TOSHIBA

TOSHIBA Original CMOS 16-Bit Microcontroller

TLCS-900/L1 Series

TMP91CW40FG

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 Consumption

CMOS 16-Bit Microcontroller

TMP91CW40FG

1. Outline and Features

The TMP91CW40 is a high-speed, high-performance 16-bit microcontroller capable of low-voltage, low-power-consumption operation.

This microcontroller comes in a 100-pin flat package and has the following features:

(1) Toshiba proprietary 16-bit CPU (900/L1 CPU)
- Instruction mnemonics are upwardly compatible with the TLCS-90 and TLCS-900.
- 16-Mbyte linear address space
- Architecture based on general-purpose registers and register banks
- 16-bit multiply/divide instructions and bit transfer/arithmetic instructions
- Micro DMA: 4 channels (593 ns/2 bytes at 27 MHz)

(2) Minimum instruction execution time: 148 ns (at 27 MHz)

(3) Internal RAM: 4 Kbytes

(4) Internal ROM: 128 Kbytes

(5) 8-bit timer: 4 channels

(6) 16-bit timer: 3 channels

(7) Divider output

(8) General-purpose serial interface: 4 channels
- Both UART and synchronous transfer modes are supported.

(9) 10-bit AD converter (with sample-and-hold): 4 channels

(10) Watchdog timer

(11) Key-on wakeup: 4 channels

(12) Real-time clock (RTC)
- Based on the TC8521A specifications

(13) Melody/Alarm generator (MLD)

(14) Program patch logic: 6 banks

(15) LCD driver/controller (voltage reducer type, reference voltage = VCC)
- LCD direct drive possible (8 to 40 segments x 4 commons)
- 1/4 duty, 1/3 duty, 1/2 duty or static drive selectable

(16) Interrupts: 43 sources
- 9 CPU interrupts: Triggered by a software interrupt instruction or undefined instruction
- 27 internal interrupts: 7 priority levels
- 7 external interrupts: 7 priority levels
  (Two interrupts support selection of triggering edge.)

(17) Input/output ports: 61 pins

(18) Standby function
Three HALT modes (programmable IDLE2, IDLE1, STOP)

(19) Clock control function
- Low-frequency clock (fs = 32.768 kHz)

(20) Operating voltage range
- Vcc = 2.7 to 3.6 V (fc max = 27 MHz)
- Vcc = 2.2 to 3.6 V (fc max = 16 MHz)
  (VCC < 2.7V: LCDD disabled.)

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

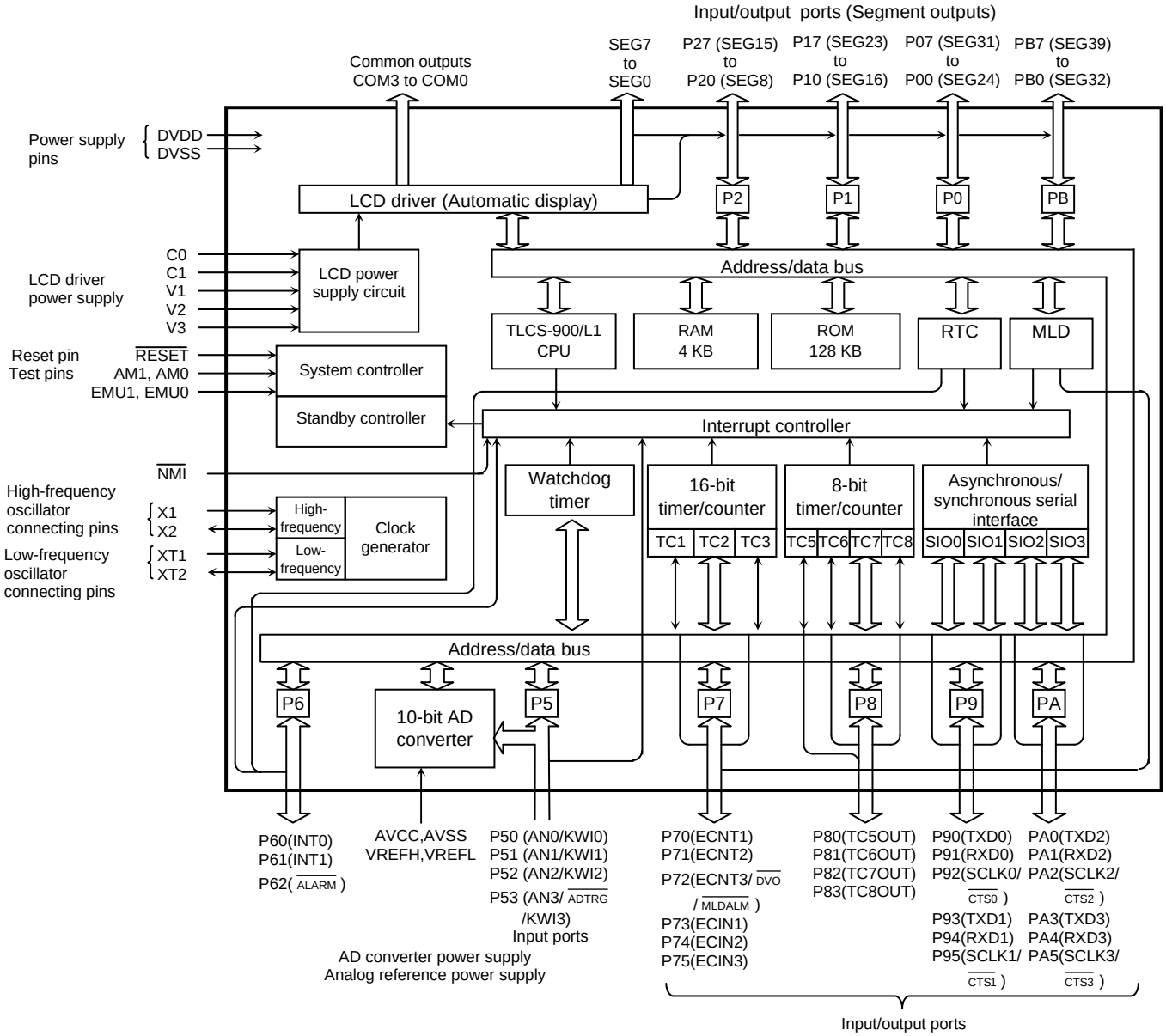


Figure 1.1 TMP91CW40 Block Diagram

2. Pin Assignments and Pin Functions

The assignment of input/output pins for the TMP91CW40, their names and functions are follows:

2.1 Pin Assignments

Figure 2.1.1 shows the pin assignments of the TMP91CW40FG.

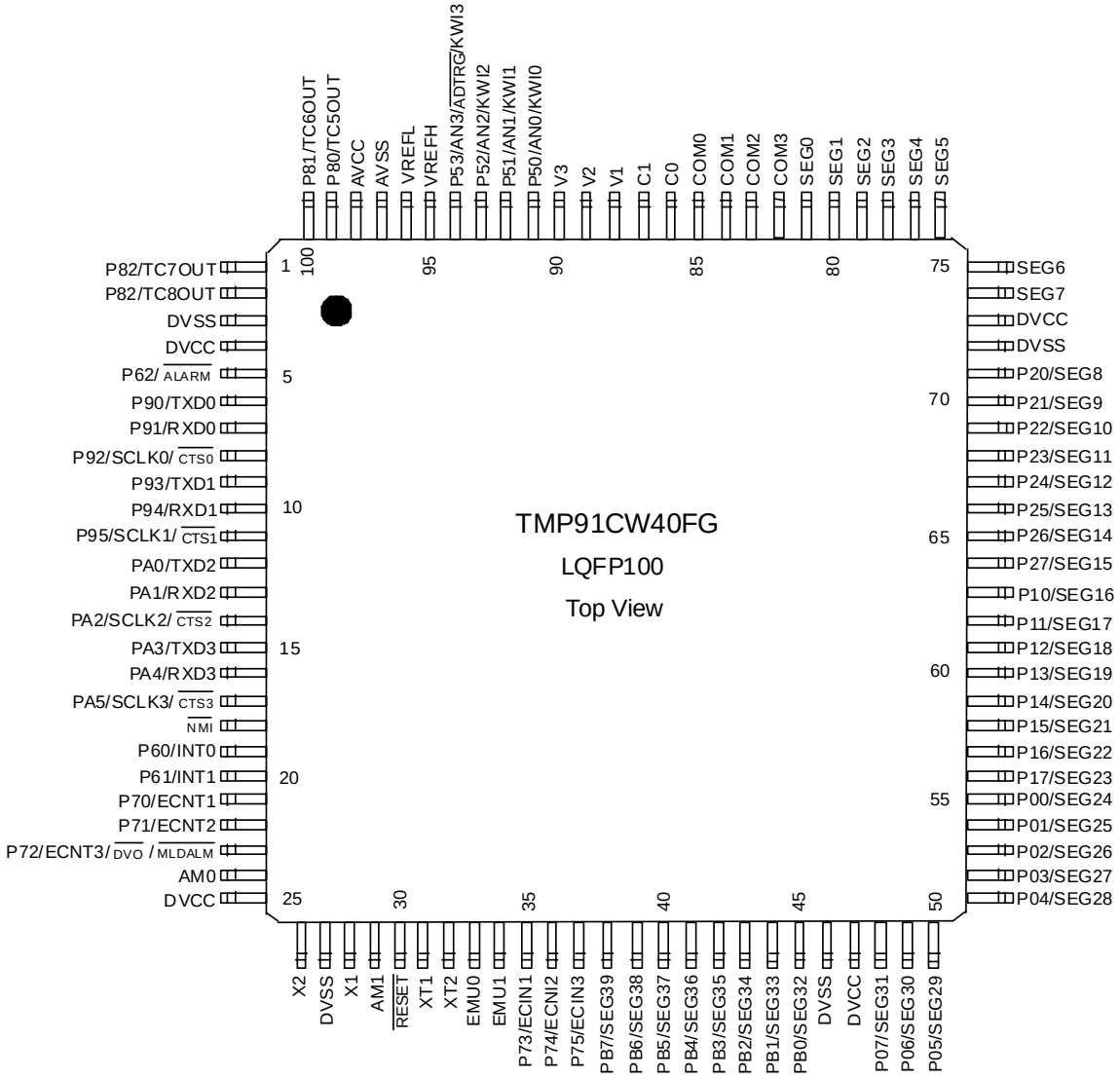


Figure 2.1.1 TMP91CW40FG Pin Assignments (100-pin LQFP, top view)

2.2 Pin Names and Functions

Table 2.2.1 to Table 2.2.2 list the names and functions of the input and output pins of the TMP91CW40.

Table 2.2.1 Pin Names and Functions (1/2)

Pin Name	Number of Pins	I/O	Function
P50 to P53	4	Input	Port 5: Input port
AN0 to AN3		Input	Analog input: Input to the AD converter
$\overline{\text{ADTRG}}$		Input	AD trigger: External start request pin for the AD converter (multiplexed with P53)
KWI0 to KWI3		Input	Key-on wakeup input (multiplexed with P50 to P53)
P60	1	Input	Port 60: Input port
INT0		Input	Interrupt request pin 0: Programmable as high-level, low-level, rising-edge or falling-edge sensitive
P61	1	I/O	Port 61: Input/output port
INT1		Input	Interrupt request pin 1: Programmable as high-level, low-level, rising-edge or falling-edge sensitive
P62	1	Output	Port 62: Input/output port
$\overline{\text{ALARM}}$		Output	RTC alarm output pin
$\overline{\text{BOOT}}$		Input	Boot mode control pin for flash memory (specifically designed for 91FW40; to be pulled up during the reset period) Note: In NORMAL mode, do not input Low level on this pin during the reset period. If Low level is input, boot mode will be entered.
P70	1	I/O	Port 70: Input/output port
ECNT1		Input	16-bit timer 1 input: Count control input for 16-bit timer TC1
P71	1	I/O	Port 71: Input/output port
ECNT2		Input	16-bit timer 2 input: Count control input for 16-bit timer TC2
P72	1	I/O	Port 72: Input/output port
ECNT3		Input	16-bit timer 3 input: Count control input for 16-bit timer TC3
$\overline{\text{DVO}}$		Output	Divider output pin
$\overline{\text{MLDALM}}$		Output	Melody/Alarm output pin
P73	1	I/O	Port 73: Input/output port
ECIN1		Input	16-bit timer 1 input: Count input for 16-bit timer TC1
P74	1	I/O	Port 74: Input/output port
ECIN2		Input	16-bit timer 2 input: Count input for 16-bit timer TC2
P75	1	I/O	Port 75: Input/output port
ECIN3		Input	16-bit timer 3 input: Count input for 16-bit timer TC3
P80	1	I/O	Port 80: Input/output port (large-current port)
TC5OUT		Output	8-bit timer 5 output: Output pin for 8-bit timer TC5 Open-drain output mode by programmable
P81	1	I/O	Port 81: Input/output port (large-current port)
TC6OUT		Output	8-bit timer 6 output: Output pin for 8-bit timer TC6 Open-drain output mode by programmable
P82	1	I/O	Port 82: Input/output port (large-current port)
TC7OUT		Output	8-bit timer 7 output: Output pin for 8-bit timer TC7 Open-drain output mode by programmable
P83	1	I/O	Port 83: Input/output port (large-current port)
TC8OUT		Output	8-bit timer 8 output: Output pin for 8-bit timer TC8 Open-drain output mode by programmable
P90	1	I/O	Port 90: Input/output port
TXD0		Output	Serial 0 transmit data Open-drain output mode by programmable
P91	1	I/O	Port 91: Input/output port
RXD0		Input	Serial 0 receive data
P92	1	I/O	Port 92: Input/output port
SCLK0		I/O	Serial 0 clock input/output
$\overline{\text{CTS0}}$		Input	Serial 0 data transmit enable (Clear to send)

Table 2.2.2 Pin Names and Functions (2/2)

Pin Name	Number of Pins	I/O	Function
P93 TXD1	1	I/O Output	Port 93: Input/output port Serial 1 transmit data Open-drain output mode by programmable
P94 RXD1	1	I/O Input	Port 94: Input/output port Serial 1 receive data
P95 SCLK1 $\overline{\text{CTS1}}$	1	I/O I/O Input	Port 95: Input/output port Serial 1 clock input/output Serial 1 data transmit enable (Clear to send)
PA0 TXD2	1	I/O Output	Port A0: Input/output port Serial 2 transmit data Open-drain output mode by programmable
PA1 RXD2	1	I/O Input	Port A1: Input/output port Serial 2 receive data
PA2 SCLK2 $\overline{\text{CTS2}}$	1	I/O I/O Input	Port A2: Input/output port Serial 2 clock input/output Serial 2 data transmit enable (Clear to send)
PA3 TXD3	1	I/O Output	Port 3: Input/output port Serial 3 transmit data Open-drain output mode by programmable
PA4 RXD3	1	I/O Input	Port A4: Input/output port Serial 3 receive data
PA5 SCLK3 $\overline{\text{CTS3}}$	1	I/O I/O Input	Port A5: Input/output port Serial 3 clock input/output Serial 3 data transmit enable (Clear to send)
SEG0 to SEG7	8	Output	Segment output
P20 to P27 SEG8 to SEG15	8	I/O Output	Port 2: Input/output port Segment output
P10 to P17 SEG16 to SEG23	8	I/O Output	Port 1: Input/output port Segment output
P00 to P07 SEG24 to SEG31	8	I/O Output	Port 0: Input/output port Segment output
PB0 to PB7 SEG32 to SEG39	8	I/O Output	Port B: Input/output port Segment output
C0,C1	2		LCD drive power supply
V1 to V3	3		LCD drive power supply
COM0 to COM3	4		Common output
$\overline{\text{NMI}}$	1	Input	Nonmaskable interrupt request pin: Causes an NMI interrupt on the falling edge; programmable to be rising-edge sensitive (Schmitt input).
AM0, AM1	2	Input	Operation mode Both AM0 and AM1 should be held at logic 1.
EMU0	1	Output	This pin should be left open.
EMU1	1	Output	This pin should be left open.
$\overline{\text{RESET}}$	1	Input	Reset: Initializes the TMP91CW40. (Schmitt input, with pull-up resistor)
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 (0 V)
X1/X2	2	I/O	Connection pins for a high-frequency oscillator
XT1/XT2	2	I/O	Connection pins for a low-frequency oscillator
DVCC DVSS	4 4		Power supply pins (The DVCC pins should be connected to power supply.) Ground pins (The DVSS pins should be connected to ground (0 V).)

3. Operation

This section describes the functions and basic operation of the TMP91CW40.

3.1 CPU

The TMP91CW40 contains a high-performance 16-bit CPU (900/L1 CPU). For a detailed description of the CPU, refer to "TLCS-900/L1 CPU" in the preceding chapter.

Functions unique to the TMP91CW40 not covered in "TLCS-900/L1 CPU" are described below.

3.1.1 Reset Operation

To reset the TMP91CW40, 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 for at least 10 system clocks (1μs at 27 MHz). After turning on the power to the TMP91CW40, hold the $\overline{\text{RESET}}$ input at low level for at least 10 system clocks with the power supply voltage within the operating voltage range and the internal high-frequency oscillator oscillating stably.

Reset operation initializes the system clock f_{SYS} to fc/2. The CPU performs the following operations as a result of a reset:

- Sets the program counter (PC) according to the reset vector stored at addresses FFFF00H to FFFF02H.
  - PC<7:0> ← Value at address FFFF00H
  - PC<15:8> ← Value at address FFFF01H
  - PC<23:16> ← Value at address FFFF02H
- Sets the stack pointer (XSP) to 100H.
- Sets the <IFF2:0> bits of the status register (SR) to 111 (setting the interrupt level mask register to level 7).
- Sets the <MAX> bit of the status register (SR) to 1 (selecting maximum mode).
- Clears the <RFP2:0> bits of the status register (SR) to 000 (selecting register bank 0).

After the reset state is released, the CPU starts executing instructions according to the PC. CPU internal registers other than the above are not changed.

The internal I/O peripherals, ports and other pins are initialized as follows upon a reset:

- All internal I/O registers are initialized.
- All port pins, including those multiplexed with internal I/O functions, are configured either as general-purpose inputs or general-purpose outputs.

Note: Reset operation does not affect the contents of the internal RAM or the CPU registers other than PC, SR and XSP.

Figure 3.1.1 shows reset timings of the TMP91CW40.

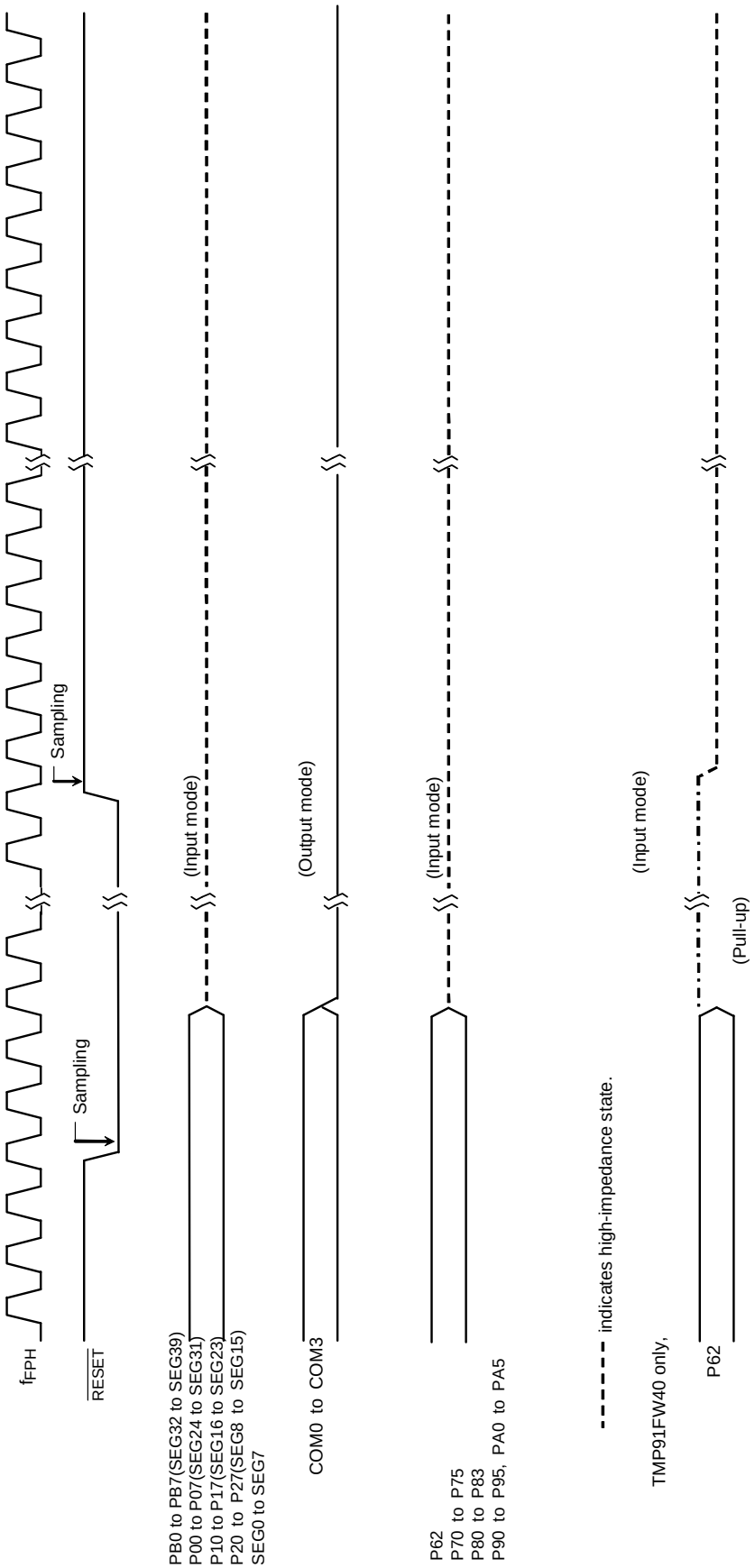


Figure 3.1.1 TMP91CW40 Reset Timings

3.2 Memory Map

Figure 3.2.1 shows a memory map of the TMP91CW40.

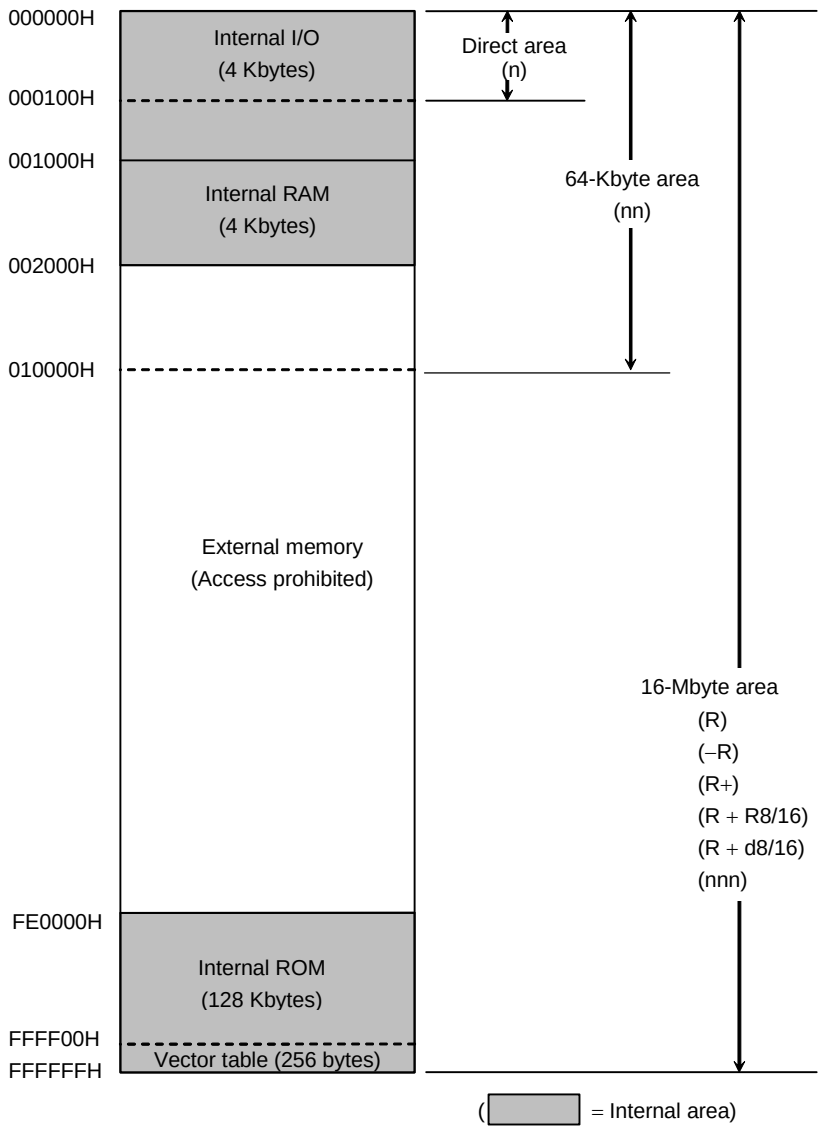


Figure 3.2.1 Memory Map

3.3 System Clock/Standby Control and Noise Reduction

The TMP91CW40 incorporates clock gear, standby control and noise reduction circuits to minimize power consumption and noise. Single-clock mode (X1 and X2 pins only) and dual-clock mode (X1, X2, XT1, and XT2 pins) are supported.

Figure 3.3.1 shows state transitions in each clock mode.

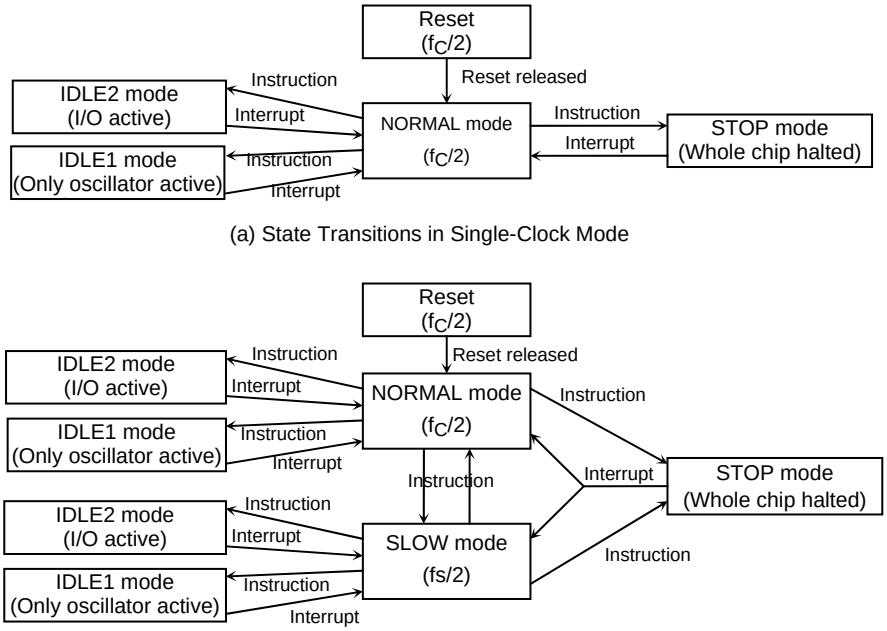


Figure 3.3.1 State Transitions in Each Operation Mode

The clock frequency terms used in this document are defined as follows:

fc: Clock frequency supplied via the X1 and X2 pins
fs: Clock frequency supplied via the XT1 and XT2 pins
f_{FPH}: Clock frequency selected by SYSCR1<SYSCK>
f_{SYS}: Clock frequency obtained by dividing f_{FPH} by two
1 state: One period of f_{SYS}

3.3.1 System Clock Block Diagram

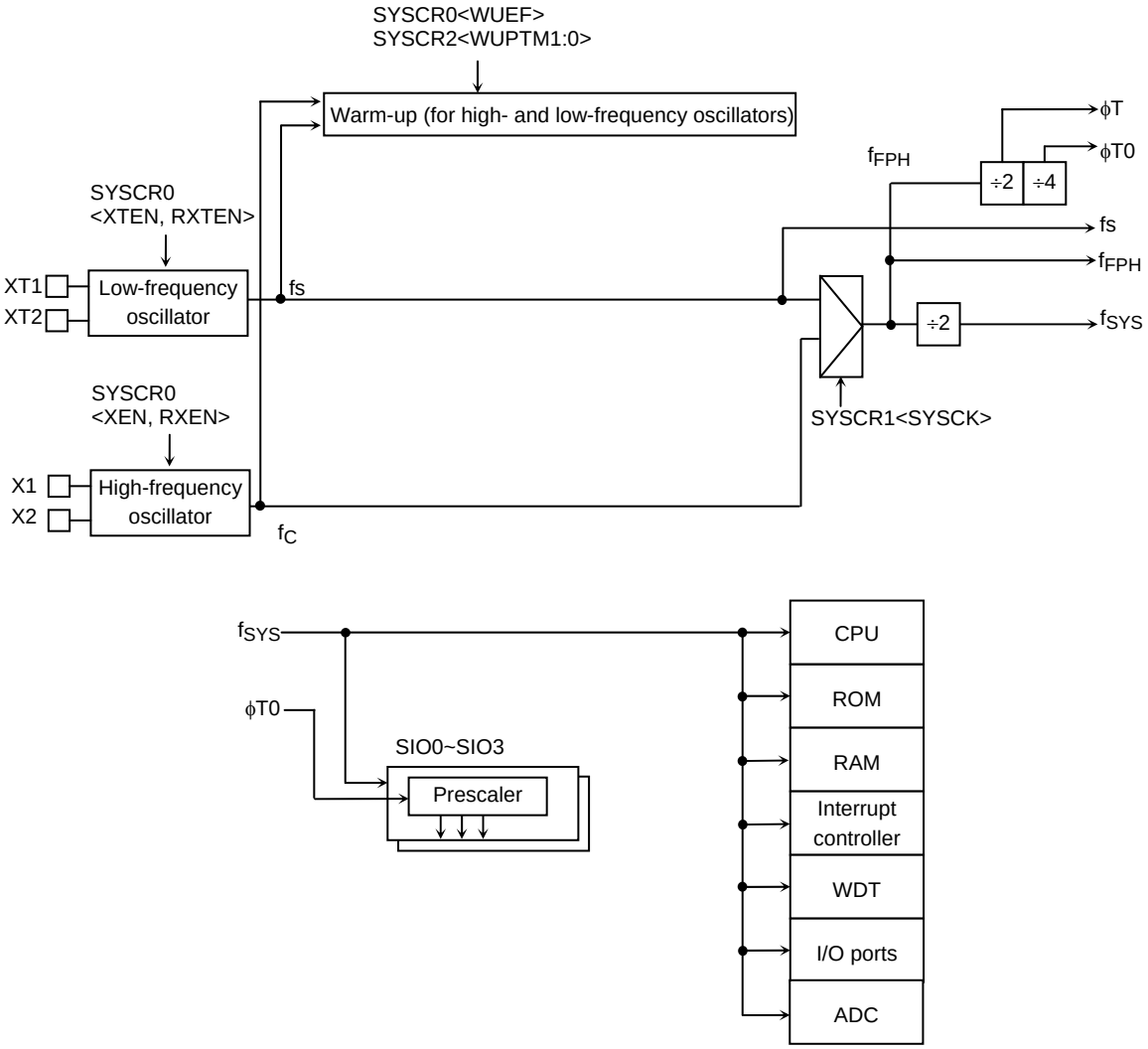


Figure 3.3.2 System Clock Block Diagram

3.3.2 SFRs

		7	6	5	4	3	2	1	0
SYSCR0 (00E0H)	Bit symbol	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	–	–
	Read/Write	R/W							
	After reset	1	0	1	0	0	0	0	0
	Function	High-frequency oscillator 0: Stop 1: Active	Low-frequency oscillator 0: Stop 1: Active	High-frequency oscillator after release of STOP mode 0: Stop 1: Active	Low-frequency oscillator after release of STOP mode 0: Stop 1: Active	Clock selection after release of STOP mode 0: High-frequency 1: Low-frequency	Warm-up timer (WUP) control 0 write: Don't care 1 write: Start WUP 0 read: WUP finished 1 read: WUP counting	Always write 00.	
SYSCR1 (00E1H)	Bit symbol					SYSCK	–	–	–
	Read/Write					R/W			
	After reset					0	0	0	0
	Function					System clock selection 0: High-frequency (fc) 1: Low-frequency (fs)	Always write 000.		
SYSCR2 (00E2H)	Bit symbol		–	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
	Read/Write		R/W	R/W	R/W	R/W	R/W		R/W
	After reset		0	1	0	1	1		0
	Function		Always write 0.	Oscillator warm-up time 00: Reserved 01: 2^8/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		HALT mode selection 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			1: Pins are driven in STOP mode.
SYSCR3 (00E5H)	Bit symbol								LCDCKMOD
	Read/Write								R/W
	After reset								0
	Function								LCD clock 0: fc 1: fs

Note: Bits 7 to 4 of the SYSCR1 and bits 7 and 1 of the SYSCR2 are read as undefined.

Figure 3.3.3 SFRs for the System Clock

		7	6	5	4	3	2	1	0
EMCCR0 (00E3H)	Bit symbol	PROTECT	–	–	–	–	EXTIN	DRVOSCH	DRVOSCL
	Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	1	0	0	0	1	1
	Function	Protection flag 0: OFF 1: ON	Always write 0.	Always write 1.	Always write 0.	Always write 0.	1: External clock used as fc	fc oscillator drive capability 1: Normal 0: Weak	fs oscillator drive capability 1: Normal 0: Weak
EMCCR1 (00E4H)	Bit symbol	Writing 1FH disables protection. Writing a value other than 1FH enables protection.							
	Read/Write								
	After reset								
	Function								

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

Figure 3.3.4 Noise-Related SFRs

3.3.3 System Clock Control Unit

The system clock control unit generates system clock pulses (f_{SYS}) that are supplied to the CPU core and internal I/O. It accepts either fc or fs clock pulses generated by the high-frequency or low-frequency oscillator, respectively. SYSCR1<SYSCK> is used to select the high-frequency or low-frequency oscillator. SYSCR0<XEN> and <XTEN> are used to enable and disable the high-frequency and low-frequency oscillators, respectively, so that power consumption can be reduced.

A system reset initializes <XEN> to 1, <XTEN> to 0 and <SYSCK> to 0, setting the system clock f_{SYS} to fc/2. For example, if a 27 MHz resonator is connected between the X1 and X2 pins, the f_{SYS} clock operates at 13.5 MHz.

(1) Switching between NORMAL mode and SLOW mode

A warm-up timer is provided to ensure stable oscillation of the resonator connected between the X1 and X2 pins or between the XT1 and XT2 pins before switching the system clock frequency. This warm-up time can be selected by SYSCR2<WUPTM1:0> according to the properties of the resonator to be used. SYSCR0<WUEF> is used to start the warm-up timer and to check whether or not the warm-up time has elapsed. For how to program the warm-up timer, refer to examples 1 and 2 on the pages that follow.

Table 3.3.1 shows the warm-up times for changing the system clock frequency.

Note 1: If the oscillator to be used has stable oscillation, no warm-up time is needed.

Note 2: Since the warm-up timer is operated by an oscillation clock, warm-up times may include some errors if there are fluctuations in oscillation frequency.

Table 3.3.1 Warm-Up Times (for changing the system clock frequency)

Warm-Up Time Setting SYSCR2<WUPTM1:0>	Changing to NORMAL Mode (fc)	Changing to SLOW Mode (fs)
01 (2^8/ oscillation frequency)	9.5 [μs]	7.8 [ms]
10 (2^{14}/oscillation frequency)	0.607 [ms]	500 [ms]
11 (2^{16}/oscillation frequency)	2.427 [ms]	2000 [ms]

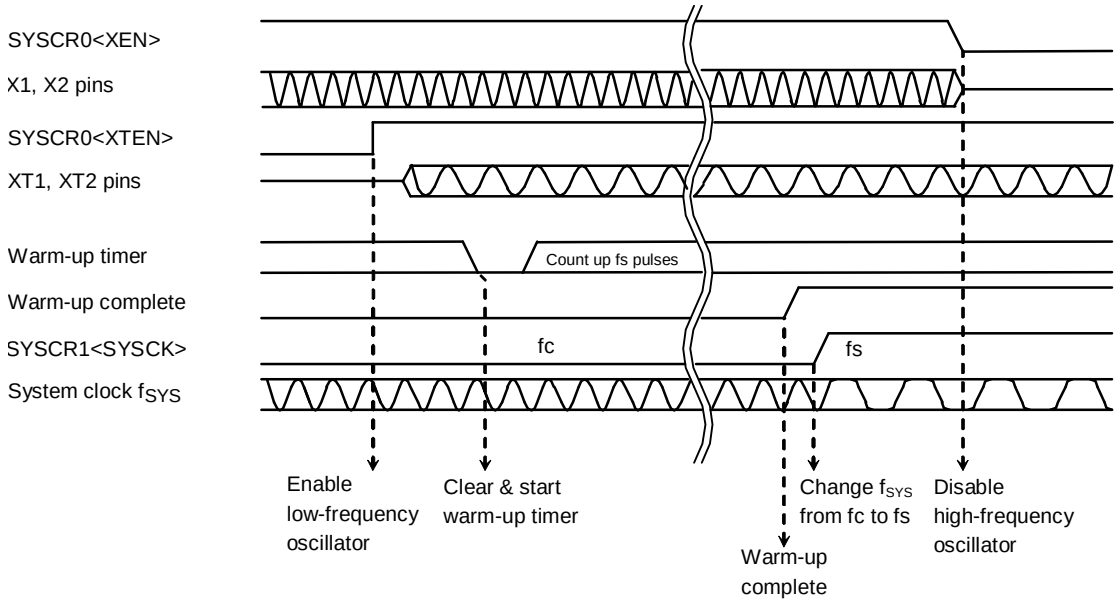
at fc = 27MHz
fs = 32.768kHz

Example 1

Changing the system clock from high-frequency (fc) to low-frequency (fs)

```
SYSCR0    EQU   00E0H
SYSCR1    EQU   00E1H
SYSCR2    EQU   00E2H
          LD    (SYSCR2), X–11– –X–B  ;  Set warm-up time to 2^16/fs.
          SET   6, (SYSCR0)           ;  Enable low-frequency oscillator.
          SET   2, (SYSCR0)           ;  Clear and start warm-up timer.
WUP:      BIT   2, (SYSCR0)           ;  } Detect completion of warming up.
          JR    NZ, WUP               ;  }
          SET   3, (SYSCR1)           ;  Change f_SYS from fc to fs.
          RES   7, (SYSCR0)           ;  Disable high-frequency oscillator.
```

X: Don't care, –: No change

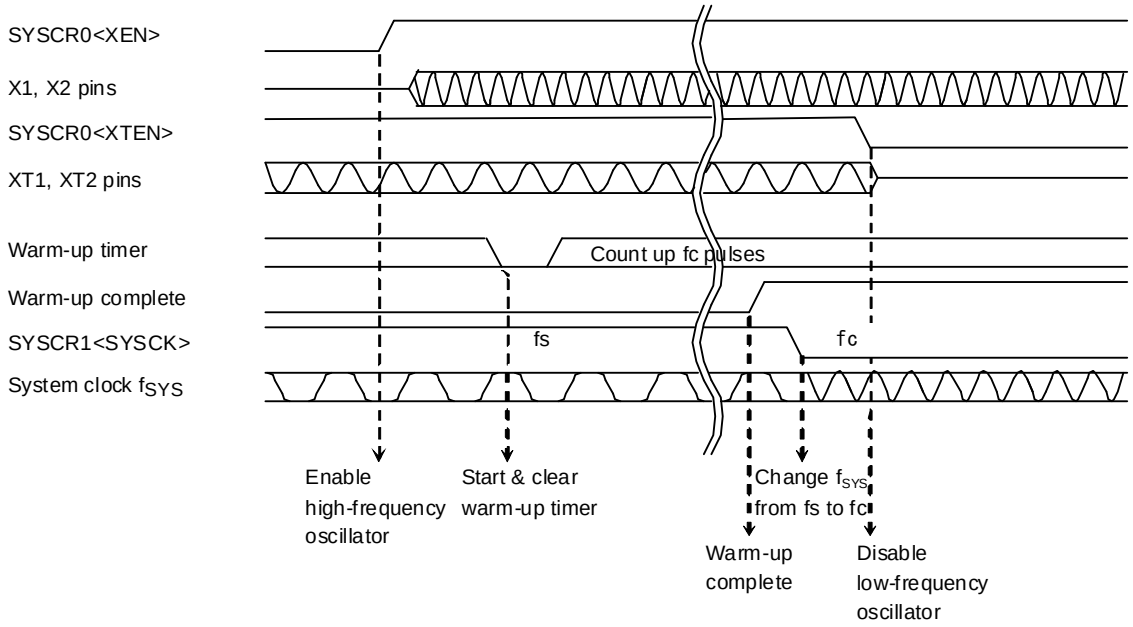


Example 2

Changing the system clock from low-frequency (fs) to high-frequency (fc)

```
SYSCR0    EQU   00E0H
SYSCR1    EQU   00E1H
SYSCR2    EQU   00E2H
          LD    (SYSCR2), X–10– –X–B  ;  Set warm-up time to 2^14/fc.
          SET   7, (SYSCR0)           ;  Enable high-frequency oscillator.
          SET   2, (SYSCR0)           ;  Clear and start warm-up timer.
WUP:      BIT   2, (SYSCR0)           ;  } Detect completion of warming up.
          JR    NZ, WUP               ;  }
          RES   3, (SYSCR1)           ;  Change f_SYS from fs to fc.
          RES   6, (SYSCR0)           ;  Disable low-frequency oscillator.
```

X: Don't care, –: No change



3.3.4 Prescaler Clock Control Unit

The internal I/O functions (SIO0 to SIO3) are provided with a clock prescaler. The prescaler clock sources φT and φT0 are $f_{FPH}/2$ and $f_{FPH}/4$, respectively.

3.3.5 Noise Reduction Circuits

The TMP91CW40 incorporates circuits providing the following features to reduce electromagnetic interference (EMI) and electromagnetic susceptibility (EMS):

(1) Reducing drive capability of the high-frequency oscillator

(2) Reducing drive capability of the low-frequency oscillator

(3) Canceling double-drive operation of the high-frequency oscillator

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

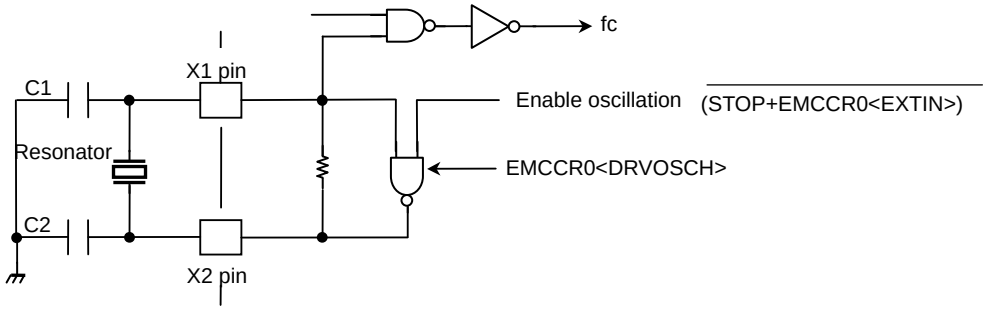
These features are specified using the EMCCR0 and EMCCR1 registers, as described below.

(1) Reducing drive capability of the high-frequency oscillator

Purpose:

To suppress noise generated by the high-frequency oscillator and to reduce power consumption of the high-frequency oscillator when an external resonator is connected.

Block diagram:



Description:

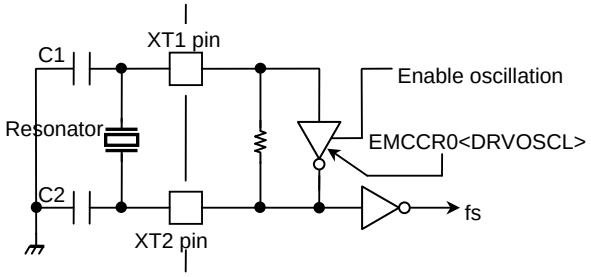
Setting the <DRVOSCH> bit of the EMCCR0 register to 0 reduces the high-frequency oscillator's drive capability. A system reset initializes the <DRVOSCH> bit to 1, so the high-frequency oscillator starts oscillating with normal drive capability upon power-on. The <DRVOSCH> bit should not be set to 0 when Vcc < 2.7V.

(2) Reducing drive capability of the low-frequency oscillator

Purpose:

To suppress noise generated by the low-frequency oscillator and to reduce power consumption of the low-frequency oscillator when an external resonator is connected.

Block diagram:



Description:

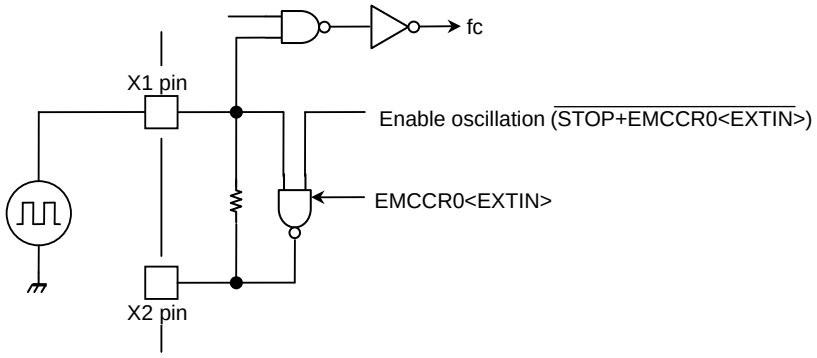
Setting the <DRVOSCL> bit of the EMCCR0 register to 0 reduces the low-frequency oscillator's drive capability. A system reset initializes the <DRVOSCL> bit to 1.

(3) Canceling double-drive operation of the high-frequency 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:



Description:

Setting the <EXTIN> bit of the EMCCR0 register to 1 causes the high-frequency oscillator to stop oscillation with the X2 pin driven high.

A system reset initializes the <EXTIN> bit to 0.

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

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

Purpose:

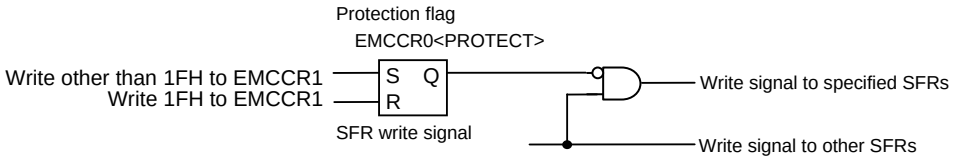
To prevent software or system lockups that may occur due to incoming noise.

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. Clock gear (Only EMCCR1 can be written.)
   SYSCR0, SYSCR1, SYSCR2, SYSCR3,
   EMCCR0

Block diagram:



Description:

Writing any code other than 1FH to the EMCCR1 register enables protection, 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

(1) HALT mode

Executing the HALT instruction causes the TMP91CW40 to enter one of the HALT modes–IDLE2, IDLE1 or STOP–as specified by the SYSCR2 <HALTM1:0> bits.

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

a. IDLE2: The CPU stops.
Each internal I/O 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

Internal I/O	SFR
SIO0	SC0MOD1<I2S0>
SIO1	SC1MOD1<I2S1>
SIO2	SC2MOD1<I2S2>
SIO3	SC3MOD1<I2S3>
AD converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

b. IDLE1: Only the oscillator, RTC(real-time clock) and MLD are operational.

c. STOP: The whole TMP91CW40 stops.

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

Table 3.3.3 TMP91CW40 Circuit Blocks in HALT Modes

HALT mode		IDLE2	IDLE1	STOP
SYSCR2<HALTM1:0>		11	10	01
Circuit block	CPU	OFF		
	I/O ports	Holding the states when the HALT instruction was executed		See Table 3.3.6
	TC1 to TC3, TC5 to TC8	ON	OFF	
	SIO0 to SIO3	Selectable programmatically on a block-by-block basis		
	AD converter			
	WDT			
	RTC, MLD	ON		
	LCDD			
	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, <IFF2:0>, 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 INT0, INT1, KWI0 to KWI3, INTRTC and INTALM0 to INTALM4 interrupts 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 TMP91CW40 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 internal 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	–	–	–
		INTWD	♦	×	×	–	–	–
		INT0,INT1, KWI0 to KWI3 Note 1)	♦	♦	♦ *1	◊	◊	◊ *1
		INTALM0 to INTALM4	♦	♦	×	◊	◊	×
		INTRTC	♦	♦	×	◊	◊	×
		INTTMR1 to INTTMR3,	♦	×	×	×	×	×
		INTTMR5 to INTTMR8	♦	×	×	×	×	×
		INTRX0 to INTRX3, INTTX0 to INTTX3	♦	×	×	×	×	×
		INTAD	♦	×	×	×	×	×
	RESET		Initializes the whole TMP91CW40.					

♦: Execution resumes with the interrupt service routine.

◊: Execution resumes with the instruction immediately following the HALT instruction. (The interrupt is not serviced.)

×: Cannot be used to exit a HALT mode.

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

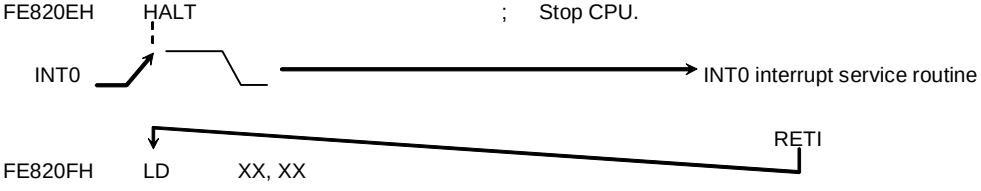
*1: The TMP91CW40 exits the HALT mode after the warm-up period timer expires.

Note 1: If the interrupt request level is greater than the mask level, an INT0 or INT1 interrupt signal which is programmed as level-sensitive must be held high until interrupt processing begins. Otherwise, the interrupt will not be serviced successfully.

Example of exiting a HALT mode

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

```
Address
FE8200H   LD     (P6FC), 01H      ;   Set P60 to INT0.
FE8203H   LD     (IIMC), 00H      ;   Set INT0 interrupt to rising-edge sensitive.
FE8206H   LD     (INTE0AD), 06H   ;   Set INT0 interrupt priority level to 6.
FE8209H   EI     5                ;   Set CPU interrupt priority level to 5.
FE820BH   LD     (SYSCR2), 28H    ;   Select IDLE1 mode.
FE820EH   HALT                    ;   Stop CPU.
```




```
FE820FH   LD     XX, XX
```

(3) Operation in HALT modes

a. IDLE2 mode

In IDLE2 mode, the CPU stops executing instructions and only the internal I/O functions 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.

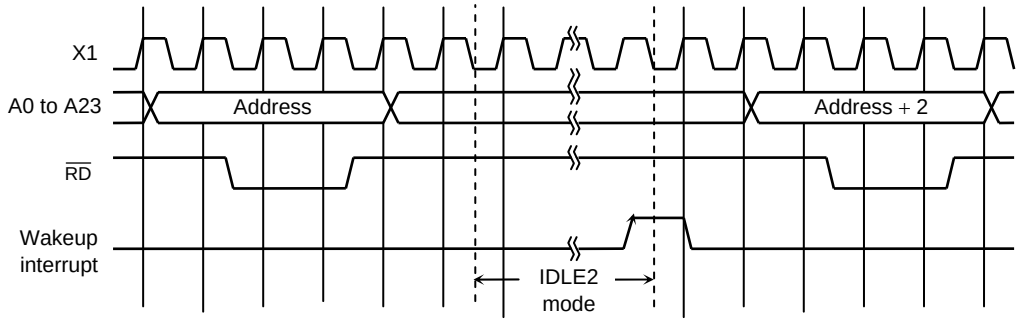


Figure 3.3.5 Example Timings for Exiting a HALT Mode (IDLE2 Mode) with an Interrupt

b. IDLE1 mode

In IDLE1 mode, the system clock stops while only the internal oscillator and time-of-day clock timer are 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.

