4 control register (P4CR) and port 4 function register (P4FC) must be set to 1.

If an address specified for any of the CS0 to CS3 spaces falls in the on-chip peripheral, RAM or ROM area, the corresponding CS pin does not drive a chip select signal and the CPU accesses the internal area.

Example:

When the CS0 space is assigned a 64-Kbyte space of 010000H through 01FFFFH with a 16-bit data bus and no wait states:

MSAR0 = 01H	Start address 010000H
MAMR0 = 07H	64-Kbyte address space
B0CS = 83H	ROM/SRAM, 16-bit data bus, 0-wait states, CS0 settings enabled

3.6.3 Application Example

Figure 3.6.6 shows an example usage of the TMP91CW28 programmable chip selects. In this example, an external ROM chip is connected through a 16-bit data bus and external RAM and peripheral chips are connected through an 8-bit data bus.

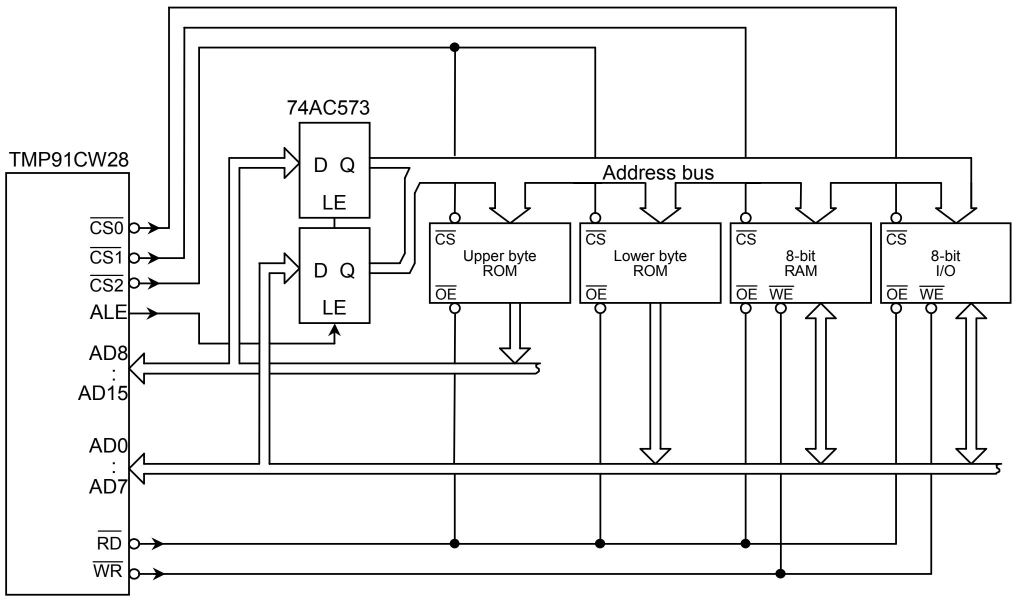


Figure 3.6.6 External Memory Connections (ROM width = 16 bits, RAM and peripheral width = 8 bits)

Both CS1 and CS2 are shared with port 4 pins. Upon reset, all port 4 pins are configured as input port pins. To use them as chip select pins, set appropriate bits in the port 4 function (P4FC) register and the port 4 control (P4CR) register to 1, in this order.

3.7 8-Bit Timers (TMRAs)

The TMP91CW28 has a four-channel 8-bit timer (TMRA0 to TMRA3), which is comprised of two modules named TMRA01 and TMRA23. The TMRA01 contains the TMRA0 and the TMRA1, and the TMRA23 contains the TMRA2 and TMRA3. Each timer module has the following operating modes:

- 8-bit interval timer mode
- 16-bit interval timer mode
- 8-bit programmable pulse generation (PPG) mode (Variable frequency, variable duty cycle)
- 8-bit pulse width modulated (PWM) signal generation mode (Fixed frequency, variable duty cycle)

Figure 3.7.1 and Figure 3.7.2 are block diagrams of the TMRA01 and TMRA23 respectively. The main components of a timer channel are an 8-bit up counter, an 8-bit comparator and an 8-bit timer register. Two timer channels share a prescaler and a timer flip-flop.

A total of five special function registers (SFRs) provide control over the operating modes and timer flip-flops for the TMRA01 and the TMRA23 each, which can be independently programmed. The TMRA01 and the TMRA23 are functionally equivalent. In the following sections, any references to the TMRA01 also apply to the TMRA23.

Table 3.7.1 gives the pins and registers for the two timer modules.

Table 3.7.1 Pins and Registers for the TMRA01 and the TMRA23

Specifications \ Module		TMRA01	TMRA23
External pins	External clock input	TA0IN (Shared with P70)	None
	Timer flip-flop output	TA1OUT (Shared with P71)	TA3OUT (Shared with P72)
Registers (Addresses)	Timer run register	TA01RUN (0100H)	TA23RUN (0108H)
	Timer registers	TA0REG (0102H) TA1REG (0103H)	TA2REG (010AH) TA3REG (010BH)
	Timer mode register	TA01MOD (0104H)	TA23MOD (010CH)
	Timer flip-flop control register	TA1FFCR (0105H)	TA3FFCR (010DH)

3.7.1 Block Diagrams

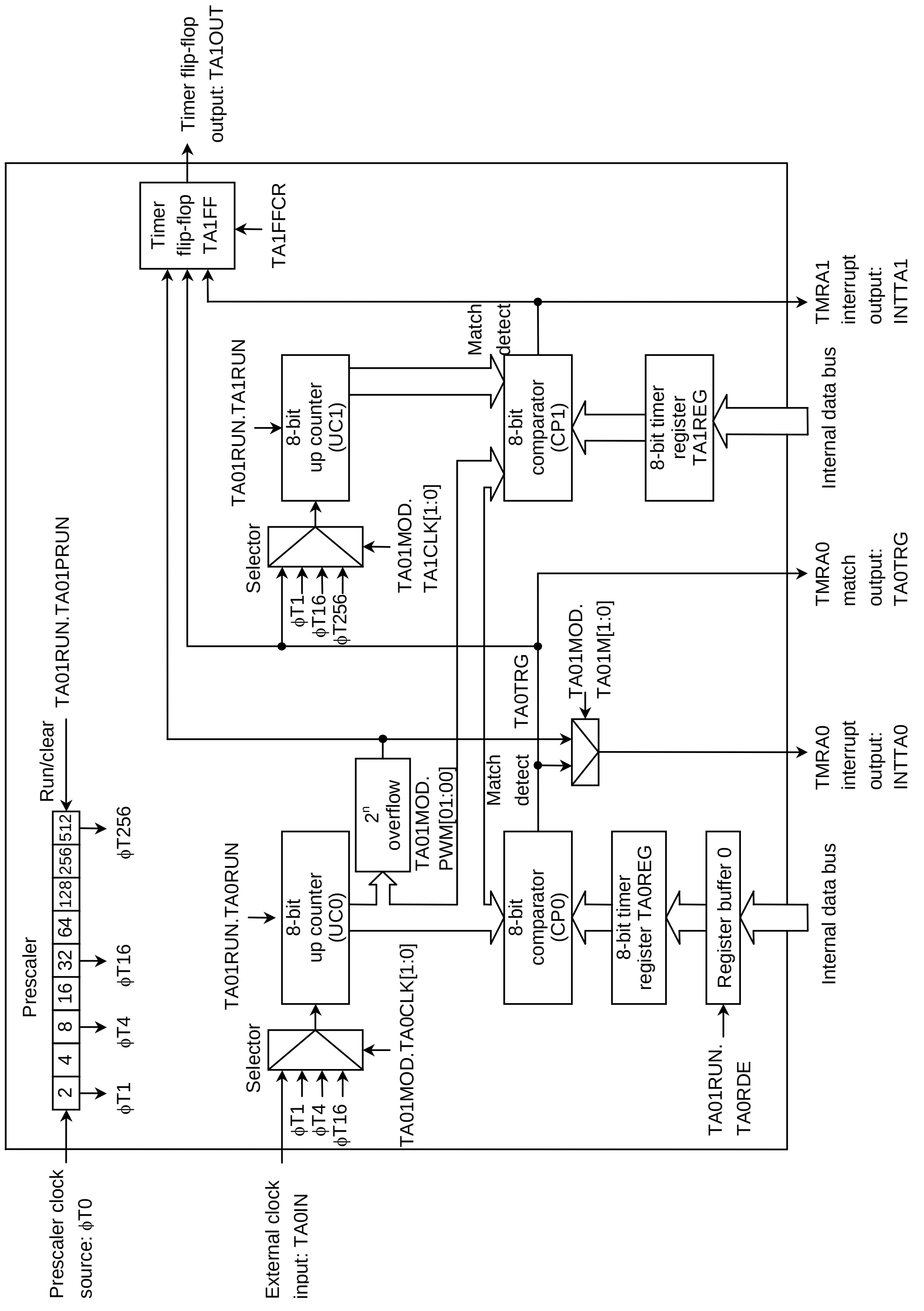


Figure 3.7.1 TMRA01 Block Diagram

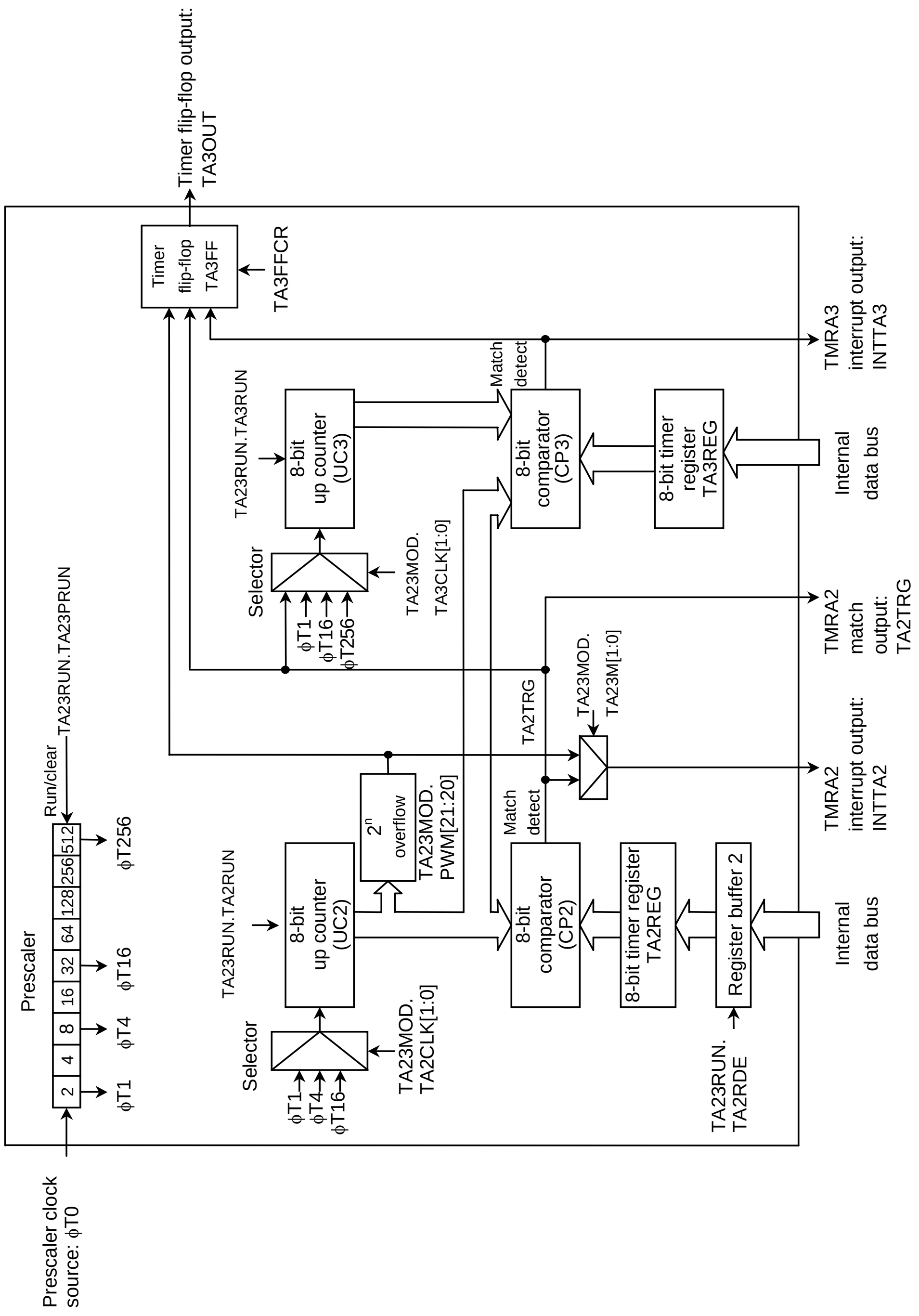


Figure 3.7.2 TMRA23 Block Diagram

3.7.2 Timer Components

(1) Prescaler

The TMRA01 has a 9-bit prescaler that slows the rate of a clocking source to the counters. The prescaler clock source (ϕT0) has one-fourth the frequency selected by programming the PRCK[1:0] field of the SYSCR0 located within the clock gear.

The TA0PRUN bit in the TA01RUN register allows the enabling and disabling of the prescaler for the TMRA01. A write of 1 to this bit starts the prescaler. A write of 0 to this bit clears and halts the prescaler. Table 3.7.2 shows prescaler output clock resolutions.

Table 3.7.2 Prescaler Output Clock Resolutions

at fc = 10 MHz

Prescaler Clock Source PRCK[1:0]	Clock Gear Value GEAR[2:0]	Prescaler Output Clock Resolution			
		ϕT1	ϕT4	ϕT16	ϕT256
00 (f_{FPH})	000 (fc)	fc/2^3 (0.8 μs)	fc/2^5 (3.2 μs)	fc/2^7 (12.8 μs)	fc/2^{11} (204.8 μs)
	001 (fc/2)	fc/2^4 (1.6 μs)	fc/2^6 (6.4 μs)	fc/2^8 (25.6 μs)	fc/2^{12} (409.6 μs)
	010 (fc/4)	fc/2^5 (3.2 μs)	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)	fc/2^{13} (819.2 μs)
	011 (fc/8)	fc/2^6 (6.4 μs)	fc/2^8 (25.6 μs)	fc/2^{10}(102.4 μs)	fc/2^{14} (1638.4 μs)
	100 (fc/16)	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)	fc/2^{11}(204.8 μs)	fc/2^{15} (3276.8 μs)
10 (fc/16 clock)	XXX	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)	fc/2^{11}(204.8 μs)	fc/2^{15} (3276.8 μs)

XXX: Don't care

(2) Up counters (UC0 and UC1)

The timer module contains two 8-bit binary up counters, each of which is driven by a clock independently selected by the TA01MOD register.

The clock input to the UC0 is either one of three prescaler outputs (ϕT1, ϕT4, ϕT16) or the external clock applied to the TA0IN pin. Which clock is to use is programmed into the TA0CLK[1:0] field in the TA01MOD register.

Possible clock sources for the UC1 depend on the selected operating mode. In 16-bit interval timer mode, the clock input to the UC1 is always the UC0 overflow output. In other operating modes, the clock input to the UC1 is either one of three prescaler outputs (ϕT1, ϕT16, ϕT256) or the TMRA0 comparator match-detect output.

The TA0RUN and TA1RUN bits in the TA01RUN register are used to start counting and to stop and clear the counter. Upon reset, the up counter is cleared to 00H and the whole timer module is disabled.

(3) Timer registers (TA0REG and TA1REG)

Each timer register is an 8-bit register containing a time constant. When the up counter reaches the time constant value in the timer register, the comparator block generates a match-detect signal. When the time constant is cleared to 00H, a match occurs upon a counter overflow.

One of the two timer registers, TA0REG, is double buffered. The double-buffering function can be enabled and disabled through the programming of the TA0RDE bit in the TA01RUN: 0 = disable, 1 = enable.

If double-buffering is enabled, the TA0REG latches a new time constant value from the register buffer. This takes place upon detection of a 2^n overflow in PWM mode and upon a match between the UC0 and the TA1REG in PPG mode. Double-buffering must be disabled in interval timer modes.

A reset clears the TA01RUN.TA0RDE bit to 0, disabling the double-buffering function. To use this function, the TA01RUN.TA0RDE bit must be cleared after loading the TA0REG with a time constant. When TA01RUN.TA0RDE = 1, the next time constant can be written to the register buffer.

Figure 3.7.3 illustrates the double-buffer structure for the TA0REG.

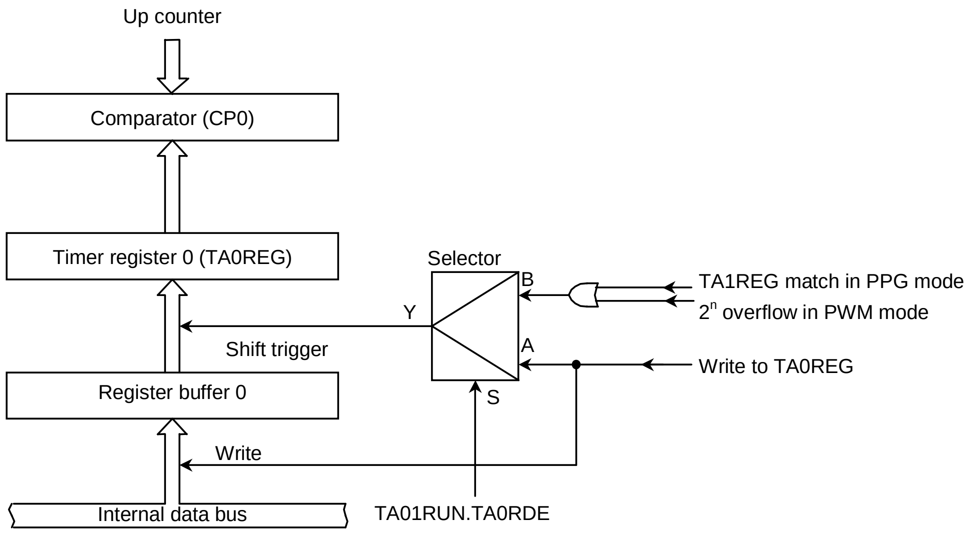


Figure 3.7.3 Timer Register 0 (TA0REG) Structure

Note: The timer register and the corresponding register buffer are mapped to the same address. When TA01RUN.TA0RDE = 0, a time constant value is written to both of the timer register and the register buffer; when TA01RUN.TA0RDE = 1, a time constant value is written only to the register buffer.

The addresses of the timer registers are as follows:

TA0REG: 000102H	TA1REG: 000103H
TA2REG: 00010AH	TA3REG: 00010BH

The timer registers are write-only registers.

(4) Comparators (CP0 and CP1)

The comparator compares the output of the 8-bit up counter with a time constant value in the 8-bit timer register. When a match is detected, an interrupt (INTTA0/INTTA1) is generated and the timer flip-flop is toggled, if so enabled.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is toggled, if so enabled, each time the comparator match-detect output is asserted. The toggling of the timer flip-flop can be enabled and disabled through the programming of the TA1FFIE bit in the TA1FFCR.

A reset clears the TAFF1IE bit, disabling the toggling of the TA1FF. The TA1FF can be initialized to 1 or 0 by writing 01 or 10 to the TA1FFC[1:0] field in the TA1FFCR. Additionally, a write of 00 by software causes the TA1FF to be toggled to the opposite value.

The value of the TA1FF can be driven onto the TA1OUT pin. The port 7 registers (P7CR and P7FC) must be programmed to configure the P71/TA1OUT pin as TA1OUT.

3.7.3 SFR Description

TMRA01 Run Register

TA01RUN (0100H)

	7	6	5	4	3	2	1	0
Bit symbol	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
Read/Write	R/W				R/W			
Reset value	0				0	0	0	0
Function	Double buffering 0: Disable 1: Enable				IDLE2 0: OFF 1: ON	8-bit timer run/stop control 0: Stop and clear 1: Run		

TA0REG double buffering control

0	Disable
1	Enable

Counting

0	Stop and clear
1	Count up

I2TA01:	Timer ON/OFF in IDLE2 mode
TA01PRUN:	Prescaler
TA1RUN:	Timer 1
TA0RUN:	Timer 0

Note: Bits4, 5, and 6 are read as undefined.

TMRA23 Run Register

TA23RUN (0108H)

	7	6	5	4	3	2	1	0
Bit symbol	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
Read/Write	R/W				R/W			
Reset value	0				0	0	0	0
Function	Double buffering 0: Disable 1: Enable				IDLE2 0: OFF 1: ON	8-bit timer run/stop control 0: Stop and clear 1: Run		

TA2REG double buffering control

0	Disable
1	Enable

Counting

0	Stop and clear
1	Count up

I2TA23:	Timer ON/OFF in IDLE2 mode
TA23PRUN:	Prescaler
TA3RUN:	Timer 3
TA2RUN:	Timer 2

Note: Bits4, 5, and 6 are read as undefined.

Figure 3.7.4 TMRA Registers (1)

TMRA01 Mode Register

TA01MOD (0104H)		7	6	5	4	3	2	1	0
	Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
	Read/Write	R/W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM period 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA1 clock source 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA0 clock source 00: TA0IN input 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA0 clock source

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

TMRA1 clock source

	TA01MOD.TA01M[1:0] $\neq$ 01	TA01MOD.TA01M[1:0] = 01
00	TMRA0 match output	TMRA0 overflow output (16-bit timer mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

Period select in 8-bit PWM mode

00	Reserved
01	$2^6 \times$ clock source
10	$2^7 \times$ clock source
11	$2^8 \times$ clock source

TMRA01 operating mode

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM generation (TMRA0) and 8-bit timer (TMRA1)

Figure 3.7.5 TMRA Registers (2)

TMRA23 Mode Register

TA23MOD (010CH)		7	6	5	4	3	2	1	0
	Bit symbol	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
	Read/Write	R/W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit interval timer 01:16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM period 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA3 clock source 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA2 clock source 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA2 clock source

00	Setting prohibited
01	ϕT1 (Prescaler)
10	ϕT4 (Prescaler)
11	ϕT16 (Prescaler)

TMRA3 clock source

	TA23MOD.TA23M[1:0] ≠ 01	TA23MOD.TA23M[1:0] = 01
00	TMRA2 match output	TMRA2 overflow output (16-bit timer mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

Period select in 8-bit PWM mode

00	Reserved
01	$2^6 \times$ clock source
10	$2^7 \times$ clock source
11	$2^8 \times$ clock source

TMRA23 operating mode

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM generation (TMRA2) and 8-bit timer (TMRA3)

Figure 3.7.6 TMRA Registers (3)

TMRA1 Flip-flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR (0105H)	Bit symbol					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
	Read/Write					R/W		R/W	
	Reset value					1	1	0	0
Read-modify-write operation is not supported.	Function					00: Toggle 01: Set 10: Clear 11: Don't care		TA1FF toggle enable 0: Disable 1: Enable	TA1FF toggle trigger 0: TMRA0 1: TMRA1

Selects a signal to toggle timer flip-flop 1 (TA1FF)
(Don't care in other than 8-bit timer mode)

0	Toggled by TMRA0
1	Toggled by TMRA1

TA1FF toggle enable

0	Disable
1	Enable

TA1FF control

00	Toggles TA1FF (Software toggle).
01	Sets TA1FF to 1.
10	Clears TA1FF to 0.
11	Don't care

Note: Bits 4 to 7 are read as undefined.

Figure 3.7.7 TMRA Registers (4)

TMRA3 Flip-flop Control Register

TA3FFCR (010DH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
Read/Write					R/W		R/W	
Reset value					1	1	0	0
Function					00: Toggle 01: Set 10: Clear 11: Don't care		TA3FF toggle enable 0: Disable 1: Enable	TA3FF toggle trigger 0: TMRA2 1: TMRA3

Selects a signal to toggle timer flip-flop 3 (TA3FF)
(Don't care in other than 8-bit timer mode)

0	Toggled by TMRA2
1	Toggled by TMRA3

TA3FF toggle enable

0	Disable
1	Enable

TA3FF control

00	Toggles TA3FF (Software toggle).
01	Sets TA3FF to 1.
10	Clears TA3FF to 0.
11	Don't care

Note: Bits 4 to 7 are read as undefined.

Figure 3.7.8 TMRA Registers (5)

3.7.4 Operating Modes

(1) 8-bit interval timer mode

The TMRA0 and the TMRA1 can be independently programmed as 8-bit interval timers. Programming these timers should only be attempted when the timers are not running.

a. Generating periodic interrupts

In the following example, the TMRA1 is used to accomplish periodic interrupt generation. First, stop the TMRA1 (if it is running). Then, set the operating mode, clock source and interrupt interval in the TA01MOD and TA1REG registers. Then, enable the INTTA1 interrupt and start the TMRA1.

Example: Generating the INTTA1 interrupt at a 20 μs interval (fc = 10 MHz)

Clocking conditions: High-speed clock gear: × 1 (fc); Prescaler clock: f_{FPH}

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	–	X	X	X	–	0	0	–	Stops and clears the TMRA1.
TA01MOD	←	0	0	X	X	0	1	X	X	Selects 8-bit interval timer mode and φ T1 as the clock source (which provides a 0.8 μs resolution at fc = 10 MHz).
TA1REG	←	0	0	0	1	1	0	0	1	Sets the time constant value in the TA1REG (20 μs ÷ φ T1 = 25 (19H)).
INTETA01	←	X	1	0	1	X	–	–	–	Enables INTTA1 and sets the interrupt level to 5.
TA01RUN	←	–	X	X	X	–	1	1	–	Starts the TMRA1.

X: Don't care, –: No change

Refer to Table 3.7.2 when selecting a timer clock source.

Note: The clock inputs to the TMRA0 and the TMRA1 can be one of the following:
TMRA0: TA0IN input, φT1, φT4, or φT16
TMRA1: Match-detect signal from the TMRA0, φT1, φT16, or φT256

b. Generating a square wave with a 50% duty cycle

The 8-bit interval timer mode can be used to generate square-wave output. This is accomplished by toggling the timer flip-flop (TA1FF) periodically. The TA1FF state can be driven out to the TA1OUT pin. Both the TMRA0 and the TMRA1 can be used as square-wave generators. The following shows an example using the TMRA1.

Example: Generating square-wave output with a 4.8 μs period on the TA1OUT pin (fc = 10 MHz).

Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	0	0	–	Stops and clears the TMRA1.
TA01MOD	←	0	0	X	X	0	1	X	X	Selects 8-bit interval timer mode and ϕT1 as the clock source (which provides a 0.8 μs resolution at fc = 10 MHz).
TA1REG	←	0	0	0	0	0	0	1	1	Sets the time constant value in the TA1REG (4.8 μs ÷ ϕT1 ÷ 2 = 03H).
TA1FFCR	←	X	X	X	X	1	0	1	1	Clears the TA1FF to 0 and selects the TMRA1 match-detect output as a toggle-trigger signal.
P7CR	←	X	X	–	–	–	–	1	–	Configures P71 as the TA1OUT output pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	–	X	X	X	–	1	1	–	Starts the TMRA1.

X: Don't care, –: No change

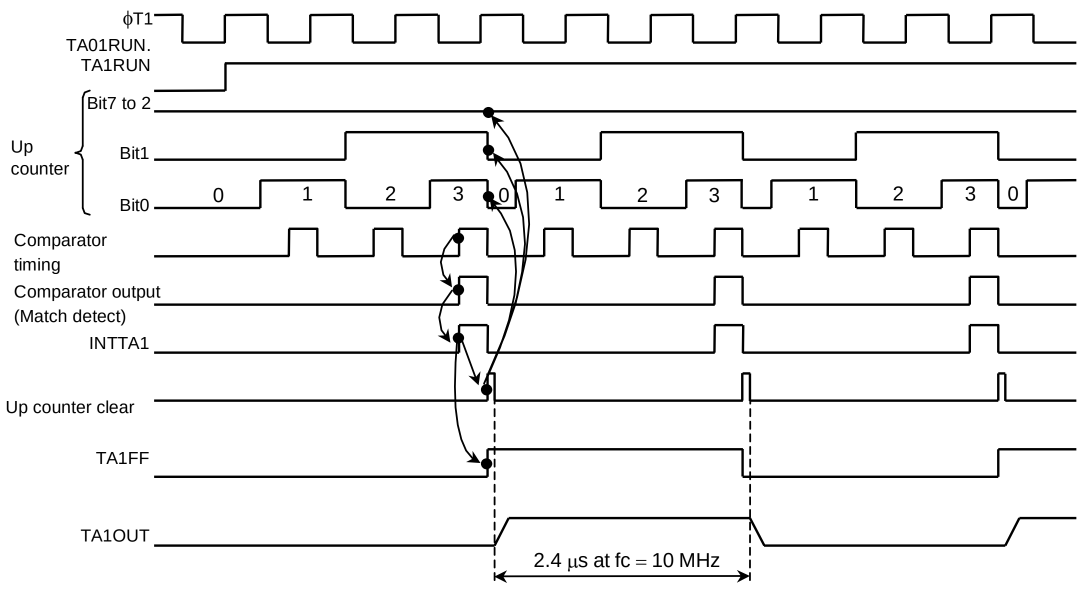


Figure 3.7.9 Square-wave Generation (50% duty cycle)

c. Using the TMRA0 match-detect output as a trigger for the TMRA1

Set the TMRA01 in 8-bit interval timer mode. Select the TMRA0 comparator match-detect output as the clock source for the TMRA1.

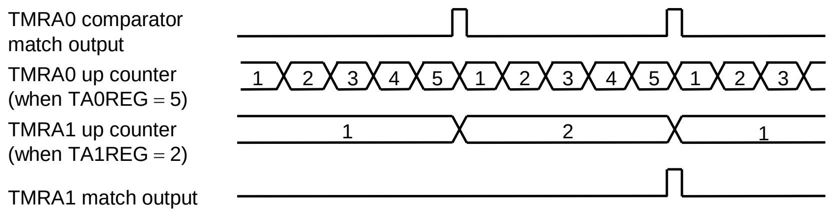

TMRA0 comparator match output
TMRA0 up counter (when TA0REG = 5)
1 2 3 4 5 1 2 3 4 5 1 2 3
TMRA1 up counter (when TA1REG = 2)
1 2 1
TMRA1 match output

Figure 3.7.10 Using the TMRA0 Match-detect Output as a Trigger for the TMRA1

(2) 16-bit interval timer mode

The TMRA0 and the TMRA1 are cascadable to form a 16-bit interval timer. The TMRA01 is put in 16-bit interval timer mode by programming the TA01M[1:0] field in the TA01MOD register to 01.

In 16-bit interval timer mode, the TMRA1 is clocked by the counter overflow output from the TMRA0. In this mode, the TA1CLK[1:0] bits in the TA01MOD register are don’t cares. The clock input to the TMRA0 can be selected as shown in Table 3.7.2.

Write the lower eight bits of a time constant value to the TA0REG and the upper eight bits to the TA1REG. Note that the TA0REG must first be programmed prior to the TA1REG. Writing data to the TA0REG causes comparison to be disabled temporarily, after which writing data to the TA1REG restarts comparison.

Example: Generating the INTTA1 interrupt at a 0.1 second interval (fc = 10 MHz).

Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

Under the above conditions, ϕT8 has a period of 6.4 μs at 10 MHz. When ϕT8 is used as the TMRA0 clock source, the required time constant value is calculated as follows:

0.5 s ÷ 6.4 μs = 15625 = 3D09H

Thus, the TA1REG is to be set to 3DH and the TA0REG to 09H.

Every time the up counter UC0 reaches the value in the TA0REG, the TMRA0 comparator generates a match-detect output, but the UC0 continues counting up. A match between the UC0 and the TA0REG does not cause an INTTA0 interrupt.

Every time the up counter UC1 reaches the value in the TA1REG, the TMRA1 comparator generates a match-detect output. When the TMRA0 and TMRA1 match-detect outputs are asserted simultaneously, both the up counters (UC0 and UC1) are reset to 00H and an interrupt is generated on INTTA1. Also, if so enabled, the timer flip-flop (TA1FF) is toggled.

Example: TA1REG = 04H and TA0REG = 80H

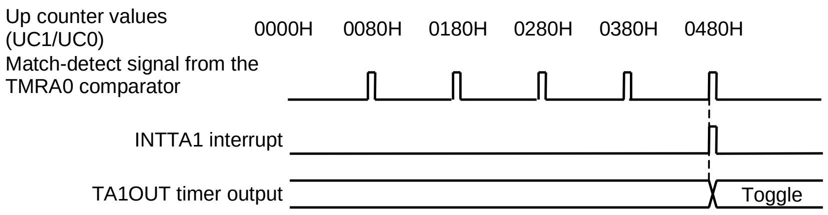
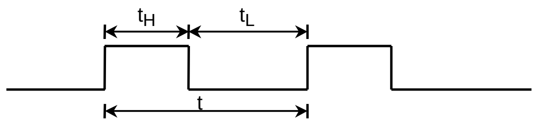

Up counter values (UC1/UC0)
0000H
0080H
0180H
0280H
0380H
0480H
Match-detect signal from the TMRA0 comparator
INTTA1 interrupt
TA1OUT timer output
Toggle

Figure 3.7.11 Timer Output in 16-Bit Interval Timer Mode

(3) 8-bit programmable pulse generation (PPG) mode

The 8-bit PPG mode can be used to generate a square wave with any frequency and duty cycle, as shown below. The pulse can be high-going and low-going, as determined by the initial setting of the timer flip-flop (TA1FF). This mode is supported by the TMRA0, but not by the TMRA1. The square-wave output is driven to the TA1OUT pin.



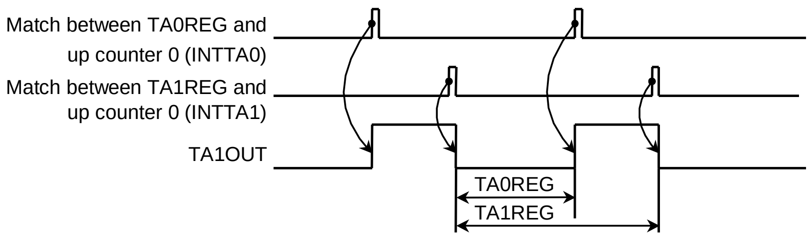

Match between TA0REG and up counter 0 (INTTA0)
Match between TA1REG and up counter 0 (INTTA1)
TA1OUT
TA0REG
TA1REG

Figure 3.7.12 8-Bit PPG Output Waveform

In this mode, a square wave is generated by toggling the timer flip-flop (TA1FF). The TA1FF changes state every time a match is detected between the UC0 and the TA0REG and between the UC0 and the TA1REG.

The TA0REG must be set to a value less than the TA1REG value.

In this mode, the TMRA1 up counter (UC1) cannot be independently used. However, the TMRA1 must be put in a running state by setting the TA1RUN bit in the TA01RUN register to 1.

Figure 3.7.13 shows a functional diagram of 8-bit PPG mode.

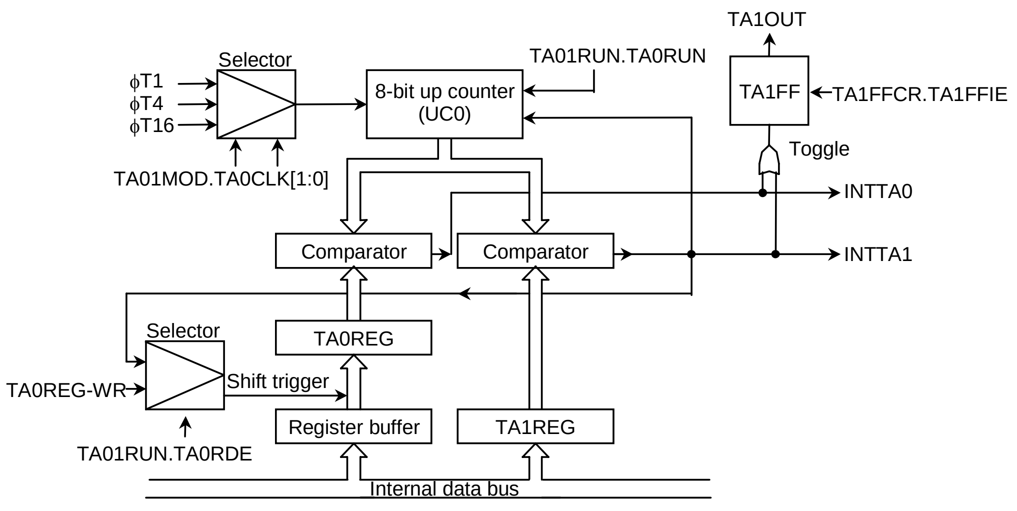


Figure 3.7.13 Functional Diagram of 8-Bit PPG Mode

In 8-bit PPG mode, if the double-buffering function is enabled, the TA0REG value can be changed dynamically by writing a new value into the register buffer. Upon a match between the TA1REG and the UC0, the TA0REG latches a new value from the register buffer.

The TA0REG can be loaded with a new value upon every match, thus making it easy to generate a square wave with virtually any (and variable) duty cycle.

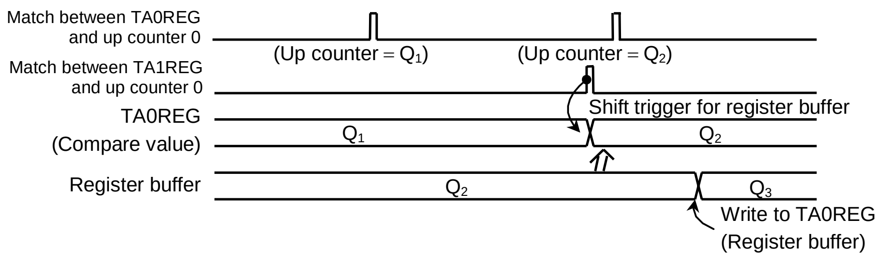
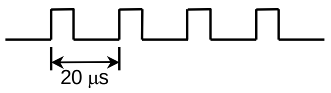

Match between TA0REG and up counter 0
(Up counter = Q1)
(Up counter = Q2)
Match between TA1REG and up counter 0
TA0REG (Compare value)
Shift trigger for register buffer
Q1
Q2
Register buffer
Q2
Q3
Write to TA0REG (Register buffer)

Figure 3.7.14 Register Buffer Operation

Example: Generating a 50 kHz square wave with a 25% duty cycle (fc = 10 MHz)



* Clocking conditions:
- High-speed clock gear: × 1 (fc)
- Prescaler clock: f_{FPH}

The time constant values to be loaded into the TA0REG and TA1REG are determined as follows:

A 50 kHz waveform has a period of 20 μs. Under the above clocking conditions, ϕT1 has a 0.8 μs resolution (at fc = 10 MHz). When ϕT1 is used as the timer clock source, the TA1REG should be loaded with:

32 μs ÷ 0.8 μs = 40

With a 25% duty cycle, the high pulse width is calculated as 20 μs × 1/4 = 5 μs. Thus, the TA0REG should be loaded with:

8 μs ÷ 0.8 μs = 10

		7	6	5	4	3	2	1	0	
TA01RUN	←	0	X	X	X	–	0	0	0	Stops and clears the TMRA0 and the TMRA1.
TA01MOD	←	1	0	X	X	X	X	0	1	Selects 8-bit PPG mode and ϕT1 as the clock source.
TA0REG	←	0	0	0	0	1	0	1	0	Writes 0AH.
TA1REG	←	0	0	1	0	1	0	0	0	Writes 28H.
TA1FFCR	←	X	X	X	X	0	1	1	X	Sets the TA1FF to 1 and enables toggling.
										If these bits are set to 10, a low-going pulse is generated.
P7CR	←	X	X	–	–	–	–	1	–	Configures P71 as the TA1OUT output pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	1	1	Starts the TMRA0 and the TMRA1.

X: Don't care, –: No change

(4) 8-bit PWM generation mode

The TMRA0 can be used as a pulse-width modulated (PWM) signal generator with up to 8 bits of resolution. This mode is supported by the TMRA0, but not by the TMRA1. The PWM signal is driven out on the TA1OUT pin.

While the TMRA01 is in this mode, the TMRA1 is usable as an 8-bit interval timer.

The timer flip-flop toggles when the up counter (UC0) reaches the TA0REG value and when a 2^n counter overflow occurs, where n is programmable to 6, 7, or 8 through the PWM[01:00] field in the TA01MOD register. The UC0 is reset to 00H upon a 2^n overflow.

In 8-bit PWM generation mode, the following must be satisfied:

(TA0REG value) < (2^n counter overflow value)

(TA0REG value) $\neq$ 0

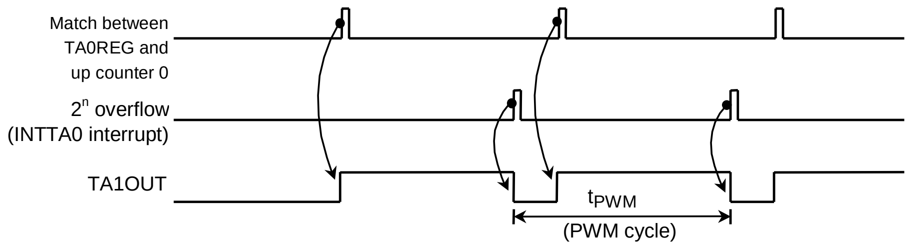

Match between TA0REG and up counter 0
2^n overflow (INTTA0 interrupt)
TA1OUT
tPWM
(PWM cycle)

Figure 3.7.15 8-Bit PWM Signal Generation

Figure 3.7.16 shows a functional diagram of 8-bit PWM generation mode.

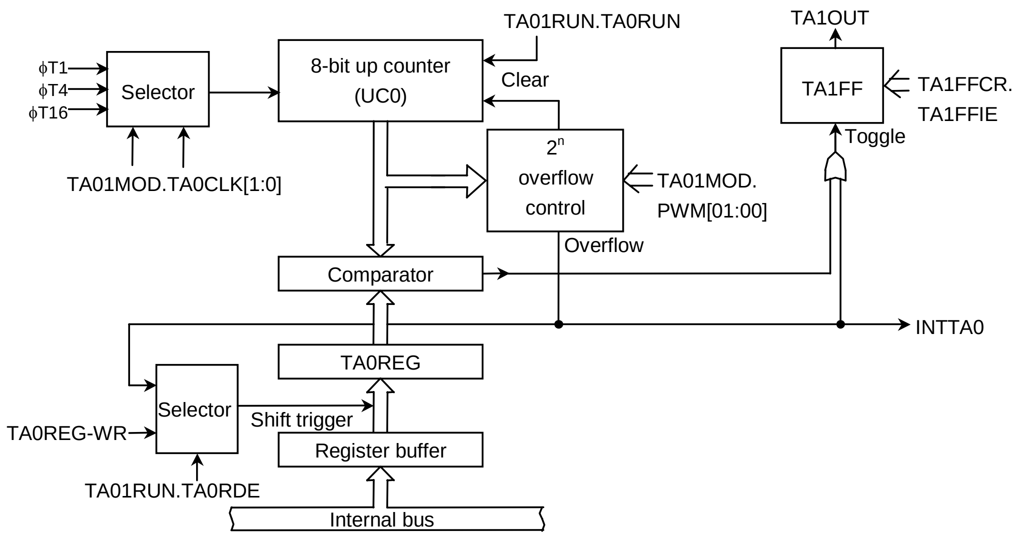


Figure 3.7.16 Functional Diagram of 8-Bit PWM Generation Mode

In 8-bit PWM generation mode, if the double-buffering function is enabled, the TA0REG value (e.g., the duty cycle) can be changed dynamically by writing a new value into the register buffer. Upon a 2^n counter overflow, the TA0REG latches a new value from the register buffer.

The TA0REG can be loaded with a new value upon every counter overflow, thus making it easy to generate a PWM signal with virtually any (and variable) duty cycle.

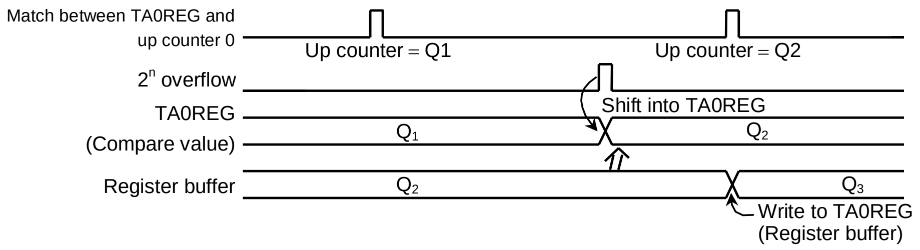
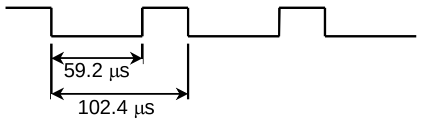


Figure 3.7.17 Register Buffer Operation

Example: Generating a PWM signal as shown below on the TA1OUT pin (fc = 10 MHz).



* Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

Under the above conditions, ϕT1 has a 0.8 μs period (at fc = 10 MHz).

102.4 μs ÷ 0.8 μs = 128

which is equal to $2^7 - 1$.

59.2 μs ÷ 0.8 μs = 74 = 4AH

Hence, the time constant value to be programmed into the TA0REG is 4AH.

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	–	X	X	X	–	0	–	0	Stops and clears the TMRA0.
TA01MOD	←	1	1	1	0	X	X	0	1	Selects 8-bit PWM mode (period = 2^7) and ϕT1 as the clock source.
TA0REG	←	0	1	0	0	1	0	1	0	Writes 4AH.
TA1FFCR	←	X	X	X	X	1	0	1	X	Clears the TA1FF to 0 and enables toggling.
P7CR	←	X	X	–	–	–	–	1	–	Configures P71 as the TA1OUT output pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	–	1	Starts the TMRA0.

X: Don’t care, –: No change

Table 3.7.3 PWM Period

at fc = 10 MHz

Prescaler Clock Source PRCK[1:0]	Clock Gear Value GEAR[2:0]	PWM Period								
		2^6			2^7			2^8		
		ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16
00 (f_{FPH})	000 (fc)	51.2 μs	204.8 μs	819.2 μs	102.4 μs	409.6 μs	1638.4 μs	204.8 μs	819.2 μs	3276.8 μs
	001 (fc/2)	102.4 μs	409.6 μs	1638.4 μs	204.8 μs	819.2 μs	3276.8 μs	409.6 μs	1638.4 μs	6553.6 μs
	010 (fc/4)	204.8 μs	819.2 μs	3276.8 μs	409.6 μs	1638.4 μs	6553.6 μs	819.2 μs	3276.8 μs	13.1072 ms
	011 (fc/8)	409.6 μs	1638.4 μs	6553.6 μs	819.2 μs	3276.8 μs	13.1072 ms	1638.4 μs	6553.6 μs	26.2144 ms
	100 (fc/16)	819.2 μs	3276.8 μs	13.1072 ms	1638.4 μs	6553.6 μs	26.2144 ms	3276.8 μs	13.1072 ms	52.4288 ms
10 (fc/16 clock)	XXX	819.2 μs	3276.8 μs	13.1072 ms	1638.4 μs	6553.6 μs	26.2144 ms	3276.8 μs	13.1072 ms	52.4288 ms

XXX: Don't care

(5) Operating mode summary

Table 3.7.4 shows the settings for the TMRA01 for each of the operating modes.

Table 3.7.4 Register Settings for Each Operating Mode

Register	TA01MOD				TA1FFCR
Field	TA01M[1:0]	PWM[01:00]	TA1CLK[1:0]	TA0CLK[1:0]	TA1FFIS
Function	Interval Timer Mode	PWM Period	UC1 Clock Source	UC0 Clock Source	Timer Flip-flop Toggle Trigger
8-bit timer × 2 ch	00	–	Match output from UC0 ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	0: UC0 output 1: UC1 output
16-bit timer mode	01	–	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PPG × 1 ch	10	–	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PWM × 1 ch	11	2^6, 2^7, 2^8 (01, 10, 11)	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PWM × 1 ch	11	–	ϕT1, ϕT16, ϕT256 (01, 10, 11)	–	Output disabled

–: Don't care

3.8 16-Bit Timers/Event Counters (TMRBs)

The TMP91CW28 has a 16-bit timer/event counter consisting of two identical channels (TMRB0 and TMRB1). Each channel has the following three basic operating modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation (PPG) mode

Each channel has the capture capability used to latch the value of the counter. The capture capability allows:

- Frequency measurement
- Pulse width measurement
- Time difference measurement

Figure 3.8.1 and Figure 3.8.2 are block diagrams of the TMRB0 and the TMRB1.

The main components of a TMRBn block are a 16-bit up counter, two 16-bit timer registers (One of which is double-buffered), two 16-bit capture registers, two comparators, capture control logic, a timer flip-flop and its associated control logic.

A total of eleven special function registers (SFRs) provide control over the operating modes and timer flip-flops for the TMRB0 and the TMRB1 each, which can be independently programmed. The TMRB0 and the TMRB1 are functionally equivalent. In the following sections, any references to the TMRB0 also apply to the TMRB1.

Table 3.8.1 gives the pins and registers for the two channels.