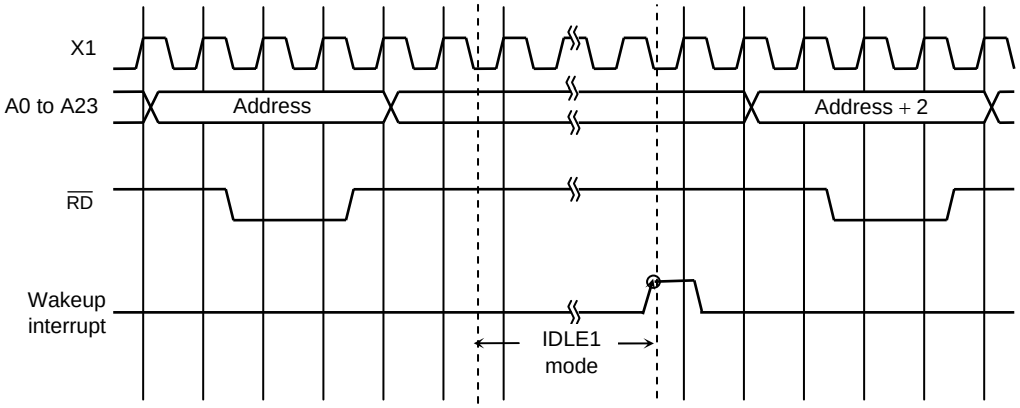


Figure 3.3.6 Example Timings for Exiting a HALT Mode (IDLE1 Mode) with an Interrupt

c. STOP mode

In STOP mode, the whole TMP91CW40 stops, including the internal oscillator. Pin states in STOP mode depend on the setting of the SYSCR2<DRVE> bit, as shown in Table 3.3.6.

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. Upon exiting STOP mode, the operation resumes according to the settings in the SYSCR0<RXEN>, <RXTEN> and <RSYSCK> bits. These bits must be set before executing the HALT instruction. The warm-up period is chosen through the SYSCR2<WUPTM1:0> bits, as shown in Table 3.3.5.

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

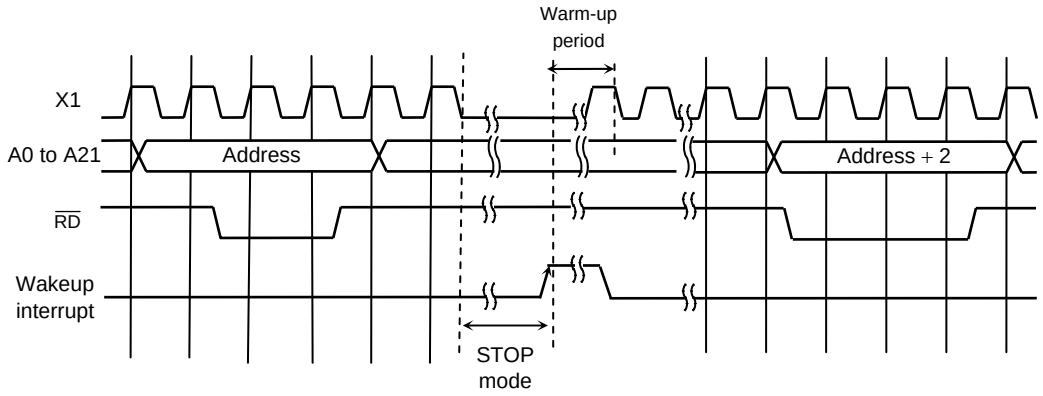


Figure 3.3.7 Example Timings for Exiting a HALT Mode (STOP Mode) with an Interrupt

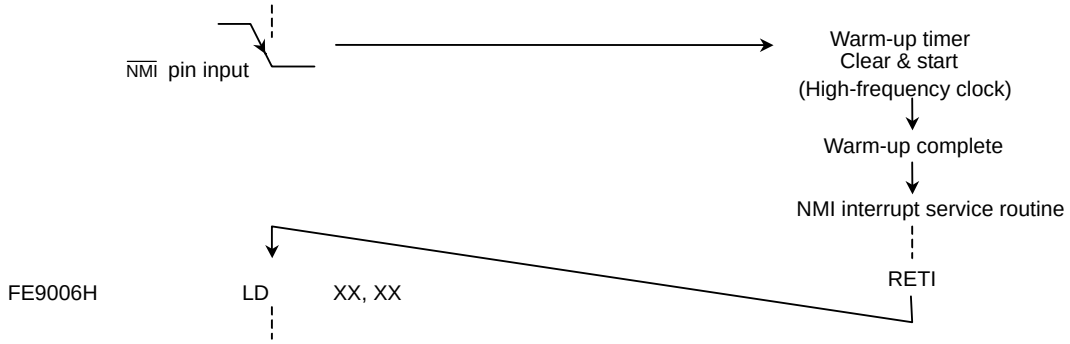
3.3.5 Example Warm-Up Period Settings (when exiting STOP mode)

fc = 27 MHz, fs = 32.768 kHz

SYSCR0 <RSYSCK>	SYSCR2<WUPTM1:0>		
	01 (2^8)	10 (2^{14})	11 (2^{16})
0 (fc)	9.5 μs	0.607 ms	2.427 ms
1 (fs)	7.8 ms	500 ms	2000 ms

Example: Entering STOP mode while using the low-frequency clock, exiting STOP mode with an NMI interrupt, and then resuming operation with the high-frequency clock

```
Address
            SYSCR0   EQU   00E0H
            SYSCR1   EQU   00E1H
            SYSCR2   EQU   00E2H                ;
FE8FFDH              LD    (SYSCR1), 08H           fSYS =fs/2
FE9000H              LD    (SYSCR2), X–1001X1B  ;  Warm-up time = 2^14/fc
FE9002H              LD    (SYSCR0), 01100000B  ;  Enable high-frequency clock after exiting STOP mode
FE9005H              HALT
```




```
FE9006H              LD    XX, XX
```

–: No change

Note: When different system clock frequencies are to be used before entering and after exiting STOP mode as shown above, if a wakeup interrupt is accepted while the HALT instruction is being executed (a period of 6 states), STOP mode may be exited without the system clock frequency being changed. In a system where interrupts are input during execution of the HALT instruction, use the same system clock frequency before entering and after exiting STOP mode.

Table 3.3.6 TMP91CW40 Input Buffer State Table

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	
P50-52	KWI0-KWI2	OFF	ON	ON (by read)	ON	OFF	ON	OFF	ON	OFF	*1
P53	KWI3										*1
	$\overline{\text{ADTRG}}$										*1
P60	INT0 input	ON		ON		ON		ON		ON	
P61	INT1 input										
P62	–		–		–		–		–		
P70	ECNT1 input		ON		ON		ON		OFF		
P71	ECNT2 input										
P72	ECNT3 input										
P73	ECIN1 input										
P74	ECIN2 input										
P75	ECIN3 input										
P80-83	–		–		–		–		–		
P90	–										
P91	RXD0 input		ON		ON		ON		OFF		
P92	SCLK0 input										
	$\overline{\text{CTS0}}$ input										
P93	–		–		–		–		–		
P94	RXD1 input		ON		ON		ON		OFF		
P95	SCLK1 input										
	$\overline{\text{CTS1}}$ input										
PA0	–		–		–		–		–		
PA1	RXD2 input		ON		ON		ON		OFF		
PA2	SCLK2 input										
	$\overline{\text{CTS2}}$ input										
PA3	–		–		–		–		–		
PA4	RXD3 input		ON		ON		ON		OFF		
PA5	SCLK3 input										
	$\overline{\text{CTS3}}$ input										
P20-27	–										
P10-17	–										
P00-07	–										
PB0-B7	–										
$\overline{\text{NMI}}$	–			–		–		–	ON	–	
$\overline{\text{RESET}}$	–										
AM0,AM1	–										
X1	–						OFF		OFF		

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

OFF: The buffer is always turned off.

-: Not applicable

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

3.4 Interrupts

Interrupt processing is controlled by the CPU interrupt mask register SR <IFF2:0> and the on-chip interrupt controller.

The TMP91CW40 supports the following 43 interrupt sources:

- 9 CPU internal interrupts
  (Software interrupts and interrupts triggered when an undefined instruction is executed)
- 7 external interrupt pins ($\overline{\text{NMI}}$, INT0, INT1, KWI0 to KWI3)
- 27 internal I/O 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 <IFF2:0>. If the sent priority level is higher than or equal to the interrupt mask level, the CPU accepts the interrupt. The contents of the <IFF2:0> bits can be modified using the EI instruction in the format of EI num, where num is the value to be set in <IFF2:0>. For example, "EI 3" causes the CPU to accept maskable interrupts having a priority level of 3 or higher, as specified with the interrupt controller, as well as all nonmaskable interrupts. The DI instruction, which sets <IFF2:0> 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 TMP91CW40 supports micro DMA mode, where the CPU automatically transfers data (1 byte, 2 bytes or 4 bytes). This mode enables faster data transfer to internal/external memory and internal I/O.

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.

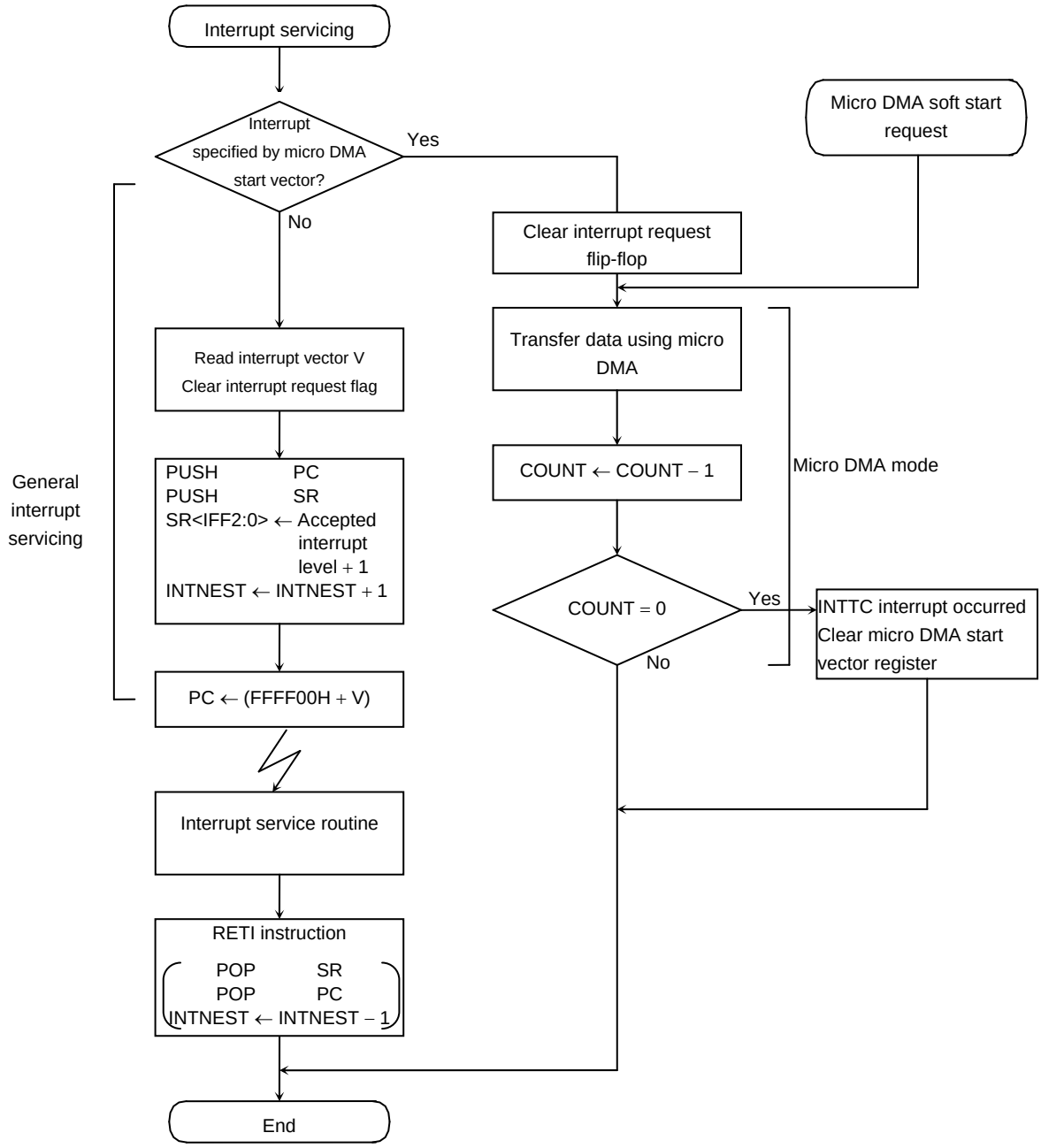


Figure 3.4.1 Overall Interrupt Servicing Flow

3.4.1 General Interrupt Servicing

The CPU performs the following operations once it accepts an interrupt. These 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 <IFF2:0> to one level higher than the accepted interrupt level. If the level is 7, however, the CPU sets <IFF2:0> 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 (1.33 μs at 27 MHz) in the best case (with 16-bit data bus and 0-wait cycles).

Upon 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 higher than or equal to the value stored in the CPU's <IFF2:0> bits. The CPU then sets the <IFF2:0> 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 in an interrupt service routine to prohibit nesting of maskable interrupts.

Upon a system reset, the <IFF2:0> bits are initialized to 7 so that maskable interrupts are disabled.

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

Table 3.4.1 Interrupt Vector Table

Default Priority	Type	Interrupt Source	Vector Value	Vector Reference Address	Micro DMA Start Vector
1	Non-maskable	Reset or SWI0 instruction	0000H	FFFF00H	–
2		SWI1 instruction	0004H	FFFF04H	–
3		INTUNDEF: 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, KWI0 to KWI3 pins	002CH	FFFF2CH	0BH
13		INTALM0: ALM0 (8192 Hz)	0030H	FFFF30H	0CH
14		INTALM1: ALM1 (512 Hz)	0034H	FFFF34H	0DH
15		INTALM2: ALM2 (64 Hz)	0038H	FFFF38H	0EH
16		INTALM3: ALM3 (2 Hz)	003CH	FFFF3CH	0FH
17		INTALM4: ALM4 (1 Hz)	0040H	FFFF40H	10H
18		INTTMR5: 8-bit timer 5 (TC5)	0044H	FFFF44H	11H
19		INTTMR6: 8-bit timer 6 (TC6)	0048H	FFFF48H	12H
20		INTTMR7: 8-bit timer 7 (TC7)	004CH	FFFF4CH	13H
21		INTTMR8: 8-bit timer 8 (TC8)	0050H	FFFF50H	14H
22		INTTMR1: 16-bit timer 1 (TC1)	0054H	FFFF54H	15H
23		INTTMR2: 16-bit timer 2 (TC2)	0058H	FFFF58H	16H
24		Reserved	005CH	FFFF5CH	–
25		Reserved	0060H	FFFF60H	–
26		Reserved	0064H	FFFF64H	–
27		Reserved	0068H	FFFF68H	–
28		INTTMR3: 16-bit timer 3 (TC3)	006CH	FFFF6CH	1BH
29		Reserved	0070H	FFFF70H	–
30		INTRX0: Serial receive (Channel 0)	0074H	FFFF74H	1DH
31		INTTX0: Serial transmit (Channel 0)	0078H	FFFF78H	1EH
32		INTRX1: Serial receive (Channel 1)	007CH	FFFF7CH	1FH
33		INTTX1: Serial transmit (Channel 1)	0080H	FFFF80H	20H
34		Reserved	0084H	FFFF84H	–
35		Reserved	0088H	FFFF88H	–
36		INTRX2: Serial receive (Channel 2)	008CH	FFFF8CH	23H
37		INTTX2: Serial transmit (Channel 2)	0090H	FFFF90H	24H
38		INTRX3: Serial receive (Channel 3)	0094H	FFFF94H	25H
39		INTTX3: Serial transmit (Channel 3)	0098H	FFFF98H	26H
40		INTAD: AD conversion complete	009CH	FFFF9CH	27H
41		INTTC0: Micro DMA complete (Channel 0)	00A0H	FFFFA0H	—
42		INTTC1: Micro DMA complete (Channel 1)	00A4H	FFFFA4H	—
43		INTTC2: Micro DMA complete (Channel 2)	00A8H	FFFFA8H	—
44		INTTC3: Micro DMA complete (Channel 3)	00ACH	FFFFACH	—
45		INTRTC: RTC (Alarm interrupt)	00B0H	FFFFB0H	2CH
		(Reserved) : (Reserved)	00B4H : 00FCH	FFFFB4H : FFFFFCH	– : –

3.4.2 Micro DMA

In addition to general interrupt servicing, the TMP91CW40 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 realized in cooperation with the CPU, micro DMA requests are ignored and remain pending if the CPU executes the HALT instruction and enters a standby state.

(1) Micro DMA operation

If an interrupt specified with the micro DMA start 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 <IFF2:0> = 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 start vector register (DMAnV) to 0, thus disabling subsequent start of the micro DMA and terminating micro DMA servicing. Even if the decremented counter does not reach zero, the CPU terminates micro DMA servicing unless burst transfer is specified. In this 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 the priority level is set to 1 to 6 and this interrupt request is generated before it is set for micro DMA transfer, the CPU will perform general interrupt servicing.

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 that 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 each 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.

Note: If the priority level of micro DMA is set higher than that of other interrupts, CPU operates as follows.
In case INTxxx interrupt is generated first and then INTyyy interrupt is generated between checking “Interrupt specified by micro DMA start vector” (in the Figure 3.4.1) and reading interrupt vector with setting below. The vector shifts to that of INTyyy at the time.
This is because the priority level of INTyyy is higher than that of INTxxx.
In the interrupt routine, CPU reads the vector of INTyyy because cheking of micro DMA has finished. And INTyyy is generated regardless of transfer counter of micro DMA.
INTxxx: level 1 without micro DMA
INTyyy: level 6 with micro DMA

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 I/O to memory, memory to I/O, and I/O to I/O. 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 19 interrupt sources as shown in Table 3.4.1 as well as a soft start.

Figure 3.4.2 shows micro DMA cycles for 2-byte transfer in the transfer destination address increment mode (with all address areas accessed with a 16-bit data bus, no wait cycles, and even-numbered source/destination addresses).

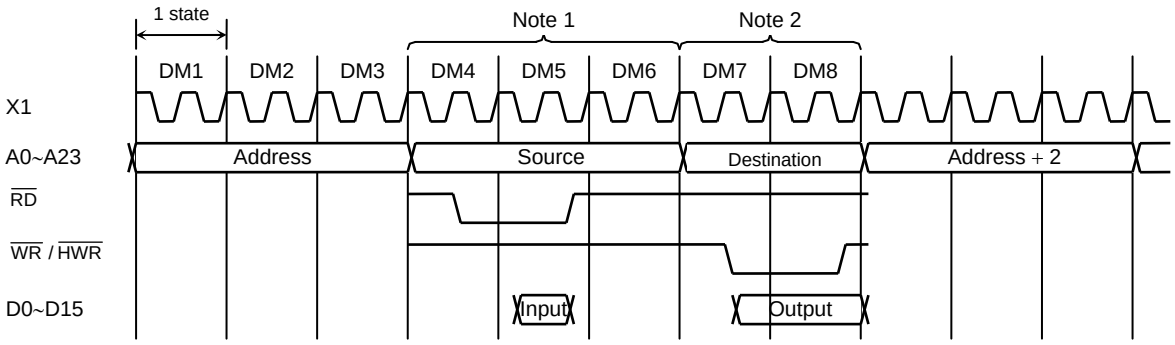


Figure 3.4.2 Micro DMA Cycles

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

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

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

(2) Soft start

In addition to interrupt sources, the micro DMA can also be started by software. This soft start feature enables the micro DMA to be started upon the detection of a write cycle to the DMAR register.

Writing 1 to each bit in the DMAR register starts a micro DMA transfer in the corresponding channel. When the transfer is completed, the bit is automatically cleared to 0. Only one channel can be started at a time. (Do not write 1 to more than one bit in the DMAR register at the same time.)

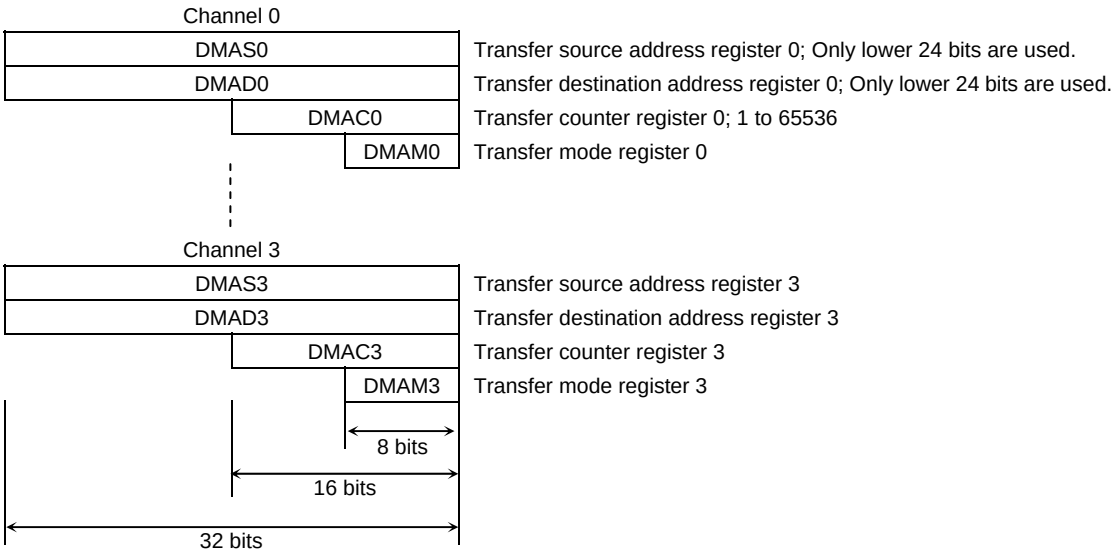
A DMAR register bit must be verified to be 0 before it can be set to 1 again. If read 1, micro DMA transfer isn't started yet.

When a burst transfer is specified in the DMAB register, the micro DMA channel that has been once started continues transferring data until the micro DMA transfer counter reaches zero. If execute soft start during micro DMA transfer by interrupt source, micro DMA transfer counter doesn't change. Don't use Read-modify-write instruction to avoid writing to other bits by mistake.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Software request register	89H (Read-modify-write instructions are prohibited)					DMAR3	DMAR2	DMAR1	DMAR0
							R/W			
							0	0	0	0
							1: DMA request			

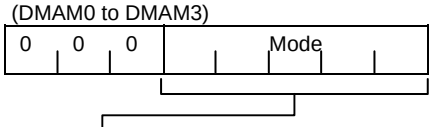
(3) Transfer control registers

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.



(4) Transfer mode registers: DMAM0 to DMAM3

(DMAM0 to DMAM3)



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

Execution time

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

0	0	0	Z	Z	Destination address increment mode ..... I/O to memory (DMADn+) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC occurs	8 states (593 ns) Byte/word transfer 12 states (889 ns) 4-byte transfer
0	0	1	Z	Z	Destination address decrement mode ..... I/O to memory (DMADn−) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC occurs	8 states (593 ns) Byte/word transfer 12 states (889 ns) 4-byte transfer
0	1	0	Z	Z	Source address increment mode..... Memory to I/O (DMADn) ← (DMASn+) DMACn ← DMACn − 1 if DMACn = 0 then INTTC occurs	8 states (593 ns) Byte/word transfer 12 states (889 ns) 4-byte transfer
0	1	1	Z	Z	Source address decrement mode ..... Memory to I/O (DMADn) ← (DMASn−) DMACn ← DMACn−1 if DMACn = 0 then INTTC occurs	8 states (593 ns) Byte/word transfer 12 states (889 ns) 4-byte transfer
1	0	0	Z	Z	Fixed address mode ..... I/O to I/O (DMADn) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC occurs	8 states (593 ns) Byte/word transfer 12 states (889 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 (370 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, "I/O" means a fixed address while "memory" means an address that can be incrementaed 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 = 27 MHz, cloock gear = x1 (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.

For each of the 25 interrupt channels there is an interrupt request flag, interrupt priority register and micro DMA start vector register. The interrupt request flag is used to latch an interrupt request issued by peripherals.

This flag is cleared in the following cases:

- Reset
- The CPU accepts the interrupt and reads the vector for the interrupt.
- An instruction that clears the interrupt is executed. (A DMA start 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 INTE1ALM0) provided for each interrupt source. Six levels of priority (1 to 6) can be set. An interrupt 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 simutaneously, the CPU accepts interrupts according to default priorities. Reading bits 3 and 7 of an interrupt priority register obtains the status of the interrupt request flag, indicating whether an interrupt request is present for the corresponding channel.

The interrupt controller determines the interrupt with the highest priority among interrupts occuring simultaneously if any, and sends it 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 <IFF2:0> bits of the status register (SR). The CPU accepts the interrupt if its priority level is higher than the register value. It then sets the <IFF2:0> bits to the accepted interrupt level plus one, so that only interrupt requests having a priority level higher than or equal to the register value can be accepted while the current interrupt is handled. Upon completion of interrupt servicing (with the execution of the RETI instruction), the <IFF2:0> bits are restored to the value before the interrupt occurred which has been saved on the stack.

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

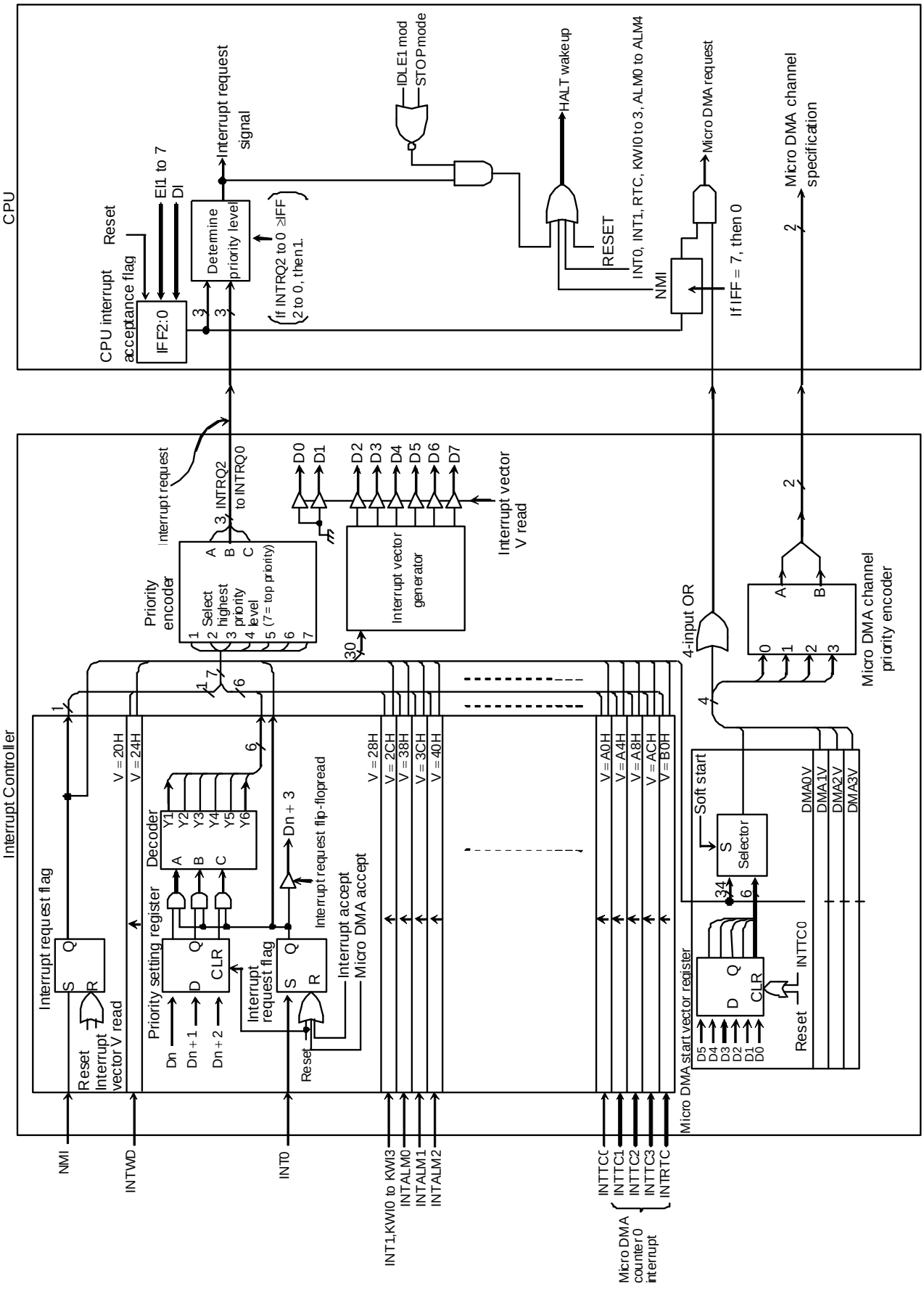
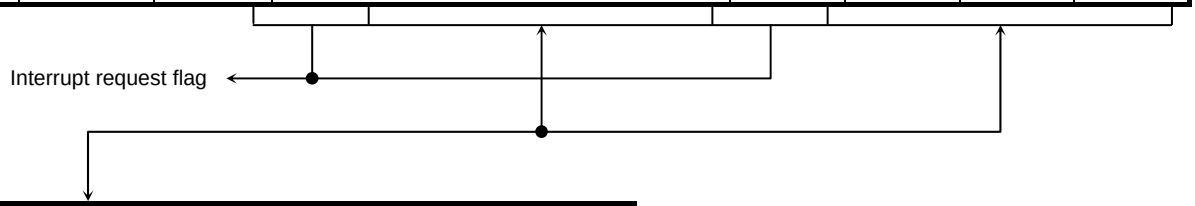


Figure 3.4.3 Interrupt Controller Block Diagram

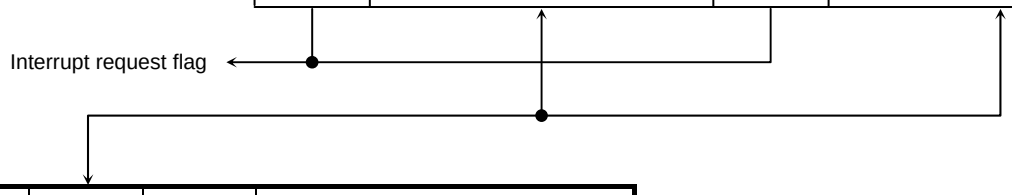
(1) Interrupt priority registers

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
INTE1ALM0	INT1& INTALM0 enable	91H	INTALM0				INT1			
			IA0C	IA0M2	IA0M1	IA0M0	I1C	I1M2	I1M1	I1M0
			R	R			R	R		
			0	0	0	0	0	0	0	0
INTEALM12	INTALM1 & INTALM2 enable	92H	INTALM2				INTALM1			
			IA2C	IA2M2	IA2M1	IA2M0	IA1C	IA1C	IA1M2	IA1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTEALM34	INTALM3 & INTALM4 enable	93H	INTALM4				INTALM3			
			IA4C	IA4M2	IA4M1	IA4M0	IA3C	IA3C	IA3M2	IA3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETMR56	INTTMR5 & INTTMR6 enable	94H	INTTMR6 (TC6)				INTTMR5 (TC5)			
			ITM6C	ITM6M2	ITM6M1	ITM6M0	ITM5C	ITM5M2	ITM5M1	ITM5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETMR78	INTTMR7 & INTTMR8 enable	95H	INTTMR8 (TC8)				INTTMR7 (TC7)			
			ITM8C	ITM8M2	ITM8M1	ITM8M0	ITM7C	ITM7M2	ITM7M1	ITM7M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETMR12	INTTMR1 & INTTMR2 enable	96H	INTTMR2 (TC2)				INTTMR1 (TC1)			
			ITM2C	ITM2M2	ITM2M1	ITM2M0	ITM1C	ITM1M2	ITM1M1	ITM1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETMR3	INTTMR3 enable	99H	–				INTTMR3 (TC3)			
			–	–	–	–	ITM3C	ITM3M2	ITM3M1	ITM3M0
			–	–			R	R/W		
			Write “0”.				0	0	0	0



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

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTES0	Interrupt enable serial 0	9AH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	Interrupt enable serial 1	9BH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTRTC	INTRTC enable	9CH	–				INTRTC			
			–	–	–	–	IRX2C	IRX2M2	IRX2M1	IRX2M0
			–	–			R	R/W		
			Write “0”.				0	0	0	0
INTES2	Interrupt enable serial 2	9DH	INTTX2				INTRX2			
			ITX2C	ITX2M2	ITX2M1	ITX2M0	IRX2C	IRX2M2	IRX2M1	IRX2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES3	Interrupt enable serial 3	9EH	INTTX3				INTRX3			
			ITX3C	ITX3M2	ITX3M1	ITX3M0	IRX3C	IRX3M2	IRX3M1	IRX3M0
			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 interrupt priority level to 1.
0	1	0	Set interrupt priority level to 2.
0	1	1	Set interrupt priority level to 3.
1	0	0	Set interrupt priority level to 4.
1	0	1	Set interrupt priority level to 5.
1	1	0	Set interrupt priority level to 6.
1	1	1	Disable interrupt requests.

(2) External interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt input control	8CH (Read-modify-write instructions are prohibited)	–	–	–	I1EDGE	I1LE	I0EDGE	I0LE	NMIREE
			W							
			0	0	0	0	0	0	0	0
			Always write "0".			INT1 edge polarity 0: Rising 1: Falling	INT1 sensitivity 0: Edge 1: Level	INT0 edge polarity 0: Rising 1: Falling	INT0 sensitivity 0: Edge 1: Level	1: Also triggered by $\overline{\text{NMI}}$ rising edge

INT1 sensitivity

0	Edge-triggered
1	Level-sensitive

INT0 sensitivity

0	Edge-triggered
1	Level-sensitive

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

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

Note: When INT1 is is set to be level-sensitive, the key-on wakeup function must be disabled.

(3) Interrupt request flag clear register

An interrupt request flag can be cleared by writing a micro DMA start 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 execution of 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 (Read-modify-write instructions are prohibited)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					0	0	0	0	0	0
					Writing a DMA start vector clears the corresponding interrupt request flag.					

(4) Micro DMA start vector registers

A micro DMA start vector register specifies which interrupt source is assigned to micro DMA processing. The interrupt source having the micro DMA start vector specified in this register is assigned as a micro DMA request source.

When the micro DMA transfer counter reaches zero, 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 start vector register, thus clearing the micro DMA request source for the channel. If it is necessary to continue micro DMA processing, the micro DMA start vector register must be set again in the service routine for the micro DMA transfer complete interrupt.

If the same vector is set in two or more micro DMA start vector registers at the same time, the channel having the smallest number takes precedence. Therefore, if the same vector is set in the micro DMA start vector registers of two channels, micro DMA transfer is first performed with the smaller-numbered channel until it completes. Unless the interrupt controller reloads the micro DMA start vector for this channel, micro DMA transfer is then performed with the larger-numbered channel (micro DMA chaining).

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

(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 each micro DMA channel to 1.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA software request register	89H (Read-modify-write instructions are prohibited)					DMAR3	DMAR2	DMAR1	DMAR0
							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 instruction execution unit and the bus interface unit of this CPU operate independently. Therefore, after accepting an interrupt the CPU may fetch an instruction that clears the interrupt request flag for this interrupt(Note) immediately before the interrupt is about to be generated. In this case, the CPU may execute this interrupt request clear instruction after accepting the interrupt request but before reading the interrupt vector for this interrupt. If this happens, the CPU reads "0008H" (interrupt vector cleared) and reads the interrupt vector from address FFFF08H.

To avoid the above situation, make sure to execute the DI instruction before an instruction for clearing an interrupt request flag. After the clear instruction is executed, at least one instruction must be inserted before the EI instruction is executed to re-enable interrupts.(e.g., "NOP" × 1 times) 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 (<IFF2:0> bits of the status register SR), the DI instruction must be executed to disable interrupts before executing the POP SR instruction.

In addition, note the following two exceptional circuits which demand special attention:

When INT0 or INT1 is set as a level-sensitive interrupt	When INT0 or INT1 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 Q output. Changing 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 level-sensitive INT0 is used to exit HALT mode, INT0 must also be held high once it is driven from low to high until HALT mode is exited. (Ensure that it is not temporarily driven low due to noise during that period.)
	When INT0 is changed from level-sensitive to edge-triggered, 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

INTRX	Clearing the interrupt request flip-flop requires a system reset or reading the serial channel receive buffer. It cannot be cleared by an instruction.

Note: The following instructions or pin state transition are equivalent to an instruction that clears an interrupt request flag:

INT0/INT1: Instruction that changes the pin mode to level-sensitive after an interrupt is generated in edge-triggered mode.

Change in the pin input level (from high to low) after an interrupt request is generated in level-sensitive mode

INTRX: Instruction that reads the receive buffer.

3.5 I/O Ports

The TMP91CW40 has a total of 69 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 CPU or internal functions. Table 3.5.1 shows the functions of the port pins of the TMP91CW40. 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 I/O Ports

Port Name	Pin Name	Number of Pins	Direction	Direction Programmability	P-OD	Alternate Functions
Port 5	P50 to P53	4	Input	(Fixed)		AN0 to AN3, $\overline{\text{ADTRG}}$ (P53)
						KWI0 to KWI3
Port 6	P60	1	Input	(Fixed)		INT0
	P61	1	Input/output	Bit		INT1
	P62	1	Input/output	Bit		$\overline{\text{ALARM}}$
Port 7	P70	1	Input/output	Bit		ECNT1
	P71	1	Input/output	Bit		ECNT2
	P72	1	Input/output	Bit		ECNT3, $\overline{\text{DVO}}$, $\overline{\text{MLDALM}}$
	P73	1	Input/output	Bit		ECIN1
	P74	1	Input/output	Bit		ECIN2
	P75	1	Input/output	Bit		ECIN3
Port 8	P80	1	Input/output	Bit	○	TC5OUT
	P81	1	Input/output	Bit	○	TC6OUT
	P82	1	Input/output	Bit	○	TC7OUT
	P83	1	Input/output	Bit	○	TC8OUT
Port 9	P90	1	Input/output	Bit	○	TXD0
	P91	1	Input/output	Bit		RXD0
	P92	1	Input/output	Bit		SCLK0/ $\overline{\text{CTS0}}$
	P93	1	Input/output	Bit	○	TXD1
	P94	1	Input/output	Bit		RXD1
	P95	1	Input/output	Bit		SCLK1/ $\overline{\text{CTS1}}$
Port A	PA0	1	Input/output	Bit	○	TXD2
	PA1	1	Input/output	Bit		RXD2
	PA2	1	Input/output	Bit		SCLK2/ $\overline{\text{CTS2}}$
	PA3	1	Input/output	Bit	○	TXD3
	PA4	1	Input/output	Bit		RXD3
	PA5	1	Input/output	Bit		SCLK3/ $\overline{\text{CTS3}}$
		8	Output	(Fixed)		SEG0 to SEG7
Port 2	P20 to P27	8	Input/output	Bit		SEG8 to SEG15
Port 1	P10 to P17	8	Input/output	Bit		SEG16 to SEG23
Port 0	P00 to P07	8	Input/output	Bit		SEG24 to SEG31
Port B	PB0 to PB7	8	Input/output	Bit		SEG32 to SEG39

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

Port	Pin Name	Direction/Function	I/O Register Settings				
			Pn	PnCR	PnFC	PnFC2	ODE
Port 5	P50 to P53	Input port	×	N/A	N/A	N/A	N/A
		AN0 to AN3 inputs Note 1)	×				
		KWI0 to KWI3 inputs Note 2)	×				
	P53	$\overline{\text{ADTRG}}$ input Note 3)	×				
Port 6	P60	Input port	×	N/A	0	N/A	N/A
		INT0 input	×		1		
	P61	Input port	×	0	0		
		Output port	×	1	0		
		INT1 input	×	0	1		
	P62	Input port	×	0	0		
		Output port	×	1	0		
		$\overline{\text{ALARM}}$ output	×	1	1		
Port 7	P70 to P75	Input port	×	0	N/A	N/A	N/A
		Output port	×	1			
	P70	ECNT1 input	×	0			
	P71	ECNT2 input	×	0			
	P72	ECNT3 input	×	0	0	0	
		$\overline{\text{DVO}}$ output	×	1	1	0	
		$\overline{\text{MLDALM}}$ output	×	1	×	1	
	P73	ECIN1 input	×	0	N/A	N/A	
	P74	ECIN2 input	×	0			
	P75	ECIN3 input	×	0			
Port 8	P80 to P83	Input port	×	0	0	N/A	—
		Output port (CMOS output)	×	1	0		0
		Output port (Open-drain output)	×	1	0		1
	P80	TC5OUT output (CMOS output)	×	1	1		0
		TC5OUT output (Open-drain output)	×	1	1		1
	P81	TC6OUT output (CMOS output)	×	1	1		0
		TC6OUT output (Open-drain output)	×	1	1		1
	P82	TC7OUT output (CMOS output)	×	1	1		0
		TC7OUT output (Open-drain output)	×	1	1		1
	P83	TC8OUT output (CMOS output)	×	1	1		0
		TC8OUT output (Open-drain output)	×	1	1		1

X: Don't care

Note 1: When P50 to P53 are used as input channels for the AD converter, the analog channel to be used is selected by the <ADCH2:0> bits of the ADMOD1 register.

Note 2: To use P50 to P53 as input ports of key-on wakeup, enable interrupts of KWI0 to KWI3 with KWIEN register.

Note 3: When P53 is used as $\overline{\text{ADTRG}}$ input, the <ADTRGE> bit of the ADMOD1 register is used to enable and disable AD conversion start by an external trigger.

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

Port	Pin Name	Direction/Function	I/O Register Settings			
			Pn	PnCR	PnFC	ODE
Port 9	P90, P93	Input port	×	0	0	—
		Output port (CMOS output)	×	1	0	0
		Output port (Open-drain output)	×	1	0	1
	P91, P94	Input port	×	0	N/A	N/A
		Output port	×	1		
	P92, P95	Input port	×	0	0	
		Output port	×	1	0	
	P90	TXD0 output (CMOS output)	×	1	1	0
		TXD0 output (Open-drain output)	×	1	1	1
	P91	RXD0 input	×	0	N/A	N/A
	P92	SCLK0 input	×	0	0	
		SCLK0 output	×	1	1	
		$\overline{\text{CTS0}}$ input	×	0	0	
	P93	TXD1 output (CMOS output)	×	1	1	0
		TXD1 output (Open-drain output)	×	1	1	1
	P94	RXD1 input	×	0	N/A	N/A
	P95	SCLK1 input	×	0	0	
		SCLK1 output	×	1	1	
		$\overline{\text{CTS1}}$ input	×	0	0	
Port A	PA0, PA3	Input port	×	0	0	—
		Output port (CMOS output)	×	1	0	0
		Output port (Open-drain output)	×	1	0	1
	PA1, PA4	Input port	×	0	N/A	N/A
		Output port	×	1		
	PA2, PA5	Input port	×	0	0	
		Output port	×	1	0	
	PA0	TXD2 output (CMOS output)	×	1	1	0
		TXD2 output (Open-drain output)	×	1	1	1
	PA1	RXD2 input	×	0	N/A	N/A
	PA2	SCLK2 input	×	0	0	
		SCLK2 output	×	1	1	
		$\overline{\text{CTS2}}$ input	×	0	0	
	PA3	TXD3 output (CMOS output)	×	1	1	0
		TXD3 output (Open-drain output)	×	1	1	1
	PA4	RXD3 input	×	0	N/A	N/A
	PA5	SCLK3 input	×	0	0	
		SCLK3 output	×	1	1	
		$\overline{\text{CTS3}}$ input	×	0	0	

X: Don't care

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

Port	Pin Name	Direction/Function	I/O Register Settings			
			Pn	PnCR	LCDSWn	MSEG07
Port 2	P00 to P07	Input port	×	0	0	N/A
		Output port	×	1	0	
		SEG8 to SEG15 outputs	×	×	1	
Port 1	P10 to P17	Input port	×	0	0	N/A
		Output port	×	1	0	
		SEG16 to SEG23 outputs	×	×	1	
Port 0	P20 to P27	Input port	×	0	0	N/A
		Output port	×	1	0	
		SEG24 to SEG31 outputs	×	×	1	
Port B	PB0 to PB7	Input port	×	0	0	N/A
		Output port	×	1	0	
		SEG32 to SEG39 outputs	×	×	1	
SEG	SEG0 to SEG7	Hi-z	N/A	N/A	0	0
		Low output (Note 4)			0	1
		SEG0 to SEG7 outputs			1	0

X: Don't care

Note 4: Do not set the LCDCR2<MSEG07> bit to 1 when the LCDCR<EDSP> bit is 1.

Upon reset, the port pins are configured as general-purpose input/output ports. Pins that can be programmed for either input or output are configured as input port pins. To use port pins for alternate functions, appropriate settings must be programmed.

3.5.1 Port 5 (P50 to P53)

Port 5 is a 4-bit input-only port that can also be used as analog input pins for the AD converter. P53 can also be used as an AD trigger input pin for the AD converter.

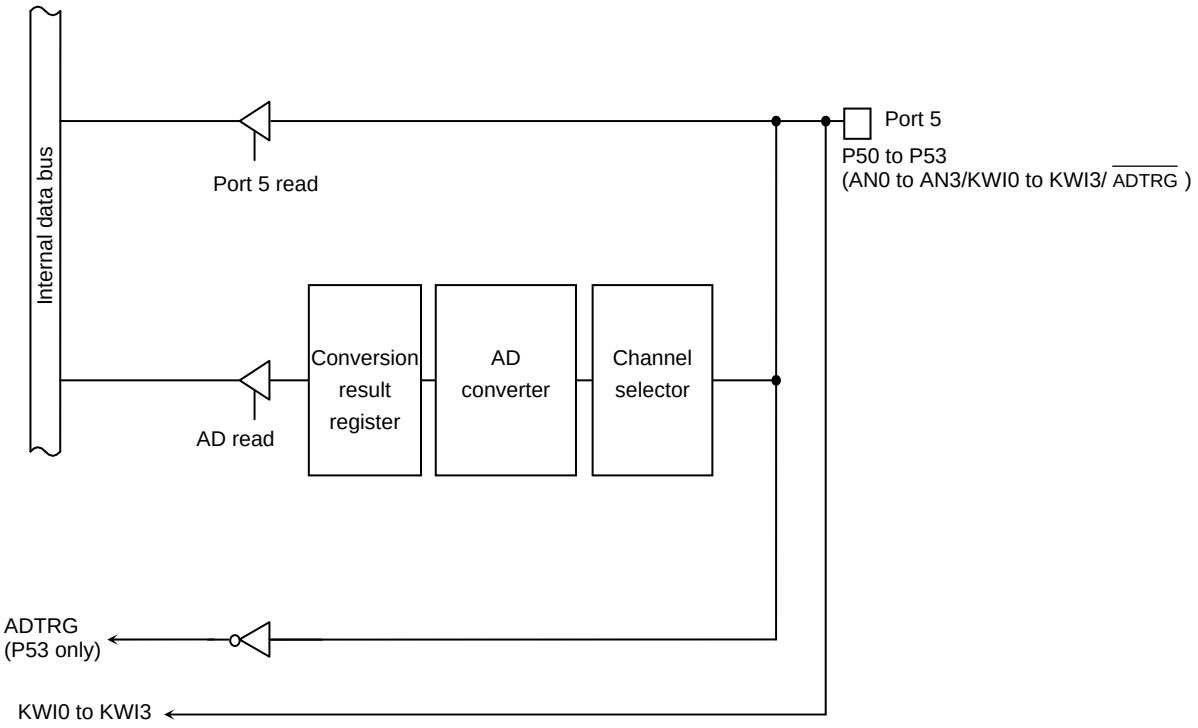


Figure 3.5.1 Port 5

Port 5 Register

		7	6	5	4	3	2	1	0
P5 (000DH)	Bit symbol					P53	P52	P51	P50
	Read/Write					R			
	After reset					Data from external port			

Key-on Wakeup Enable Register

		7	6	5	4	3	2	1	0
KWIEN (03A0H)	Bit symbol					KWI3EN	KWI2EN	KWI1EN	KWI0EN
	Read/Write					W			
	After reset					0	0	0	0
	Function					KWI interrupt input 0: Disable 1: Enable			

Figure 3.5.2 Port 5 Registers

Note 1: The KWIEN do not support read-modify-write operation.

Note 2: The AD converter mode register (ADMOD1) is used to select the AD converter input channel to be used and to enable and disable AD conversion start by an external trigger.

Note 3: The key-on wakeup enable register (KWIEN) is used to enable and disable key-on wakeup.

3.5.2 Port 6 (P60 to P62)

The port 6 is composed of a 1-bit input port (P60) and 2-bit input/output ports (P61 and P62) of which inputs and outputs can be specified in units of bits. A reset allows the port 6 to be put in input mode and bits 1 and 2 of the output latch register P6 are set to "1". Besides the input/output function, the port 6 inputs external interrupt and outputs alarm.

(1) P60 (INT0)

P60 can be used either as a general-purpose input port pin or an input pin for external interrupt INT0.

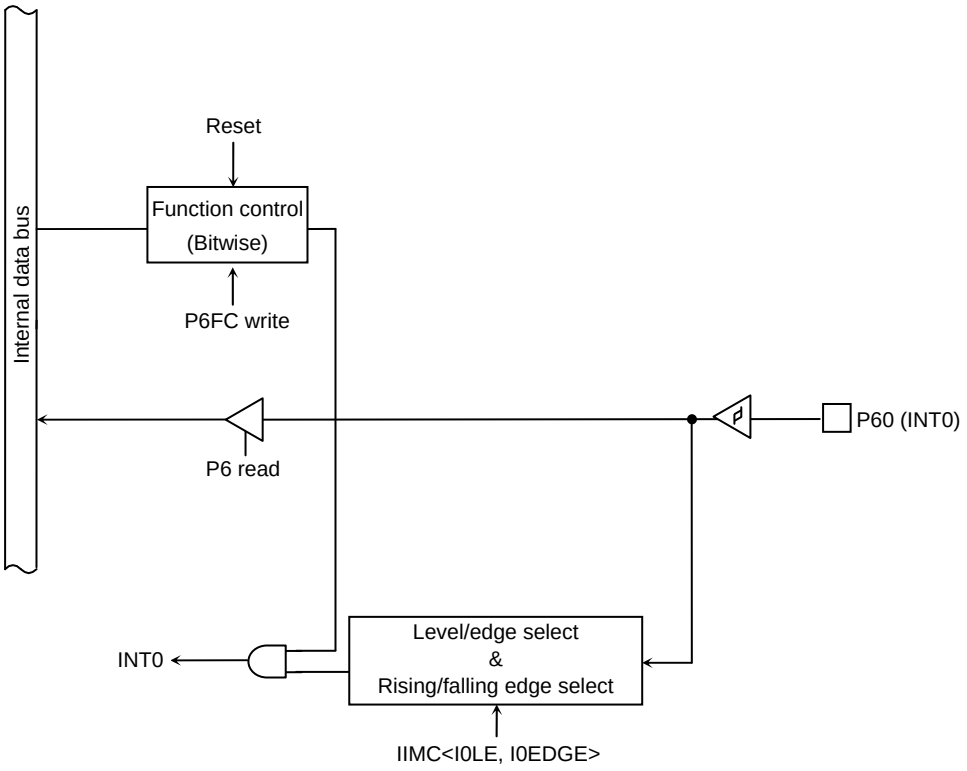


Figure 3.5.3 P60

(2) P61 (INT1)

P61 can be used either as a general-purpose input/output port pin or an input pin for external interrupt INT1.

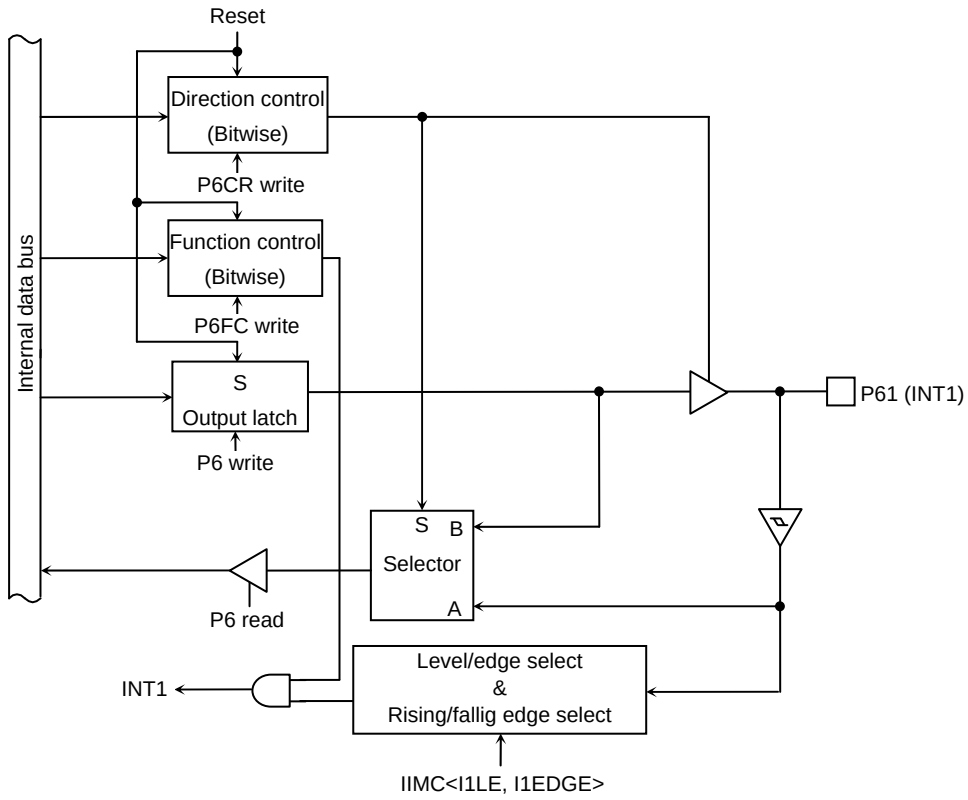


Figure 3.5.4 P61

(3) P62 ($\overline{\text{ALARM}}$)

P62 can be used either as a general-purpose input/output port pin or an output pin for the alarm function.

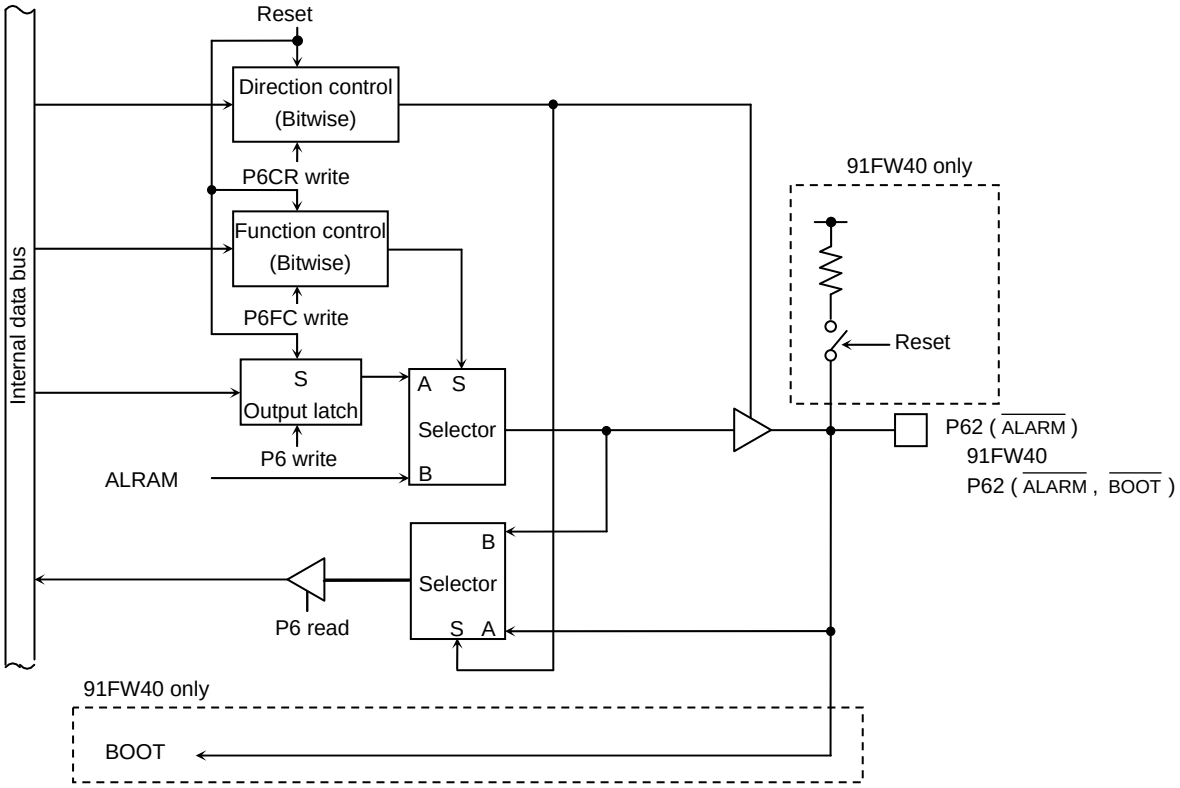


Figure 3.5.5 P62

Port 6 Register

P6 (0012H)

	7	6	5	4	3	2	1	0
Bit symbol						P62	P61	P60
Read/Write						R/W		R
After reset						Data from external port (Output latch register is set to 1.)		Data from external port

Port 6 Control Register

P6CR (0014H)

	7	6	5	4	3	2	1	0
Bit symbol						P62C	P61C	
Read/Write						W		
After reset						0	0	
Function						0: Input 1: Output	0: Input 1: Output	

Port 6 Function Register

P6FC (0015H)

	7	6	5	4	3	2	1	0
Bit symbol						P62F	P61F	P60F
Read/Write						W		
After reset						0	0	0
Function						0: Port 1: $\overline{\text{ALARM}}$ output	0: Port 1: INT1 input	0: Port 1: INT0 input

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

P60 INT0 input setting

P6FC<P60F>	1

P61 INT1 input setting

P6FC<P61F>	1
P6CR<P61C>	0

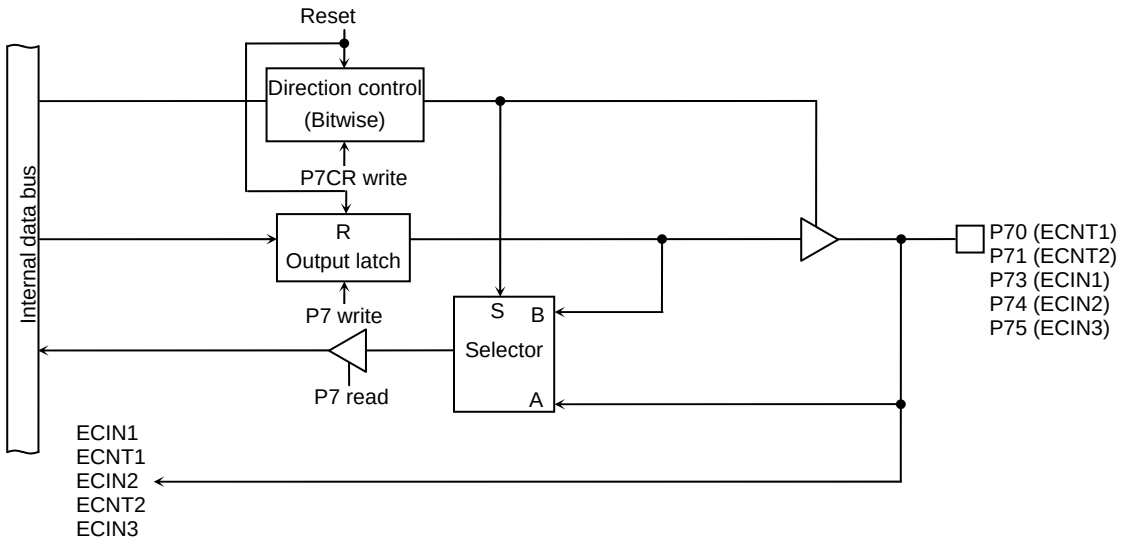
P62 $\overline{\text{ALARM}}$ output setting

P6CR<P62C>	1
P6FC<P62F>	1

Figure 3.5.6 Port 6 Registers

3.5.3 Port 7 (P70 to P75)

Port 7 is a 6-bit general-purpose I/O port. Each bit can be individually programmed for input or output. Reset operation initializes all pins to input port pins. In addition to functioning as a general-purpose input/output port, port 7 can also function as input pins for 16-bit timers 1, 2, and 3 (ECIN1, ECNT1, ECIN2, ECNT2, ECIN3, ECNT3) , a divider output pin ($\overline{\text{DVO}}$), a melody/alarm output pin ($\overline{\text{MLDALM}}$).



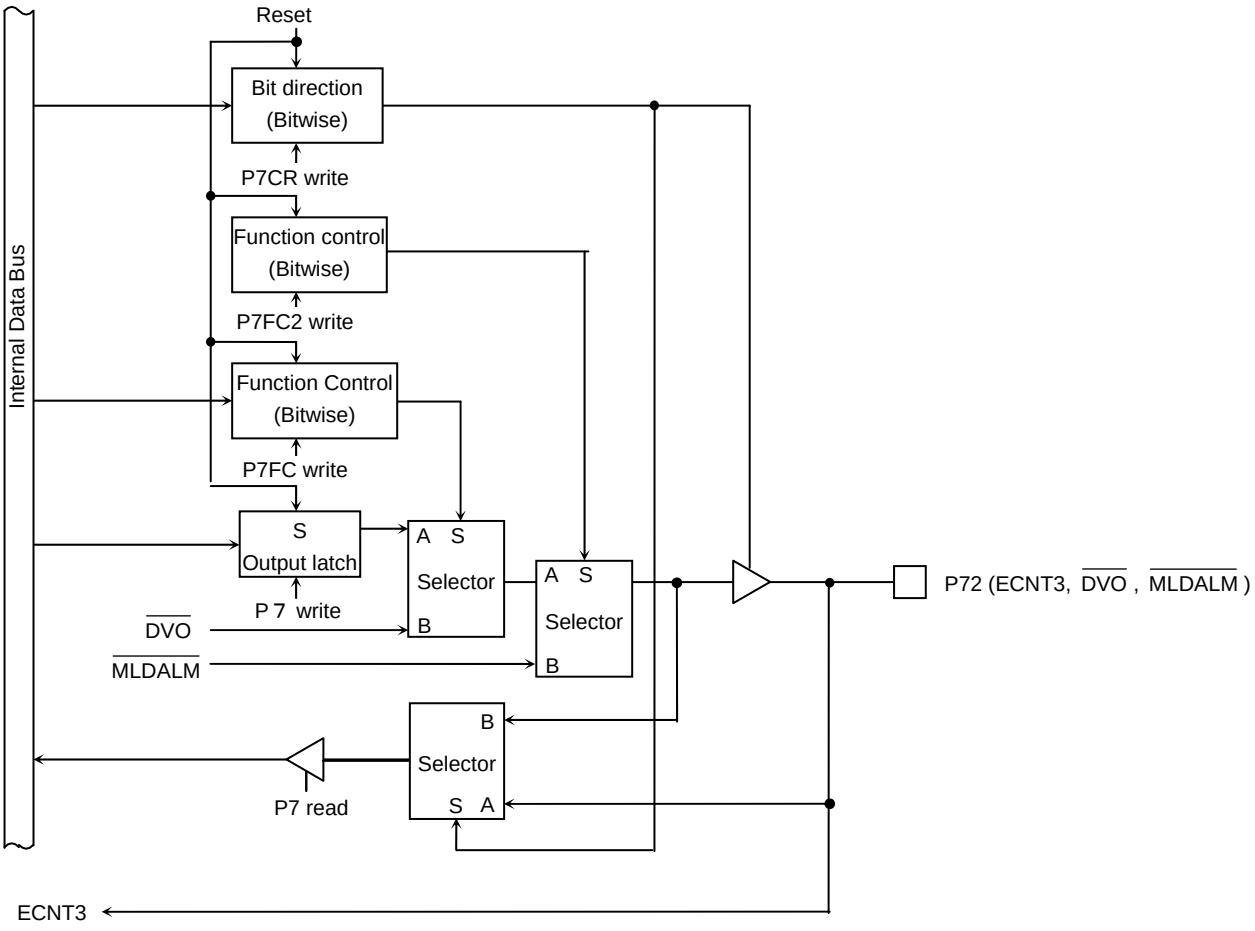


Figure 3.5.7 Port 7

Port 7 Register

P7 (0013H)

	7	6	5	4	3	2	1	0
Bit symbol			P75	P74	P73	P72	P71	P70
Read/Write			R/W					
After reset			Data from external port (Output latch register is reset to 0.)					

Port 7 Control Register

P7CR (0016H)

	7	6	5	4	3	2	1	0
Bit symbol			P75C	P74C	P73C	P72C	P71C	P70C
Read/Write			W					
After reset			0	0	0	0	0	0
Function			0: Input 1: Output					

Port 7 input/output setting

0	Input
1	Output

Port 7 Function Register

P7FC (0017H)

	7	6	5	4	3	2	1	0
Bit symbol						P72F		
Read/Write						W		
After reset						0		
Function						0: Port 1: $\overline{DVO}$		

P72 $\overline{DVO}$ output setting

P7FC<P72F>	1
P7FC2<P72F2>	0
P7CR<P72C>	1

Port 7 Function Register 2

P7FC2 (002DH)

	7	6	5	4	3	2	1	0
Bit symbol						P72F2		
Read/Write						W		
After reset						0		
Function						0: Port/$\overline{DVO}$ 1: $\overline{MLDALM}$		

P72 $\overline{MLDALM}$ output setting

P7FC2<P72F2>	1
P7CR<P72C>	1

Note 1: The P7CR, P7FC and P7FC2 do not support read-modify-write operation.

Note 2: The P70 to P75 (ECNT1 to ECNT3, ECIN1 to ECIN3) pins do not have a register bit for selecting the port or timer function. The input to these pins is always directed to 16-bit timers 1 to 3 even when they are used as general-purpose input pins.

Figure 3.5.8 Port 7 Registers

3.5.4 Port 8 (P80 to P83)

Port 8 is a 4-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins to input port pins. All bits in the output latch register (P8) are set to 1. In addition to functioning as a general-purpose input/output port, Port 8 can also function as output pins for 8-bit timers. This alternate function can be enabled by writing 1 to respective bits of the Port 8 function register (P8FC). Upon reset, the P8CR and P8FC registers are all initialized to 0, setting all pins as input port pins.

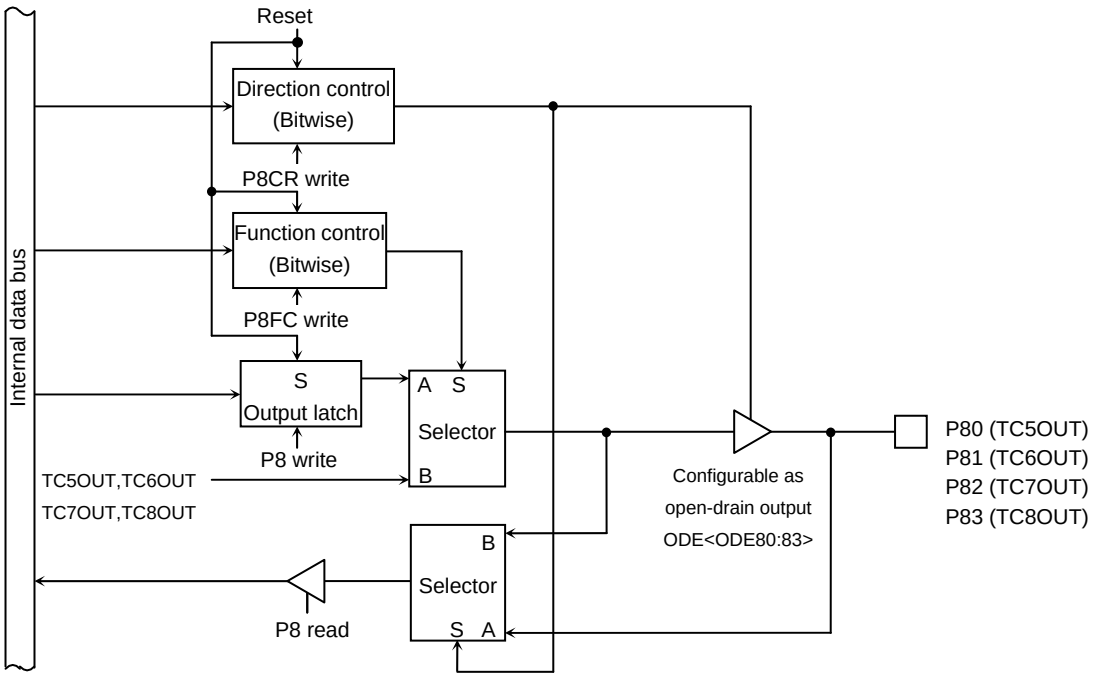


Figure 3.5.9 Port 8 (P80 to P83)

Port 8 Register

P8 (0018H)

	7	6	5	4	3	2	1	0
Bit symbol					P83	P82	P81	P80
Read/Write					R/W			
After reset					Data from external port (Output latch register is set to 1.)			

Port 8 Control Register

P8CR (001AH)

	7	6	5	4	3	2	1	0
Bit symbol					P83C	P82C	P81C	P80C
Read/Write					W			
After reset					0	0	0	0
Function					0: Input 1: Output			

→ Port 8 input/output setting

0	Input
1	Output

Port 8 Function Register

P8FC (001BH)

	7	6	5	4	3	2	1	0
Bit symbol					P83F	P82F	P81F	P80F
Read/Write					W	W	W	W
After reset					0	0	0	0
Function					0: Port 1: TC8OUT	0: Port 1: TC7OUT	0: Port 1: TC6OUT	0: Port 1: TC5OUT

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

→ P80 TC5OUT output setting

P8FC<P80F>	1
P8CR<P80C>	1

→ P81 TC6OUT output setting

P8FC<P81F>	1
P8CR<P81C>	1

→ P82 TC7OUT output setting

P8FC<P82F>	1
P8CR<P82C>	1

→ P83 TC8OUT output setting

P8FC<P83F>	1
P8CR<P83C>	1

Open-Drain Register

ODE (002FH)

	7	6	5	4	3	2	1	0
Bit symbol	ODEA3	ODEA0	ODE93	ODE90	ODE83	ODE82	ODE81	ODE80
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	PA3 open-drain output 0: Disable 1: Enable	PA0 open-drain output 0: Disable 1: Enable	P93 open-drain output 0: Disable 1: Enable	P90 open-drain output 0: Disable 1: Enable	P83 open-drain output 0: Disable 1: Enable	P82 open-drain output 0: Disable 1: Enable	P81 open-drain output 0: Disable 1: Enable	P80 open-drain output 0: Disable 1: Enable

→ Open-drain output setting

0	Disable
1	Enable

Figure 3.5.10 Port 8 Registers

3.5.5 Port 9 (P90 to P95)

Port 9 is a 6-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins as input port pins. All bits in the output latch register (P9) are set to "1". In addition to functioning as a general-purpose input/output port, Port 9 can also function as input/output pins for serial channels 0 and 1. This alternate function can be enabled by writing "1" to respective bits of the Port 9 function register (P9FC). Upon reset the P9CR and P9FC registers are all initialized to "0", setting all pins as input port pins.

(1) P90, P93 (TXD0, TXD1)

P90 and P93 can be used either as general-purpose input/output port pins or TXD output pins for serial channels 0 and 1.

The output buffer is configurable as an open-drain output using the <ODE90> and <ODE93> bits of the ODE register.

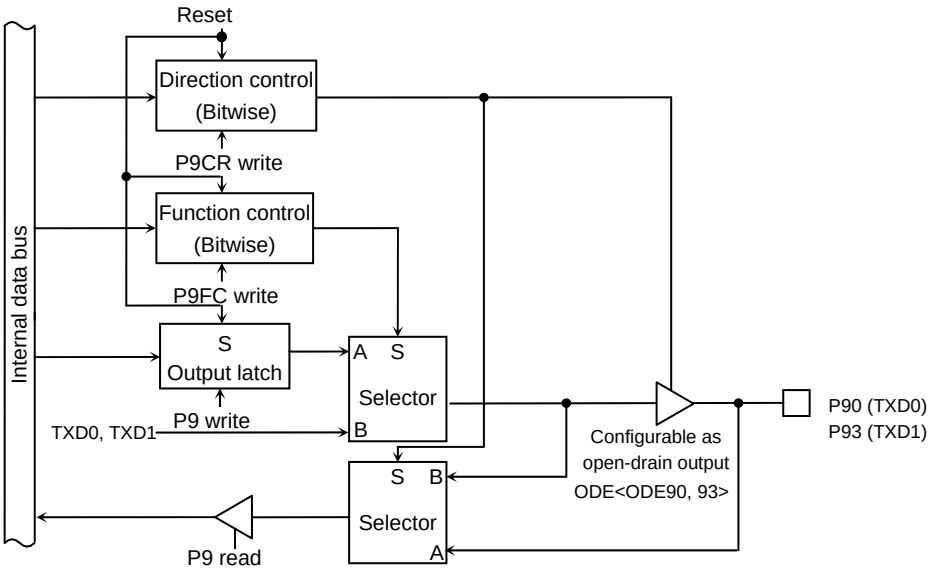


Figure 3.5.11 Port 9 (P90, P93)

(2) P91, P94 (RXD0, RXD1)

P91 and P94 can be used either as input/output port pins or RXD input pins for serial channels 0 and 1.

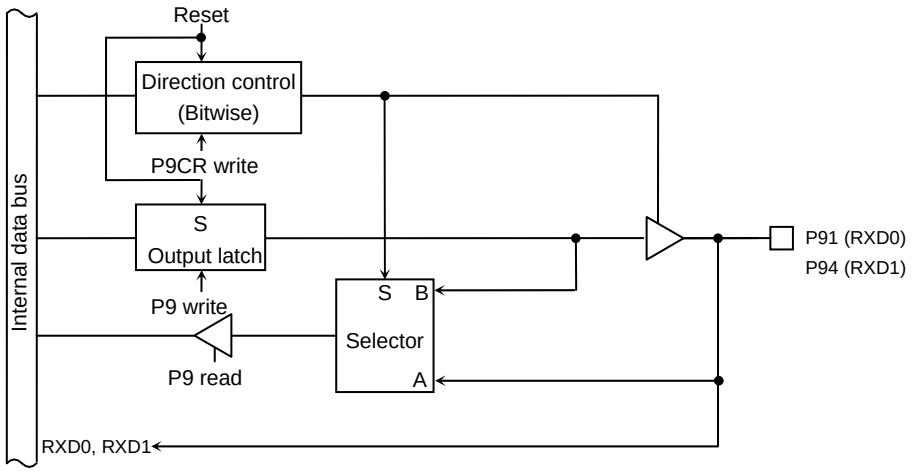


Figure 3.5.12 Port 9 (P91, P94)

(3) P92, P95 ($\overline{\text{CTS0}}$/SCLK0, $\overline{\text{CTS1}}$/SCLK1)

P92 and P95 can be used as general-purpose input/output port pins, $\overline{\text{CTS}}$ input pins for serial channels 0 and 1, or SCLK input/output pins.

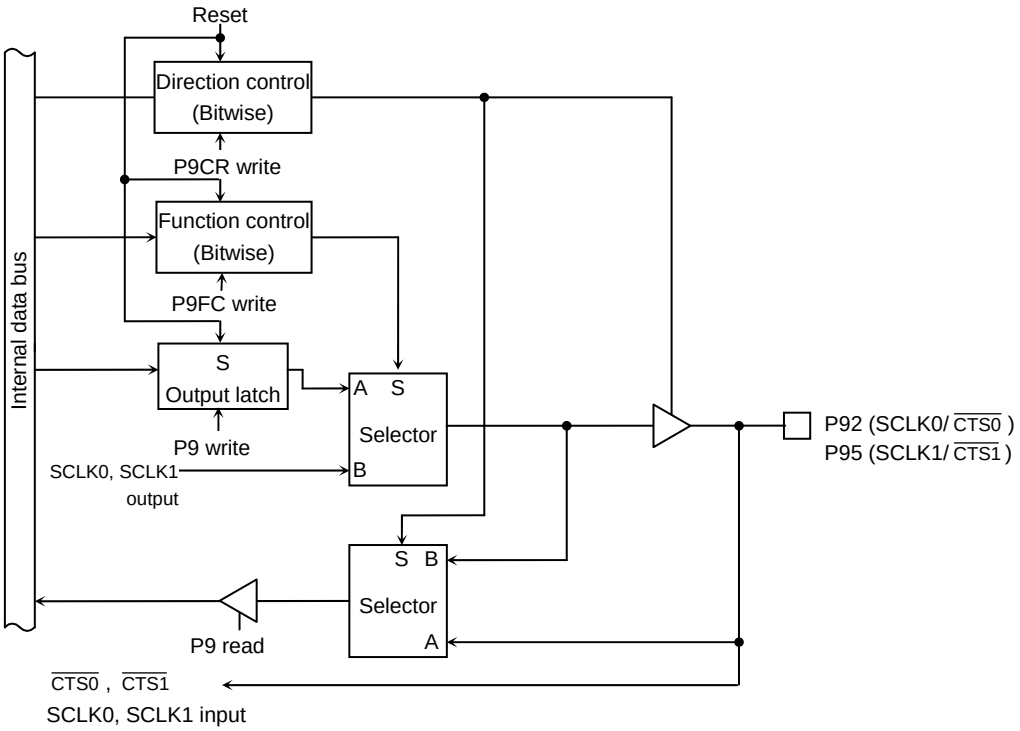


Figure 3.5.13 Port 9 (P92, P95)

Port 9 Register

P9 (0019H)

	7	6	5	4	3	2	1	0
Bit symbol			P95	P94	P93	P92	P91	P90
Read/Write			R/W					
After reset			Data from external port (Output latch register is set to 1.)					

Port 9 Control Register

P9CR (001CH)

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

Port 9 input/output setting

0	Input
1	Output

Port 9 Function Register

P9FC (001DH)

	7	6	5	4	3	2	1	0
Bit symbol			P95F		P93F	P92F		P90F
Read/Write			W		W	W		W
After reset			0		0	0		0
Function			0: Port 1: SCLK1 output		0: Port 1: TXD1	0: Port 1: SCLK0 output		0: Port 1: TXD0

P90 TXD0 output setting

P9FC<P90F>	1
P9CR<P90C>	1

P92 SCLK0 output setting

P9FC<P92F>	1
P9CR<P92C>	1

P93 TXD1 output setting

P9FC<P93F>	1
P9CR<P93C>	1

P95 SCLK1 output setting

P9FC<P95F>	1
P9CR<P95C>	1

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

Note 2: To specify the TXD pin as an open-drain output, write 1 to bit 4 (for the TXD0 pin) or bit 5 (for the TXD1 pin) of the ODE register. The P91/RXD0 and P94/RXD1 pins do not have a register bit for selecting the port or SIO function. The input to these pins is always directed to the SIO as serial receive data even when they are used as general-purpose input pins.

Open-Drain Register

ODE (002FH)

	7	6	5	4	3	2	1	0
Bit symbol	ODEA3	ODEA0	ODE93	ODE90	ODE83	ODE82	ODE81	ODE80
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	PA3 open-drain output 0: Disable 1: Enable	PA0 open-drain output 0: Disable 1: Enable	P93 open-drain output 0: Disable 1: Enable	P90 open-drain output 0: Disable 1: Enable	P83 open-drain output 0: Disable 1: Enable	P82 open-drain output 0: Disable 1: Enable	P81 open-drain output 0: Disable 1: Enable	P80 open-drain output 0: Disable 1: Enable

Open-drain output setting

0	Disable
1	Enable

Figure 3.5.14 Port 9 Registers

3.5.6 Port A (PA0 to PA5)

Port A is a 6-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initialize all pins as input port pins. All bits in the output latch register (PA) are set to 1. In addition to functioning as a general-purpose input/output port, Port A can also function as input/output pins for serial channels 2 and 3. This alternate function can be enabled by writing 1 in respective bits of the Port A function register (PAFC). Upon reset, the PACR and PAFC are all initialized to 0, setting all pins as input port pins.

(1) PA0, PA3 (TXD2, TXD3)

PA0 and PA3 can be used either as general-purpose input/output port pins or TXD output pins for serial channels 2 and 3.

The output buffer is configurable as an open-drain output using the <ODEA0> and <ODEA3> bits of the ODE register.

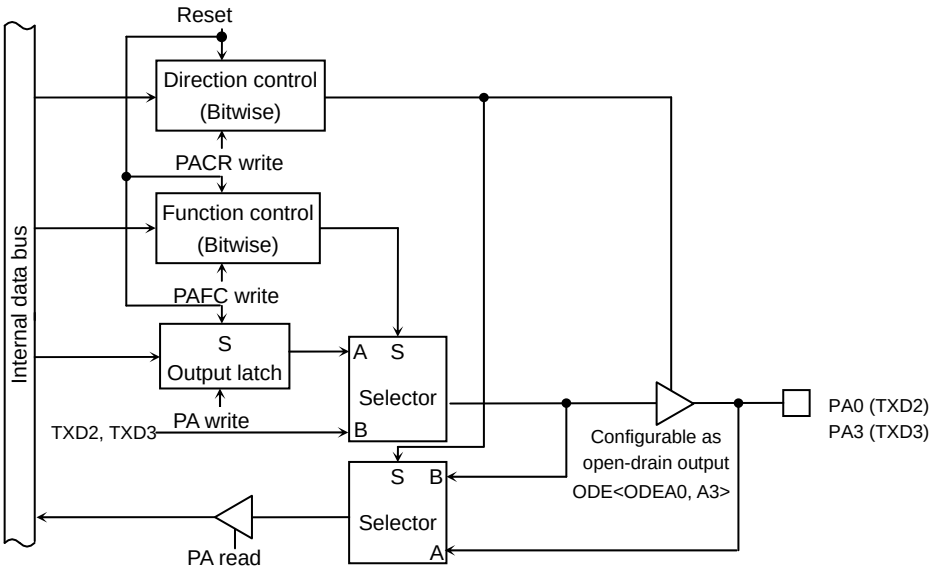


Figure 3.5.15 Port A (PA0, PA3)

(2) PA1, PA4 (RXD2, RXD3)

PA1 and PA4 can be used either as general-purpose input/output port pins or RXD input pins for serial channels 2 and 3.

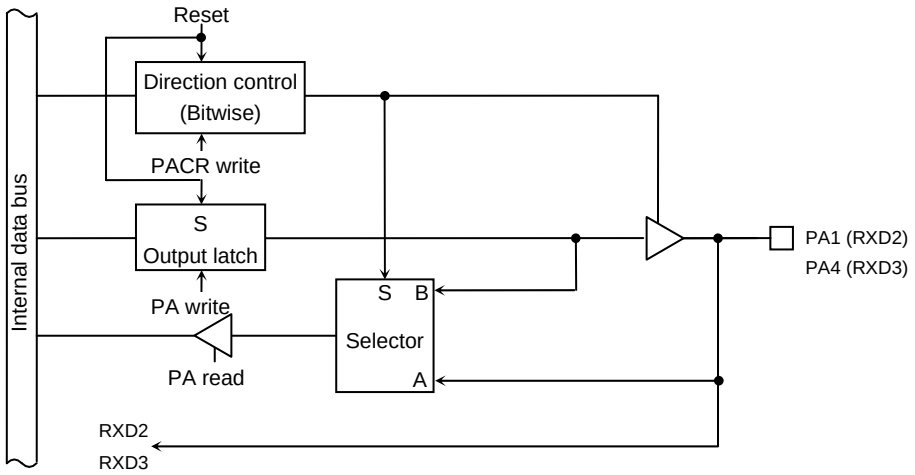


Figure 3.5.16 Port A (PA1, PA4)

(3) PA2, PA5 ($\overline{\text{CTS2}}$/SCLK2, $\overline{\text{CTS3}}$/SCLK3)

PA2 and PA5 can be used either as general-purpose input/output port pins, $\overline{\text{CTS}}$ input pins for serial channels 2 and 3, or SCLK input/output pins.

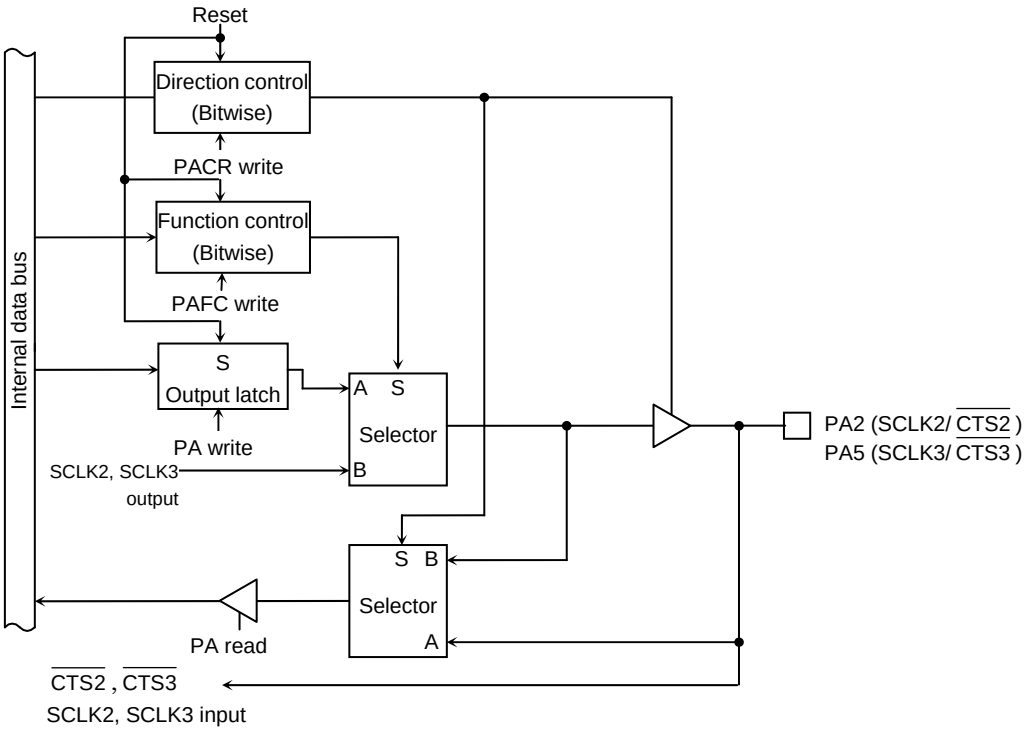


Figure 3.5.17 Port A (PA2, PA5)

Port A Register

PA (001EH)

	7	6	5	4	3	2	1	0
Bit symbol			PA5	PA4	PA3	PA2	PA1	PA0
Read/Write			R/W					
After reset			Data from external port (Output latch register is set to 1.)					

Port A Control Register

PACR (0020H)

	7	6	5	4	3	2	1	0
Bit symbol			PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
Read/Write			W					
After reset			0	0	0	0	0	0
Function			0: Input 1: Output					

Port A input/output setting

0	Input
1	Output

Port A Function Register

PAFC (0021H)

	7	6	5	4	3	2	1	0
Bit symbol			PA5F		PA3F	PA2F		PA0F
Read/Write			W		W	W		W
After reset			0		0	0		0
Function			0: Port 1: SCLK3 output		0: Port 1: TXD3	0: Port 1: SCLK2 output		0: Port 1: TXD2

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

Note 2: To specify the TXD pin as an open-drain output, write 1 to bit 6 (for the TXD2 pin) or bit 7 (for the TXD3 pin). The PA1/RXD2 and PA4/RXD3 pins do not have a register bit for selecting the port or SIO function. The input to these pins is always directed to the SIO as serial receive data even when the pins are used as general-purpose input pins.

PA0 TXD2 output setting

PAFC<PA0F>	1
PACR<PA0C>	1

PA2 SCLK2 output setting

PAFC<PA2F>	1
PACR<PA2C>	1

PA3 TXD3 output setting

PAFC<PA3F>	1
PACR<PA3C>	1

PA5 SCLK3 output setting

PAFC<PA5F>	1
PACR<PA5C>	1

Open-Drain Register

ODE (002FH)

	7	6	5	4	3	2	1	0
Bit symbol	ODEA3	ODEA0	ODE93	ODE90	ODE83	ODE82	ODE81	ODE80
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	PA3 open-drain output 0: Disable 1: Enable	PA0 open-drain output 0: Disable 1: Enable	P93 open-drain output 0: Disable 1: Enable	P90 open-drain output 0: Disable 1: Enable	P83 open-drain output 0: Disable 1: Enable	P82 open-drain output 0: Disable 1: Enable	P81 open-drain output 0: Disable 1: Enable	P80 open-drain output 0: Disable 1: Enable

Open-drain output setting

0	Disable
1	Enable

Figure 3.5.18 Port A Registers

3.5.7 Port 2 (P20 to P27)

Port 2 is an 8-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins as input port pins. All bits of the output latch register (P2) are set to 1. In addition to functioning as a general-purpose input/output port, Port 2 can also function as LCD segment output pins. This alternate function can be enabled by writing 1 to respective bits of the LCD output control 1 register (LCDSW1). Upon reset, the P2CR and LCDSW1 registers are all initialized to 0, setting all pins as input port pins.

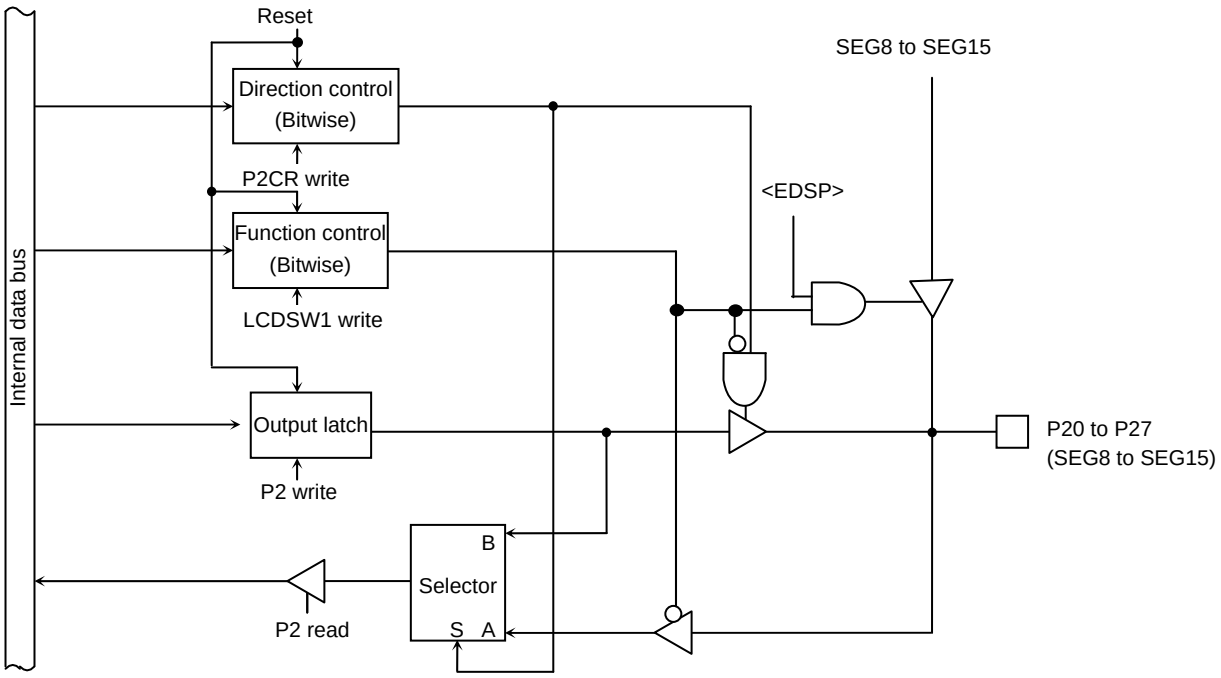


Figure 3.5.19 Port 2 (P20 to P27)

Port 2 Register

P2 (0006H)

	7	6	5	4	3	2	1	0
Bit symbol	P27	P26	P25	P24	P23	P22	P21	P20
Read/Write	R/W							
After reset	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							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 2 input/output setting

0	Input
1	Output

(LCDSW1<SEGnC>=0)

LCD Output Control 1 Register

LCDSW1 (03D9H)

	7	6	5	4	3	2	1	0
Bit symbol	SEG15C	SEG14C	SEG13C	SEG12C	SEG11C	SEG10C	SEG9C	SEG8C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG15	0: Port 1: SEG14	0: Port 1: SEG13	0: Port 1: SEG12	0: Port 1: SEG11	0: Port 1: SEG10	0: Port 1: SEG9	0: Port 1: SEG8

Note: The P2CR and LCDSW1 do not support read-modify-write operation.

SEGn output setting example

LCDSW1<SEGnC>	1

Figure 3.5.20 Port 2 Registers

Note: The LCD output control register is also provided for SEG0 to SEG7 which do not support the port function.

LCD Output Control 0 Register

LCDSW0 (03D8H)

	7	6	5	4	3	2	1	0
Bit symbol	SEG7C	SEG6C	SEG5C	SEG4C	SEG3C	SEG2C	SEG1C	SEG0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Hi-z 1: SEG7	0: Hi-z 1: SEG6	0: Hi-z 1: SEG5	0: Hi-z 1: SEG4	0: Hi-z 1: SEG3	0: Hi-z 1: SEG2	0: Hi-z 1: SEG1	0: Hi-z 1: SEG0

SEGn output setting example

LCDSW0<SEGnC>	1

Note: The LCDSW0 do not support read-modify-write operation.

3.5.8 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins to input port pins. All bits of the output latch register (P1) are set to 0. In addition to functioning as a general-purpose input/output port, Port 1 can also function as LCD segment output pins. This alternate function can be enabled by writing 1 to respective bits in the LCD output control 2 register. Upon reset, the P1CR and LCDSW2 are all initialized to 0, setting all pins as input port pins.

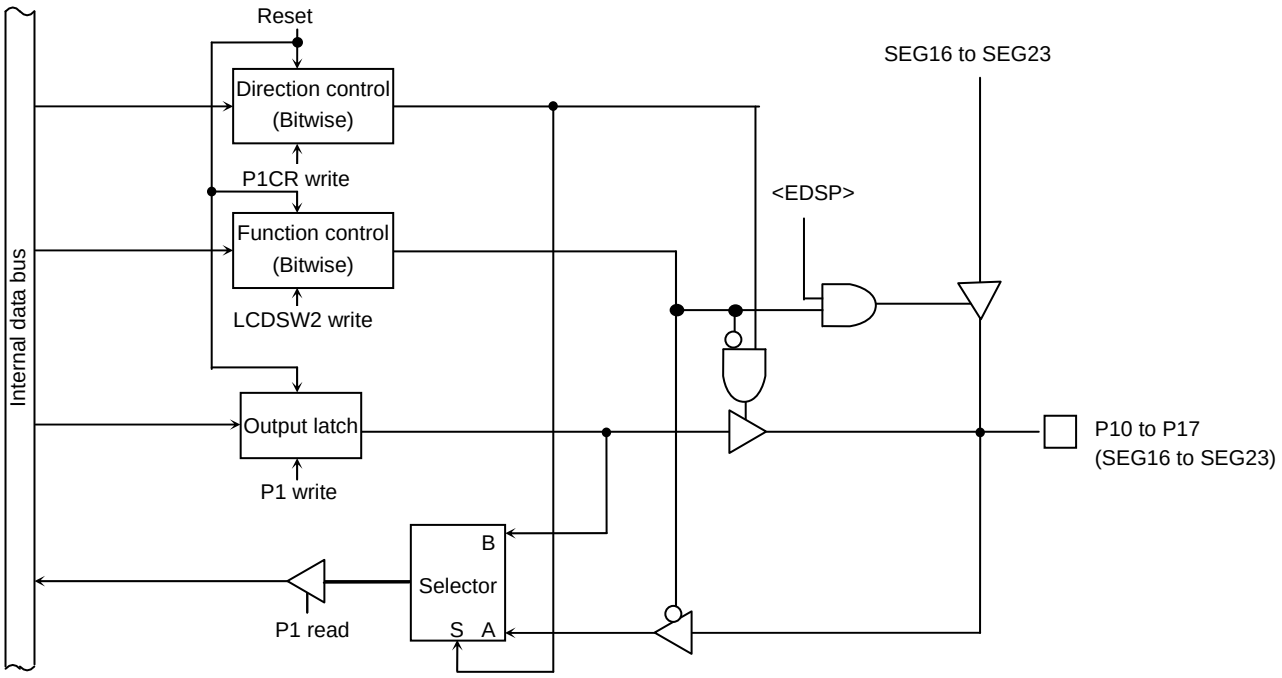


Figure 3.5.21 Port 1 (P10 to P17)

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							
After reset	Data from external port (Output latch register is set 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							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 1 input/output setting

0	Input
1	Output

LDCSW2<SECnC>=0

LCD Output Control 2 Register

LCDSW2 (03DAH)

	7	6	5	4	3	2	1	0
Bit symbol	SEG23C	SEG22C	SEG21C	SEG20C	SEG19C	SEG18C	SEG17C	SEG16C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG23	0: Port 1: SEG22	0: Port 1: SEG21	0: Port 1: SEG20	0: Port 1: SEG19	0: Port 1: SEG18	0: Port 1: SEG17	0: Port 1: SEG16

Note: The P1CR and LDCSW2 do not support read-modify-write operation.

SEGn output setting example

LCDSW2<SEGnC>	1

Figure 3.5.22 Port 1 Registers

3.5.9 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins as input port pins. All bits of the output latch register (P0) are set to 0. In addition to functioning as a general-purpose input/output port, Port 0 can also function as LCD segment output pins. This alternate function can be enabled by writing 1 to respective bits of the LCD output control 3 register. Upon reset, the P0CR and LCDSW3 registers are all initialized to 0, setting all pins as input port pins.

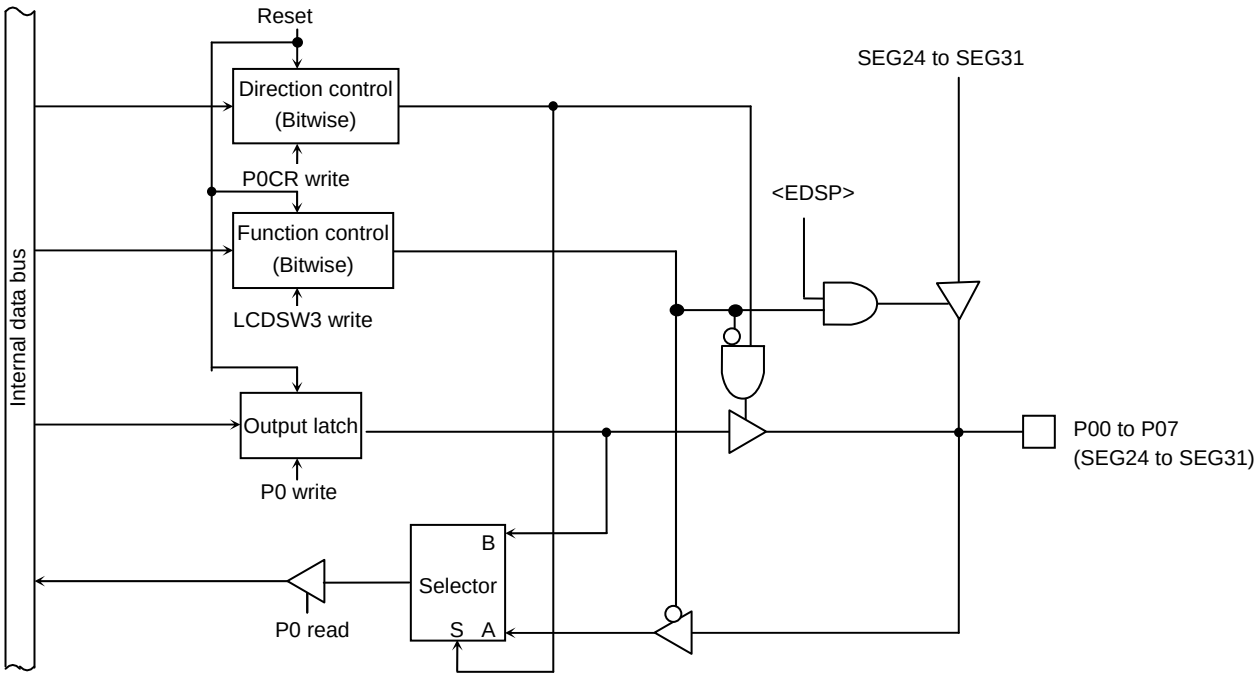


Figure 3.5.23 Port 0 (P00 to P07)

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							
After reset	Data from external pins (Output latch register is set to 0.)							

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							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 0 input/output setting

0	Input
1	Output

LCDSW3<SEGnC>=0

LCD Output Control 3 Register

LCDSW3 (03DBH)

	7	6	5	4	3	2	1	0
Bit symbol	SEG31C	SEG30C	SEG29C	SEG28C	SEG27C	SEG26C	SEG25C	SEG24C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG31	0: Port 1: SEG30	0: Port 1: SEG29	0: Port 1: SEG28	0: Port 1: SEG27	0: Port 1: SEG26	0: Port 1: SEG25	0: Port 1: SEG24

Note: The P0CR and LCDSW3 do not support read-modify-write operation.

SEGn output setting example

LCDSW3<SEGnC>	1

Figure 3.5.24 Port 0 Registers

3.5.10 Port B (PB0 to PB7)

Port B is an 8-bit general-purpose input/output port. Each bit can be individually programmed for input or output. Reset operation initializes all pins as input port pins. All bits of the output latch register (PB) are set to 0. In addition to functioning as a general-purpose input/output port, Port B can also function as LCD segment output pins. This alternate function can be enabled by writing 1 to respective bits of the LCD output control 4 register (LCDSW4). Upon reset, the PBCR and LCDSW4 registers are all initialized to 0, setting all pins as input port pins.

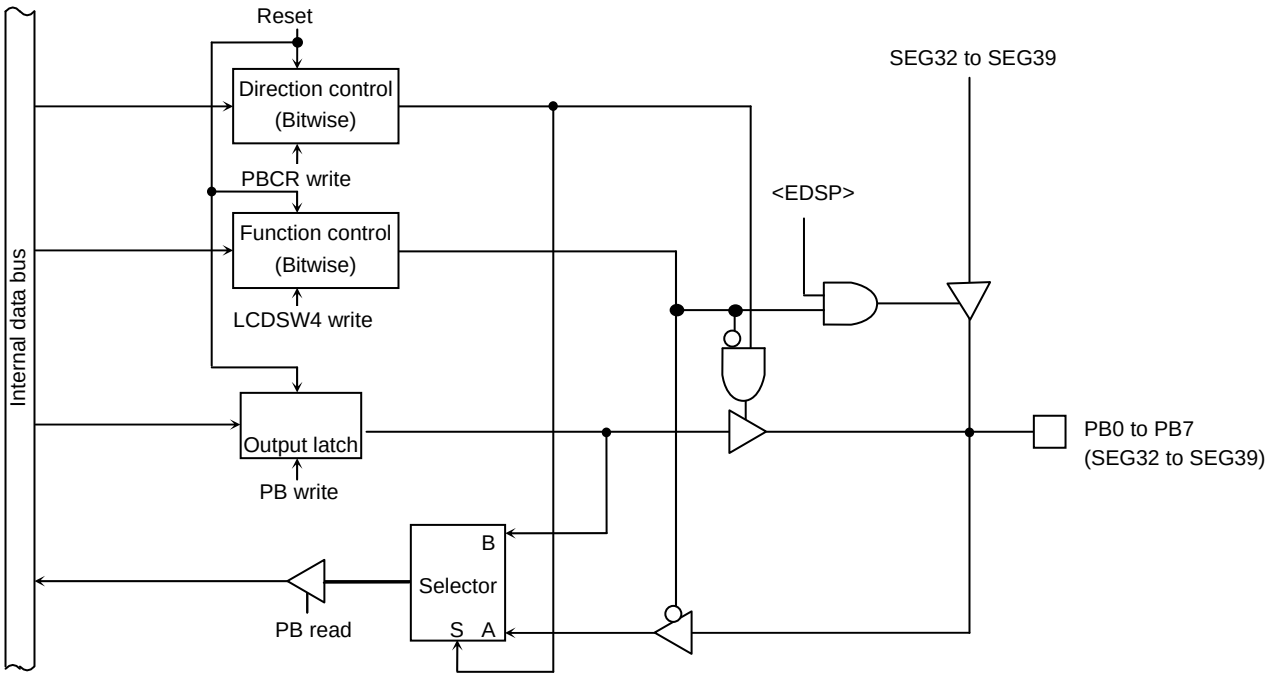


Figure 3.5.25 Port B (PB0 to PB7)

Port B Register

PB (0024H)

	7	6	5	4	3	2	1	0
Bit symbol	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Read/Write	R/W							
After reset	Data from external port (Output latch is set to 0.)							