Table 3.8.1 Pins and Registers for the TMRB0 and the TMRB1

Specifications \ Channel		TMRB0	TMRB1
External pins	External clock/capture trigger inputs	TB0IN0 (Shared with P80) TB0IN1 (Shared with P81)	TB1IN0 (Shared with P84) TB1IN1 (Shared with P85)
	Timer flip-flop output	TB0OUT0 (Shared with P82) TB0OUT1 (Shared with P83)	TB1OUT0 (Shared with P86) TB1OUT1 (Shared with P87)
Registers (Addresses)	Timer run register	TB0RUN (0180H)	TB1RUN (0190H)
	Timer mode register	TB0MOD (0182H)	TB1MOD (0192H)
	Timer flip-flop control register	TB0FFCR (0183H)	TB1FFCR (0193H)
	Timer registers	TB0RG0L (0188H) TB0RG0H (0189H) TB0RG1L (018AH) TB0RG1H (018BH)	TB1RG0L (0198H) TB1RG0H (0199H) TB1RG1L (019AH) TB1RG1H (019BH)
	Capture registers	TB0CP0L (018CH) TB0CP0H (018DH) TB0CP1L (018EH) TB0CP1H (018FH)	TB1CP0L (019CH) TB1CP0H (019DH) TB1CP1L (019EH) TB1CP1H (019FH)

3.8.1 Block Diagrams

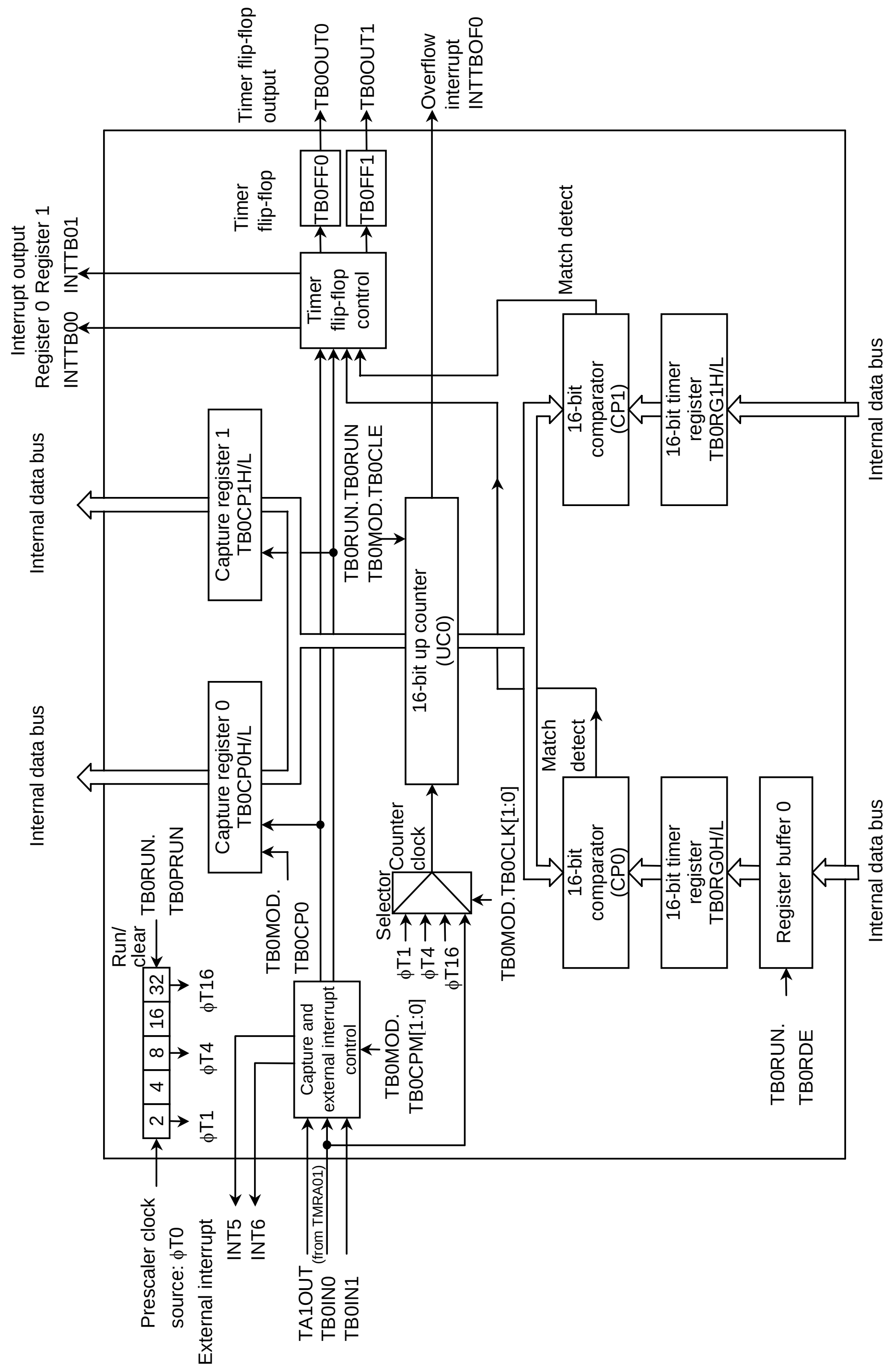


Figure 3.8.1 TMRB0 Block Diagram

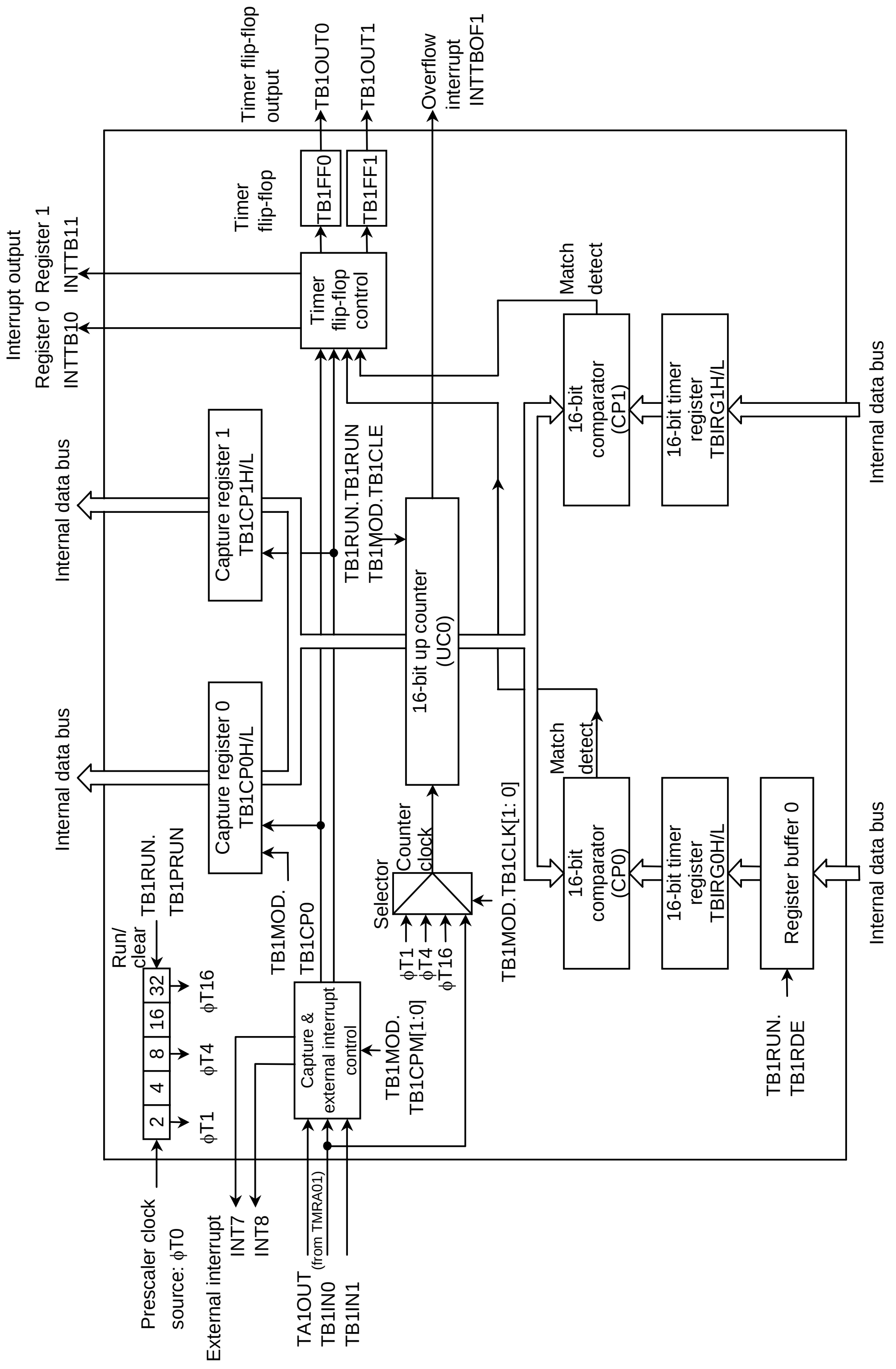


Figure 3.8.2 TMRB1 Block Diagram

3.8.2 Timer Components

(1) Prescaler

The TMRB0 has a 5-bit prescaler that slows the rate of a clocking source to the counter. The prescaler clock source (ϕT0) has one-fourth the frequency selected by programming the PRCK[1:0] field of the SYSCR0 located within the clock gear.

The TB0RUN bit in the TB0RUN register allows the enabling and disabling of the prescaler for the TMRB0. A write of 1 to this bit starts the prescaler. A write of 0 to this bit clears and halts the prescaler. Table 3.8.2 shows prescaler output clock resolutions.

Table 3.8.2 Prescaler Output Clock Resolutions

at fc = 10 MHz

Prescaler Clock Source PRCK[1:0]	Clock Gear Value GEAR[2:0]	Prescaler Output Clock Resolution		
		ϕT1	ϕT4	ϕT16
00 (f_{FPH})	000 (fc)	fc/2^3 (0.8 μs)	fc/2^5 (3.2 μs)	fc/2^7 (12.8 μs)
	001 (fc/2)	fc/2^4 (1.6 μs)	fc/2^6 (6.4 μs)	fc/2^8 (25.6 μs)
	010 (fc/4)	fc/2^5 (3.2 μs)	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)
	011 (fc/8)	fc/2^6 (6.4 μs)	fc/2^8 (25.6 μs)	fc/2^{10} (102.4 μs)
	100 (fc/16)	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)	fc/2^{11} (204.8 μs)
10 (fc/16 clock)	XXX	fc/2^7 (12.8 μs)	fc/2^9 (51.2 μs)	fc/2^{11} (204.8 μs)

xxx: Don't care

(2) Up counter (UC0)

The TMRB0 contains a 16-bit binary up counter, which is driven by a clock selected by the TB0CLK[1:0] field in the TB0MOD register. The clock input to the UC0 is either one of three prescaler outputs (ϕT1, ϕT4, ϕT16) or the external clock applied to the TB0IN0 pin.

The TB0RUN bit in the TB0RUN register is used to start the UC0 and to stop and clear the UC0. The UC0 is cleared to 0000H, if so enabled, when it reaches the value in the TB0RG1H/L register. The TB0CLE bit in the TB0MOD register allows the user to enable and disable this clearing. If it is disabled, the UC0 acts as a free-running counter.

An overflow interrupt (INTTBOF0) is generated upon a counter overflow.

(3) Timer registers (TB0RG0H/L and TB0RG1H/L)

Each timer channel has two 16-bit timer registers containing a time constant. When the up counter reaches the time constant value in each timer register, the associated comparator block generates a match-detect signal.

Setting data for both upper and lower timer registers TB0RG0 and TB0RG1 is always needed. And each of the timer registers (TB0RG0H/L, TB0RG1H/L) can be written with either a halfword-store instruction or a series of two byte-store instructions. When byte-store instructions are used, the low-order byte must be stored first, followed by the high-order byte. The 16-bit timer registers are often simply referred to as TB0RG0 and TB0RG1 without the high and low suffix.

One of the two timer registers, TB0RG0, is double-buffered. The double-buffering function can be enabled and disabled through the programming of the TB0RDE bit in the TB0RUN: 0 = disable, 1 = enable.

If double-buffering is enabled, the TB0RG0 latches a new time constant value from the register buffer. This takes place when a match is detected between the UC0 and the TB0RG1.

Upon reset, the contents of the TB0RG0 and TB0RG1 are undefined; thus, they must be loaded with valid values before the timer can be used. A reset clears the TB0RUN.TB0RDE bit to 0, disabling the double-buffering function. To use this function, the TB0RUN.TB0RDE bit must be set to 1 after loading the TB0RG0 and TB0RG1 with time constants. When TB0RUN.TB0RDE = 1, the next time constant can be written to the register buffer.

The TB0RG0 and the corresponding register buffer are mapped to the same address (0188H and 0189H). When TB0RUN.TB0RDE = 0, a time constant value is written to both the TB0RG0 and the register buffer; when TB0RUN.TB0RDE = 1, a time constant value is written only to the register buffer. Therefore, the double-buffering function should be disabled when writing an initial time constant to the timer register.

The following diagram shows the addresses of each timer register.

TMRB0

TB0RG0		TB0RG1	
8 high-order bits (TB0RG0H)	8 low-order bits (TB0RG0L)	8 high-order bits (TB0RG1H)	8 low-order bits (TB0RG1L)
000189H	000188H	00018BH	00018AH

TMRB1

TB1RG0		TB1RG1	
8 high-order bits (TB1RG0H)	8 low-order bits (TB1RG0L)	8 high-order bits (TB1RG1H)	8 low-order bits (TB1RG1L)
000199H	000198H	00019BH	00019AH

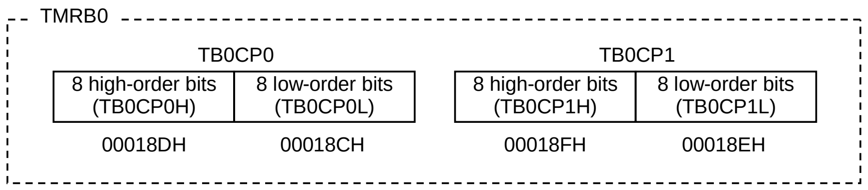
The timer registers are write only registers and cannot be read.

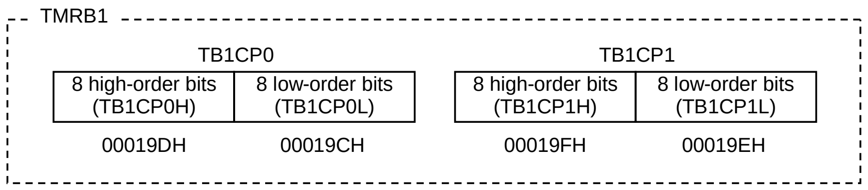
(4) Capture registers (TB0CP0H/L and TB0CP1H/L)

The capture registers are 16-bit registers used to latch the value of the up counter (UC0).

Data in the capture registers should be read all 16 bits. And each of the capture registers can be read with either a halfword-load instruction or a series of two byte-load instructions. When byte-load instructions are used, the low-order byte must be read first, followed by the high-order byte. The 16-bit capture registers are often simply referred to as TBnCP and TBnCP1 without the high and low suffix.

The following diagram shows the addresses of each capture register.


TMRB0
TB0CP0
8 high-order bits (TB0CP0H)
8 low-order bits (TB0CP0L)
00018DH
00018CH
TB0CP1
8 high-order bits (TB0CP1H)
8 low-order bits (TB0CP1L)
00018FH
00018EH


TMRB1
TB1CP0
8 high-order bits (TB1CP0H)
8 low-order bits (TB1CP0L)
00019DH
00019CH
TB1CP1
8 high-order bits (TB1CP1H)
8 low-order bits (TB1CP1L)
00019FH
00019EH

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

(5) Capture and external interrupt control logic

This circuit block controls the capture of an up counter (UC0) value into the capture registers (TB0CP0 and TB0CP1). It also controls generation of external interrupts.

The TB0CPM[1:0] field in the TB0MOD register selects a capture trigger input to be sensed by the control logic, as well as the edge that triggers an external interrupt.

Furthermore, a counter value can be captured under software control; a write of 0 to the TB0MOD.TB0CP0I bit causes the current UC0 value to be latched into the TB0CP0. To use the capture capability, the prescaler must be running (e.g., TB0RUN.TB0PRUN = 1).

(6) Comparators (CP0 and CP1)

The TMRB0 contains two 16-bit comparators. The CP0 block compares the output of the up counter (UC0) with a time constant value in the TB0RG0. The CP1 block compares the output of the UC0 with a time constant value in the TB0RG1. When a match is detected, an interrupt (INTTB00/INTTB01) is generated.

(7) Timer flip-flops (TB0FF0 and TB0FF1)

The timer flip-flops (TB0FF0 and TB0FF1) are toggled, if so enabled, upon assertion of match-detect signals from the comparators and latch signals from the capture control logic. The toggling of the TB0FF0 and TB0FF1 can be enabled and disabled through the programming of the TB0C1T1, TB0C0T1, TB0E1T1 and TB0E0T1 bits in the TB0FFCR register.

Upon reset, the TB0FF0 and TB0FF1 assume an undefined state. They can be initialized to 1 or 0 by writing 01 or 10 to the TB0FF0C[1:0] and TB0FF1C[1:0] fields in the TB0FFCR. A write of 01 to one of these fields sets the corresponding timer flip-flop; a write of 10 clears the timer flip-flop. Additionally, a write of 00 causes the timer flip-flop to be toggled to the opposite value.

The values of the TB0FF0 and TB0FF1 can be driven onto the TB0OUT0 pin, which is multiplexed with P82 and the TB0OUT1 pin, which is multiplexed with P83, respectively. The port 8 registers (P8CR and P8FC) must be programmed to configure the P82/TB0OUT0 pin as TB0OUT0 or the P83/TB0OUT1 pin as TB0OUT1.

3.8.3 SFR Description

TMRB0 Run Register

TB0RUN (0180H)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
Reset value	0	0			0	0		0
Function	Double buffering 0: Disable 1: Enable	Must be written as "0".			IDLE2 0: OFF 1: ON	16-bit timer run/stop control 0: Stop and clear 1: Run		

Counting

0	Stop and clear
1	Count up

I2TB0: Timer ON/OFF in IDLE2 mode
TB0PRUN: Prescaler
TB0RUN: Timer B0

Note: Bits1, 4, and 5 are read as undefined.

TMRB1 Run Register

TB1RUN (0190H)

	7	6	5	4	3	2	1	0
Bit symbol	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
Reset value	0	0			0	0		0
Function	Double buffering 0: Disable 1: Enable	Must be written as "0".			IDLE2 0: OFF 1: ON	16-bit timer run/stop control 0: Stop and clear 1: Run		

Counting

0	Stop and clear
1	Count up

I2TB1: Timer ON/OFF in IDLE2 mode
TB1PRUN: Prescaler
TB1RUN: Timer B1

Note: Bits1, 4, and 5 are read as undefined.

Figure 3.8.3 TMRB Registers (1)

TMRB0 Mode Register

		7	6	5	4	3	2	1	0
TB0MOD (0182H)	Bit symbol	TB0CT1	TB0ET1	TB0CPI	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
	Read/Write	R/W		W*	R/W				
	Reset value	0	0	1	0	0	0	0	0
	Function	TB0FF1 toggle trigger 0: Trigger disabled 1: Trigger enabled		Software capture 0: Capture 1: Undefined	Capture triggers 00: Disabled INT5 rising edge 01: TB0IN0 ↑, TB0IN1 ↑ INT5 rising edge 10: TB0IN0 ↑, TB0IN0 ↓ INT5 falling edge 11: TA0TRG ↑ TA0TRG ↓ INT5 rising edge		Up counter clear control 0: Disable 1: Enable	TMRB0 source clock 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
Read-modify-write operation is not supported.		When UC0 value is latched into capture register 1	When UC0 reaches timer register 1 value						

Input clock

00	External input clock (TB0IN0 input)
01	ϕT1
10	ϕT4
11	ϕT16

Up counter (UC0) clear control

0	Disabled
1	Cleared upon a match with TB0RG1

Capture/interrupt triggers

	Capture control	INT5 control
00	Capture disabled	INT5 occurs at rising edges of TB0IN0.
01	Latches UC0 value into TB0CP0 at rising edges of TB0IN0. Latches UC0 value into TB0CP1 at rising edges of TB0IN1.	
10	Latches UC0 value into TB0CP0 at rising edges of TB0IN0. Latches UC0 value into TB0CP1 at falling edges of TB0IN0.	INT5 occurs at falling edges of TB0IN0.
11	Latches UC0 value into TB0CP0 at rising edges of TA0TRG. Latches UC0 value into TB0CP1 at falling edges of TA0TRG.	INT5 occurs at rising edges of TB0IN0.

Software capture

0	Latches UC0 value into TB0CP0.
1	Undefined

Figure 3.8.4 TMRB Registers (2)

TMRB1 Mode Register

TB1MOD (0192H)

		7	6	5	4	3	2	1	0
TB1MOD (0192H)	Bit symbol	TB1CT1	TB1ET1	TB1CPI	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
	Read/Write	R/W		W*	R/W				
	Reset value	0	0	1	0	0	0	0	0
Read-modify-write operation is not supported.	Function	TB1FF1 toggle trigger 0: Trigger disabled 1: Trigger enabled When UC0 value is latched into capture register 1	When UC0 reaches timer register 1 value	Software capture 0: Capture 1: Undefined	Capture triggers 00: Disabled INT7 rising edge 01: TB1IN0↑, TB1IN1↑ INT7 rising edge 10: TB1IN0↑, TB1IN0↓ INT7 falling edge 11: TA1TRG↑, TA1TRG↓ INT7 rising edge		Up counter clear control 0: Disable 1: Enable	TMRB1 source clock 00: TB1IN0 input 01: φT1 10: φT4 11: φT16	

Input clock

00	External input clock (TB1IN0 input)
01	φT1
10	φT4
11	φT16

Up counter (UC0) clear control

0	Disabled
1	Cleared upon a match with TB1RG1

Capture/interrupt triggers

	Capture control	INT7 control
00	Capture disabled	INT7 occurs at rising edges of TB1IN0.
01	Latches UC0 value into TB1CP0 at rising edges of TB1IN0. Latches UC0 value into TB1CP1 at rising edges of TB1IN1.	
10	Latches UC0 value into TB1CP0 at rising edges of TB1IN0. Latches UC0 value into TB1CP1 at falling edges of TB1IN0.	INT7 occurs at falling edges of TB1IN0.
11	Latches UC0 value into TB1CP0 at rising edges of TA1TRG. Latches UC0 value into TB1CP1 at falling edges of TA1TRG.	INT7 occurs at rising edges of TB1IN0.

Software capture

0	Latches UC0 value into TB1CP0
1	Undefined

Figure 3.8.5 TMRB Registers (3)

TMRB0 Flip-flop Control Register

		7	6	5	4	3	2	1	0
TB0FFCR (0183H)	Bit symbol	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
	Read/Write	W*		R/W				W*	
	Reset value	1	1	0	0	0	0	1	1
Read-modify-write operation is not supported.	Function	TB0FF1 control 00: Toggle 01: Set 10: Clear 11: Don't care This field is always read as "11".		TB0FF0 toggle trigger 0: Trigger disabled 1: Trigger enabled UC0 → TB0CP1	UC0 → TB0CP0	UC0 = TB0RG1	UC0 = TB0RG0	TB0FF0 control 00: Toggle 01: Set 10: Clear 11: Don't care This field is always read as "11".	

Timer flip-flop (TB0FF0) control

00	Toggles TB0FF0 (Software toggle)
01	Sets TB0FF0 to 1
10	Clears TB0FF0 to 0
11	Don't care (Read as 11)

When UC0 reaches TB0RG0 value

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 reaches TB0RG1 value

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 value is latched into TB0CP0

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 value is latched into TB0CP1

0	Toggle trigger disabled
1	Toggle trigger enabled

Figure 3.8.6 TMRB Registers (4)

TMRB1 Flip-flop Control Register

		7	6	5	4	3	2	1	0
TB1FFCR (0193H)	Bit symbol	TB1FF1C1	TB1FF1C0	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
	Read/Write	W*		R/W				W*	
	Reset value	1	1	0	0	0	0	1	1
Read-modify-write operation is not supported.	Function	TB1FF1 control 00: Toggle 01: Set 10: Clear 11: Don't care This field is always read as "11".		TB1FF0 toggle trigger 0: Trigger disabled 1: Trigger enabled UC0 → TB1CP1	UC0 → TB1CP0	UC0 = TB1RG1	UC0 = TB1RG0	TB1FF0 control 00: Toggle 01: Set 10: Clear 11: Don't care This field is always read as "11".	

Timer flip-flop (TB1FF0) control

00	Toggles TB1FF0 (Software toggle)
01	Sets TB1FF0 to 1
10	Clears TB1FF0 to 0
11	Don't care (Read as 11)

When UC0 reaches TB1RG0 value

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 reaches TB1RG1 value

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 value is latched into TB1CP0

0	Toggle trigger disabled
1	Toggle trigger enabled

When UC0 value is latched into TB1CP1

0	Toggle trigger disabled
1	Toggle trigger enabled

Figure 3.8.7 TMRB Registers (5)

3.8.4 Operating Modes

(1) 16-bit interval timer mode

In the following example, the TMRB0 is used to accomplish periodic interrupt generation. The interval time is set in timer register 1 (TB0RG1), and the INTTB01 interrupt is enabled.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	0	X	0	Stops the TMRB0.
INTETB0	←	X	1	0	0	X	0	0	0	Enables INTTB01, sets its priority level to 4 and disables INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disables the timer flip-flop toggle trigger.
TB0MOD	←	0	0	1	0	0	1	*	*	Selects a prescaler output clock as the timer
						(** = 01, 10, 11)				clock source and disables the capture function.
TB0RG1	←	*	*	*	*	*	*	*	*	Sets the interval time
		*	*	*	*	*	*	*	*	(16 bits).
TB0RUN	←	–	0	X	X	–	1	X	1	Starts the TMRB0.

X: Don't care, –: No change

(2) 16-bit event counter mode

This mode is used to count events by interpreting the rising edges of the external counter clock (TB0IN0) as events.

The up counter (UC0) counts up on each rising clock edge. The counter value is latched into a capture register under software control. To determine the number of events (e.g., cycles) counted, the value in the capture register must be read.

		7	6	5	4	3	2	1	0	
TB0RUN	←	–	0	X	X	–	0	X	0	Stops the TMRB0.
P8CR	←	–	–	–	–	–	–	–	0	Configures the P80 pin for input mode.
P8FC	←	–	–	–	–	–	–	–	1	
INTETB0	←	X	1	0	0	X	0	0	0	Enables INTTB01 (Interrupt level = 4) and disables INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disables the timer flip-flop toggle trigger.
TB0MOD	←	0	0	1	0	0	1	0	0	Selects the TB0IN0 input as the timer clock source.
TB0RG1	←	*	*	*	*	*	*	*	*	Sets a count value (16 bits).
TB0RUN	←	–	0	X	X	–	1	X	1	Starts the TMRB0.

X: Don't care, –: No change

Even when the timer is used for event counting, the prescaler must be programmed to run (e.g., the TB0RUN.TB0PRUN bit must be set to 1).

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

The 16-bit PPG mode can be used to generate a square wave with any frequency and duty cycle. The pulse can be high going and low going, as determined by the initial setting of the timer flip-flop (TB0FF).

A square wave is generated by toggling the timer flip-flop every time the up counter UC0 reaches the values in each timer register (TB0RG0 and TB0RG1). The square-wave output is driven to the TB0OUT0 pin. In this mode, the following relationship must be satisfied:

(TB0RG0 value) < (TB0RG1 value)

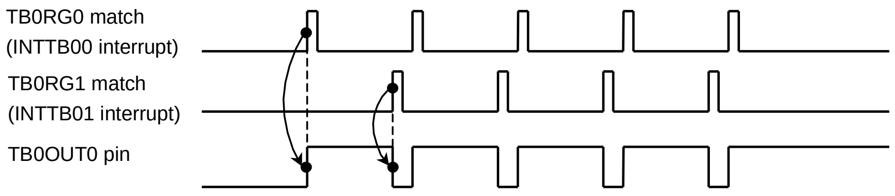


Figure 3.8.8 PPG Output Waveform

If the double-buffering function is enabled, the TB0RG0 value can be changed dynamically by writing a new value into the register buffer. Upon a match between the TB0RG1 and the UC0, the TB0RG0 latches a new value from the register buffer. The TB0RG0 can be loaded with a new value upon every match, thus making it easy to generate a square wave with virtually any duty cycle.

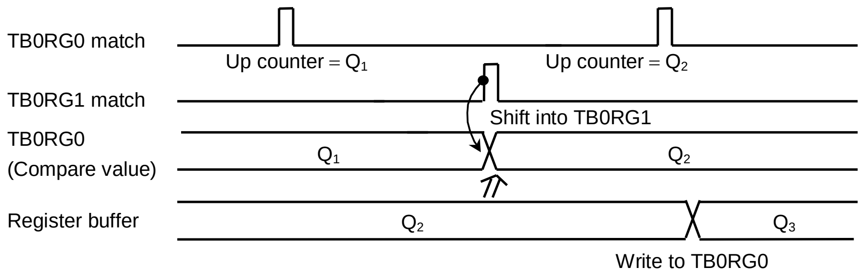


Figure 3.8.9 Register Buffer Operation

Figure 3.8.10 shows a functional diagram of 16-bit PPG mode.

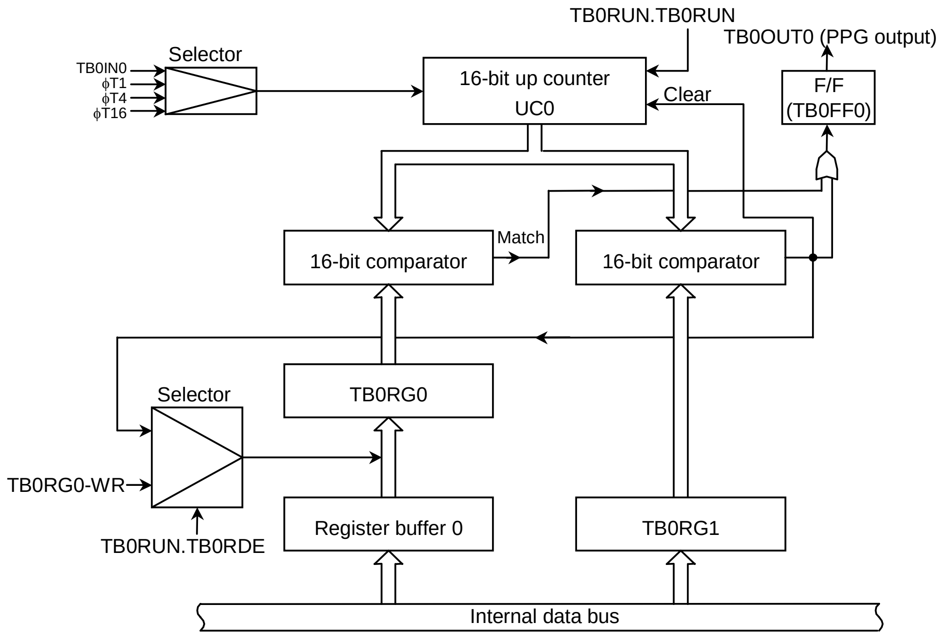


Figure 3.8.10 Functional Diagram of 16-Bit PPG Mode

The following is an example of running the timer in 16-bit PPG mode.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Disables the TB0RG0 double buffering and stops the TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Defines the duty cycle (16 bits).
TB0RG1	←	*	*	*	*	*	*	*	*	Defines the cycle period (16 bits).
TB0RUN	←	1	0	X	X	–	0	X	0	Enables the TB0RG0 double buffering. (The duty cycle and cycle period are changed by the INTTB01 interrupt.)
TB0FFCR	←	X	X	0	0	1	1	1	0	Toggles the TB0FF0 when a match is detected between UC0 and TB0RG0 and between UC0 and TB0RG1. Initially clears the TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Selects a prescaler output clock as the timer clock source and disables the capture function.
						(** = 01, 10, 11)				
P8CR	←	–	–	–	–	–	1	–	–	Configures the P82 pin as TB0OUT0.
P8FC	←	–	–	–	–	–	1	–	–	
TB0RUN	←	1	0	X	X	–	1	X	1	Starts the TMRB0.

X: Don't care, –: No change

(4) Timing and measurement functions using the capture capability

The capture capability of the TMRBn provides versatile timing and measurement functions, including the following:

a. One-shot pulse generation using an external trigger pulse
b. Frequency measurement
c. Pulse width measurement
d. Time difference measurement

a. One-shot pulse generation using an external trigger pulse

The TMRBn can be used to produce a one-time pulse as follows.

The 16-bit up counter (UC0) is programmed to function as a free-running counter, clocked by one of the prescaler outputs. The TB0IN0 pin is used as an active-high external trigger pulse input for latching the counter value into capture register 0 (TB0CP0).

An INT5 interrupt is generated upon detection of a rising edge on the TB0IN0/INT5 pin. A one-shot pulse has a delay and width controlled by the values stored in the timer registers (TB0RG0 and TB0RG1). Programming the TB0RG0 and TB0RG1 is the responsibility of the INT5 interrupt handler. The TB0RG0 is loaded with the sum of the TB0CP0 value (c) plus the pulse delay (d) – e.g., (c) + (d). The TB0RG1 is loaded with the sum of the TB0RG0 value plus the pulse width (p) – e.g., (c) + (d) + (p).

Next, the TB0E1T1 and TB0E0T1 bits in the timer flip-flop control register (TB0FFCR) are set to 11, so that the timer flip-flop (TB0FF0) will toggle when a match is detected between the UC0 and the TB0RG0 and between the UC0 and the TB0RG1. With the TB0FF0 toggled twice, a one-shot pulse is produced. Upon a match between the UC0 and the TB0RG1, the TMRB0 generates the INTTB01 interrupt, which must disable the toggle trigger for the TB0FF0.

Figure 3.8.11 depicts one-shot pulse generation, with annotations showing (c), (d), and (p).

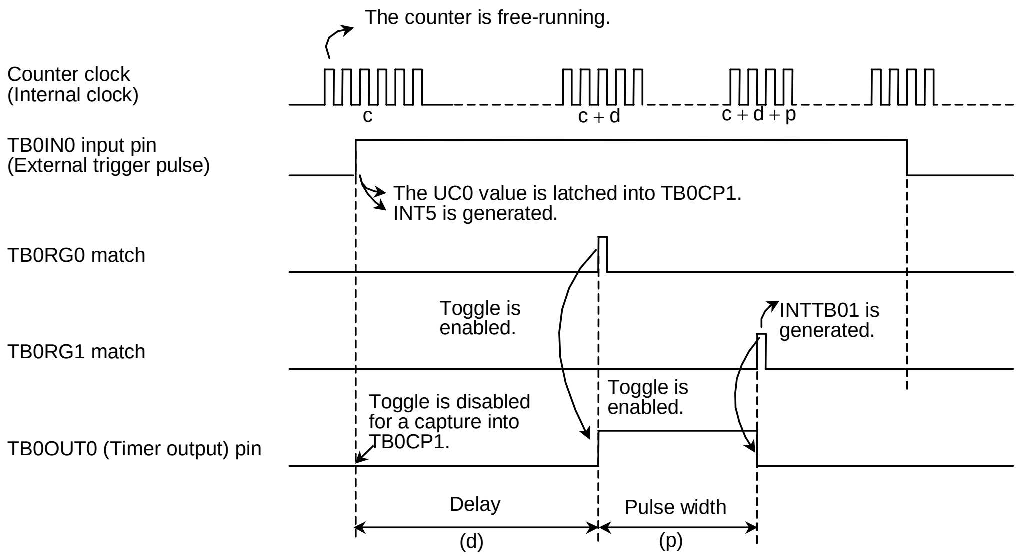

The counter is free-running.
Counter clock (Internal clock)
c
c + d
c + d + p
TB0IN0 input pin (External trigger pulse)
The UC0 value is latched into TB0CP1. INT5 is generated.
TB0RG0 match
Toggle is enabled.
INTTB01 is generated.
TB0RG1 match
Toggle is enabled.
Toggle is disabled for a capture into TB0CP1.
TB0OUT0 (Timer output) pin
Delay
(d)
Pulse width
(p)

Figure 3.8.11 One-shot Pulse Generation (with a delay)

Example: Generating a one-shot pulse with a width of 2 ms and a delay of 3 ms on assertion of an external trigger pulse on the TB0IN0 pin

Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

Settings in the main routine

```
                                             → Places the counter in free-running mode.
                                             → Selects φT1 as the counter clock source.
TB0MOD    ←  X  X  1  0  1  0  0  1
                                             → Latches UC0 value into TB0CP0 at rising edges of the
TB0FFCR   ←  X  X  0  0  0  0  1  0            TB0IN0 input.
                                             → Clears TB0FF0 to 0.
                                             → Disables the toggle trigger for TB0FF0.
P8CR      ←  –  –  –  –  –  1  –  –          } Configures the P82 pin as TB0OUT0.
P8FC      ←  –  –  –  –  –  1  –  –

INTE56    ←  X  –  –  –  X  1  0  0            Enables INT5 and disables INTTB00 and INTTB01.
INTETB0   ←  X  0  0  0  X  0  0  0
TB0RUN    ←  –  0  X  X  –  1  X  1            Starts the TMRB0.
```

Settings in INT5

```
TB0RG0    ←  TB0CP0 + 3 ms/φT1
TB0RG1    ←  TB0RG0 + 2 ms/φT1
TB0FFCR   ←  X  X  –  –  1  1  –  –
                                             → Enables the TB0FF0 toggle trigger for TB0RG0 and
                                               TB0RG1 matches.
INTETB0   ←  X  1  0  0  X  –  –  –            Enables INTTB01.
```

Settings in INTTB01

```
TB0FFCR   ←  X  X  –  –  0  0  –  –
                                             → Disables the TB0FF0 toggle trigger for TB0RG0 and
                                               TB0RG1 matches.
INTETB0   ←  X  0  0  0  X  –  –  –            Disables INTTB01.
```

X: Don't care, –: No change

If no delay is necessary, enable the TB0FF0 toggle trigger for a capture of the UC0 value into the TB0CP0. Use the INT5 interrupt to load the TB0RG1 with a sum of the TB0CP0 value (c) plus the pulse width (p) and to enable the TB0FF0 toggle trigger for a match between the UC0 and TB0RG1 values. A match generates the INTTB01 interrupt, which then is to disable the TB0FF0 toggle trigger.

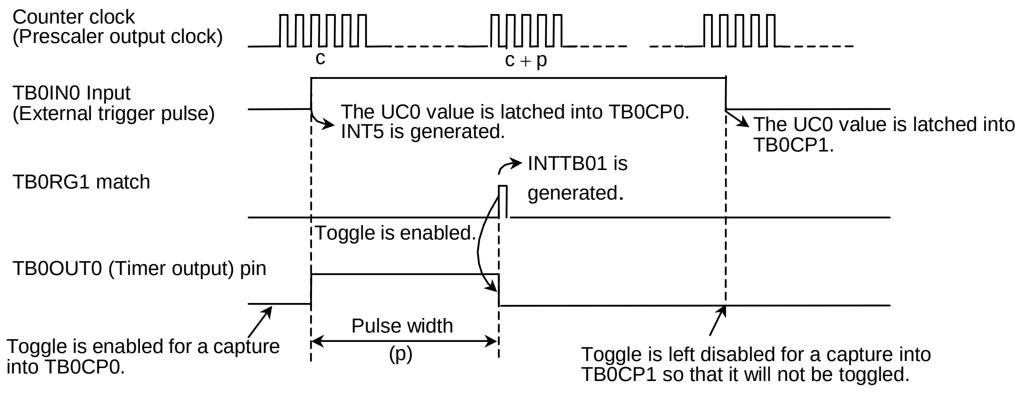

Counter clock
(Prescaler output clock)
c
c + p
TB0IN0 Input
(External trigger pulse)
The UC0 value is latched into TB0CP0.
INT5 is generated.
The UC0 value is latched into TB0CP1.
TB0RG1 match
INTTB01 is generated.
Toggle is enabled.
TB0OUT0 (Timer output) pin
Pulse width
(p)
Toggle is enabled for a capture into TB0CP0.
Toggle is left disabled for a capture into TB0CP1 so that it will not be toggled.

Figure 3.8.12 One-shot Pulse Generation (without a delay)

b. Frequency measurement

The capture function can be used to measure the frequency of an external clock. Frequency measurement requires a 16-bit TMRBn channel running in event counter mode and the 8-bit TMRA01. The timer flip-flop (TA1FF) in the TMRA01 is used to define the duration during which a measurement is taken.

Select the TB0IN0 pin as the clock source for the TMRB0. Set the TB0CPM[1:0] field in the TB0MOD to 11 to select the TA1FF output signal from the TMRA01 as a capture trigger input. This causes the TMRB0 to latch the 16-bit up counter (UC0) value into capture register 0 (TB0CP0) on the low-to-high transition of the TA1FF and into capture register 1 (TB0CP1) on the next high-to-low transition of the TA1FF.

Either the INTTA0 or INTTA1 interrupt generated by the 8-bit timer can be used to make a frequency calculation.

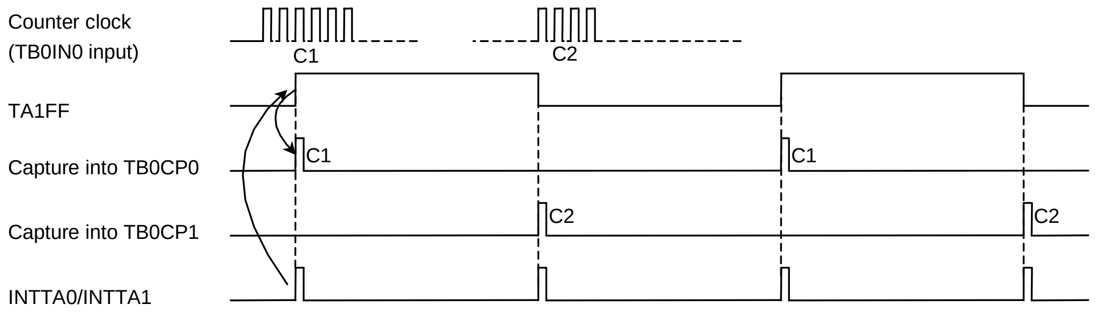


Figure 3.8.13 Frequency Measurement

For example, if the TA1FF of the 8-bit timer is programmed to be at logic 1 for a period of 0.5 seconds and the difference between the values captured into the TB0CP0 and TB0CP1 is 100, then the TB0IN0 frequency is calculated as 100 ÷ 0.5 s = 200 Hz.

c. Pulse width measurement

The capture function can be used to measure the pulse width of an external clock. The external clock is applied to the TB0IN0 pin. The up counter (UC0) is programmed to operate as a free-running counter, clocked by one of the prescaler outputs. The capture function is used to latch the UC0 value into capture register 0 (TB0CP0) at the clock rising edge and into capture register 1 (TB0CP1) at the next clock falling edge. An INT5 interrupt is generated at the falling edge of the TB0IN0 input.

Multiplying the counter clock period by the difference between the values captured into the TB0CP0 and TB0CP1 gives the high pulse width of the TB0IN0 clock.

For example, if the prescalar output clock has a period of 0.8 μs and the difference between the TB0CP0 and TB0CP1 is 100, the high pulse width is calculated as 0.8 μs × 100 = 80 μs.

Measuring a pulse width exceeding the maximum counting time for the UC0, which depends on the clock source, requires software programming.

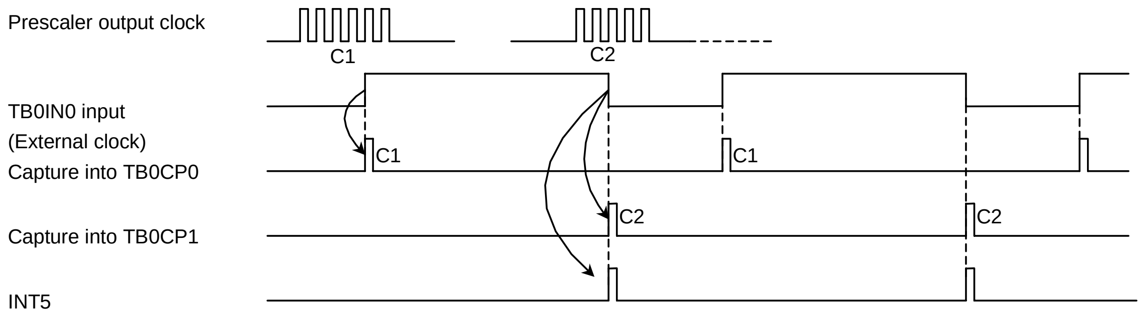


Figure 3.8.14 Pulse Width Measurement

Note: To measure a pulse width, set the TB0CPM[1:0] field of the TB0MOD to 10, so that an INT5 external interrupt is generated at the falling edge of the TB0IN0 input. Otherwise, an INT5 interrupt is generated at the rising edge of the TB0IN0 input.

The low pulse width can be measured by the second INT5 interrupt. This is accomplished by multiplying the counter clock period by the difference between the TB0CP0 value at the first C2 and the TB0CP1 value at the second C1.

d. Time difference measurement

The capture function can be used to measure the time difference between two event occurrences. The 16-bit up counter (UC0) is programmed to operate as a free-running counter. The UC0 value is latched into capture register 0 (TB0CP0) on the rising edge of TB0IN0. An INT5 interrupt is generated at this time.

Then, the UC0 value is latched into capture register 1 (TB0CP1) on the rising edge of TB0IN1. An INT6 interrupt is generated at this time.

The time difference between the two events that occurred on the TB0IN0 and TB0IN1 pins is calculated by multiplying the counter clock period by the difference between the TB0CP1 and TB0CP0 values.

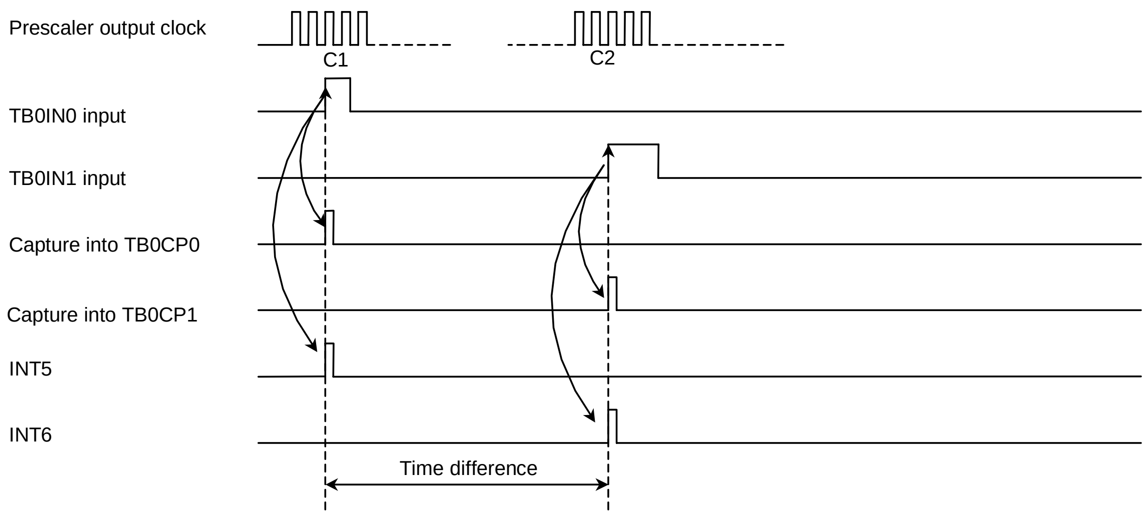


Figure 3.8.15 Time Difference Measurement

3.9 Serial I/O (SIO)

The TMP91CW28 contains a single serial I/O channel (SIO). The SIO provides universal asynchronous receiver/transmitter (UART) mode and synchronous I/O interface mode.

- I/O interface mode —— Transmits/receives a serial clock (SCLK) as well as data streams for a synchronous clock mode of operation.
- UART mode —— Mode 1: 7 data bits
  Mode 2: 8 data bits
  Mode 3: 9 data bits

In mode 1 and mode 2, each character can include a parity bit. In mode 3, the SIO channel operates in a wakeup mode for multidrop applications in which a master station is connected to several slave stations through a serial link.

Figure 3.9.2 is a block diagram of the SIO channel. The main components of the SIO channel are a clock prescaler, a serial clock generator, a receive buffer, a receive controller, a transmit buffer and a transmit controller.