Port B Control Register

PBCR (0025H)

	7	6	5	4	3	2	1	0
Bit symbol	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

→ Port B input/output setting

0	Input
1	Output

LCDSW4<SEGnC>=0

LCD Output Control 4 Register

LCDSW4 (03DCH)

	7	6	5	4	3	2	1	0
Bit symbol	SEG39C	SEG38C	SEG37C	SEG36C	SEG35C	SEG34C	SEG33C	SEG32C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG39	0: Port 1: SEG38	0: Port 1: SEG37	0: Port 1: SEG36	0: Port 1: SEG35	0: Port 1: SEG34	0: Port 1: SEG33	0: Port 1: SEG32

Note: The PBCR and LCDSW4 do not support read-modify-write operation.

→ SEGn output setting example

LCDSW4<SEGnC>	1

Figure 3.5.26 Port B Registers

3.6 Timing Generator

The timing generator generates various system clocks to be supplied to peripheral hardware based on the basic clock (fc or fs).

(1) Configuration

The timing generator consists of two counters, one for the high-frequency clock and one for the low-frequency clock.

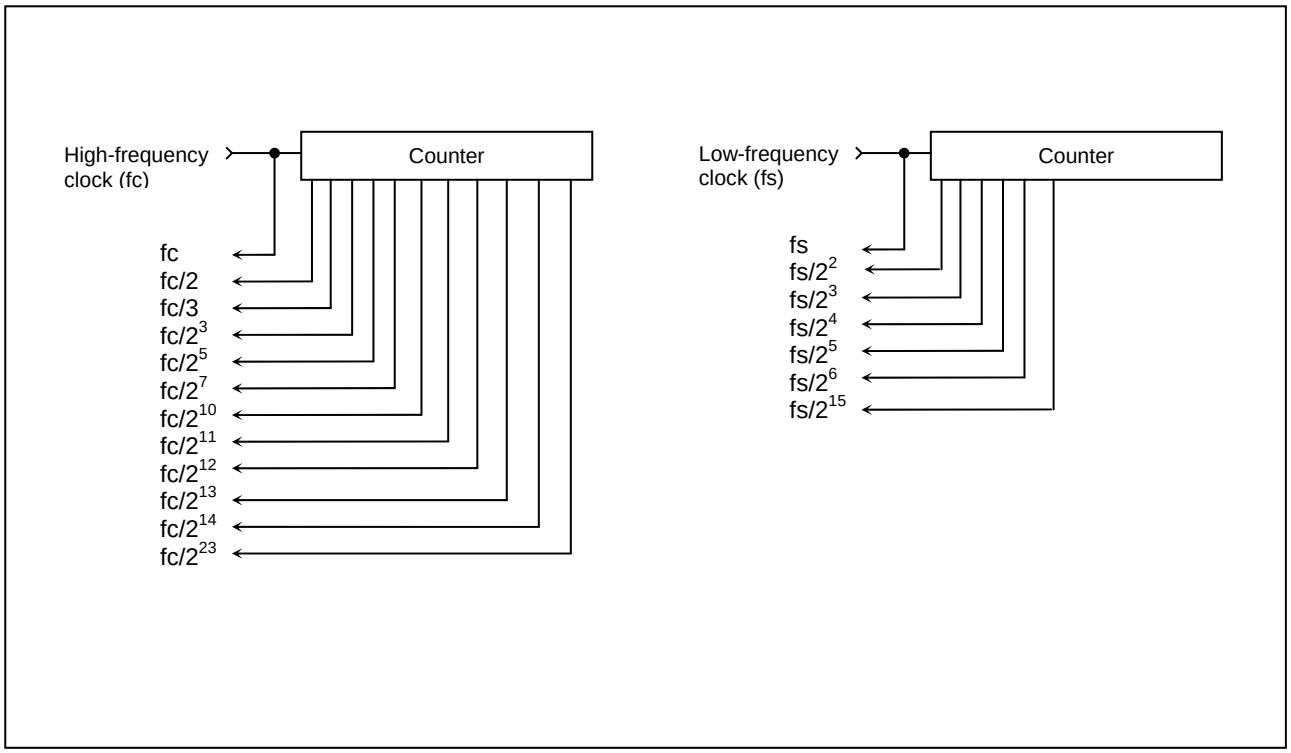


Figure 3.6.1 Configuration of the Timing Generator

3.7 Divider Output ($\overline{DVO}$)

The timing generator is provided with a divider output feature which enables output of approximately 50% duty pulses. This feature is useful for driving a piezoelectric beeper. Divider output is implemented on the P72 ($\overline{DVO}$) pin.

Note: The divider output frequency (<DVOCK>, <DVSEL>) must be specified and the timing generator operating status (<FCDIS>, <FSDIS>) must be changed while divider output is disabled (<DVOEN>=0).
Also note that the peripheral circuits using the timing generator (8-bit/16-bit timers) must also be stopped before changing the <FCDIS> and <FSDIS> bits.

TBTCR (0340_H)

7	6	5	4	3	2	1	0
DVOEN	DVOCK		DVSEL	—	—	FSDIS	FCDIS

(Initial value:0000 **00)

DVOEN	Divider output enable/disable	0: Disable output 1: Enable output			R/W
DVOCK	Divider output ($\overline{DVO}$ pin) frequency [Hz]		DVSEL = 0	DVSEL = 1	
		00	$fc/2^{13}$	$fs/2^5$	
		01	$fc/2^{12}$	$fs/2^4$	
		10	$fc/2^{11}$	$fs/2^3$	
		11	$fc/2^{10}$	$fs/2^2$	
FSDIS	Timing generator control for low-frequency clock (fs)	0: Operate 1: Stop			W
FCDIS	Timing generator control for high-frequency control (fc)	0: Operate 1: Stop			

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Undefined value

Note 2: <DVSEL>=0 must not be set in SLOW and SLEEP modes.

Note 3: The TBTCR does not support read-modify-write operation.

Figure 3.7.1 Divider Output Control Register

Table 3.7.1 Divider Output Frequencies (at fc = 27.0 MHz, fs = 32.768 kHz)

DVOCK	Divider Output Frequency [Hz]	
	DVSEL = 0	DVSEL = 1
00	3.296 k	1.024 k
01	6.592 k	2.048 k
10	13.184 k	4.096 k
11	26.367 k	8.192 k

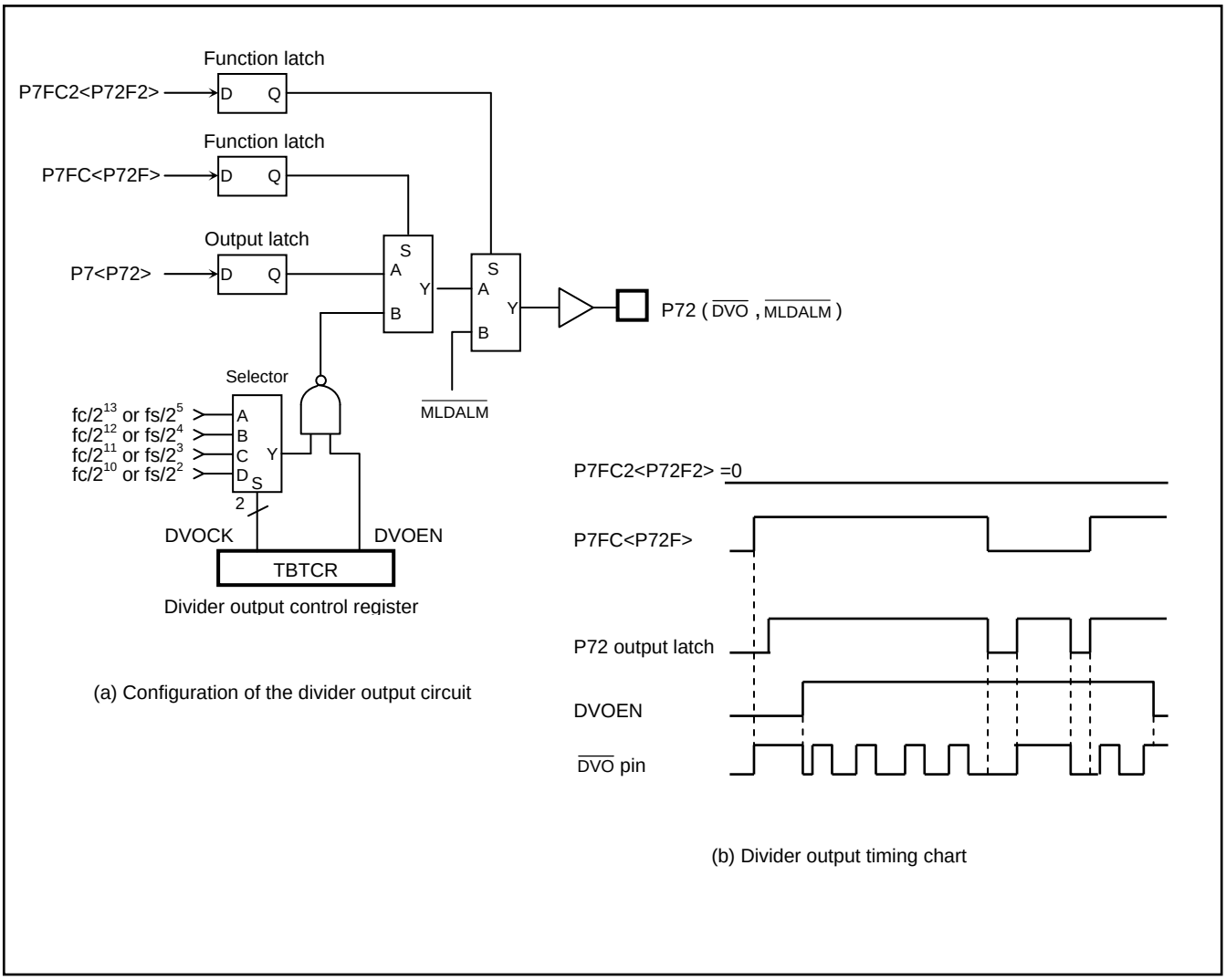


Figure 3.7.2 Divider Output

3.8 16-Bit Timer/Counter

The TMP91CW40 has three channels of 16-bit timers (TC1, TC2 and TC3). Each of the three channels operates independently, and is functionally equivalent. In the following sections, any references to TC1 also apply to other channels.

3.8.1 Configuration

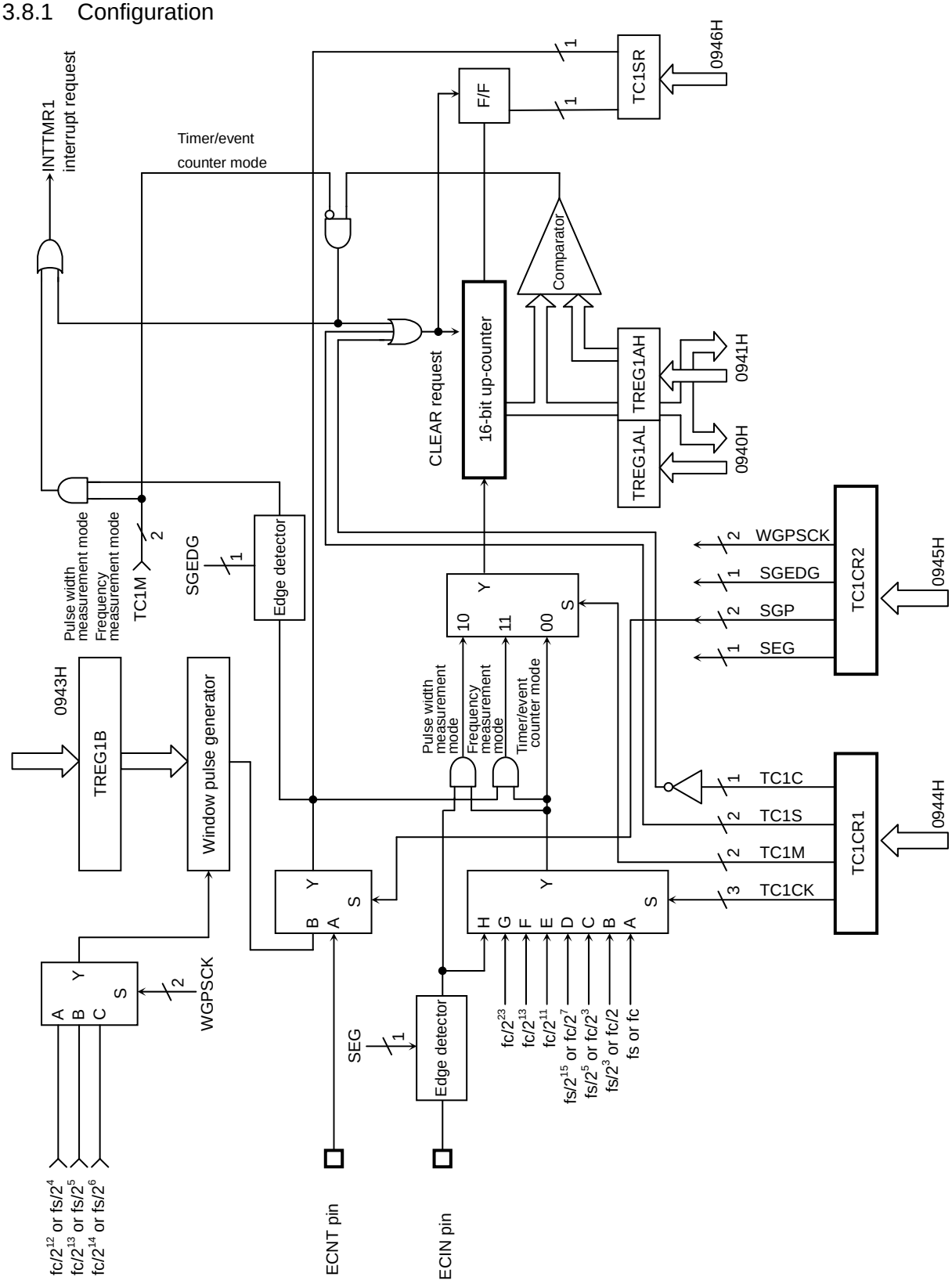


Figure 3.8.1 Timer/Counter 1 (TC1)

3.8.2 Control

The timer/counter 1 (TC1) is controlled by the timer/counter 1 control registers (TC1CR1/TC1CR2), timer register (TREG1A) and internal window gate pulse setting register (TREG1B).

Timer Register

TREG1A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(0941H, 0940H)	TREG1AH (0941H)								TREG1AL (0940H)							

Read/Write (Initial value: 0000 0000 0000 0000)

Internal Window Gate Pulse Setting Register

TREG1B	7	6	5	4	3	2	1	0
(0943H)	Ta				Tb			

(Initial value: 0000 0000)

Ta	Window gate pulse "H" level period	$(16 - Ta) \times 2^{13}$/fc or $(16 - Ta) \times 2^{5}$/fs [s]	R/W
Tb	Window gate pulse "L" level period	$(16 - Tb) \times 2^{13}$/fc or $(16 - Tb) \times 2^{5}$/fs [s]	

Note: WGPSCK = 01

Timer/Counter 1 Control Register 1

TC1CR1	7	6	5	4	3	2	1	0
(0944H)	TC1C	TC1S			TC1CK		TC1M	

(Initial value: 1000 0000)

TC1C		0: Clear counter and overflow flag (Automatically set to 1 after clearing)				R/W
TC1S	TC1 start control	00: Stop and clear counter (and overflow flag) 10: Start TC1 *1: Reserved				
TC1CK	TC1 source clock select		TC1SEL=0	TC1SEL=1		
		000	fc (Note 4)	fs (Note 4)	[Hz]	
		001	fc/2 (Note 4)	$fs/2^{3}$		
		010	$fc/2^{3}$	$fs/2^{5}$		
		011	$fc/2^{7}$	$fs/2^{15}$		
		100	$fc/2^{11}$	–		
		101	$fc/2^{13}$	–		
		110	$fc/2^{23}$	–		
		111	External clock (ECIN1 pin input)			
TC1M	TC1 operating mode select	00: Timer/event counter mode 01: Reserved 10: Pulse width measurement mode 11: Frequency measurement mode				

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 2: After the lower byte of the timer register (TREG1AL) is written, match detection is temporarily disabled until a write to the upper byte (TREG1AH) is completed. (It is not possible to modify only the lower byte of the timer register.) Also note that match detection is enabled again after one machine cycle has elapsed upon completion of a write to the upper byte.

Note 3: Before changing the operating mode, source clock and edge selection, be sure to stop the timer/counter (<TC1S>=00). TC1CR1<TC1S> and TC1CR1<TC1C> must not be changed simultaneously.

Note 4: The source clock can be set to fc, fc/2 and fs only in the pulse width measurement mode (in NORMAL or IDLE2 mode).

Note 5: When a read instruction is executed on the timer register (TREG1A), the counter immediate value, not the value written to the TREG1A is read out. Before reading the TREG1A, therefore, make sure that the timer/counter is stopped; otherwise an undefined value may be read out.

Note 6: The timer register value should be TREG1A ≥ 1.

Note 7: In the timer or pulse width measurement mode, select an internal clock as the source clock (TC1CR1<TC1CK>).

Note 8: In the event counter mode, select an external clock as the source clock (TC1CR1<TC1CK>).

Note 9: Since the timer register (TREG1A) has different write and read values, read-modify-write instructions must not be used on this register.

Note 10: In NORMAL mode, changing TC1CR1 and TREG1A while selecting fs (TC1CR2<TC1SEL>="1") is prohibited.

Note 11: In SLOW mode, do not set TC1CR2<TC1SEL>=0.

Note 12: "–"indicates a setting that must not be selected.

Figure 3.8.2 TC1 Timer Register/Window Gate Pulse Setting Register/Control Register

Timer/Counter 1 Control Register 2

TC1CR2 (0945H)

7	6	5	4	3	2	1	0
SEG	SGP		SGEDG	WGPSCK		—	TC1SEL

(Initial value: 0000 00*0)

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 2: Before setting the TC1CR2, be sure to stop the timer/counter (<TC1S> = 00).

Timer/Counter 1 Status Register

SEG	External input clock (ECIN1) edge select	0: Count at the rising edge 1: Count at the rising and falling edges			R/W
SGP	Window gate pulse select	00: ECNT1 input 01: Internal window gate pulse (TREG1B) 10: Reserved 11: Reserved			
SGEDG	Window gate pulse interrupt edge select	0: Interrupt at the falling edge 1: Interrupt at the falling and rising edges			
WGPSCK	Window gate pulse source clock select		TC1SEL=0	TC1SEL=1	
		00:	$fc/2^{12}$	$fs/2^{4}$	
		01:	$fc/2^{13}$	$fs/2^{5}$	
		10:	$fc/2^{14}$	$fs/2^{6}$	
		11:	Reserved		

TC1SR (0946H)

7	6	5	4	3	2	1	0
HECF	HEOVF	—	—	—	—	—	—

(Initial value: 00** ****)

HECF	Operating status monitor	0: Stopped (Tb period) or disabled 1: Counting (Ta period)	Read only
HEOVF	Counter overflow monitor	0: Overflow not occurred 1: Overflow occurred	

Note: <HECF> can be used only in the frequency measurement mode.

Figure 3.8.3 TC1 Control Register/Status Register

TC1		TC2		TC3	
TREG1AL	0940H	TREG2AL	0950H	TREG3AL	0960H
TREG1AH	0941H	TREG2AH	0951H	TREG3AH	0961H
TREG1B	0943H	TREG2B	0953H	TREG3B	0963H
TC1CR1	0944H	TC2CR1	0954H	TC3CR1	0964H
TC1CR2	0945H	TC2CR2	0955H	TC3CR2	0965H
TC1SR	0946H	TC2SR	0956H	TC3SR	0966H

Note: Do not access locations where no registers exist in the 0940H to 096FH area.

Source Clocks That Can Be Used in Each Operating Mode

(NORMAL or IDLE2 mode)

Operating mode	$fc/2^{23}$	$Fc/2^{13}$	$fc/2^{11}$ or $fs/2^{15}$	$fc/2^{7}$ or $fs/2^{5}$	$fc/2^{3}$ or $fs/2^{3}$	fc/2	fc or fs	ECIN
Timer mode	Yes	Yes	Yes	Yes	Yes	No	No	–
Event counter mode	No	No	No	No	No	No	No	TC1CR2<SEG>=0 $fc/2^{4}$ or $fs/2^{4}$ (max) TC1CR2<SEG>=1 $fs/2^{5}$ or $fs/2^{5}$ (max)
Pulse width measurement mode	Yes	Yes	Yes	Yes	Yes	Yes	Yes	–
Frequency measurement mode	No	No	No	No	No	No	No	fc=16 MHz or slower ECIN1=fc/2 (max) fc=16 MHz or faster ECIN1=8 MHz (max)

(Slow or IDLE2 mode)

Operating mode	$fs/2^{15}$	$fs/2^{5}$	$fs/2^{3}$	fs	ECIN
Timer mode	Yes	Yes	Yes	No	–
Event counter mode	No	No	No	No	TC1CR2<SEG>=0 $fs/2^{4}$ max) TC1CR2<SEG>=1 $fs/2^{5}$ (max)
Pulse width measurement mode	Yes	Yes	Yes	Yes	–
Frequency measurement mode	No	No	No	No	ECIN1=fs/2 (max)

3.8.3 Functional Description

The timer/counter 1 has the following four operating modes:

(1) Timer mode

In the timer mode, the counter counts up on the rising edge of the internal clock. When a match between the counter value and the TREG1A register value is detected, an INTTMR1 interrupt is generated and the counter is cleared. The counter continues counting up after it has been cleared.

Table 3.8.1 Timer/Counter 1 Source Clock (Internal Clock)

Source Clock		Resolution		Maximum Setting Time	
TC1SEL = 0	TC1SEL = 1	fc = 27 MHz	fs = 32.768 kHz	fc = 27 MHz	fs = 32.768 kHz
$fc/2^{23}$ [Hz]	$fs/2^{15}$ [Hz]	0.31 s	1s	5.66 h	18.2 h
$fc/2^{13}$	$fs/2^{5}$	303.41 μs	0.98 ms	19.88 s	1.07 min
$fc/2^{11}$	$fs/2^{3}$	75.85 μs	244 μs	4.97 s	16 s
$fc/2^{7}$	–	4.74 μs	–	310.69 ms	–
$fc/2^{3}$	–	0.3 μs	–	19.42 ms	–

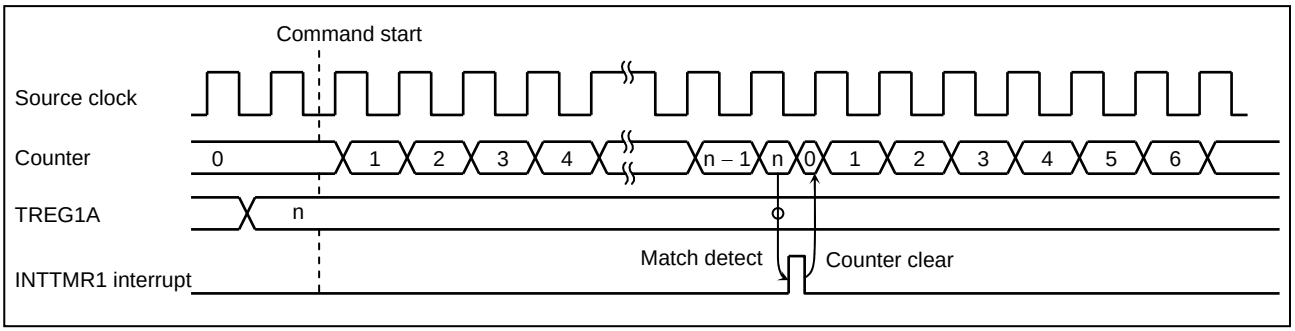


Figure 3.8.4 Timer Mode Timing Chart

Programming sequences (Be sure to follow these sequences.)

- Setting the timer mode with the system clock fc and the counter source clock fc/2^3

```
LD    (TC1CR2),00H     : Set the <TC1SEL> bit. (<TC1SEL> = 0)
LD    (TC1CR1),80H     : Select the timer mode.
LDW   (TREG1A),0100H   : Set the timer register. (TREG1AH=01H, TREG1AL=00H)
LD    (TC1CR1),88H     : Set the source clock to fc/2^3.
LD    (TC1CR1),0C8H    : Start the timer.
```

- Changing the source clock and the timer register contents (after the timer is started)

```
LD    (TC1CR1),88H     : Stop the timer & clear the counter.
LDW   (TREG1A),0080H   : Set the timer register. (TREG1AH=00H,  TREG1AL=80H)
LD    (TC1CR1),8CH     : Change the source clock from fc/2^3 to fc/2^7.
LD    (TC1CR1),0CCH    : Start the timer.
```

- Changing the source clock to fs/2^3 (after the timer is started)

```
LD    (TC1CR1),8CH     : Stop the timer & clear the counter.
LD    (TC1CR2),00H     : Set <TC1SEL>=0 to select fc (in NORMAL mode only) once.
                           Note: In NORMAL mode, do not change the TC1CR1/TREG1A with fs selected.
LD    (TC1CR1),8CH     : Change the source clock to fs/2^15.
LD    (TC1CR2),01H     : Set <TC1SEL>=1 to select fs.
LD    (TC1CR1),0CCH    : Start the timer.
```

(2) Event counter mode

In the event counter mode, the counter counts up on the rising edge of the ECIN1 pin input. When a match between the counter value and the TREG1A register value is detected, an INTTMR1 interrupt is generated and the counter is cleared. Then, the counter continues counting up on each rising edge of the ECIN1 pin input. The maximum allowed frequency is $fc/2^4$ [Hz] (in NORMAL or IDLE2 mode) and $f/2^4$ [Hz] (in SLOW or SLEEP mode) when TC1CR2<SEG>=0. Both high and low levels require a pulse width of at least two machine cycles.

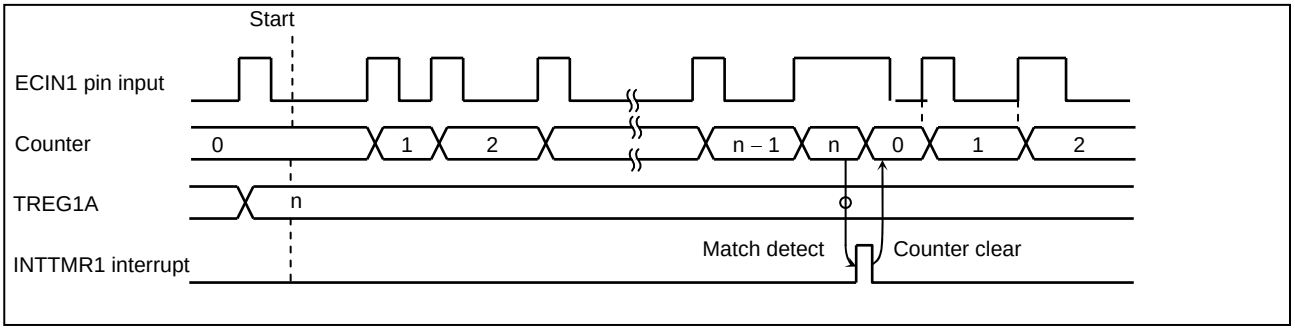


Figure 3.8.5 Event Counter Mode Timing Chart

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD    (TC1CR2),00H      : Set the <TC1SEL> bit. (<TC1SEL>=0)
                          <SEG>=0 to count up on the rising edge of ECIN1
LD    (TC1CR1),80H      : Select the event counter mode.
LDW   (TREG1A),0100H    : Set the timer register. (TREG1AH=01H, TREG1AL=00H)
LD    (TC1CR1),9CH      : Set the source clock to the ECIN1 pin input.
LD    (TC1CR1),0DCH     : Start the timer.
```

- Changing the timer register contents (after the timer is started)

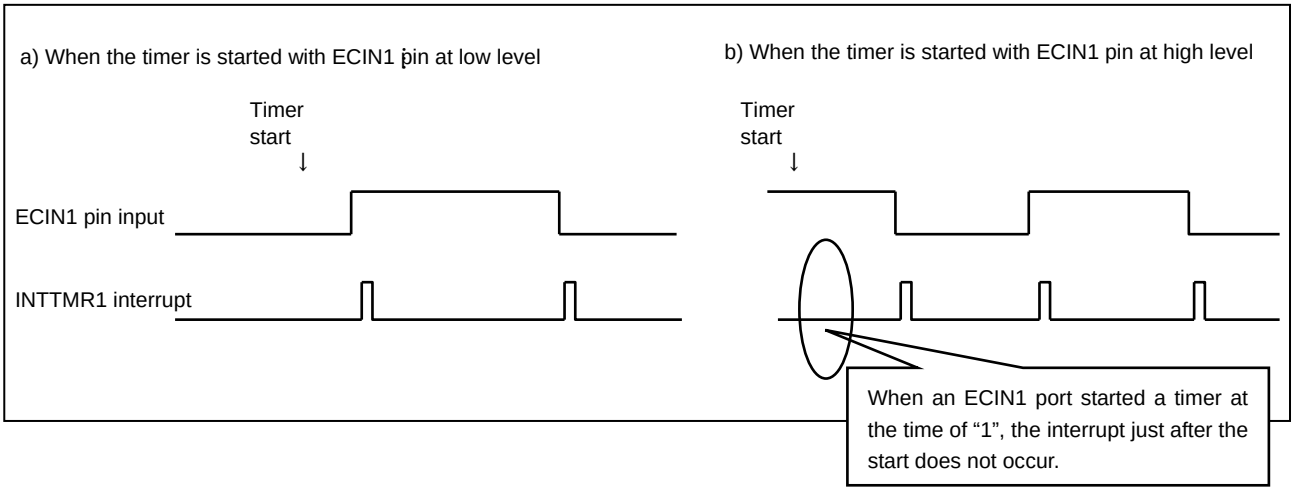
```
LD    (TC1CR1),9CH      : Stop the timer & clear the counter.
LD    (TC1CR1),80H      : Set the source clock to an internal clock once.
LDW   (TREG1A),0080H    : Set the timer register. (TREG1AH=00H, TREG1AL=80H)
LD    (TC1CR1),9CH      : Change the source clock to the ECIN1 pin input.
LD    (TC1CR1),0DCH     : Start the timer.
```

* Before changing the TC1CR2<TC1SEL>, TC1CR1<TC1M> and TREG1A, be sure to once change the source clock to an internal clock.

(3) Pulse width measurement mode

In the pulse width measurement mode, the counter counts up on the rising edge of the AND pulse of the ECIN1 pin input (window pulse) and the internal clock. The internal clock is selected by TC1CR1<TC1CK>. An INTTMR1 interrupt is generated at the falling edge or at both the rising and falling edges of the window pulse, as programmed in TC1CR2<SGEDG>. The counter value (TREG1A) should be read in the interrupt service routine while the counter is stopped (i.e., ECIN1 pin is at low level). Then, the counter should be cleared by using TC1CR<TC1C>. If the counter is not cleared, it resumes counting up from the current value when counting is started again. When the TREG1A counts up from FFFFH to 0000H, an overflow occurs. Whether or not an overflow occurred can be monitored by TC1SR <HEOVF>. The overflow flag state is retained until the counter is cleared.

Note 1: INTTMR1 interrupt generation timing when TC1CR2<SGEDG> = 1 (falling and rising edges) in the pulse width measurement mode



Note 2: In the pulse width measurement mode, the operation status monitor (TC1SR<HECF>) cannot be used.

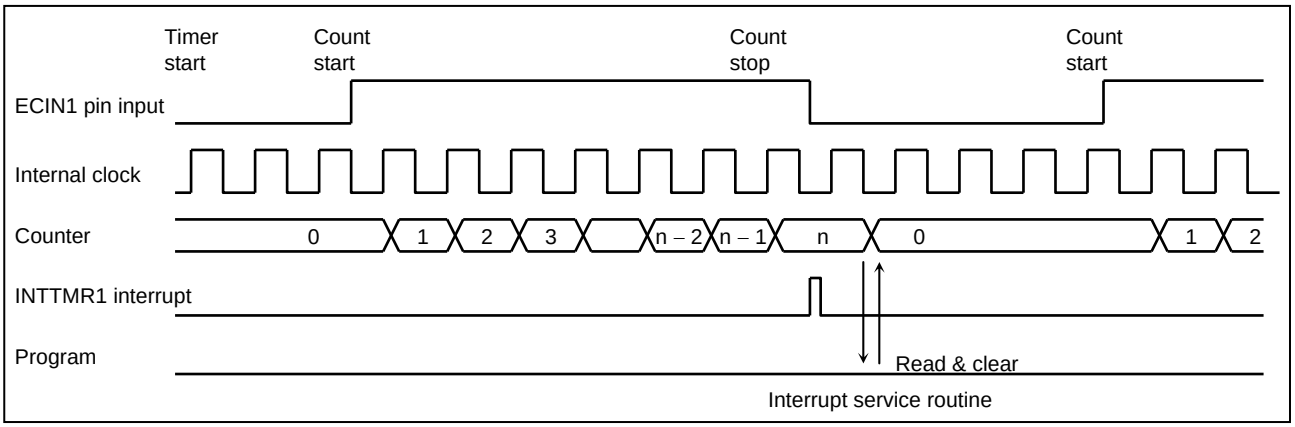


Figure 3.8.6 Pulse Width Measurement Mode Timing Chart

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC1CR2),00H    : Set the <TC1SEL> bit. (<TC1SEL>=0)
LD   (TC1CR1),82H    : Select the pulse width measurement mode.
LD   (TC1CR1),8AH    : Set the source clock to fc/2^3.
LD   (TC1CR1),0CAH   : Start the timer.
```

- Changing the timer register contents (after the timer is started)

```
LD   (TC1CR1),8AH    : Stop the timer & clear the counter.
LD   (TC1CR1),8EH    : Change the source clock to fc/2^7.
LD   (TC1CR1),0CEH   : Start the timer.
```

(4) Frequency measurement mode

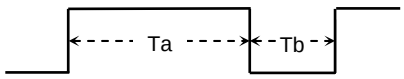
The frequency measurement mode is used to measure the frequency of the ECIN1 pin input pulse. (In this mode, TC1CR1<TC1CK> should be set to external clock.) The counter counts up at the rising edge of the input pulse while the window gate pulse selected by TC1CR2<SGP> is at high level. An INTTMR1 interrupt is generated at the falling edge or at both the rising and falling edges of the window gate pulse, as programmed in TC1CR2<SGEDG>. To use the ECNT1 pin input as the window gate pulse, set TC1CR2<SGP> to 00. The counter value (TREG1A) should be read out in the interrupt service routine while the counter is stopped (i.e., the window gate pulse is at low level). Then, the counter should be cleared by using TC1CR<TC1C>. If the counter is not cleared, it resumes counting up from the current value when counting is started again. The window gate pulse state can be monitored by TC1SR<HECF>. Whether or not an overflow occurred in the binary counter can be monitored by TC1SR<HEOVF>. The overflow flag state is retained until the counter is cleared.

- The internal window gate pulse, when selected, is set as explained below.

  The internal window gate pulse is comprised of a high level period (Ta) in which counting is performed and a low level period (Tb) in which counting is stopped. The Ta and Tb periods can be programmed independently in the TREG1B register. One cycle of the window gate pulse is defined as "Ta + Tb".

Note 1: Since the internal window gate pulse is generated in synchronization with the internal divider, a delay of up to one source clock (WGPSCK) period may occur immediately after the timer is started.

Note 2: The window gate pulse must be programmed while the timer is stopped.



The Ta and Tb periods are represented by the following equations:

When (TC1CR2<WGPSCK> = 10):

$(16 - n) \times 2^{14}/fc$ [s] or

$(16 - n) \times 2^{6}/fs$ [s]

Table 3.8.2 at the right shows the Ta and Tb times when fs = 32.768 kHz.

The Ta period is set by the upper 4 bits (bits 7 to 4) of the TREG1B and the Tb period by the lower 4 bits (bits 3 to 0).

Table 3.8.2 Ta and Tb Times

(TC1CR2<WGPSCK> = 10, fs = 32.768 kHz)

Value n	Time	Value n	Time
0	31.25 ms	8	15.63 ms
1	29.30 ms	9	13.67 ms
2	27.34 ms	A	11.72 ms
3	25.39 ms	B	9.77 ms
4	23.44 ms	C	7.81 ms
5	21.48 ms	D	5.86 ms
6	19.53 ms	E	3.91 ms
7	17.58 ms	F	1.95 ms

Figure 3.8.7 Window Gate Pulse Times

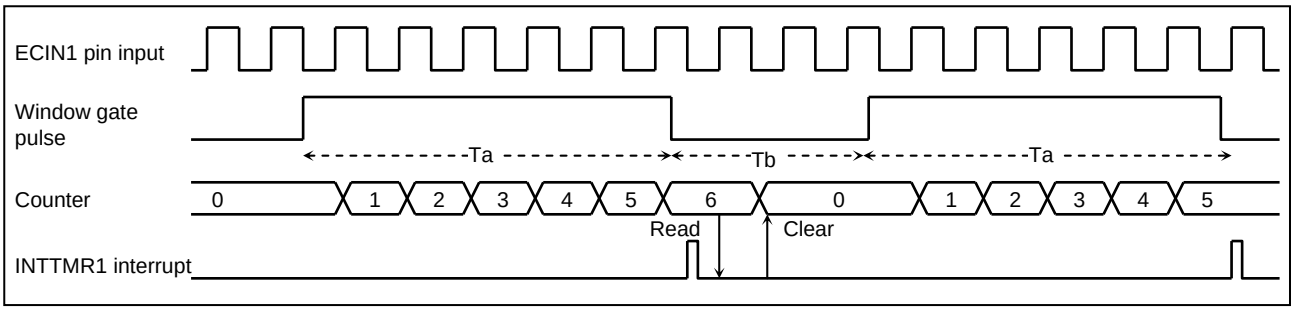


Figure 3.8.8 Frequency Measurement Mode Timing Chart
(Interrupt at the falling edge of the window gate pulse)

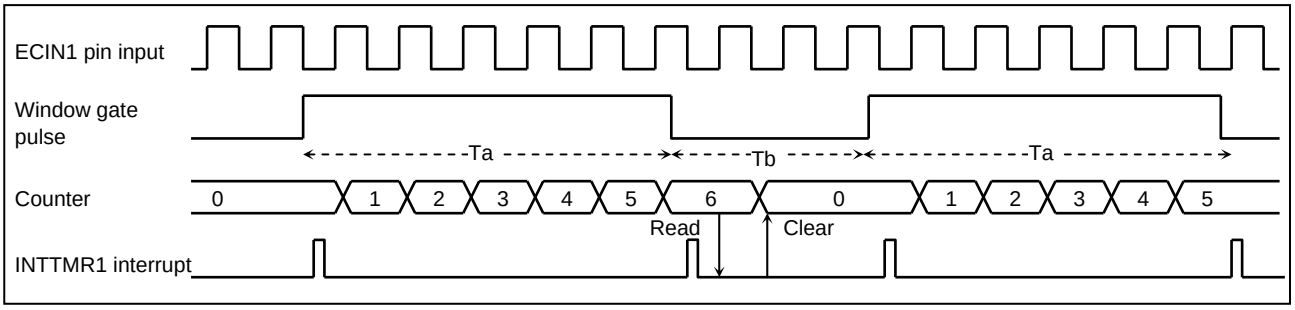


Figure 3.8.9 Frequency Measurement Mode Timing Chart
(Interrupt at the rising/falling edges of the window gate pulse)

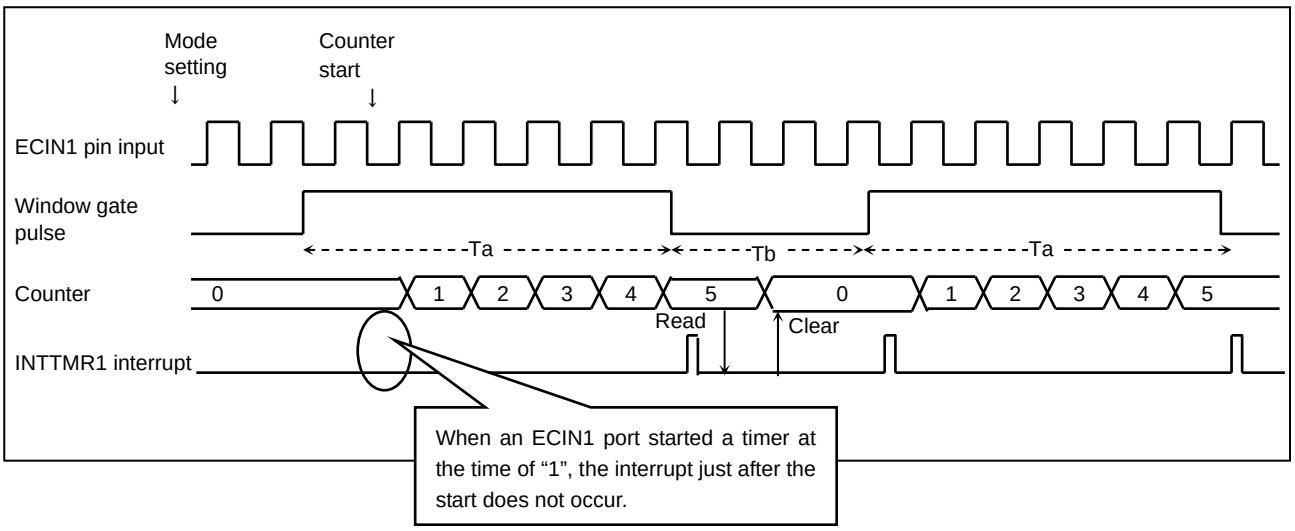


Figure 3.8.10 Frequency Measurement Mode Timing Chart
(Interrupt at the rising/falling edges of the window gate pulse)

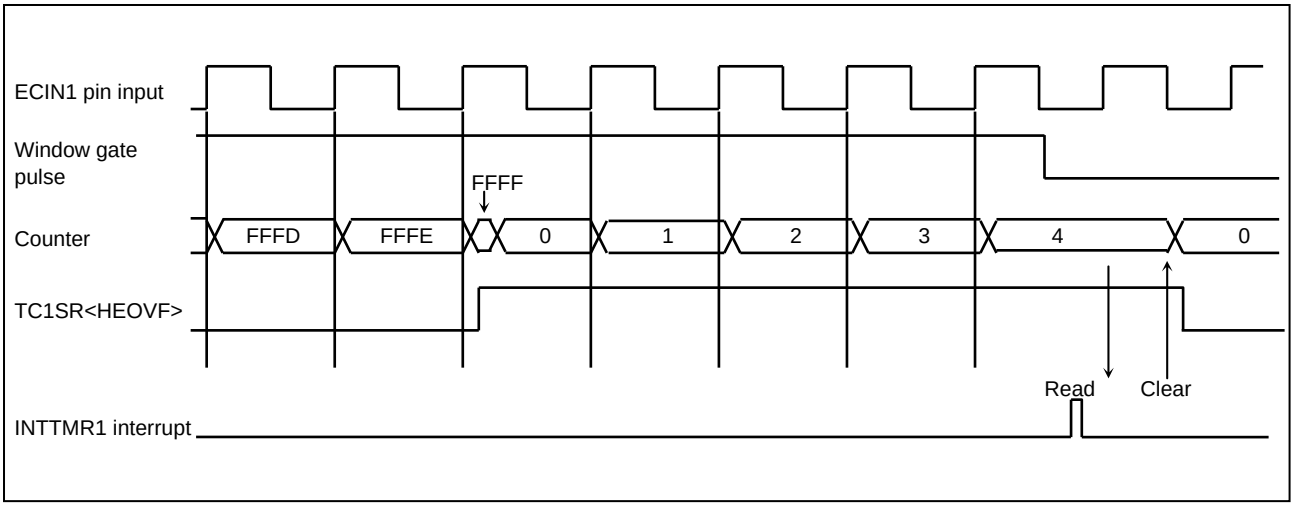


Figure 3.8.11 Frequency Measurement Mode Timing Chart
(Counter overlow)

Programming sequences (Be sure to follow these sequences)

- Setting initial values

```
LD   (TC1CR2),0A8H   : <TC1SEL>=0 to select fc
                       <SEG>=1 (Count on the rising/falling edges of ECIN1)
                       <SGP>=01 (Internal window gate pulse)
                       <SGEDG>=0 (Interrupt at the falling edge of WGP)
                       <WGPSCK>=10 (fc/2^14)
LD   (TC1CR1),83H    : Select the frequency measurement mode.
LD   (TREG1B),11H    : <Ta>=1, <Tb>=1
LD   (TC1CR1),9FH    : Set the source clock to the ECIN1 pin input.
LD   (TC1CR1),0DFH   : Start the timer.
```

- Changing the source clock (after the timer is started)

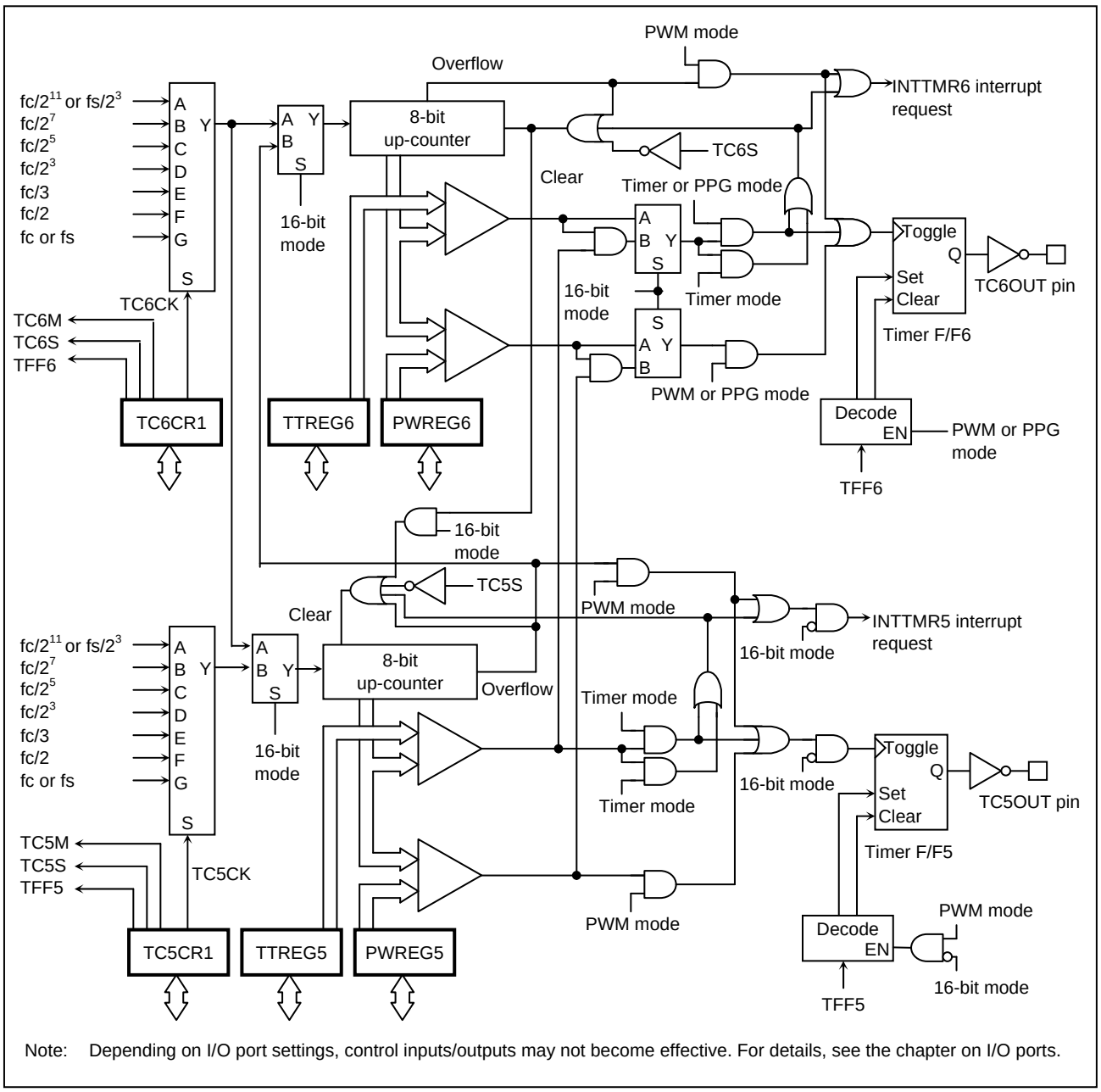
```
LD   (TC1CR1),9FH    : Stop the timer & clear the counter.
LD   (TC1CR1),83H    : Set the source clock to internal clock once.
LD   (TC1CR2),0A9H   : Set <TC1SEL>=1 to change to fs.
LD   (TC1CR1),9FH    : Set the source clock to the ECIN1 pin input.
LD   (TC1CR1),0DFH   : Start the timer.
```

* Before changing the <TC1SEL>, <TC1M> and TREG1A, be sure to once change the source clock to an internal clock.

3.9 8-Bit Timer/Counter

The TMP91CW40 has four channels of 8-bit timers (TC5,TC6、TC7 and TC8). These channels are configured into two modules, each comprising two channels (TC5 and TC6; TC7 and TC8). Each module operates independently, and is functionally equivalent. In the following sections, any references to TC5 and TC6 also apply to TC7 and TC8.

3.9.1 Configuration



Note: Depending on I/O port settings, control inputs/outputs may not become effective. For details, see the chapter on I/O ports.

Figure 3.9.1 8-Bit Timer/Counters 5 & 6

3.9.2 Control

The timer/counter 5 is controlled by the timer/counter 5 control register 1 (TC5CR1), timer/counter 5 control register 2 (TC5CR2) and two 8-bit timer registers (TTREG5 and PWREG5).

Timer Registers

TTREG5 (0904H) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

PWREG5 (0908H) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

Note 1: Do not change the TTREG5 value while the timer is running.

Note 2: In the 8-bit or 16-bit PWM mode, do not change the PWREG5 value while the timer is running.

Note 3: Values that can be set in each timer register are limited depending on the timer's operating mode. For details, see Table 3.9.3.

Timer/Counter 5 Control Register 1

TC5CR1 (0900H)

7	6	5	4	3	2	1	0
TFF5		TC5CK		TC5S		TC5M	

(Initial value: 0000 0000)

TFF5	Timer F/F5 control	0: Clear 1: Set			R/W
TC5CK	Operating clock select [Hz]		TC5SEL=0	TC5SEL= 1	
		000	fc/2^{11}	fs/2^3	
		001	fc/2^7	–	
		010	fc/2^5	–	
		011	fc/2^3	–	
		100	fc/3 (Note 6)	–	
		101	fc/2 (Note 6)	–	
		110	fc (Note 6)	fs (Note 6)	
		111	–		
TC5S	Timer start control	0: Stop & clear counter 1: Start			
TC5M	Operating mode select	000: 8-bit timer mode 001: Reserved 010: 8-bit pulse width modulation (PWM) output mode 011: 16-bit mode (Use TC6CR1<TC6M> to specify which 16-bit mode to use.) 1**: Reserved			

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz]

Note 2: Do not change the <TC5M>, <TC5CK>, <TFF5> and <TC5SEL> settings while the timer is running.

Note 3: Do not change the <TC5M>, <TC5CK> and <TFF5> settings at the same time as stopping the timer (<TC5S> = 1 → 0) or starting the timer (<TC5S> = 0 → 1).

Note 4: When the timer/counter 5 is used in 16-bit mode, the operating mode is selected by TC6CR1<TC6M> and TC5CR1<TC5M> must be set to 011.

Note 5: When the timer/counter 5 is used in 16-bit mode, various control settings are made in the TC6CR1 register. TC5CR1<TC5S> must be set to 0.

Note 6: In selecting operation clock, fc/3, fc/2, fc and fs are selectable only in 8 or 16 bit PWM modes. See Table 3.9.1 and Table 3.9.2 for details.

Timer/Counter 5 Control Register 2

TC5CR2	7	6	5	4	3	2	1	0	
(0902H)	–	–	–	–	–	–	–	TC5SEL	(Initial value: **** ***0)

TC5SEL	Timer input clock control	0: fc 1: fs	R/W

Note 1: Do not set <TC5SEL>=0 in SLOW or SLEEP mode.

Note 2: Do not change TTREG5 when using for fs in NORMAL mode.

Note 3: *: Undefined value

Figure 3.9.2 Timer Registers and Control Registers for Timer/Counter 5

The timer/counter 6 is controlled by the timer/counter 6 control register 1 (TC6CR1), timer/counter 6 control register 2 (TC6CR2), and two 8-bit timer registers (TTREG6 and PWREG6).

Timer Register

TTREG6 (0905H) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

PWREG6 (0909H) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

Note 1: Do not change the TTREG6 value while the timer is running.

Note 2: In the 8-bit or 16-bit PWM mode, do not change the PWREG6 value while the timer is running.

Note 3: Values that can be set in each timer register are limited depending on the timer's operating mode. For details, see Table 3.9.3.

Timer/Counter 6 Control Register

TC6CR1 (0911H)

7	6	5	4	3	2	1	0
TFF6	TC6CK			TC6S	TC6M		

(Initial value: 0000 0000)

TFF6	Timer F/F6 control	0: Clear 1: Set			R/W
TC6CK	Operating clock select [Hz]		TC6SEL = 0	TC6SEL = 1	
		000	$fc/2^{11}$	$fs/2^3$	
		001	$fc/2^7$	–	
		010	$fc/2^5$	–	
		011	$fc/2^3$	–	
		100	fc/3 (Note 6)	–	
		101	fc/2 (Note 6)	–	
		110	fc (Note 6)	fs (Note 6)	
		111	–		
TC6S	Timer start control	0: Stop & clear counter 1: Start			
TC6M	Operating mode select	000: 8-bit timer mode 001: Reserved 010: 8-bit pulse width modulation (PWM) output mode 011: Reserved 100: 16-bit timer mode 101: Reserved 110: 16-bit pulse width modulation (PwM) output mode 111: 16-bit PPG mode			

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz]

Note 2 Do not change the <TC6M>, <TC6CK>, <TFF6> and <TC6SEL> settings while the timer is running

Note 3: Do not change the <TC6M>, <TC6CK> and <TFF6> settings at the same time as stopping the timer (TC6CR1<TC6S> = 1 → 0) or starting the timer (TC6CR1<TC6S> = 0 → 1).

Note 4: When the timer/counter 6 is used in 16-bit mode, the operating mode is selected by TC6CR1<TC6M> and TC5CR1<TC5M> must be set to 011.

Note 5: When the timer/counter 6 is used in 16-bit mode, various control settings are made in the TC6CR1 register. TC5CR1<TC5S> must be set to 0.

Note 6: Selection of operating clock may be limited depending on the timer's operating mode. For details, see Table 3.9.1 and Table 3.9.2.

Timer/Counter 6 Control Register 2

TC6CR2	7	6	5	4	3	2	1	0	
(0903H)	–	–	–	–	–	–	–	TC6SEL	(Initial value: **** ***0)

TC6SEL	Timer input clock control	0: fc 1: fs	R/W

Note 1: Do not set <TC6SEL>=0 in SLOW or SLEEP mode.

Note 2: Do not change TTREG6 when using for fs in NORMAL mode.

Note 3: *: Undefined value

Figure 3.9.3 Timer Registers and Control Registers for Timer/Counter 6

TC5		TC6		TC7		TC8	
TC5CR	0900H	TC6CR	0901H	TC7CR	0910H	TC8CR	0911H
TC5CR2	0902H	TC6CR2	0903H	TC7CR2	0912H	TC8CR2	0913H
TTREG5	0904H	TTREG6	0905H	TTREG7	0914H	TTREG8	0915H
PWREG5	0908H	PWREG6	0909H	PWREG7	0918H	PWREG8	0919H

Note: Do not access locations where no registers exist in the 0900H to 091FH area.

Table 3.9.1 Source Clocks That Can Be Used in Each Operating Mode

(in NORMAL or IDLE2 mode)

Operating Mode	$fc/2^{11}$ or $fs/2^3$	$fc/2^7$	$fc/2^5$	$fc/2^3$	fc/3	fc/2	fc or fs
8-bit timer	Yes	Yes	Yes	Yes	No	No	No
8-bit PWM	Yes	Yes	Yes	Yes	Yes	Yes	Yes
16-bit timer	Yes	Yes	Yes	Yes	No	No	No
16-bit PWM	Yes	Yes	Yes	Yes	Yes	Yes	Yes
16-bit PPG	Yes	Yes	Yes	Yes	No	No	No

Note: In 16-bit mode (16-bit timer, 16-bit PWM, or 16-bit PPG), the source clock is specified by TC6CR1<TC6CK>.

Table 3.9.2 Source Clocks That Can Be Used in Each Operating Mode

(in SLOW or IDLE2 mode)

Operating Mode	$fs/2^3$	fs
8-bit timer	Yes	No (* Note 2)
8-bit PWM	Yes	Yes
16-bit timer	Yes	No (* Note 2)
16-bit PWM	Yes	Yes
16-bit PPG	Yes	No (* Note 2)

Note 1: In 16-bit mode (16-bit timer, 16-bit PWM, or 16-bit PPG), the source clock is specified by TC6CR1<TC6CK>.

Note 2: Setting is prohibited.

Table 3.9.3 Limitations on Timer Register Settings

Operating Mode	Timer Register Setting
8-bit timer	$1 \leqq (TTREGj) \leqq 255$
8-bit PWM	$2 \leqq (PWREGj) \leqq 254$
16-bit timer	$1 \leqq (TTREG6, 5) \leqq 65535$
16-bit PWM	$2 \leqq (PWREG6, 5) \leqq 65534$
16-bit PPG	$1 \leqq (PWREG6, 5) < (TTREG6, 5) \leqq 65535$ and $(PWREG6, 5) + 1 < (TTREG6, 5)$

Note: $j = 5, 6$

3.9.3 Functional Description

The timer/counters 5 and 6 (TC5 and TC6) have the following five operating modes:

- 8-bit timer mode
- 8-bit pulse width modulation (PWM) output mode
- 16-bit timer mode
- 16-bit pulse width modulation (PWM) output mode
- 16-bit programmable pulse generation (PPG) mode

Each 16-bit mode is realized by cascading the timer/counters 5 and 6.

(1) 8-bit timer mode (TC5 and TC6)

In the 8-bit timer mode, the counter counts up internal clock pulses. When a match between the counter value and the timer register (TTREGj) value is detected, an INTTMRj interrupt is generated and the counter is cleared. The counter then continues counting up.

Note 1: In the 8-bit timer mode, do not change the TTREGj register value while the timer is running. In this mode, the TTREGj does not have a shift register and the value written to the TTREGj is reflected immediately after the write operation. Therefore, if the TTREGj value is changed while the timer is running, unexpected operation may result.

Note 2: j = 5, 6

Table 3.9.4 Source Clock in 8-Bit Timer Mode (Internal Clock/TC5)

Source Clock		Resolution		Maximum Setting Time	
TC5CR2 <TC5SEL> = 0	TC5CR2 <TC5SEL> = 1	fc = 27 MHz	fs = 32.768 kHz	fc = 27 MHz	fs = 32.768 kHz
$fc/2^{11}$ [Hz]	$fs/2^{3}$ [Hz]	75.9 µs	244.14 µs	19.3 ms	62.3 ms
$fc/2^{7}$	–	4.7 µs	–	1.2 ms	–
$fc/2^{5}$	–	1.2 µs	–	302.2 µs	–
$fc/2^{3}$	–	296.3 ns	–	75.6 µs	–

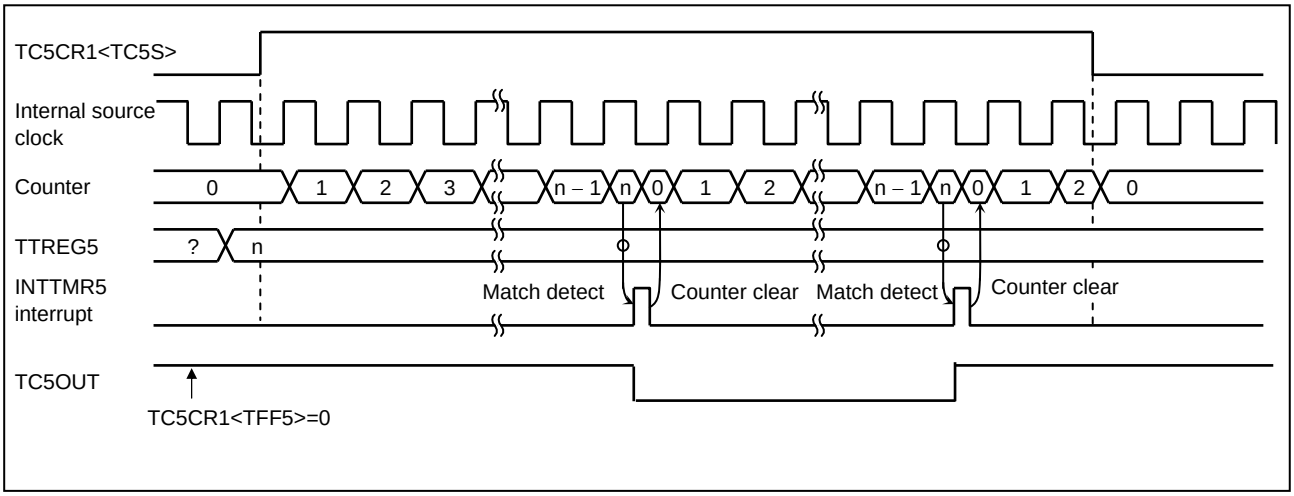


Figure 3.9.4 8-Bit Timer Mode Timing Chart (TC5)

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC5CR2),00H     : Set the <TC5SEL> bit. (<TC5SEL>=0 to select fc)
LD   (TC5CR1),30H     : <TFF5>=0 (Drive TC5OUT pin high)
                          <TC5CK>=011 (fc/2^3)
                          <TC5M>=000 (8-bit timer mode)
LD   (TTREG5),55H     : Set the timer register. (TTREG5=55H)
LD   (TC5CR1),38H     : Start the timer.
```

- Changing the timer register contents (after the timer is started)

```
LD   (TC5CR1),30H     : Stop the timer & clear the counter.
LD   (TTREG5),80H     : Set the timer register. (TTREG5=80H)
LD   (TC5CR1),38H     : Start the timer.
```

- Changing the source clock to fs/2^3 (after the timer is started)

```
LD   (TC5CR1),30H     : Stop the timer & clear the counter.
LD   (TC5CR2),00H     : Set <TC5SEL>=0 to select fc. (NORMAL mode only)
                          Note: In NORMAL mode, do not change TTREG while fs is selected.
LD   (TC5CR1),00H     : <TFF5>=0 (Drive TC5OUT pin high)
                          <TC5CK>=000 (fs/2^3)
LD   (TTREG5),80H     : Set the timer register. (TTREG5=80H)
LD   (TC5CR2),01H     : <TC5SEL>=1 (fs)
LD   (TC5CR1),08H     : Start the timer.
```

(2) 8-bit pulse width modulation (PWM) output mode (TC5 and TC6)

This mode is used to generate pulse width modulated (PWM) signals with a resolution of 8 bits. The counter counts up internal clock pulses. When a match between the counter value and the PWREGi value is detected, the timer flip-flop (F/Fi) is toggled. The counter then continues counting up. When an overflow occurs, the timer F/Fi is toggled again and the counter is cleared. The output from the timer F/Fi is output on the TCiOUT pin after being inverted. When an overflow occurs, an INTTMRi interrupt is generated.

In the PWM mode, the PWREGi register is serially connected to a shift register, enabling the PWREGi value to be changed while the timer is running. While the timer is running, the value written to the PWREGi is shifted into the shift register and becomes valid by an INTTMRi interrupt. This feature makes it possible to change the pulse width continuously. When the timer is not running, the value written to the PWREGi is immediately shifted into the shift register.

When a read instruction is executed on the PWREGi during PWM output, the shift register value is returned instead of the value set in the PWREGi. This means that the new value written to the PWREGi cannot be read out until an INTTMRi interrupt occurs; up to that point the previous PWREGi value is read out.

Note 1: In the PWM mode, the PWREGi register should be written to immediately after an INTTMRi interrupt occurred (normally in the INTTMRi interrupt service routine). If a write to the PWREGi and an INTTMRi interrupt occur simultaneously, an unstable value is shifted into the shift register, causing unexpected pulses to be generated until the next INTTMRi interrupt occurs.

Note 2: When the timer is stopped during PWM output, the TCiOUT pin retains its current output state. After the timer stops, the TCiOUT pin state can be changed to a desired level by using TCiCR1<TFFi>. Be careful not to set TCiCR1<TFFi> at the same time as stopping the timer.

Note 3: i = 5, 6

Table 3.9.5 PWM Output Mode

Source Clock		Resolution		Repeat Cycle	
TC5SEL = 0	TC5SEL = 1	fc = 27 MHz	fs = 32.768 kHz	fc = 27 MHz	fs = 32.768 kHz
$fc/2^{11}$ [Hz]	$fs/2^{3}$ [Hz]	75.9 μs	244.14 μs	19.4 ms	62.5 ms
$fc/2^{7}$		4.7 μs		1.2 ms	
$fc/2^{5}$		1.2 μs		303.4 μs	
$fc/2^{3}$		296.3 ns		75.9 μs	
fc/3		111.1 ns		28.4 μs	
fc/2		74.1 ns		19.0 μs	
fc	fs	37.0 ns	30.5 μs	9.5 μs	7.81 ms

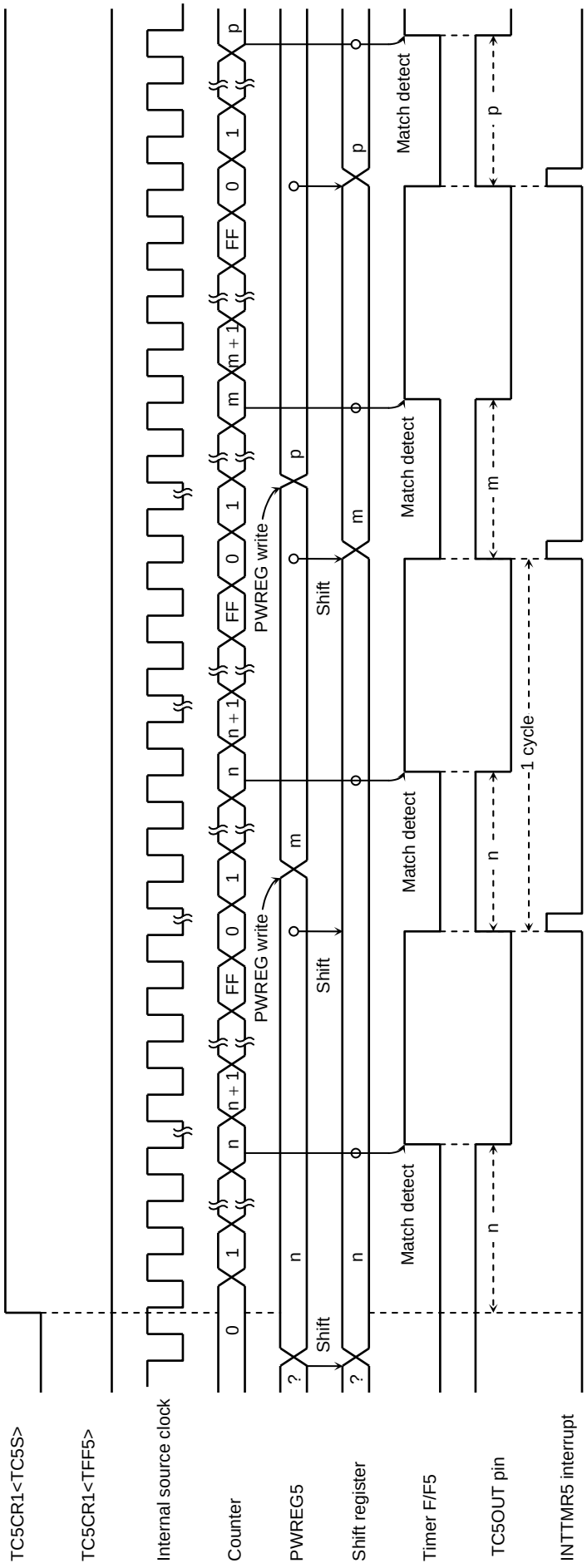


Figure 3.9.5 8-Bit PWM Output Mode Timing Chart (TC5)

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC5CR2),00H     : Set the <TC5SEL> bit. (<TC5SEL>=0 to select fc)
LD   (TC5CR1),62H     : <TFF5>=0 (Drive TC5OUT pin high)
                          <TC5CK>=110 (fc)
                          <TC5M>=010 (8-bit PWM mode)
LD   (PWREG5),55H     : Set the timer register. (PWREG5=55H)
LD   (TC5CR1),6AH     : Start the timer.
```

- Stopping the timer

```
LD   (TC5CR1),62H     : Stop the timer & clear the counter.
LD   (TC5CR1),62H     : <TFF5>=0 (Drive TC5OUT pin high)
                          <TFF5> must be set again after the timer is stopped.
```

(3) 16-bit timer mode (TC5 + TC6)

In the 16-bit timer mode, the counter counts up internal clock pulses. The timer/counters 5 and 6 are cascaded to function as a 16-bit timer.

After the timer is started by setting TC6CR1<TC6S>, a match between the counter value and the timer register (TTREG5, TTREG6) value generates an INTTMR6 interrupt and clears the counter. The counter then continues counting up. The timer register must be set in the order of lower byte (TTREG5) and upper byte (TTREG6). (It is also possible to change only the lower or upper byte of the timer register.)

Note 1: In the 16-bit timer mode, do not change the TTREGj register value while the timer is running. In this mode, the TTREGj does not have a shift register and the value written to the TTREGj is reflected immediately after the write operation. Therefore, if the TTREGj value is changed while the timer is running, unexpected operation may result.

Note 2: j = 5, 6

Table 3.9.6 1 Source Clock in 16-Bit Timer Mode (TC6)

Source clock		Resolution		Repeat Cycle	
TC6CR2 <TC6SEL> = 0	TC6CR2 <TC6SEL> = 1	fc = 27 MHz	fs = 32.768 kHz	fc = 27 MHz	fs = 32.768 kHz
$fc/2^{11}$	$fs/2^3$	75.9 μs	244.14 μs	4.97 s	16 s
$fc/2^7$		4.7 μs	–	310.7 ms	–
$fc/2^5$		1.2 μs	–	77.7 ms	–
$fc/2^3$		296.3 ns	–	19.4 ms	–

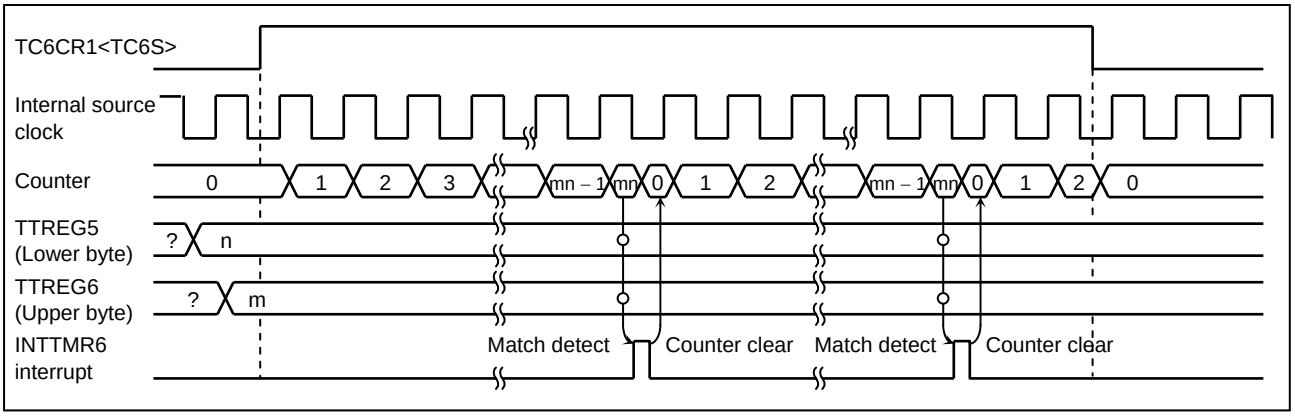


Figure 3.9.6 16-Bit Timer Mode Timing Chart (TC5 + TC6)

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC6CR2),00H    : Set the <TC6SEL> bit. (<TC6SEL>=0 to select fc)
LD   (TC5CR1),03H    : <TC5M>=011 (16-bit mode)
LD   (TC6CR1),B4H    : <TFF6>=1 (Drive TC6OUT pin low)
                         <TC6CK>=011 (fc/2^3)
                         <TC6M>=100 (16-bit timer mode)
LD   (TTREG5),55H    : Set the timer register. (TTREG5=55H)
LD   (TTREG6),AAH    : Set the timer register. (TTREG6=AAH)
LD   (TC6CR1),BCH    : Start the timer.
```

- Changing the timer register contents (after the timer is started)

```
LD   (TC6CR1),B4H    : Stop the timer & clear the counter.
                         <TFF6>=1 (Drive TC6OUT pin low)
LD   (TTREG5),80H    : Set the timer register. (TTREG5=80H)
LD   (TC6CR1),BCH    : Start the timer.
```

- Changing the source clock to $fs/2^3$ (after the timer is started)

```
LD   (TC6CR1),B4H    : Stop the timer & clear the counter.
LD   (TC6CR2),00H    : Set <TC6SEL>=0 to select fc once. (NORMAL mode only)
                         Note: In NORMAL mode, do not change TTREG while fs is selected.
LD   (TC6CR1),04H    : <TFF6>= 0 (Drive TC6OUT high)
                         <TC6CK>=000 (fs/2^3)
                         <TC6M>=100 (16-bit timer mode)
LD   (TTREG5),80H    : Set the timer register. (TTREG5=80H)
LD   (TC6CR2),01H    : <TC6SEL>=1 (fs)
LD   (TC6CR1),0CH    : Start the timer.
```

(4) 16-bit pulse width modulation (PWM) output mode (TC5 + TC6)

This mode is used to generate pulse width modulated (PWM) signals with a resolution of 16 bits. The timer/counters 5 and 6 are cascaded to realize the 16-bit PWM output mode.

When a match between the counter value and the timer register (PWREG5, PWREG6) value is detected, the timer flip-flop 6 (F/F6) is toggled. The counter then continues counting up. When an overflow occurs, the timer F/F6 is toggled again, the counter is cleared, and an INTTMR6 interrupt is generated.

In the PWM mode, the PWREG5 and PWREG6 registers are serially connected to a shift register, enabling the PWREG5 and PWREG6 values to be changed while the timer is running. While the timer is running, the values written to the PWREG5 and PWREG6 are shifted into the shift register and become valid by an INTTMR6 interrupt. This feature makes it possible to change the pulse width continuously. When the timer is not running, the values written to the PWREG5 and PWREG6 are immediately shifted into the shift register. When writing to the PWREG5 and PWREG6, be sure to write in the order of lower byte (PWREG5) and upper byte (PWREG6). (It is also possible to change only the lower or upper byte of the timer register.)

When a read instruction is executed on the PWREG5 and PWREG6 during PWM output, the shift register value is returned instead of the value set in the PWREG5 and PWREG6. This means that the new value written to the PWREG5 and PWREG6 cannot be read out until an INTTMR6 interrut occurs; up to that point the previous PWREG5 abd PWREG6 values are read out.

Note 1: In the PWM mode, the timer registers PWREG6 and PWREG5 should be written to immediately after an INTTMR6 interrupt occurred (normally in the INTTMR6 interrupt service routine). If a write to the PWREG6 and PWREG5 and an INTTMR6 interrupt occur simultaneously, an unstable value is shifted into the shift register, causing unexpected pulses to be generated until the next INTTMR6 interrupt occurs.

Note 2: If the timer is stopped during PWM output, the TC6OUT pin retaines its current output state. After the timer stops, the TC6OUT pin state can be chagned to a desired level by using TC6CR1<TFF6>. Be careful not to set TC6CR1<TFF6> at the same time as stopping the timer.

(For example, the TC6OUT pin should be fixed to high level while the timer/counter is not running.)

```
RES   3, (TC6CR1)    : Stop the timer & clear the counter.
RES   7, (TC6CR1)    : <TFF6>=0 (Drive TC6OUT pin high)
```

Table 3.9.7 16-Bit PWM Output Mode

Source Clock		Resolution		Repeat Cycle	
TC6CR2 <TC6SEL>= 0	TC6CR2 <TC6SEL>= 1	fc = 27 MHz	fs = 32.768 kHz	fc = 27 MHz	fs = 32.768 kHz
$fc/2^{11}$	$fs/2^3$	75.9 μs	244.14μs	4.97 s	16 s
$fc/2^7$		4.7 μs	–	310.7 ms	–
$fc/2^5$		1.2 μs	–	77.7 ms	–
$fc/2^3$		296.3 ns	–	19.4 ms	–
fc/3		111.1 ns	–	7.3 ms	–
fc/2		74.1 ns	–	4.9 ms	–
fc	fs	37.0 ns	30.52 μs	2.4 ms	2 s

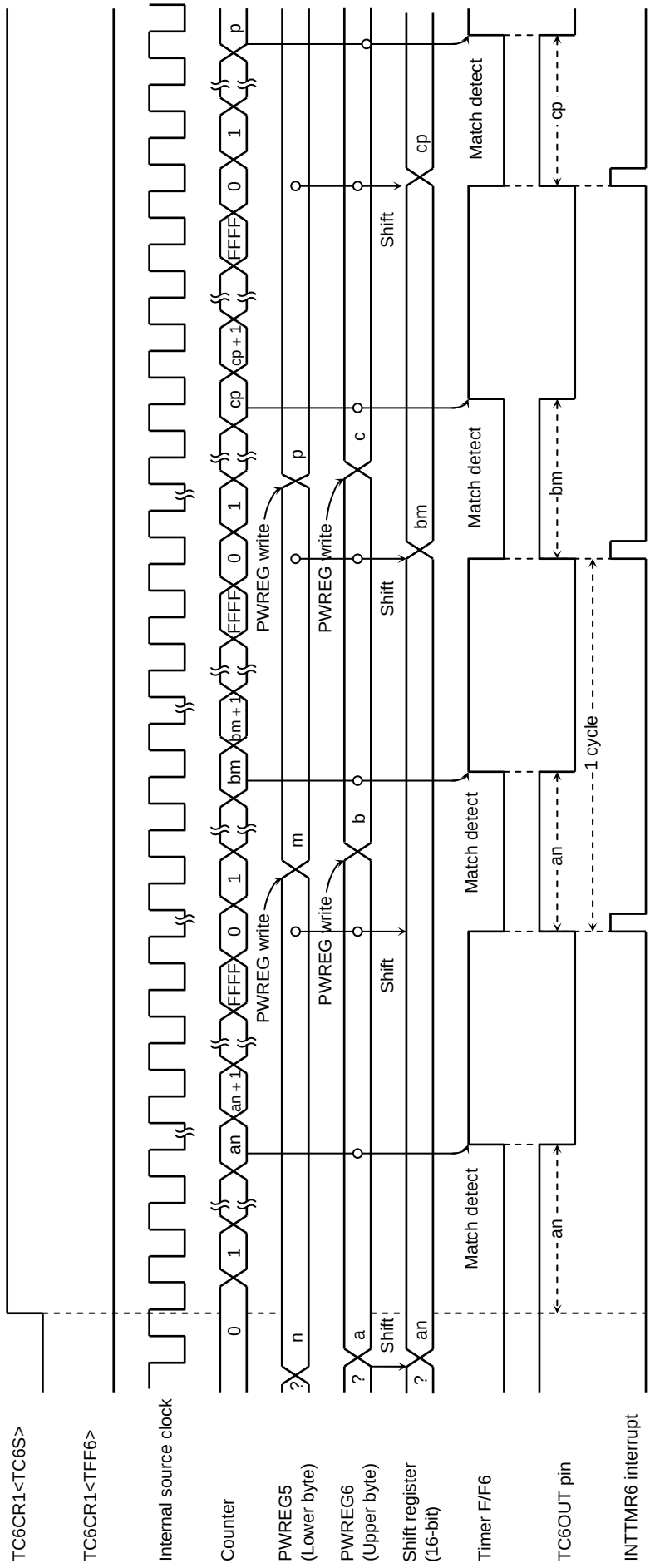


Figure 3.9.7 16-Bit PWM Mode Timing Chart (TC5 + TC6)

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC6CR2),00H      : Set the <TC6SEL> bit. (<TC6SEL>=0 to select fc)
LD   (TC5CR1),03H      : <TC5M>=011 (16-bit mode)
LD   (TC6CR1),36H      : <TFF6>=0 (Drive TC6OUT pin high)
                           <TC6CK>=011 (fc/2^3)
                           <TC6M>=110 (16-bit PWM mode)
LD   (PWREG5),55H      : Set the timer register. (PWREG5=55H)
LD   (PWREG6),0AAH     : Set the timer register. (PWREG6=AAH)
LD   (TC6CR1),3EH      : Start the timer.
```

- Changing the timer register contents (after the timer is started)

```
LD   (TC6CR1),036H     : Stop the timer & clear the counter.
                           <TFF6>=0 (Drive TC6OUT pin high)
LD   (PWREG5),80H      : Set the timer register. (PWREG=80H).
LD   (TC6CR1),3EH      : Start the timer.
```

- Changing the source clock to $fs/2^3$ (after the timer is started)

```
LD   (TC6CR1),36H      : Stop the timer & clear the counter.
LD   (TC6CR2),00H      : Set <TC6SEL>=0 to select fc once. (NORMAL mode only)
                           Note: In NORMAL mode, do not change PWREG while fs is selected.
LD   (TC6CR1),06H      : <TFF6>=0 (Drive TC6OUT pin high)
                           <TC6CK>=000 (fs/2^3)
                           <TC6M>=110 (16-bit PWM mode)
LD   (PWREG5),80H      : Set the timer register. (PWREG5=80H)
LD   (TC6CR2),01H      : <TC6SEL>=1 (fs)
LD   (TC6CR1),0EH      : Start the timer.
```

(5) 16-bit programmable pulse generation (PPG) mode (TC5 + TC6)

In the 16-bit programmable pulse generation (PPG) mode, the timer/counters 5 and 6 are cascaded to function as a 16-bit timer.

When a match between the counter value and the timer register (PWREG5, PWREG6) value is detected, the timer flip-flop 6 (F/F6) is toggled. The counter then continues counting up. When a match between the counter value and the timer register (TTREG5, TTREG6) value is detected, the timer F/F6 is toggled again, the counter is cleared, and an INTTMR6 interrupt is generated. A reset clears the timer F/F6 to 0. The timer F/F6 value can be set in TC6CR1<TFF6>, enabling generation of either high-going or low-going pulses. The timer registers must be set in the order of lower byte and upper byte (i.e. TTREG5 → TTREG6、PWREG5 → PWREG6). (It is also possible to change only the lower or upper byte of the timer register.)

Note 1: In the PPG mode, do not change the PWREGi and TTREGi register values while the timer is running. In this mode, the PWREGi and TTREGi do not have a shift register and the value set to the PWREGi and TTREGi is reflected immediately after the write operation. Therefore, if the PWREGi or TTREGi value is changed while the timer is running, unexpected operation may result.

Note 2: When the timer is stopped during PPG output, the TC6OUT pin retains its current output state. After the timer stops, the TC6OUT pin state can be changed to a desired level by using TC6CR1<TFF6>. Be careful not to set TC6CR1<TFF6> at the same time as stopping the timer.
(For example, the TC6OUT pin should be fixed to high level when the timer is not running.)

```
RES   3, (TC6CR1)      : Stop the timer & clear the counter.
RES   7, (TC6CR1)      : <TFF6>=0 (Drive TC6OUT pin high)
```

Note 3: i = 5, 6

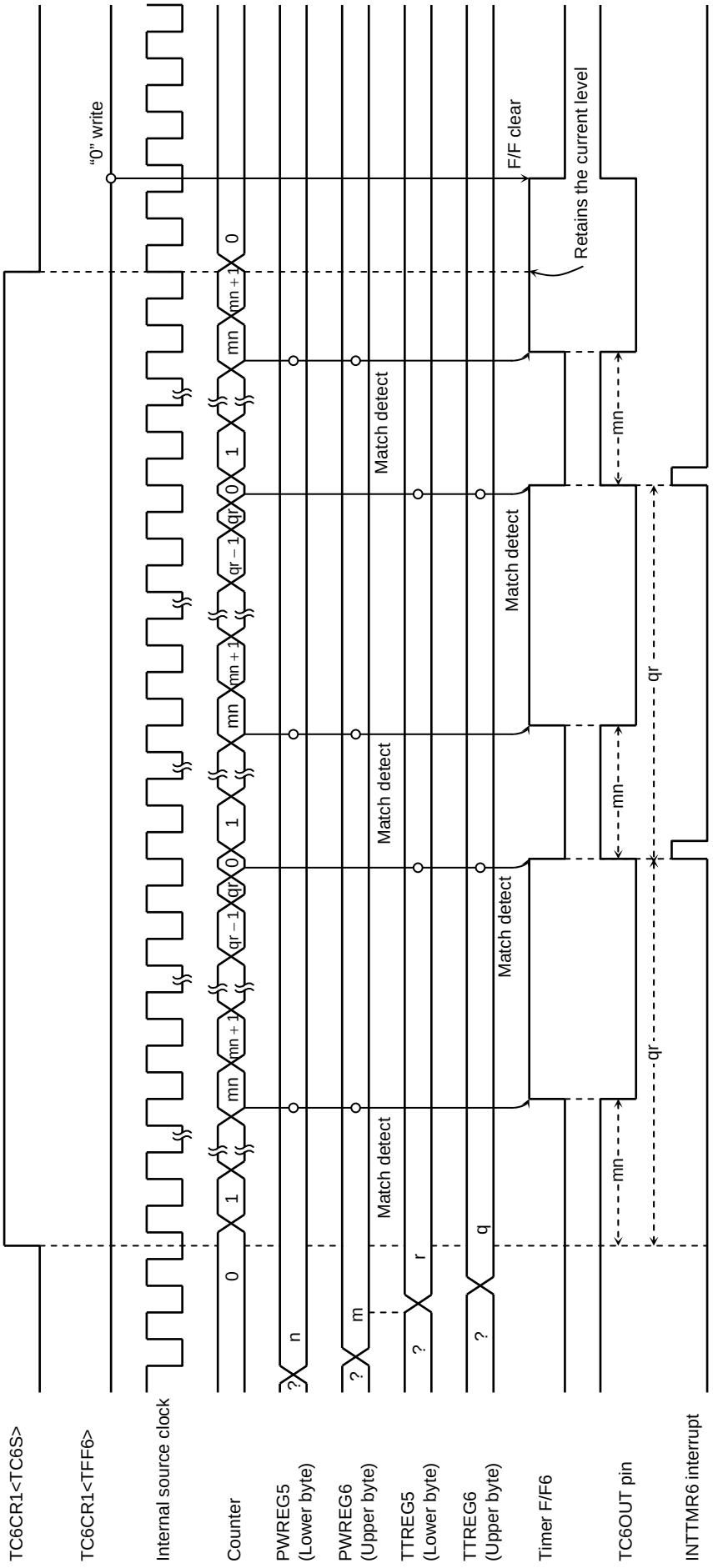


Figure 3.9.8 16-Bit PPG Mode Timing Chart (TC5 + TC6)

Programming sequences (Be sure to follow these sequences.)

- Setting initial values

```
LD   (TC6CR2),00H     : Set the <TC6SEL> bit. (<TC6SEL>=0 to select fc)
LD   (TC5CR1),03H     : <TC5M>=011 (16-bit mode)
LD   (TC6CR1),37H     :<TFF6>=0 (Drive TC6OUT pin high)
                                <TC6CK>=011 (fc/2^3)
                                <TC6M>=111 (16-bit PPG mode)
LD   (PWREG5),80H     : Set the timer register. (PWREG5=80H)
LD   (PWREG6),00H     : Set the timer register. (PWREG6=00H)
LD   (TTREG5),00H     : Set the timer register. (TTREG5=00H)
LD   (TTREG6),02H     : Set the timer register. (TTREG6=02H)
LD   (TC6CR1),3FH     : Start the timer.
```

- Changing the timer register contents (after the timer is started)

```
LD   (TC6CR1),37H     : Stop the timer & clear the counter.
                          <TFF6>=0 (Drive TC6OUT pin high)
LD   (PWREG5),0FFH    : Set the timer register (PWREG5=FFH)
LD   (TC6CR1),3FH     : Start the timer.
```

- Changing the source clock to $fs/2^3$ (after the timer is started)

```
LD   (TC6CR1),37H     : Stop the timer & clear the counter.
LD   (TC6CR2),00H     : Set <TC6SEL>=0 to select fc once. (NORMAL mode only)
                          Note: In NORMAL mode, do not change PWREG while fs is selected.
LD   (TC6CR1),07H     : <TFF6>=0 (Drive TC6OUT pin high)
                          <TC6CK>=000 (fs/2^3)
                          <TC6M>=111 (16-bit PPG mode)
LD   (PWREG5),80H     : Set the timer register. (PWREG5=80H)
LD   (TC6CR2),01H     : Set <TC6SEL>=1 to select fs.
LD   (TC6CR1),0FH     : Start the timer.
```

3.10 Serial I/O (SIO)

The TMP91CW40 contains four serial I/O channels (SIO0, SIO1, SIO2 and SIO3). For each channel, universal asynchronous receiver/transmitter (UART) mode or synchronous I/O interface mode can be selected.

- I/O interface mode ——— Mode 0: 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, a wakeup mode is available for multidrop applications in which a master controller is connected to several slave controllers through a serial link.

Figure 3.10.2 to Figure 3.10.5 show block diagrams of SIO0, SIO1, SIO2 and SIO3, respectively.

The main components of each SIO channel are a clock prescaler, a serial clock generator, a receive buffer, a receive controller, a transmit buffer and a transmit controller.

Each of the four channels operates independently, and is functionally equivalent. In the following sections, any references to SIO0 also apply to other channels, unless otherwise noted. Table 3.10.1 shows the pins used for each SIO channel.

Table 3.10.1 Pins Used for Each SIO Channel

	SIO0	SIO1	SIO2	SIO3
Pin	TXD0 (P90) $\overline{\text{RXD0}}$ (P91) $\overline{\text{CTS0}}$ /SCLK0 (P92)	TXD1 (P93) $\overline{\text{RXD1}}$ (P94) $\overline{\text{CTS1}}$ /SCLK1 (P95)	TXD2 (PA0) $\overline{\text{RXD2}}$ (PA1) $\overline{\text{CTS2}}$ /SCLK2 (PA2)	TXD3 (PA3) $\overline{\text{RXD3}}$ (PA4) $\overline{\text{CTS3}}$ /SCLK3 (PA5)

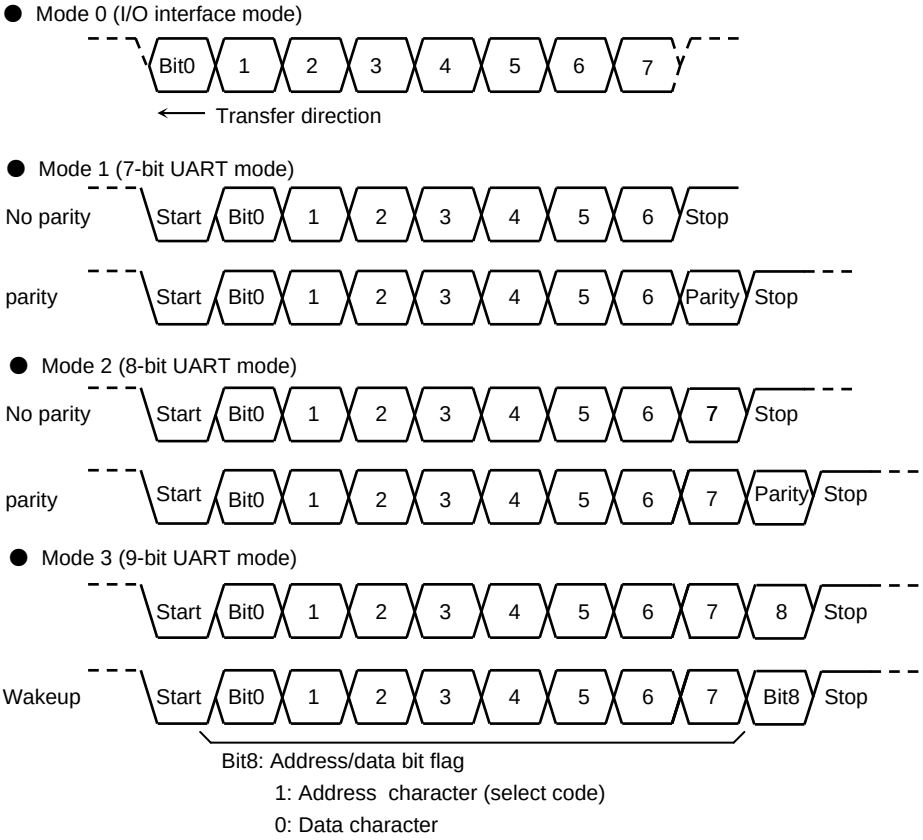


Figure 3.10.1 Data Formats

3.10.1 Block Diagrams

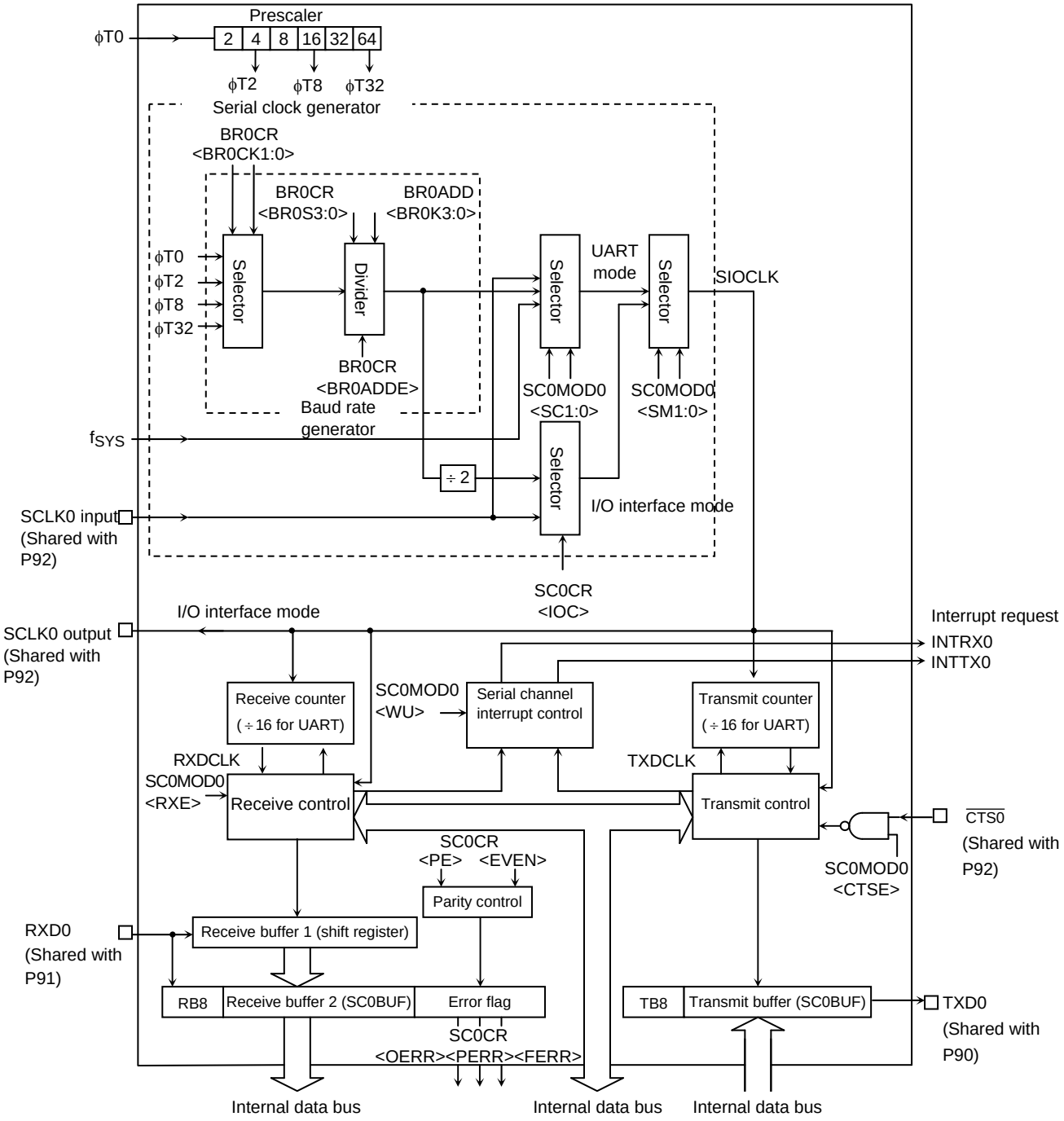


Figure 3.10.2 SIO0 Block Diagram

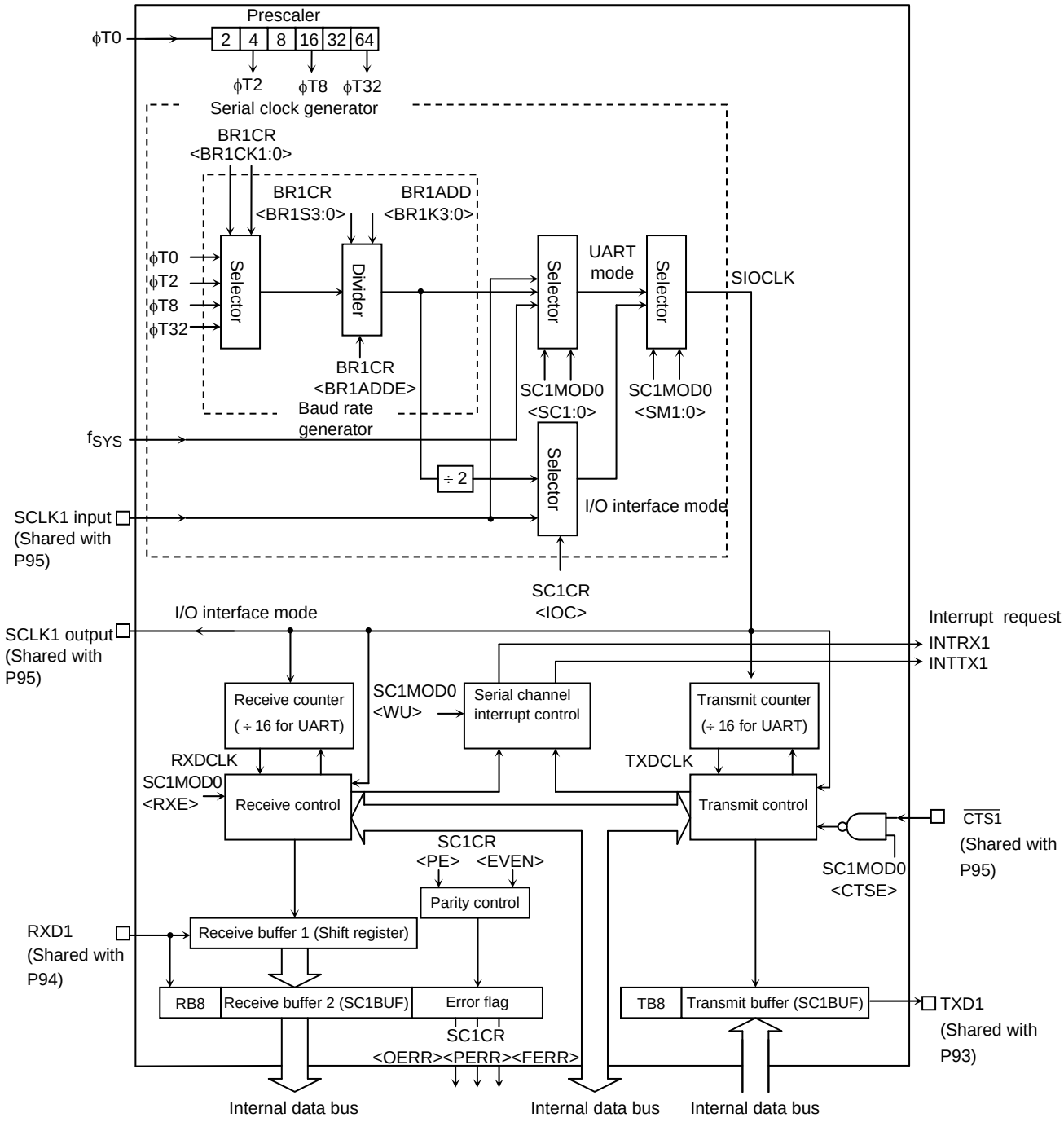


Figure 3.10.3 SIO1 Block Diagram

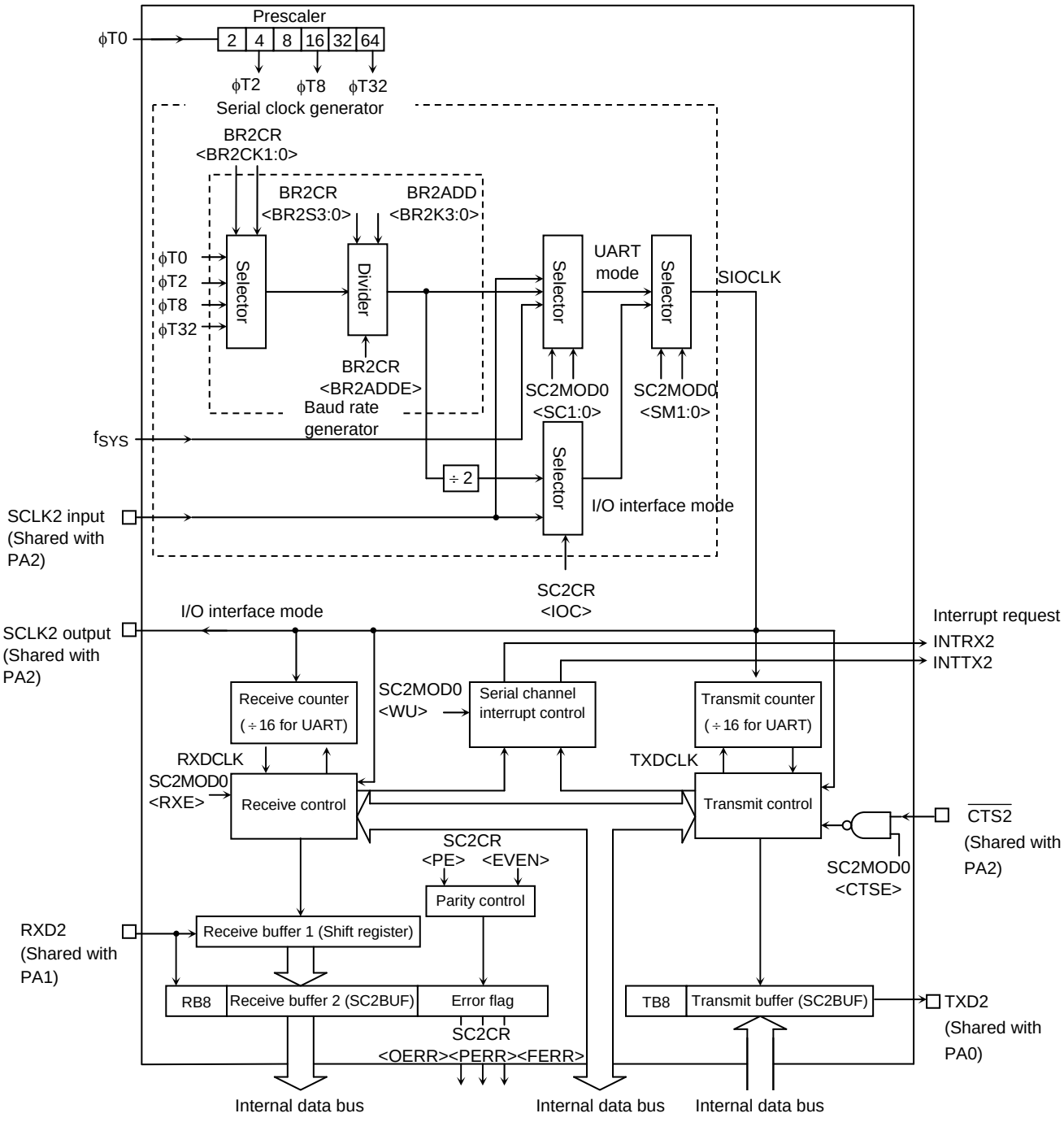


Figure 3.10.4 SIO2 Block Diagram

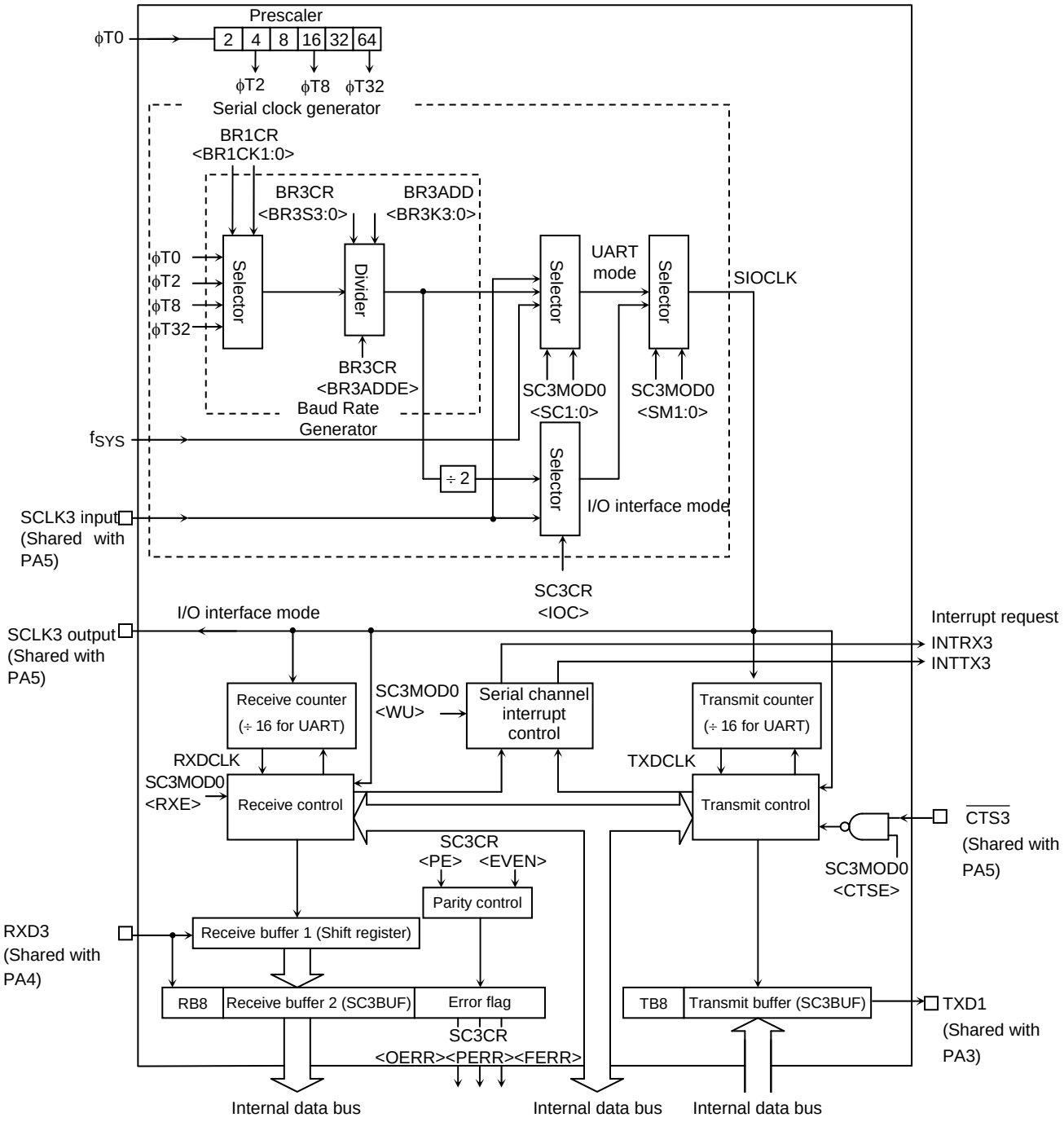


Figure 3.10.5 SIO3 Block Diagram

3.10.2 SIO Components

(1) Prescaler

The SIO0 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 of the clock selected by the <PRCK1:0> field in the SYSCR0 located within the clock gear.