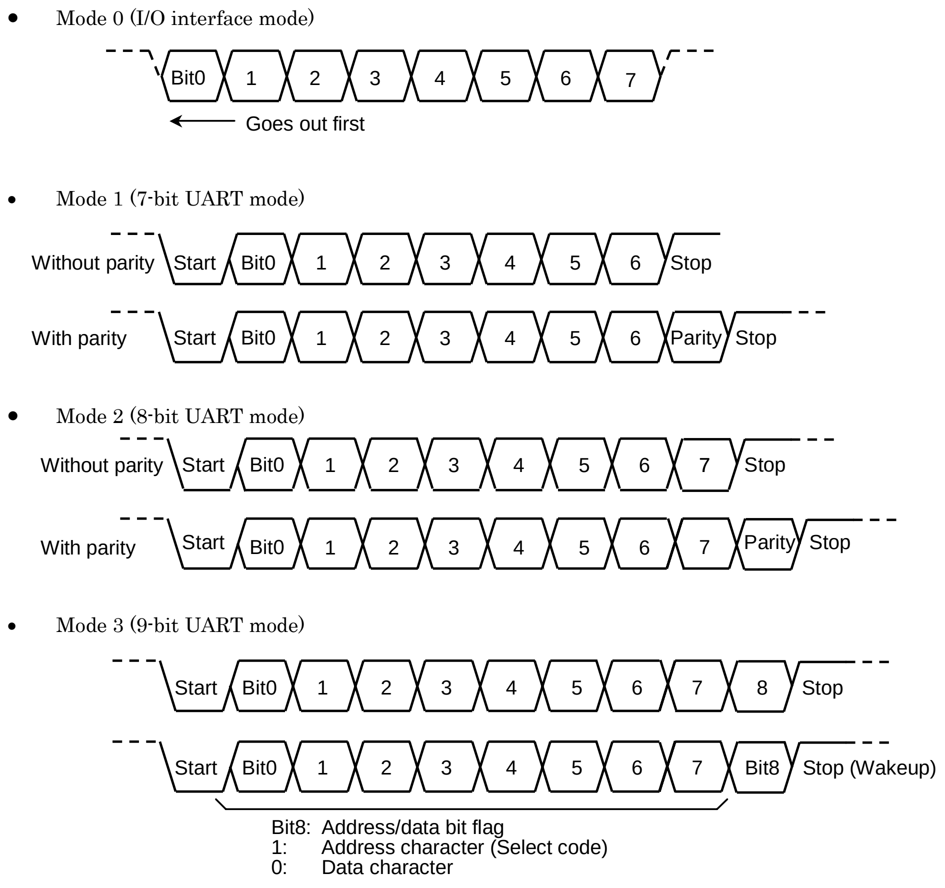

• Mode 0 (I/O interface mode)
Bit0 1 2 3 4 5 6 7
Goes out first
• Mode 1 (7-bit UART mode)
Without parity Start Bit0 1 2 3 4 5 6 Stop
With parity Start Bit0 1 2 3 4 5 6 Parity Stop
• Mode 2 (8-bit UART mode)
Without parity Start Bit0 1 2 3 4 5 6 7 Stop
With parity Start Bit0 1 2 3 4 5 6 7 Parity Stop
• Mode 3 (9-bit UART mode)
Start Bit0 1 2 3 4 5 6 7 8 Stop
Start Bit0 1 2 3 4 5 6 7 Bit8 Stop (Wakeup)
Bit8: Address/data bit flag
1: Address character (Select code)
0: Data character

Figure 3.9.1 Data Formats

3.9.1 Block Diagrams

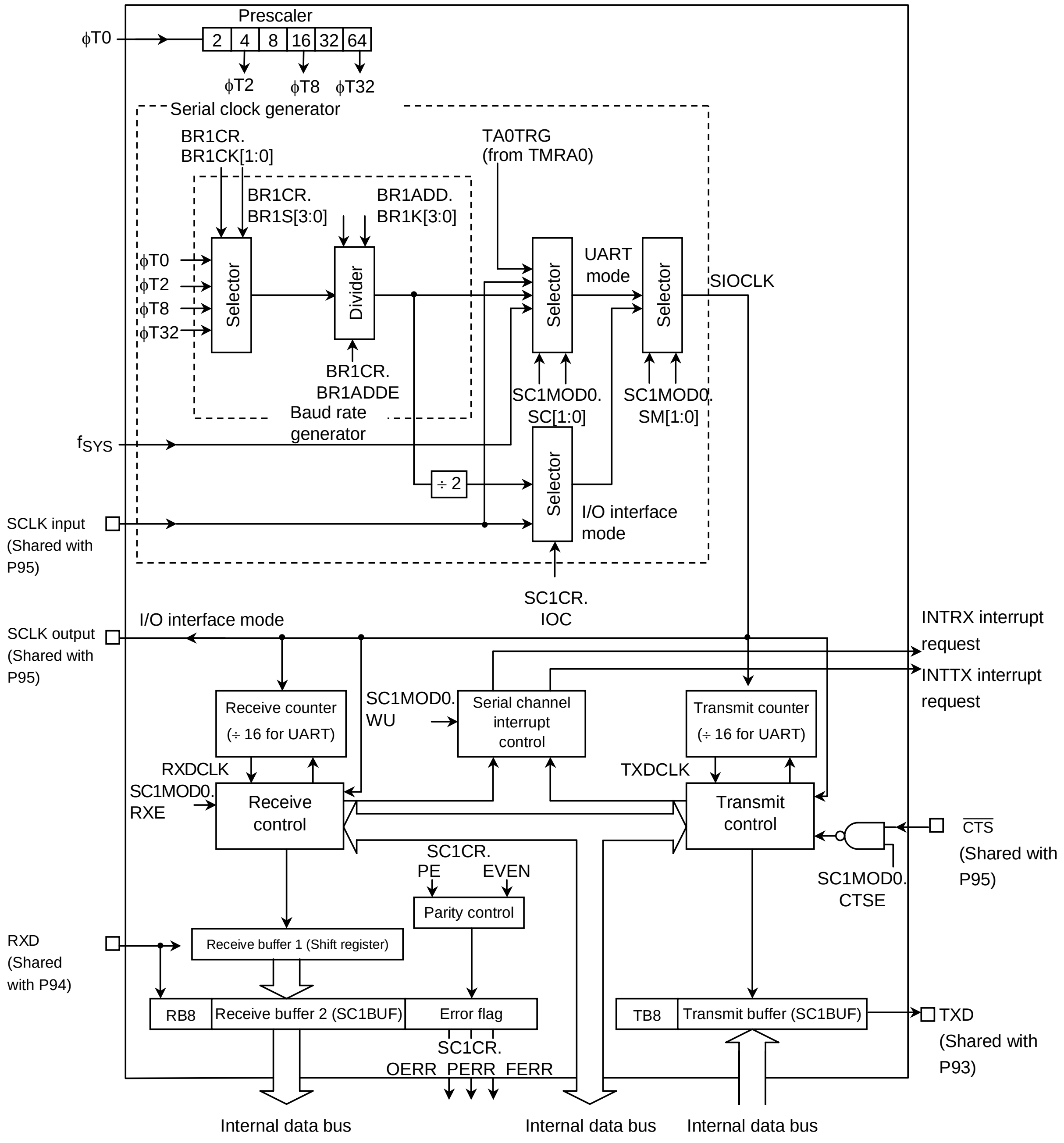


Figure 3.9.2 SIO Block Diagram

3.9.2 SIO Components

(1) Prescaler

The SIO has a 6-bit prescaler that slows the rate of a clocking source to the serial clock generator. The prescaler clock source (ϕT0) has one-fourth the frequency selected by programming the PRCK[1:0] field of the SYSCR0 located within the clock gear.

The serial clock is selectable from several clocks, the prescaler is only enabled when the baud rate generator output clock is selected as a serial clock. Table 3.9.1 shows prescaler output clock resolutions.

Table 3.9.1 Prescaler Output Clock Resolutions

Prescaler Clock Source PRCK[1:0]	Clock Gear Value GEAR[2:0]	Prescaler Output Clock Resolution			
		ϕT0	ϕT2	ϕT8	ϕT32
00 (f_{FPH})	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}
10 (fc/16 clock)	XXX	–	fc/2^8	fc/2^{10}	fc/2^{12}

XXX: Don’t care, –: Setting prohibited

Prescaler output taps can be divide-by-1 (ϕT0), divide-by-4 (ϕT2), divide-by-16 (ϕT8), and divide-by-64 (ϕT32).

(2) Baud rate generator

The frequency used to transmit and receive data through the SIO is derived from the baud rate generator. The clock source for the baud rate generator can be selected from the 6-bit prescaler outputs (ϕT0, ϕT2, ϕT8, ϕT32) through the programming of the BR0CK[1:0] field in the BR1CR.

The baud rate generator contains a clock divider that can divide the selected clock by 1, N + (16 − K)/16, or 16. The clock divisor is programmed into the BR1ADDE and BR0S[3:0] bits in the BR1CR and the BR0K[3:0] bits in the BR1ADD.

- UART mode

(1) When BR1CR.BR1ADDE = 0

When the BR1CR.BR1ADDE bit is cleared, the BR1ADD.BR0K[3:0] field has no meaning or effect. In this case, the baud rate generator input clock is divided down by a value of N (1 to 16) programmed in the BR1CR.BR0S[3:0] field.

(2) When BR1CR.BR1ADDE = 1

Setting the BR1CR.BR1ADDE bit enables the N + (16 − K)/16 clock division function. The baud rate generator input clock is divided down according to the value of N (2 to 15) programmed in the BR1CR.BR0S[3:0] field and the value of K (1 to 15) programmed in the BR1ADD.BR0K[3:0] field.

Note: Setting N to 1 or 16 disables the N + (16 − K)/16 clock division function. When N = 1 or 16, the BR1CR.BR1ADDE bit must be cleared.

- I/O interface mode

I/O Interface mode cannot utilize the N + (16 − K)/16 clock division function. The BR1CR.BR1ADDE must be cleared, so the baud rate generator input clock is divided down by a value of N (1 to 16) programmed in the BR1CR.BR0S[3:0] field.

When the baud rate generator is used, the baud rate is calculated as follows:

- UART mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 16$$

- I/O interface mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 2$$

- Integral clock division (divide-by-N)

  fc = 9.8304 MHz

  Input clock: $\phi T2$

  Clock divisor N (BR1CR.BR0S[3:0]) = 5

  BR1CR.BR1ADDE = 0

  Clocking conditions: $\left\{ \begin{array}{ll} \text{High-speed clock gear:} & \times 1\ (\text{fc}) \\ \text{Prescaler clock:} & f_{FPH} \end{array} \right.$

  The baud rate is determind as follows:

  $$\text{Baud rate} = \frac{\text{fc}/16}{4} \div 16$$

  $$= 9.8304 \times 10^6 \div 16 \div 4 \div 16 = 9600\ (\text{bps})$$

  Note: Clearing the BR1CR.BR1ADDE bit to 0 disables the N + (16 − K)/16 clock division function. At this time, the BR1ADD.BR0K[3:0] field is ignored.

- N + (16 − K)/16 clock division (UART mode only)

  fc = 4.8 MHz

  Input clock: $\phi T0$

  N (BR1CR.BR0S[3:0]) = 7

  K (BR1ADD.BR0K[3:0]) = 3

  BR1CR.BR1ADDE = 1

  Clocking conditions: $\left\{ \begin{array}{ll} \text{High-speed clock gear:} & \times 1\ (\text{fc}) \\ \text{Prescaler clock:} & f_{FPH} \end{array} \right.$

  The baud rate is determind as follows:

  $$\text{Baud rate} = \frac{\text{fc}/4}{7 + \frac{(16-3)}{16}} \div 16$$

  $$= 4.8 \times 10^6 \div 4 \div \left(7 + \frac{13}{16}\right) \div 16 = 9600\ (\text{bps})$$

Table 3.9.2 and Table 3.9.3 show the UART baud rates obtained with various combinations of clock inputs and clock divisor values.

The SIO can use an external clock as a serial clock, bypassing the baud rate generator. When an external clock is used, the baud rate is determined as shown below.

- UART mode

  Baud rate = external clock input ÷ 16

  The external clock period must be greater than or equal to 4/fc.

- I/O interface mode

  Baud rate = external clock input

  The external clock period must be greater than or equal to 16/fc.

Table 3.9.2 UART Baud Rate Selection

(when the baud rate generator is used and BR1CR.BR1ADDE = 0) Unit: kbps

fc [MHz]	Baud Rate Generator Input Clock / Divisor N (Programmed in BR1CR. BR0S[3:0])	ϕT0	ϕT2	ϕT8	ϕT32
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

Note 1: In I/O interface mode, the transfer rate is eight times the value shown in this table.

Note 2: This table assumes: clock gear = fc, prescaler clock source (ϕT0) = f_{FPH}

Table 3.9.3 UART Baud Rate Selection

(when the TMRA0 timer trigger output is used and the TMRA0 input clock is ϕT1)

Unit: kbps

fc / TA0REG0	9.8304 MHz	8 MHz	6.144 MHz
1H	76.8	62.5	48
2H	38.4	31.25	24
3H			16
4H	19.2		12
5H			9.6
8H	9.6		6
AH			4.8
10H	4.8		3
14H			2.4

When the 8-bit timer TMRA0 is used to generate a serial clock, the baud rate is determined by the following equation:

$$\text{Baud rate} = \frac{\text{clock frequency selected by SYSCR0.PRCK[1:0]}}{\text{TA0REG} \times \underset{\substack{\uparrow\\ \text{When the TMRA0 clock source is } \phi\text{T1}}}{\underline{8}} \times 16}$$

Note 1: I/O interface mode cannot utilize the trigger output signal from the 8-bit timer TMRA0 as a serial clock.

Note 2: This table assumes: Clock gear = fc, prescaler clock source (ϕT0) = $f_{FPH.}$

(3) Serial clock generator

This block generates a basic clock (SIOCLK) that controls the transmit and receive circuit.

- I/O interface mode

  When the SCLK pin is configured as an output by clearing the SC1CR.IOC bit to 0, the output clock from the baud rate generator is divided by two to generate the SIOCLK clock. When the SCLK pin is configured as an input by setting the SC1CR.IOC bit to 1, the external SCLK clock is used as the SIOCLK clock; the SC1CR.SCLKS bit determines the active clock edge.

- UART mode

  The SIOCLK clock is selected from a clock produced by the baud rate generator, the system clock (f_{SYS}), the trigger output signal from the 8-bit timer TMRA0, and the external SCLK clock, according to the setting of the SC1MOD0.SC[1:0] field.

(4) Receive counter

The receive counter is a 4-bit binary up counter used in UART mode. This counter is clocked by SIOCLK. The receiver utilizes 16 clocks for each received bit, and oversamples each bit three times around their center (with 7th to 9th clocks), unless f_{SYS} is used for the basic clock. The value of a bit is determined by voting logic which takes the value of the majority of three samples. For example, if the three samples of a bit are 1, 0 and 1, then that bit is interpreted as a 1; if the three samples of a bit are 0, 0 and 1, then that bit is interpreted as a 0.

(5) Receive controller

- I/O interface mode

  If the SCLK pin is configured as an output by clearing the SC1CR.IOC bit to 0, the receive controller samples the RXD input at the rising edge of the shift clock driven out from the SCLK pin. If the SCLK pin is configured as an input by setting the SC1CR.IOC bit to 1, the receive controller samples the RXD input at either the rising or falling edge of the SCLK clock, as programmed in the SC1CR.SCLKS bit.

- UART mode

  The receive controller contains the start bit detection logic. Once a valid start bit is detected (at least two 0 are detected among three samples), the receive controller begins sampling the incoming data streams. The start bit, each data bit and the stop bit are sampled three times for 2-of-3 majority voting.

(6) Receive buffer

The receive buffer is double-buffered to prevent overrun errors. Received data is serially shifted bit by bit into receive buffer 1. When a whole character (e.g., 7 or 8 bits, as programmed) is loaded into receive buffer 1, it is transferred to receive buffer 2 (SC1BUF), and a receive-done interrupt (INTRX) is generated.

The CPU reads a character from receive buffer 2 (SC1BUF). Receive buffer 1 can accept a new character through the RXD pin before the CPU picks up the previous character in receive buffer 2. However, the CPU must read receive buffer 2 before receive buffer 1 is filled with a new character. Otherwise, an overrun error occurs, causing the character previously in receive buffer 1 to be lost. Even in that case, the contents of receive buffer 2 and the SC1CR.RB8 bit are preserved.

The SC1CR.RB8 bit holds the parity bit for an 8-bit UART character and the most-significant (e.g., address/data flag) bit for a 9-bit UART character.

In 9-bit UART mode, the receiver wakeup feature allows the slave station in a multidrop system to wakeup whenever an address character is received. Setting the SC1MOD0.WU bit enables the wakeup feature. When the SC1CR.RB8 bit has received an address/data flag bit set to 1, the receiver generates the INTRX interrupt.

(7) Transmit counter

The transmit counter is a 4-bit binary up counter used in UART mode. Like the receive counter, the transmit counter is also clocked by SIOCLK. The transmitter generates a transmit clock (TXDCLK) pulse every 16 SIOCLK pulses.

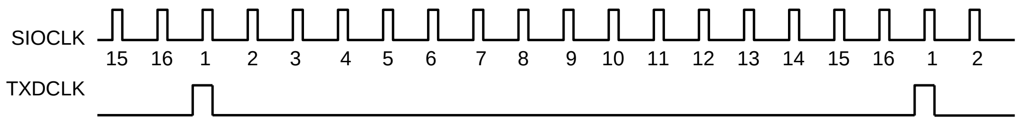


Figure 3.9.3 Transmit Clock Generation

(8) Transmit controller

- I/O interface mode

  If the SCLK pin is configured as an output by clearing the SC1CR.IOC bit to 0, the transmit controller shifts out each bit in the transmit buffer to the TXD pin at the rising edge of the shift clock driven out on the SCLK pin. If the SCLK pin is configured as an input by setting the SC1CR.IOC bit to 1, the transmit controller shifts out each bit in the transmit buffer to the TXD pin at either the rising or falling edge of the SCLK input, as programmed in the SC1CR.SCLKS bit.

- UART mode

  Once the CPU loads a character into the transmit buffer, the transmit controller begins transmission at the next rising edge of TXDCLK, producing a transmit shift clock (TXDSFT).

Handshaking

The SIO has the clear-to-send ($\overline{\text{CTS}}$) pin. If the $\overline{\text{CTS}}$ operation is enabled, the $\overline{\text{CTS}}$ input must be low in order for the character to be transmitted. This feature can be used for flow control to prevent overrun in the receiver. The SC1MOD0.CTSE bit enables and disables the $\overline{\text{CTS}}$ operation.

If the $\overline{\text{CTS}}$ pin goes high in the middle of a transmission, the transmit controller stops transmission upon completion of the current character until $\overline{\text{CTS}}$ again goes low. If so enabled, the transmit controller generates the INTTX interrupt to notify the CPU that the transmit buffer is empty. After the CPU loads the next character into the transmit buffer, the transmit controller remains in idle state until it detects $\overline{\text{CTS}}$ going low.

Although the SIO does not have the $\overline{\text{RTS}}$ pin, any general-purpose port pins can serve as the $\overline{\text{RTS}}$ pin. The receiving device uses the $\overline{\text{RTS}}$ output to control the $\overline{\text{CTS}}$ input of the transmitting device. Once the receiving device has received a character, $\overline{\text{RTS}}$ should be set to high in the receive-done interrupt handler to temporarily stop the transmitting device from sending the next character. This way, the user can easily implement a two-way handshake protocol.

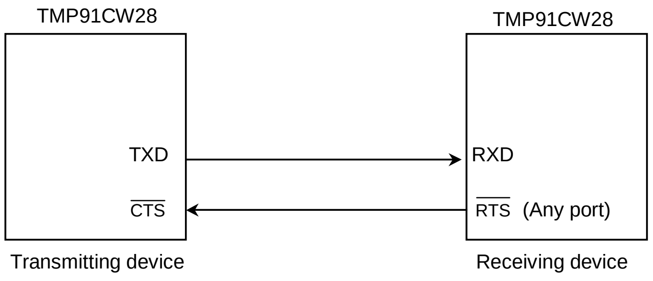
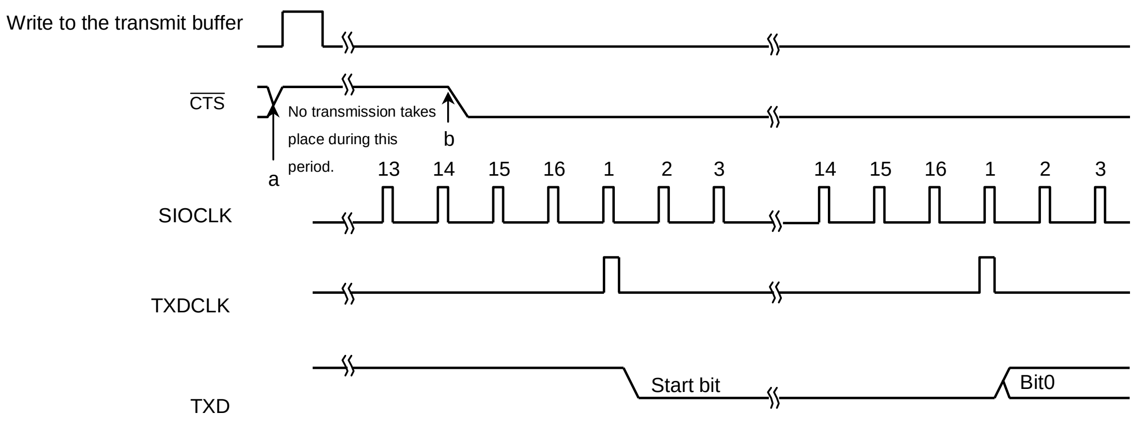


Figure 3.9.4 Handshaking Signals


Write to the transmit buffer
CTS
No transmission takes place during this period.
a
b
13 14 15 16 1 2 3
14 15 16 1 2 3
SIOCLK
TXDCLK
TXD
Start bit
Bit0

Note: a. When $\overline{\text{CTS}}$ goes high in the middle of transmission, the transmitter stops transmission after the current character has been sent.

b. The transmitter starts transmission at the first falling edge of the TXDCLK clock after the $\overline{\text{CTS}}$ signal goes low.

Figure 3.9.5 Clear-to-send ($\overline{\text{CTS}}$) Signal Timing

(9) Transmit buffer

Once the CPU loads a character into the transmit buffer (SC1BUF), it is shifted out on the TXD output, with the least-significant bit first, clocked by the transmit shift clock TXDSFT from the transmit controller. When the transmit buffer is empty and ready to be loaded with the next character, the INTTX interrupt is generated to the CPU.

(10) Parity controller

For transmit operations, setting the SC1CR.PE enables parity generation in 7- and 8-bit UART modes. The SC1CR.EVEN bit selects either even or odd parity.

If enabled, the parity controller automatically generates parity for the character in the transmit buffer (SC1BUF). In 7-bit UART mode, the TB7 bit in the SC1BUF holds the parity bit. In 8-bit UART mode, the TB8 bit in the SC1MOD holds the parity bit. The parity bit is set after the character has been transmitted. The SC1CR.PE and SC1CR.EVEN bits must be programmed prior to a write to the transmit buffer.

For receive operations, the parity controller automatically computes the expected parity when a character in receive buffer 1 is transferred to receive buffer 2 (SC1BUF). The received parity bit is compared to the SC1BUF.RB7 bit in 7-bit UART mode and to the SC1CR.RB8 bit in 8-bit UART mode. If a character is received with incorrect parity, the SC1CR.PERR bit is set.

(11) Error flags

The SC1CR has the following error flag bits that indicate the status of the received character for improved data reception reliability.

1. Overrun error (OERR)

   An overrun error is reported if all bits of a new character are received into receive buffer 1 when receive buffer 2 (SC1BUF) still contains a valid character.

   The following shows an example processing flow when an overrun error occurs:

```
(Receive interrupt routine)
  1) Read the receive buffer.
  2) Read the error flags.
  3) if OERR = 1
     then
       a) Disable reception: Write 0 to RXE.
       b) Wait until the current frame is completed.
       c) Read the receive buffer.
       d) Read the error flags.
       e) Enable reception: Write 1 to RXE.
       f) Request retransmission.
  4) Other processing
```

2. Parity error (PERR)

A parity error is reported when the parity bit attached to a character received on the RXD pin does not match the expected parity computed from the character transferred to receive buffer 2 (SC1BUF).

- Restrictions on the use of the parity error flag (Only for the TMP91CW12 and TMP91PW12)

  The parity error flag (SC1CR.PERR) cannot be used when parity generation is enabled in full-duplex UART mode. In that case, detect a parity error using the CPU parity flag based on the received data, as shown below.

(1) 8 bits + odd parity

```
ld     wa,(SC1BUF)      ; Read the received data and parity bit.
ld     b,w              ; Save the overrun and framing error flags.
and    wa,80ffh         ; Mask other bits and set the parity flag.
jr     pe,parity_err    ; Branch to the error handling routine if the parity flag = 1.
```

(2) 7 bits + odd parity

```
ld     a,(SC1BUF)       ; Read the received data and parity bit.
and    a,0ffh           ; Set the parity flag.
jr     pe,parity_err    ; Branch to the error handling routine if the parity flag = 1.
```

3. Framing error (FERR)

A framing error is reported when a 0 is detected where a stop bit was expected. (The middle three of the 16 samples are used to determine the bit value.)

(12) Signal generation timing

a. UART mode

Receive operation

Mode	9 Data Bits	8 Data Bits with Parity	8 Data Bits with No Parity 7 Data Bits with Parity 7 Data Bits with No Parity
Interrupt	Middle of the last bit (e.g., bit8)	Middle of the last bit (e.g., parity bit)	Middle of the stop bit
Framing error	Middle of the stop bit	Middle of the stop bit	Middle of the stop bit
Parity error	–	Middle of the last bit (e.g., parity bit)	←
Overrun error	Middle of the last bit (e.g., bit8)	Middle of the last bit (e.g., parity bit)	Middle of the stop bit

Note: In 9 data bits and 8 data bits with No 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.

Transmit operation

Mode	9 Data Bits	8 Data Bits with Parity	8 Data Bits with No Parity 7 Data Bits with Parity 7 Data Bits with No Parity
Interrupt	Immediately before the stop bit is shifted out	←	←

b. I/O interface mode

Transmit interrupt	SCLK output mode	Immediately after the rising edge of the last SCLK pulse (See Figure 3.9.13)
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed (See Figure 3.9.14)
Receive interrupt	SCLK output mode	When a received character has been transferred to receive buffer 2 (SC1BUF) (e.g., immediately after the last SCLK pulse) (See Figure 3.9.15)
	SCLK input mode	When a received character has been transferred to receive buffer 2 (SC1BUF) (e.g., immediately after the last SCLK pulse) (See Figure 3.9.16)

3.9.3 SFR Description

Serial Mode Control Register 0

SC1MOD0 (020AH)		7	6	5	4	3	2	1	0
	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Bit8 of a transmitted character	Handshake control 0: Disables $\overline{\text{CTS}}$ operation 1: Enables $\overline{\text{CTS}}$ operation	Receive control 0: Disables receiver 1: Enables receiver	Wakeup function 0: Disabled 1: Enabled	Serial transfer mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial clock (for UART) 00: TA0TRG (Timer) 01: Baud rate generator 10: Internal f_{SYS} clock 11: External clock (SCLK input)	

Serial clock (for UART)

00	Trigger output signal from the TMRA0 timer
01	Baud rate generator
10	Internal f_{SYS} clock
11	External clock (SCLK input)

Note: In I/O interface mode, the serial control register (SC1CR) is used to select the clock source.

Serial transfer mode

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

Wakeup function

	9-Bit UART mode	Other modes
0	Interrupt on every received character	Don't care
1	Interrupt only when RB8 = 1	

Receive control

0	Disables receiver
1	Enables receiver

Handshake ($\overline{\text{CTS}}$) control

0	Disable (Accepts data streams at all times)
1	Enable

Bit8 of a transmitted character

Figure 3.9.6 SIO Registers (1)

Serial Control Register 1

SC1CR (0209H)

	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 when read)			R/W	
Reset value	Undefined	0	0	0	0	0	0	0
Function	Bit8 of a received character	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error has occurred Overrun	Parity	Framing	0: SCLK ↑ 1: SCLK ↓	0: Baud rate generator 1: SCLK input

Input clock in I/O interface mode

0	Baud rate generator
1	SCLK input

Active edge for the SCLK input

0	Data is transmitted/received on the SCLK rising edge. ↑
1	Data is transmitted/received on the SCLK falling edge. ↓

Framing error flag
Parity error flag
Overrun error flag

These bits are cleared to 0 when read.

Parity generation

0	Disabled
1	Enabled

Parity type

0	Odd parity
1	Even parity

Bit8 of a received character

Note: All error flags are cleared to 0 when read. These bits should not be tested using a bit test instruction.

Figure 3.9.7 SIO Registers (2)

Baud Rate Generator Control Register 1

BR1CR (020BH)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
Reset value	0	0	0	0	0	0	0	0
Function	Must be written as "0".	N + (16 − K) /16 function 0: Disabled 1: Enabled	00: φT0 01: φT2 10: φT8 11: φT32		Clock divisor value N			

N + (16 − K)/16 function

0	Disabled
1	Enabled

Clock source for baud rate generator

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

Baud Rate Generator K Value Register 1

BR1ADD (020CH)

	7	6	5	4	3	2	1	0
Bit symbol					BR1K3	BR1K2	BR1K1	BR1K0
Read/Write					R/W			
Reset value					0	0	0	0
Function					Value of K in N + (16 − K)/16			

Clock divisor value for baud rate generator

	BR1CR.BR1ADDE = 1		BR1CR.BR1ADDE = 0
BR1CR. BR1S[3:0] / BR1ADD. BR1K[3:0]	0000 (N = 16) or 0001 (N = 1)	0010 (N = 2) : 1111 (N = 15)	0001(N = 1) (Only UART) : 1111 (N = 15) 0000 (N = 16)
0000	Don't use.	Don't use.	Divided by N

0001(K = 1)
:
1111(K = 15)
	Don't use.	Divided by
		$N + \frac{16-K}{16}$	

Note 1: The N + (16 − K)/16 clock division function can be used under the following condition:

N	UART mode	I/O mode
2 to 15	Allowed	Not allowed
1, 16	Not allowed	Not allowed

Note 2: To use the N + (16 − K)/16 clock division function, the value of K (K = 1 to 15) must be programmed in the BR1ADD.BR1K[3:0] field before setting BR1CR.BR1ADDE to 1.

Figure 3.9.8 SIO Registers (3)

SC1BUF (0208H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(for transmit)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(for receive)

Note: The SC1BUF register does not support read-modify-write operation.

Figure 3.9.9 Serial Transmit/Receive Buffer Register (SC1BUF)

Serial Mode Control Register 1

SC1MOD1 (020DH)

	7	6	5	4	3	2	1	0
Bit symbol	I2S1	FDPX1						
Read/Write	R/W	R/W						
Reset value	0	0						
Function	SIO operation in IDLE2 mode 0: OFF 1: ON	Synchronous 0: Half duplex 1: Full duplex						

Figure 3.9.10 SIO Registers (4)

3.9.4 Operating Modes

(1) Mode 0 (I/O interface mode)

Mode 0 is used to increase the number of input/output pins. In this mode, the TMP91CW28 transmits or receives data to and from an external device, such as a shift register.

Mode 0 utilizes a synchronization clock (SCLK), which can be configured for either output mode in which the SCLK clock is driven out from the TMP91CW28 or input mode in which the SCLK clock is supplied externally.

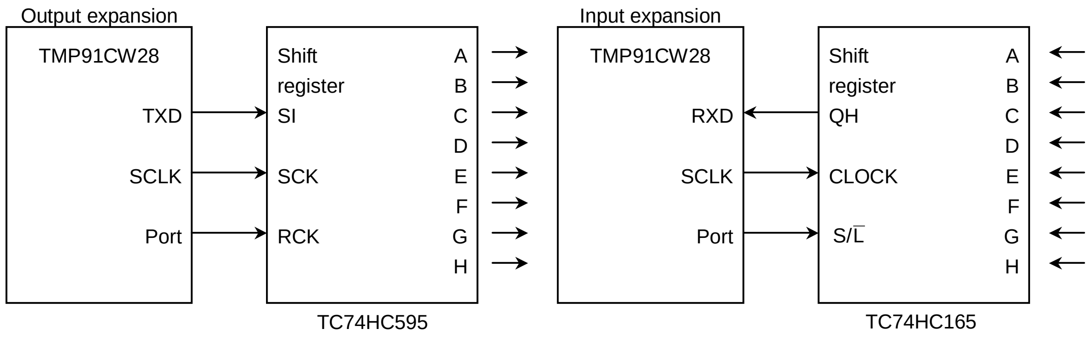


Figure 3.9.11 Example Connection in SCLK Output Mode

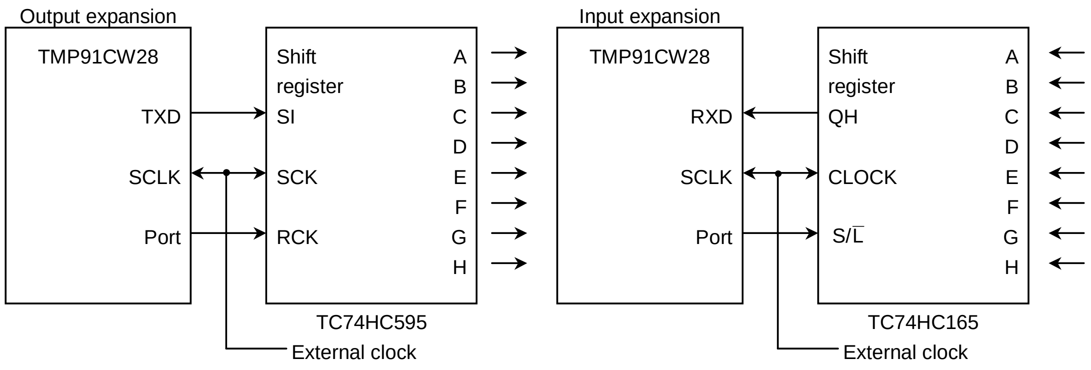


Figure 3.9.12 Example Connection in SCLK Input Mode

a. Transmit operations

In SCLK output mode, each time the CPU writes a character to the transmit buffer, the eight bits of the character is shifted out on the TXD pin, and the synchronization clock is driven out from the SCLK pin. When all the bits have been shifted out, the INTES1.ITX1C bit is set and the transmit-done interrupt (INTTX) is generated.

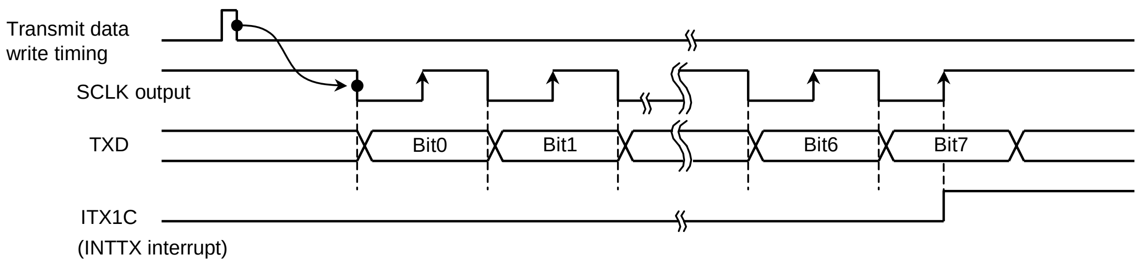


Figure 3.9.13 Transmit Operation in I/O Interface Mode (SCLK output mode)

In SCLK input mode, the CPU must write a character to the transmit buffer before the SCLK input is activated. The 8 bits of a character in the transmit buffer are shifted out on the TXD pin, synchronous to the programmed edge of the SCLK input. When all the bits have been shifted out, the INTES1.ITX1C bit is set and the transmit-done interrupt (INTTX) is generated.

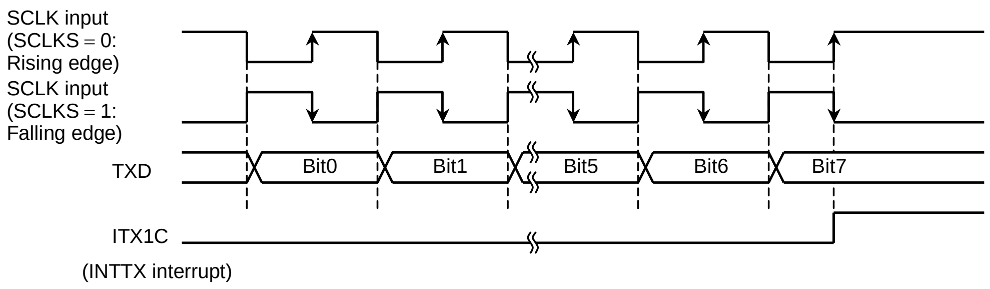

SCLK input (SCLKS = 0: Rising edge)
SCLK input (SCLKS = 1: Falling edge)
TXD
Bit0
Bit1
Bit5
Bit6
Bit7
ITX1C
(INTTX interrupt)

Figure 3.9.14 Transmit Operation in I/O Interface Mode (SCLK input mode)

b. Receive operations

In SCLK output mode, each time the CPU picks up the character in receive buffer 2, clearing the receive-done interrupt flag (INTES1.IRX1C), the synchronization clock is driven out from the SCLK pin to shift the next character into receive buffer 1. When a whole 8-bit character has been loaded into receive buffer 1, it is transferred to receive buffer 2 (SC1BUF), and the INTES1.IRX1C flag is set to 1 again, generating the INTRX interrupt.

The SCLK output is initiated by setting the SC1MOD0.RXE bit to 1.

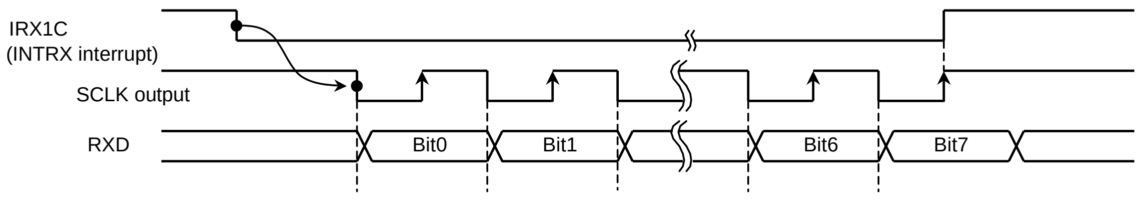


Figure 3.9.15 Receive Operation in I/O Interface Mode (SCLK output mode)

In SCLK input mode, the CPU must pick up the character in the receive buffer 2, clearing the receive-done interrupt flag (INTES1.IRX1C), before the SCLK input is activated to shift the next character into receive buffer 1. When a whole 8-bit character has been loaded into receive buffer 1, it is transferred to receive buffer 2 (SC1BUF), and the INTES1.IRX1C flag is set to 1 again, generating the INTRX interrupt.

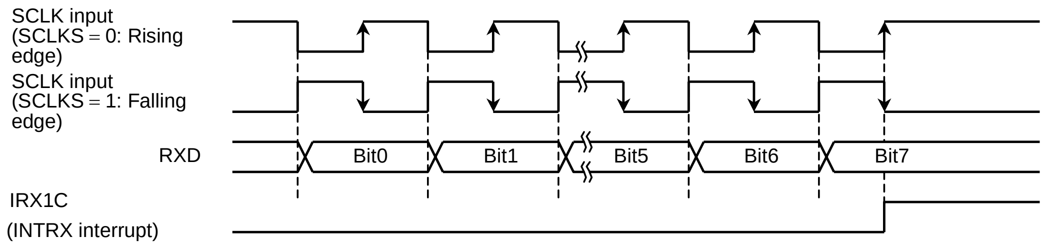


Figure 3.9.16 Receive Operation in I/O Interface Mode (SCLK input mode)

Note: Regardless of whether SCLK is in input mode or output mode, the receiver must be enabled by setting the SC1MOD0.RXE bit to 1 in order to perform receive operations.

c. Full-duplex transmit/receive operations

To perform full-duplex transmit/receive operations, the receive interrupt priority level must be set to 0, with the transmit interrupt priority level set to an appropriate value (1 to 6).

In the transmit interrupt handling routine, receive operation must be performed before loading the transmit buffer with a character, as shown below.

Example: SCLK output mode
Transfer rate: 9600 bps
fc = 9.8304 MHz

High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

Settings in the main routine

	7	6	5	4	3	2	1	0	
INTES1	X	0	0	1	X	0	0	0	Sets a transmit interrupt priority level and disables receive interrupts.
P9CR	–	–	–	0	1	–	–	–	Configures the P93 pin as TXD and the P94 pin as RXD.
P9FC	–	–	–	–	1	–	–	–	
SC1MOD0	–	–	0	–	0	0	–	–	Selects I/O interface mode.
SC1MOD1	1	1	X	X	X	X	X	X	Selects full-duplex operation.
SC1CR	–	–	–	–	–	–	0	0	Selects SCLK output mode, receiving at the rising edge and transmitting at the falling edge.
BR1CR	0	0	1	0	0	0	1	1	Sets the transfer rate to 9600 bps.
SC1MOD0	–	–	1	–	–	–	–	–	Enables receive operation.
SC1BUF	*	*	*	*	*	*	*	*	Loads the transmit buffer with a character.

Transmit interrupt handling routine

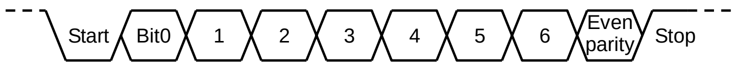
Acc SC1BUF									Reads received data.
SC1BUF	*	*	*	*	*	*	*	*	Loads the transmit buffer with a character.

X: Don't care, –: No change

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

Setting the SM[1:0] field in the SC1MOD0 to 01 puts the SIO in 7-bit UART mode. In this mode of operation, the parity bit can be added to the transmitted character, and the receiver can perform a parity check on incoming data. Parity can be enabled and disabled through the programming of the PE bit in the SC1CR. When PE = 1, the SC1CR.EVEN bit selects even or odd parity.

Example: Transmitting 7-bit UART characters with an even-parity bit



← Goes out first (Transfer rate = 2400 bps at fc = 9.8304 MHz)

Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: System clock

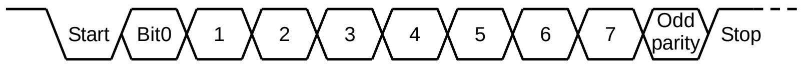
	7	6	5	4	3	2	1	0	
P9CR ←	–	–	–	–	1	–	–	–	Configures the P93 pin as TXD.
P9FC ←	–	–	–	–	1	–	–	–	
SC1MOD0 ←	–	–	–	–	0	1	0	1	Selects 7-bit UART mode.
SC1CR ←	X	1	1	X	X	X	–	–	Selects even parity.
BR1CR ←	0	0	1	0	0	1	0	0	Sets the transfer rate to 2400 bps.
INTES1 ←	X	1	0	0	–	–	–	–	Enables the INTTX interrupt and sets its priority level to 4.
SC1BUF ←	*	*	*	*	*	*	*	*	Loads the transmit buffer with a character.

X: Don't care, –: No change

(3) Mode 2 (8-bit UART mode)

Setting the SM[1:0] field in the SC1MOD0 to 10 puts the SIO in 8-bit UART mode. In this mode of operation, the parity bit can be added to the transmitted character, and the receiver can perform a parity check on incoming data. Parity can be enabled and disabled through the programming of the PE bit in the SC1CR. When PE = 1, the SC1CR.EVEN bit selects even or odd parity.

Example: Transmitting 8-bit UART characters with an odd-parity bit



← Goes out first (Transfer rate = 9600 bps at fc = 9.8304 MHz)

Clocking conditions: High-speed clock gear: × 1 (fc)
Prescaler clock: f_{FPH}

Settings in the main routine

	7 6 5 4 3 2 1 0	
P9CR	← – – – 0 – – – –	Configures P94 (RXD) to be an input.
SC1MOD0	← – – 1 – 1 0 0 1	Selects 8-bit UART mode and enables the receiver.
SC1CR	← X 0 1 X X X – –	Selects odd parity.
BR1CR	← 0 0 0 1 0 1 0 0	Sets the transfer rate to 9600 bps.
INTES1	← X – – – X 1 0 0	Enables the INTRX interrupt and sets its priority level to 4.

Example of interrupt routine processing.

```
Acc ← SC1CR AND 00011100     } Checks for errors.
if Acc ≠ 0 then ERROR        }
Acc ← SC1BUF                   Reads received data.
```

X: Don't care, –: No change

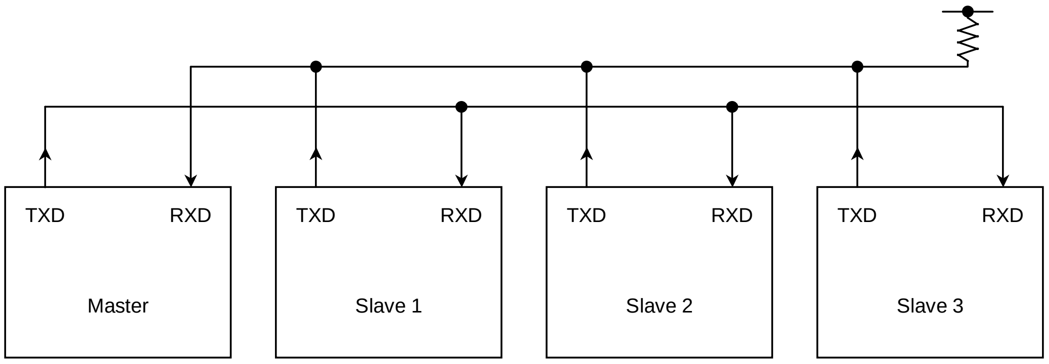
(4) Mode 3 (9-bit UART mode)

Setting the SM[1:0] field in the SC1MOD0 to 11 puts the SIO in 9-bit UART mode. In this mode, a parity bit cannot be used.

For transmit operations, the most-significant bit (9th bit) is stored in the TB8 bit in the SC1MOD0. For receive operations, the most-significant bit is stored in the RB8 bit in the SC1CR. Reads and writes of the transmit/receive character must be done with the most-significant bit first, followed by the SC1BUF.

Wakeup feature

In 9-bit UART mode, the receiver wakeup feature allows the slave station in a multidrop system to wakeup whenever an address character is received. Setting the SC1MOD0.WU bit enables the wakeup feature. When the SC1CR.RB8 bit has received an address/data flag bit set to 1, the receiver generates the INTRX interrupt.

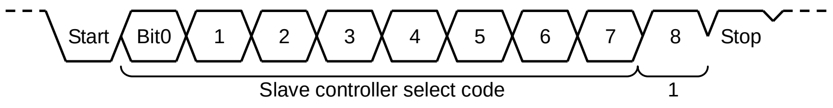


Note: The slave controller's TXD pin must be configured as an open-drain output by programming the ODE register.

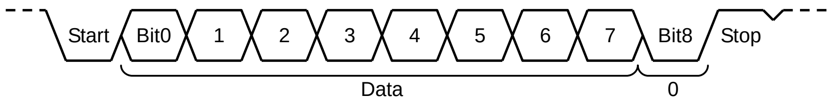
Figure 3.9.17 Serial Link Using the Wakeup Function

Protocol

1. Put all the master and slave controllers in 9-bit UART mode.
2. Enables the receiver in each slave controller by setting the SC1MOD0.WU bit to 1.
3. The master controller transmits an address character (e.g., select code) that identifies a slave controller. The address character has the most-significant bit (Bit8) set to 1.

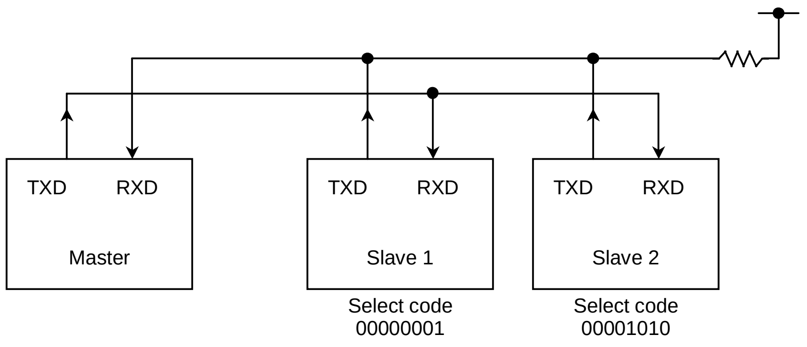


4. Each slave controller compares the received address to its station address and clears the WU bit if they match.
5. The master controller transmits data characters or block of data to the selected slave controller (with SC1MOD0.WU bit cleared). Data characters have the most-significant bit (Bit8) cleared to 0.



6. Slave controllers not addressed continue to monitor the data stream, but discard any characters with the most-significant bit (RB8) cleared, and thus does not generate receive-done interrupts (INTRX). The addressed slave controller with its WU bit cleared can transmit data to the master controller to notify that it has successfully received the message.

Example: Connecting a master station with two slave stations through a serial link using the f_{SYS} clock as a serial clock



- Master controller settings

Main routine

		7	6	5	4	3	2	1	0	
P9CR	←	–	–	–	0	1	–	–	–	Configures the P93 pin as TXD and the P94 pin as RXD.
P9FC	←	–	–	–	X	1	–	–	–	
INTES1	←	X	1	0	0	X	1	0	1	Enables INTTX and sets its interrupt level to 4. Enables INTRX and sets its interrupt level to 5.
SC1MOD0	←	1	0	1	0	1	1	1	0	Selects 9-bit UART mode and selects f_{SYS} as a serial clock.
SC1BUF	←	0	0	0	0	0	0	0	1	Loads the select code for slave 1.

Interrupt routine (INTTX)

SC1MOD0	←	0	–	–	–	–	–	–	–	Clears the TB8 bit to 0.
SC1BUF	←	*	*	*	*	*	*	*	*	Loads the transmit data.

- Slave controller settings

Main routine

		7	6	5	4	3	2	1	0	
P9CR	←	–	–	–	0	1	–	–	–	Configures the P93 pin as TXD (Open-drain output) and the P94 pin as RXD.
P9FC	←	–	–	–	X	1	–	–	–	
ODE	←	X	X	X	X	X	X	1	–	
INTES1	←	X	1	0	1	X	1	1	0	Enables INTTX and INTRX.
SC1MOD0	←	0	0	1	1	1	1	1	0	Selects 9-bit UART mode, selects f_{SYS} as the serial clock and sets the WU bit to 1.

Interrupt routine (INTRX)

Acc ← SC1BUF
if Acc = Select code
Then SC1MOD0 ← – – – 0 – – – – – WU = Clears the WU bit to 0.

3.10 Serial Bus Interface (SBI)

The TMP91CW28 serial bus interface (SBI) contains two channels named SBI0 and SBI1. Each channel has the following two operating modes:

- I^2C bus mode (with multi-master capability)
- Clock-synchronous 8-bit SIO mode

In I^2C bus mode, the SBI0 is connected to external devices via the P61 (SDA0) and P62 (SCL0) pins and the SBI1 via the P91 (SDA1) and P92 (SCL1) pins. In clock-synchronous 8-bit SIO mode, the SBI0 is connected to external devices via the P60 (SCK0), P61 (SO0) and P62 (SI0) pins and the SBI1 via the P90 (SCK1), P91 (SO1) and P92 (SI1) pins.

Each SBI channel is independently programmable, and functionally equivalent. In the following sections, any references to the SBI0 also apply to the SBI1.

The following table shows the programming required to put the SBI0 in each operating mode.

	ODE.ODE62 thru ODE.ODE61	P6CR.P62C thru P6CR.P60C	P6FC.P62F thru P6FC.P60F
I^2C bus mode	11	11X	11X
Clock-synchronous 8-bit SIO mode	XX	011 010	X11

The following table shows the programming required to put the SBI1 in each operating mode.

	ODE.ODE92 thru ODE.ODE91	P9CR.P92C thru P9CR.P90C	P9FC.P92F thru P9FC.P90F
I^2C bus mode	11	11X	11X
Clock-synchronous 8-bit SIO mode	XX	011 010	X11

X: Don't care

3.10.1 Block Diagram

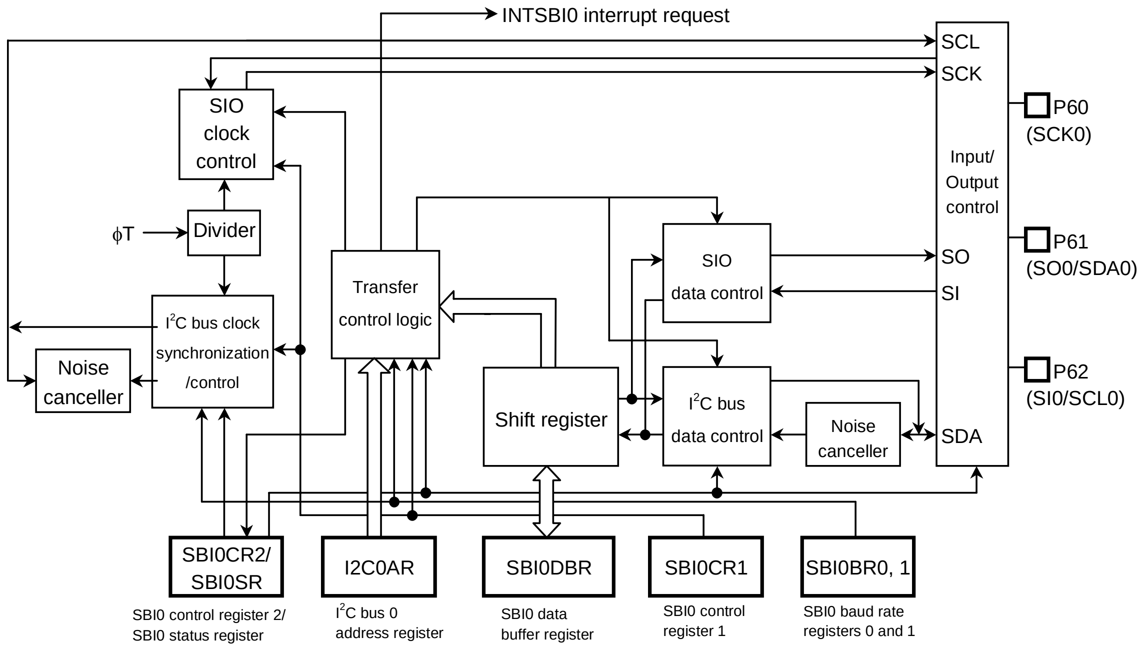


Figure 3.10.1 SBI0 Block Diagram

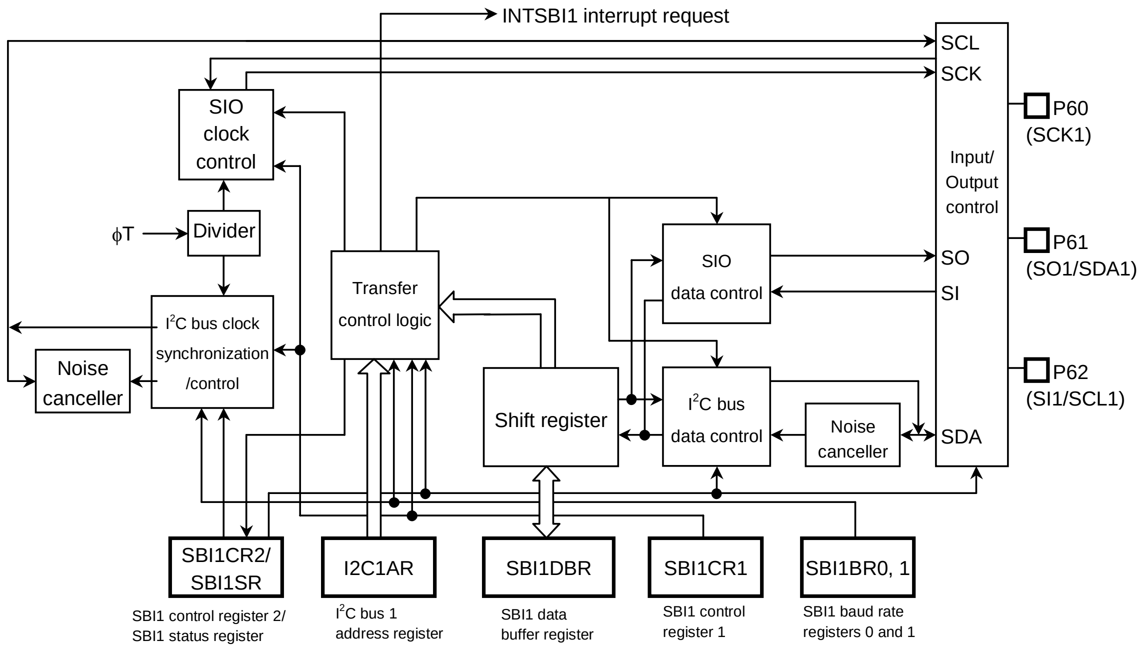


Figure 3.10.2 SBI1 Block Diagram

3.10.2 Registers

A listing of the registers used to control the SBI0 and SBI1 follows:

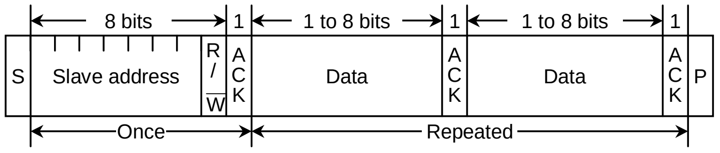
- Serial bus interface control register 1 (SBI0CR1, SBI1CR1)
- Serial bus interface control register 2 (SBI0CR2, SBI1CR2)
- Serial bus interface data buffer register (SBI0DBR, SBI1DBR)
- I^{2}C bus address register (I2C0AR, I2C1AR)
- Serial bus interface status register (SBI0SR, SBI1SR)
- Serial bus interface baud rate register 0 (SBI0BR0, SBI1BR0)
- Serial bus interface baud rate register 1 (SBI0BR1, SBI1BR1)

The functions of these registers vary, depending on the mode in which the SBI channel is operating. For a detailed description of the registers, refer to section 3.10.4, "Description of the Registers Used in I^{2}C Bus Mode", and section 3.10.7, "Description of Registers Used in Clock-synchronous 8-Bit SIO Mode".

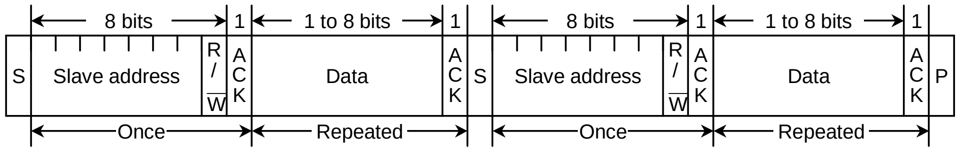
3.10.3 I^{2}C Bus Mode Data Formats

Figure 3.10.3 shows the serial bus interface data formats used in I^{2}C bus mode.

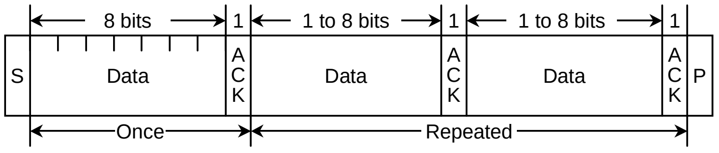
(a) Addressing format



(b) Addressing format (with repeated START condition)



(c) Free data format (Master transmitter to slave-receiver)



S:	START condition
R/$\overline{W}$:	Direction bit
ACK:	Acknowledge bit
P:	STOP condition

Figure 3.10.3 I^{2}C Bus Mode Data Formats

3.10.4 Description of the Registers Used in I^{2}C Bus Mode

This section provides a summary of the registers which control I^{2}C bus operation and provide I^{2}C bus status information for bus access/monitoring.

Serial Bus Interface Control Register 1

SBI0CR1 (0240H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
Read/Write	W			R/W		W		R/W
Reset value	0	0	0	0		0	0	0/1 (Note 3)
Function	Number of bits per transfer (Note 1)			ACK clock pulse 0: No ACK 1: ACK		Internal SCL output clock frequency (Note 2)/Software reset monitor		

On writes: SCK[2:0] = Internal SCL output clock frequency

000	n = 5	− kHz
001	n = 6	− kHz
010	n = 7	− kHz
011	n = 8	60.6 kHz
100	n = 9	30.8 kHz
101	n = 10	15.5 kHz
110	n = 11	7.78 kHz
111	(Reserved)	(Reserved)

Assumptions:
System clock: fc
Clock gear: fc/1
fc = 16 MHz (Output to the SCL pin)

$$\text{Frequency} = \frac{fc}{2^n + 8} \ [\text{Hz}]$$

On reads: SWRMON = Software reset monitor

0	Software reset operation is in progress.
1	Initial Data

Acknowledgement clock generation

0	No acknowledgement clock pulse is generated.
1	An acknowledgement clock pulse is generated.

Number of bits per transfer

BC[2:0]	ACK = 0		ACK = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Note 1: Clear the BC[2:0] field to 000 before switching the operating mode to “Clock-synchronous” 8-bit SIO mode.

Note 2: For details on the SCL bus clock frequency, refer to section 3.10.5 (3) “Serial clock”.

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

Note 4: This I^{2}C bus circuit does not support high-speed mode. It supports standard mode only.

Figure 3.10.4 I^{2}C Bus Mode Registers (1) (SBI0CR1 for the SBI0)

Serial Bus Interface Control Register 1

		7	6	5	4	3	2	1	0
SBI1CR1 (0248H)	Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
	Read/Write	W			R/W		W		R/W
	Reset value	0	0	0	0		0	0	0/1 (Note 3)
Read-modify-write operation is not supported.	Function	Number of bits per transfer (Note 1)			ACK clock pulse 0: No ACK 1: ACK		Internal SCL output clock frequency (Note 2)/ Software reset monitor		

On writes: SCK[2:0] = Internal SCL output clock frequency

000	n = 5	– kHz
001	n = 6	– kHz
010	n = 7	– kHz
011	n = 8	60.6 kHz
100	n = 9	30.8 kHz
101	n = 10	15.5 kHz
110	n = 11	7.78 kHz
111	(Reserved)	(Reserved)

Assumptions:
System clock: fc
Clock gear: fc/1
fc = 16 MHz (Output to the SCL pin)

$$\text{Frequency} = \frac{fc}{2^n + 8}\ [\text{Hz}]$$

On reads: SWRMON = Software reset monitor

0	Software reset operation is in progress.
1	Initial data

Acknowledgement clock generation

0	No acknowledgement clock pulse is generated.
1	An acknowledgement clock pulse is generated.

Number of bits per transfer

BC[2:0]	ACK = 0		ACK = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Note 1: Clear the BC[2:0] field to 000 before switching the operating mode to “Clock-synchronous” 8-bit SIO mode.

Note 2: For details on the SCL bus clock frequency, refer to section 3.10.5 (3) “Serial clock”.

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

Note 4: This I^{2}C bus circuit does not support high-speed mode. It supports standard mode only.

Figure 3.10.5 I^{2}C Bus Mode Registers (2) (SBI1CR1 for the SBI1)

Serial Bus Interface Control Register 2

SBI0CR2 (0243H)		7	6	5	4	3	2	1	0
	Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
	Read/Write	W				W (Note 1)		W (Note 1)	
	Reset value	0	0	0	1	0	0	0	0
Read-modify-write operation is not supported.	Function	Master/ slave	Transmit/ receive	START/ STOP condition generation	INTSBI0 interrupt clear	Operating mode (Note 2) 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)		Software reset A write of 10 followed by a write of "01".	

Operating mode (Note 2)

00	Port mode (Serial bus interface output disabled)
01	Clock-synchronous 8-bit SIO mode
10	I^2C bus mode
11	(Reserved)

INTSBI0 interrupt clear

0	–
1	Interrupt clear

START/STOP condition generation

0	STOP condition
1	START condition

Transmit/receive

0	Receive
1	Transmit

Master/slave

0	Slave
1	Master

Note 1: Reading this register causes it to function as a status register (SBI0SR).

Note 2: Ensure that the bus is free before switching the operating mode to Port mode.
Ensure that the port is at logic high before switching from Port mode to I^2C bus or SIO mode.

Figure 3.10.6 I^2C Bus Mode Registers (3) (SBI0CR2 for the SBI0)

Serial Bus Interface Control Register 2

SBI0CR2 (024BH)		7	6	5	4	3	2	1	0
	Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
	Read/Write	W				W (Note 1)		W (Note 1)	
	Reset value	0	0	0	1	0	0	0	0
Read-modify-write operation is not supported.	Function	Master/ slave	Transmit/ receive	START/ STOP condition generation	INTSBI1 interrupt clear	Operating mode (Note 2) 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		Software reset A write of 10 followed by a write of "01".	

Operating mode (Note 2)

00	Port mode (Serial bus interface output disabled)
01	Clock-synchronous 8-bit SIO mode
10	I^{2}C bus mode
11	(Reserved)

INTSBI1 interrupt clear

0	−
1	Interrupt clear

START/STOP condition generation

0	STOP condition
1	START condition

Transmit/receive

0	Receive
1	Transmit

Master/slave

0	Slave
1	Master

Note 1: Reading this register causes it to function as a status register (SBI0SR).

Note 2: Ensure that the bus is free before switching the operating mode to Port mode.
Ensure that the port is at logic high before switching from Port mode to I^{2}C bus or SIO mode.

Figure 3.10.7 I^{2}C Bus Mode Registers (4) (SBI1CR2 for the SBI1)

Serial Bus Interface Status Register

SBI0SR
(0243H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
Read/Write	R							
Reset value	0	0	0	1	0	0	0	0
Function	Master/ slave	Transmit/ receive	I^2C bus status	INTSBI0 interrupt status	Arbitration lost 0: – 1: Detected	Addressed as slave 0: Undetected 1: Detected	General call 0: Undetected 1: Detected	Last received bit 0: 0 1: 1

Last received bit

0	The last bit received was “0”.
1	The last bit received was “1”.

General call

0	Undetected
1	The address on the bus matches the general-call address

Addressed as slave

0	Undetected
1	The address on the bus matches the slave address or general-call address

Arbitration lost

0	–
1	Arbitration was lost to another master.

INTSBI0 interrupt status

0	The interrupt is asserted.
1	The interrupt is not asserted.

I^2C bus status

0	Free
1	Busy

Transmit/receive

0	Receive
1	Transmit

Master/slave

0	Slave
1	Master

Note: Writing to this register causes it to function as a control register (SBI0CR2).

Figure 3.10.8 I^2C Bus Mode Registers (5) (SBI0SR for the SBI0)

Serial Bus Interface Status Register

		7	6	5	4	3	2	1	0
SBI1SR	Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
(024BH)	Read/Write	R							
	Reset value	0	0	0	1	0	0	0	0
Read-modify-write operation is not supported.	Function	Master/ slave	Transmit/ receive	I^{2}C bus status	INTSBI1 interrupt status	Arbitration lost 0: – 1: Detected	Addressed as slave 0: Undetected 1: Detected	General call 0: Undetected 1: Detected	Last received bit 0: 0 1: 1

Last received bit

0	The last bit received was “0”.
1	The last bit received was “1”.

General call

0	Undetected
1	The address on the bus matches the general-call address

Addressed as slave

0	Undetected
1	The address on the bus matches the slave address or general-call address

Arbitration lost

0	–
1	Arbitration was lost to another master.

INTSBI1 interrupt status

0	The interrupt is asserted.
1	The interrupt is not asserted.

I^{2}C bus status

0	Free
1	Busy

Transmit/receive

0	Receive
1	Transmit

Master/slave

0	Slave
1	Master

Note: Writing to this register causes it to function as a control register (SBI0CR2).

Figure 3.10.9 I^{2}C Bus Mode Registers (6) (SBI1SR for the SBI1)

Serial Bus Interface Baud Rate Register 0

SBI0BR0 (0244H)

Read-modify-write operation is not supported.

		7	6	5	4	3	2	1	0
SBI0BR0 (0244H)	Bit symbol	–	I2SBI0						
	Read/Write	W	R/W						
	Reset value	0	0						
	Function	Must be written as "0".	IDLE2 0: OFF 1: ON						

SBI ON/OFF in IDLE2 mode

0	OFF
1	ON

Serial Bus Interface Baud Rate Register 1

Read-modify-write operation is not supported.

		7	6	5	4	3	2	1	0
SBI0BR1 (0245H)	Bit symbol	P4EN	–						
	Read/Write	W	W						
	Reset value	0	0						
	Function	Internal clock 0: OFF 1: ON	Must be written as "0".						

Controls the internal baud rate generator

0	OFF
1	ON

Serial Bus Interface Data Buffer Register

Read-modify-write operation is not supported.

		7	6	5	4	3	2	1	0
SBI0DBR (0241H)	Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	Read/Write	R (receive)/W (transmit)							
	Reset value	Undefined							

Note 1: In transmitter mode, data must be written to this register, with bit7 being the most-significant bit (MSB).

Note 2: SBIDBR can't be read the written data. Therefore read-modify-write instruction (e.g., "BIT" instruction) is prohibited.

Note 3: Written data in SBI0DBR is cleared by INTSBI0 signal.

I^{2}C Bus Address Register

Read-modify-write operation is not supported.

		7	6	5	4	3	2	1	0
I2C0AR (0242H)	Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	When the SBI0 is addressed as a slave, this field specifies a 7-bit I^{2}C bus address to which the SBI0 responds.							Address recognition mode

Address recognition mode

0	Recognizes the slave address.
1	Does not recognize the slave address.

Figure 3.10.10 I^{2}C Bus Mode Registers (7) (SBI0BR0, SBI0BR1, SBI0DBR, and I2C0AR for the SBI0)

Serial Bus Interface Baud Rate Register 0

SBI1BR0 (024CH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI1						
Read/Write	W	R/W						
Reset value	0	0						
Function	Must be written as "0".	IDLE2 0: OFF 1: ON						

SBI ON/OFF in IDLE2 mode

0	OFF
1	ON

Serial Bus Interface Baud Rate Register 1

SBI1BR1 (024DH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	P4EN	–						
Read/Write	W	W						
Reset value	0	0						
Function	Internal clock 0: OFF 1: ON	Must be written as "0".						

Controls the internal baud rate generator

0	OFF
1	ON

Serial Bus Interface Data Buffer Register

SBI1DBR (0249H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Transmit)							
Reset value	Undefined							

Note 1: In transmitter mode, data must be written to this register, with bit7 being the most-significant bit (MSB).

Note 2: SBIDBR can't be read the written data. Therefore read-modify-write instruction (e.g., "BIT" instruction) is prohibited.

Note 3: Written data in SBI1DBR is cleared by INTSBI1 signal.

I^{2}C Bus Address Register

I2C1AR (024AH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	When the SBI1 is addressed as a slave, this field specifies a 7-bit I^{2}C bus address to which the SBI1 responds.							Address recognition mode

Address recognition mode

0	Recognizes the slave address.
1	Does not recognize the slave address.

Figure 3.10.11 I^{2}C Bus Mode Registers (8) (SBI1BR0, SBI1BR1, SBI1DBR, and I2C1AR for the SBI1)

3.10.5 I^2C Bus Mode Configuration

(1) Acknowledge mode

Set the SBI0CR1.ACK to 1 for operation in the acknowledge mode. The TMP91CW28 generates an additional clock pulse for an acknowledge signal when operating in master mode.

In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the low in order to generate the acknowledge signal.

Clear the SBI0CR1.ACK to 0 for operation in the non-acknowledge mode, the TMP91CW28 does not generate a clock pulse for the acknowledge signal when operating in the master mode.

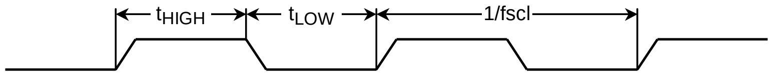
(2) Number of bits per transfer

The SBI0CR1.BC[2:0] field specifies the number of bits of the next data item to be transmitted or received. After a reset, this field is cleared to 000, causing a 7-bit slave address and the data direction ($R/\overline{W}$) bit to be transferred in a packet of 8 bits. At other times, the SBI0CR1.BC[2:0] field keeps a previously programmed value. Set the baud rate, which have been calculated according to the formula below to meet the specifications of the I^2C bus, such as the smallest pulse width of t_{LOW}.

(3) Serial clock

a. I^2C bus clock source

The SBI0CR1.SCK[2:0] field controls the maximum frequency of the SCL0 clock driven out on the SCL0 pin in master mode, as illustrated below. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I^2C bus, such as the smallest pulse width of t_{LOW}.



$t_{LOW} = 2^{n-1}/f_{SBI}$

$t_{HIGH} = 2^{n-1}/f_{SBI} + 8/f_{SBI}$

$$= \frac{f_{SBI}}{2^n + 8}$$

SBI0CR1.SCK[2:0]	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Note 1: $f_{SBI} = f_{FPH}$

Note 2: It's prohibited to use fc/16 prescaler clock when using SBI block. (I^2C bus and clock synchrounous)

Figure 3.10.12 I^2C Bus Clock Source

b. Clock synchronization

Clock synchronization is performed using the wired-AND connection of all I^{2}C bus components to the bus. If two or more masters try to transfer messages on the I^{2}C bus, the first to pull its clock line low wins the arbitration, overriding other masters producing a high on their clock lines.

Clock signals of two or more devices on the I^{2}C bus are synchronized to ensure correct data transfers. Figure 3.10.13 shows a depiction of the clock synchronization mechanism for the I^{2}C bus with two masters.

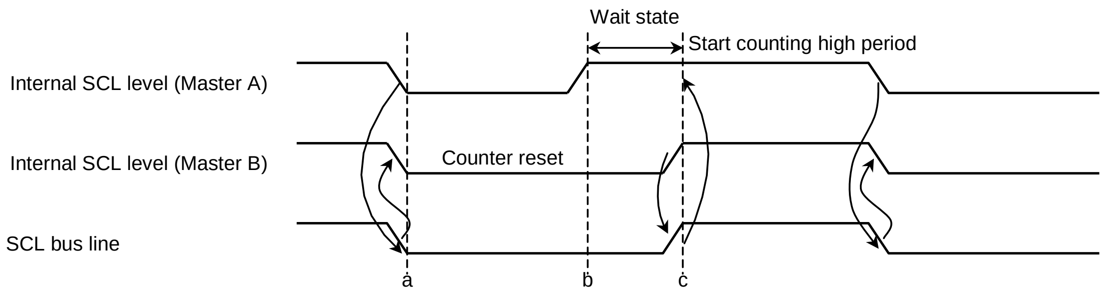

Wait state
Start counting high period
Internal SCL level (Master A)
Internal SCL level (Master B)
Counter reset
SCL bus line
a
b
c

Figure 3.10.13 Clock Synchronization Example

At point a, master A pulls its internal SCL0 level low, bringing the SCL bus line low. The high-to-low transition on the SCL0 bus line causes master B to reset its high-level counter and pull its internal SCL0 level low.

Master A completes its low period at point b. However, the low-to-high transition on its internal SCL0 level does not change the state of the SCL0 bus line if master B's internal SCL0 level is still within its low period. Therefore, master A enters a high wait state, where it does not start counting off its high period.

When master B has counted off its low period at point c, its internal SCL0 level goes high, releasing the SCL0 bus line (High). There will then be no difference between the internal SCL0 levels and the state of the SCL0 bus line, and both master A and master B start counting off their high periods.

This way, a synchronized SCL0 clock is generated with its high period determined by the master with the shortest clock high period and its low period determined by the one with the longest clock low period.

(4) Slave addressing and address recognition mode

When the SBI0 is configured to operate as a slave, the SA[6:0] field in the I2C0AR must be loaded with the 7-bit I^{2}C bus address to which the SBI0 is to respond. The ALS bit must be cleared for the SBI0 to recognize the incoming slave address.

(5) Configuring the SBI0 as a master or a slave

Setting the SBI0CR2.MST bit configures the SBI as a master, and clearing it configures the SBI0 as a slave. This bit is cleared by hardware when a STOP condition has been detected and when arbitration for the I^{2}C bus has been lost.

(6) Configuring the SBI0 as a transmitter or a receiver

The SBI0CR2.TRX bit is set or cleared by hardware to configure the SBI0 as a transmitter or a receiver. As a slave, the SBI0 is put in either slave-receiver or slave-transmitter mode, depending on the value of the data direction ($R/\overline{W}$) bit transmitted by the master. When the SBI0 is addressed as a slave, the TRX bit reflects the value of the $R/\overline{W}$ bit. The TRX bit is set or cleared on the following occasions:

- when transferring data using addressing format
- when the received slave address matches the value in the I2C0AR
- when a general-call address is received; e.g., the eight bits following the START condition are all zeros.

As a master, the SBI0 is put in either master-transmitter or a master-receiver mode upon reception of an acknowledge from an addressed slave. The TRX bit changes to the opposite value of the $R/\overline{W}$ bit sent by the SBI0. If the SBI0 does not receive an acknowledge from a slave, the TRX bit retains the previous value.

The TRX bit is cleared by hardware when a STOP condition has been detected and when arbitration for the I²C bus has been lost.

(7) Generating START and STOP conditions

When the SBI0SR.BB bit is cleared, the bus is free. At this time, writing 1s to the MST, TRX, BB and PIN bits in the SBI0CR2 causes the SBI0 to generate a START condition on the bus and shift out slave address and direction bit. Before generating a START condition, the ACK bit must be set to 1.

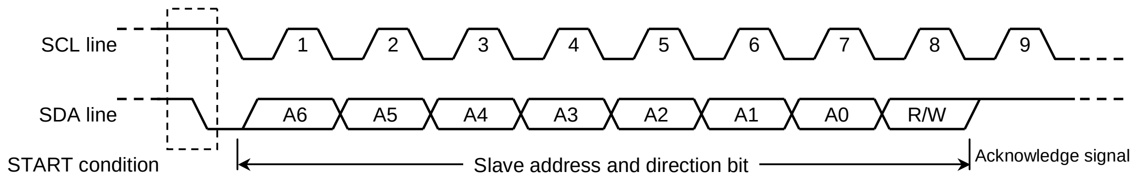


Figure 3.10.14 Generating a START Condition and a Slave Address

When the SBI0SR.BB bit is set, the bus is busy. When SBI0SR.BB = 1, writing 1s to the MST, TRX and PIN bits and a 0 to the BB bit causes the SBI0 to start a sequence for generating a STOP condition on the bus to abort the transfer. The MST, TRX, BB and PIN bits should not be altered until a STOP condition appears on the bus.

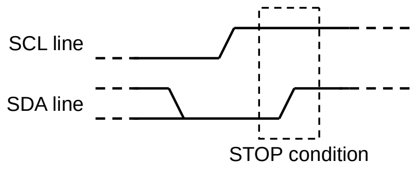


Figure 3.10.15 Generating a STOP Condition

The BB bit can be read to determine if the I²C bus is in use. The BB bit is set when a START condition is detected and cleared when a STOP condition is detected.

Some restrictions are imposed on the generation of a STOP condition when the SBI0 is a master. See section 3.10.6 (4) "Generating a STOP condition".

(8) Asserting and deasserting interrupt requests

When an SBI0 interrupt (INTSBI0) is generated, the pending interrupt not (PIN) bit in the SBI0CR2 is cleared to 0. While the PIN bit is 0, the SBI0 pulls the SCL0 line low.

After transmission or reception of one data word on the I²C bus, the PIN bit is automatically cleared. In transmitter mode, the PIN bit is subsequently set to 1 each time the SBI0DBR is written. In receiver mode, the PIN bit is set to 1 each time the SBI0DBR is read.

It takes a period of t_{LOW} for the SCL0 line to be released after the PIN bit is set.

In address recognition mode (ALS = 0), the PIN bit is cleared when the SBI0 is addressed as a slave and the received slave address matches the value in the I2C0AR or is all 0s (e.g., a general call). A write of 1 by software sets the PIN bit, but a write of 0 has no effect on this bit.

(9) SBI0 operating modes

The SBIM[1:0] field in the SBI0CR2 is used to select an operating mode of the SBI0. To configure the SBI0 for I²C bus mode, set the SBIM[1:0] field to 10 after confirming pin condition of serial bus interface to "H".

A switch to port mode should only be attempted when the bus is free.

(10) Lost-arbitration detection monitor

The I²C bus is a multi-master bus and has an arbitration procedure to ensure correct data transfers. A master may start a transfer only if the bus is free.

The I²C bus arbitration takes place on the SDA0 line.

Figure 3.10.16 shows the arbitration procedure for two masters. Up until point a, the internal data levels of master A and master B are the same. At point a master B's internal data level makes a low-to-high transition while master A's internal data level remains at logic low. However, the SDA0 bus line is held low because it is the wired-AND of the two data outputs. When the SCL0 bus clock goes high at point b, the addressed slave device reads the data transmitted by master A (e.g., winning master). Master B loses arbitration and switches off its data output stage, releasing its SDA0 line (High), so that it does not affect the data transfer initiated by the winning master. In case two competing masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.

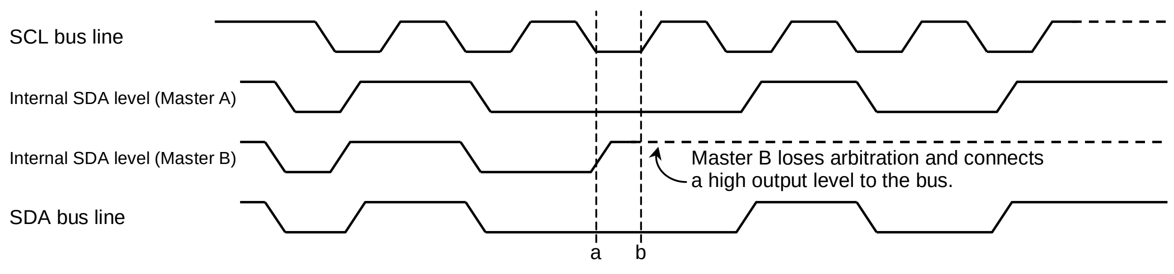

SCL bus line
Internal SDA level (Master A)
Internal SDA level (Master B)
SDA bus line
Master B loses arbitration and connects a high output level to the bus.
a
b

Figure 3.10.16 Arbitration Procedure of Two Masters

A master compares its internal data level to the actual level on the SDA0 line at the rising edge of the SCL0 clock. The master loses arbitration if there is a difference between these two values. The losing master sets the AL bit in the SBI0SR to 1, which causes the MST and TRX bits in the same register to be cleared. That is, the losing master switches to slave-receiver mode. Thus, clock output is stopped in data transfer after setting AL bit to 1.

The AL bit is subsequently cleared when data is written to or read from the SBI0DBR and when the SBI0CR2 is programmed with new parameters.

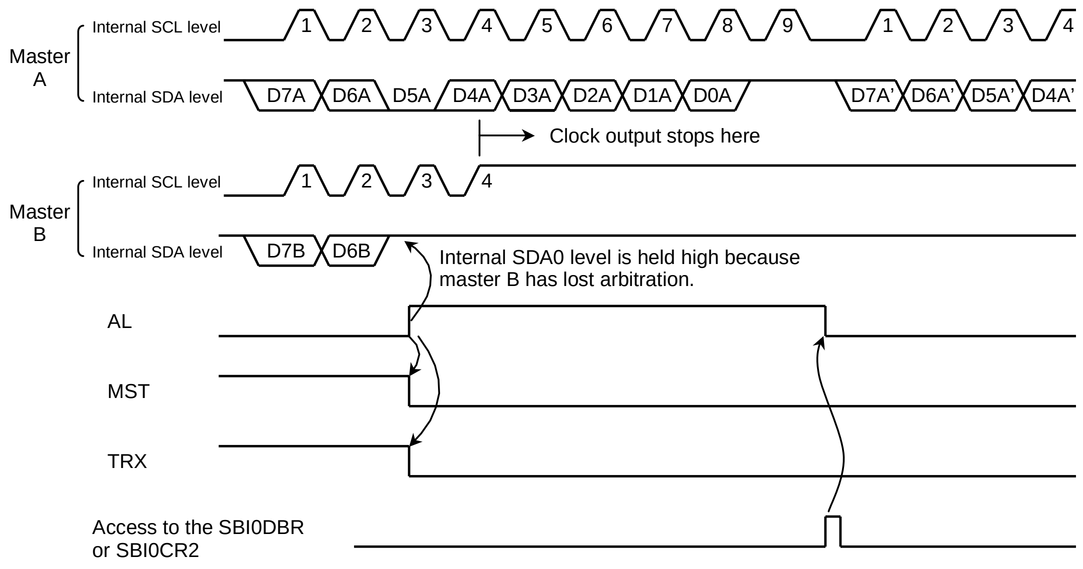


Figure 3.10.17 Master B Loses Arbitration (D7A = D7B, D6A = D6B)

(11) Slave address match monitor

When acting as a slave-receiver, the ALS bit in the I2C0AR determines whether the SBI0 recognizes the incoming slave address or not. In address recognition mode (e.g., ALS = 0), the addressed-as-slave (AAS) bit in the SBI0SR is set when an incoming address over the I^{2}C bus matches the value in the I2C0AR or when the general-call address has been received. When ALS = 1, the AAS bit is set when the first data word has been received. The AAS bit is cleared each time the SBI0DBR is read or written.

(12) General-call detection monitor

When acting as a slave-receiver, the AD0 bit in the SBI0SR is set when a general-call address has been received. The general-call address is detected when the eight bits following a START condition are all 0s. The AD0 bit is cleared when a START or STOP condition is detected on the bus.

(13) Last received bit monitor

The LRB bit in the SBI0SR holds the value of the last bit received over the SDA0 line at the rising edge of the SCL0 clock. In acknowledge mode, reading this bit immediately after generation of the INTSBI0 interrupt returns the value of the ACK signal.

(14) Software reset

The SBI0 provides a software reset, which permits recovery from system lockups caused by external noise. A software reset is performed by a write of 10 followed by a write of 01 to the SWRST[1:0] field in the SBI0CR2. After a software reset, all control and status register bits (except SBI0CR2. SBIM[1:0]) are initialized to their reset values. And SBI0CR1.SWRMON is automatically set to 1 after the SBI circuit has been initialized.

(15) Serial bus interface data buffer register (SBI0DBR)

The SBI0DBR is a data buffer interfacing to the I^2C bus. All read and write operations to/from the I^2C bus are done via this register.

When the SBI0 is acting as a master, loading this register with a slave address and a data direction bit causes a START condition to be generated.

(16) I^2C bus address register (I2C0AR)

When the SBI0 is configured as a slave, the SA[6:0] field in the I2C0AR must be loaded with the 7-bit I^2C bus address to which the SBI0 is to respond.

If the ALS bit in the I2C0AR is cleared, the SBI0 recognizes a slave address transmitted by the master device, interpreting incoming frame structures as per addressing format. If the ALS bit is set, the SBI0 does not recognize a slave address and interprets all frame structures as per free data format.

(17) Baud rate register (SBI0BR1)

Before the I^2C bus can be used, the P4EN bit in the SBI0BR1 must be set to enable the SBI0 internal baud rate generation logic.

(18) IDLE2 setting register (SBI0BR0)

The I2SBI0 bit in the SBI0BR0 determines whether the SBI0 is shut down or not when the TMP91CW28 is put in IDLE2 standby mode. This register must be programmed before executing the HALT instruction.

3.10.6 Programming Sequences in I^{2}C Bus Mode

(1) SBI0 initialization

First, program the P4EN bit in the SBI0BR1, and the ACK and SCK[2:0] bits in the SBI0CR1. Set the SBI0BR1.P4EN bit to 1 to enable the internal baud rate generation logic. Write 0s to bits 7 to 5 and bit3 in the SBI0CR1.

Next, program the I2C0AR. The SA[6:0] field in the I2C0AR defines the chip's slave address, and the ALS bit (Bit0) selects an address recognition mode. (The ALS bit must be cleared when using the addressing format.)

Next, program the SBI0CR2 to initially configure the SBI0 in slave-receiver mode; e.g., clear the MST, TRX and BB bits to 0, set the PIN bit to 1 and set the SBIM[1:0] field to 10. Write 00 to the SWRST[1:0] field.

(2) Generating a START condition and a slave address

a. Master mode

In master mode, the following steps are required to generate a START condition and a slave address on the I^{2}C bus.

First, ensure that the bus is free (e.g., SBI0CR2.BB = 0).

Next, set the ACK bit in the SBI0CR1 to enable generation of acknowledge clock pulses. Then, load the SBI0DBR with a slave address and a data direction bit to be transmitted via the I^{2}C bus.

When BB = 0, writing 1s to the MST, TRX, BB and PIN bits in the SBI0CR2 causes a START condition to be generated on the bus. Following a START condition, the SBI0 generates SCL0 clock pulses nine times: the SBI0 shifts out the contents of the SBI0DBR with the first eight SCL0 clocks, and releases the SDA0 line during the last (e.g., 9th) SCL0 clock to receive an acknowledgement signal from the addressed slave.

The INTSBI0 interrupt request is generated on the falling edge of the ninth SCL0 clock pulse, and the PIN bit in the SBI0CR2 is cleared to 0. In master mode, the SBI0 holds the SCL0 line low while the PIN bit is 0. Upon interrupt, the TRX bit either remains set or is cleared according to the value of the transmitted direction bit, provided an acknowledgement signal has been returned from the slave.

b. Slave mode

In slave mode, the following steps are required to receive a START condition and a slave address via the I^{2}C bus.

Upon detection of a START condition, the SBI0 clocks in a 7-bit slave address and a data direction bit transmitted by the master during the first eight SCL0 clock pulses. If the received slave address matches its own address in the I2C0AR or is equal to the general-call address (00H), the SBI0 pulls the SDA0 line low during the last (e.g., 9th) SCL0 clock for acknowledgement.

The INTSBI0 interrupt request is generated on the falling edge of the 9th SCL0 clock pulse, and the PIN bit in the SBI0CR2 is cleared to 0. In slave mode, the SBI0 holds the SCL0 line low while the PIN bit is 0.

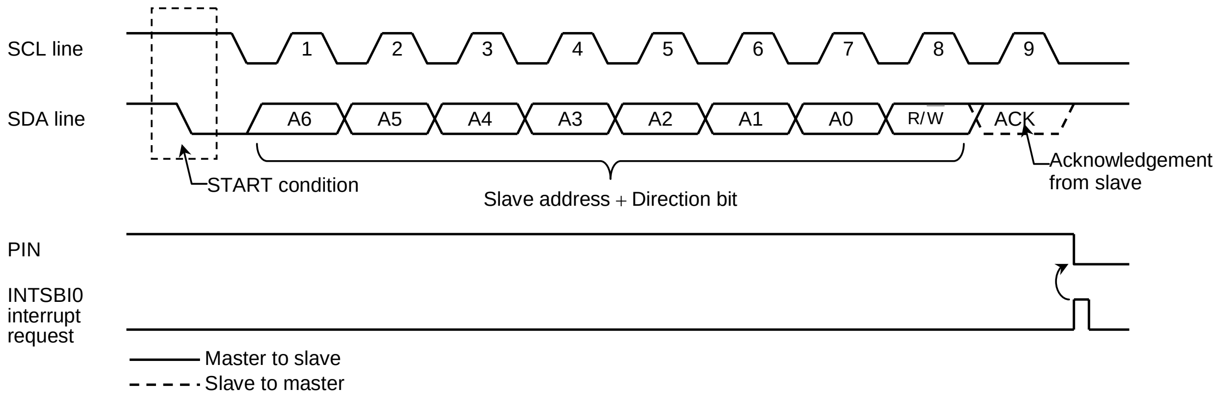


Figure 3.10.18 Generation of a START Condition and a Slave Address

(3) Transferring a data word

Each time a data word has been transmitted or received, the INTSBI0 interrupt is generated. It is the responsibility of the INTSBI0 interrupt service routine to test the MST bit in the SBI0CR2 to determine whether the SBI is in master or slave mode.

a. Master mode (SBI0CR2.MST = 1)

If the MST bit in the SBI0CR2 is set, then test the TRX bit in the same register to determine whether the SBI0 is in master-transmitter or master-receiver mode.

<u>Master-transmitter mode (SBI0CR2.TRX = 1)</u>

Test the LRB bit in the SBI0SR. If the LRB bit is set, that means the slave-receiver requires no further data to be sent from the master-transmitter. The master-transmitter must then generate a STOP condition as described later to stop transmission.

If the LRB bit is cleared, that means the slave-receiver requires further data. If the number of bits per transfer is 8, then write the transmit data into the SBI0DBR. When using other data length, program the BC[2:0] and ACK bits in the SBI0CR1, and then write the transmit data into the SBI0DBR. When the SBI0DBR is loaded, the PIN bit in the SBI0SR is set to 1, and the transmit data is shifted out from the SDA0 pin, clocked by the SCL0 clock. Once the transfer is complete, the INTSBI0 interrupt is generated, the PIN bit is cleared, and the SCL0 line is pulled low. To transmit further data, test the LRB bit again and repeat the above procedure.

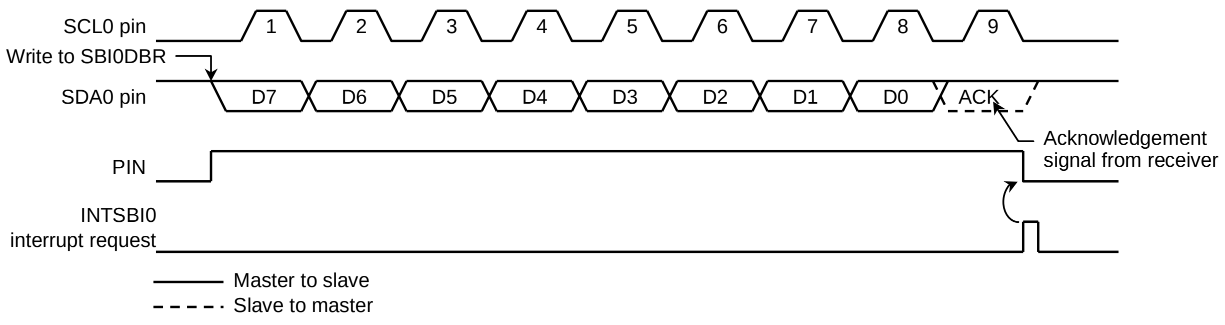


Figure 3.10.19 SBI0CR1.BC[2:0] = 000 and SBI0CR1.ACK = 1 (Master-transmitter mode)

Master-receiver mode (SBI0CR2.TRX = 0)

When using other than 8 bits data length, program the BC[2:0] and ACK bits in the SBI0CR1, and then read the SBI0DBR. (The first read of the SBI0DBR is a dummy read because data has not yet been received. A dummy read returns an undefined value.) Upon this read, the SCL0 line is released, the PIN bit in the SBI0SR is set. Serial clock pulse for transferring new 1 word of data is defined SCL and outputs "L" level from SDA pin with acknowledge timing.

Once the transfer is complete, the INTSBI0 interrupt is generated, the PIN bit is cleared, and the SCL0 line is pulled low. Each subsequent read from the SBI0DBR is accompanied by an SCL0 clock pulse for a data word and an acknowledgement signal.

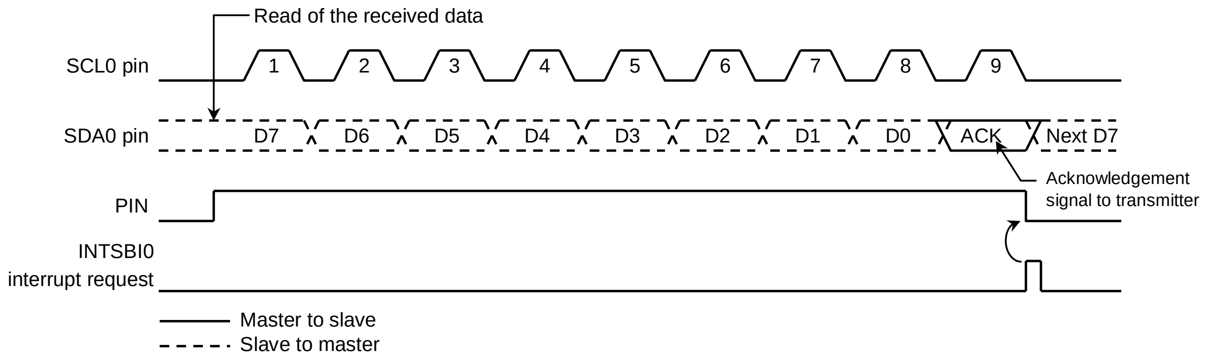

Read of the received data
SCL0 pin
1
2
3
4
5
6
7
8
9
SDA0 pin
D7
D6
D5
D4
D3
D2
D1
D0
ACK
Next D7
Acknowledgement signal to transmitter
PIN
INTSBI0 interrupt request
Master to slave
Slave to master

Figure 3.10.20 SBI0CR1.BC[2:0] = 000 and SBI0CR1.ACK = 1 (Master-receiver mode)

To prepare to terminate the data transfer, the master-receiver must clear the ACK bit in the SBI0CR1 immediately before the read of the second to last data word. This causes an acknowledge clock pulse not to be generated on the last data word.

When the transfer is complete, the INTSBI0 interrupt is generated. After interrupt processing, the INTSBI0 interrupt handler must set the BC[2:0] field in the SBI0CR1 to 001 and read the SBI0DBR, so that a clock is generated on the SCL0 line once. With the ACK bit cleared, the master-receiver holds the SDA0 line high, which signals the end of transfer to the slave-transmitter.

Then, the SBI0 generates the INTSBI0 interrupt again, whereupon the INTSBI0 interrupt service routine must generate a STOP condition to stop communication via the I^{2}C bus.

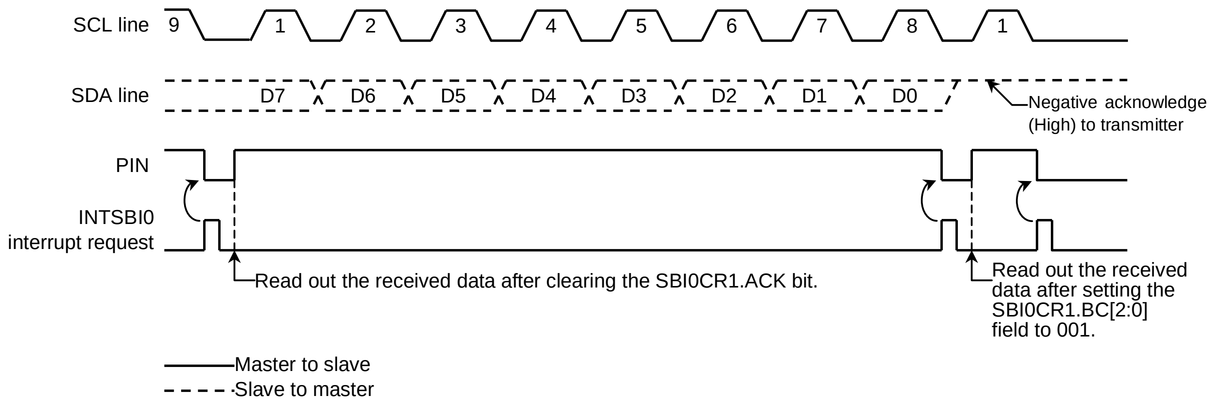

SCL line
9
1
2
3
4
5
6
7
8
1
SDA line
D7
D6
D5
D4
D3
D2
D1
D0
Negative acknowledge (High) to transmitter
PIN
INTSBI0 interrupt request
Read out the received data after clearing the SBI0CR1.ACK bit.
Read out the received data after setting the SBI0CR1.BC[2:0] field to 001.
Master to slave
Slave to master

Figure 3.10.21 Terminating Data Transmission in Master Receiver Mode

b. Slave mode (SBI0CR2.MST = 0)

Processing to be done in slave mode varies, depending on whether or not the SBI0 has switched over to slave mode as a result of lost arbitration.