The prescaler is only enabled when the baud rate generator output clock is selected as a serial clock. Table 3.10.2 shows prescaler output clock resolutions.

Table 3.10.2 Prescaler Output Clock Resolutions

System Clock Source SYSCR1 <SYSCK>	—	Prescaler Output Clock Resolution BR0CR<BR0CK1:0>			
		ϕT0(1/1)	ϕT2(1/4)	ϕT8(1/16)	ϕT32(1/64)
1 (fs)	1/4	fs/4	fs/16	fs/64	fs/256
0 (fc)		fc/4	fc/16	fc/64	fc/256

The prescaler can output four types of clock (ϕT0, ϕT2, ϕT8, ϕT32) to the baud rate generator.

(2) Baud rate generator

The frequency used to transmit and receive data through the SIO0 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 <BR0CK1:0> field in the BR0CR.

The baud rate generator contains a clock divisor that can divide the selected clock by 1, N+(16–K)/16, or 16. The clock divisor is programmed into the <BR0ADDE> and <BR0S3:0> bits in the BR0CR and the <BR0K3:0> bits in the BR0ADD.

- UART mode

(1) When BR0CR<BR0ADDE> = 0

When the <BR0ADDE> bit is cleared, the BR0ADD<BR0K3:0> field has no meaning or effect. The baud rate generator input clock is divided down by a value of N (1 to 16) programmed in the BR0CR<BR0S3:0> field.

(2) When BR0CR<BR0ADDE> = 1

Setting the <BR0ADDE> bit enables the N+ (16 – K)/16 clock division function. The baud rate generator input clock is divided down according to a value of N (2 to 15) programmed in the BR0CR<BR0S3:0> field and a value of K (1 to 15) programmed in the BR0ADD<BR0K3:0> field.

Note: Setting N to 1 or 16 disables the N+(16–K)/16 clock division function. When N=1 or 16, the BR0CR<BR0ADDE> bit must be cleared to 0.

- I/O interface mode

In I/O interface mode, the N + (16 – K)/16 clock division function cannot be used. The BR0CR<BR0ADDE> bit must be cleared to 0, so the baud rate generator input clock is divided down by a value of N (1 to 16) programmed in the BR0CR<BR0S3:0> 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 = 12.288 MHz

  Input clock: ϕT2

  Clock divisor N (BR0CR<BR0S3:0>) = 5

  BR0CR<BR0ADDE> = 0

  * Clocking conditions: System clock: High-speed (fc)

  The baud rate is determined as follows:

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

$$= \frac{fc/16}{5} \div 16$$

$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \text{ (bps)}$$

  Note: Clearing the BR0CR<BR0ADDE> to 0 disables the N + (16 – K)/16 clock division function. At this time, the BR0ADD<BR0K3:0> field is ignored.

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

  fc = 4.8 MHz

  Input clock: ϕT0

  N (BR0CR<BR0S3:0>) = 7

  K (BR0ADD<BR0K3:0>) = 3

  BR0CR<BR0ADDE> = 1

  * Clocking conditions: System clock: High-speed (fc)

  The baud rate is determined as follows:

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

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

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

Table 3.10.3 shows the UART baud rates obtained with various combinations of clock inputs and clocl 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 3.10.3 UART Baud Rate Selection

(when the baud rate generator is used and BR0CR<BR0ADDE> = 0) Unit: (kbps)

fc [MHz]	Input Clock / Divisor N (Programmed in BR0CR<BR0S3:0>)	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
12.288000	5	38.400	9.600	2.400	0.600
↑	A	19.200	4.800	1.200	0.300
14.745600	2	115.200	28.800	7.200	1.800
↑	3	76.800	19.200	4.800	1.200
↑	6	38.400	9.600	2.400	0.600
↑	C	19.200	4.800	1.200	0.300
19.6608	1	307.200	76.800	19.200	4.800
↑	2	153.600	38.400	9.600	2.400
↑	4	76.800	19.200	4.800	1.200
↑	8	38.400	9.600	2.400	0.600
↑	10	19.200	4.800	1.200	0.300
22.1184	3	115.200	28.800	7.200	1.800
24.576	1	384.000	96.000	24.000	6.000
↑	2	192.000	48.000	12.000	3.000
↑	4	96.000	24.000	6.000	1.500
↑	5	76.800	19.200	4.800	1.200
↑	8	48.000	12.000	3.000	0.750
↑	A	38.400	9.600	2.400	0.600
↑	10	24.000	6.000	1.500	0.375

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

Note 2: This table assumes: system clock = fc.

Baud Rate Setting Examples

fc [MHz]	19,200 bps	9,600 bps	4,800 bps
8 MHz	19231bps (error +0.16%) φT0, N=6, K=8	9615bps (error +0.16%) φT0, N=13, K=not used	4808bps (error +0.16%) φT2, N=6, K=8
16 MHz	19231bps (error +0.16%) φT0, N=13, K=not used	9615bps (error +0.16%) φT2, N=6, K=8	4808bps (error +0.16%) φT2, N=13, K=not used
24 MHz	19231bps (error +0.16%) φT2, N=4, K=2	9615bps (error +0.16%) φT2, N=9, K=4	4808bps (error +0.16%) φT8, N=4, K=2

Note 1: This table assumes: system clock = fc.

Example: Transferring data with fc = 16 MHz, 8-bit UART mode, transfer rate = 9600 bps

```
LD   (SC0MOD0),09H     : Select the baud rate generator.
LD   (BR0ADD),08H      : K = 8 for N+(16-K)/16 division
LD   (BR0CR),56H       : Select N+(16-K)/16 division.
                       : Select φT2 as the baud rate generator source clock.
                       : Divisor N = 6
```

(3) Serial clock generator

This block generates a basic clock (SIOCLK) for controlling transmit and receive operations.

- I/O interface mode

  When the SCLK pin is configured as an output by clearing the SC0CR<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 SC0CR<IOC> bit to 1, the external SCLK clock is used as the SIOCLK clock; the SC0CR<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 timer TMRA0, and the external SCLK0 clock according to the setting of the SC0MOD0<SC1:0> 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 uses 16 clocks for each received bit, and oversamples each bit three times around their center (with 7th to 9th clocks). 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

  When the SCLK pin is configured as an output by clearing the SC0CR<IOC> bit to 0, the receive controller samples the RXD0 input at the rising or falling edge of the shift clock driven out from the SCLK pin.

  When the SCLK pin is configured as an input by setting the SC0CR<IOC> bit to 1, the receive controller samples the RXD0 pin at either the rising or falling edge of the SCLK clock, as programmed in the SC0CR<SCLKS> bit.

- UART mode

  The receive controller contains the start bit detection logic. Once a valid start bit is detected (at least two 0s 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 (i.e., 7 or 8 bits, as programmed) is loaded into receive buffer 1, it is transferred to receive buffer 2 (SC0BUF), and the receive-done interrupt (INTRX0) is generated.

The CPU reads a character from receive buffer 2 (SC0BUF). Receive buffer 1 can start accepting a new character 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 SC0CR<RB8> bit are preserved.

The SC0CR<RB8> bit holds the parity bit in 8-bit UART mode with parity and the most-significant bit in 9-bit UART mode.

In 9-bit UART mode, the slave controller wakeup feature allows the slave controller in a multidrop system to wake up whenever an address character is received. Setting the SC0MOD0<WU> bit to 1 enables the wakeup feature. When the SC0CR<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.

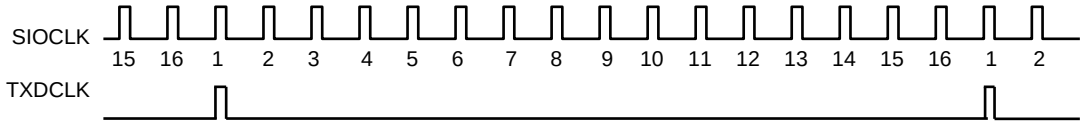


Figure 3.10.6 Transmit Clock Generation

(8) Transmit controller

- I/O interface mode

  When the SCLK pin is configured as an output by clearing the SC0CR<IOC> to 0, the transmit controller shifts out each bit in the transmit buffer to the TXD0 pin at the rising or falling edge of the shift clock driven out on the SCLK0 pin.

  When the SCLK0 pin is configured as an input by setting the SC0CR<IOC> bit to 1, the transmit controller shift out each bit in the transmit buffer to the TXD0 pin at the rising or falling edge of the SCLK input, as programmed in the SC0CR<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 each have the clear-to-send ($\overline{CTS}$) pin. When the $\overline{CTS}$ operation is enabled, the $\overline{CTS}$ input must be low in order for a character to be transmitted. This feature can be used for flow control to prevent overrun errors in the receiver. The SC0MOD0<CTSE> bit enables and disables the $\overline{CTS}$ operation.

If the $\overline{CTS0}$ pin goes high in the middle of a transmission, the transmit controller stops transmission upon completion of the current character until $\overline{CTS0}$ goes low again. The transmit controller generates the INTTX0 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 $\overline{CTS0}$ going low.

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

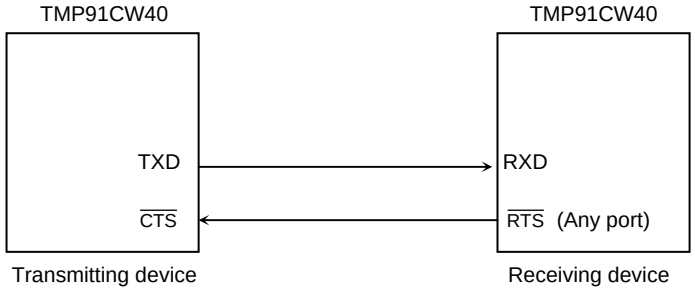
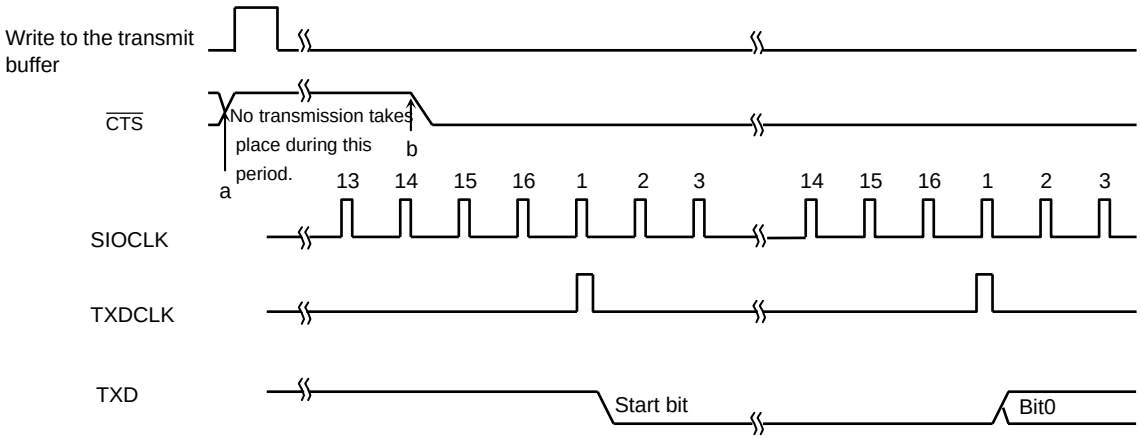


Figure 3.10.7 Handshaking Signals



Note: a. When $\overline{CTS}$ goes high in the middle of transmission, the transmiter 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 $\overline{CTS}$ signal goes low.

Figure 3.10.8 $\overline{CTS}$ (Clear-to-send) Signal Timing

(9) Transmit buffer

Once the CPU loads a character into the transmit buffer (SC0BUF), 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 INTT0 interrupt is generated to the CPU.

(10) Parity controller

For transmit operations, setting the SC0CR<PE> bit to 1 enales parity generation in 7- and 8-bit UART modes. The SC0CR<EVEN> bit selects either even or odd parity.

If enabled, the parity controller automatically generates parity for the character in the transmit buffer (SC0BUF). In 7-bit UART mode, the SC0BUF<TB7> bit holds the parity bit. In 8-bit UART mode, the SC0MOD0<TB8> bit holds the parity bit. The SC0CR<PE> and <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 (SC0BUF) . The received parity bit is compared to the SC0BUF<RB7> bit in 7-bit UART mode and to the SC0CR<RB8> bit in 8-bit UART mode. If a character is received with incorrect parity, the SC0CR<PERR> bit is set.

(11) Error flags

The SC1CR register 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 (SC0BUF) 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 (SC0BUF).

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 timing	Middle of the last bit (bit 8)	Middle of the last bit (parity bit)	Middle of the stop bit
Framing error timing	Middle of the stop bit	Middle of the stop bit	Middle of the stop bit
Parity error timing	—	Middle of the last bit (parity bit)	Middle of the stop bit
Overrun error timing	Middle of the last bit (bit 8)	Middle of the last bit (parity bit)	Middle of the stop bit

Note: In 9 data bits and 8 data bits with parity mode, interrupts coincide with the ninth bit pulse. Thus, when an interrupt occurs, it is necessary to wait for a 1-bit period (to allow the stop bit to be transferred) before 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 timing	Immediately before the stop bit is shifted out	Immediately before the stop bit is shifted out	Immediately before the stop bit is shifted out

b. I/O interface mode

Transmit interrupt timing	SCLK output mode	Immediately after the last bit data. (See Figure 3.10.16)
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed. (See Figure 3.10.17)
Receive interrupt timing	SCLK output mode	When a received character has been transferred to receive buffer 2 (SC0BUF) (i.e. immediately after the last SCLK pulse) (See Figure 3.10.18)
	SCLK input mode	When a received character has been transferred to receive buffer 2 (SC0BUF) (i.e. immediately after the last SCLK pulse) (See Figure 3.10.19)

3.10.3 SFRs

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Bit 8 of a transmitted character	Handshake control 0: Disable CTS 1: Enable CTS	Receive control 0: Disable 1: Enable	Wakeup function 0: Disable 1: Enable	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: Setting prohibited 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLKn input)	

SC0MOD0 (0202H)
SC1MOD0 (020AH)
SC2MOD0 (0212H)
SC3MOD0 (021AH)

Serial clock (for UART)

00	Setting prohibited
01	Baud rate generator
10	Internal clock f_{SYS}
11	External clock (SCLKn input)

Note: In I/O interface mode, the serial control register (SCnCR) 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 SCnCR<RB8> = 1	

Receive control

0	Disable
1	Enable

Handshake ($\overline{CTS}$ pin)

0	Disable (Data streams accepted at all times)
1	Enable

Bit 8 of a transmitted character

Figure 3.10.9 Serial Mode Control Register 0

SC0CR (0201H)
SC1CR (0209H)
SC2CR (0211H)
SC3CR (0219H)

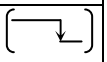
A read -modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (Cleared when read)			R/W	
After reset	Undefined	0	0	0	0	0	0	0
Function	Bit 8 of a received character	Parity type 0: Odd 1: Even	Parity 0: Disable 1: Enable	1: Error has occurred			0: SCLKn  1: SCLKn 	0: Baud rate generator 1: SCLKn pin input
				Overrun error	Parity error	Framing error		

Input clock in I/O interface mode

0	Baud rate generator
1	SCLKn pin input

Active edge for the SCLKn input/output

0	Transmit/receive data on the SCLKn rising edge. 
1	Transmit/receive data on the SCLKn falling edge. 

Framing error flag
Parity error flag
Overrun error flag

These bits are cleared to 0 when read.

Parity generation

0	Disable
1	Enable

Parity type

0	Odd parity
1	Even parity

Bit 8 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 3.10.10 Serial Control Register

	7	6	5	4	3	2	1	0
BR0CR (0203H) BR1CR (020BH) BR2CR (0213H) BR3CR (021BH) Bit symbol	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Always write 0.	+ (16–K) /16 function 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Divisor N			

+ (16 – K)/16 function

0	Disable
1	Enable

Clock source for baud rate generator

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

	7	6	5	4	3	2	1	0
BR0ADD (0204H) BR1ADD (020CH) BR2ADD (0214H) BR3ADD (021CH) Bit symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write					R/W			
After reset					0	0	0	0
Function					Value K for N + (16 – K)/16 division			

Clock divisor value for baud rate generator

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0CR <BR0S3:0> / BR0ADD <BR0K3:0>	0000 (N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Don't use	Don't use	Divide by N
0001 (K = 1) to 1111 (K = 15)	Don't use	Divide by $N + \frac{(16-K)}{16}$	

Note1: Availability of + (16 – K)/16 division function

N	UART mode	I/O mode
2 to15	Allowed	Not allowed
1, 16	Not allowed	Not allowed

The baud rate generator can be set to "1" in UART mode only when the +(16-K)/16 division function is not used. Do not use in I/O interface mode.

Note2: Set Br0cr<BR0ADDE> to 1 after setting K (K = 1 to 15) to BR0ADD<BR0K3:0> when +(16-K)/16 division function is used. If the unused bits in the BR0ADD register is written, it does not affect operation. If that bits is read, it becomes undefined.

Figure 3.10.11 Baud Rate Generator Control Register

	7	6	5	4	3	2	1	0	
SC0BUF (0200H) SC1BUF (0208H) SC2BUF (0210H)	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(for transmit)
SC3BUF (0218H)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(for receive)

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

Figure 3.10.12 Serial Transmit/Receive Buffer Register

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

SC0MOD1 (0205H)
SC1MOD1 (020DH)
SC2MOD1 (0215H)
SC3MOD1 (021DH)

Figure 3.10.13 Serial Mode Control Register 1

3.10.4 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 TMP91CW40 transmits or receives data to and from an external device, such as a shift register.

Mode 0 uses a synchronization clock (SCLK), which can be configured for either output mode in which the SCLK clock is driven out from the TMP91CW40 or input mode in which the SCLK clock is supplied externally.

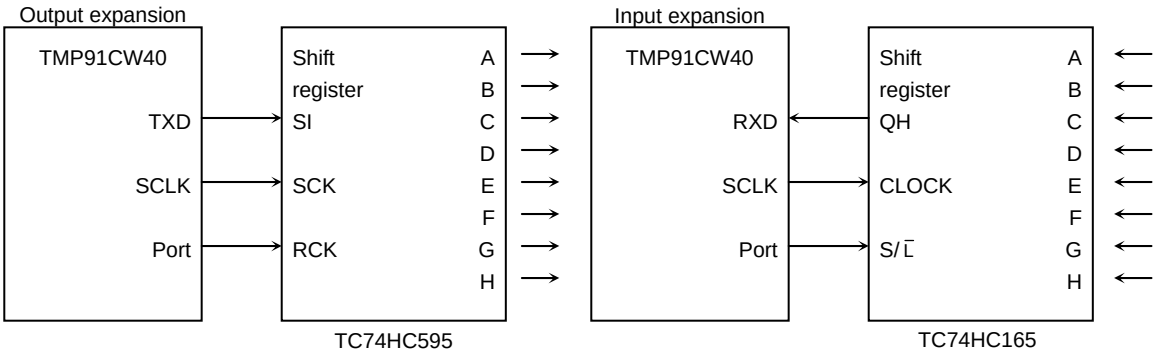


Figure 3.10.14 Example Connection in SCLK Output Mode

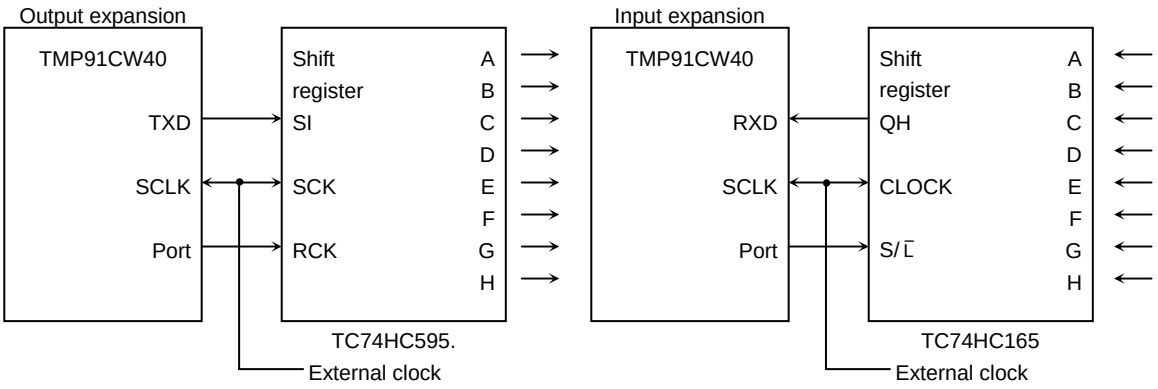


Figure 3.10.15 Example Connection in SCLK Input Mode

a. Transmit operations

In SCLK output mode, each time the CPU writes a character to the transmit buffer, the eight bits of the character are shifted out on the TXD0 pin and the synchronization clock is driven out from the SCLK pin. When all the bits have been shifted out, the INTES0<ITX0C> is set and the transmit-done interrupt (INTTX0) is generated.

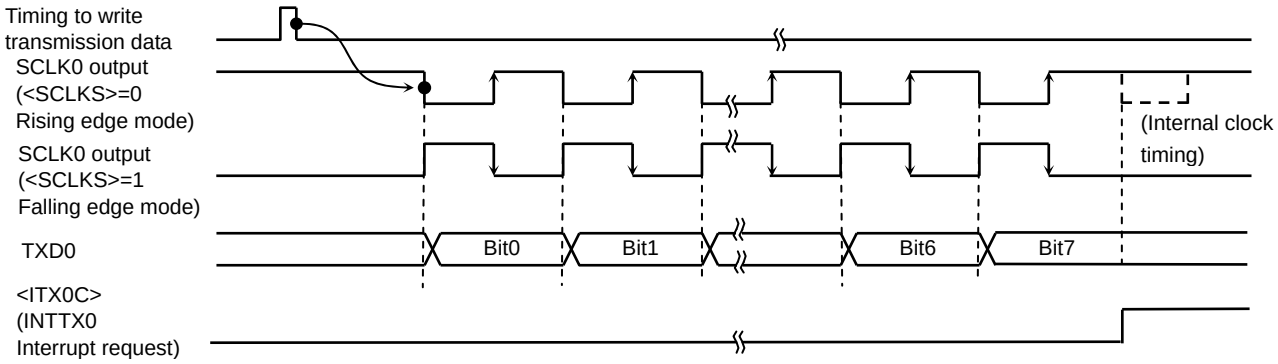


Figure 3.10.16 Transmit Operation in I/O Interface Mode (SCLK output mode)

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 TXD0 pin, synchronous to the programmed edge of the SCLK0 input. When all the bits have been shifted out, the INTES0<ITX0C> is set and the transmit-done interrupt (INTTX0) is generated.

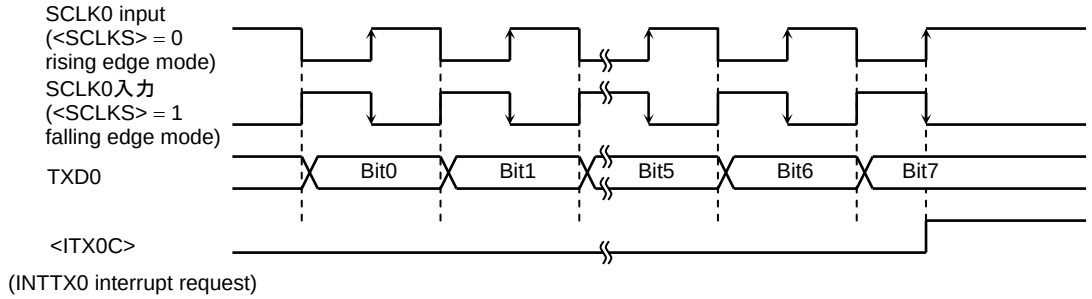


Figure 3.10.17 Transmit Operation in I/O Interface Mode (SCLK0 input mode)

b. Receive operations

In SCLK output mode, each time the CPU picks up a character in receive buffer 2 clearing the receive-done interrupt flag (INTES0<IRX0C>), 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 (SC0BUF), and the INTES0<IRX0C> flag is set to 1, generating the INTRX0 interrupt.

The SCLK output is initiated by setting the SC0MOD0<RXE> bit to 1.

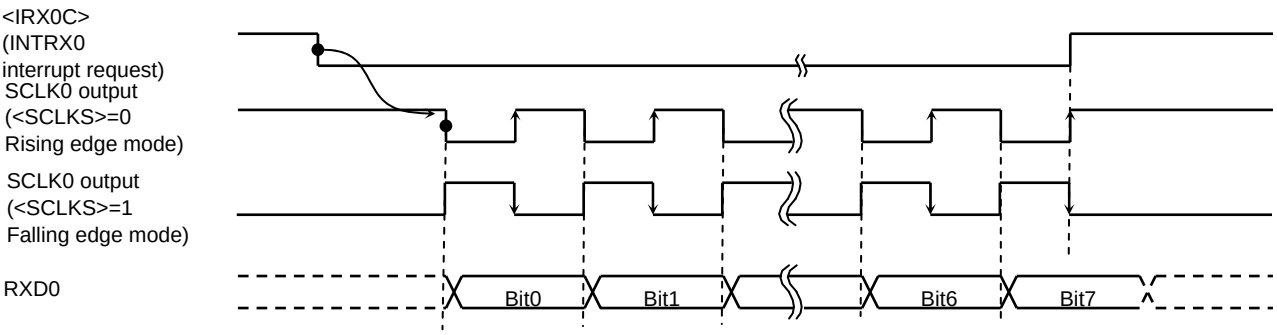


Figure 3.10.18 Receive Operation in I/O Interface Mode (SCLK output mode)

In SCLK input mode, the CPU must pick up a character in receive buffer 2, clearing the receive-done interrupt flag (INTES0<IRX0C>), 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 (SC0BUF), and the INTES0<IRX0C> flag is set to 1 again, generating the INTRX0 interrupt.

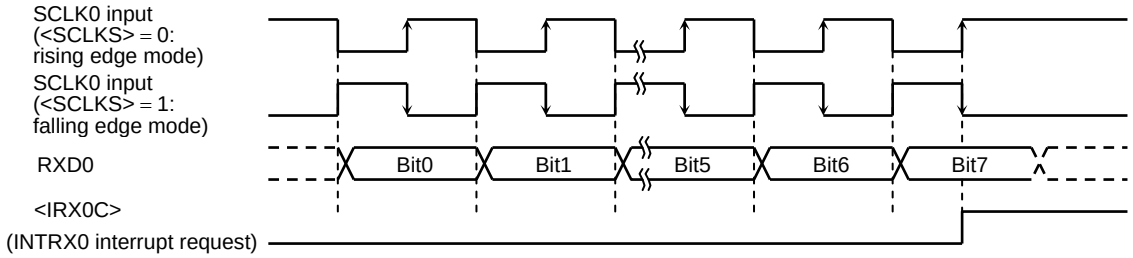


Figure 3.10.19 Receive Operation in I/O Interface Mode (SCLK0 input mode)

Note: Regardless of whether SCLK is in input mode or output mode, the receiver must be enabled by setting the SC0MOD0<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 service routine, receive operation must be performed before loading the transmit buffer with a character, as shown below.

Example: Channel 0
SCLK output mode
Transfer rate: 9600 bps
fc = 14.7456 MHz

* Clocking conditions System clock: High-speed (fc)

Settings in the main routine

	7	6	5	4	3	2	1	0	
INTES0	X	0	0	1	X	0	0	0	Set the transmit interrupt level and disable receive interrupts.
P9CR	–	–	–	–	–	1	0	1	Configure the P90 pin as TXD0 and the P92 pin as SCLK0.
P9FC	–	–	–	–	–	1	–	1	
SC0MOD0	0	0	0	0	0	0	0	0	Select I/O interface mode.
SC0MOD1	1	1	X	X	X	X	X	X	Select full-duplex mode.
SC0CR	0	0	0	0	0	0	0	0	Select SCLK output mode, receiving at the rising edge and transmitting at the rising edge.
BR0CR	0	0	1	1	0	0	1	1	Set the transfer rate to 9600 bps.
SC0MOD0	0	0	1	0	0	0	0	0	Enable receive operation.
SC0BUF	*	*	*	*	*	*	*	*	Load the transmit buffer with transmit data.

Transmit interrupt service routine

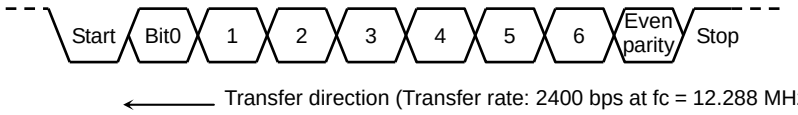
Acc ← SC0BUF									Read received data.
SC0BUF	*	*	*	*	*	*	*	*	Load the transmit buffer with transmit data.

X: Don't care、–: No change

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

Setting the SC0MOD0<SM1:0> field to 01 puts the SIO0 in 7-bit UART mode. In this mode, 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 by programming the SC0CR<PE> bit. When <PE> is set to 1 to enable parity, the SC0CR<EVEN> bit selects even or odd parity.

Example: Transmitting data with even parity in 7-bit UART mode



Transfer direction (Transfer rate: 2400 bps at fc = 12.288 MHz)

* Clock conditions System clock: High-speed (fc)

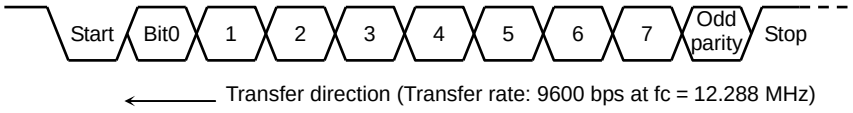
	7	6	5	4	3	2	1	0	
P9CR ←	–	–	–	–	–	–	–	1	Configure the P90 pin as TXD0.
P9FC ←	–	–	–	–	–	–	–	1	
SC0MOD0 ←	X	0	–	X	0	1	0	1	Select 7-bit UART mode.
SC0CR ←	X	1	1	X	X	X	0	0	Select even parity.
BR0CR ←	0	0	1	0	0	1	0	1	Set the transfer rate to 2400 bps.
INTES0 ←	X	1	0	0	–	–	–	–	Enable the INTTX0 interrupt and set its interrupt level to 4.
SC0BUF ←	*	*	*	*	*	*	*	*	Load the transmit buffer with transmit data.

X: Don't care、–: No change

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

Setting the SC0MOD0<SM1:0> field to 10 puts the SIO0 in 8-bit UART mode. In this mode, 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 by programming the SC0CR<PE> bit. When <PE> is set to 1 to enable parity, the SC0CR<EVEN> bit selects even or odd parity.

Example: Receiving data with odd parity in 8-bit UART mode



* Clock conditions System clock: High-speed (fc)

Settings in the main routine

		7	6	5	4	3	2	1	0	
P9CR	←	–	–	–	–	–	–	0	–	Configure the P91(RXD0) pin as an input.
SC0MOD0	←	–	0	1	X	1	0	0	1	Select 8-bit UART mode and enable the receiver.
SC0CR	←	X	0	1	X	X	X	0	0	Select odd parity.
BR0CR	←	0	0	0	1	0	1	0	1	Set the transfer rate to 9600 bps.
INTES0	←	–	–	–	–	X	1	0	0	Enable the INTRX0 interrupt and set its interrupt level to 4.

Example of interrupt routine processing

```
Acc ← SC0CR AND 00011100        Check for errors.
if Acc ← 0 then ERROR
Acc ← SC0BUF                    Read received data.
```

X: Don't care、–: No change

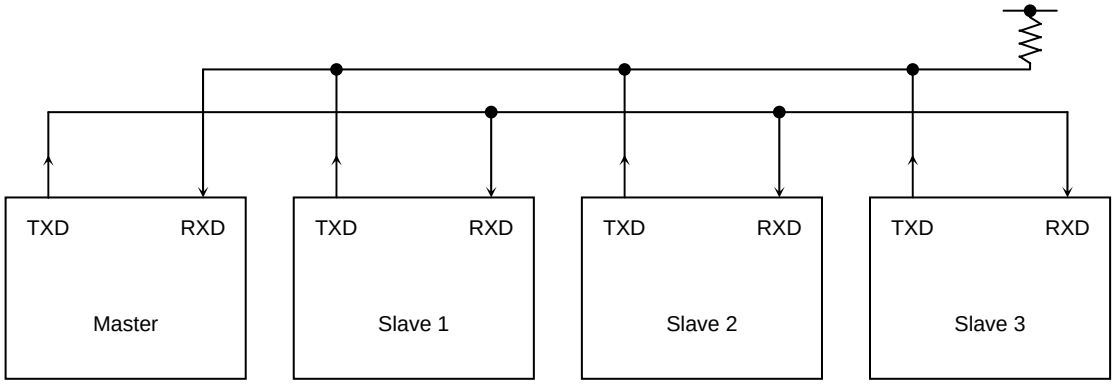
(4) Mode 3 (9-bit UART)

Setting the SC0MOD0<SM1:0> field to 11 puts the SIO0 in 9-bit UART mode. In this mode, no parity bit can be added.

The most-significant bit (9th bit) is stored in the SC0MOD0<TB8> bit in transmit operations and in the SC0CR<RB8> bit in receive operations. Transmit and receive data must be read and written with the most-significant bit first, followed by the SC0BUF.

Wakeup feature

In 9-bit UART mode, the receiver wakeup feature allows the slave controller in a multidrop system to wake up whenever an address character is received. Setting the SC0MOD0<WU> bit to 1 enables the wakeup feature. When the SC0CR<RB8> bit has received an address/data flag bit set to 1, the receiver generates the INTRX0 interrupt.

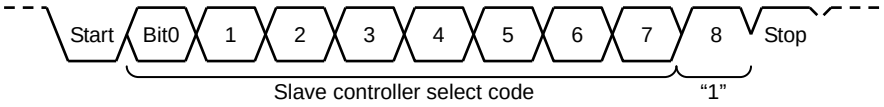


Note: The slave controller's TXD pin must be configured as an open-drain output by programming the ODE register.

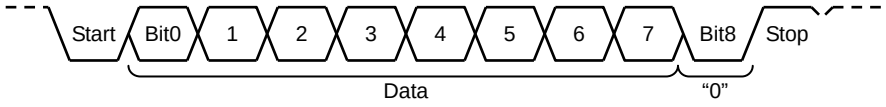
Figure 3.10.20 Serial Link Using the Wakeup Function

Protocol

1. Put all the master and salve controllers in 9-bit UART mode.
2. Enable the receiver in each slave controller by setting the SC0MOD0 <WU> bit to 1.
3. The master controller transmits an address character (i.e., select code) that identifies a slave controller. The address character has the most-significant bit (bit 8) set to 1.

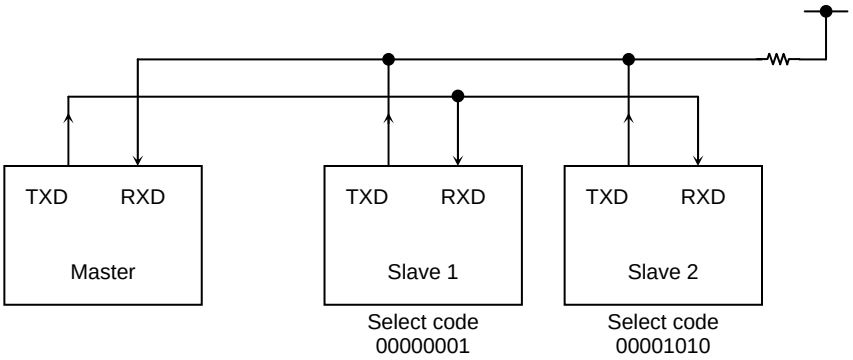


4. Each slave controller compares the received select code to its own select code and clears the <WU> bit if they match.
5. The master controller transmits data character to the selected slave controller (with the SC1MOD0<WU> bit cleared). Data characters have the most-significant bit (bit 8) cleared to 0.



6. Slave controllers not addressed (with <WU> = 1) 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 <WU> = 0) can transmit data to the master controller to notify that it has successfully received the message.

Example: Connecting a master controller and two slave controllers through a serial link using the system clock (f_{SYS}) as a serial clock



- Master controller settings

Main routine

P9CR	← – – – – – – 0 1	Configure the P90 pin as TXD0 and the P91 pin as RXD0.
P9FC	← – – – – – – X 1	
INTES0	← X 1 0 0 X 1 0 1	Enable INTTX0 and set its interrupt level to 4. Enable INTRX0 and set its interrupt level to 5.
SC0MOD0	← 1 0 1 0 1 1 1 0	Select 9-bit UART mode and select f_{SYS} as a serial clock.
SC0BUF	← 0 0 0 0 0 0 0 1	Load the select code for slave 1.

Interrupt routine (INTTX0)

SC0MOD0	← 0 – – – – – – –	Set the <TB8> bit to 0 .
SC0BUF	← * * * * * * * *	Loads the transmit data.

- Salve controller settings

Main routine

P9CR	← – – – – – – 0 1	Set the P90 pin as TXD0 (open-drain output) and the P91 pin as RXD0.
P9FC	← – – – – – – X 1	
ODE	← – – – 1 – – – –	
INTES0	← X 1 0 1 X 1 1 0	Enable INTTX0 and INTRX0.
SC0MOD0	← 0 0 1 1 1 1 1 0	Select 9-bit UART mode, select f_{SYS} as a serial clock, and set the <WU>bit to 1.

Interrupt routine (INTRX0)

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

3.11 LCD Driver

The TMP91CW40 contains a driver and a control circuit for directly driving a liquid crystal display (LCD). The LCD is connected using the following pins:

a. Segment output pins : 8 pins (SEG7 to SEG0)
b. Segment output/port (P0, P1, P2, PB) multiplexed pins : 32 pins (SEG39 to SEG8)
c. Common output pins : 4 pins (COM3 to COM0)

The C0, C1, V1, V2 and V3 pins are also available for the voltage reducer in the LCD driver. The LCD driver can directly drive the following four types of LCDs:

a. 1/4 duty (1/3 bias) LCD: up to 160 pixels (8 segments x 20 commons)
b. 1/3 duty (1/3 bias) LCD: up to 120 pixels (8 segments x 15 commons)
c. 1/2 duty (1/2 bias) LCD: up to 80 pixels (8 segments x 10 commons)
d. Static LCD: up to 40 pixles (8 segments x 5 commons)

3.11.1 Configuration

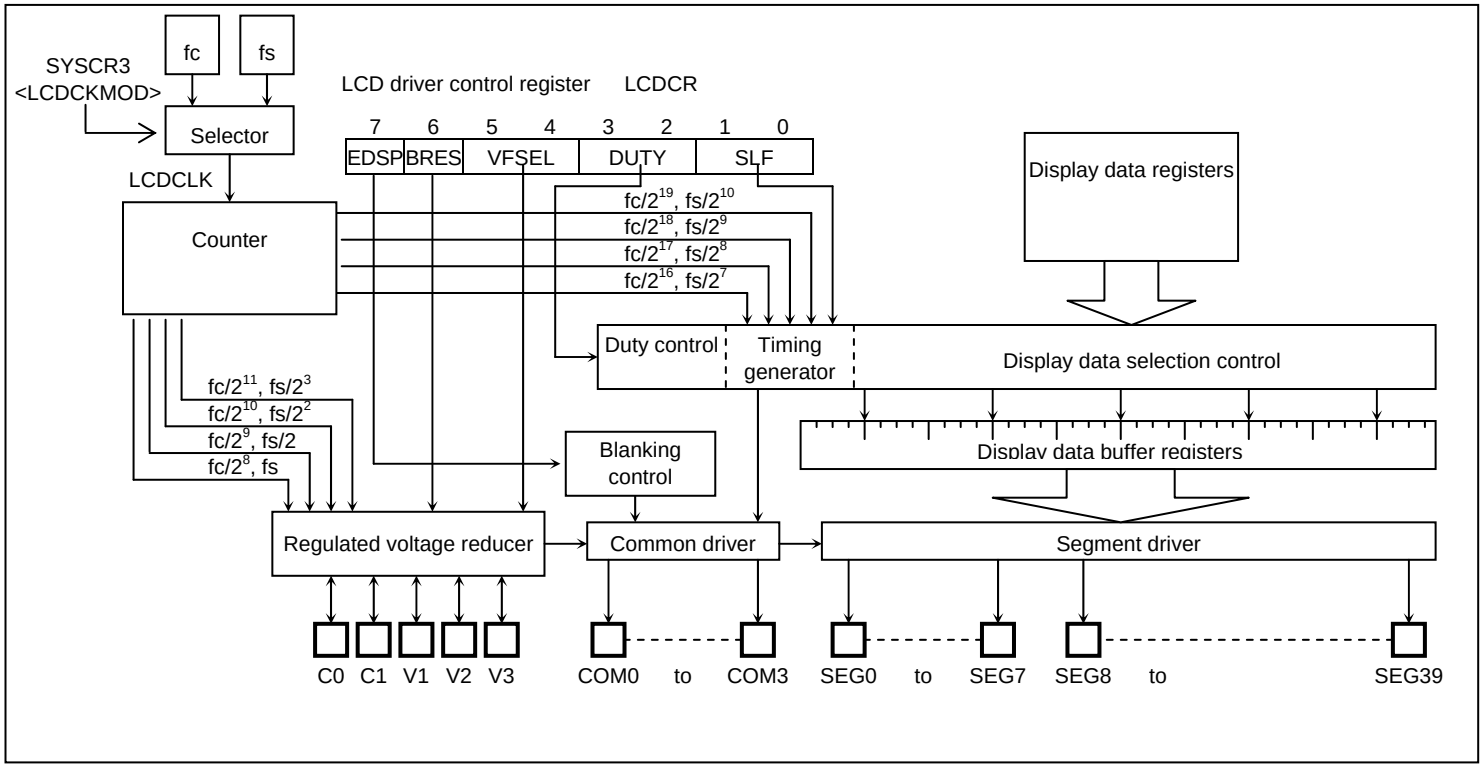


Figure 3.11.1 LCD Driver

3.11.2 Control

The LCD driver is controlled by the LCD control register (LCDCR). The <EDSP> bit in the LCDCR is used to enable LCD display.

LCDCR (03D0H)

7	6	5	4	3	2	1	0
EDSP	BRES	VFSEL		DUTY		SLF	

(Initial value: 0000 0000)

SLF	Base frequency [Hz]		SYSCR1<SYSCK>=0		SYSCR1 <SYSCK>=1	R/W
			SYSCR3 <LCDCKMOD>=0	SYSCR3 <LCDCKMOD>=1	SYSCR3 <LCDCKMOD>=1	
		00	$fc/2^{19}$	$fs/2^{10}$	$fs/2^{10}$	
		01	$fc/2^{18}$	$fs/2^{9}$	$fs/2^{9}$	
		10	$fc/2^{17}$	$fs/2^{8}$	$fs/2^{8}$	
		11	$fc/2^{16}$	$fs/2^{7}$	$fs/2^{7}$	
DUTY	LCD drive method	00: 1/4 duty (1/3 bias) 01: 1/3 duty (1/3 bias) 10: 1/2 duty (1/2 bias) 11: Static				
VFSEL	Voltage reducer frequency [Hz]		SYSCR1<SYSCK>=0		SYSCR1 <SYSCK>=1	
			SYSCR3 <LCDCKMOD>=0	SYSCR3 <LCDCKMOD>=1	SYSCR3 <LCDCKMOD>=1	
		00	$fc/2^{11}$	$fs/2^{3}$	$fs/2^{3}$	
		01	$fc/2^{10}$	$fs/2^{2}$	$fs/2^{2}$	
		10	$fc/2^{9}$	fs/2	fs/2	
		11	$fc/2^{8}$	fs	fs	
BRES	Voltage reducer enable/disable	0: Disable (Use external divider resistors) 1: Enable				
EDSP	LCD display control	0: Disable 1: Enable				

Note 1: When <BRES>=0, $V_{DD} \geq V_3 \geq V_2 \geq V_1 \geq V_{SS}$ must be satisfied. When <BRES>=1, $V_{DD} = V_3$ must be satisfied. Ignoring these conditions may not only affect the quality of LCD display but also damage the device due to overcurrent that flows through ports.

Note 2: The <SLF1:0> and <DUTY1:0> fields should be set when <EDSP>=0. (Nor is it allowed to set these fields with the same instruction that sets <EDSP> to 0.) Otherwise, the expected duty cannot be obtained and the LCD cannot be displayed properly.

Note 3: The reference clock (LCDCLK) for the base frequency of the LCD driver is independent of the system clock and can be switched between low-frequency (fs) and high-frequency (fc) by the programming of the SYSCR3 <LCDCKMOD> bit. For proper operation of the LCD, be careful about the following points:

- Before changing LCDCLK to low-frequency (fs), start up the low-frequency oscillator (fs) by programming SYSCR0<XTEN> and make sure that the warming-up period has completed by checking SYSCR0<WUEFL>.
- It is not allowed to set the system clock to low-frequency (fs) and LCDCLK to high-frequency (fc).
- Before changing the system clock from high-frequency (fc) to low-frequency (fs), make sure to set LCDCLK to low-frequency (fs).

When the low-frequency (fs) clock is used for the system clock, it is recommended to set LCDCLK to low-frequency (fs) in the application's startup routine and to always use the low-frequency (fs) clock for LCDCLK.

Note 4: To change the SYSCR3<LCDCKMOD> bit, the <EDSP> bit must be 0.

Note 5: When the device enters STOP mode, the <EDSP> bit is automatically cleared to 0. If HALT mode is activated immediately after display data is written to the LCDREG, the LCD is not displayed correctly and the device immediately enters HALT mode.
After exiting STOP mode, do not make any settings for the LCD driver until the warming up of fs has completed. This can be checked by the SYSCR0 register.

Note 6 When used as LCD pins, COM output pins output low level and SEG output pins are placed in a high-impednce state when the <EDSP> bit is cleared to 0 (except when SEG output pins are used as ports). When LCDCR2<MSEG07>=1, SEG0 to SEG7 pins output low level.

Figure 3.11.2 LCD Driver Control Register (1)

LCDCR2 (03DEH)	7	6	5	4	3	2	1	0	(Initial value: 0000 0000)
	"0"	"0"	MSEG07	"0"	"0"	"0"	"0"	"0"	

			LCDCR<EDSP>=0	LCDCR<EDSP>=1	
MSEG07	SEG0 to SEG7 pins output control	0	High impedance	Display data output	W
		1	Low output	Don't use	

Note 1: Do not set the <MSEG07> bit to 1 when the LCDCR<EDSP> bit is 1.

Note 2: Except for bit 5, all the bits of this register must always be written as 0.

Note 3: This register does not support read-modify-write operation.