If the MST bit in the SBI0CR2 is cleared, the SBI0 is in slave mode. In slave mode, the SBI0 generates the INTSBI0 interrupt on four occasions: 1) when the SBI0 has received any slave address; 2) when the SBI0 has received a general-call address; 3) when the received slave address matches its own address in the I2C0AR; and 4) when a data transfer has been completed in response to a general-call.

Also, if the SBI0, as a master, loses arbitration for the I^{2}C bus, it switches to slave mode. If arbitration is lost during a data transfer, SCL0 continues to be generated until the data word is complete, then the INTSBI0 interrupt is generated.

When the INTSBI0 interrupt occurs, the PIN bit in the SBI0SR is cleared, and the SCL0 line is pulled low. When the SBI0DBR is read or written or when the PIN bit is set back to 1, the SCL0 line is released after a period of t_{LOW}.

Test the AL, TRX, AAS and AD0 bits in the SBI0SR to determine the processing required, as summarized in Table 3.10.1.

Table 3.10.1 Processing in Slave Mode

TRX	AL	AAS	AD0	State	Processing
1	1	1	0	Arbitration was lost while the slave address was being transmitted, and the SBI0 received a slave address with the direction bit set transmitted by another master.	Set the SBI0CR1.BC[2:0] field to the number of bits in a data word and write the transmit data into the SBI0DBR.
	0	1	0	In slave-receiver mode, the SBI0 received a slave address with the direction bit set transmitted by the master.	
		0	0	In slave-transmitter mode, the SBI0 has completed a transmission of one data word.	Test the SBI0SR.LRB bit. If the LRB bit is set, that means the master-receiver does not require further data. Set the SBI0CR2.PIN bit to 1 and clear the TRX bit to 0 to release the bus. If the LRB bit is cleared, that means the master-receiver requires further data. Set the SBI0CR1.BC[2:0] field to the number of bits in the data word and write the transmit data to the SBI0DBR.
0	1	1	1/0	Arbitration was lost while a slave address was being transmitted, and received either a slave address with the direction bit cleared or a general-call address transmitted by another master.	Read the SBI0DBR (a dummy read) to set the SBI0CR2.PIN bit to 1, or write a 1 to this bit.
		0	0	Arbitration was lost while a slave address or a data word was being transmitted, and the transfer terminated.	
	0	1	1/0	In slave-receiver mode, the SBI0 received either a slave address with the direction bit cleared or a general-call address transmitted by the master.	
		0	1/0	In slave-receiver mode, the SBI0 has completed a reception of a data word.	Set the SBI0CR1.BC[2:0] field to the number of bits in the data word and read the received data from the SBI0DBR.

(4) Generating a STOP condition

When the SBI0SR.BB bit is set, setting the MST, TRX and PIN bits in the SBI0CR2 to 1 and clearing the BB bit in the same register causes the SBI0 to start a sequence for generating a STOP condition on the I^{2}C bus. Do not alter the contents of these bits until the STOP condition is present on the bus.

If another device is holding down the SCL0 bus line, the SBI0 waits until the SCL0 line is released (High) again; when SCL0 is high, the SBI0 drives the SDA0 pin high to generate a STOP condition.

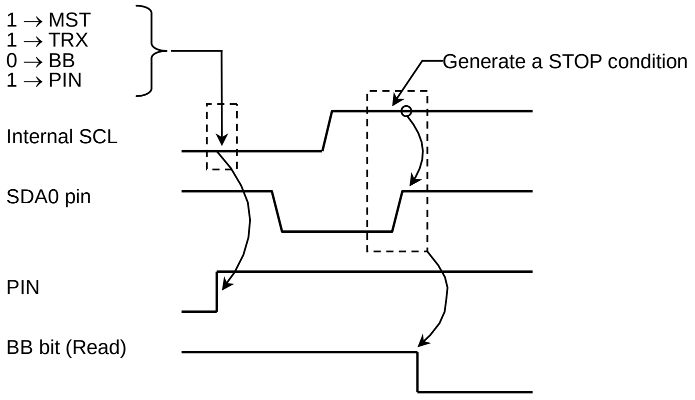


Figure 3.10.22 Generating a STOP Condition (Single master)

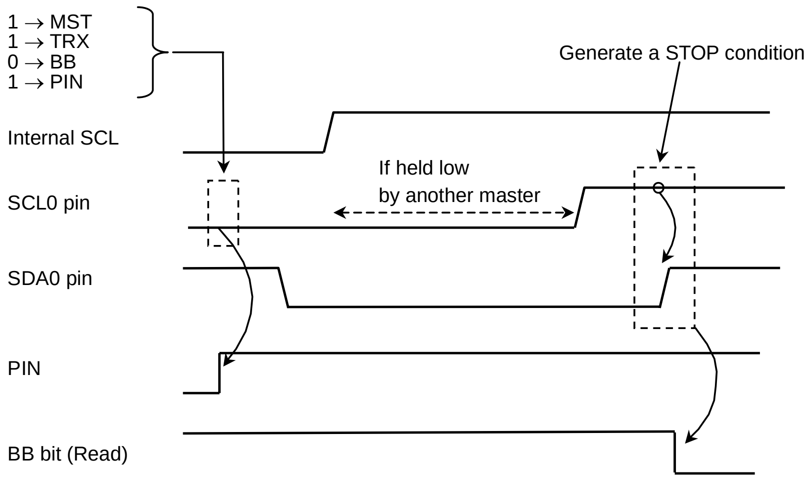


Figure 3.10.23 Generating a STOP Condition (Multi master)

(5) Repeated START condition

A data transfer is always terminated by a STOP condition. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition and address another slave or change the data direction without first generating a STOP condition. The following describes the steps required to generate a repeated START condition.

First, clear the MST, TRX and BB bits in the SBI0CR2 and set the PIN bit in the same register to release the bus. This causes the SDA0 pin to be held high and the SCL0 pin to be released. Because no STOP condition is generated on the bus, other devices think that the bus is busy.

Then, poll the SBI0SR.BB bit until it is cleared to ensure that the SCL0 pin is released. Next, poll the LRB bit until it is set to ensure that no other device is pulling the SCL0 bus line low. Once the bus is determined to be free this way, use the steps described in (2), above, to generate a START condition.

To satisfy the minimum setup time of the START condition, at least 4.7 μs wait period must be created by software after the bus becomes free.

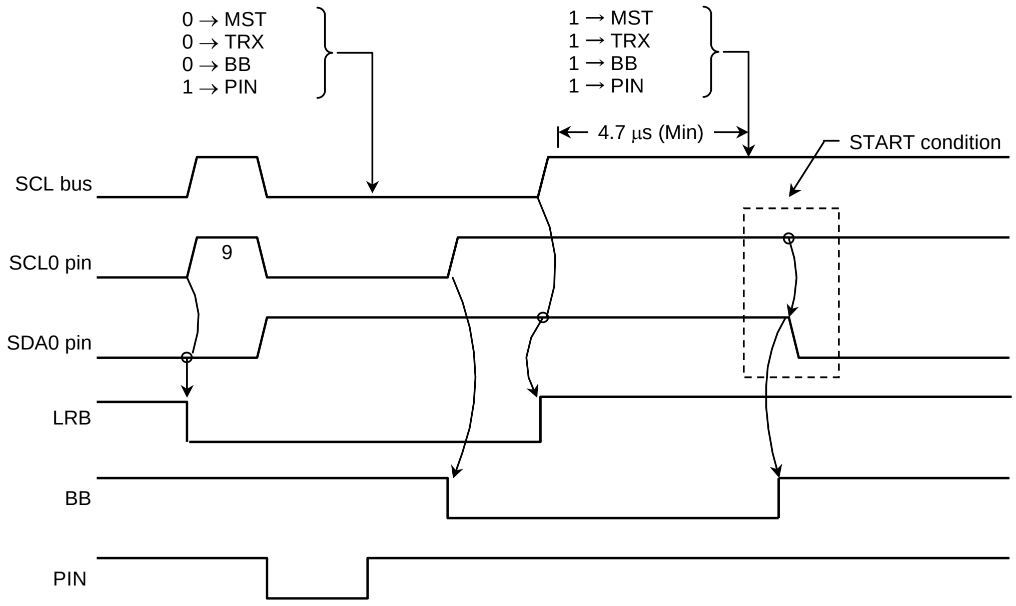


Figure 3.10.24 Repeated START Condition

3.10.7 Description of Registers Used in Clock-synchronous 8-Bit SIO Mode

This section provides a summary of the registers which control clock-synchronous 8-bit SIO operation and provides its status information for monitoring.

Serial Bus Interface Control Register 1

SBI0CR1 (0240H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
Read/Write	W					W		W
Reset value	0	0	0	0		0	0	0
Function	Start transfer 0: Stop 1: Start	Abort transfer 0: Continue 1: Abort	Transfer mode 00: Transmit mode 01: (Reserved) 10: Transmit/Receive mode 11: Receive mode			Serial clock frequency/ software reset monitor		

On writes: SCK[2:0] = Serial clock frequency

000	n = 4	1 MHz	
001	n = 5	500 kHz	
010	n = 6	250 kHz	
011	n = 7	125 kHz	
100	n = 8	62.5 kHz	
101	n = 9	31.25 kHz	
110	n = 10	15.625 kHz	
111	–	External clock (Input from the SCK pin)	

Assumptions
System clock: fc
Clock gear: fc/1
fc = 16 MHz (Output to the SCK pin)

$$\text{Frequency} = \frac{fc}{2^n} \text{ [Hz]}$$

Transfer mode

00	8-bit transmit mode
01	(Reserved)
10	8-bit transmit/receive mode
11	8-bit receive mode

Abort transfer

0	Continue
1	Abort (This bit is automatically cleared after transfer is aborted.)

Start transfer

0	Stop
1	Start

Note: Clear the SIOS bit and set the SIOINH bit before programming the transfer mode and serial clock frequency bits.

Serial Bus Interface Data Buffer Register

SBI0DBR (0241H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Transmit)							
Reset value	Undefined							

Figure 3.10.25 SIO Mode Registers (1) (SBI0CR1 and SBI0DBR for the SBI0)

Serial Bus Interface Control Register 1

SBI1CR1 (0248H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
Read/Write	W					W		W
Reset value	0	0	0	0		0	0	0
Function	Start transfer 0: Stop 1: Start	Abort transfer 0: Continue 1: Abort	Transfer mode 00: Transmit mode 01: (Reserved) 10: Transmit/Receive mode 11: Receive mode			Serial clock frequency/ software reset monitor		

On writes: SCK[2:0] = Serial clock frequency

000	n = 4	1 MHz	Assumptions System clock: fc Clock gear: fc/1 fc = 16 MHz (Output to the SCK pin) Frequency $= \frac{fc}{2^n}$ [Hz]
001	n = 5	500 kHz	
010	n = 6	250 kHz	
011	n = 7	125 kHz	
100	n = 8	62.5 kHz	
101	n = 9	31.25 kHz	
110	n = 10	15.625 kHz	
111	–	External clock (Input from the SCK pin)	

Transfer mode

00	8-bit transmit mode
01	(Reserved)
10	8-bit transmit/receive mode
11	8-bit receive mode

Abort transfer

0	Continue
1	Abort (This bit is automatically cleared after transfer is aborted.)

Start transfer

0	Stop
1	Start

Note: Clear the SIOS bit and set the SIOINH bit before programming the transfer mode and serial clock frequency bits.

Serial Bus Interface Data Buffer Register

SBI0DBR (0249H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Transmit)							
Reset value	Undefined							

Figure 3.10.26 SIO Mode Registers (2) (SBI1CR1 and SBI1DBR for the SBI1)

Serial Bus Interface Control Register 2

SBI0CR2 (0243H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol					SBIM1	SBIM0	–	–
Read/Write					W		W	W
Reset value					0	0	0	0
Function					Operating mode 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		(Note 2)	(Note 2)

Note 1: The BC[2:0] bits in the SBI0CR1 register must be cleared to 000 before selecting clock-synchronous 8-bit SIO mode.

Note 2: Please always write SBI0CR2[1:0] to 00.

Operating mode

00	Port mode (Serial bus interface output disabled)
01	Clock-synchronous 8-bit SIO mode
10	I^{2}C bus mode
11	(Reserved)

Serial Bus Interface Register

SBI0SR (0243H)

	7	6	5	4	3	2	1	0
Bit symbol					SIOF	SEF		
Read/Write					R			
Reset value					0	0		
Function					Serial transfer status	Shift operation status		

Serial transfer status monitor

0	Terminated
1	In progress

Shift operation status monitor

0	Terminated
1	In progress

Serial Bus Interface Baud Rate Register 0

SBI0BR0 (0244H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI0						
Read/Write	W	R/W						
Reset value	0	0						
Function	Must be written as “0”.	IDLE2 0: OFF 1: ON						

Operation in IDLE2 mode

0	OFF
1	ON

Serial Bus Interface Baud Rate Register 1

SBI0BR1 (0245H)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	P4EN	–						
Read/Write	W	W						
Reset value	0	0						
Function	Internal clock 0: OFF 1: ON	Must be written as “0”.						

Internal baud rate generator

0	OFF
1	ON

Figure 3.10.27 SIO Mode Registers (3) (SBI0CR2, SBI0SR, SBI0BR0, and SBI0BR1 for the SBI0)

Serial Bus Interface Control Register 2

SBI1CR2 (024BH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol					SBIM1	SBIM0	–	–
Read/Write					W		W	W
Reset value					0	0	0	0
Function					Operating mode 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		(Note 2)	(Note 2)

Note 1: The BC[2:0] bits in the SBI1CR1 register must be cleared to 000 before selecting clock-synchronous 8-bit SIO mode.

Note 2: Please always write SBI1CR2[1:0] to 00.

Operating mode

00	Port mode (Serial bus interface output disabled)
01	Clock-synchronous 8-bit SIO mode
10	I^{2}C bus mode
11	(Reserved)

Serial Bus Interface Register

SBI1SR (024BH)

	7	6	5	4	3	2	1	0
Bit symbol					SIOF	SEF		
Read/Write					R			
Reset value					0	0		
Function					Serial transfer status	Shift operation status		

Serial transfer status monitor

0	Terminated
1	In progress

Shift operation status monitor

0	Terminated
1	In progress

Serial Bus Interface Baud Rate Register 0

SBI1BR0 (024CH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI0						
Read/Write	W	R/W						
Reset value	0	0						
Function	Must be written as “0”.	IDLE2 0: OFF 1: ON						

Operation in IDLE2 mode

0	OFF
1	ON

Serial Bus Interface Baud Rate Register 1

SBI1BR1 (024DH)

Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	P4EN	–						
Read/Write	W	W						
Reset value	0	0						
Function	Internal clock 0: OFF 1: ON	Must be written as “0”.						

Internal baud rate generator

0	OFF
1	ON

Figure 3.10.28 SIO Mode Registers (4) (SBI1CR2, SBI1SR, SBI1BR0, and SBI1BR1 for the SBI1)

(1) Serial clock

a. Clock source

The clock source for SIO mode can be selected from internal and external clocks through the programming of the SCK[2:0] field in the SBI0CR1.

Internal clocks

One of the seven internal clocks can be used as a serial clock, which is driven onto the SCK0 pin. At the beginning of a transfer, the SCK0 clock will start out at logic high.

If software is slow and the reading of the received data or the writing of the transmit data cannot keep up with the serial clock rate, the SBI0 automatically inserts a wait period, as shown below. During this period, the serial clock is temporarily stopped to suspend a shift operation.

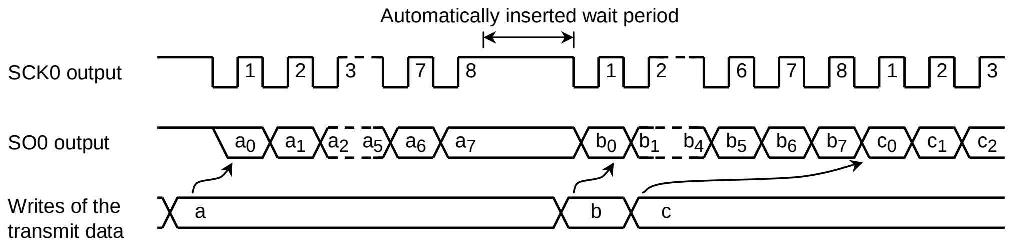
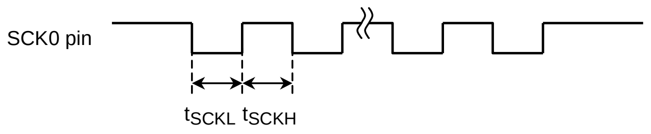

Automatically inserted wait period
SCK0 output
1 2 3 7 8 1 2 6 7 8 1 2 3
SO0 output
a_0 a_1 a_2 a_5 a_6 a_7 b_0 b_1 b_4 b_5 b_6 b_7 c_0 c_1 c_2
Writes of the transmit data
a
b
c

Figure 3.10.29 Automatic Wait Insertion

External clock (SBI0CR1.SCK[2:0] = 111)

If the SCK[2:0] field in the SBI0CR1 contains 111, the SBI0 uses an external clock supplied from the SCK0 pin as a serial clock. For proper shift operations, the clock high width and the clock low width must satisfy the following relationship, so that the maximum transfer frequency is 1 MHz (when fc = 16 MHz).



T_{SCKL}, t_{SCKH} > 8/fc

Figure 3.10.30 Maximum External Clock Frequency

b. Shift edge types

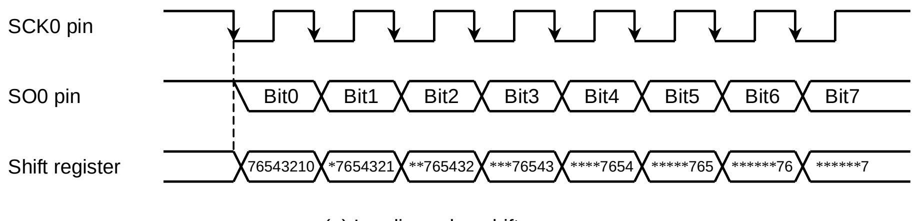
In transmit mode, leading-edge shift is used. In receive mode, trailing-edge shift is used.

Leading-edge shift

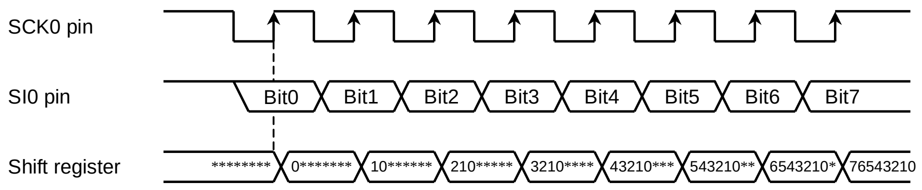
Every bit of SIO data is shifted by the leading edge of the serial clock (Falling edge of SCK0).

Trailing-edge shift

Every bit of SIO data is shifted by the trailing edge of the serial clock (Rising edge of SCK0).



(a) Leading-edge shift



(b) Trailing-edge shift

*: Don't care

Figure 3.10.31 Shift Edge Types

(2) Transfer modes

The SBI0 supports three SIO transfer modes: Receive mode, transmit mode and transmit/receive mode. The SIOM[1:0] field in the SBI0CR1 is used to select a transfer mode.

a. 8-bit transmit mode

Configure the SIO interface in transmit mode and write the transmit data into the SBI0DBR. Then setting the SIOS bit in the SBI0CR1 initiates a transmission. The contents of the SBI0DBR are moved to an internal shift register and then shifted out on the SO0 pin, with the least-significant bit (LSB) first, synchronous to the serial clock. Once the transmit data is transferred to the shift register, the SBI0DBR becomes empty, and the buffer-empty interrupt (INTSBI0) is generated.

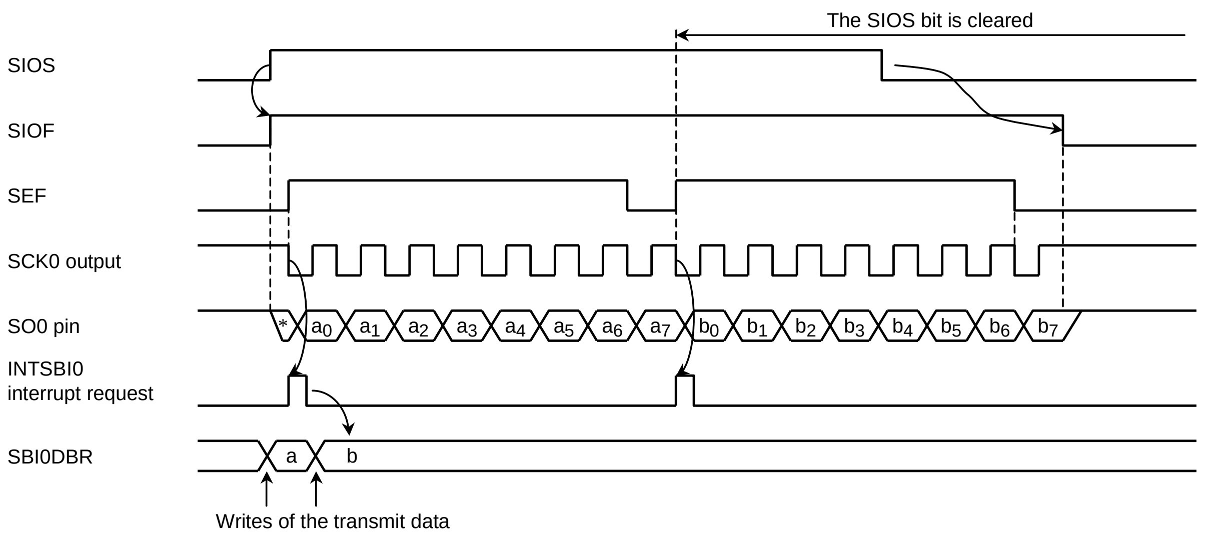
In internal clock mode, the SIO interface will be in wait state (SCK will stop) until the INTSBI0 interrupt service routine provides the next transmit data to the SBI0DBR. Once the SBI0DBR is loaded, the SIO interface will automatically get out of the wait state.

In external clock mode, the INTSBI0 interrupt service routine must provide the next transmit data to the SBI0DBR before the previous transmit data has been shifted out. Therefore, the data rate is a function of the maximum latency between when the INTSBI0 interrupt is generated and when the SBI0DBR is loaded by the interrupt service routine.

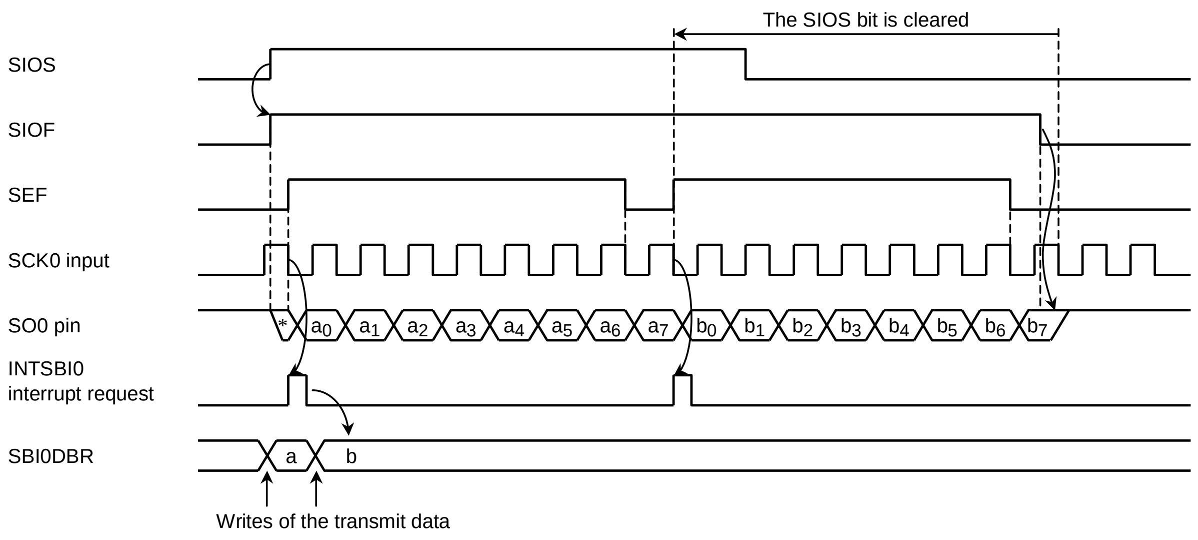
At the beginning of a transmission, the value of the last bit of the previously transmitted byte appears on the SO0 pin between when the SBI0SR.SIOF bit is set and when SCK subsequently goes low.

Transmission can be terminated by the INTSBI0 interrupt service routine clearing the SIOS bit to 0 or setting the SIOINH bit to 1. If the SIOS bit is cleared, the remaining bits in the SBI0DBR continue to be shifted out before transmission ends. In this case, software can check the SBI0SR.SIOF bit to determine whether transmission has come to an end (0 = end-of-transmission). If the SIOINH bit is set, the ongoing transmission is aborted immediately, and the SIOF bit is cleared at that point.

In external clock mode, the SIOS bit must be cleared before the SIO interface begins shifting out the next transmit data. Otherwise, the SIO will stop after sending out dummy data.


The SIOS bit is cleared
SIOS
SIOF
SEF
SCK0 output
SO0 pin
*
a_0 a_1 a_2 a_3 a_4 a_5 a_6 a_7 b_0 b_1 b_2 b_3 b_4 b_5 b_6 b_7
INTSBI0 interrupt request
SBI0DBR
a
b
Writes of the transmit data

(a) Internal clock mode


The SIOS bit is cleared
SIOS
SIOF
SEF
SCK0 input
SO0 pin
*
a_0 a_1 a_2 a_3 a_4 a_5 a_6 a_7 b_0 b_1 b_2 b_3 b_4 b_5 b_6 b_7
INTSBI0 interrupt request
SBI0DBR
a
b
Writes of the transmit data

(b) External clock mode

Figure 3.10.32 Transmit Mode

Example: Terminating transmission by SIOS (External clock mode)

```
STEST1:  BIT  SEF, (SBI0SR)            ; If SEF = 1  then loop.
         JR   NZ, STEST1
STEST2:  BIT  0, (P6)                  ; If SCK = 0  then loop.
         JR   Z, STEST2
         LD   (SBI0CR1), 00000111B     ; SIOS ← 0.
```

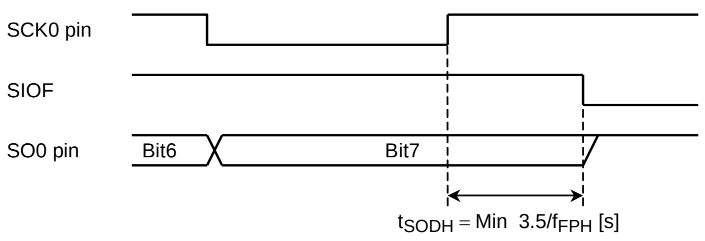


Figure 3.10.33 Retention Time of the Last Transmitted Bit

b. 8-bit receive mode

Configure the SIO interface in receive mode. Then setting the SIOS bit in the SBI0CR1 enables reception. The receive data is clocked into the internal shift register via the SI0 pin, with the least-significant bit (LSB) first, synchronous to the serial clock. Once the shift register is fully loaded, the received byte is transferred to the SBI0DBR, and the buffer-full interrupt (INTSBI0) is generated. The INTSBI0 interrupt service routine must then pick up the received data from the SBI0DBR.

In internal clock mode, the SIO interface will be in wait state (SCK will stop) until the INTSBI0 interrupt service routine reads the data from the SBI0DBR.

In external clock mode, shift operations continue, synchronous to the external clock. The INTSBI0 interrupt service routine must read the data from the SBI0DBR before the next serial clock pulse is applied. Otherwise, subsequently received data will be canceled. In this mode, the maximum data rate is a function of the maximum latency between when the INTSBI0 interrupt is generated and when the SBI0DBR is read by the interrupt service routine.

Reception can be terminated by the INTSBI0 interrupt service routine clearing the SIOS bit to 0 or setting the SIOINH bit to 1. If the SIOS bit is cleared, reception continues until the shift register is fully loaded and transferred to the SBI0DBR. In this case, software can check the SBI0SR.SIOF bit to determine whether reception has come to an end (0 = end-of-reception). If the SIOINH bit is set, the ongoing reception is aborted immediately, and the SIOF bit is cleared at that point. (The received data becomes invalid, there is no need to read it out.)

Note: The contents of the SBI0DBR are not preserved after changing the transfer mode. Before changing the transfer mode, clear the SIOS bit to complete the ongoing reception and have the INTSBI0 interrupt service routine pick up the last received data.

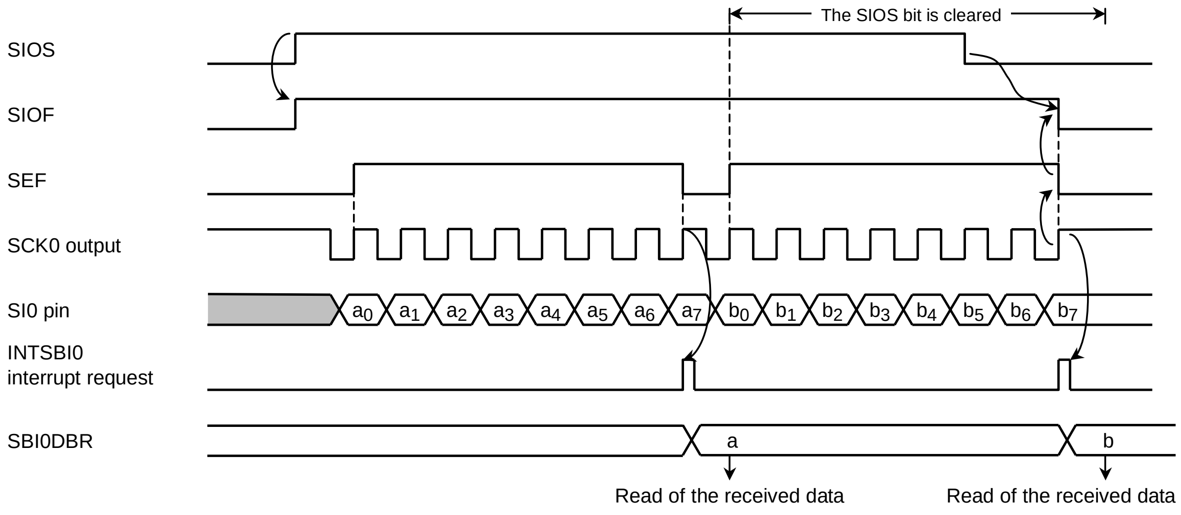

The SIOS bit is cleared
SIOS
SIOF
SEF
SCK0 output
SI0 pin
a0 a1 a2 a3 a4 a5 a6 a7 b0 b1 b2 b3 b4 b5 b6 b7
INTSBI0
interrupt request
SBI0DBR
a
b
Read of the received data
Read of the received data

Figure 3.10.34 Receive Mode (Internal clock mode)

c. 8-bit transmit/receive mode

Configure the SIO interface in transmit/receive mode and write the transmit data into the SBI0DBR. Then setting the SIOS bit in the SBI0CR1 initiates transmission and reception. The transmit data is shifted out through the SO0 pin, with the least-significant bit (LSB) first, with the falling edge of the serial clock, while at the same time the receive data is shifted in through the SI0 pin with the rising edge of the serial clock. Once the shift register is fully loaded with eight bits of the received data, it is transferred to the SBI0DBR, and the INTSBI0 interrupt is generated. The INTSBI0 interrupt service routine must then pick up the received data from the SBI0DBR and writes the next transmit data into the SBI0DBR. Because the SBI0DBR is shared between transmit and receive operations, the received data must be read before the next transmit data is written.

In internal clock mode, the SIO interface will be in wait state (SCK will stop) after a read of the received data until a write of the transmit data.

In external clock mode, shift operations continue, synchronous to the external clock. Therefore, software must read the received data and write the transmit data before the next shift operation begins. In this mode, the maximum data rate is a function of the maximum latency between when the INTSBI0 interrupt is generated and when the interrupt service routine reads the received data and writes the transmit data.

At the beginning of a transmission, the value of the last bit of the previously transmitted byte appears on the SO0 pin between when the SBI0SR.SIOF bit is set and when SCK subsequently goes low.

Transmission/reception can be terminated by the INTSBI0 interrupt service routine clearing the SIOS bit to 0 or setting the SIOINH bit to 1. If the SIOS bit is cleared, reception continues until the shift register is fully loaded and transferred to the SBI0DBR. In this case, software can check the SBI0SR.SIOF bit to determine whether transmission/reception has come to an end (0 = end-of-reception/transmission). If the SIOINH bit is set, the ongoing transmission/reception is aborted immediately, and the SIOF bit is cleared at that point.

Note: The contents of the SBI0DBR are not preserved after changing the transfer mode. Before changing the transfer mode, clear the SIOS bit to complete the ongoing transmission/reception and have the INTSBI0 interrupt service routine pick up the last received data.

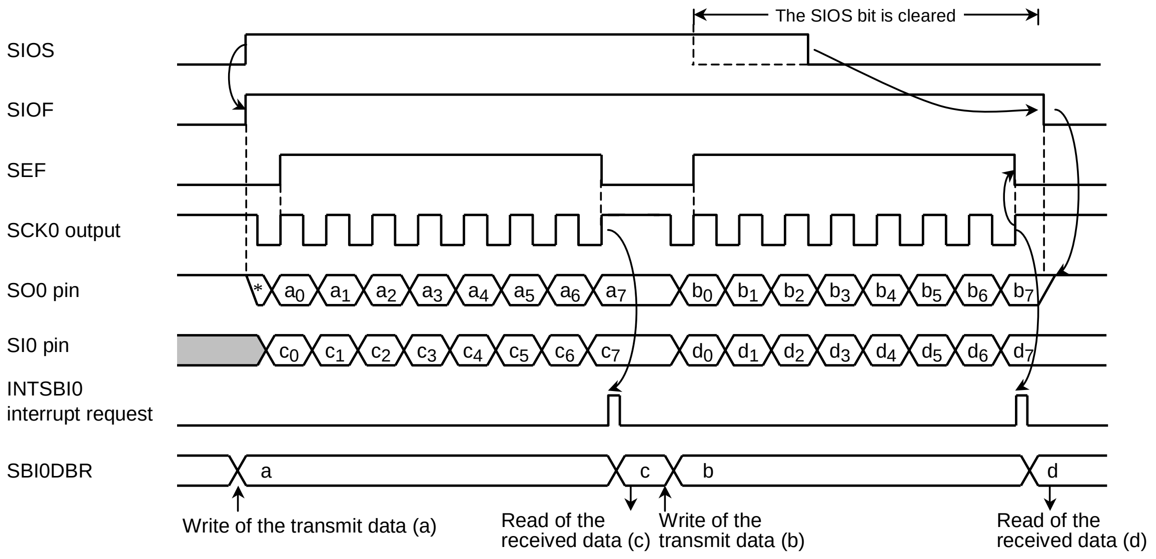

The SIOS bit is cleared
SIOS
SIOF
SEF
SCK0 output
SO0 pin
SI0 pin
INTSBI0
interrupt request
SBI0DBR
Write of the transmit data (a)
Read of the received data (c)
Write of the transmit data (b)
Read of the received data (d)

Figure 3.10.35 Receive/Transmit Mode (Internal clock mode)

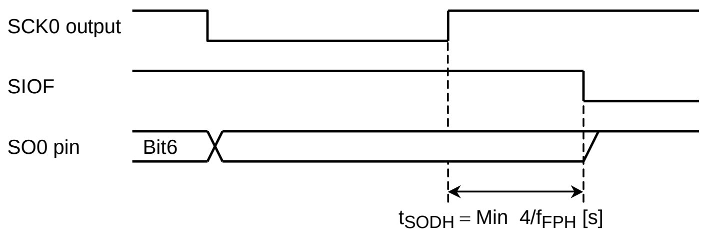


Figure 3.10.36 Retention Time of the Transmit Data in Receive/Transmit Mode

3.11 Analog-to-Digital Converter (ADC)

The TMP91CW28 has a 8-channel, multiplexed-input, 10-bit successive-approximation analog-to-digital converter (ADC).

Figure 3.11.1 shows a block diagram of the ADC. The eight analog input channels (AN0 to AN7) can be used as general-purpose digital inputs (Port 5) if not needed as analog channels.

Note: Ensure that the ADC has halted before executing the HALT instruction to place the TMP91CW28 in IDLE2, IDLE1 or STOP mode to reduce power supply current. Otherwise, the TMP91CW28 might go into a standby mode while the internal analog comparator is still active.

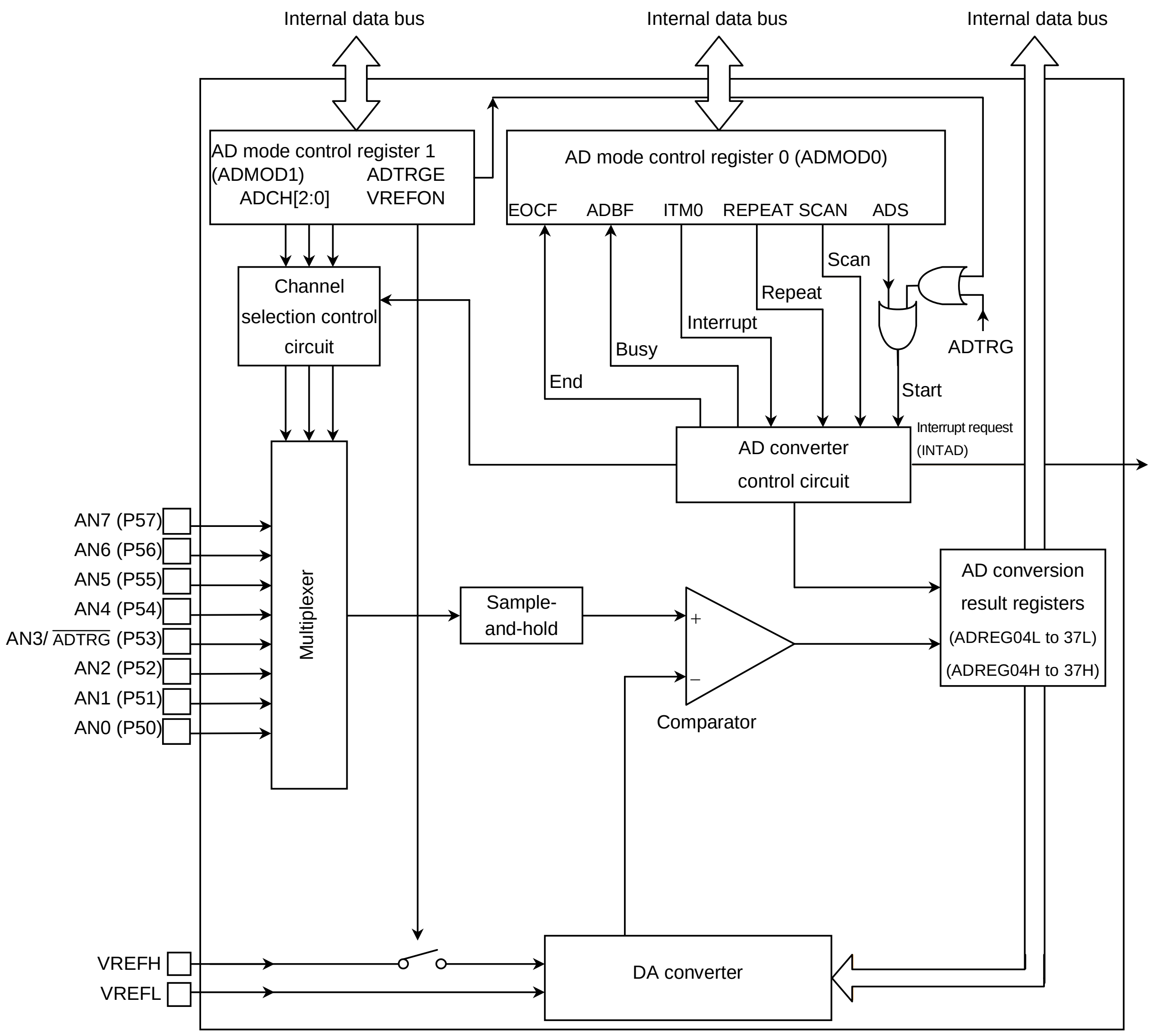

Internal data bus
Internal data bus
Internal data bus
AD mode control register 1 (ADMOD1) ADTRGE ADCH[2:0] VREFON
AD mode control register 0 (ADMOD0)
EOCF ADBF ITM0 REPEAT SCAN ADS
Scan
Repeat
Interrupt
Busy
End
Start
ADTRG
Channel selection control circuit
AD converter control circuit
Interrupt request (INTAD)
AN7 (P57)
AN6 (P56)
AN5 (P55)
AN4 (P54)
AN3/ $\overline{\text{ADTRG}}$ (P53)
AN2 (P52)
AN1 (P51)
AN0 (P50)
Multiplexer
Sample-and-hold
+
−
Comparator
AD conversion result registers (ADREG04L to 37L) (ADREG04H to 37H)
VREFH
VREFL
DA converter

Figure 3.11.1 ADC Block Diagram

3.11.1 Register Description

The ADC has two mode control registers (ADMOD0 and ADMOD1) and four conversion result high/low register pairs (ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L). The conversion result registers contain the digital values of completed conversions.

Figure 3.11.2 to Figure 3.11.5 show the registers available in the ADC.

AD Mode Control Register 0

ADMOD0 (02B0H)		7	6	5	4	3	2	1	0
	Bit symbol	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
	Read/Write	R		R/W					
	Reset value	0	0	0	0	0	0	0	0
	Function	End-of-conversion flag 0: During conversion 1: Completed	AD conversion busy flag 0: Idle 1: During conversion	Must be written as "0".	Must be written as "0".	Interrupt in fixed-channel continuous conversion mode 0: Generated after each conversion 1: Generated after every four conversions	Continuous conversion mode 0: Single 1: Continuous	Channel scan mode 0: Fixed-channel 1: Channel scan	AD conversion start 0: Don't care 1: Start This bit is always read as "0".

AD conversion start

0	Don't care
1	Starts AD conversion.

Note: This bit is always read as 0.

Channel scan mode

0	Fixed-channel mode
1	Channel scan mode

Continuous conversion mode

0	Single conversion mode
1	Continuous conversion mode

Interrupt in fixed-channel continuous conversion mode

	Fixed-channel continuous conversion mode SCAN = 0, REPEAT = 1
0	Generates INTAD interrupt when a single conversion has been completed.
1	Generates INTAD interrupt when a sequence of four conversions has been completed.

AD conversion busy flag

0	Idle
1	During conversion

End-of-conversion flag

0	Before or during conversion
1	Completed

Figure 3.11.2 AD Conversion Registers (1)

AD Mode Control Register 1

ADMOD1 (02B1H)

	7	6	5	4	3	2	1	0
Bit symbol	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
Read/Write	R/W	R/W			R/W			
Reset value	0	0			0	0	0	0
Function	VREF control 0: OFF 1: ON	ADC operation in IDLE2 mode 0: OFF 1: ON			External conversion trigger 0: Disable 1: Enable	Analog input channel select		

Analog input channel select

ADCH[2:0] \ SCAN	0 (Fixed-channel Mode)	1 (Channel Scan Mode)
000	AN0	AN0
001	AN1	AN0→AN1
010	AN2	AN0→AN1→AN2
011 (Note)	AN3	AN0→AN1→AN2→AN3
100	AN4	AN4
101	AN5	AN4→AN5
110	AN6	AN4→AN5→AN6
111	AN7	AN4→AN5→AN6→AN7

AD external conversion trigger ($\overline{\text{ADTRG}}$)

0	Disable
1	Enable

ADC operation in IDLE2 mode

0	OFF
1	ON

Reference voltage for the ADC

0	OFF
1	ON

Set the VREFON bit to 1 before setting the ADS bit in the ADMOD0 to start a conversion.

Note: The AN3 pin is shared with the $\overline{\text{ADTRG}}$ pin. Therefore, when the external conversion trigger input ($\overline{\text{ADTRG}}$) is enabled (e.g., when ADMOD1.ADTRGE = 1), the ADCH[2:0] field must not be programmed to 011.

Figure 3.11.3 AD Conversion Registers (2)

AD Conversion Result Low Register 0/4

ADREG04L (02A0H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00						ADR0RF
Read/Write	R							R
Reset value	Undefined							0
Function	Lower 2 bits of an AD conversion result							Conversion result store flag 1: Stored

AD Conversion Result High Register 0/4

ADREG04H (02A1H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
Reset value	Undefined							
Function	Upper 8 bits of an AD conversion result							

AD Conversion Result Low Register 1/5

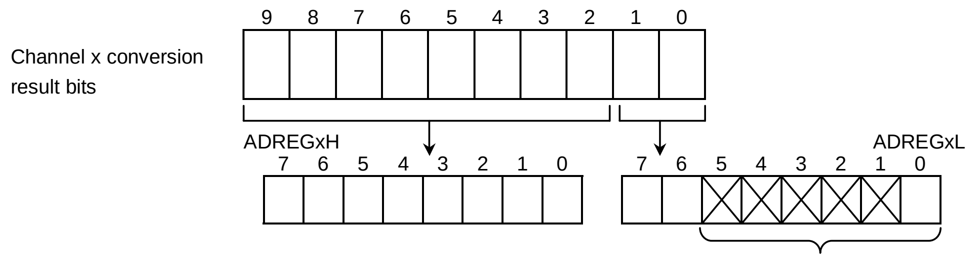
ADREG15L (02A2H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
Reset value	Undefined							0
Function	Lower 2 bits of an AD conversion result							Conversion result store flag 1: Stored

Lower 2 bits of an AD conversion result

ADREG15H (02A3H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
Reset value	Undefined							
Function	Upper 8 bits of an AD conversion result							



- Bits 5 to 1 are always read as 1s.
- Bit0 (ADRxRF), when set, indicates that the conversion result has been stored in the ADREGxH/L register pair. This bit is cleared when either the ADREGxH or the ADREGxL is read.

Figure 3.11.4 AD Conversion Registers (3)

AD Conversion Result Low Register 2/6

ADREG26L (02A4H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20						ADR2RF
Read/Write	R							R
Reset value	Undefined							0
Function	Lower 2 bits of an AD conversion result							Conversion result store flag 1: Stored

AD Conversion Result High Register 2/6

ADREG26H (02A5H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
Reset value	Undefined							
Function	Upper 8 bits of an AD conversion result							

AD Conversion Result Low Register 3/7

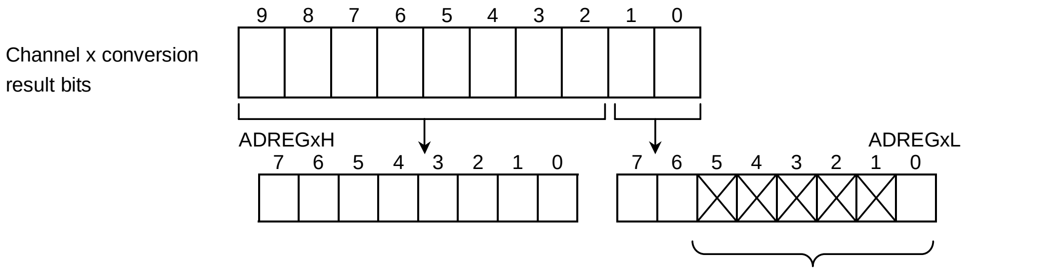
ADREG37L (02A6H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR31	ADR30						ADR3RF
Read/Write	R							R
Reset value	Undefined							0
Function	Lower 2 bits of an AD conversion result							Conversion result store flag 1: Stored

AD Conversion Result High Register 3/7

ADREG37H (02A7H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
Read/Write	R							
Reset value	Undefined							
Function	Upper 8 bits of an AD conversion result							



- Bits 5 to 1 are always read as 1s.
- Bit0 (ADRxRF), when set, indicates that the conversion result has been stored in the ADREGxH/L register pair. This bit is cleared when either the ADREGxH or the ADREGxL is read.

Figure 3.11.5 AD Conversion Registers (4)

3.11.2 Operation

(1) Analog reference voltages

The VREFH and VREFL pins provide the reference voltages for the ADC. These pins establish the full-scale range for the internal resistor string, which divides the range into 1024 steps. The digital result of the conversion is derived by comparing the sampled analog input voltage to the resistor string voltages.

Clearing the VREFON bit in the ADMOD1 turns off the switch between VREFH and VREFL. Once the VREFON bit is cleared, the internal reference voltage requires a recovery time of 3 μs to stabilize after the VREFON bit is again set to 1. This recovery time is independent of the system clock frequency. The ADS bit in the ADMOD0 must then be set to initiate a conversion.

(2) Selecting an analog input channel(s)

There are two basic conversion modes: Fixed-channel mode and channel scan mode. The SCAN bit in the ADMOD0 affects the conversion channel(s) that will be selected as follows.

- Fixed-channel mode (ADMOD0.SCAN = 0)

  When the SCAN bit in the ADMOD0 is cleared, the ADC runs conversions on a single input channel selected from AN0 to AN7 via the ADCH[2:0] field in the ADMOD1.

- Channel scan mode (ADMOD0.SCAN = 1)

  When the SCAN bit in the ADMOD0 is set, the ADC runs conversions on sequential channels in a specific group selected via the ADCH[2:0] field in the ADMOD1.

Refer to Table 3.11.1. After a reset, the ADMOD0.SCAN bit defaults to 0, and the ADMOD1.ADCH[2:0] field defaults to 000. Thus, the AN0 pin is selected as the conversion channel. The AN0 to AN7 pins can be used as general-purpose input ports if not used as analog input channels.

Table 3.11.1 Analog Input Channel Selection

ADMOD1.ADCH[2:0]	Fixed-channel Mode ADMOD0.SCAN = 0	Channel Scan Mode ADMOD0.SCAN = 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

(3) Starting an AD conversion

The ADC initiates a conversion or a sequence of conversions when the ADS bit in the ADMOD0 is set, or when a falling edge is applied to the $\overline{\text{ADTRG}}$ pin if the ADTRGE bit in the ADMOD1 is set. When a conversion starts, the busy flag (ADMOD0.ADBF) is set.

Writing a 1 to the ADS bit causes the ADC to abort any ongoing conversion and start sampling the selected channel to begin a new conversion. The conversion result store flag (ADREGxL.ADRxRF) indicates whether the result register contains a valid digital result at that point.

In external conversion trigger mode, a falling edge on the $\overline{\text{ADTRG}}$ pin is ignored while a conversion is in progress.

(4) Conversion modes and conversion-done interrupts

The ADC supports the following four conversion modes:

- Fixed-channel single conversion mode
- Channel scan single conversion mode
- Fixed-channel continuous conversion mode
- Channel scan continuous conversion mode

The REPEAT and SCAN bits in the ADMOD0 select the conversion mode.

The ADC generates the INTAD interrupt and sets the EOCF bit in the ADMOD0 at the end of the conversion process.

a. Fixed-channel single conversion mode

This mode is selected by programming the REPEAT and SCAN bits in the ADMOD0 to 00. In this mode, the ADC performs a single conversion on a single selected channel. When a conversion is completed, the ADC sets the ADMOD0.EOCF bit, clears the ADMOD0.ADBF bit and generates the INTAD interrupt.

b. Channel scan single conversion mode

This mode is selected by programming the REPEAT and SCAN bits in the ADMOD0 to 01. In this mode, the ADC performs a single conversion on each of a selected group of channels. When a single conversion sequence is completed, the ADC sets the ADMOD0.EOCF bit, clears the ADMOD0.ADBF bit and generates the INTAD interrupt.

c. Fixed-channel continuous conversion mode

This mode is selected by programming the REPEAT and SCAN bits in the ADMOD0 to 10. In this mode, the ADC repeatedly converts a single selected channel. When a conversion process is completed, the ADC sets the ADMOD.EOCF bit. The ADMOD0.ADBF bit remains set.

The ITM0 bit in the ADMOD0 controls interrupt generation in this mode. If the ITM0 bit is cleared, the ADC generates an interrupt after each conversion. If the ITM0 bit is set, the ADC generates an interrupt after every four conversions.

d. Channel scan continuous conversion mode

This mode is selected by programming the REPEAT and SCAN bits in the ADMOD0 to 11. In this mode, the ADC repeatedly converts the selected group of channels. When a single conversion sequence is completed, the ADC sets the ADMOD0.EOCF bit and generates the INTAD interrupt. The ADMOD0.ADBF bit remains set.

In continuous conversion modes (c and d), clearing the ADMOD0.REPEAT bit stops the conversion sequence after the ongoing conversion process is completed. The ADMOD0.ADBF bit is cleared.

If the I2AD bit in the ADMOD1 is cleared, putting the TMP91CW28 in any HALT mode (IDLE2, IDLE1, or STOP) causes the ADC to be immediately disabled, even if a conversion is in progress. Once the TMP91CW28 exits the HALT mode, the ADC restarts a conversion sequence when in a continuous conversion mode (c or d), but remains inactive when in a single conversion mode (a or b).

Table 3.11.2 summarizes interrupt request generation in each of the conversion modes.

Table 3.11.2 Interrupt Request Generation in Each AD Conversion Mode

Mode	Interrupt Request Generation	ADMOD0		
		ITM0	REPEAT	SCAN
Fixed-channel single conversion mode	After a conversion	X	0	0
Channel scan single conversion mode	After a scan conversion sequence	X	0	1
Fixed-channel continuous conversion mode	After each conversion	0	1	0
	After every four conversions	1		
Channel scan continuous conversion mode	After each scan conversion sequence	X	1	1

X: Don't care

(5) Conversion time

The conversion process requires 84 conversion states per channel. For example, this results in a conversion time of 16.8 μs with 10 MHz f_{FPH}.

(6) Storing and reading the AD conversion result

Conversion results are loaded into conversion result high/low register pairs (ADREG04H/L to ADREG37H/L). These registers are read only.

In fixed-channel continuous conversion mode, conversion data goes into the ADREG04H/L to the ADREG37H/L sequentially. In other modes, channels AN0 and AN4 share the ADREG04H/L; channels AN1 and AN5 share the ADREG15H/L; channels AN2 and AN6 share the ADREG26H/L; and channels AN3 and AN7 share the ADREG37H/L.

Table 3.11.3 shows the relationships between the analog input channels and the AD conversion result registers.

Table 3.11.3 Relationships between Analog Input Channels and AD Conversion Result Registers

Analog Input Channel (Port A)	AD Conversion Result Registers	
	Modes Other Than Fixed-channel Continuous Conversion Mode	Fixed-channel Continuous Conversion Mode (for each sequence of four conversions)
AN0	ADREG04H/L	$\begin{tikzcd} \text{ADREG04H/L} \arrow[d] \\ \text{ADREG15H/L} \arrow[d] \\ \text{ADREG26H/L} \arrow[d] \\ \text{ADREG37H/L} \arrow[uuu, bend right] \end{tikzcd}$
AN1	ADREG15H/L	
AN2	ADREG26H/L	
AN3	ADREG37H/L	
AN4	ADREG04H/L	
AN5	ADREG15H/L	
AN6	ADREG26H/L	
AN7	ADREG37H/L	

Bit0 (ADRxRF) in each ADREGxL register indicates whether the conversion result has been read. This bit is set when the conversion result is loaded into the ADREGxH/L pair, and cleared when either the ADREGxH or ADREGxL is read.

Reading the conversion result clears the end-of-conversion flag (ADMOD0.EOCF).

Programming examples:

a. Converting the analog input voltage on the AN3 pin to a digital value and storing the converted value in a memory location (0800H) using an AD interrupt (INTAD) handler routine.

Settings in the main routine

```
                 7 6 5 4 3 2 1 0
┌INTE0AD   ←  X 1 0 0 X – – –      Enables INTAD and sets its priority level to 4.
│ADMOD1    ←  1 1 X X 0 0 1 1      Selects AN3 as the analog input channel.
└ADMOD0    ←  X X 0 0 0 0 0 1      Starts conversion in fixed-channel single conversion mode.
```

Interrupt routine processing example

```
┌WA        ←  ADREG37              Loads the conversion result into 16-bit general-purpose register
│                                  WA from ADREG37L and ADREG37H.
│WA        > > 6                   Shifts the contents of WA 6 bits to the right, padding 0s to the
│                                  vacated high-order bits.
└(0800H)   ←  WA                   Stores the contents of WA to address 0800H.
```

b. Converting the analog input voltages on AN0 to AN2 sequentially in channel scan continuous conversion mode.

```
┌INTE0AD   ←  X 0 0 0 X – – –      Disables INTAD.
│ADMOD1    ←  1 1 X X 0 0 1 0      Selects AN2 as analog input channels.
└ADMOD0    ←  X X 0 0 0 1 1 1      Starts conversion in channel scan continuous conversion mode.
```

X: Don’t care, –: No change

3.12 Watchdog Timer (WDT)

The TMP91CW28 contains a watchdog timer (WDT). The WDT is used to regain control of the system in the event of software or system lockups due to spurious noises, etc. When a watchdog timer time-out occurs, the WDT generates a nonmaskable interrupt to the CPU.

Also, the time-out event can be programmed for system reset generation, which is accomplished by routing the time-out signal to the internal reset pin.

3.12.1 Implementation

Figure 3.12.1 shows a block diagram of the WDT.

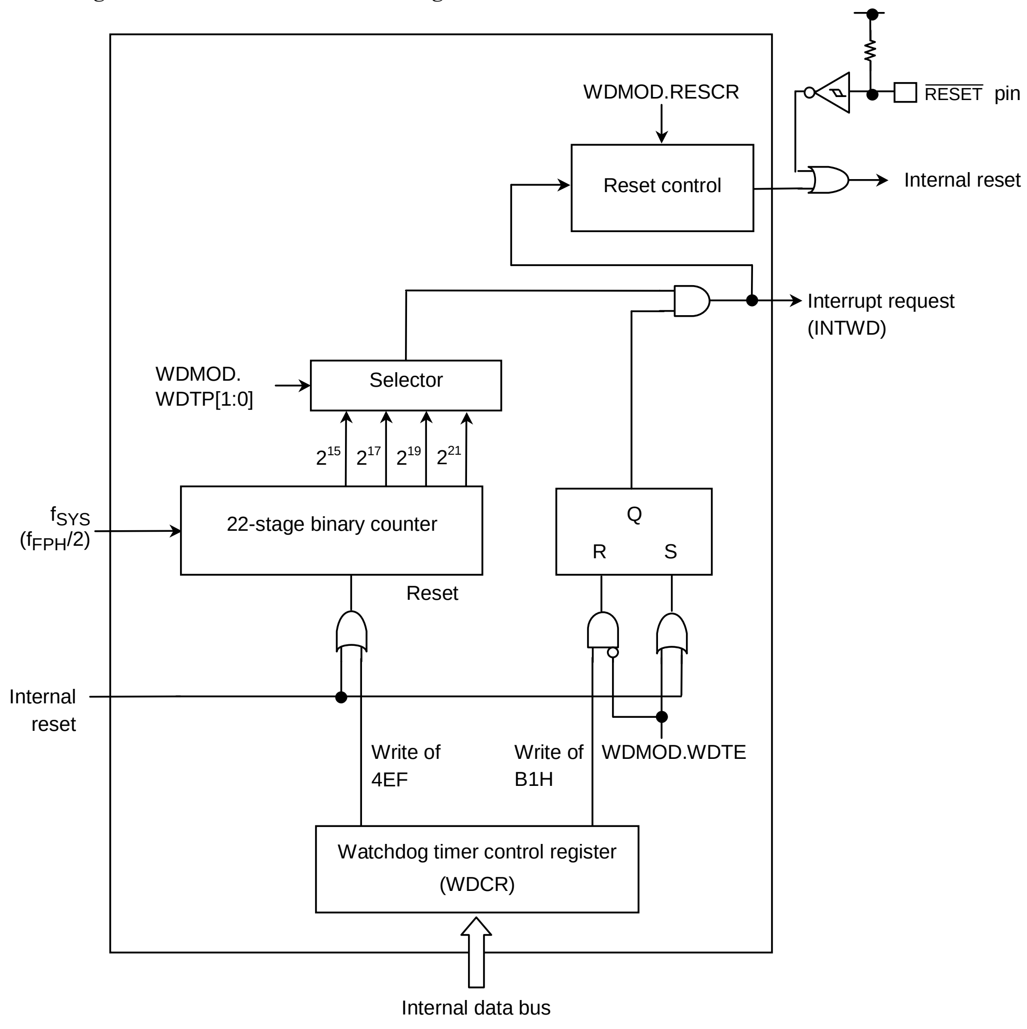


Figure 3.12.1 WDT Block Diagram

Note: Care must be exercised in the design of equipment because external noise or other factors may prevent the watchdog timer from functioning properly.

The WDT contains a 22-stage binary counter clocked by the f_{SYS} clock. This binary counter provides 2^{15}, 2^{17}, 2^{19}, or 2^{21} as a counter overflow signal, as programmed into the WDTP[1:0] field in the WDMOD. When a counter overflow occurs, the WDT generates a WDT interrupt, as shown below.

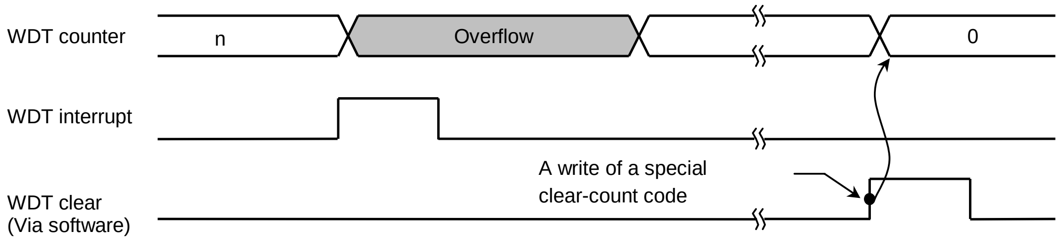


Figure 3.12.2 Default Operation

Also, the counter overflow can be programmed to cause a system reset as the time-out action. If so programmed, a counter overflow causes the WDT to assert the internal reset signal for a 22 to 29 state time (70.4 to 92.8 μs when f_{OSCH} = 10 MHz and f_{FPH} = 625 kHz). After a reset, the f_{SYS} clock (1 cycle = 1 state) is $f_{FPH}/2$, where f_{FPH} is generated by dividing the high-speed oscillator clock (f_{OSCH}) by sixteen through the clock gear function.

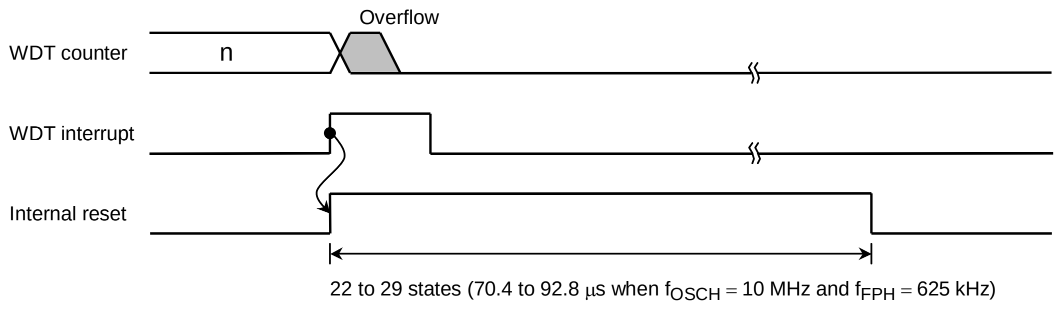

Overflow
WDT counter
n
WDT interrupt
Internal reset
22 to 29 states (70.4 to 92.8 μs when f_{OSCH} = 10 MHz and f_{FPH} = 625 kHz)

Figure 3.12.3 Reset Operation

3.12.2 Register Description

The WDT is controlled by two registers called WDMOD and WDCR.

(1) Watchdog timer mode register (WDMOD)

a. Time-out period (WDMOD.WDTP[1:0])

This 2-bit field determines the duration of the WDT time-out interval. Upon reset, the WDTP[1:0] field defaults to 00. Figure 3.12.4 shows possible time-out periods.

b. WDT enable (WDMOD.WDTE)

Upon reset, the WDTE bit is set to 1, enabling the WDT. To disable the WDT, the clearing of the WDTE bit must be followed by a write of a special key code (B1H) to the WDCR register. This prevents a "lost" program from disabling the WDT operation. The WDT can be re-enabled only by setting the WDTE bit.

c. System reset (WDMOD.RESCR)

This bit is used to program the WDT to generate a system reset on a time-out. Upon reset, this bit is cleared, thus the time-out does not cause a system reset.

(2) Watchdog timer control register (WDCR)

This register is used to disable the WDT and to clear the WDT binary counter.

- Disabling the WDT

  The WDT can be disabled by clearing the WDMOD.WDTE to 0 and then writing the special disable code (B1H) to the WDCR register.

```
WDMOD   ← 0 – – X X – – 0     Clears the WDTE bit to 0.
WDCR    ← 1 0 1 1 0 0 0 1     Writes the disable code (B1H) to the WDCR.
```

- Enabling the WDT

  The WDT can be enabled only by setting the WDTE bit in the WDMOD to 1.

- Clearing the WDT counter

  Writing the special clear-count code (4EH) to the WDCR resets the binary counter to 0. The counting process begins again.

```
WDCR    ← 0 1 0 0 1 1 1 0     Writes the clear-count code (4EH) to the WDCR.
```

Watchdog Timer Mode Register

WDMOD (0300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
Read/Write	R/W	R/W				R/W		R/W
Reset value	1	0	0			0	0	0
Function	WDT enable 0: Disable 1: Enable	Time-out period 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$				IDLE2 0: OFF 1: ON	System reset by WDT 1: Reset	Must be written as "0".

System reset

0	Don't care
1	Internally routes the WDT time-out signal to the system reset

Operation in IDLE2 mode

0	OFF
1	ON

Time-out period at fc = 10 MHz

Clock Gear Value SYSCR1.GEAR [2:0]	Watchdog Timer Time-out Period			
	WDMOD.WDTP[1:0]			
	00	01	10	11
000 (fc)	6.5536 ms	26.2144 ms	104.8576 ms	419.4304 ms
001 (fc/2)	13.1072 ms	52.4288 ms	209.7152 ms	838.8608 ms
010 (fc/4)	26.2144 ms	104.8576 ms	419.4304 ms	1.6784 s
011 (fc/8)	52.4288 ms	209.7152 ms	838.8608 ms	3.3552 s
100 (fc/16)	104.8576 ms	419.4304 ms	1.6784 s	6.7104 s

WDT enable

0	Disable
1	Enable

Figure 3.12.4 Watchdog Timer Mode Register (WDMOD)

Watchdog Timer Control Register

		7	6	5	4	3	2	1	0
WDCR (0301H)	Bit symbol	–							
	Read/Write	W							
	Reset value	–							
Read-modify-write operation is not supported.	Function	B1H: WDT disable code 4EH: WDT clear-count code							

Special code

B1H	WDT disable code
4EH	WDT clear-count code
Other values	Don't care

Figure 3.12.5 Watchdog Timer Control Register (WDCR)

3.12.3 Operation

The watchdog timer is a kind of timer that generates an interrupt request if it times out. The WDT allows the user to program the time-out period in the WDTP[1:0] field in the WDMOD register. While enabled, the software can reset the counter to 0 at any time by writing a special clear-count code. If the software is unable to reset the counter before it reaches the time-out count, the WDT generates the INTWD interrupt. In response to the interrupt, the CPU jumps to a system recovery routine to regain control of the system.

The WDT begins counting immediately after reset.

When the TMP91CW28 goes into IDLE1 or STOP mode, the WDT counter is reset to 0 automatically and stops counting. The WDT continues counting while an off-chip peripheral has mastership of the bus (e.g., $\overline{\text{BUSAK}}$ = 0).

In IDLE2 mode, the I2WDT bit in the WDMOD determines whether or not to disable the WDT. The I2WDT bit can be programmed before putting the TMP91CW28 in IDLE2 mode.

Examples:

1. Clearing the WDT binary counter

```
WDCR    ← 0 1 0 0 1 1 1 0    Writes the clear-count code (4EH) to the WDCR.
```

2. Programming the time-out interval to $2^{17}/f_{SYS}$

```
WDMOD   ← 1 0 1 X X − − 0
```

3. Disabling the watchdog timer

```
WDMOD   ← 0 − − X X − − 0    Clears the WDTE bit to 0.
WDCR    ← 1 0 1 1 0 0 0 1    Writes the disable code (B1H) to the WDCR.
```

3.13 Key Wakeup Interrupt

In addition to the INT0 to INT4 interrupt source pins, the TMP91CW28 has eight interrupt channels that enable the pressing of a key to terminate a HALT mode, called key wakeup interrupts (KWI).

Figure 3.13.1 shows a block diagram of the KWI circuit.

3.13.1 Block Diagram

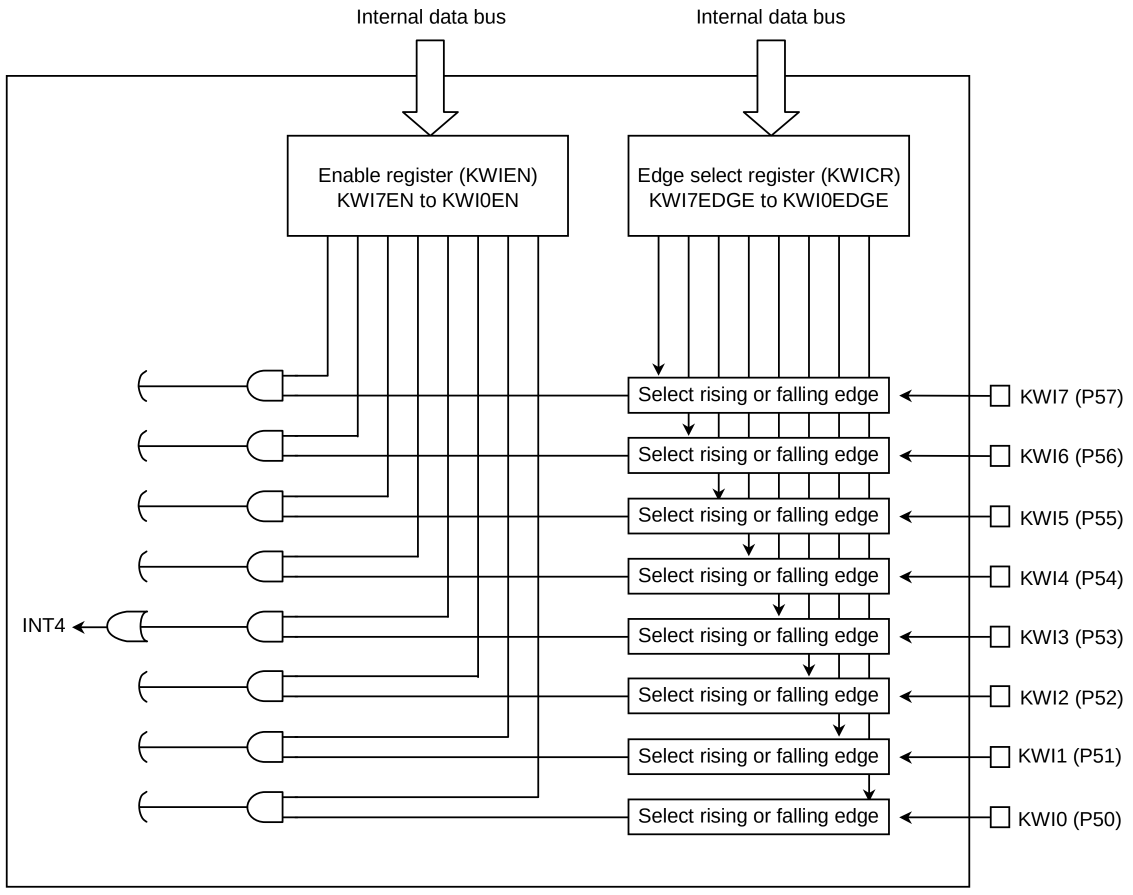


Figure 3.13.1 KWI Block Diagram

3.13.2 SFR Descriptions

Key Wakeup Enable Register

		7	6	5	4	3	2	1	0
KWIEN (03A0H)	Bit symbol	KWI7EN	KWI6EN	KWI5EN	KWI4EN	KWI3EN	KWI2EN	KWI1EN	KWI0EN
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	KWI7 interrupt input 0: Disable 1: Enable	KWI6 interrupt input 0: Disable 1: Enable	KWI5 interrupt input 0: Disable 1: Enable	KWI4 interrupt input 0: Disable 1: Enable	KWI3 interrupt input 0: Disable 1: Enable	KWI2 interrupt input 0: Disable 1: Enable	KWI1 interrupt input 0: Disable 1: Enable	KWI0 interrupt input 0: Disable 1: Enable

Key Wakeup Control Register

		7	6	5	4	3	2	1	0
KWICR (03A1H)	Bit symbol	KWI7EDGE	KWI6EDGE	KWI5EDGE	KWI4EDGE	KWI3EDGE	KWI2EDGE	KWI1EDGE	KWI0EDGE
	Read/Write	W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	KWI7 edge polarity 0: Rising 1: Falling	KWI6 edge polarity 0: Rising 1: Falling	KWI5 edge polarity 0: Rising 1: Falling	KWI4 edge polarity 0: Rising 1: Falling	KWI3 edge polarity 0: Rising 1: Falling	KWI2 edge polarity 0: Rising 1: Falling	KWI1 edge polarity 0: Rising 1: Falling	KWI0 edge polarity 0: Rising 1: Falling

Note: The KWIEN and KWICR do not support read-modify-write operation.

Figure 3.13.2 Key-pressed Wakeup Registers

3.13.3 Control

The P50 to P57 pins function as KWI0 to KWI7 when the corresponding bits (KWIEN[7:0]) in the KWIEN register are set. The MCU accepts KWI0 to KWI7 inputs as INT4. The KWI0 to KWI7 pins can be used as external interrupt sources by setting an interrupt priority level in the I4M[2:0] bits of the INTE34 register.

Example: To detect a falling edge on key wakeup channel 0 to generate an interrupt, configure registers in the following sequence:

```
KWICR    ←  –  –  –  –  –  –  –  1    Selects falling-edge detection for key wakeup
                                      channel 0.
KWIEN    ←  –  –  –  –  –  –  –  1    Enables key wakeup channel 0.

INTE34   ←  X  1  0  0  X  –  –  –    Enables INT4 and sets its priority level to 4.
```

X: Don’t care, –: No change

3.14 $\overline{\text{WAKE}}$ Pin

The TMP91CW28 can drive out a monitor signal immediately after it recovers from STOP mode in response to an external interrupt signal. The monitor signal is driven out through a dedicated N-ch open-drain output pin. External devices can know, in real time, when the TMP91CW28 enters or exits STOP mode.

This function is useful for a system that requires stop control for peripheral devices connected to the microcontroller.

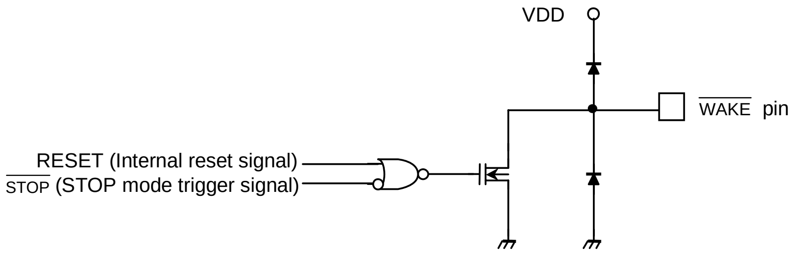

VDD
$\overline{\text{WAKE}}$ pin
RESET (Internal reset signal)
$\overline{\text{STOP}}$ (STOP mode trigger signal)

Figure 3.14.1 $\overline{\text{WAKE}}$ Pin

3.14.1 Basic Operation

While the TMP91CW28 is operating in IDLE1 or IDLE2 mode, a logic 0 is driven out from the $\overline{\text{WAKE}}$ pin. The pin is put into high-impedance state when the TMP91CW28 enters STOP mode. It is driven low when an external interrupt signal terminates STOP mode. The $\overline{\text{WAKE}}$ pin is, therefore, low during the warm-up period. It is placed in high-impedance state while a system reset (including a reset caused by the watchdog timer) is being performed.

Table 3.14.1 shows the states of the $\overline{\text{WAKE}}$ pin under different conditions. Figure 3.14.2 shows the $\overline{\text{WAKE}}$ signal output timing.

Table 3.14.1 $\overline{\text{WAKE}}$ Pin State

MCU State	$\overline{\text{WAKE}}$ Pin State
During a reset	High impedance
Operating (in IDLE1 or IDLE2 mode)	Low
In STOP mode	High impedance
During a warm up	Low

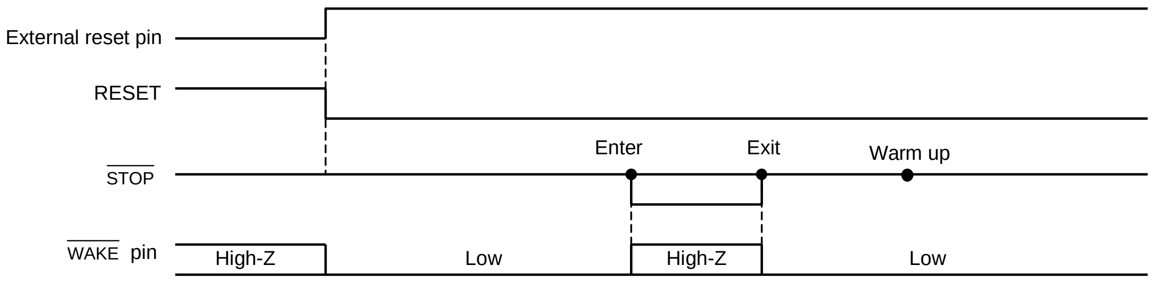


Figure 3.14.2 $\overline{\text{WAKE}}$ Signal Output Timing

3.15 BCD Adder/Subtractor

The TMP91CW28 has a BCD adder/subtractor that implements operation specific to the calculation of time data for a CD-ROM system. It can handle six-digit decimal data, consisting of two minute digits (0 to 99), two second digits (0 to 59) and two frame digits (0 to 74). A six-digit operand can be added to or subtracted from another six-digit operand.

As a result of calculation, the adder/subtractor stores the six-digit operation result as well as flags indicating whether there is a carry (CY) or a borrow (BR) produced from the minute digits.

(Input) Operand A:	Minutes (0 to 99)	Seconds (0 to 59)	Frames (0 to 74)	
(Input) Operand B: ±	Minutes (0 to 99)	Seconds (0 to 59)	Frames (0 to 74)	
(Output) Operation result:	Minutes (0 to 99)	Seconds (0 to 59)	Frames (0 to 74)	CY/BR

3.15.1 Block Diagram

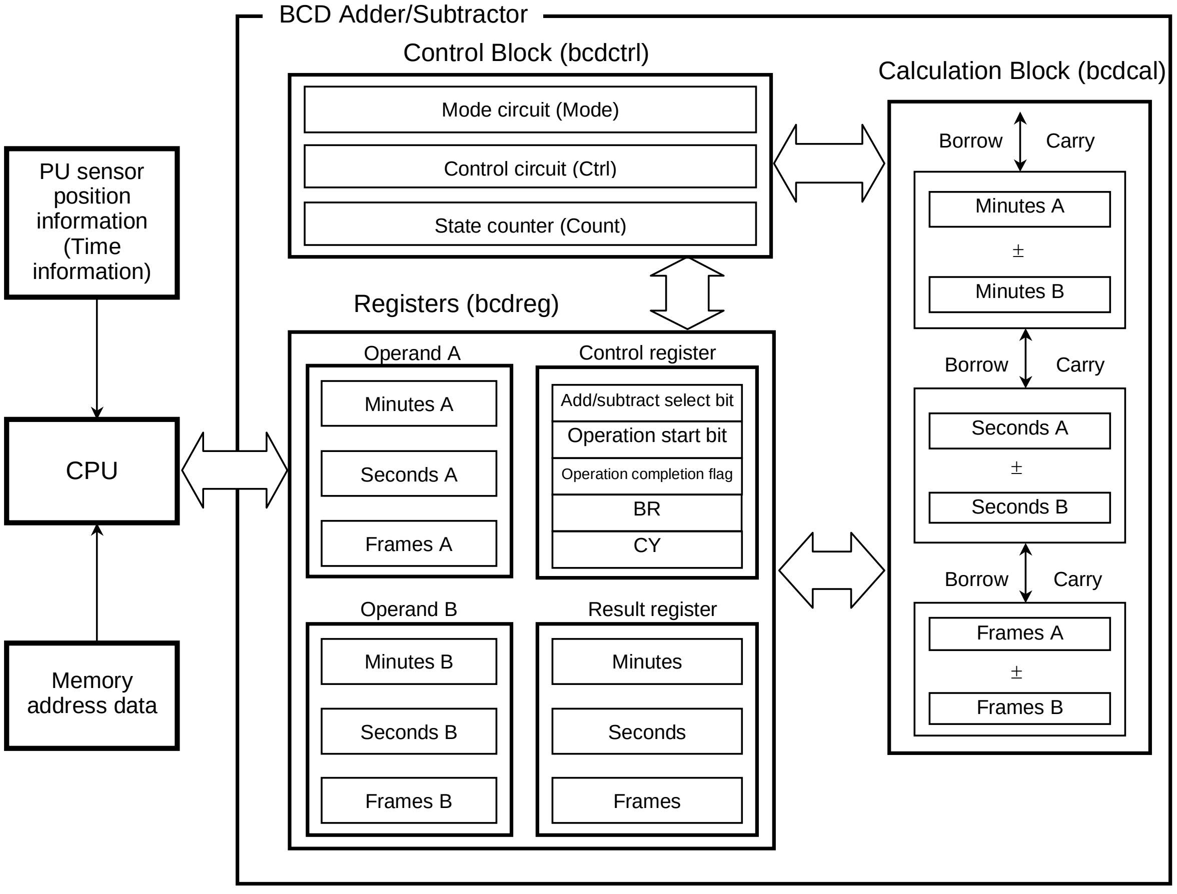


Figure 3.15.1 Block Diagram

3.15.2 SFR Descriptions

Minute Operand Register A

BCDMINA
03B0H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	MINA7	MINA6	MINA5	MINA4	MINA3	MINA2	MINA1	MINA0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand A							

Second Operand Register A

BCDSECA
03B1H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	SECA7	SECA6	SECA5	SECA4	SECA3	SECA2	SECA1	SECA0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand A							

Frame Operand Register A

BCDFRAA
03B2H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	FRAA7	FRAA6	FRAA5	FRAA4	FRAA3	FRAA2	FRAA1	FRAA0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand A							

Figure 3.15.2 BCD Registers (1/4)

Minute Operand Register B

BCDMINB
03B4H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	MINB7	MINB6	MINB5	MINB4	MINB3	MINB2	MINB1	MINB0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand B							

Second Operand Register B

BCDSECB
03B5H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	SECB7	SECB6	SECB5	SECB4	SECB3	SECB2	SECB1	SECB0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand B							

Frame Operand Register B

BCDFRAB
03B6H
Read-modify-write operation is not supported.

	7	6	5	4	3	2	1	0
Bit symbol	FRAB7	FRAB6	FRAB5	FRAB4	FRAB3	FRAB2	FRAB1	FRAB0
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operand B							

Figure 3.15.3 BCD Registers (2/4)

Minute Result Register

BCDMINR
03B8H

	7	6	5	4	3	2	1	0
Bit symbol	MINR7	MINR6	MINR5	MINR4	MINR3	MINR2	MINR1	MINR0
Read/Write	R							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operation result							

Second Result Register

BCDSECR
03B9H

	7	6	5	4	3	2	1	0
Bit symbol	SECR7	SECR6	SECR5	SECR4	SECR3	SECR2	SECR1	SECR0
Read/Write	R							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operation result							

Frame Result Register

BCDFRAR
03BAH

	7	6	5	4	3	2	1	0
Bit symbol	FRAR7	FRAR6	FRAR5	FRAR4	FRAR3	FRAR2	FRAR1	FRAR0
Read/Write	R							
Reset value	0	0	0	0	0	0	0	0
Function	BCD operation result							

Figure 3.15.4 BCD Registers (3/4)

BCD Control Register

BCDCR (03BCH)

	7	6	5	4	3	2	1	0
Bit symbol	ENDFLAG	CY	BR			–	CALSEL	START
Read/Write	R					R/W	R/W	R/W
Reset value	0	0	0			0	0	0
Function	Operation completion flag 0: During operation 1: Completed	Minute carry flag 0: Carry not produced 1: Carry produced	Minute borrow flag 0: Borrow not produced 1: Borrow produced			Must be written as “0”.	Add/sub-tract select 0: Add 1: Subtract	Operation start 0: Don’t care 1: Starts operation.

Operation start

0	Don’t care
1	Starts operation

Add/subtract select

0	Add
1	Subtract

Minute borrow flag

0	Borrow not produced
1	Borrow produced

Minute carry flag

0	Carry not produced
1	Carry produced

Operation completion flag

0	Before or during operation
1	Completed

Note: Bits3 and 4 of the BCDCR are read as undefined.

Figure 3.15.5 BCD Registers (4/4)

3.15.3 Operation

(1) Operand A

Operand A, to/from which operand B is added/subtracted, is stored in the BCDMINA, BCDSECA and BCDFRAA registers.

The operand must consist of decimal digits (0 to 9). If it contains a hexadecimal digit (A to F), operation is performed in decimal correction format, so that the result will not be an expected hexadecimal value.

The contents of the BCDMINA, BCDSECA, and BCDFRAA cannot be modified during operation. The operation completion flag must be checked to determine that operation has completed, before the registers can be modified.

(2) Operand B

Operand B, which is added to or subtracted from operand A, is stored in the BCDMINB, BCDSECB and BCDFRAB registers.

The operand must consist of decimal digits (0 to 9). If it contains a hexadecimal digit (A to F), operation is performed in decimal correction format, so that the result will not be an expected hexadecimal value.

The contents of the BCDMINB, BCDSECB, and BCDFRAB cannot be modified during operation. The operation completion flag must be checked to determine that operation has completed, before the registers can be modified.

(3) Operation result

The frame, second and minute digits of the result of addition or subtraction are sequentially stored in the BCDFRAR, BCDSECR and BCDMINR registers, respectively, in that order.

(4) Operation start

Writing 1 to the START bit in the BCDCR starts operation. Addition or subtraction is performed sequentially for frames, seconds and minutes in the stated order. The START bit is cleared upon the completion of operation.

The START bit cannot be cleared during operation.

(5) Add/subtract select

The CALSEL bit in the BCDCR specifies whether addition or subtraction is performed. Writing 0 to the bit selects addition and writing 1 to the bit selects subtraction.

The CALSEL bit cannot be modified during operation.

(6) BR (Borrow) flag

The BR flag bit in the BCDCR indicates whether a borrow is produced as a result of subtraction on minute digits. If the flag is cleared, that means a borrow is not produced. If the flag is set, that means a borrow is produced.

The BR flag is read as undefined when addition is performed.

(7) CY (Carry) flag

The CY flag bit in the BCDCR indicates whether a carry is produced as a result of addition on minute digits. If the flag is cleared, that means a carry is not produced. If the flag is set, that means a carry is produced.

The CY flag is read as undefined when subtraction is performed.

(8) Operation completion flag

The ENDFLAG bit in the BCDCR indicates whether operation has completed.

If the flag is cleared, that means operation is still in progress or not yet started. If the flag is set, that means operation has completed.

The ENDFLAG bit is set to 1 once the operation result for minutes has been stored in the BCDMINR. Reading any of the BCDMINR, BCDSECR and BCDFRAR causes the ENDFLAG to be cleared.

(9) Operation completion interrupt

When the BCDCR.ENDFLAG bit is set to 1, an INTBCD interrupt occurs.

3.15.4 Example

This section shows an example of operation using the BCD adder/subtractor.

1. Read the START bit of the BCDCR register to determine that no operation is in progress.
2. Set six-digit operand A (Minutes, seconds and frames) and six-digit operand B (Minutes, seconds and frames) in the appropriate registers.
3. Select addition or subtraction by clearing or setting the BCDCR.CALSEL bit.
4. Write 1 to the BCDCR.START bit to start operation.
5. Determine that operation has completed by reading 1 from the BCDCR.ENDFLAG bit or detecting the occurrence of an INTBCD interrupt.
6. Read the six-digit operation result (Minutes, seconds and frames) from the BCDMINR, BCDSECR and BCDFRAR registers as well as the CY and BR flags in the BCDCR. (BR is read as 0 after addition and CY is read as 0 after subtraction.)

The operand must consist of decimal digits (0 to 9). If it contains a hexadecimal digit (A to F), operation is performed in decimal correction format, so that the result will not be an expected hexadecimal value.

(1) Addition

Example: To add 99 minutes, 54 seconds, 32 frames to 1 minute, 5 seconds, 42 frames and generate an INTBCD interrupt, configure registers as follows:

		7	6	5	4	3	2	1	0	
A	←				BCDCR					Transfers the contents of BCDCR to 8-bit general-purpose register A to determine that no operation is in progress.
BCDMINA	←	1	0	0	1	1	0	0	0	Stores 98 in BCDMINA.
BCDSECA	←	0	1	0	1	0	1	0	0	Stores 54 in BCDSECA.
BCDFRAA	←	0	0	1	1	0	0	1	0	Stores 32 in BCDFRAA.
BCDMINB	←	0	0	0	0	0	0	0	1	Stores 01 in BCDMINB.
BCDSECB	←	0	0	0	0	0	1	0	1	Stores 05 in BCDSECB.
BCDFRAB	←	0	1	0	0	0	0	1	0	Stores 42 in BCDFRAB.
INTEBCD	←	X	X	X	X	X	1	0	0	Enables INTBCD and sets its priority level to 4.
BCDCR	←	X	X	X	X	X	X	0	1	Selects addition and starts operation.

Example of interrupt routine processing

A	←	BCDCR	Transfers the contents of BCDCR to 8-bit general-purpose register A to determine that no operation is in progress.
B	←	BCDMINR	Reads the result of operation for minutes.
C	←	BCDSECR	Reads the result of operation for seconds.
D	←	BCDFRAR	Reads the result of operation for frames.
E	←	BCDCR	Reads the contents of BCDCR to 8-bit general-purpose register E to determine whether a carry has been produced.

X: Don't care, –: No change

3.16 Program Patch Logic

The TMP91CW28 has a program patch logic, which enables the user to fix the program code in the on-chip ROM without generating a new mask. Patch program must be read into on-chip RAM from external memory during the startup routine.

Up to six two-byte sequences, or banks (Twelve bytes in total) can be replaced with patch code. More significant code correction can be performed by replacing program code with single-byte instruction code which generates a software interrupt (SWI) to make a branch to a specified location in the on-chip RAM area.

The program patch logic only compares addresses in the on-chip ROM area; it cannot fix the program code in the on-chip peripheral, on-chip RAM and external ROM areas.

Each of six banks is independently programmable, and functionally equivalent. In the following sections, any references to bank0 also apply to other banks.

3.16.1 Block Diagram

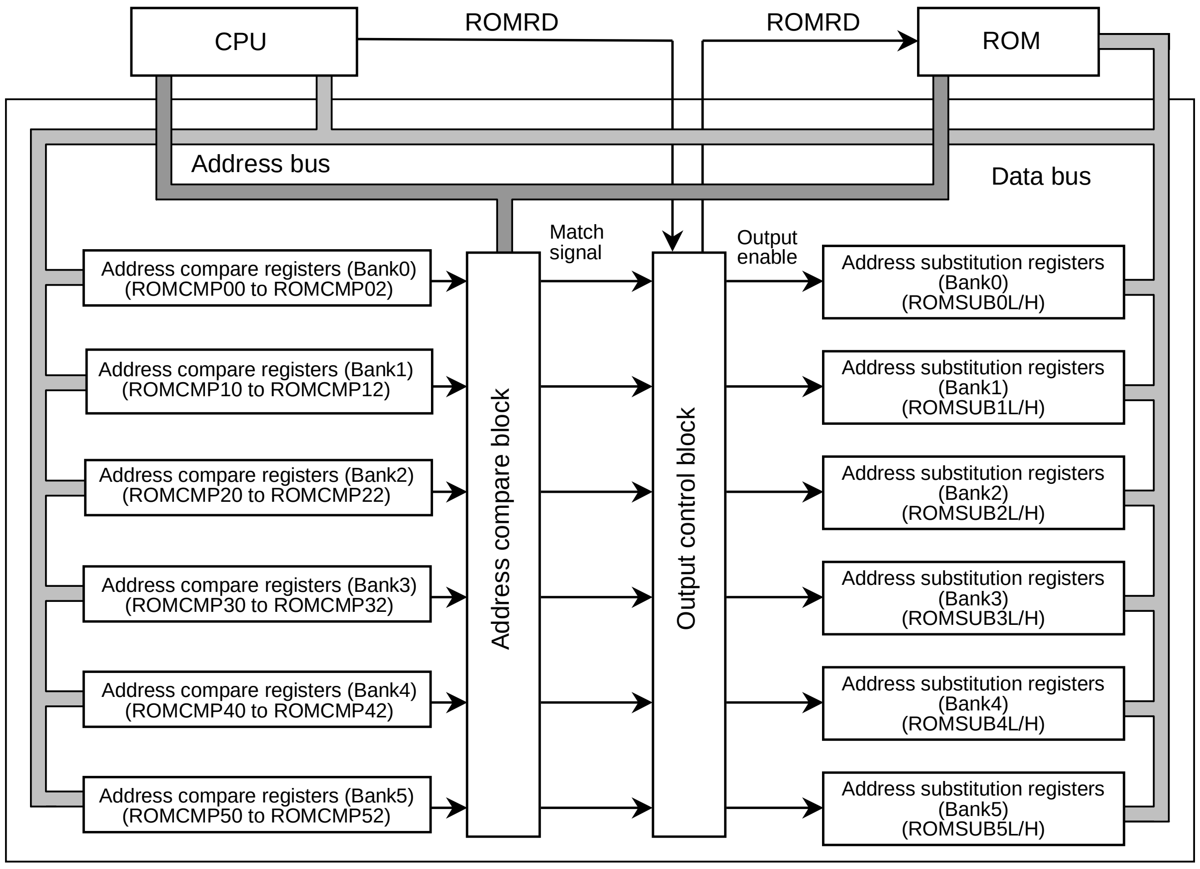

CPU
ROMRD
ROMRD
ROM
Address bus
Data bus
Match signal
Output enable
Address compare registers (Bank0) (ROMCMP00 to ROMCMP02)
Address compare registers (Bank1) (ROMCMP10 to ROMCMP12)
Address compare registers (Bank2) (ROMCMP20 to ROMCMP22)
Address compare registers (Bank3) (ROMCMP30 to ROMCMP32)
Address compare registers (Bank4) (ROMCMP40 to ROMCMP42)
Address compare registers (Bank5) (ROMCMP50 to ROMCMP52)
Address compare block
Output control block
Address substitution registers (Bank0) (ROMSUB0L/H)
Address substitution registers (Bank1) (ROMSUB1L/H)
Address substitution registers (Bank2) (ROMSUB2L/H)
Address substitution registers (Bank3) (ROMSUB3L/H)
Address substitution registers (Bank4) (ROMSUB4L/H)
Address substitution registers (Bank5) (ROMSUB5L/H)

Figure 3.16.1 Program Patch Logic Diagram

3.16.2 SFR Descriptions

The program patch logic consists of six banks (0 to 5). Each bank is provided with three bytes of address compare registers (ROMCMPx0 to ROMCMPx2) and two bytes of address substitution registers (ROMSUBxL and ROMSUBxH).

Bank0 Address Compare Register 0

		7	6	5	4	3	2	1	0
ROMCMP00 (0400H)	Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	
	Function	Target ROM address (Lower 7 bits)							

Bank0 Address Compare Register 1

		7	6	5	4	3	2	1	0
ROMCMP01 (0401H)	Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Middle 8 bits)							

Bank0 Address Compare Register 2

		7	6	5	4	3	2	1	0
ROMCMP02 (0402H)	Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP00, ROMCMP01, and ROMCMP02 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP00 is read as undefined.

Figure 3.16.2 Address Compare Registers (Bank0)

Bank1 Address Compare Register 0

ROMCMP10 (0408H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
Read/Write	W							
Reset value	0	0	0	0	0	0	0	
Function	Target ROM address (Lower 7 bits)							

Bank1 Address Compare Register 1

ROMCMP11 (0409H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank1 Address Compare Register 2

ROMCMP12 (040AH)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP10, ROMCMP11, and ROMCMP12 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP10 is read as undefined.

Figure 3.16.3 Address Compare Registers (Bank1)

Bank2 Address Compare Register 0

		7	6	5	4	3	2	1	0
ROMCMP20 (0410H)	Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	
	Function	Target ROM address (Lower 7 bits)							

Bank2 Address Compare Register 1

		7	6	5	4	3	2	1	0
ROMCMP21 (0411H)	Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Middle 8 bits)							

Bank2 Address Compare Register 2

		7	6	5	4	3	2	1	0
ROMCMP22 (0412H)	Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP20, ROMCMP21, and ROMCMP22 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP20 is read as undefined.

Figure 3.16.4 Address Compare Registers (Bank2)

Bank3 Address Compare Register 0

ROMCMP30 (0418H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	
	Function	Target ROM address (Lower 7 bits)							

Bank3 Address Compare Register 1

ROMCMP31 (0419H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Middle 8 bits)							

Bank3 Address Compare Register 2

ROMCMP32 (041AH)		7	6	5	4	3	2	1	0
	Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP30, ROMCMP31, and ROMCMP32 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP30 is read as undefined.