Figure 3.11.3 LCD Driver Control Register (2)

(1) LCD drive method

The LCD drive method can be selected from four types by the programming of the <DUTY> field in the LCDCR. The LCD drive method should be set in the initialization program according to the LCD to be used.

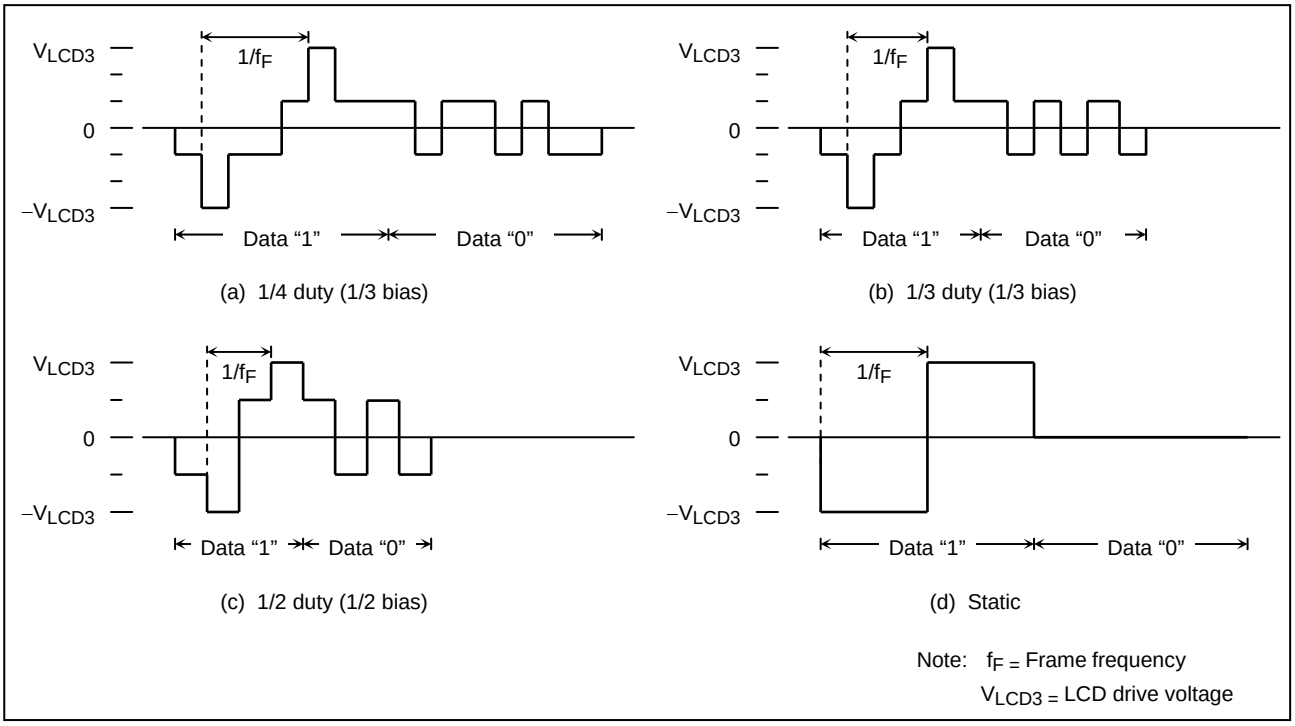


Figure 3.11.4 LCD Drive Waveforms (Potential Differences between COM and SEG Pins)

(2) Frame frequency

The frame frequency (f_F) is determined according to the LCD drive method and base frequency, as shown in Table 3.11.1.

The base frequency is selected by the LCDCR<SLF> field according to the basic clock frequencies fc and fs to be used.

Table 3.11.1 Frame Frequency Settings

a. SYSCR3<LCDCKMOD> = 0

SLF	Base Frequency [Hz]	Frame Frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$fc/2^{19}$	$fc/2^{19}$	$4/3 \times fc/2^{19}$	$4/2 \times fc/2^{19}$	$fc/2^{19}$
	(fc = 24 MHz)	46	61	92	46
	(fc = 16 MHz)	31	41	61	31
	(fc = 8 MHz)	15	20	31	15
01	$fc/2^{18}$	$fc/2^{18}$	$4/3 \times fc/2^{18}$	$4/2 \times fc/2^{18}$	$fc/2^{18}$
	(fc = 24 MHz)	92	122	183	92
	(fc = 16 MHz)	61	81	122	61
	(fc = 8 MHz)	31	41	61	31
10	$fc/2^{17}$	$fc/2^{17}$	$4/3 \times fc/2^{17}$	$4/2 \times fc/2^{17}$	$fc/2^{17}$
	(fc = 24 MHz)	183	244	366	183
	(fc = 16 MHz)	122	163	244	122
	(fc = 8 MHz)	61	81	122	61
11	$fc/2^{16}$	$fc/2^{16}$	$4/3 \times fc/2^{16}$	$4/2 \times fc/2^{16}$	$fc/2^{16}$
	(fc = 24 MHz)	366	488	732	366
	(fc = 16 MHz)	244	326	488	244
	(fc = 8 MHz)	122	163	244	122

Note: fc = High-frequency clock frequency [Hz]

b. SYSCR3<LCDCKMOD> = 1

SLF	Base Frequency [Hz]	Frame Frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$fs/2^{10}$	$fs/2^{10}$	$4/3 \times fs/2^{10}$	$4/2 \times fs/2^{10}$	$fs/2^{10}$
	(fs = 32.768 kHz)	32	43	64	32
01	$fs/2^{9}$	$fs/2^{9}$	$4/3 \times fs/2^{9}$	$4/2 \times fs/2^{9}$	$fs/2^{9}$
	(fs = 32.768 kHz)	64	85	128	64
10	$fs/2^{8}$	$fs/2^{8}$	$4/3 \times fs/2^{8}$	$4/2 \times fs/2^{8}$	$fs/2^{8}$
	(fs = 32.768 kHz)	128	171	256	128
11	$fs/2^{7}$	$fs/2^{7}$	$4/3 \times fs/2^{7}$	$4/2 \times fs/2^{7}$	$fs/2^{7}$
	(fs = 32.768 kHz)	256	341	512	256

Note: fs = Low-frequency clock frequency [Hz]

(3) LCD drive power supply

To obtain the LCD drive power supply, the TMP91CW40 can use either the voltage reducer incorporated in the LCD driver that reduces the external reference voltage, or external divider resistors that divide the exernal reference voltage. This selection is made in the <BRES> field in the LCD control register (LCDCR).

When the voltage reducer is used, the reference voltage connected to the V3 pin is reduced to two-thirds (2/3) or one-third (1/3) to generate the output voltage for segment/common signals.

When external divider resistors are used, the external power supply is divided by external resistors and the divided voltages are input to the V1, V2 and V3 pins to generate the output voltage for segment/common signals.

The voltage reducer only supports 1/3 bias.

The base frequency for the voltage reducer is selected by the <VFSEL> field in the LCDCR. The segment/common drive capability can be increased by selecting a higher frequency.

Table 3.11.2 shows the current carrying capacities of the V3 pin according to the selected base frequency for the voltage reducer.

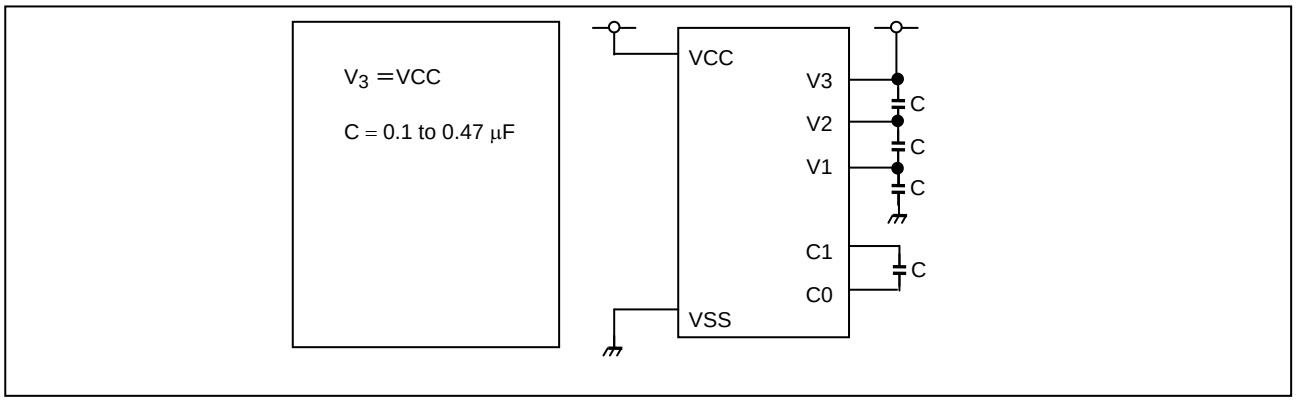


Figure 3.11.5 Example of LCD Power Supply Connection when Using the Voltage Reducer (LCDCR<BRES> = 1)

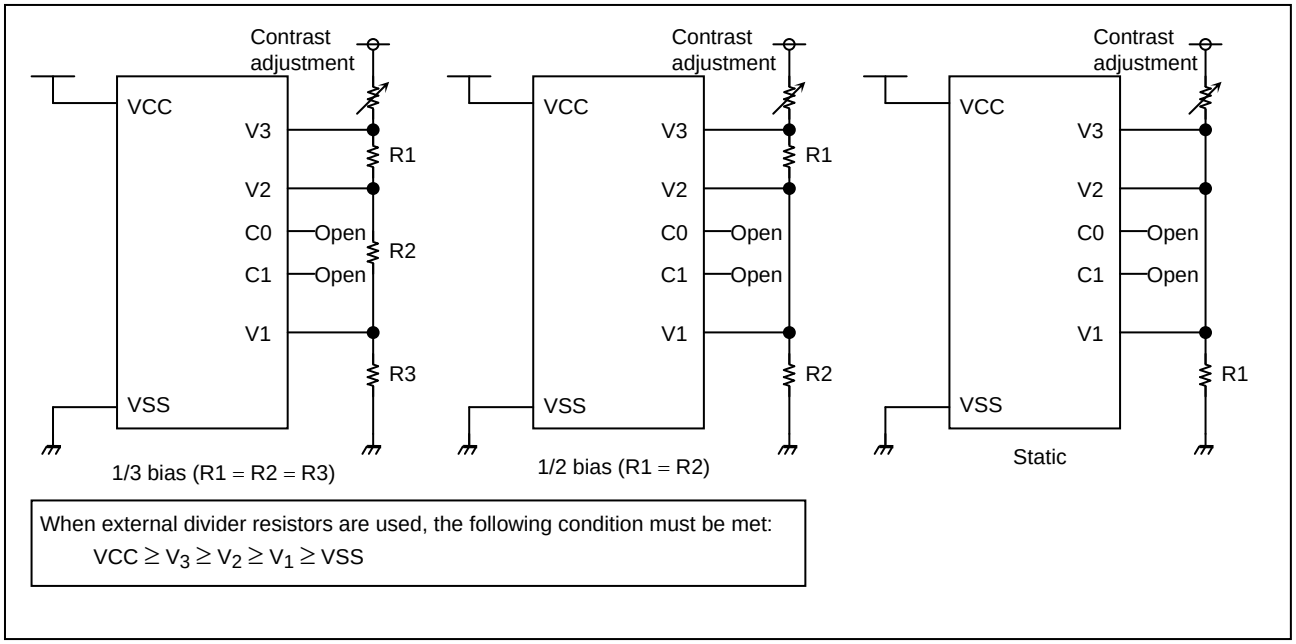


Figure 3.11.6 Example of LCD Power Supply Connection when Using External Divider Resistors (LCDCR<BRES> = 0)

Table 3.11.2 Current Carrying Capacities of the V2 Pin according to the Voltage Reducer Frequency (typ.)

VCC=V3=3.0V, Ta=25°C

LCDCR <VFSEL>	Voltage Reducer Frequency	fc = 24 MHz	fc = 16 MHz	fc = 8 MHz	fs = 32.768 kHz
00	$fc/2^{11}$ or $fs/2^3$	−1.63 mV / μA	−1.82 mV / μA	−2.63 mV / μA	−3.77 mV / μA
01	$fc/2^{10}$ or $fs/2^2$	−1.04 mV / μA	−1.12 mV / μA	−1.42 mV / μA	−1.60 mV / μA
10	$fc/2^9$ or fs/2	−0.97 mV / μA	−1.06 mV / μA	−1.07 mV / μA	−1.16 mV / μA
11	$fc/2^8$ or fs	−0.90 mV / μA	−0.90 mV / μA	−0.90 mV / μA	−0.97 mV / μA

Note 1: The current carrying capacity indicates the amount of voltage that drops per 1 μA.

Note 2: The base frequency for the voltage reducer should be selected according to the LCD panel to be used.

Table 3.11.3 Current Carrying Capacities of the V1 Pin according to the Voltage Reducer Frequency (typ.)

VCC=V3=3.0V, Ta=25°C

LCDCR <VFSEL>	Voltage Reducer Frequency	fc = 24 MHz	fc = 16 MHz	fc = 8 MHz	fs = 32.768 kHz
00	$fc/2^{11}$ or $fs/2^3$	−0.57 mV / μA	−0.70 mV / μA	−1.17 mV / μA	−1.01 mV / μA
01	$fc/2^{10}$ or $fs/2^2$	−0.55 mV / μA	−0.60 mV / μA	−0.78 mV / μA	−0.72 mV / μA
10	$fc/2^9$ or fs/2	−0.53 mV / μA	−0.54 mV / μA	−0.61 mV / μA	−0.57 mV / μA
11	$fc/2^8$ or fs	−0.52 mV / μA	−0.53 mV / μA	−0.56 mV / μA	−0.54 mV / μA

Note 1: The current carrying capacity indicates the amount of voltage that drops per 1 μA.

Note 2: The base frequency for the voltage reducer should be selected according to the LCD panel to be used.

3.11.3 LCD Display Operation

(1) Display data setting

The display data stored in the display data area is automatically read and sent to the LCD driver by hardware. The LCD driver generates segment and common signals according to the received display data and the specified drive method. Therefore, it is only required to program the contents of the display data area to change display patterns. After display data is written to the LCDREG0 to LCDREG19, a wait period of six LCDCLK pulses is needed before new display data can be written. If new display data is written without waiting for this interval, the previous display data may be overwritten.

Figure 3.11.7 shows the correspondence between the display data area and the SEG and COM pins. The LCD light is turned on when display data is 1 and turned off when display data is 0.

The number of pixels that can be driven varies with the LCD drive method, so the number of bits to be used in the display data area also varies.

Note: The contents of the display data area are initialized to 00H after reset.

	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
LCDREG0	03E0H	SEG1				SEG0				
LCDREG1	03E1H	SEG3				SEG2				
LCDREG2	03E2H	SEG5				SEG4				
LCDREG3	03E3H	SEG7				SEG6				
LCDREG4	03E4H	SEG9				SEG8				
LCDREG5	03E5H	SEG11				SEG10				
LCDREG6	03E6H	SEG13				SEG12				
LCDREG7	03E7H	SEG15				SEG14				
LCDREG8	03E8H	SEG17				SEG16				
LCDREG9	03E9H	SEG19				SEG18				Initial value: 00H
LCDREG10	03F0H	SEG21				SEG20				
LCDREG11	03F1H	SEG23				SEG22				
LCDREG12	03F2H	SEG25				SEG24				
LCDREG13	03F3H	SEG27				SEG26				
LCDREG14	03F4H	SEG29				SEG28				
LCDREG15	03F5H	SEG31				SEG30				
LCDREG16	03F6H	SEG33				SEG32				
LCDREG17	03F7H	SEG35				SEG34				
LCDREG18	03F8H	SEG37				SEG36				
LCDREG19	03F9H	SEG39				SEG38				
		COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0	

Figure 3.11.7 LCD Display Data

Table 3.11.4 Bits Used for Storing Display Data

Drive Method	Bits 7/3	Bits 6/2	Bits 5/1	Bits 4/0
1/4 duty	COM3	COM2	COM1	COM0
1/3 duty	–	COM2	COM1	COM0
1/2 duty	–	–	COM1	COM0
Static	–	–	–	COM0

Note: "–" indicates bits that are not used for storing display data.

(2) Blanking

When the <EDSP> bit in the LCDCR is cleared to 0, the COM pins are driven to GND level and the SEG pins are placed in a high-impedance state.

When the TMP91CW40 enters STOP mode, the <EDSP> bit is cleared to 0. After STOP mode is exited, the <EDSP> bit need be set to 1 to display data on the LCD again.

The following shows a programming example for fixing the SEG pins to low level.

```
LD    (P2CR),0FFH
LD    (P1CR),0FFH
LD    (P0CR),0FFH
LD    (PBCR),0FFH
LD    (P2),00H
LD    (P1),00H
LD    (P0),00H
LD    (PB),00H
LD    (LCDSW1),00H
LD    (LCDSW2),00H
LD    (LCDSW3),00H
LD    (LCDSW4),00H
LD    (LCDCR),00H
LD    (LCDCR2),20H
```

Configure ports for low output.

Configure port/SEG multiplexed pins as port pins.
The SEG pins become ports fixed to low level.

<EDSP>=0

<MSEG07>=1, setting SEG0 to SEG 7 for low output

Note: During reset, COM outputs are initialized to GND level, but SEG outputs (SEG0 to SEG7) and port/SEG multiplexed pins (P0, P1, P2 and PB ports) are placed in a high-impedance state. Therefore, if a considerably long external reset input occurs, the LCD may not be displayed properly.

3.11.4 LCD Driver Control Method

(1) Initial setting

Figure 3.11.7 shows the flowchart for initializing the LCD driver.

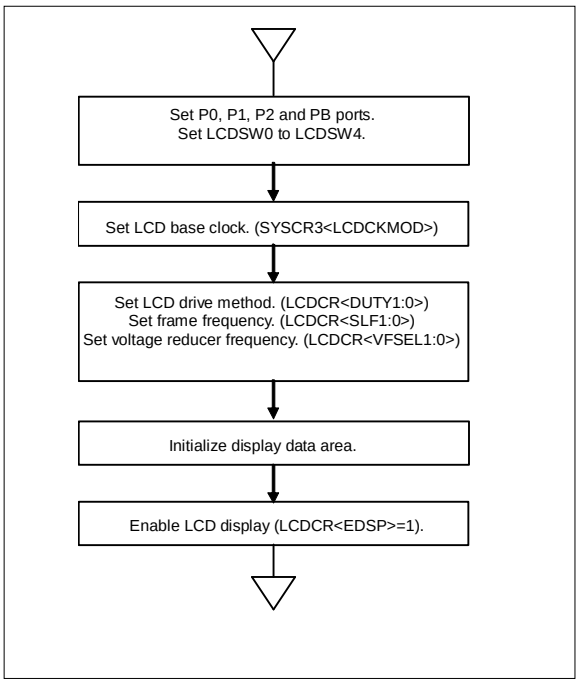


Figure 3.11.7 Initial Setting of LCD Driver

(2) Storing display data

Display data is normally stored in the program memory (ROM) as fixed data, and transferred to the display data area by load instructions.

Example 1: Table 3.11.4 shows the display data for displaying the numbers corresponding to the BCD data stored at address 1400H in RAM using a 1/4 duty LCD, with the COM and SEG pins connected to the LCD as shown in Figure 3.11.8.

```
        LD   XHL,(1400H)
        ADD XHL,TABLE
        LD   B,(XHL)
        LD   (LCDREG0),B
        RET
TABLE:
        DB  11011111B, 00000110B
        DB  11100011B, 10100111B
        DB  00110110B, 10110101B
        DB  11110101B, 00010111B
        DB  11110111B, 10110111B
```

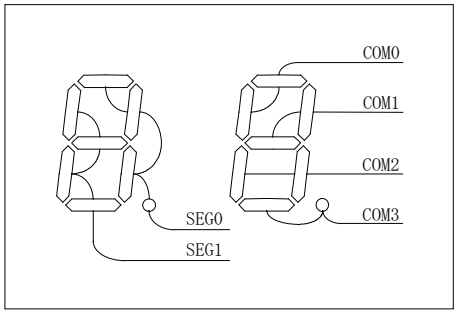
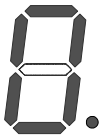
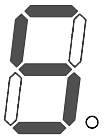
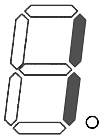
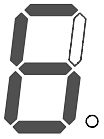
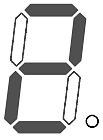
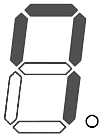
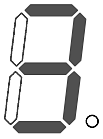
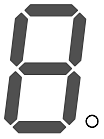
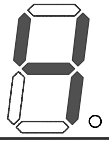
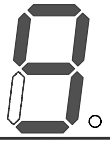


Figure 3.11.8 Example of COM and SEG Pin Connections

Note: DB = Byte data definition instruction

Table 3.11.4 Example of Display Data (1/4 Duty)

No.	Display	Display data	No.	Display	Display data
0.		11011111	5		10110101
1		00000110	6		11110101
2		11100011	7		00010111
3		10100111	8		11110111
4		00110110	9		10110111

Example 2: Table 3.11.5 shows the display data for displaying the numbers shown in Table 3.11.4 using a 1/2 duty LCD, with the COM and SEG pins connected to the LCD as shown in Figure 3.11.9.

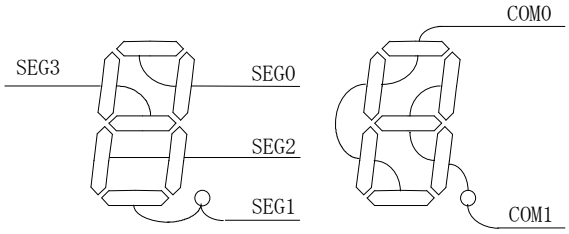


Figure 3.11.9 Example of COM and SEG Pin Connections

Table 3.11.5 Example of Display Data (1/2 Duty)

Number	Display Data		Number	Display Data	
	LCDREG1	LCDREG0		LCDREG1	LCDREG0
0.	**01**11	**11**11	5	**11**10	**01**01
1	**00**10	**00**10	6	**11**11	**01**01
2	**10**01	**01**11	7	**01**10	**00**11
3	**10**10	**01**11	8	**11**11	**01**11
4	**11**10	**00**10	9	**11**10	**01**11

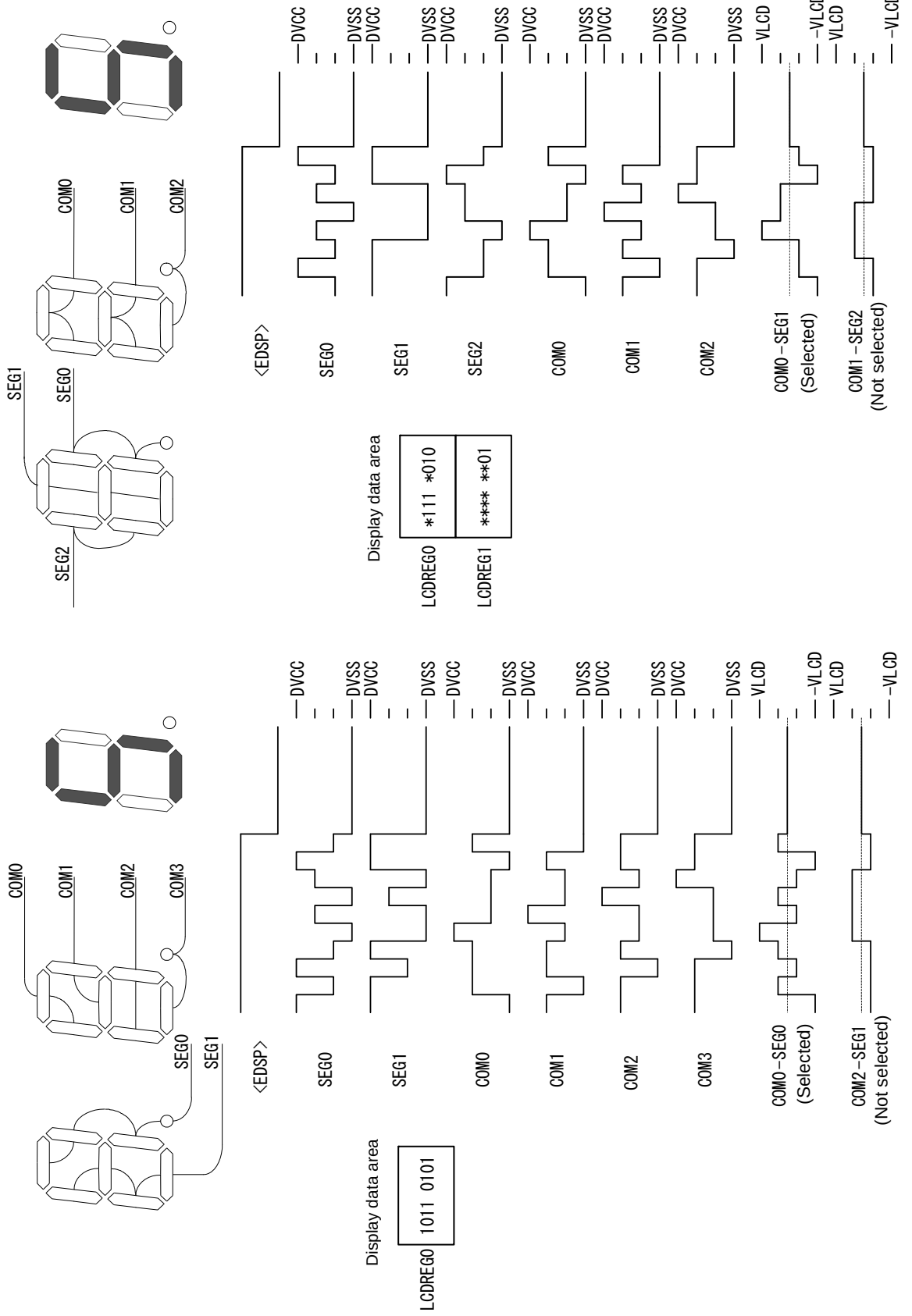


Figure3.11.10 1/4 Duty (1/3 Bias)

Figure 3.11.11 1/3 Duty (1/3 Bias)

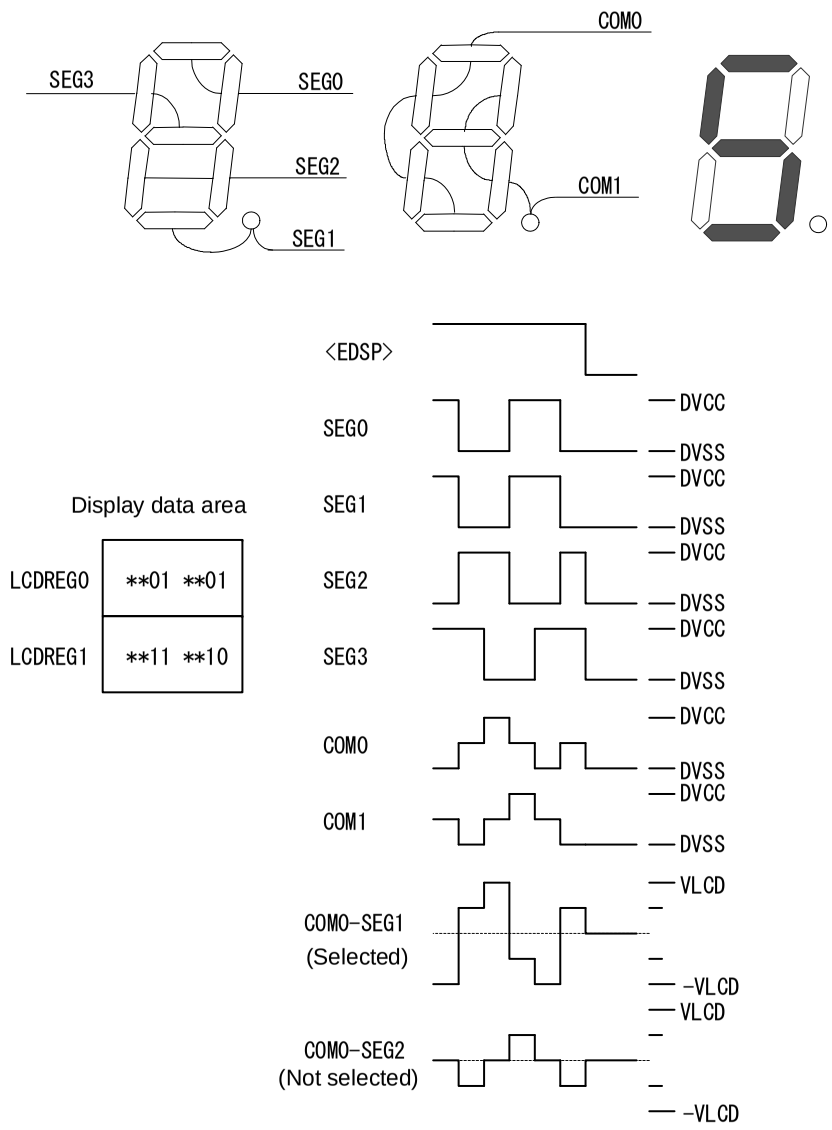


Figure 3.11.12 1/2 Duty (1/2 Bias)

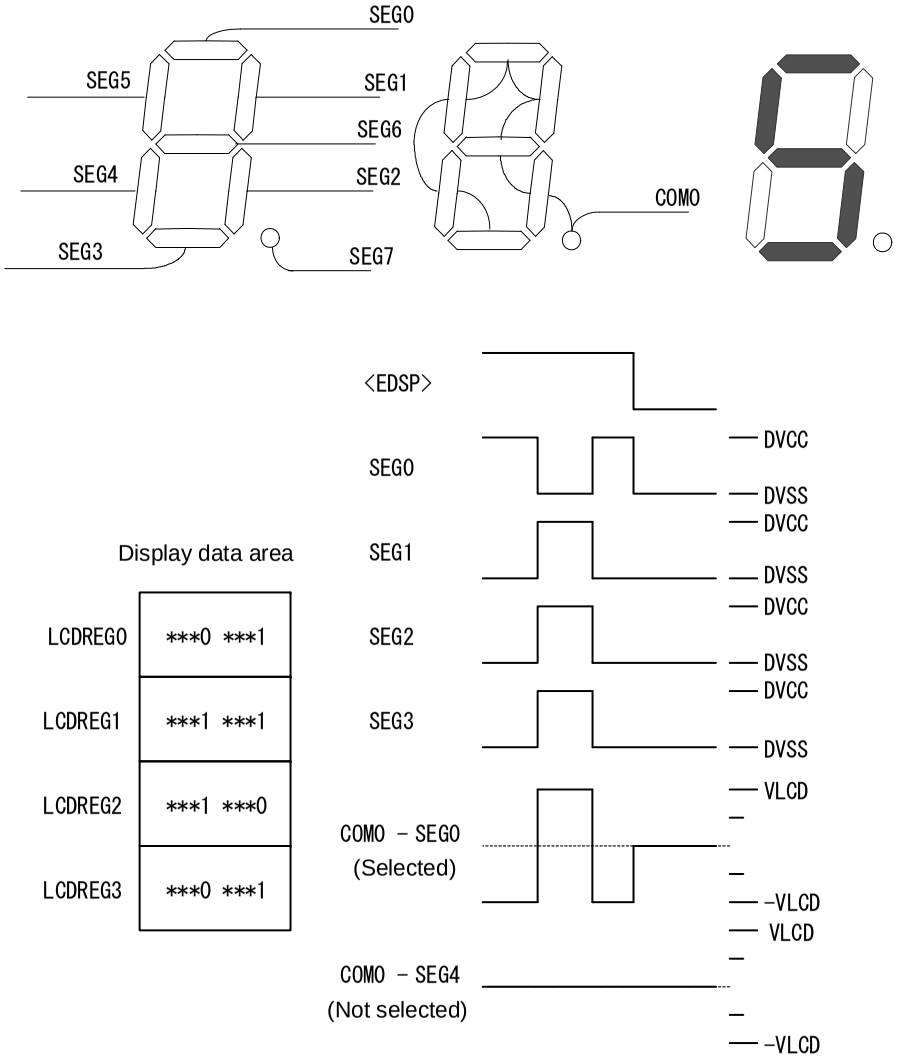


Figure 3.11.13 Static

3.12 Real time clock (RTC)

3.12.1 Function description for RTC

1) Clock function (hour, minute, second)
2) Calendar function (month and day, day of the week, and leap year)
3) 24 or 12-hour (AM/PM) clock function
4) +/− 30 second adjustment function (by software)
5) Alarm function (Alarm output)
6) Alarm interrupt generate

3.12.2 Block diagram

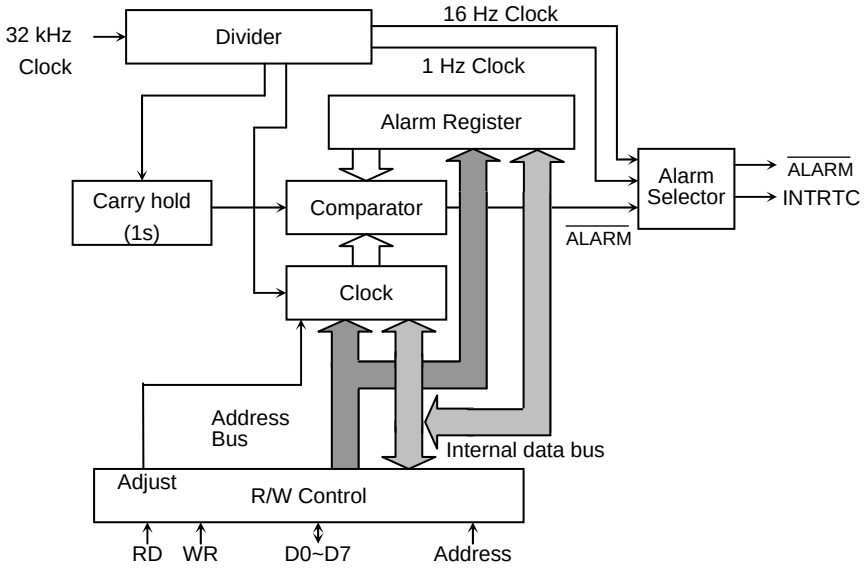


Figure 3.12.1 RTC block diagram

Note 1: Western calendar year column:

This product uses only the final two digits of the year. Therefore, the year following 99 is 00 years. In use, please take into account the first two digits when handling years in the western calendar.

Note 2: Leap year:

A leap year is divisible by 4, but the exception is any leap year which is divisible by 100; this is not considered a leap year. However, any year which is divisible by 400, is a leap year. This product does not take into account the above exceptions . Since this product accounts only for leap years divisible by 4, please adjust the system for any problems.

3.12.3 Control registers

Table 3.12.1 PAGE 0 (Clock function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	0320H		40 sec	20 sec	10 sec	8 sec	4 sec	2 sec	1 sec	Second column	R/W
MINR	0321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	0322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	0323H						W2	W1	W0	Day of the week column	R/W
DATER	0324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	0325H				Oct.	Aug.	Apr.	Feb.	Jan.	Month column	R/W
YEARR	0326H	Year 80	Year 40	Year 20	Year 10	Year 8	Year 4	Year 2	Year 1	Year column (Lower two columns)	R/W
PAGER	0327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	0328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, DATER, MONTHR, YEARR of PAGE0, the current state is read.

Table 3.12.2 PAGE1 (Alarm function) registers

Symbol	Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Function	Read/Write
SECR	0320H										R/W
MINR	0321H		40 min	20 min	10 min	8 min	4 min	2 min	1 min	Minute column	R/W
HOURR	0322H			20 hours/ PM/AM	10 hours	8 hours	4 hours	2 hours	1 hour	Hour column	R/W
DAYR	0323H						W2	W1	W0	Day of the week column	R/W
DATER	0324H			Day 20	Day 10	Day 8	Day 4	Day 2	Day 1	Day column	R/W
MONTHR	0325H								24/12	24-hour clock mode	R/W
YEARR	0326H							LEAP1	LEAP0	Leap-year mode	R/W
PAGER	0327H	Interrupt enable			Adjustment function	Clock enable	Alarm enable		PAGE setting	PAGE register	W, R/W
RESTR	0328H	1Hz enable	16Hz enable	Clock reset	Alarm reset	Always write “0”				Reset register	W only

Note: When reading SECR, MINR, HOURR, DAYR, DATER, MONTHR, YEARR of PAGE1, the current state is read.

3.12.4 Detailed explanation of control register

RTC is not initialized by system reset. Therefore, all registers must be initialized at the beginning of the program.

(1) Second column register (for PAGE0 only)

SECR (0320H)

	7	6	5	4	3	2	1	0
Bit symbol		SE6	SE5	SE4	SE3	SE2	SE1	SE0
Read/Write		R/W						
Reset State		Undefined						
Function	"0" is read.	40 sec. column	20 sec. column	10 sec. column	8 sec. column	4 sec. column	2 sec. column	1 sec. column

0	0	0	0	0	0	0	0 sec
0	0	0	0	0	0	1	1 sec
0	0	0	0	0	1	0	2 sec
0	0	0	0	0	1	1	3 sec
0	0	0	0	1	0	0	4 sec
0	0	0	0	1	0	1	5 sec
0	0	0	0	1	1	0	6 sec
0	0	0	0	1	1	1	7 sec
0	0	0	1	0	0	0	8 sec
0	0	0	1	0	0	1	9 sec
0	0	1	0	0	0	0	10 sec
			:				
0	0	1	1	0	0	1	19 sec
0	1	0	0	0	0	0	20 sec
			:				
0	1	0	1	0	0	1	29 sec
0	1	1	0	0	0	0	30 sec
			:				
0	1	1	1	0	0	1	39 sec
1	0	0	0	0	0	0	40 sec
			:				
1	0	0	1	0	0	1	49 sec
1	0	1	0	0	0	0	50 sec
			:				
1	0	1	1	0	0	1	59 sec

Note: Do not set data other than as shown above.

(2) Minute column register (for PAGE0/1)

MINR (0321H)

	7	6	5	4	3	2	1	0
Bit symbol		MI6	MI5	MI4	MI3	MI2	MI1	MI0
Read/Write		R/W						
Reset State		Undefined						
Function	"0" is read.	40 min, column	20 min, column	10 min, column	8 min, column	4 min, column	2 min, column	1 min, column

0	0	0	0	0	0	0	0 min
0	0	0	0	0	0	1	1 min
0	0	0	0	0	1	0	2 min
0	0	0	0	0	1	1	3 min
0	0	0	0	1	0	0	4 min
0	0	0	0	1	0	1	5 min
0	0	0	0	1	1	0	6 min
0	0	0	0	1	1	1	7 min
0	0	0	1	0	0	0	8 min
0	0	0	1	0	0	1	9 min
0	0	1	0	0	0	0	10 min
			:				
0	0	1	1	0	0	1	19 min
0	1	0	0	0	0	0	20 min
			:				
0	1	0	1	0	0	1	29 min
0	1	1	0	0	0	0	30 min
			:				
0	1	1	1	0	0	1	39 min
1	0	0	0	0	0	0	40 min
			:				
1	0	0	1	0	0	1	49 min
1	0	1	0	0	0	0	50 min
			:				
1	0	1	1	0	0	1	59 min

Note: Do not set data other than as shown above.

(3) Hour column register (for PAGE0/1)

1. In case of 24-hour clock mode (MONTHR<MO0>= "1")

HOURR (0322H)

	7	6	5	4	3	2	1	0
Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
Read/Write			R/W					
Reset State			Undefined					
Function	"0" is read.		20 hour column	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o'clock
0	0	0	0	0	1	1 o'clock
0	0	0	0	1	0	2 o'clock
			:			
0	0	1	0	0	0	8 o'clock
0	0	1	0	0	1	9 o'clock
0	1	0	0	0	0	10 o'clock
			:			
0	1	1	0	0	1	19 o'clock
1	0	0	0	0	0	20 o'clock
			:			
1	0	0	0	1	1	23 o'clock

Note: Do not set data other than as shown above.

2. In case of 12-hour clock mode (MONTHR<MO0>= "0")

HOURR (0322H)

	7	6	5	4	3	2	1	0
Bit symbol			HO5	HO4	HO3	HO2	HO1	HO0
Read/Write			R/W					
Reset State			Undefined					
Function	"0" is read.		PM/AM	10 hour column	8 hour column	4 hour column	2 hour column	1 hour column

0	0	0	0	0	0	0 o'clock (AM)
0	0	0	0	0	1	1 o'clock
0	0	0	0	1	0	2 o'clock
			:			
0	0	1	0	0	1	9 o'clock
0	1	0	0	0	0	10 o'clock
0	1	0	0	0	1	11 o'clock
1	0	0	0	0	0	0 o'clock (PM)
1	0	0	0	0	1	1 o'clock

Note: Do not set data other than as shown above.

(4) Day of the week column register (for PAGE0/1)

DAYR (0323H)

	7	6	5	4	3	2	1	0
Bit symbol						WE2	WE1	WE0
Read/Write						R/W		
Reset State						Undefined		
Function	"0" is read.					W2	W1	W0

0	0	0	Sunday
0	0	1	Monday
0	1	0	Tuesday
0	1	1	Wednesday
1	0	0	Thursday
1	0	1	Friday
1	1	0	Saturday

Note: Do not set data other than as shown above.

(5) Day column register (PAGE0/1)

DATER (0324H)

	7	6	5	4	3	2	1	0
Bit symbol			DA5	DA4	DA3	DA2	DA1	DA0
Read/Write			R/W					
Reset State			Undefined					
Function	"0" is read.		Day 20	Day 10	Day 8	Day 4	Day 2	Day 1

0	0	0	0	0	1	1st day
0	0	0	0	1	0	2nd day
0	0	0	0	1	1	3rd day
0	0	0	1	0	0	4th day
			:			
0	0	1	0	0	1	9th day
0	1	0	0	0	0	10th day
0	1	0	0	0	1	11th day
			:			
0	1	1	0	0	1	19th day
1	0	0	0	0	0	20th day
			:			
1	0	1	0	0	1	29th day
1	1	0	0	0	0	30th day
1	1	0	0	0	1	31st day

Note1: Do not set data other than as shown above.

Note2: Do not set for non-existent days (e.g.: 30th Feb)

(6) Month column register (for PAGE0 only)

MONTHR (0325H)

	7	6	5	4	3	2	1	0
Bit symbol				MO4	MO4	MO2	MO1	MO0
Read/Write				R/W				
Reset State				Undefined				
Function	"0" is read.			10 months	8 months	4 months	2 months	1 month

0	0	0	0	1	January
0	0	0	1	0	February
0	0	0	1	1	March
0	0	1	0	0	April
0	0	1	0	1	May
0	0	1	1	0	June
0	0	1	1	1	July
0	1	0	0	0	August
0	1	0	0	1	September
1	0	0	0	0	October
1	0	0	0	1	November
1	0	0	1	0	December

Note: Do not set data other than as shown above.

(7) Select 24-hour clock or 12-hour clock (for PAGE1 only)

MONTHR (0325H)

	7	6	5	4	3	2	1	0
Bit symbol								MO0
Read/Write								R/W
Reset State								Undefined
Function	"0" is read.							0: 12-hour 1: 24-hour

(8) Year column register (for PAGE0 only)

YEARR (0326H)

	7	6	5	4	3	2	1	0
Bit symbol	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0
Read/Write	R/W							
Reset State	Undefined							
Function	80 Years	40 Years	20 Years	10 Years	8 Years	4 Years	2 Years	1 Year

0	0	0	0	0	0	0	0	00 years
0	0	0	0	0	0	0	1	01 years
0	0	0	0	0	0	1	0	02 years
0	0	0	0	0	0	1	1	03 years
0	0	0	0	0	1	0	0	04 years
0	0	0	0	0	1	0	1	05 years
				:				
1	0	0	1	1	0	0	1	99 years

Note: Do not set data other than as shown above.

(9) Leap-year register (for PAGE1 only)

YEARR (0326H)

	7	6	5	4	3	2	1	0
Bit symbol							LEAP1	LEAP0
Read/Write							R/W	
Reset State							Undefined	
Function	"0" is read.						00: leap-year 01: one year after leap-year 10: two years after leap-year 11: three years after leap-year	

0	0	Current year is a leap-year
0	1	Current year is the year following a leap year
1	0	Current year is two years after a leap year
1	1	Current year is three years after a leap year

(10)PAGE register (for PAGE0/1)

PAGER (0327H)

A Read-modify- write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	INTENA			ADJUST	ENATMR	ENAALM		PAGE
Read/Write	R/W			W	R/W			R/W
Reset State	0			Undefined	Undefined			Undefined
Function	Interrupt 0: Disable 1: Enable	"0" is read.		0: Don't care 1: Adjust	Clock 0: Disable 1: Enable	ALARM 0: Disable 1: Enable	"0" is read.	PAGE selection

Note: Please keep the setting order below of <ENATMR>, <ENAAML> and <INTENA>. Set difference time for Clock/Alarm setting and interrupt setting.

Example: Clock setting/Alarm setting

LD (PAGER), 0CH : Clock, Alarm enable

LD (PAGER), 8CH : Interrupt enable

PAGE	0	Select Page0
	1	Select Page1

ADJUST	0	Don't care
	1	Adjust sec. counter. When this bit is set to "1" the sec. counter becomes to "0" when the value of the sec. counter is 0-29. When the value of the sec. counter is 30-59, the min. counter is carried and sec. counter becomes "0". Output Adjust signal during 1 cycle of f_{SYS}. After being adjusted once, Adjust is released automatically. (PAGE0 only)

(11) Reset register (for PAGE0/1)

RESTR (0328H)

A Read-modify-write operation cannot be performed

	7	6	5	4	3	2	1	0
Bit symbol	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	–	–	–	–
Read/Write	W							
Reset State	Undefined							
Function	1Hz 0: Enable 1: Disable	16Hz 0: Enable 1: Disable	1:Clock reset	1:Alarm reset	Always write "0"			

RSTALM	0	Unused
	1	Reset alarm register

RSTTMR	0	Unused
	1	Reset clock register

<DIS1HZ>	<DIS16HZ>	PAGER<ENAALM>	Interrupt source signal
1	1	1	Alarm
0	1	0	1Hz
1	0	0	16Hz
Others			Output "0"

3.12.5 Operational description

(1) Reading clock data

1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can read correctly if reading data after 1Hz interrupt occurred.

2. Using two times reading

There is a possibility of incorrect clock data reading when the internal counter carries over. To ensure correct data reading, please read twice, as follows:

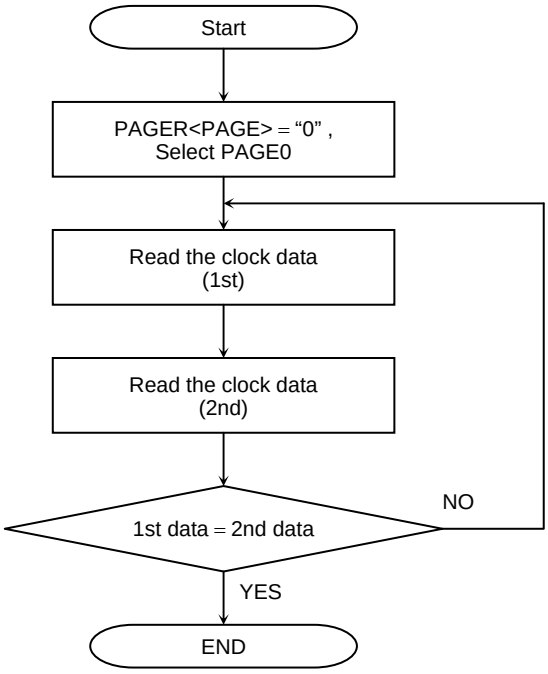


Figure 3.12.2 Flowchart of clock data read

(2) Writing clock data

When a carry over occurs during a write operation, the data cannot be written correctly. Please use the following method to ensure data is written correctly.

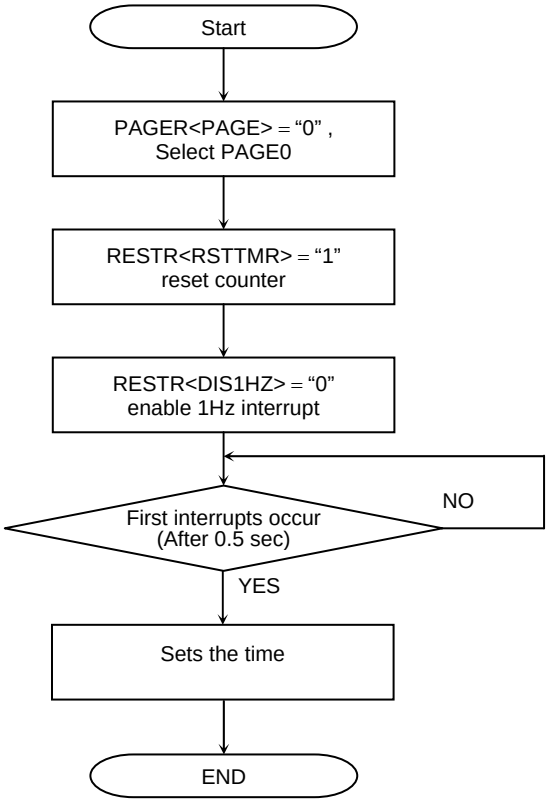
1. Using 1Hz interrupt

1Hz interrupt and the count up of internal data synchronize. Therefore, data can write correctly if writing data after 1Hz interrupt occurred.

2. Resets counter

There are 15-stage counter inside the RTC, which generate a 1Hz clock from 32.768 KHz. The data is written after reset this counter.

However, if clearing the counter, it is counted up only first writing at half of the setting time, first writing only. Therefore, if setting the clock counter correctly, after clearing the counter, set the 1Hz-interrupt to enable. And set the time after the first interrupt (occurs at 0.5Hz) is occurred.



3. Disabling the clock

A clock carry over is prohibited when "0" is written to PAGER<ENATMR> in order to prevent malfunction caused by the Carry hold circuit. While the clock is prohibited, the Carry hold circuit holds a one sec. carry signal from a divider. When the clock becomes enabled, the carry signal is output to the clock, the time is revised and operation continues. However, the clock is delayed when clock-disabled state continues for one second or more. Note that at this time system power is down while the clock is disabled. In this case the clock is stopped and clock is delayed.

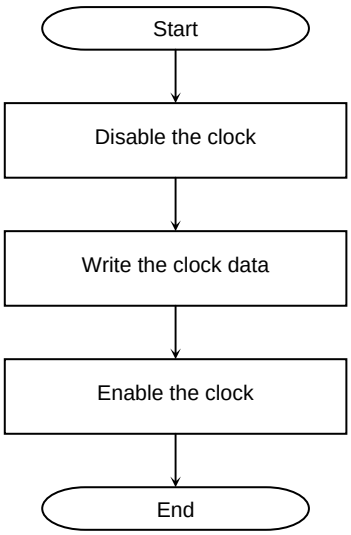


Figure 3.12.3 Flowchart of Clock disable

3.12.6 Explanation of the interrupt signal and alarm signal

The alarm function used by setting the PAGE1 register and outputting either of the following three signals from $\overline{\text{ALARM}}$ pin by writing "1" to PAGER<PAGE>. INTRTC outputs a 1-shot pulse when the falling edge is detected. RTC is not initialized by RESET. Therefore, when the clock or alarm function is used, clear interrupt request flag in INTC (interrupt controller).

(1) When the alarm register and the clock correspond, output "0".
(2) 1Hz Output clock.
(3) 16Hz Output clock.

(1) When the alarm register and the clock correspond, output "0"

When PAGER<ENAALM>= "1", and the value of PAGE0 clock corresponds with PAGE1 alarm register output "0" to $\overline{\text{ALARM}}$ pin and generate INTRTC.

The methods for using the alarm are as follows:

Initialization of alarm is done by writing in "1" to RESTR<RSTALM>. All alarm settings become Don't care. In this case, the alarm always corresponds with value of the clock, and if PAGER<ENAALM> is "1", INTRTC interrupt request is generated.

Setting alarm min., alarm hour, alarm date and alarm day is done by writing data to the relevant PAGE1 register.

When all setting contents correspond, RTC generates an INTRTC interrupt, if PAGER<INTENA><ENAALM> is "1". However, contents which have not been set up (don't care state) are always considered to correspond.

Contents which have already been set up, cannot be returned independently to the Don't care state. In this case, the alarm must be initialized and alarm register reset.