Figure 3.16.5 Address Compare Registers (Bank3)

Bank4 Address Compare Register 0

		7	6	5	4	3	2	1	0
ROMCMP40 (0420H)	Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	
	Function	Target ROM address (Lower 7 bits)							

Bank4 Address Compare Register 1

		7	6	5	4	3	2	1	0
ROMCMP41 (0421H)	Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Middle 8 bits)							

Bank4 Address Compare Register 2

		7	6	5	4	3	2	1	0
ROMCMP42 (0422H)	Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP40, ROMCMP41, and ROMCMP42 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP40 is read as undefined.

Figure 3.16.6 Address Compare Registers (Bank4)

Bank5 Address Compare Register 0

		7	6	5	4	3	2	1	0
ROMCMP50 (0428H)	Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	
	Function	Target ROM address (Lower 7 bits)							

Bank5 Address Compare Register 1

		7	6	5	4	3	2	1	0
ROMCMP51 (0429H)	Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Middle 8 bits)							

Bank5 Address Compare Register 2

		7	6	5	4	3	2	1	0
ROMCMP52 (042AH)	Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Target ROM address (Upper 8 bits)							

Note 1: The ROMCMP50, ROMCMP51, and ROMCMP52 registers do not support read-modify-write operation.

Note 2: Bit0 of the ROMCMP50 is read as undefined.

Figure 3.16.7 Address Compare Registers (Bank5)

Bank0 Address Substitution Register L

ROMSUB0L (0404H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Lower 8 bits)							

Bank0 Address Substitution Register H

ROMSUB0H (0405H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Upper 8 bits)							

Bank1 Address Substitution Register L

ROMSUB1L (040CH)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Lower 8 bits)							

Bank1 Address Substitution Register H

ROMSUB1H (040DH)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Upper 8 bits)							

Note: The ROMSUB0L, ROMSUB0H, ROMSUB1L, and ROMSUB1H registers do not support read-modify-write operation.

Figure 3.16.8 Address Substitution Registers (Banks 0 and 1)

Bank2 Address Substitution Register L

ROMSUB2L (0414H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank2 Address Substitution Register H

ROMSUB2H (0415H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Bank3 Address Substitution Register L

ROMSUB3L (041CH)

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank3 Address Substitution Register H

ROMSUB3H (041DH)

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
Reset value	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note: The ROMSUB2L, ROMSUB2H, ROMSUB3L, and ROMSUB3H registers do not support read-modify-write operation.

Figure 3.16.9 Address Substitution Registers (Banks 2 and 3)

Bank4 Address Substitution Register L

ROMSUB4L (0424H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Lower 8 bits)							

Bank4 Address Substitution Register H

ROMSUB4H (0425H)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Upper 8 bits)							

Bank5 Address Substitution Register L

ROMSUB5L (042CH)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Lower 8 bits)							

Bank5 Address Substitution Register H

ROMSUB5H (042DH)		7	6	5	4	3	2	1	0
	Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
	Read/Write	W							
	Reset value	0	0	0	0	0	0	0	0
	Function	Patch code (Upper 8 bits)							

Note: The ROMSUB4L, ROMSUB4H, ROMSUB5L, and ROMSUB5H registers do not support read-modify-write operation.

Figure 3.16.10 Address Substitution Registers (Banks 4 and 5)

3.16.3 Operation

(1) Replacing data

Two consecutive bytes of data can be replaced for each bank. A two-byte sequence to be replaced must start at an even address. If only a single byte at an even or odd address need be replaced, set the current masked ROM data in the other byte.

Correction procedure:

Load the address compare registers (ROMCMP00 to ROMCMP02) with the target address where ROM data need be replaced. Store 2-byte patch code in the ROMSUBL and ROMSUBH registers.

When the CPU address matches the value stored in the ROMCMP00 to ROMCMP02 registers, the program patch logic disables RD output to the masked ROM and drives out the code stored in the ROMSUBL and ROMSUBH to the internal bus. The CPU thus fetches the patch code.

The following shows some examples:

Examples:

a. Replacing 00H at address FF1230H with AAH

		7	6	5	4	3	2	1	0	
ROMCMP00	←	0	0	1	1	0	0	0	0	Stores 30 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	←	1	0	1	0	1	0	1	0	Store AA in address substitution register low for bank0.
ROMSUB0H	←	0	0	0	1	0	0	0	1	Store 11 in address substitution register high for bank0.

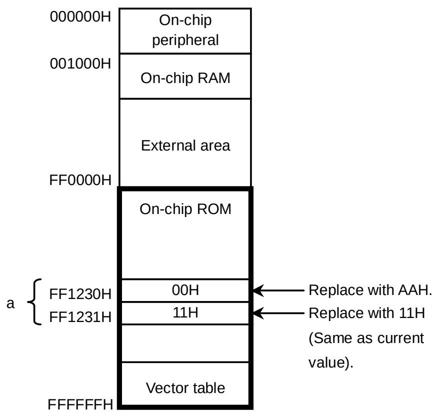


Figure 3.16.11 Example Patch Code Implementation

b. Replacing 33H at address FF1233H with BBH

		7	6	5	4	3	2	1	0	
ROMCMP00	←	0	0	1	1	0	0	1	0	Stores 32 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	←	0	0	1	0	0	0	1	0	Store 22 in address substitution register low for bank0.
ROMSUB0H	←	1	0	1	1	1	0	1	1	Store BB in address substitution register high for bank0.

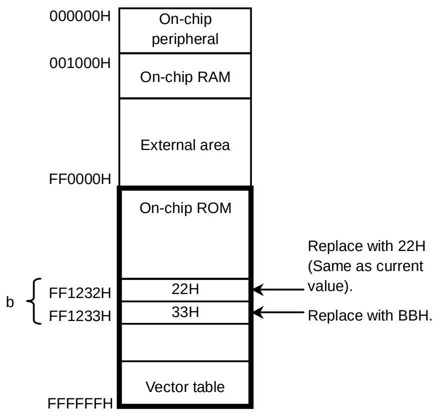


Figure 3.16.12 Example Patch Code Implementation

c. Replacing 44H at address FF1234H with CCH and 55H at address FF1235H with DDH

		7	6	5	4	3	2	1	0	
ROMCMP00	←	0	0	1	1	0	1	0	0	Stores 34 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	←	1	1	0	0	1	1	0	0	Store CC in address substitution register low for bank0.
ROMSUB0H	←	1	1	0	1	1	1	0	1	Store DD in address substitution register high for bank0.

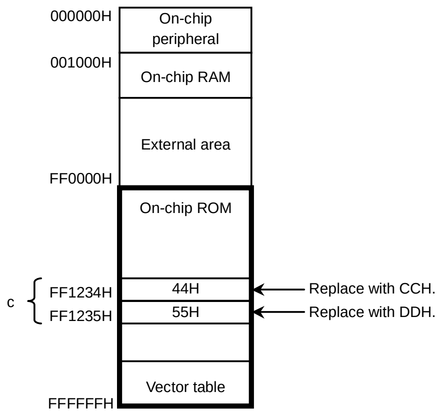


Figure 3.16.13 Example Patch Code Implementation

d. Replacing 77H at address FF1237H with EEH and 88H at address FF1238H with FFH (Requiring two banks)

		7	6	5	4	3	2	1	0	
ROMCMP00	←	0	0	1	1	0	1	1	0	Stores 36 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	←	0	1	1	0	0	1	1	0	Store 66 in address substitution register low for bank0.
ROMSUB0H	←	1	1	1	0	1	1	1	0	Store EE in address substitution register high for bank0.
ROMCMP10	←	0	0	1	1	1	0	0	0	Stores 38 in address compare register 0 for bank1.
ROMCMP11	←	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank1.
ROMCMP12	←	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank1.
ROMSUB1L	←	1	1	1	1	1	1	1	1	Store FF in address substitution register low for bank1.
ROMSUB1H	←	1	0	0	1	1	0	0	1	Store 99 in address substitution register high for bank1.

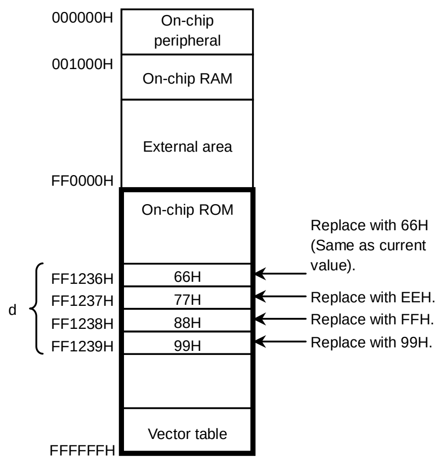


Figure 3.16.14 Example Patch Code Implementation

(2) Using an interrupt to cause a branch

A wider range of program code can also be fixed using a software interrupt (SWI). With a patch code loaded into on-chip RAM, the program patch logic can be used to replace program code at a specified address with a single-byte SWI instruction, which causes a branch to the patch program.

Note that this method can only be used if the original masked ROM has been developed with <u>on-chip RAM addresses specified as SWI vector addresses</u>.

Correction procedure:

Load the address compare registers (ROMCMP00 to ROMCMP02) with the start address of the program code that is to be fixed. If it is an even address, store an SWI instruction code (e.g., SWI:F9H) in the ROMSUBL. If the start address is an odd address, store an SWI instruction code in the ROMSUBH and the current ROM data at the preceding even address in the ROMSUBL.

When the CPU address matches the value stored in the ROMCMP00 to ROMCMP02 registers, the program patch logic disables RD output to the masked ROM and drives out the SWI instruction code to the internal bus. Upon fetching the SWI code, the CPU makes a branch to the internal RAM area to execute the preloaded code.

At the end of the patch program executed from the internal RAM, the CPU directly rewrites the saved PC value so that it points to the address following the patch code, and then executes a RETI.

The following shows an example:

Example: Fixing a program within the range from FF5000H to FF507FH

Before developing the original masked ROM, set the SWI1 vector reference address to 001500H (on-chip RAM area).

Use the startup routine to load the patch code to on-chip RAM (001500H to 0015EFH). Store the start address (FF5000H) of the ROM area to be fixed in the ROMCMP00 to ROMCMP02. Store the SWI1 instruction code (F9H) in the ROMSUB0L and the current data at FF5001H (AAH) in the ROMSUB0H. When the CPU address matches the value stored in ROMCMP00 to ROMCMP02, the program patch logic replaces the ROM-based code at FF5000H with F9H. The CPU then executes the SWI1 instruction, which causes a branch to 001500H in the on-chip RAM area. After executing the patch program the CPU finally rewrites the saved PC value to FF5080H and executes a RETI.

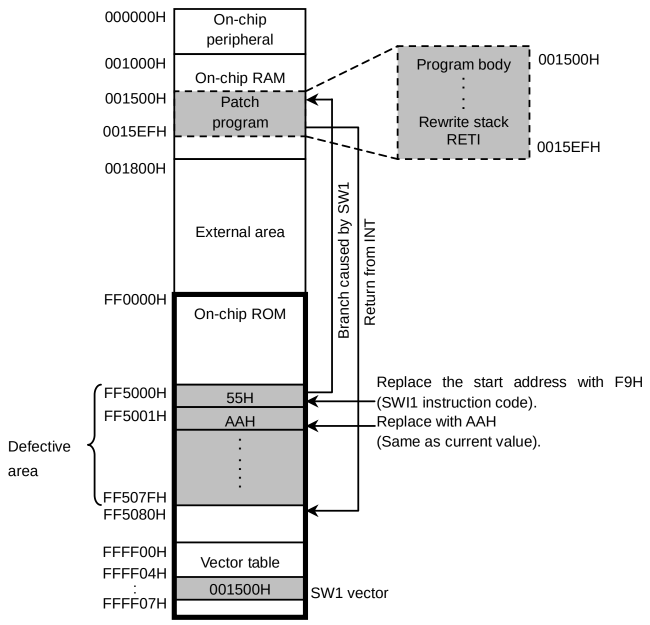


Figure 3.16.15 Example ROM Correction

4. Electrical Characteristics

4.1 Maximum Ratings

Parameter	Symbol	Rating	Unit
Supply voltage	Vcc	−0.5 to 3.0	V
Input voltage	VIN	−0.5 to Vcc + 0.5	
Output current (Per pin)	IOL	2	mA
Output current (Per pin)	IOH	−2	
Output current (Total)	ΣIOL	80	
Output current (Total)	ΣIOH	−80	
Power dissipation (Ta = 85°C)	PD	600	mW
Soldering temperature (10 s)	TSOLDER	260	°C
Storage temperature	TSTG	−55 to 125	
Operating temperature	TOPR	−20 to 70	

Note: Maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no maximum rating value is exceeded. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.

Point of note about solderability of lead free products (attach "G" to package name)

Test Parameter	Test Condition	Note
Solderability	(1) Use of Sn-63Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 [s] The number of times = One, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	(2) Use of Sn-3.0Ag-0.5 Cu solder Bath Solder bath temperature = 245, Dipping time = 5 [s] The number of times = One, Use of R-type flux (use of lead free)	

4.2 DC Electrical Characteristics (1/2)

Parameter		Symbol	Condition		Min	Typ. (Note)	Max	Unit
Power supply voltage (AVcc = DVcc, AVss = DVss = 0 V)		VCC	fc = 4 to 10 MHz		1.8		2.6	V
Low-level input voltage	P00 to P17 (AD0 to AD15)	VIL	V_{CC} = 1.8 to 2.6 V		−0.3		0.2 Vcc	V
	P20 to P37	VIL1	V_{CC} = 1.8 to 2.6 V				0.2 Vcc	
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P40 to PA7	VIL2	V_{CC} = 1.8 to 2.6 V				0.15 Vcc	
	AM0 to AM1	VIL3	V_{CC} = 1.8 to 2.6 V				0.3	
	X1	VIL4	V_{CC} = 1.8 to 2.6 V				0.1 Vcc	
High-level input voltage	P00 to P17 (AD0 to AD15)	VIH	V_{CC} = 1.8 to 2.6 V		0.7 Vcc		Vcc + 0.3	V
	P20 to P37	VIH1	V_{CC} = 1.8 to 2.6 V		0.8 Vcc			
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P40 to PA7	VIH2	V_{CC} = 1.8 to 2.6 V		0.85 Vcc			
	AM0 to AM1	VIH3	V_{CC} = 1.8 to 2.6 V		Vcc − 0.3			
	X1	VIH4	V_{CC} = 1.8 to 2.6 V		0.9 Vcc			
Low-level output voltage		VOL	IOL = 0.4 mA	V_{CC} = 1.8 to 2.6 V			0.15 Vcc	V
High-level output voltage		VOH	IOH = −200 μA	V_{CC} = 1.8 to 2.6 V	0.8 Vcc			

Note: Ta = 25°C, Vcc = 2.0 V, unless otherwise noted.

DC Electrical Characteristics (2/2)

Parameter	Symbol	Condition	Min	Typ. (Note 1)	Max	Unit
Input leakage current	ILI	$0.0 \le V_{IN} \le Vcc$		0.02	±5	μA
Output leakage current	ILO	$0.2 \le V_{IN} \le Vcc - 0.2$		0.05	±10	
Power down voltage (while RAM is being backed up in STOP mode)	VSTOP	V IL2 = 0.2 Vcc, V IH2 = 0.8 Vcc	1.8		2.6	V
$\overline{\text{RESET}}$ pull-up resistor	RRST	V_{CC} = 1.8 to 2.2 V	200		1000	kΩ
		V_{CC} = 2.2 to 2.6 V	100		600	
Pin capacitance	CIO	fc = 1 MHz			10	pF
Schmitt width $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P40 to P43, KWI0 to KWI7, P60 to PA7	VTH	V_{CC} = 1.8 to 2.6 V	0.3	0.8		V
Programmable pull-up resistor	RKH	V_{CC} = 1.8 to 2.2 V	200		1000	kΩ
		V_{CC} = 2.2 to 2.6 V	100		600	
NORMAL (Note 2)	Icc	V_{CC} = 1.8 to 2.6 V fc = 10 MHz (Typ. value Vcc = 2.0 V)		2.2	4.0	mA
IDLE2				0.7	1.6	
IDLE1				0.3	0.9	
STOP		V_{CC} = 1.8 to 2.6 V		0.1	10	μA

Note 1: Ta = 25°C, V_{CC} = 2.0 V, unless otherwise noted.

Note 2: Test conditions for NORMAL Icc: All blocks operating, output pins open, and input pin levels fixed.

4.3 AC Electrical Characteristics

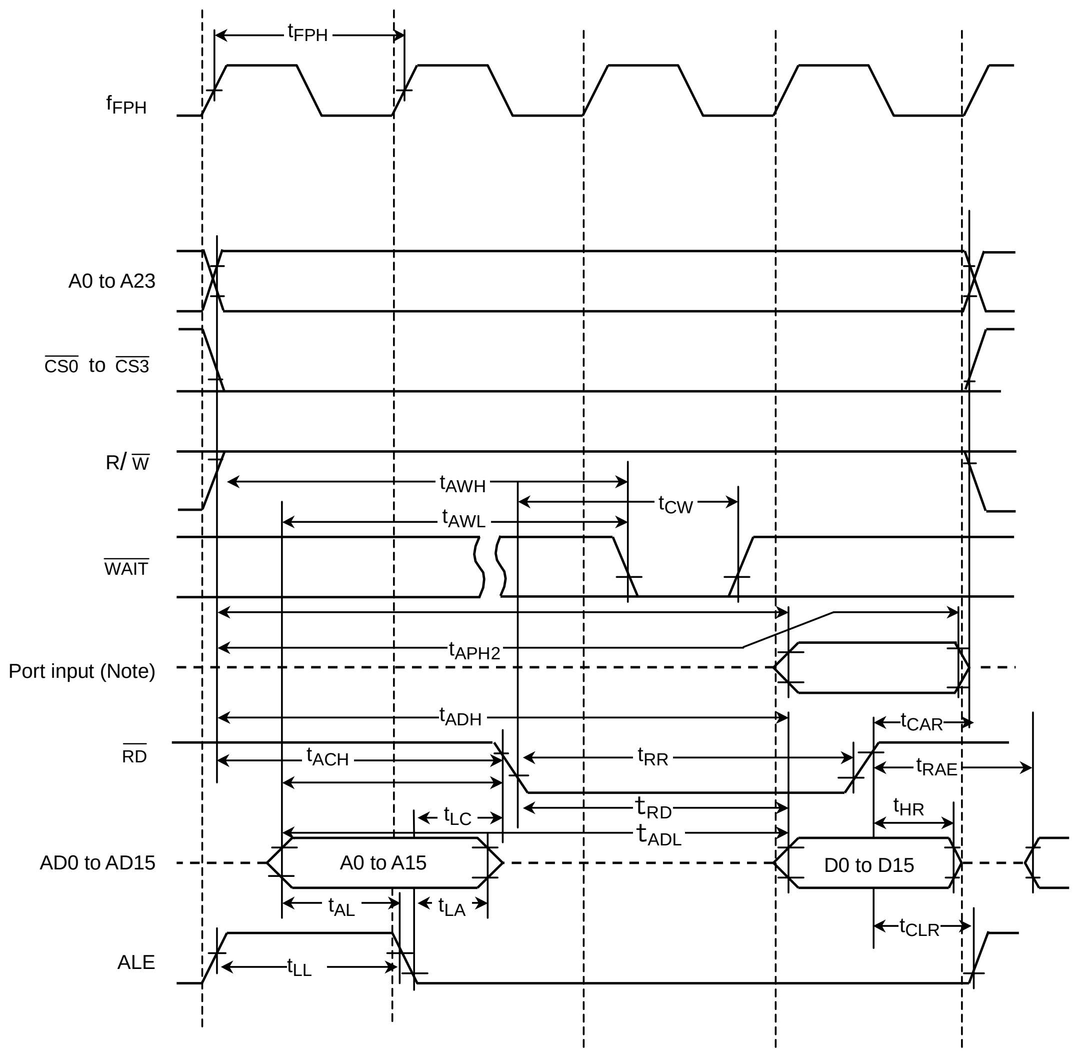
(1) $V_{CC} = 1.8$ to 2.6 V

No.	Parameter	Symbol	Equation		$f_{FPH} = 10$ MHz		Unit
			Min	Max	Min	Max	
1	f_{FPH} cycle period (= x)	t_{FPH}	100	250	100		ns
2	A0 to A15 valid to ALE low	t_{AL}	0.5x – 28		22		ns
3	A0 to A15 hold after ALE low	t_{LA}	0.5x – 35		15		ns
4	ALE pulse width high	t_{LL}	x – 40		60		ns
5	ALE low to $\overline{RD}$ or $\overline{WR}$ asserted	t_{LC}	0.5x – 28		22		ns
6	$\overline{RD}$ negated to ALE high	t_{CLR}	0.5x – 20		30		ns
7	$\overline{WR}$ negated to ALE high	t_{CLW}	x – 20		80		ns
8	A0 to A15 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACL}	x – 75		25		ns
9	A0 to A23 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACH}	1.5x – 70		80		ns
10	A0 to A23 hold after $\overline{RD}$ negated	t_{CAR}	0.5x – 30		20		ns
11	A0 to A23 hold after $\overline{WR}$ negated	t_{CAW}	x – 30		70		ns
12	A0 to A15 valid to D0 to D15 data in	t_{ADL}		3.0x – 76		224	ns
13	A0 to A23 valid to D0 to D15 data in	t_{ADH}		3.5x – 82		268	ns
14	$\overline{RD}$ asserted to D0 to D15 data in	t_{RD}		2.0x – 60		140	ns
15	$\overline{RD}$ width low	t_{RR}	2.0x – 30		170		ns
16	D0 to D15 hold after $\overline{RD}$ negated	t_{HR}	0		0		ns
17	$\overline{RD}$ negated to next A0 to A15 output	t_{RAE}	x – 30		70		ns
18	$\overline{WR}$ width low	t_{WW}	1.5x – 30		120		ns
19	D0 to D15 valid to $\overline{WR}$ negated	t_{DW}	1.5x – 70		80		ns
20	D0 to D15 hold after $\overline{WR}$ negated	t_{WD}	x – 50		50		ns
21	A0 to A23 valid to $\overline{WAIT}$ input [(1 + N) wait states]	t_{AWH}		3.5x – 120		230	ns
22	A0 to A15 valid to $\overline{WAIT}$ input [(1 + N) wait states]	t_{AWL}		3.0x – 100		200	ns
23	$\overline{WAIT}$ hold after $\overline{RD}$ or $\overline{WR}$ asserted [(1 + N) wait states]	t_{CW}	2.0x + 0		200		ns
24	A0 to A23 valid to port data in	t_{APH}		3.5x – 170		180	ns
25	Port data hold after A0 to A23 valid	t_{APH2}	3.5x		350		ns
26	A0 to A23 valid to port data valid	t_{AP}		3.5x + 170		520	ns

AC measurement conditions:
- Output levels: High 0.7 × Vcc/Low 0.3 × Vcc, CL = 50 pF
- Input levels: High 0.9 × Vcc/Low 0.1 × Vcc

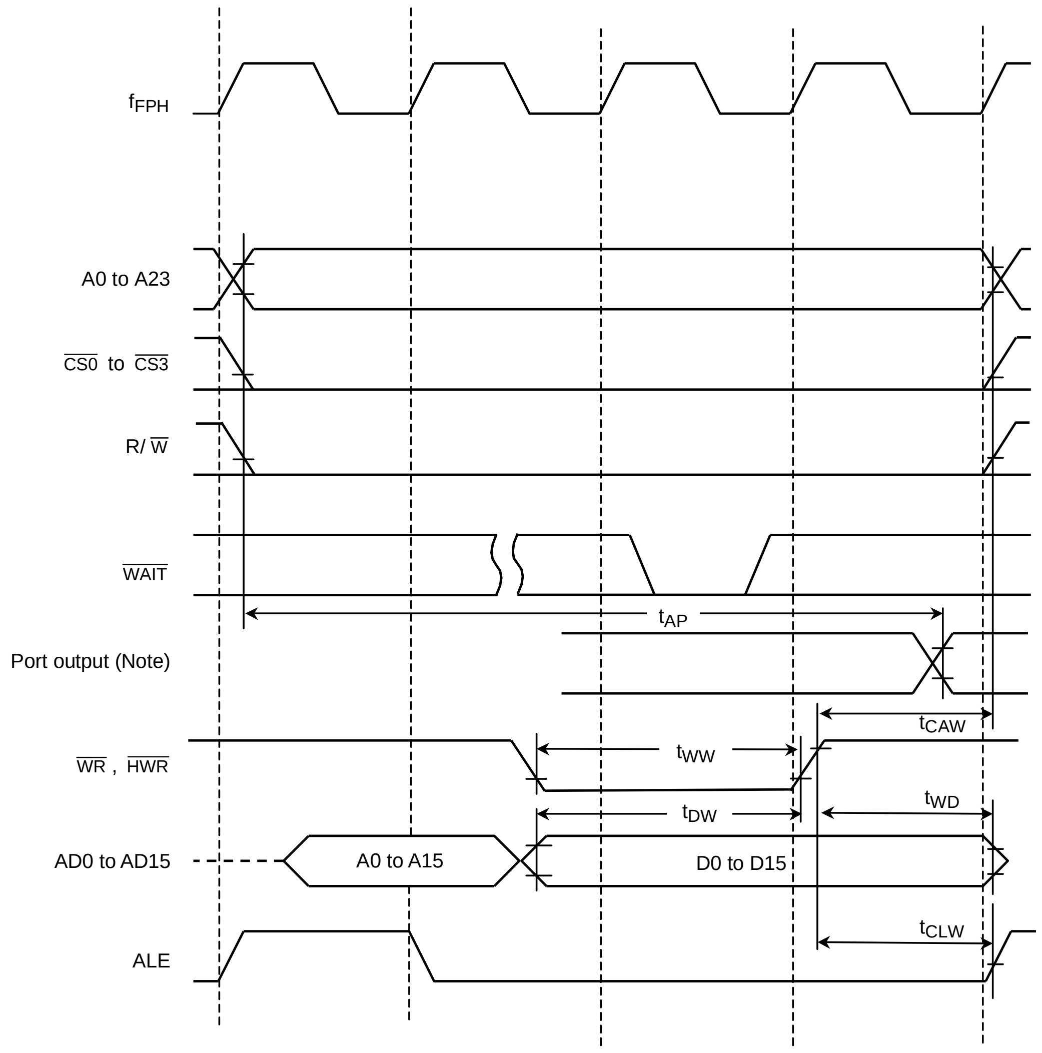
Note: In the above table, the letter x represents the f_{FPH} cycle period, which is half the system clock (f_{SYS}) cycle period used in the CPU core.
The f_{FPH} cycle period varies, depending on the programming of the clock gear function.

(2) Read operation timing



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.

(3) Write operation timing



Note: Since the CPU accesses the internal area to write data to a port, the control signals of external pins such as $\overline{WR}$ 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.

4.4 ADC Electrical Characteristics

AVcc = Vcc, AVss = Vss

Parameter		Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)		VREFH	V_{CC} = 1.8 to 2.6 V	Vcc	Vcc	Vcc	V
Analog reference voltage (−)		VREFL	V_{CC} = 1.8 to 2.6 V	Vss	Vss	Vss	
Analog input voltage		VAIN		VREFL		VREFH	
Analog supply current	ADMOD1.VREFON = 1	IREF	V_{CC} = 1.8 to 2.6 V		0.65	1.0	mA
	ADMOD1.VREFON = 0		V_{CC} = 1.8 to 2.6 V		0.02	5.0	μA
Total error (Not including quantization error)		−	V_{CC} = 1.8 to 2.6 V		±1.0	±4.0	LSB

Note 1: 1 LSB = (VREFH − VREFL)/1024 (V)

Note 2: Minimum operating frequency
Guaranteed when the frequency of the clock selected with the clock gear is 4 MHz or higher with fc used.

Note 3: The supply current flowing through the AV_{CC} pin is included in the VCC pin supply current parameter (I_{CC}).

4.5 SIO Timing (I/O interface mode)

Note: In the tables below, the letter x represents the f_{FPH} cycle period, which is half the system clock (f_{SYS}) cycle period used in the CPU core.
The f_{FPH} cycle period varies, depending on the programming of the clock gear function.

(1) SCLK input mode

Parameter	Symbol	Equation		10 MHz		Unit
		Min	Max	Min	Max	
SCLK period	t_{SCY}	16X		1.6		μs
TXD data to SCLK rise or fall*	t_{OSS}	t_{SCY}/2 − 4X − 180 (V_{CC} = 2V ± 10%)		220		ns
TXD data hold after SCLK rise or fall*	t_{OHS}	t_{SCY}/2 + 2X + 0		1000		ns
RXD data hold after SCLK rise or fall*	t_{HSR}	3X + 10		310		ns
SCLK rise or fall* to RXD data valid	t_{SRD}		t_{SCY} − 0		1600	ns
RXD data valid to SCLK rise or fall*	t_{RDS}	0		0		ns

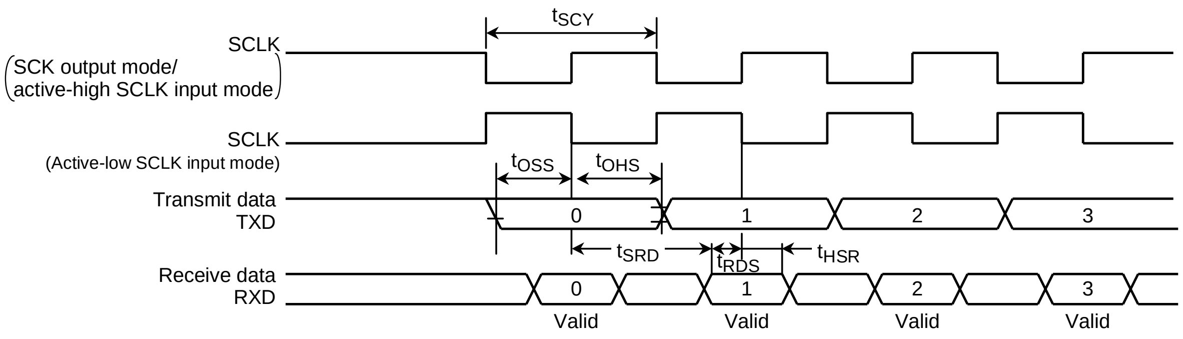
SCLK rise or fall*: Measured relative to the programmed active edge of SCLK.

Note: The values shown in the "10 MHz" column are measured with t_{SCY} = 16X.

(2) SCLK output mode

Parameter	Symbol	Equation		10 MHz		Unit
		Min	Max	Min	Max	
SCLK period	t_{SCY}	16X	8192X	1.6	819	μs
TXD data to SCLK rise or fall*	t_{OSS}	$t_{SCY}/2 - 40$		760		ns
TXD data hold after SCLK rise or fall*	t_{OHS}	$t_{SCY}/2 - 40$		760		ns
RXD data hold after SCLK rise or fall*	t_{HSR}	0		0		ns
SCLK rise or fall* to RXD data valid	t_{SRD}		$t_{SCY} - 1X - 180$		1320	ns
RXD data valid to SCLK rise or fall*	t_{RDS}	1X + 180		280		ns

Note: The values shown in the “10 MHz” column are measured with $t_{SCY} = 16X$.



4.6 Event Counters (TA0IN, TB0IN0, TB0IN1, TB1IN0, TB1IN1)

Parameter	Symbol	Equation		10 MHz		Unit
		Min	Max	Min	Max	
Clock cycle period	t_{VCK}	8X + 100		900		ns
Clock low pulse width	t_{VCKL}	4X + 40		440		ns
Clock high pulse width	t_{VCKH}	4X + 40		440		ns

Note: In the above table, the letter x represents the f_{FPH} cycle period, which is half the system clock (f_{SYS}) cycle period used in the CPU core.
The f_{FPH} cycle period varies, depending on the programming of the clock gear function.

4.7 Interrupt and Timer Capture

Note: In the tables below, the letter x represents the f_{FPH} cycle period, which is half the system clock (f_{SYS}) cycle period used in the CPU core.
The f_{FPH} cycle period varies, depending on the programming of the clock gear function.

(1) $\overline{\text{NMI}}$, and INT0 to INT4 interrupts

Parameter	Symbol	Equation		10 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for $\overline{\text{NMI}}$ and INT0 to INT4	t_{INTAL}	4X + 40		440		ns
High pulse width for $\overline{\text{NMI}}$ and INT0 to INT4	t_{INTAH}	4X + 40		440		ns

(2) INT5 to INT8 interrupts and capture

The input pulse widths for INT5 to INT8 vary with the selected system clock and prescaler clock. The following table shows the pulse widths for different operating clocks:

Selected Prescaler Clock PRCK[1:0]	t_{INTBL} (Low pulse width for INT5 to INT8)		t_{INTBH} (High pulse width for INT5 to INT8)		Unit
	Equation	f_{FPH} = 10 MHz	Equation	f_{FPH} = 10 MHz	
	Min	Min	Min	Min	
00 (f_{FPH})	8X + 100	900	8X + 100	900	ns
10 (fc/16)	128Xc + 0.1	12.9	128Xc + 0.1	12.9	μs

Note: Xc represents the cycle period of the high-speed oscillator clock (fc).

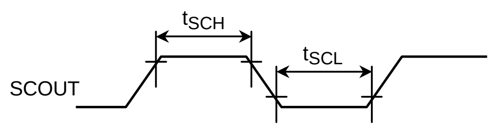
4.8 SCOUT Pin

Parameter	Symbol	Equation		10 MHz		Condition	Unit
		Min	Max	Min	Max		
Clock high pulse width	t_{SCH}	0.5T – 25		25		V_{CC} = 1.8 to 2.6 V	ns
Clock low pulse width	t_{SCL}	0.5T – 25		25		V_{CC} = 1.8 to 2.6 V	ns

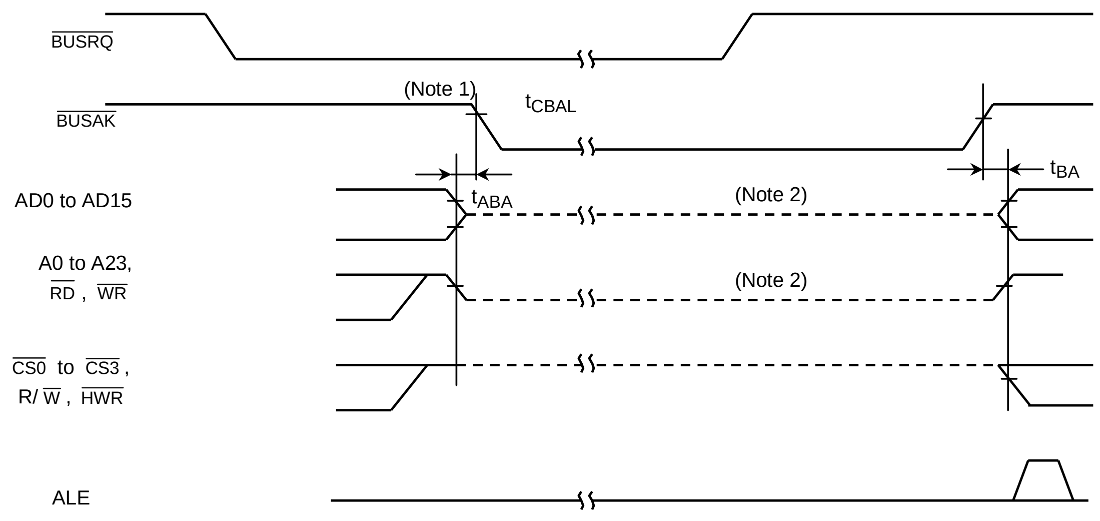
Note: In the table above, the letter T represents the cycle period of the SCOUT output clock.

Measurement condition:

- Output levels: High = 0.7 V_{CC}/Low = 0.3 V_{CC}, CL = 10 PF



4.9 Bus Request and Bus Acknowledge Signals


$\overline{\text{BUSRQ}}$
$\overline{\text{BUSAK}}$
(Note 1)
t_{CBAL}
AD0 to AD15
t_{ABA}
(Note 2)
t_{BA}
A0 to A23, $\overline{\text{RD}}$, $\overline{\text{WR}}$
(Note 2)
$\overline{\text{CS0}}$ to $\overline{\text{CS3}}$, R/ $\overline{\text{W}}$, $\overline{\text{HWR}}$
ALE

Parameter	Symbol	Equation		f_{FPH} = 10 MHz		Condition	Unit
		Min	Max	Min	Max		
Bus float to $\overline{\text{BUSAK}}$ asserted	t_{ABA}	0	300	0	300	V_{CC} = 1.8 to 2.6 V	ns
Bus float after $\overline{\text{BUSAK}}$ negated	t_{BAA}	0	300	0	300	V_{CC} = 1.8 to 2.6 V	ns

Note 1: If the current bus cycle has not terminated due to wait-state insertion, the TMP91CW28 does not respond to $\overline{\text{BUSRQ}}$ until the wait state ends.

Note 2: This broken lines indicate that output buffers are disabled, not that the signals are at indeterminate states. The pin holds the last logic value present at that pin before the bus is relinquished. This is dynamically accomplished through external load capacitances. The equipment manufacturer may maintain the bus at a predefined state by means of off-chip resistors, but he or she should design, considering the time (Determined by the CR constant) it takes for a signal to reach a desired state. The on-chip, integrated programmable pull-up/pull-down resistors remain active, depending on internal signal states.

4.10 Recommended Oscillator Circuit

The TMP91CW28 is evaluated by the following resonator manufacturer. The results of evaluation are shown below.

Note: The additional capacitance of the resonator connecting pins are the sum of load capacitance C1, C2 and the stray capacitance on the target board. Even when recommended constants for C1 and C2 are used, actual load capacitance may vary with the board, possibly resulting in the malfunction of the oscillator. The board should be designed so that the patterns around the oscillator are as short as possible. Toshiba recommends that the resonator be finally evaluated after it is mounted on the target board.

(1) Sample crystal circuit

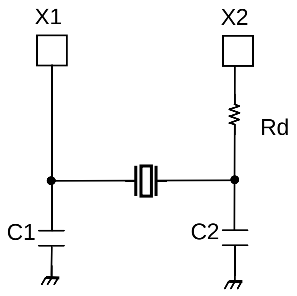


Figure 4.10.1 High-frequency Oscillator Connection Diagram

(2) Recommended ceramic resonators for the TMP91CW28, manufactured by Murata Manufacturing Co., Ltd.

Ta = −20 to 70°C

Component	Oscillating Frequency (MHz)	Recommended Resonator	Recommended Constants C1 [pF]	C2 [pF]	Rd [kΩ]	VCC[V]	Remarks
High-speed oscillator	4.0	CSTCR4M00G55-R0	(39)	(39)	0	1.8 to 2.6	−
		CSTLS4M00G56-B0	(47)	(47)			
	8.0	CSTCE8M00G55-R0	(33)	(33)			
		CSTLS8M00G56-B0	(47)	(47)			
	10.0	CSTCE10M0G52-R0	(10)	(10)			
		CSTLS10M0G53-B0	(15)	(15)			

- The C1 and C2 constants are enclosed in parentheses for resonator models having built-in capacitors.
- The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change.
  For up-to-date information, please refer to the following URL:
  http://www.murata.co.jp/search/index.html

5. Special Function Register Summary

The special function registers (SFRs) configure and access the I/O ports, and control on-chip functions. These registers occupy 4-Kbyte addresses from 000000H through 000FFFH.

(1) I/O ports

(2) I/O port control

(3) Interrupt control

(4) Chip select/wait controller

(5) Clock control

(6) 8-bit timer control

(7) 16-bit timer control

(8) UART serial channel

(9) I^{2}C bus serial bus interface

(10) AD converter control

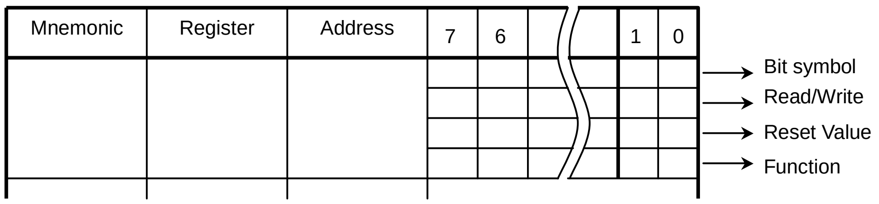
(11) Watchdog timer

(12) Key wakeup

(13) BCD adder/subtractor

(14) Program patch logic

Table Organization



* In the following tables, “RMW prohibited” indicates that the register does not support the use of a read-modify-write instruction.

Example: When setting only bit0 in the P0CR register to 1, the “SET 0, (0002H)” instruction is usually used. That is not, however, allowed because RMW is prohibited for the P0CR. Instead, the LD (Transfer) instruction must be used to write to 8 bits.

Access

R/W:	Read/write. The user can read and write the register bit.
R:	Read only.
W:	Write only.
W*:	The user can read and write the register bit, but a read always returns a value of 1.
RMW prohibited:	The user cannot perform a read-modify-write instruction (EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD, and RRD).
*R/W:	The user cannot use a read-modify-write instruction to control the pull-up resistor for the port.

Table 5.1 SFR Address Map (1)

[1] PORT

Address	Mnemonic
0000H	P0
1H	P1
2H	P0CR
3H	
4H	P1CR
5H	P1FC
6H	P2
7H	P3
8H	P2CR
9H	P2FC
AH	P3CR
BH	P3FC
CH	P4
DH	P5
EH	P4CR
FH	P4FC

Address	Mnemonic
0010H	
1H	
2H	P6
3H	P7
4H	P6CR
5H	P6FC
6H	P7CR
7H	P7FC
8H	P8
9H	P9
AH	P8CR
BH	P8FC
CH	P9CR
DH	P9FC
EH	PA
FH	

Address	Mnemonic
0020H	PACR
1H	PAFC
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	PUP
FH	ODE

[2] INTC

Address	Mnemonic
0080H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	
5H	
6H	
7H	
8H	INTCLR
9H	DMAR
AH	DMAB
BH	
CH	IIMC
DH	
EH	
FH	

Address	Mnemonic
0090H	INTE0AD
1H	INTE12
2H	INTE34
3H	INTE56
4H	INTE78
5H	INTETA01
6H	INTETA23
7H	
8H	
9H	INTETB0
AH	INTETB1
BH	INTETB01V
CH	INTEBCD
DH	INTES1
EH	INTES2
FH	

Address	Mnemonic
00A0H	INTETC01
1H	INTETC23
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[3] CS/WAIT

Address	Mnemonic
00C0H	B0CS
1H	B1CS
2H	B2CS
3H	B3CS
4H	
5H	
6H	
7H	BEXCS
8H	MSAR0
9H	MAMR0
AH	MSAR1
BH	MAMR1
CH	MSAR2
DH	MAMR2
EH	MSAR3
FH	MAMR3

Note: Only the addresses with mnemonics shown in the tables can be accessed.

Table 5.2 SFR Address Map (2)

[4] CGEAR,DFM

Address	Mnemonic
00E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[5] TMRA

Address	Mnemonic
0100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA1FFCR
6H	
7H	
8H	TA23RUN
9H	
AH	TA2REG
BH	TA3REG
CH	TA23MOD
DH	TA3FFCR
EH	
FH	

[6] TMRB

Address	Mnemonic
0180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

Address	Mnemonic
0190H	TB1RUN
1H	
2H	TB1MOD
3H	TB1FFCR
4H	
5H	
6H	
7H	
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

Note: Only the addresses with mnemonics shown in the tables can be accessed.

Table 5.3 SFR Address Map (3)

[7] UART/SIO

Address	Mnemonic
0200H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	

[8] I^2C bus/SIO

Address	Mnemonic
0240H	SBI0CR1
1H	SBI0DBR
2H	I2C0AR
3H	SBI0CR2/SBI0SR
4H	SBI0BR0
5H	SBI0BR1
6H	
7H	
8H	SBI1CR1
9H	SBI1DBR
AH	I2C1AR
BH	SBI1CR2/SBI1SR
CH	SBI1BR0
DH	SBI1BR1
EH	
FH	

[9] 10-bit ADC

Address	Mnemonic
02A0H	ADREG04L
1H	ADREG04H
2H	ADREG15L
3H	ADREG15H
4H	ADREG26L
5H	ADREG26H
6H	ADREG37L
7H	ADREG37H
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
02B0H	ADMOD0
1H	ADMOD1
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] WDT

Address	Mnemonic
0300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Only the addresses with mnemonics shown in the tables can be accessed.

Table 5.4 SFR Address Map (4)

[11] Key wakeup

Address	Mnemonic
03A0H	KWIEN
1H	KWICR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12] BCD adder/subtractor

Address	Mnemonic
03B0H	BCDMINA
1H	BCDSECA
2H	BCDFRAA
3H	
4H	BCDMINB
5H	BCDSECB
6H	BCDFRAB
7H	
8H	BCDMINR
9H	BCDSECR
AH	BCDFRAR
BH	
CH	BCDCR
DH	
EH	
FH	

[13] Program patch logic

Address	Mnemonic
0400H	ROMCMP00
1H	ROMCMP01
2H	ROMCMP02
3H	
4H	ROMSUB0L
5H	ROMSUB0H
6H	
7H	
8H	ROMCMP10
9H	ROMCMP11
AH	ROMCMP12
BH	
CH	ROMSUB1L
DH	ROMSUB1H
EH	
FH	

Address	Mnemonic
0410H	ROMCMP20
1H	ROMCMP21
2H	ROMCMP22
3H	
4H	ROMSUB2L
5H	ROMSUB2H
6H	
7H	
8H	ROMCMP30
9H	ROMCMP31
AH	ROMCMP32
BH	
CH	ROMSUB3L
DH	ROMSUB3H
EH	
FH	

Address	Mnemonic
0420H	ROMCMP40
1H	ROMCMP41
2H	ROMCMP42
3H	
4H	ROMSUB4L
5H	ROMSUB4H
6H	
7H	
8H	ROMCMP50
9H	ROMCMP51
AH	ROMCMP52
BH	
CH	ROMSUB5L
DH	ROMSUB5H
EH	
FH	

Note: Only the addresses with mnemonics shown in the tables can be accessed.

(1) Input/output ports

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
P0	Port 0	00H	P07	P06	P05	P04	P03	P02	P01	P00
			R/W							
			Data from external port (Output latch register is undefined)							
P1	Port 1	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to 0)							
P2	Port 2	06H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			Data from external port (Output latch register is set to 1)							
P3	Port 3	07H	P37	P36	P35	P34	P33	P32	P31	P30
			*R/W							
			Data from external port (Output latch register is set to 1)						1	1
			0: Pull-up resistor disabled 1: Pull-up resistor enabled						–	
P4	Port 4	0CH					P43	P42	P41	P40
							*R/W			
							Data from external port (Output latch register is set to 1)			
							0: Pull-up resistor disabled 1: Pull-up resistor enabled			
P5	Port 5	0DH	P57	P56	P55	P54	P53	P52	P51	P50
			R							
			Data from external port							
P6	Port 6	12H		P66	P65	P64	P63	P62	P61	P60
				R/W						
				Data from external port (Output latch register is set to 1)						
				–				0: Pull-up resistor disabled 1: Pull-up resistor enabled		–
P7	Port 7	13H			P75	P74	P73	P72	P71	P70
					R/W					
					Data from external port (Output latch register is set to 1)					
					0: Pull-up resistor disabled 1: Pull-up resistor enabled					
P8	Port 8	18H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			Data from external port (Output latch register is set to 1)							
			0: Pull-up resistor disabled 1: Pull-up resistor enabled							
P9	Port 9	19H		P96	P95	P94	P93	P92	P91	P90
				R/W						
				Data from external port (output latch register is set to 1)						
				–				0: Pull-up resistor disabled 1: Pull-up resistor enabled		–
PA	Port A	1EH	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R/W							
			Data from external port (Output latch register is set to 1)							
			0: Pull-up resistor disabled 1: Pull-up resistor enabled							

(2) Input/output port control (1/2)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
P0CR	Port 0 control	02H (RMW prohibited)	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P1CR	Port 1 control	04H (RMW prohibited)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P1FC	Port 1 function	05H (RMW prohibited)	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
			W							
			0	0	0	0	0	0	0	0
			P1FC/P1CR = 00: Input port, 01: Output port, 10: AD15 to AD8, 11: A15 to A8							
P2CR	Port 2 control	08H (RMW prohibited)	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P2FC	Port 2 function	09H (RMW prohibited)	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
			W							
			0	0	0	0	0	0	0	0
			P2FC/P2CR = 00: Input port, 01: Output port, 10: A7 to A0, 11: A23 to A16							
P3CR	Port 3 control	0AH (RMW prohibited)	P37C	P36C	P35C	P34C	P33C	P32C		
			W							
			0	0	0	0	0	0		
			0: Input 1: Output							
P3FC	Port 3 function	0BH (RMW prohibited)	–	P36F	P35F	P34F		P32F	P31F	P30F
			W					W		
			0	0	0	0		0	0	0
			Must be written as "0".	0: Port 1: R/$\overline{W}$	0: Port 1: $\overline{BUSAK}$	0: Port 1: $\overline{BUSRQ}$		0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$
P4CR	Port 4 control	0EH (RMW prohibited)					P43C	P42C	P41C	P40C
							W			
							0	0	0	0
							0: Input 1: Output			
P4FC	Port 4 function	0FH (RMW prohibited)					P43F	P42F	P41F	P40F
							W			
							0	0	0	0
							0: Port 1: $\overline{CS3}$	0: Port 1: $\overline{CS2}$	0: Port 1: $\overline{CS1}$	0: Port 1: $\overline{CS0}$

Note: Writing 0 to the P3.P30 bit and 1 to the P3FC.P30F bit causes the P30 to be driven low also when on-chip address space is accessed.

Input/output port control (2/2)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
P6CR	Port 6 control	14H (RMW prohibited)		P66C	P65C	P64C	P63C	P62C	P61C	P60C
				W						
				0	0	0	0	0	0	0
				0: Input 1: Output						
P6FC	Port 6 function	15H (RMW prohibited)				P64F	P63F	P62F	P61F	P60F
						W				
						0	0	0	0	0
						0: Port 1: SCOUT	0: Port 1: INT0	0: Port 1: SCL0	0: Port 1: SDA0/SO0	0: Port 1: SCK0
P7CR	Port 7 control	16H (RMW prohibited)			P75C	P74C	P73C	P72C	P71C	P70C
					W					
					0	0	0	0	0	0
					0: Input 1: Output					
P7FC	Port 7 function	17H (RMW prohibited)						P72F	P71F	
								W		
								0	0	
								0: Port 1: TA3OUT	0: Port 1: TA1OUT	
P8CR	Port 8 control	1AH (RMW prohibited)	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P8FC	Port 8 function	1BH (RMW prohibited)	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: TB1OUT	0: Port 1: TB1OUT	0: Port 1: INT8/ TB1IN1	0: Port 1: INT7/ TB1IN0	0: Port 1: TB0OUT1	0: Port 1: TB0OUT0	0: Port 1: INT6/ TB0IN1	0: Port 1: INT5/ TB0IN0
P9CR	Port 9 control	1CH (RMW prohibited)		P96C	P95C	P94C	P93C	P92C	P91C	P90C
				W						
				0	0	0	0	0	0	0
				0: Input 1: Output						
P9FC	Port 9 function	1DH (RMW prohibited)			P95F		P93F	P92F	P91F	P90F
					W		W			
					0		0	0	0	0
					0: Port 1: SCLK		0: Port 1: TXD	0: Port 1: SCL1	0: Port 1: SDA1/ SO1	0: Port 1: SCK1
PACR	Port A control	20H (RMW prohibited)	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
PAFC	Port A function	21H (RMW prohibited)					PA3F	PA2F	PA1F	PA0F
							W			
							0	0	0	0
							INT1 to INT4 input enable			
PUP	Pull-up enable	2EH			PUP92	PUP91	PUP62	PUP61		
					R/W					
					1	1	1	1		
					0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable	0: Disable 1: Enable		
ODE	Serial open-drain enable	2FH			ODE92	ODE91	ODE62	ODE61	ODE93	
					R/W					
					0	0	0	0	0	
					1: P92ODE	1: P91ODE	1: P62ODE	1: P61ODE	1: P93ODE	

Note 1: External interrupt INT0
The P6FC.P63F bit enables input. The IIMC.I0LE and IIMC.I0EDGE bits control the interrupt sensitivity (High level, low level, rising edge or falling edge).

Note 2: External interrupts INT1 to INT4
The PAFC.PA3F to PAFC.PA0F bits enable input. The IIMC.I4EDGE to IIMC.I1EDGE bits control the edge polarity (Rising or falling).

Note 3: External interrupts INT5 to INT8
The P85F, P84F, P81F, and P80F bits of the P8FC enable input. The TB0MOD and TB1MOD registers (TMRB registers) control the edge polarity.

(3) Interrupt control (1/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	Interrupt enable 0 & AD	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTAD	Interrupt priority level			1: INT0	Interrupt priority level		
INTE12	Interrupt enable 2/1	91H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT2	Interrupt priority level			1: INT1	Interrupt priority level		
INTE34	Interrupt enable 4/3	92H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT4	Interrupt priority level			1: INT3	Interrupt priority level		
INTE56	Interrupt enable 6/5	93H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT6	Interrupt priority level			1: INT5	Interrupt priority level		
INTE78	Interrupt enable 8/7	94H	INT8				INT7			
			I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT8	Interrupt priority level			1: INT7	Interrupt priority level		
INTETA01	Interrupt enable timer A 1/0	95H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA1	Interrupt priority level			1: INTTA0	Interrupt priority level		
INTETA23	Interrupt enable timer A 3/2	96H	INTTA3 (TMRA5)				INTTA2 (TMRA4)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA3	Interrupt priority level			1: INTTA2	Interrupt priority level		

Interrupt control (2/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
INTETB0	Interrupt enable TMRB0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB01	Interrupt priority level			1: INTTB00	Interrupt priority level		
INTETB1	Interrupt enable TMRB1	9AH	INTTB11 (TMRB1)				INTTB10 (TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB11	Interrupt priority level			1: INTTB10	Interrupt priority level		
INTETB01V	Interrupt enable TMRB0/1 (Overflow)	9BH	INTTBOF1 (TMRB1 overflow)				INTTBOF0 (TMRB0 overflow)			
			ITF1C	ITF1M2	ITF1M1	ITF1M0	ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTBOF1	Interrupt priority level			1: INTTBOF0	Interrupt priority level		
INTEBCD	Interrupt enable BCD	9CH					INTBCD			
							IBCDC	IBCDM2	IBCDM1	IBCDM0
							R	R/W		
							0	0	0	0
							1: INTBCD	Interrupt priority level		
INTES1	Interrupt enable serial 1	9DH	INTTX				INTRX			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX0	Interrupt priority level			1: INTRX0	Interrupt priority level		
INTES2	Interrupt enable SBI 0/1	9EH	INTSBI1				INTSBI0			
			IS1C	IS1M2	IS1M1	IS1M0	IS0C	IS0M2	IS0M1	IS0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTSBI1	Interrupt priority level			1: INTSBI0	Interrupt priority level		
INTETC01	Interrupt enable TC0/1	A0H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	Interrupt enable TC2/3	A1H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt control (3/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA 0 request vector	80H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 startup vector					
DMA1V	DMA 1 request vector	81H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 startup vector					
DMA2V	DMA 2 request vector	82H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 startup vector					
DMA3V	DMA 3 request vector	83H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 startup vector					
INTCLR	Interrupt clear control	88H (RMW prohibited)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					0	0	0	0	0	0
					Write the DMA startup vector to clear an interrupt.					
DMAR	DMA software request register	89H					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA soft request			
DMAB	DMA burst request register	8AH					DMAB3	DMAB2	DMAB1	DMAB0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA burst request			
IIMC	Interrupt input mode control	8CH (RMW prohibited)	–	I4EDGE	I3EDGE	I2EDGE	I1EDGE	I0EDGE	I0LE	NMIREE
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			Must be written as “0”.	INT4 edge polarity 0: Rising 1: Falling	INT3 edge polarity 0: Rising 1: Falling	INT2 edge polarity 0: Rising 1: Falling	INT1 edge polarity 0: Rising 1: Falling	INT0 edge polarity 0: Rising 1: Falling	INT0 sensitivity 0: Edge-triggered 1: Level-sensitive	1: Also triggered by $\overline{\text{NMI}}$ rising edge

(4) Chip select/wait controller (1/2)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
B0CS	Block 0 CS/WAIT control register	C0H (RMW prohibited)	B0E		B0OM1	B0OM0	B0BUS	B0W2	B0W1	B0W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 0: 16 bits 1: 8 bits	000: 2 wait states 1xx: Reserved 001: 1 wait state 010: (1 + N) wait states 011: 0 wait states		
B1CS	Block 1 CS/WAIT control register	C1H (RMW prohibited)	B1E		B1OM1	B1OM0	B1BUS	B1W2	B1W1	B1W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 0: 16 bits 1: 8 bits	000: 2 wait states 1xx: Reserved 001: 1 wait state 010: (1 + N) wait states 011: 0 wait states		
B2CS	Block 2 CS/WAIT control register	C2H (RMW prohibited)	B2E	B2M	B2OM1	B2OM0	B2BUS	B2W2	B2W1	B2W0
			W	W	W	W	W	W	W	W
			1	0	0	0	0	0	0	0
			0: Disable 1: Enable	0: Whole 16-Mbyte space 1: CS space	00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 0: 16 bits 1: 8 bits	000: 2 wait states 1xx: Reserved 001: 1 wait state 010: (1 + N) wait states 011: 0 wait states		
B3CS	Block 3 CS/WAIT control register	C3H (RMW prohibited)	B3E		B3OM1	B3OM0	B3BUS	B3W2	B3W1	B3W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: Reserved 10: Reserved 11: Reserved		Data bus width 0: 16 bits 1: 8 bits	000: 2 wait states 1xx: Reserved 001: 1 wait state 010: (1 + N) wait states 011: 0 wait states		
BEXCS	External CS/WAIT control register	C7H (RMW prohibited)					BEXBUS	BEXW2	BEXW1	BEXW0
							W	W	W	W
							0	0	0	0
							Data bus width 0: 16 bits 1: 8 bits	000: 2 wait states 1xx: Reserved 001: 1 wait state 010: (1 + N) wait states 011: 0 wait states		
MSAR0	Memory start address register 0	C8H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Set A23 to A16 of the start address							
MAMR0	Memory address mask register 0	C9H	V20	V19	V18	V17	V16	V15	V14 to V9	V8
			R/W							
			1	1	1	1	1	1	1	1
			CS0 space size 0: Bit to be compared							
MSAR1	Memory start address register 1	CAH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Set A23 to A16 of the start address							
MAMR1	Memory address mask register 1	CBH	V21	V20	V19	V18	V17	V16	V15 to V9	V8
			R/W							
			1	1	1	1	1	1	1	
			CS1 space size 0: Bit to be compared							

Chip select/wait controller (2/2)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
MSAR2	Memory start address register 2	CCH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Set A23 to A16 of the start address							
MAMR2	Memory address mask register 2	CDH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS2 space size 0: Bit to be compared							
MSAR3	Memory start address register 3	CEH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Set A23 to A16 of the start address							
MAMR3	Memory address mask register 3	CFH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS3 space size 0: Bit to be compared							

(5) Clock control

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register 0	E0H	–	–	–	–	–	–	PRCK1	PRCK0
			W					R/W		
			1	0	1	0	0	0	0	0
			Must be written as “1”.	Must be written as “0”.	Must be written as “1”.	Must be written as “0”.	Must be written as “0”.	Must be written as “0”.	Prescaler clock select 00: f_{FPH} 01: Reserved 10: fc/16 11: Reserved	
SYSCR1	System clock control register 1	E1H					–	GEAR2	GEAR1	GEAR0
							W	R/W		
							0	1	0	0
							Must be written as “0”.	High-speed clock gear select 000: High-speed clock 001: High-speed clock/2 010: High-speed clock/4 011: High-speed clock/8 100: High-speed clock/16 Others: Reserved		
SYSCR2	System clock control register 2	E2H		SCOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
				R/W	R/W	R/W	R/W	R/W		R/W
				0	1	0	1	1		0
				0: Low level 1: f_{FPH}	Oscillator warm-up time 00: Reserved 01: 2^8/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			1: Pins are driven in STOP mode
EMCCR0	EMC control register 0	E3H	PROTECT	–	–	–	ALEEN	EXTIN	–	–
			R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	1	0	0	0	1	1
			Protection flag 0: Disabled 1: Enabled	Must be written as “0”.	Must be written as “1”.	Must be written as “0”.	1: ALE output enabled	1: External clock used as fc	Must be written as “1”.	Must be written as “1”.
EMCCR1	EMC control register 1	E4H	On writes: 1FH: Protection disabled Other than 1FH: Protection enabled							

Note: Enabling protection using the EMCCR1 register prevents writes to the following SFRs:

1. Chip select/wait controller
   B0CS, B1CS, B2CS, B3CS, BEXCS,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3
2. Clock gear (Only the EMCCR1 can be written.)
   SYSCR0, SYSCR1, SYSCR2, EMCCR0

(6) 8-bit timer control

(6–1) TMRA01

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
TA01RUN	Timer run	100H	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double buffering 0: Disable 1: Enable				IDLE2 0: OFF 1: ON	8-bit timer run/stop control 0: Stop and clear 1: Run		
TA0REG	8-bit timer register 0	102H (RMW prohibited)	–							
			W							
			Undefined							
TA1REG	8-bit timer register 1	103H (RMW prohibited)	–							
			W							
			Undefined							
TA01MOD	8-bit timer source clock & mode	104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operating mode 00: 8-bit interval timer 01:16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM period 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA1 clock source 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA0 clock source 00: TA0IN input 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	8-bit timer flip-flop control	105H (RMW prohibited)					TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
							R/W		R/W	
							1	1	0	0
							00: Toggles TA1FF 01: Sets TA1FF to 1 10: Clears TA1FF to 0 11: Don’t care		1: TA1FF toggle enable	TA1FF toggle trigger 0: TMRA0 1: TMRA1

(6–2) TMRA23

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
TA23RUN	Timer run	108H	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double buffering 0: Disable 1: Enable				IDLE2 0: OFF 1: ON	8-bit timer run/stop control 0: Stop and clear 1: Run		
TA2REG	8-bit timer register 0	10AH (RMW prohibited)	–							
			W							
			Undefined							
TA3REG	8-bit timer register 1	10BH (RMW prohibited)	–							
			W							
			Undefined							
TA23MOD	8-bit timer source clock & mode	10CH	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operating mode 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM period 00: Reserved 01: 2^6 10: 2^7 11: 2^8		TMRA3 clock source 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA2 clock source 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	8-bit timer flip-flop control	10DH (RMW prohibited)					TA3FFC1	TA3FFC0	TA3FFIE	TA3FFIS
							R/W		R/W	
							1	1	0	0
							00: Toggles TA3FF 01: Sets TA3FF to 1 10: Clears TA3FF to 0 11: Don’t care		1: TA3FF toggle enable	TA3FF toggle trigger 0: TMRA2 1: TMRA3

(7) 16-bit timer control (1/2)

(7–1) TMRB0

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	Timer run	180H	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffering 0: Disable 1: Enable	Must be written as “0”.			IDLE2 0: OFF 1: ON	16-bit timer run/stop control 0: Stop and clear 1: Run		
TB0MOD	16-bit timer source clock & mode	182H (RMW prohibited)	TB0CT1	TB0ET1	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB0FF1 toggle trigger 0: Trigger disabled 1: Trigger enabled		0: Soft capture 1: Undefined	Capture triggers (TB0IN0, TB0IN1) 00: Disabled 01: ↑, ↑ 10: ↑, ↓ 11: ↑, ↓ (TA1OUT)		UC0 clear control 1: Enable	TMRB0 clock source 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
			When latches UC0 value into capture register 1	Upon a match with timer register 1						
TB0FFCR	16-bit timer flip-flop control	183H (RMW prohibited)	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	0	0
			TB0FF1 control 00: Invert 01: Set 10: Clear 11: Don’t care * Always read “11”.		TB0FF0 toggle trigger 0: Trigger disabled 1: Trigger enabled				TB0FF0 control 00: Invert 01: Set 10: Clear 11: Don’t care * Always read “11”.	
					When the up counter value is latched into TB0CP1	When the up counter value is latched into TB0CP0	When the up counter value reaches the TB0RG1 value	When the up counter value reaches the TB0RG0 value		
TB0RG0L	16-bit timer register 0 low	188H (RMW prohibited)	–							
			W							
			Undefined							
TB0RG0H	16-bit timer register 0 high	189H (RMW prohibited)	–							
			W							
			Undefined							
TB0RG1L	16-bit timer register 1 low	18AH (RMW prohibited)	–							
			W							
			Undefined							
TB0RG1H	16-bit timer register 1 high	18BH (RMW prohibited)	–							
			W							
			Undefined							
TB0CP0L	Capture register 0 low	18CH	–							
			R							
			Undefined							
TB0CP0H	Capture register 0 high	18DH	–							
			R							
			Undefined							
TB0CP1L	Capture register 1 low	18EH	–							
			R							
			Undefined							
TB0CP1H	Capture register 1 high	18FH	–							
			R							
			Undefined							

16-bit timer control (2/2)

(7–2) TMRB1

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
TB1RUN	Timer run	190H	TB1RDE	–			I2TB1	TB1PRUN		TB1RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffering 0: Disable 1: Enable	Must be written as "0".			IDLE2 0: OFF 1: ON	16-bit timer run/stop control 0: Stop and clear 1: Run		
TB1MOD	16-bit timer source clock & mode	192H (RMW prohibited)	TB1CT1	TB1ET1	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB1FF1 toggle trogger 0: Trigger disabled 1: Trigger enabled		0: Soft capture 1: Undefined	Capture triggers (TB0IN0, TB0IN1) 00: Disabled 01: ↑, ↑ 10: ↑, ↓ 11: ↑, ↓ (TA1OUT)		UC0 clear control 1: Enable	TMRB1 clock source 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
			When latches UC0 value into capture register 1	Upon a match with timer register 1						
TB1FFCR	16-bit timer flip-flop control	193H (RMW prohibited)	TB1FF1C1	TB1FF1C0	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	0	0
			TB1FF1 control 00: Invert 01: Set 10: Clear 11: Don't care * Always read "11".		TB1FF0 toggle trigger 0: Trigger disabled 1: Trigger enabled				TB1FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * Always read "11".	
					When the up counter value is latched into TB1CP1	When the up counter value is latched into TB1CP0	When the up counter value reaches the TB1RG1 value 1	When the up counter value reaches the TB1RG0 value 1		
TB1RG0L	16-bit timer register 0 low	198H (RMW prohibited)	–							
			W							
			Undefined							
TB1RG0H	16-bit timer register 0 high	199H (RMW prohibited)	–							
			W							
			Undefined							
TB1RG1L	16-bit timer register 1 low	19AH (RMW prohibited)	–							
			W							
			Undefined							
TB1RG1H	16-bit timer register 1 high	19BH (RMW prohibited)	–							
			W							
			Undefined							
TB1CP0L	Capture register 0 low	19CH	–							
			R							
			Undefined							
TB1CP0H	Capture register 0 high	19DH	–							
			R							
			Undefined							
TB1CP1L	Capture register 1 low	19EH	–							
			R							
			Undefined							
TB1CP1H	Capture register 1 high	19FH	–							
			R							
			Undefined							

(8) UART serial channel

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial channel 1 buffer	208H (RMW prohibited)	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	RB0/TB0
			R (Receive)/W (Transmit)							
			Undefined							
SC1CR	Serial channel 1 control	209H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to "0" when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Bit8 of a received character	Parity type 0: Odd 1: Even	1: Parity enable	Error has occurred			0: SCLK↑ 1: SCLK↓	1: SCLK1 input
						Overrun	Parity	Framing		
SC1MOD0	Serial channel 1 mode	20AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Bit8 of a transmitted character	Hand shake control 1: Enables CTS operation	Receive control 1: Enables receiver	Wakeup function 1: Enabled	Serial transfer mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial clock (for UART) 00: TA0TRG 01: Baud rate generator 10: Internal f_{SYS} clock 11: External clock (SCLK input)	
BR1CR	Baud rate control	20BH	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Must be written as "0".	N + (16 – K)/16 function 1: Enabled	00: φT0 01: φT2 10: φT8 11: φT32		Clock divisor value for baud rate generator "0" to F			
BR1ADD	Serial channel 1 K setting register	20CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Value of K in N + (16 – K)/16 1 to F			
SC1MOD1	Serial channel 1 mode1	20DH	I2S1	FDPX1						
			R/W	R/W						
			0	0						
			IDLE2 0: OFF 1: ON	Synchronous 0: Half duplex 1: Full duplex						

(9) I^{2}C bus serial bus interface (1/2)

(9−1) I^{2}C/SIO Channel 0

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SBI0CR1	Serial bus interface control register 1	240H (I^{2}C bus mode) (RMW prohibited)	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0 /SWRMON
			W			R/W		W	W	R/W
			0	0	0	0		0	0	0/1
			Number of bits per transfer 000: 8, 001: 1, 010: 2 011: 3, 100: 4, 101: 5 110: 6, 111: 7			ACK clock pulse 0: No ACK 1: ACK		Serial clock frequency (on writes) 000: 5, 001: 6, 010: 7 011: 8, 100: 9, 101: 10 110: 11, 111: Reserved		
		240H (SIO mode) (RMW prohibited)	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
			W	W	W	W		W	W	W
			0	0	0	0		0	0	0
			Start transfer 0: Stop 1: Start	Abort transfer 0: Continue 1: Abort	Transfer mode 00: 8-bit transmit mode 10: 8-bit transmit/receive mode 11: 8-bit receive mode			Serial clock frequency (on writes) 000: 4, 001: 5, 010: 6 011: 7, 100: 8, 101: 9 110: 10, 111: External clock (input from the SCK pin)		
SBI0DBR	SBI buffer register	241H (RMW prohibited)	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	PB0/TB0
			R (Receive)/W (Transmit)							
			Undefined							
I2C0AR	I^{2}C bus address register	242H (RMW prohibited)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			Slave address							Address Recognition 0 : Recognize 1 : Does not recognize
When read SBI0SR	Serial bus interface status register	243H (I^{2}C bus mode) (RMW prohibited)	MST	TRX	BB	PIN	AL/SBIM1	AAS/SBIM0	AD0/ SWRST	LRB/ SWRST0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	1	0	0	0	0
			0: Slave 1: Master	0: Receive 1: Transmit	Bus status monitor 0: Free 1: Busy	INTSBI0 interrupt status 0: Asserted 1: Not asserted	Arbitration lost detection monitor 1: Detect	Slave address match detection monitor 1: Detect	GENERAL CALL detection monitor 1: Detect	Last receive bit monitor 0: 0 1: 1
When write SBI0CR2	Serial bus interface control register 2				Start/stop condition generation 0: Start condition 1: Stop condition		Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		Software reset generate write “10” and “01”, then an internal software reset signal is generated.	
When read SBI0SR	Serial bus interface status register	243H (SIO mode) (RMW prohibited)					SIOF/ SBIM1	SEF/ SBIM2	–	–
							R/W	R/W	W	W
							0	0	0	0
							Transfer status monitor 0: Stopped 1: Terminated in process	Shift operation status monitor 0: Stopped 1: Terminated in process		
When write SBI0CR2	Serial bus interface control register 2						Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		Always write “0”.	Always write “0”.

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SBI0BR0	Serial bus interface baud rate register 0	244H (RMW prohibited)	–	I2SBI0						
			W	R/W						
			0	0						
			Must be written as “0”.	IDLE2 0: OFF 1: ON						
SBI0BR1	Serial bus interface baud rate register 1	245H (RMW prohibited)	P4EN	–						
			W	W						
			0	0						
			Internal clock 0: OFF 1: ON	Must be written as “0”.						

I^{2}C bus serial bus interface (2/2)

(9–2) I^{2}C/SIO Channel 1

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SBI1CR1	Serial bus interface control register 1	248H (I^{2}C bus mode) (RMW prohibited)	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
			W			R/W		W	W	R/W
			0	0	0	0		0	0	0/1
			Number of bits per transfer 000: 8, 001: 1, 010: 2 011: 3, 100: 4, 101: 5 110: 6, 111: 7			ACK clock pulse 0: No ACK 1: ACK		Serial clock frequency (on writes) 000: 5, 001: 6, 010: 7 011: 8, 100: 9, 101: 10 110: 11, 111: Reserved		
		248H (SIO mode) (RMW prohibited)	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
			W	W	W	W		W	W	W
			0	0	0	0		0	0	0
			Start transfer 0: Stop 1: Start	Abort transfer 0: Continue 1: Abort	Transfer mode 00: 8-bit transmit mode 10: 8-bit transmit/receive mode 11: 8-bit receive mode			Serial clock frequency (on writes) 000: 4, 001: 5, 010: 6 011: 7, 100: 8, 101: 9 110: 10, 111: External clock (input from the SCK pin)		
SBI1DBR	SBI buffer register	249H (RMW prohibited)	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	PB0/TB0
			R (Receive)/W (Transmit)							
			Undefined							
I2C1AR	I^{2}C bus address register	24AH (RMW prohibited)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			Slave address							Address Recognition 0: Recognize 1: Does not recognize
When read SBI1SR	Serial bus interface status register	24BH (I^{2}C bus mode) (RMW prohibited)	MST	TRX	BB	PIN	AL/SBIM1	AAS/SBIM0	AD0/ SWRST	LRB/ SWRST0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	1	0	0	0	0
			0: Slave 1: Master	0: Receive 1: Transmit	Bus status monitor 0: Free 1: Busy	INTSBI1 interrupt status 0: Asserted 1: Not asserted	Arbitration lost detection monitor 1: Detect	Slave address match detection monitor 1: Detect	GENERAL CALL detection monitor 1: Detect	Last receive bit monitor 0: 0 1: 1
When write SBI1CR2	Serial bus interface control register 2				Start/stop condition generation 0: Start condition 1: Stop condition		Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		Software reset generate write “10” and “01”, then an internal software reset signal is generated.	
When read SBI1SR	Serial bus interface status register	24BH (SIO mode) (RMW prohibited)					SIOF/ SBIM1	SEF/ SBIM2	–	–
							R/W	R/W	W	W
							0	0	0	0
							Transfer status monitor 0: Stopped 1: Terminated in process	Shift operation status monitor 0: Stopped 1: Terminated in process		
When write SBI1CR2	Serial bus interface control register 2						Serial bus interface operating mode selection 00: Port mode 01: SIO mode 10: I^{2}C bus mode 11: (Reserved)		Always write “0”.	Always write “0”.

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
SBI1BR0	Serial bus interface baud rate register 0	24CH (RMW prohibited)	–	I2SBI1						
			R/W	R/W						
			0	0						
			Must be written as "0".	IDLE2 0: OFF 1: ON						
SBI1BR1	Serial bus interface baud rate register 1	24DH (RMW prohibited)	P4EN	–						
			W	W						
			0	0						
			Internal clock 0: OFF 1: ON	Must be written as "0".						

(10) A/D converter control

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
ADMOD0	AD mode register 0	2B0H	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
			R		R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			End-of-conversion flag 1: Conversion completed	AD conversion busy flag 1: Conversion in progress	Must be written as "0".	Must be written as "0".	Interrupt timing in fixed-channel continuous conversion mode	1: Continuous conversion	Channel scan conversion	AD conversion start
ADMOD1	AD mode register 1	2B1H	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
			R/W	R/W			R/W	R/W		
			0	0			0	0	0	0
			1: VREF control ON	IDLE2 0: OFF 1: ON			AD conversion start	Analog input channel select 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		
ADREG04L	AD result register 0/4 low	2A0H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
ADREG04H	AD result register 0/4 high	2A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
ADREG15L	AD result register 1/5 low	2A2H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
ADREG15H	AD result register 1/5 high	2A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
ADREG26L	AD result register 2/6 low	2A4H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
ADREG26H	AD result register 2/6 high	2A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
ADREG37L	AD result register 3/7 low	2A6H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
ADREG37H	AD result register 3/7 high	2A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							

(11) Watchdog timer

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT mode register	300H	WDTE	WDTP1	WDTP0			I2WDT	RESCR	–
			R/W	R/W	R/W			R/W	R/W	R/W
			1	0	0			0	0	0
			WDT control 0: Disable 1: Enable	00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$				IDLE2 0: OFF 1: ON	System reset by WDT 1: Reset	Must be written as "0".
WDCR	WD control	301H (RMW prohibited)	–							
			W							
			–							
			B1H: WDT disable code, 4EH: WDT clear-count code							

(12) Key wakeup

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
KWIEN	KWI enable register	3A0H (RMW prohibited)	KWI7EN	KWI6EN	KWI5EN	KWI4EN	KWI3EN	KWI2EN	KWI1EN	KWI0EN
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			KWI7 interrupt input 0: Disable 1: Enable	KWI6 interrupt input 0: Disable 1: Enable	KWI5 interrupt input 0: Disable 1: Enable	KWI4 interrupt input 0: Disable 1: Enable	KWI3 interrupt input 0: Disable 1: Enable	KWI2 interrupt input 0: Disable 1: Enable	KWI1 interrupt input 0: Disable 1: Enable	KWI0 interrupt input 0: Disable 1: Enable
KWICR	KWI control register	3A1H (RMW prohibited)	KWI7EDGE	KWI6EDGE	KWI5EDGE	KWI4EDGE	KWI3EDGE	KWI2EDGE	KWI1EDGE	KWI0EDGE
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			KWI7 edge polarity 0: Rising 1: Falling	KWI6 edge polarity 0: Rising 1: Falling	KWI5 edge polarity 0: Rising 1: Falling	KWI4 edge polarity 0: Rising 1: Falling	KWI3 edge polarity 0: Rising 1: Falling	KWI2 edge polarity 0: Rising 1: Falling	KWI1 edge polarity 0: Rising 1: Falling	KWI0 edge polarity 0: Rising 1: Falling

(13) BCD adder/subtractor

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
BCDMINA	BCD minute operand register A	3B0H (RMW prohibited)	MINA7	MINA6	MINA5	MINA4	MINA3	MINA2	MINA1	MINA0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand A							
BCDSECA	BCD second operand register A	3B1H (RMW prohibited)	SECA7	SECA6	SECA5	SECA4	SECA3	SECA2	SECA1	SECA0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand A							
BCDFRAA	BCD frame operand register A	3B2H (RMW prohibited)	FRAA7	FRAA6	FRAA5	FRAA4	FRAA3	FRAA2	FRAA1	FRAA0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand A							
BCDMINB	BCD minute operand register B	3B4H (RMW prohibited)	MINB7	MINB6	MINB5	MINB4	MINB3	MINB2	MINB1	MINB0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand B							
BCDSECB	BCD second operand register B	3B5H (RMW prohibited)	SECB7	SECB6	SECB5	SECB4	SECB3	SECB2	SECB1	SECB0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand B							
BCDFRAB	BCD frame operand register B	3B6H (RMW prohibited)	FRAB7	FRAB6	FRAB5	FRAB4	FRAB3	FRAB2	FRAB1	FRAB0
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			BCD operand B							
BCDMINR	BCD minute result register	3B8H	MINR7	MINR6	MINR5	MINR4	MINR3	MINR2	MINR1	MINR0
			R	R	R	R	R	R	R	R
			0	0	0	0	0	0	0	0
			BCD operation result							
BCDSECR	BCD second result register	3B9H	SECR7	SECR6	SECR5	SECR4	SECR3	SECR2	SECR1	SECR0
			R	R	R	R	R	R	R	R
			0	0	0	0	0	0	0	0
			BCD operation result							
BCDFRAR	BCD frame result register	3BAH	FRAR7	FRAR6	FRAR5	FRAR4	FRAR3	FRAR2	FRAR1	FRAR0
			R	R	R	R	R	R	R	R
			0	0	0	0	0	0	0	0
			BCD operation result							
BCDCR	BCD control register	3BCH	ENDFLAG	CY	BR			–	CALSEL	START
			R	R	R			R/W	R/W	R/W
			0	0	0			0	0	0
			Operation completion flag 1: Completed	Carry	Borrow			Must be written as “0”.	Add/sub-tract select	Operation start

(14) Program patch logic (1/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
ROMCMP00	Address compare register 00	400H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP01	Address compare register 01	401H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP02	Address compare register 02	402H (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB0L	Address substitution register 0 low	404H (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB0H	Address substitution register 0 high	405H (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP10	Address compare register 10	408H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP11	Address compare register 11	409H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP12	Address compare register 12	40AH (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB1L	Address substitution register 1 low	40CH (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB1H	Address substitution register 1 high	40DH (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

Program patch logic (2/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
ROMCMP20	Address compare register 20	410H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP21	Address compare register 21	411H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP22	Address compare register 22	412H (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB2L	Address substitution register 2 low	414H (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB2H	Address substitution register 2 high	415H (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP30	Address compare register 30	418H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP31	Address compare register 31	419H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP32	Address compare register 32	41AH (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB3L	Address substitution register 3 low	41CH (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB3H	Address substitution register 3 high	41DH (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

Program patch logic (3/3)

Mnemonic	Name	Address	7	6	5	4	3	2	1	0
ROMCMP40	Address compare register 40	420H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP41	Address compare register 41	421H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP42	Address compare register 42	422H (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB4L	Address substitution register 4 low	424H (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB4H	Address substitution register 4 high	425H (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP50	Address compare register 50	428H (RMW prohibited)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	
			W							
			0	0	0	0	0	0	0	
			Target ROM address (Lower 7 bits)							
ROMCMP51	Address compare register 51	429H (RMW prohibited)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP52	Address compare register 52	42AH (RMW prohibited)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB5L	Address substitution register 5 low	42CH (RMW prohibited)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB5H	Address substitution register 5 high	42DH (RMW prohibited)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

6. I/O Port Equivalent-circuit Diagrams

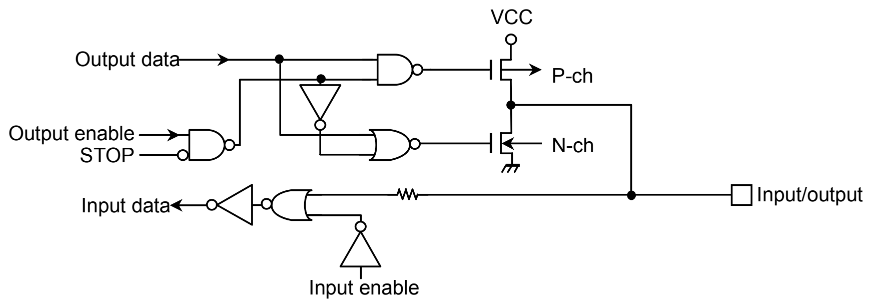
- How to read circuit diagrams

  The circuit diagrams in this chapter are drawn using the same gate symbols as for the 74HCxx series standard CMOS logic ICs.

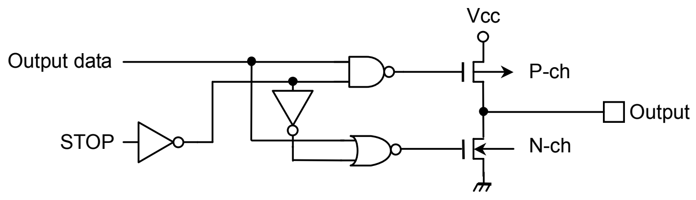
  The signal named STOP has a unique function. This signal goes active-high if the CPU sets the HALT bit when the HALTM[1:0] field in the SYSCR2 register is programmed to 01 (e.g., STOP mode) and the drive enable (DRVE) bit in the same register is cleared. If the DRVE bit is set, the STOP signal remains inactive (at logic 0).

- The input protection circuit has a resistor in the range of several tens to several hundreds of ohms.

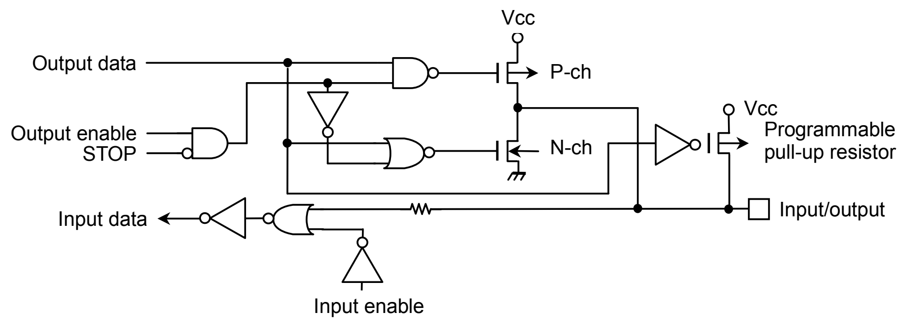
■ Port 0 (AD0 to AD7), Port 1 (AD8 to AD15, A8 to A15), Port 2 (A16 to A23, A0 to A7)



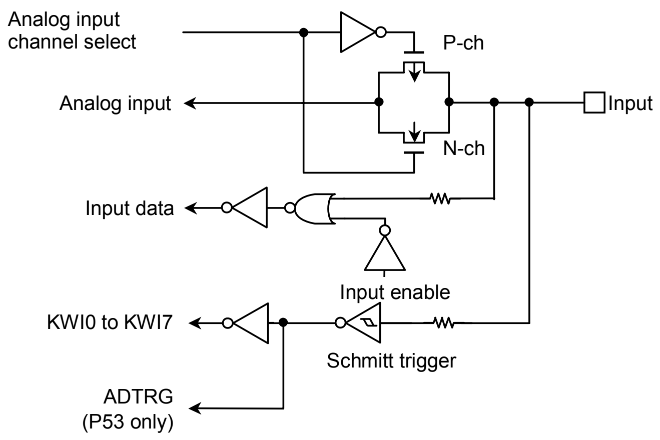
■ P30 ($\overline{\text{RD}}$), P31 ($\overline{\text{WR}}$)



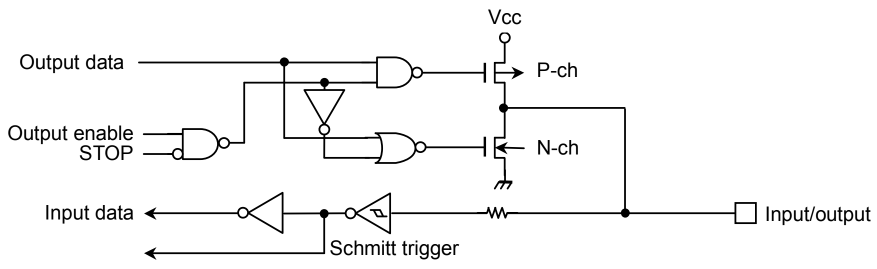
■ P32 to P37



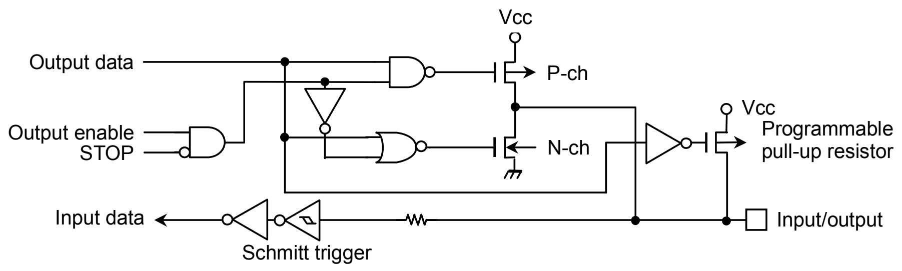
■ Port 5 (AN0 to AN7)



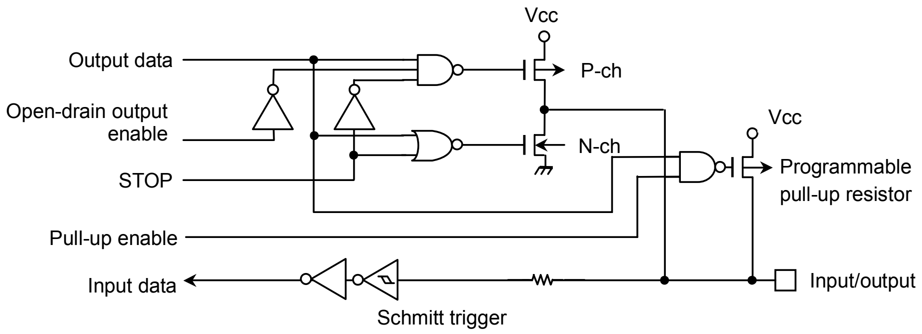
■ P63 (INT0)



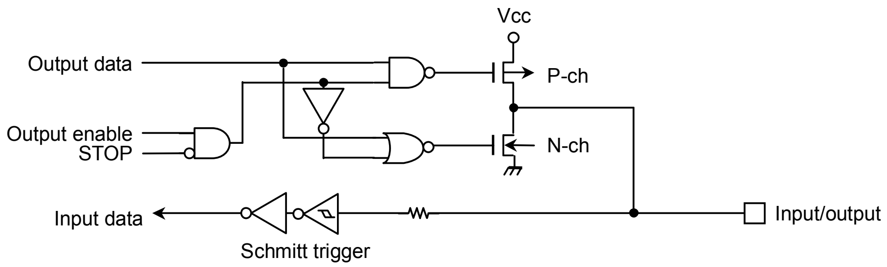
■ P40 to P43, P70 to P75, P80 to P87, PA0 to PA7



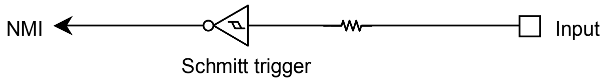
■ P61 (SO0/SDA0), P62 (SI0/SCL0), P91 (SO1/SDA1), P92 (SI1/SCL1), P93 (TXD1)



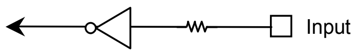
■ P60, P64 to P66, P90, P93 to P96



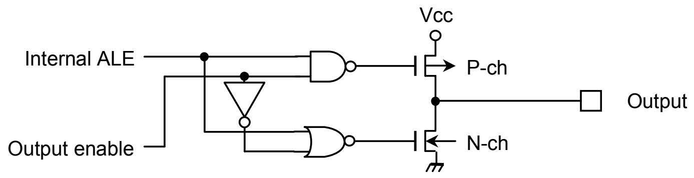
■ $\overline{\text{NMI}}$



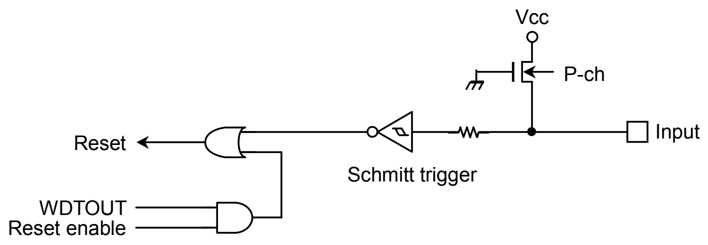
■ AM0, AM1



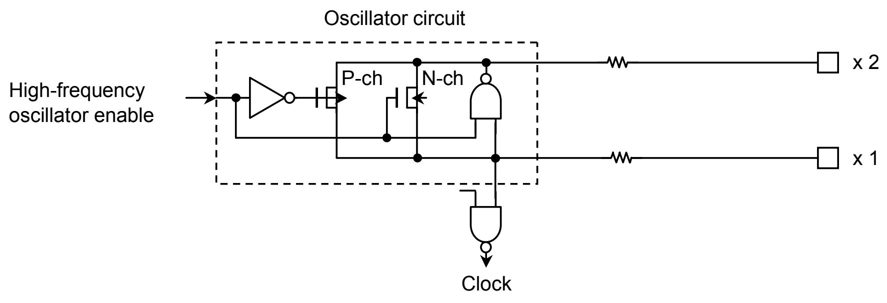
■ ALE



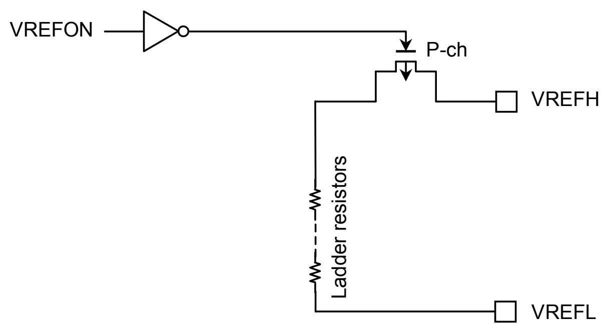
■ $\overline{\text{RESET}}$



■ X1, X2



■ VREFH, VREFL



7. Points of Note and Restrictions

(1) Notations and terms

a. I/O register fields are often referred to as <register_mnemonic>.<field_name> for the interest of brevity. For example, TRUN.T0RUN means the T0RUN bit in the TRUN register.

b. Read-modify-write instructions

Read-modify-write instructions allow the CPU to read data from memory, manipulate the data and then write the result to the same memory address, through the execution of a single instruction.

Example 1: SET 3, (TRUN)	Sets bit3 of the TRUN register.
Example 2: INC 1, (100H)	Increments the data at address 100H by one.

- Read-modify-write instructions supported by the TLCS-900.

Exchange

```
EX     (mem), R
```

Arithmetical operation

```
ADD  (mem), R/#       ADC  (mem), R/#
SUB  (mem), R/#       SBC  (mem), R/#
INC  #3, (mem)        DEC  #3, (mem)
```

Logical 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)
```

Rotation and shift

```
RLC  (mem)            RRC  (mem)
RL   (mem)            RR   (mem)
SLA  (mem)            SRA  (mem)
SLL  (mem)            SRL  (mem)
RLD  (mem)            RRD  (mem)
```

c. fc, f_{FPH}, f_{SYS}, state

f_{OSCH}, fc: Clock frequency supplied via the X1 and X2 pins

f_{FPH}: Clock frequency selected by the GEAR[2:0] bit in the SYSCR1

f_{SYS}: System clock frequency, created by dividing f_{FPH} by two

1 state: One period of f_{SYS}

(2) Precautions and restrictions

a. AM0 and AM1 pins

The AM0 and AM1 pins must be connected to the DV_{CC} pin to ensure that their signal levels do not fluctuate during chip operation.

b. EMU0 and EMU1 pins

The EMU0 and EMU1 pins must be left open.

c. Reserved address space

The TMP91CW28 does not have any address space reserved.

d. Oscillator warm-up counter

If an external crystal is utilized, an interrupt signal programmed to bring the TMP91CW28 out of STOP mode triggers the on-chip warm-up counter. The system clock is not supplied to the on-chip logic until the warm-up counter expires.

e. Programmable pull-up resistors

When port pins are configured as input ports, the integrated pull-up resistors can be enabled and disabled under software control. The pull-up resistors are not programmable when port pins are configured as output ports, except for P61, P62, P91 and P92.

The relevant port registers (e.g., the P6 register) must be programmed by using store instructions; read-modify-write instructions cannot be used.

f. External bus mastership

The pin states while the bus is granted to an external device are described in section 3.5, I/O ports.

g. Watchdog timer

Upon reset, the watchdog timer is enabled. If the watchdog timer function is not required, it must be disabled after reset.

h. Watchdog timer

When relevant pins are configured as bus arbitration signals, the I/O peripherals including the watchdog timer can operate during external bus mastership.

i. AD converter

The ladder resistor network between the VREFH and VREFL pins can be disconnected under software control. If it is necessary to reduce power dissipation in STOP mode, the ladder resistor network should be disconnected before executing the HALT instruction.

j. CPU (Micro DMA)

Only the “LDC cr, r” and “LDC r, cr” instructions can write or read control registers within the CPU, such as the transfer source register (DMASn).

k. Undefined bits in I/O registers

Undefined I/O register bits are read as undefined states. Therefore, software must be coded without relying on the states of any undefined bits.

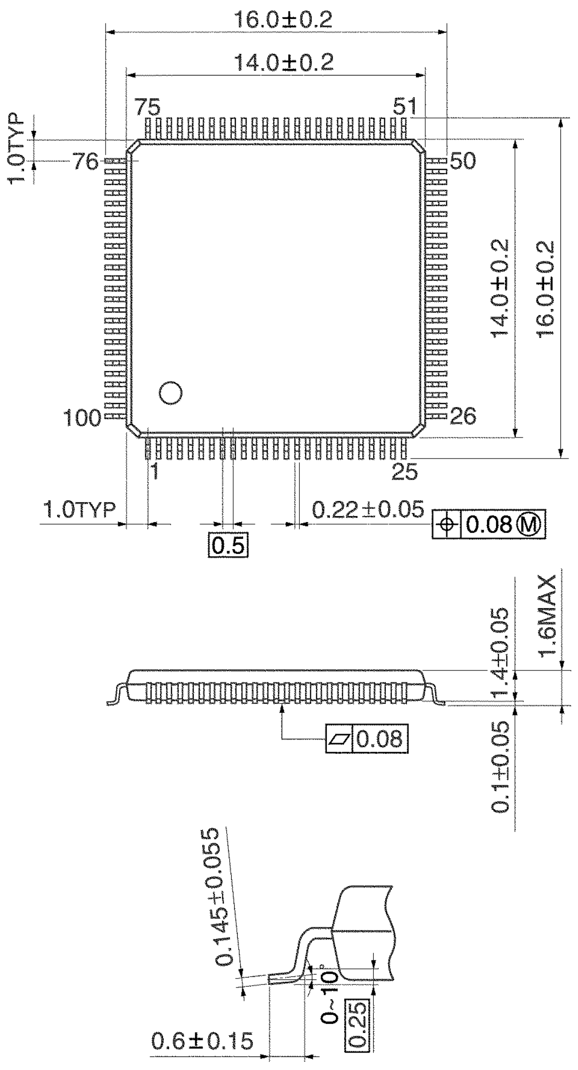
l. POP SR instruction

The POP SR instruction must be executed in DI mode.

8. Package Dimensions

P-LQFP100-1414-0.50F

Unit: mm


16.0±0.2
14.0±0.2
75
51
1.0TYP
76
50
14.0±0.2
16.0±0.2
100
26
1
25
1.0TYP
0.22±0.05
0.08 Ⓜ
0.5
1.6MAX
1.4±0.05
0.08
0.1±0.05
0.145±0.055
0~10°
0.25
0.6±0.15