The following is an example program for outputting an alarm from $\overline{\text{ALARM}}$ -pin at noon (PM12:00) every day.

```
    LD      (PAGER), 09H        ;   Alarm disable, setting PAGE1
    LD      (RESTR), D0H        ;   Alarm initialize
    LD      (DAYR), 01H         ;   W0
    LD      (DATAR),01H             1 day
    LD      (HOURR), 12H        ;   Setting 12 o'clock
    LD      (MINR), 00H         ;   Setting 00 min
                                ;   Set up time 31 μs (Note)
    LD      (PAGER), 0CH        ;   Alarm enable
(   LD      (PAGER), 8CH        ;   Interrupt enable )
```

When the CPU is operating at high frequency oscillation, it may take a maximum of one clock at 32 kHz (about 30us) for the time register setting to become valid. In the above example, it is necessary to set 31us of set up time between setting the time register and enabling the alarm register.

Note: This set up time is unnecessary when you use only internal interruption.

(2) With 1Hz output clock

RTC outputs a clock of 1Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “0”, <DIS16HZ>= “1”. RTC also generates an INTRTC interrupt on the falling edge of the clock.

(3) With 16Hz output clock

RTC outputs a clock of 16Hz to $\overline{\text{ALARM}}$ pin by setting up PAGER<ENAALM>= “0”, RESTR<DIS1HZ>= “1”, <DIS16HZ>= “0”. RTC also generates INTRTC an interrupt on the falling edge of the clock.

3.13 Melody/Alarm Generator (MLD)

The TMP91CW40 contains a melody/alarm generator (MLD) for generating melody and alarm waveforms. The MLD can output either alarm or melody waveforms on the $\overline{\text{MLDALM}}$ pin. The alarm generator uses a 15-bit free-running counter that can generate five types of interrupts at fixed intervals.

The MLD has the following features:

- Melody generator

  Based on the low-frequency clock (32.768 kHz), the melody generator can generate clock waveforms at frequencies from 4 Hz to 5461 Hz on the $\overline{\text{MLDALM}}$ pin. By connecting an external speaker, the melody output function can easily be implemented.

- Alarm generator

  The alarm generator can generate eight patterns of alarm waveforms at a frequency of 4096 Hz modulated from the low-frequency clock (32.768 kHz). These waveforms are output on the $\overline{\text{MLDALM}}$ pin and can also be inverted by register programming. By connecting an external speaker, the alarm output function can easily be implemented. The free-running counter in the alarm generator can be used to generate five types of interrupts (1 Hz, 2 Hz, 64 Hz, 512 Hz, 8192 Hz) at fixed intervals.

3.13.1 Block Diagram

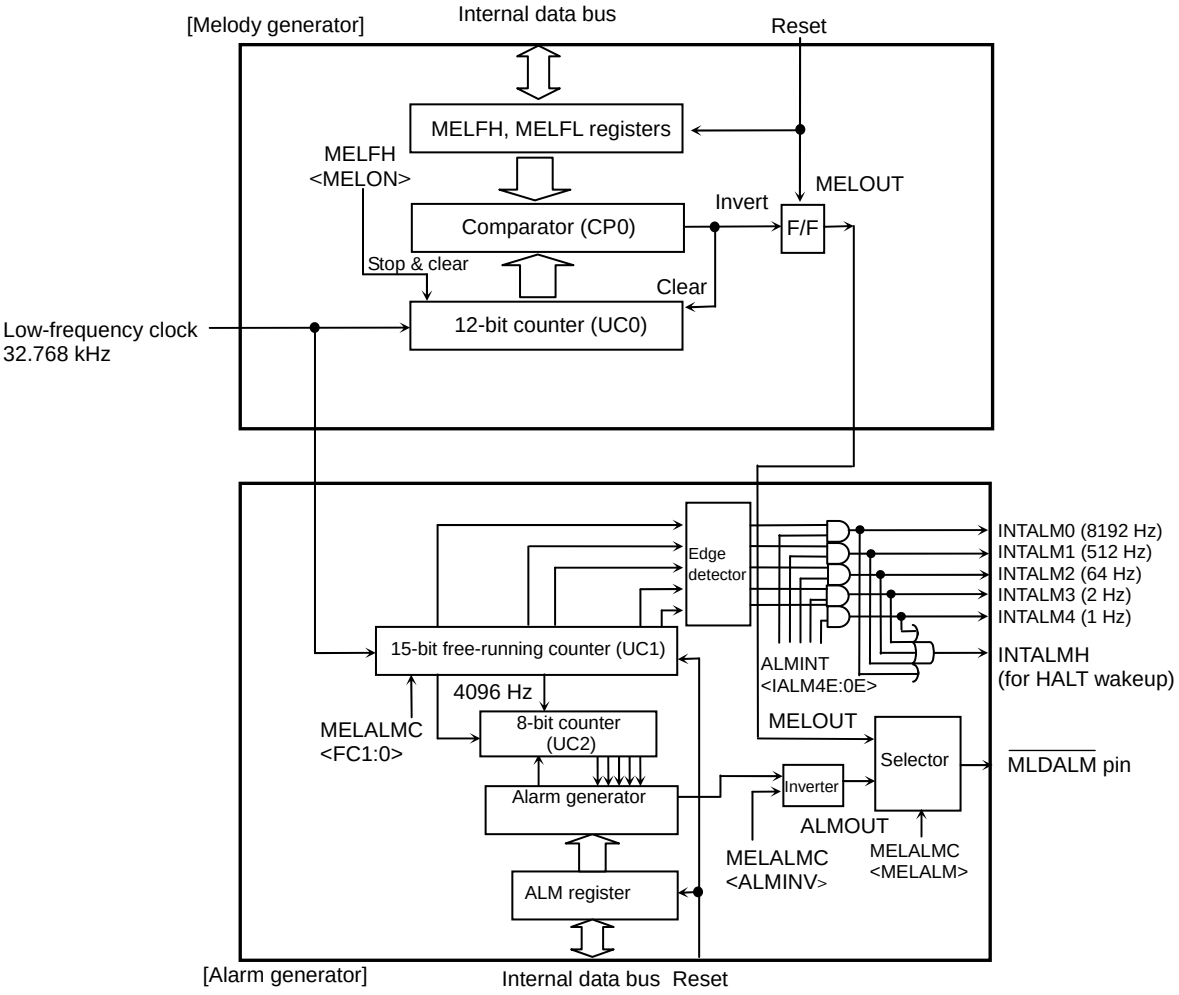


Figure 3.13.1 MLD Block Diagram

3.13.2 SFRs

ALM Register

ALM (0330H)

	7	6	5	4	3	2	1	0
Bit symbol	AL8	AL7	AL6	AL5	AL4	AL3	AL2	AL1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Alarm pattern							

MELALMC Register

MELALMC (0331H)

	7	6	5	4	3	2	1	0
Bit symbol	FC1	FC0	ALMINV	–	–	–	–	MELALM
Read/Write	R/W		R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	Free-running counter control 00: Hold 01: Resume 10: Clear & stop 11: Clear & start		Alarm waveform inversion 1: Invert	Always write 0.				Output waveform select 0: Alarm 1: Melody

Note 1: The <FC1> bit in the MELALMC register is always read as 0.

Note 2: To set bits other than <FC1:0> in the MELALMC register while the free-running counter is running, <FC1:0> must be set to 01.

MELFL Register

MELFL (0332H)

	7	6	5	4	3	2	1	0
Bit symbol	ML7	ML6	ML5	ML4	ML3	ML2	ML1	ML0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Melody frequency (lower 8 bits)							

MELFH Register

MELFH (0333H)

	7	6	5	4	3	2	1	0
Bit symbol	MELON				ML11	ML10	ML9	ML8
Read/Write	R/W				R/W			
After reset	0				0	0	0	0
Function	Melody counter control 0: Stop & clear 1: Start				Melody frequency (upper 4 bits)			

ALMINT Register

ALMINT (0334H)

	7	6	5	4	3	2	1	0
Bit symbol			–	IALM4E	IALM3E	IALM2E	IALM1E	IALM0E
Read/Write			R/W	R/W				
After reset			0	0	0	0	0	0
Function			Always write 0.	1:INTALM4 (1Hz) enable	1:INTALM3 (2Hz) enable	1:INTALM2 (64Hz) enable	1:INTALM1 (512Hz) enable	1:INTALM0 (8192Hz) enable

3.13.3 Operational Description

3.13.3.1 Melody Generator

Based on the low-frequency clock (32.768 kHz), the melody generator can generate clock waveforms at frequencies from 4 Hz to 5461 Hz on the $\overline{\text{MLDALM}}$ pin. By connecting an external speaker, the melody output function can easily be implemented.

(How to use the melody generator)

First, set the MELALMC<MELALM> bit to 1 so that melody waveforms can be output on the $\overline{\text{MLDALM}}$ pin. Then, set the desired melody output frequency in 12 bits of the MELFH and MELFL registers. Finally, setting the MELFH<MELON> bit to 1 starts the counter and melody waveform generation.

How to calculate the melody output frequency and a programming example are shown below.

(How to calculate the melody output frequency)

at fs = 32.768 [kHz]

Melody output frequency: f_{MLD} [Hz] = $32768/(2 \times N + 4)$

Melody setting value: $N = (16384/f_{MLD}) - 2$

(The value N should be a natural number from 1 to 4095 (001H to FFFH); 0 is not allowed.)

(Programming example)

Generating the scale "A" (440 Hz)

```
LD   (MELALMC),--XXXXX1B    ; Select melody waveforms
LD   (MELFL), 23H           ; N = 16384/440 – 2 = 35.2 = 023H
LD   (MELFH), 80H           ; Start waveform generation
```

(Basic scale setting table)

Scale	Frequency [Hz]	Register Value: N
C	264	03CH
D	297	035H
E	330	030H
F	352	02DH
G	396	027H
A	440	023H
B	495	01FH
C	528	01DH

3.13.3.2 Alarm Generator

The alarm generator can generate eight patterns of alarm waveforms at a frequency of 4096 Hz modulated from the low-frequency clock (32.768 kHz). These waveforms are output on the $\overline{\text{MLDALM}}$ pin and can also be inverted by register programming. By connecting an external speaker, the alarm output function can easily be implemented. The free-running counter in the alarm generator can be used to generate five types of interrupts (1 Hz, 2Hz, 64 Hz, 512 Hz, 8192 Hz) at fixed intervals.

(How to use the alarm generator)

First, set the MELALMC<MELALM> bit to 0 so that alarm waveforms can be output on the $\overline{\text{MLDALM}}$ pin, and set the MELALMC<FC1:0> field to 10 to clear the free-running counter. Next the alarm pattern must then be set on the 8-bit register of ALM. If it is inverted output-data, set MELALMC<ALMINV> as invert. Finally set the MELALMC<FC1:0> to 11 to start the free-running counter.

To stop the alarm output, write 00H to the ALM register.

Alarm pattern setting values, a programming example and alarm pattern output waveforms are shown below.

(Alarm pattern setting table)

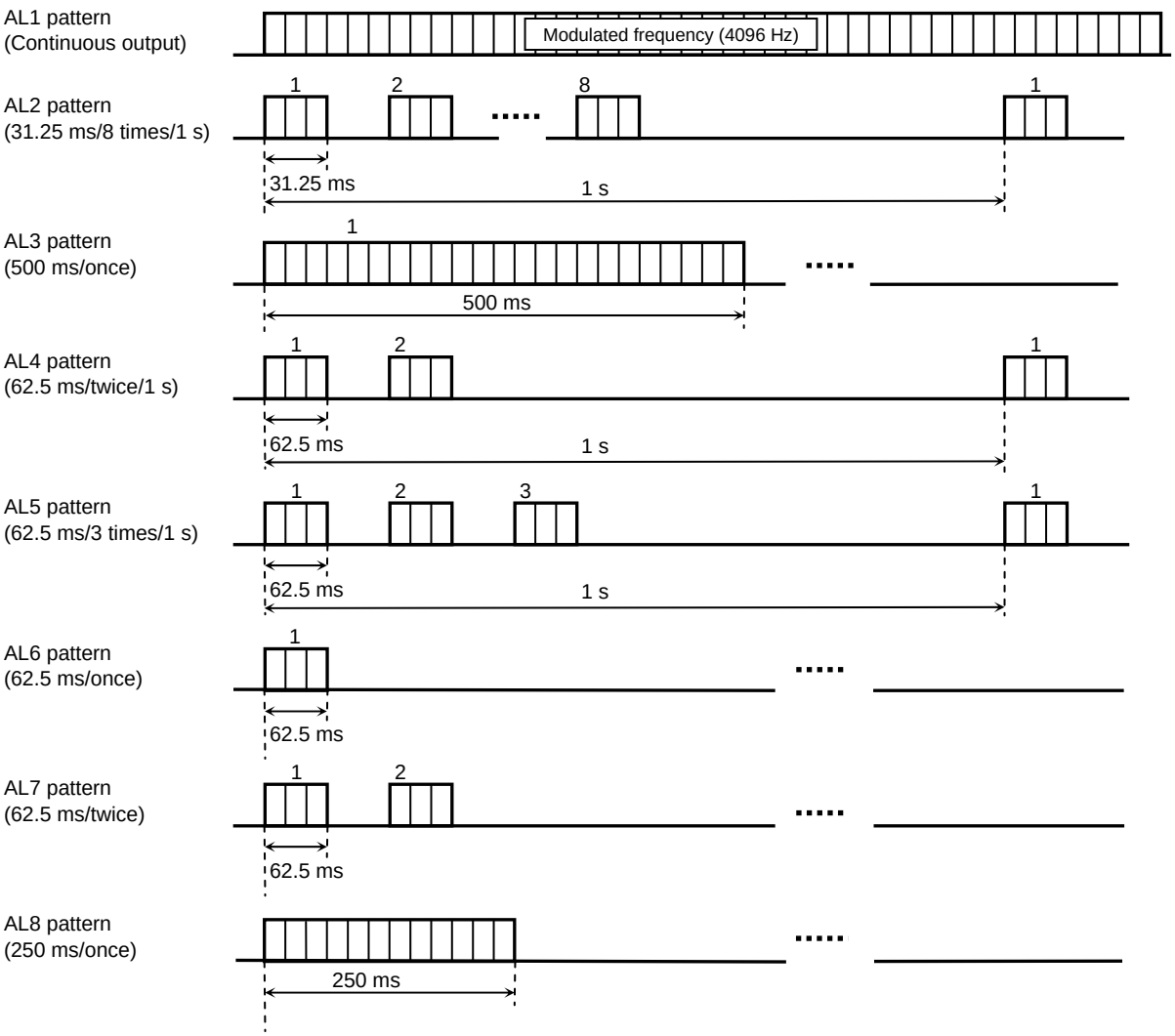
ALM Register Value	Alarm Waveform
00H	Fixed to 0
01H	AL1 pattern
02H	AL2 pattern
04H	AL3 pattern
08H	AL4 pattern
10H	AL5 pattern
20H	AL6 pattern
40H	AL7 pattern
80H	AL8 pattern
Other	Undefined (Don't use)

(Programming example)

Generating the AL2 pattern (31.25 ms/8 times/1 s) alarm

```
LD    (MELALMC), 80H        ; Select output alarm waveform
                            ; Free-running counter clear
LD    (ALM), 02H            ; Set AL2 pattern
LD    (MELALMC), C0H        ; Free-running counter start
```

(Alarm Pattern Output Waveforms: No Inversion)



3.14 Program Patch Logic

The TMP91CW40 has a program patch logic, which enables the user to fix the program code in the internal ROM. Patch program code must be read into the internal RAM from external memory during the startup routine.

Up to six 2-byte sequences (12 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 internal RAM area.

The program patch logic only compares addresses in the internal ROM area; it cannot fix the program code in the internal I/O, internal 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.

3.14.1 Block Diagram

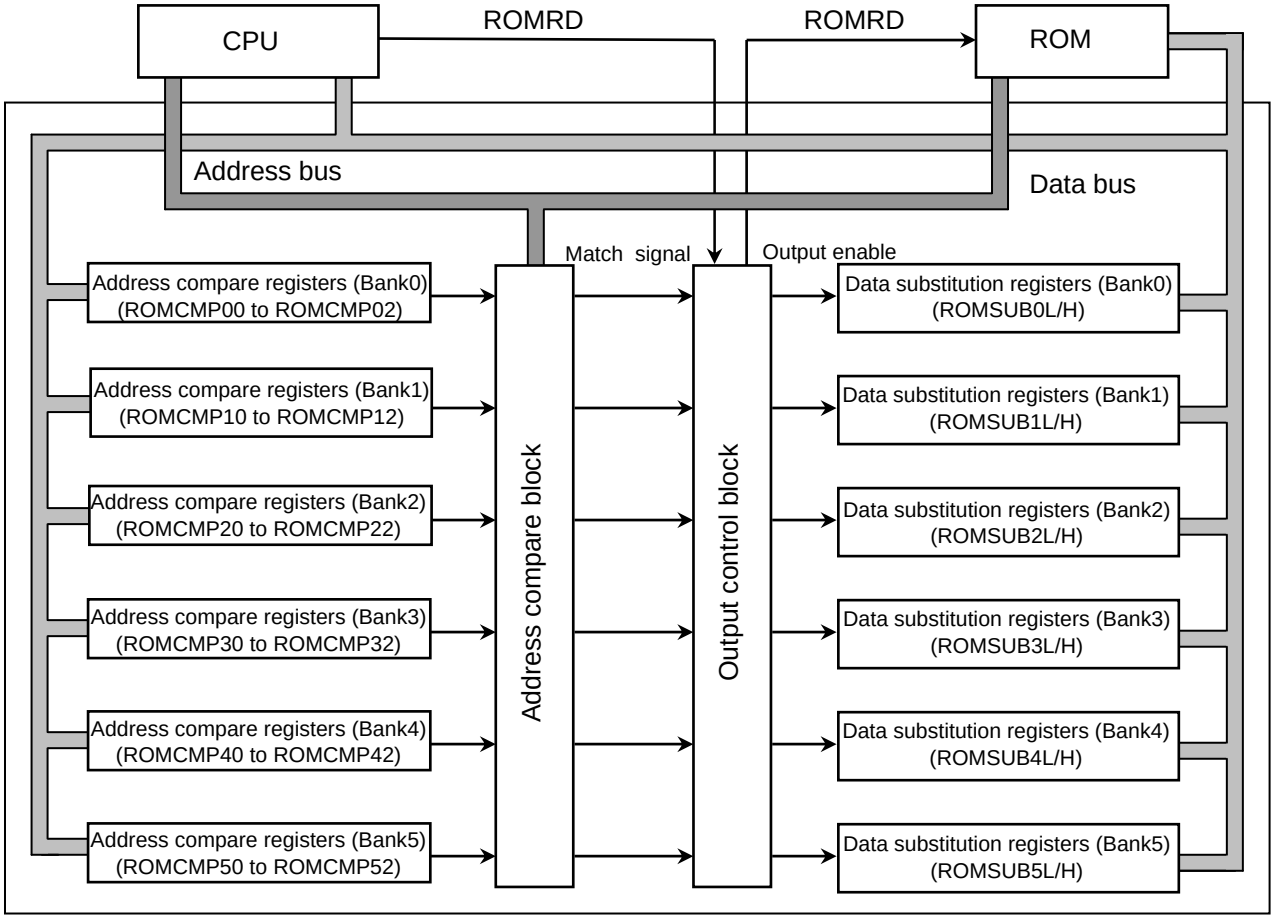


Figure 3.14.1 Program Patch Logic Diagram

3.14.2 SFRs

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 patch code registers (ROMSUBxL and ROMSUBxH).

Bank0 Address Compare Register 0

ROMCMP00 (0400H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	—
Read/Write	W							W
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Lower 7 bits)							Write “0”.

Bank0 Address Compare Register 1

ROMCMP01 (0401H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank0 Address Compare Register 2

ROMCMP02 (0402H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank0 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank0 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP00/01/02 and ROMSUB0L/H registers do not support read-modify-write operation.

Note 2: The ROMCMP00/01/02 and ROMSUB0L/H is read as undefined.

Note 3: Write 0 to bit 0 of the ROMCMP00.

Figure 3.14.2 Program Patch Logic 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							W
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Lower 7 bits)							Write 0

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							
After reset	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							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank1 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank1 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP10/11/12 and ROMSUB1L/H registers do not support read-modify-write operation.

Note 2: The ROMCMP10/11/12 and ROMSUB1L/H is read as undefined.

Note 3: Write 0 to bit 0 of the ROMCMP10.

Figure 3.14.3 Program Patch Logic Registers (Bank1)

Bank2 Address Compare Register 0

ROMCMP20 (0410H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	—
Read/Write	W							W
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Lower 7 bits)							Write 0.

Bank2 Address Compare Register 1

ROMCMP21 (0411H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank2 Address Compare Register 2

ROMCMP22 (0412H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank2 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank 2 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP20/21/22 and ROMSUB2L/H registers do not support read-modify-write operation.

Note 2: The ROMCMP20/21/22 and ROMSUB2L/H is read as undefined.

Note 3: Write 0 to bit 0 of the ROMCMP20.

Figure 3.14.4 Program Patch Logic 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							W
After reset	0	0	0	0	0	0	0	−
Function	Target ROM address (Lower 7 bits)							Write 0.

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							
After reset	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							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank3 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank3 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP30/31/32 and ROMSUB3L/H registers do not support read-modify-write operation.

Note 2: The ROMCMP30/31/32 and ROMSUB3L/H is read as undefined.

Note 3: Write 0 to bit 0 of the ROMCMP30.

Figure 3.14.5 Program Patch Logic Registers (Bank3)

Bank4 Address Compare Register 0

ROMCMP40 (0420H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							W
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Lower 7 bits)							Write 0.

Bank4 Address Compare Register 1

ROMCMP41 (0421H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank 4 Address Compare Register 2

ROMCMP42 (0422H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank4 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank4 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP40/41/42 and ROMSUB4L/H registers do not support read-modify-write operation.

Note 2: The ROMCMP40/41/42 and ROMSUB4L/H is read as undefined.

Note 3: Write 0 to bit 0 of the ROMCMP40.

Figure 3.14.6 Program Patch Logic Registers (Bank4)

Bank5 Address Compare Register 0

ROMCMP50 (0428H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	−
Read/Write	W							W
After reset	0	0	0	0	0	0	0	−
Function	Target ROM address (Lower 7 bits)							Write 0.

Bank5 Address Compare Register 1

ROMCMP51 (0429H)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank5 Address Compare Register 2

ROMCMP52 (042AH)

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank5 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank5 Data 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							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP50/51/52 and ROMSUB5L/H registers do not support read-modify-write operation.

Note 2:The ROMCMP50/51/52 and ROMSUB5L/H is read as undefined.

Note 3:Write 0 to bit 0 of the ROMCMP50.

Figure 3.14.7 Program Patch Logic Registers (Bank5)

3.14.3 Operational Description

(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 RMOSUB0L and ROMSUB0H 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 ROMSUB0L and ROMSUB0H 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	Store 30 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Store 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Store FF in address compare register 2 for bank0.
ROMSUB0L	←	1	0	1	0	1	0	1	0	Store AA in data substitution register L for bank0.
ROMSUB0H	←	0	0	0	1	0	0	0	1	Store 11 in data substitution register H for bank0.

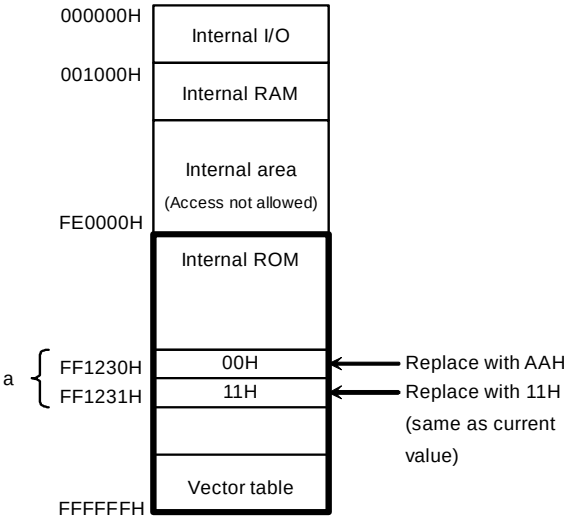


Figure 3.14.8 Example Patch Code Implementation

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	Store 32 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Store 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Store FF in address compare register 2 for bank0.
ROMSUB0L	←	0	0	1	0	0	0	1	0	Store 22 in data substitution register L for bank0.
ROMSUB0H	←	1	0	1	1	1	0	1	1	Store BB in data substitution register H for bank0.

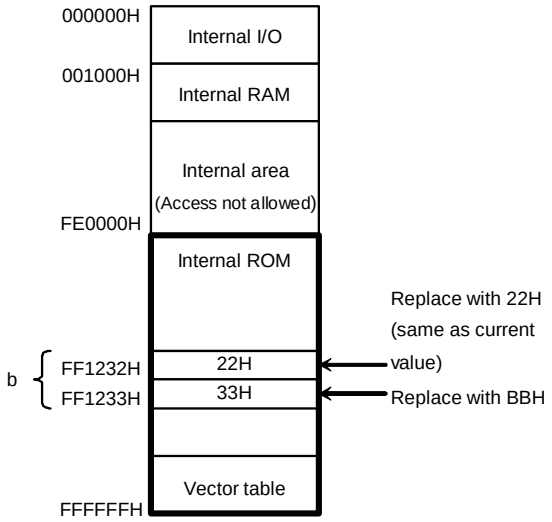


Figure 3.14.9 Example Patch Code Implementation

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	Store 34 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Store 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Store FF in address compare register 2 for bank0.
ROMSUB0L	←	1	1	0	0	1	1	0	0	Store CC in data substitution register L for bank0.
ROMSUB0H	←	1	1	0	1	1	1	0	1	Store DD in data substitution register H for bank0.

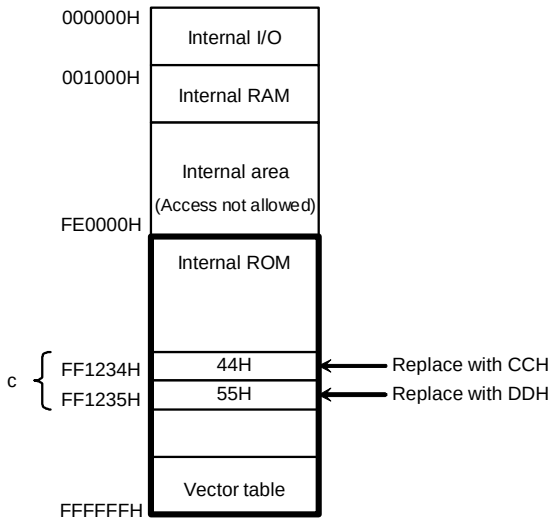


Figure 3.14.10 Example Patch Code Implementation

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	Store 36 in address compare register 0 for bank0.
ROMCMP01	←	0	0	0	1	0	0	1	0	Store 12 in address compare register 1 for bank0.
ROMCMP02	←	1	1	1	1	1	1	1	1	Store FF in address compare register 2 for bank0.
ROMSUB0L	←	0	1	1	0	0	1	1	0	Store 66 in data substitution register L for bank0.
ROMSUB0H	←	1	1	1	0	1	1	1	0	Store EE in data substitution register H for bank0.
ROMCMP10	←	0	0	1	1	1	0	0	0	Store 38 in address compare register 0 for bank1.
ROMCMP11	←	0	0	0	1	0	0	1	0	Store 12 in address compare register 1 for bank1.
ROMCMP12	←	1	1	1	1	1	1	1	1	Store FF in address compare register 2 for bank1.
ROMSUB1L	←	1	1	1	1	1	1	1	1	Store FF in data substitution register L for bank1.
ROMSUB1H	←	1	0	0	1	1	0	0	1	Store 99 in data substitution register H for bank1.

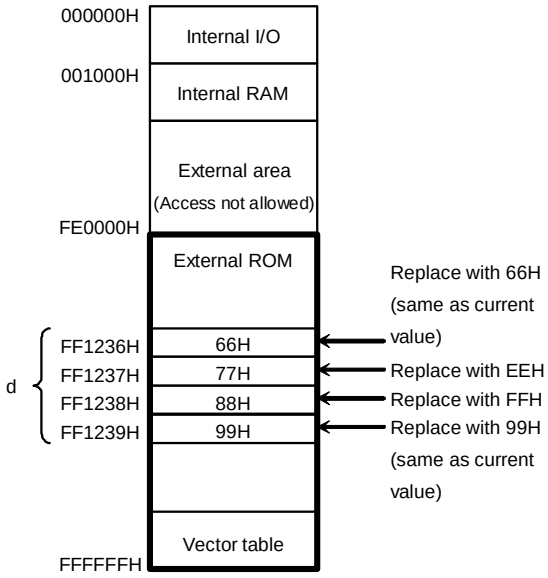


Figure 3.14.11 Example Patch Code Implementation

(2) Using an interrupt to cause a branch

A wider range of program code can also be fixed using a software interrupt (SWI). With patch code loaded into the internal 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 internal RAM addresses specified as SWI vector addresses.

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 ROMSUB0L. If the start address is an odd address, store an SWI instruction code in the ROMSUB0H and the current ROM data at the preceding even address in the ROMSUB0L.

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 a range from FF5000H to FF507F

Before developing the original masked ROM, set the SWI1 vector reference address to address 001500H (in the internal RAM area).

Use the startup routine to load the patch code to the internal 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 the 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 internal RAM area. After executing the patch program the CPU finally rewrites the saved PC value to FF5080H and executes a RETI.

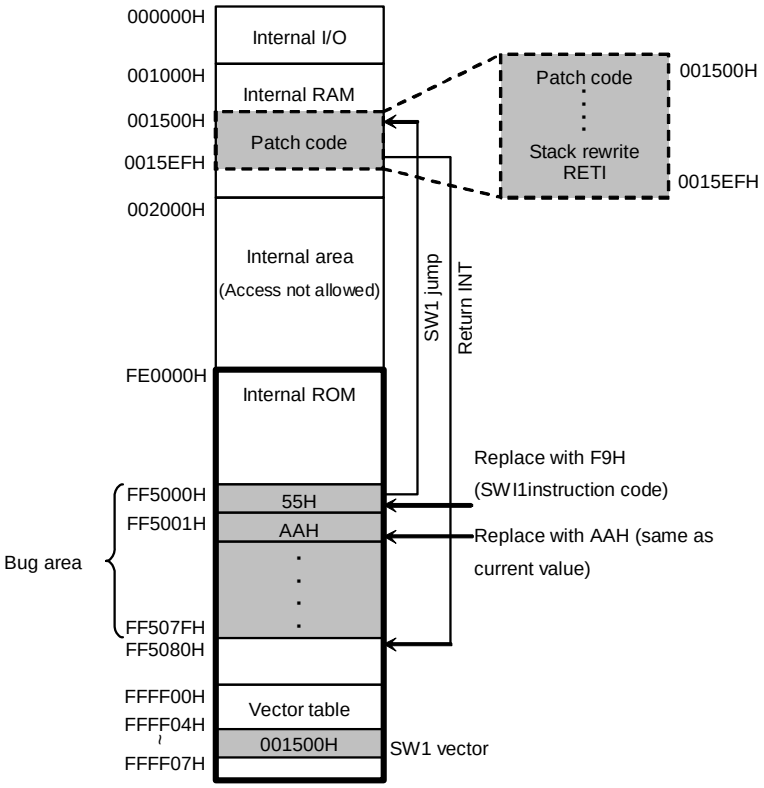


Figure 3.14.12 Example Patch Code Implementation

3.15 Key-On Wakeup

In addition to the INT0 and INT1 interrupt source pins, the TMP91CW40 has four interrupt channels that enable the pressing of a key to terminate HALT mode, called key-on wakeup interrupts (KWI).

Figure 3.15.1 shows a block diagram of the KWI circuit.

3.15.1 Block Diagram

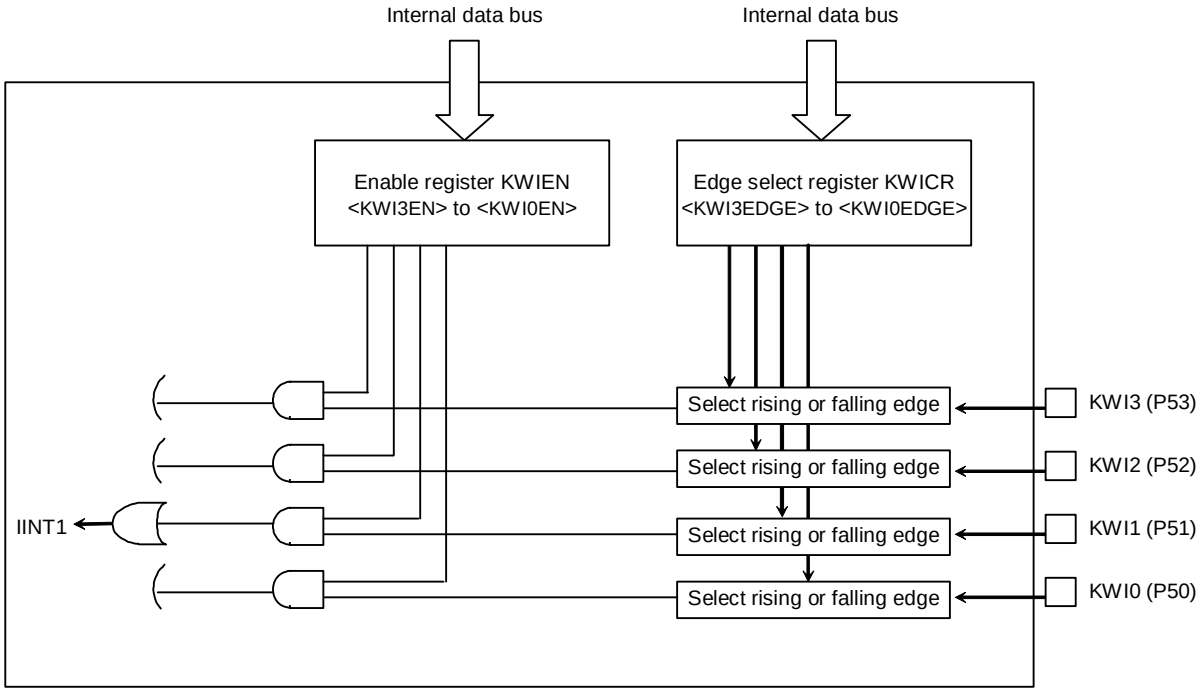


Figure 3.15.1 KWI Block Diagram

3.15.2 SFRs

Key-On Wakeup Enable Register

KWIEN (03A0H)

	7	6	5	4	3	2	1	0
Bit symbol					KWI3EN	KWI2EN	KWI1EN	KWI0EN
Read/Write					W			
After reset					0	0	0	0
Function					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-On Wakeup Control Register

KWICR (03A1H)

	7	6	5	4	3	2	1	0
Bit symbol					KWI3EDGE	KWI2EDGE	KWI1EDGE	KWI0EDGE
Read/Write					W			
After reset					0	0	0	0
Function					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 registers do not support read-modify-write operation.

Figure 3.15.2 Key-On Wakeup Registers

3.15.3 Control

The P50 to P53 pins function as KWI0 to KWI3 when the corresponding bits (<KWIEN3:0>) in the KWIEN register are set. The MCU accepts KWI0 to KWI3 inputs as INT1. The KWI0 to KWI3 pins can be used as external interrupt sources by setting an interrupt priority level in the <I1M2:0> bits of the INTE1ALM0 register.

Example: To detect a falling edge on key-on wakeup channel 0 to generate an interrupt, configure registers in the following sequence:

KWICR	← - - - - - - - 1	Select falling-edge detection for key-on wakeup channel 0.
KWIEN	← - - - - - - - 1	Enable key-on wakeup channel 0.
INTE1ALM0	← X 1 0 0 X - - -	Enable INT1 and set its priority level to 4.

X: Don't care, –: No change

3.16 Analog-to-Digital Converter (AD Converter)

The TMP91CW40 has a 10-bit successive-approximation analog-to-digital converter (AD converter) having 4 channels of analog inputs.

Figure 3.16.1 shows a block diagram of the AD converter. The four analog input channels (AN0 to AN3) can be used as general-purpose digital inputs (Port 5) if not needed as analog channels.

Note: Ensure that the AD converter has halted before executing the HALT instruction to place the TMP91CW40 in IDLE2, IDLE1 or STOP mode to reduce power supply current. Otherwise, the TMP91CW40 might go into a standby mode while the internal analog comparator is still active.

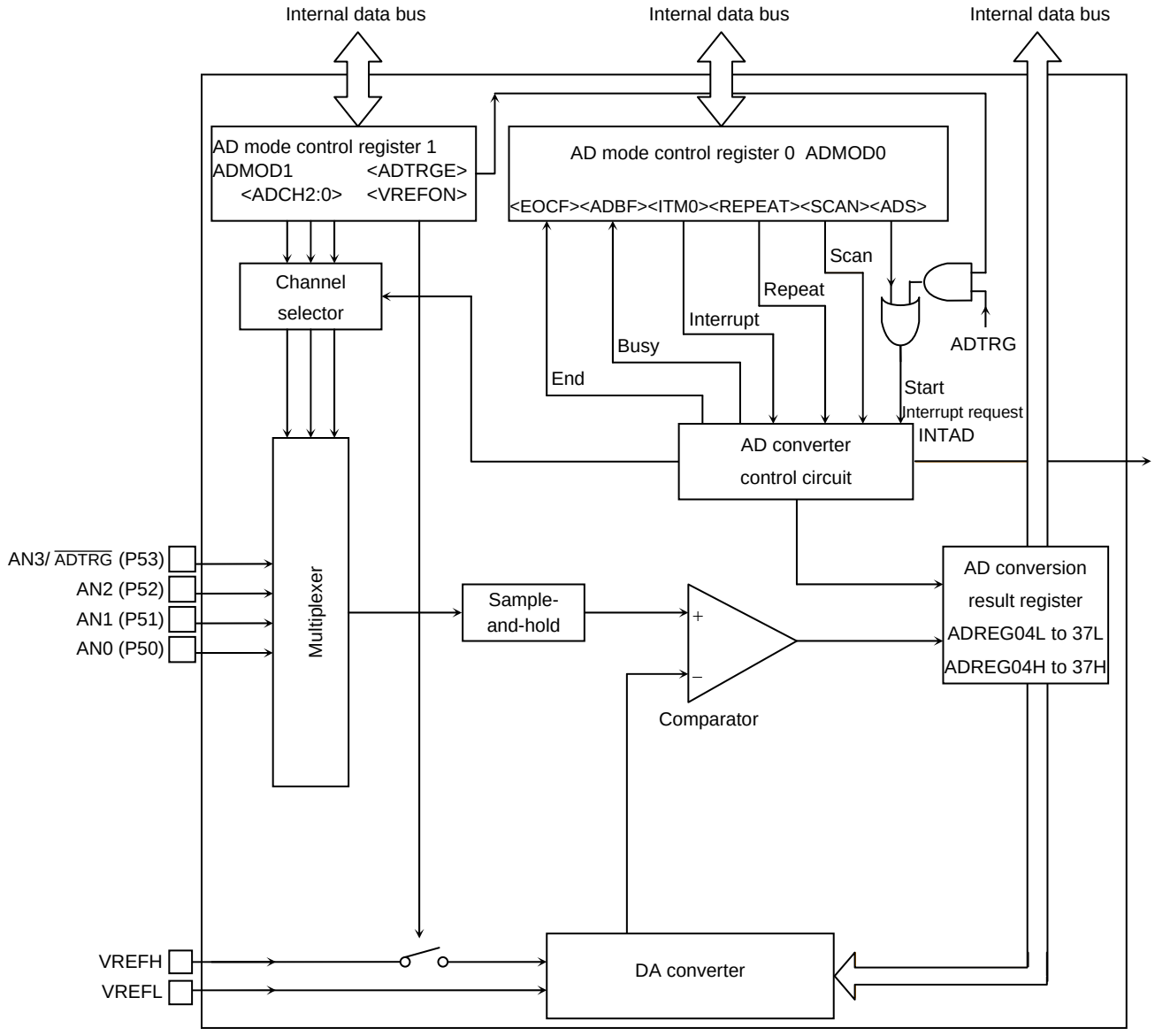


Figure 3.16.1 AD Converter Block Diagram

3.16.1 Control Registers

The AD converter is controlled by the AD mode control registers (ADMOD0 and ADMOD1). AD conversion results are stored in four conversion result high/low register pairs (ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L).

Figure 3.16.2 to Figure 3.16.5 show the registers related to the AD converter.

AD Mode Control Register 0

		7	6	5	4	3	2	1	0
ADMOD0 (02B0H)	Bit symbol	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
	Read/Write	R		R/W					
Read-modify-write instructions cannot be used.	After reset	0	0	0	0	0	0	0	0
	Function	AD conversion end flag 0: Converting 1: Completed	AD conversion busy flag 0: Idle 1: Converting	Always write 0.	Always write 0.	Interrupt in fixed-channel continuous conversion mode 0: After each conversion 1: 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	Start 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	Generate an interrupt after each conversion.
1	Generate an interrupt after every four conversions.

AD conversion busy flag

0	Idle
1	Converting

AD conversion end flag

0	Before or during conversion
1	Completed

Figure 3.16.2 AD Conversion Registers (1)

AD Mode Control Register 1

		7	6	5	4	3	2	1	0
ADMOD1 (02B1H)	Bit symbol	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
	Read/Write	R/W	R/W			R/W			
	After reset	0	0			0	0	0	0
	Function	VREF control 0: OFF 1: ON	ADC operation in IDLE2 mode 0: Stop 1: Operation			External conversion trigger 0: Disable 1: Enable	Analog input channel select		

Analog input channel select

<ADCH2:0> \ <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	Must not be selected.	
101		
110		
111		

AD conversion start by external trigger ($\overline{\text{ADTRG}}$ input)

0	Disable
1	Enable

AD converter operation in IDLE2 mode

0	Stop
1	Operation

Reference voltage for AD converter

0	OFF
1	ON

Set <VREFON> to 1 before setting ADMOD0<ADS> to 1 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 (i.e., ADMOD1<ADTRGE> = 1), the <ADCH2:0> field must not be set to 011.

Figure 3.16.3 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
After reset	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							
After reset	Undefined							
Function	Upper 8 bits of an AD conversion result							

AD Conversion Result Low Register 1/5

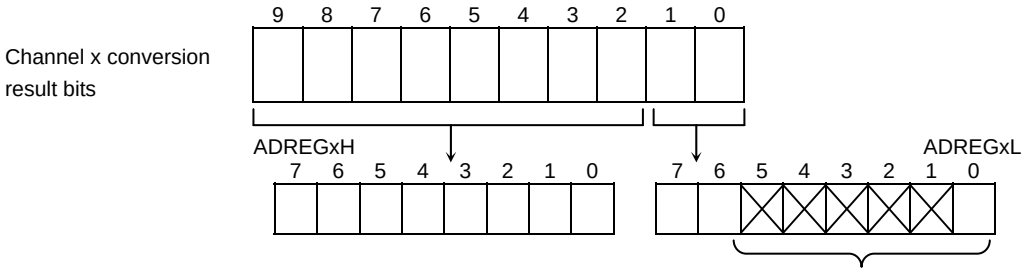
ADREG15L (02A2H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
After reset	Undefined							0
Function	Lower 2 bits of an AD conversion result							Conversion result store flag 1: Stored

AD Conversion Result High Register1/5

ADREG15H (02A3H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	Undefined							
Function	Upper 8 bits of an AD conversion result							



- Bits 5 to 1 are always read as 1.
- 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 ADREGxL is read.

Figure 3.16.4 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
After reset	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							
After reset	Undefined							
Function	Upper 8 bits of an AD conversion result							

AD Conversion Result Low Register 3/7

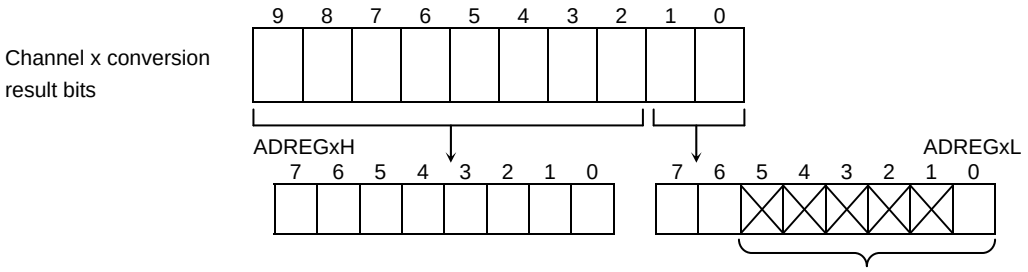
ADREG37L (02A6H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR31	ADR30						ADR3RF
Read/Write	R							R
After reset	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							
After reset	Undefined							
Function	Upper 8 bits of an AD conversion result							



- Bits 5 to 1 are always read as 1.
- 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 ADREGxL is read.

Figure 3.16.5 AD Conversion Registers (4)

3.16.2 Operational Description

(1) Analog reference voltages

The VREFH and VREFL pins provide the reference voltages for the AD converter. 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 <VREFON> is cleared, the internal reference voltage requires a recovery time of 3 μs to stabilize after <VREFON> is 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)

  In this mode, the AD converter runs conversions on a single analog input channel selected from AN0 to AN3 via the ADMOD1<ADCH2:0> bits.

- Channel scan mode (ADMOD0<SCAN> = 1)

  In this mode, the AD converter runs conversions on sequential channels selected from four patterns via the ADMOD1<ADCH2:0> bits.

Table 3.16.1 shows how analog input channels are selected in each conversion mode.

After a reset, ADMOD0<SCAN> is initialized to 0 and ADMOD1<ADCH2:0> to 000, selecting the AN0 pin as the conversion channel in channel-fixed mode. The AN0 to AN3 pins can be used as general-purpose input ports when not used as analog input channels.

Table 3.16.1 Analog Input Channel Selection

<ADCH2:0>	Fixed-Channel Mode <SCAN> = 0	Channel Scan Mode <SCAN> = 1
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3
100	Must not be selected.	
101		
110		
111		

(3) Starting an AD conversion

The AD converter starts a conversion when ADMOD0<ADS> is set to 1, or when a falling edge is applied to the $\overline{\text{ADTRG}}$ pin with ADMOD1<ADTRGE> set to 1. When a conversion starts, the AD conversion busy flag (ADMOD0<ADBF>) is set to 1.

Setting the <ADS> bit to 1 causes the AD converter 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 or not 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 end interrupts

The AD converter supports the following four conversion modes:

- Fixed-channel single conversion mode
- Channel scan conversion mode
- Fixed-channel continuous conversion mode
- Channel scan continuous conversion mode

The conversion mode is selected by the <REPEAT> and <SCAN> bits in the ADMOD0.

At the end of the conversion process, an INTAD interrupt is generated and ADMOD0<EOCF> is set to 1.

a. Fixed-channel single conversion mode

This mode is selected by programming the <REPEAT> and <SCAN> bits in the ADMOD0 to 00. In fixed-channel single conversion mode, the AD converter performs a single conversion on a single selected channel. When the conversion is completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt is generated.

b. Channel scan single conversion mode

This mode is selected by programming the <REPEAT> and <SCAN> bits in the ADMOD0 to 01. In channel scan single conversion mode, the AD converter performs a single conversion on each of a selected group of channels. When the single conversion sequence is completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt is generated.

c. Fixed-channel continuous conversion mode

This mode is selected by programming the <REPEAT> and <SCAN> bits in the ADMOD0 to 10. In fixed-channel continuous conversion mode, the AD converter repeatedly converts a single selected channel. When the conversion process is completed, ADMOD0<EOCF> is set to 1. ADMOD0<ADBF> is not cleared to 0 and remains set.

The INTAD interrupt generation timing can be selected by ADMOD0<ITM0>. When <ITM0>=0, an interrupt is generated after each conversion. When <ITM0>=1, an interrupt is generated 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 channel scan continuous conversion mode, the AD converter repeatedly converts a selected group of channels. When a single conversion sequence is completed, ADMOD0<EOCF> is set to 1 and an INTAD interrupt is generated. ADMOD0<ADBF> is not cleared to 0 and remains set.

In fixed-channel continuous conversion and channel scan continuous conversion modes, setting ADMOD0<REPEAT> to 0 stops the conversion sequence after the ongoing conversion is completed and clears ADMOD0<ADBF> to 0.

When ADMOD1<I2AD>=0, putting the TMP91CW40 in any HALT mode (IDLE2, IDLE1, or STOP) causes the AD converter to be immediately disabled, even if a conversion is in progress. Once the TMP91CW40 exits the HALT mode, the AD converter restarts a conversion sequence in fixed-channel continuous conversion or channel scan continuous conversion mode, but remains inactive in fixed-channel single conversion or channel scan single conversion mode.

Table 3.16.2 summarizes interrupt request generation in each of the conversion modes.

Table 3.16.2 Interrupt Request Generation in Each AD Conversion Mode

Mode	Interrupt Request Generation	ADMOD0		
		<ITM0>	<REPEAT>	<SCAN>
Fixed-channel single conversion mode	After each conversion	X	0	0
Channel scan single conversion mode	After each 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 (6.2 μs when f_{FPH} = 27 MHz).

(6) Storing and reading AD conversion results

AD conversion results are stored in the conversion result high/low register pairs (ADREG04H/L to ADREG37H/L). These registers are read only.

In fixed-channel continuous conversion mode with <ITM0> set to 1, conversion data goes into the ADREG04H/L to ADREG37H/L sequentially. In other modes, conversion results in channels AN0, AN1, AN2 and AN3 are stored in the ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L respectively.

Table 3.16.3 shows the relationships between the analog input channels and the AD conversion result registers.

Table 3.16.3 Relationships between Analog Input Channels and AD Conversion Result Registers

Analog Input Channel (Port 5)	AD Conversion Result Registers	
	Other Modes	Fixed-Channel Continuous Conversion Mode (<ITM0>=1)
AN0	ADREG04H/L	ADREG04H/L → ADREG15H/L → ADREG26H/L → ADREG37H/L → ADREG04H/L
AN1	ADREG15H/L	
AN2	ADREG26H/L	
AN3	ADREG37H/L	

Bit0 (<ADRxRF>) in each ADREGxL register indicates whether or not the conversion result register has been read. This bit is set when the conversion result is loaded into the ADREGxH/ADREGxL register pair, and cleared when either the ADREGxH or ADREGxL is read.

Reading the conversion result also clears the conversion end 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 (1800H) using the AD interrupt (INTAD) service routine

Settings in the main routine

		7	6	5	4	3	2	1	0	
INTE0AD	←	X	1	0	0	–	–	–	–	Enable INTAD and set its priority level to 4.
ADMOD1	←	1	1	X	X	0	0	1	1	Select AN3 as the analog input channel.
ADMOD0	←	X	X	0	0	0	0	0	1	Start conversion in fixed-channel single conversion mode.

Interrupt routine processing example

WA	← ADREG37	Load the conversion result into 16-bit general-purpose register WA from ADREG37L and ADREG37H.
WA	> > 6	Shift the contents of WA 6 bits to the right, padding 0s to the vacated high-order bits.
(1800H)	← WA	Store the contents of WA at address 1800H.

b. Converting the analog input voltages on the AN0 to AN2 pins sequentially in channel scan continuous conversion mode

		7	6	5	4	3	2	1	0	
INTE0AD	←	X	0	0	0	–	–	–	–	Disable INTAD.
ADMOD1	←	1	1	X	X	0	0	1	0	Select AN0 to AN2 as the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in channel scan continuous conversion mode.

X: Don't care, –: No change

3.17 Watchdog Timer (WDT)

The TMP91CW40 contains a watchdog timer. The watchdog timer is used to regain control of the system in the event of software or system lockups due to spurious noise, etc. When a watchdog timer time-out occurs, the watchdog timer generates a nonmaskable interrupt to the CPU.

Connecting the watchdog timer output to the reset pin internally enables a forced reset. (The level of external $\overline{\text{RESET}}$ pin is not changed.)

3.17.1 Configuration

Figure 3.17.1 shows a block diagram of the watchdog timer.

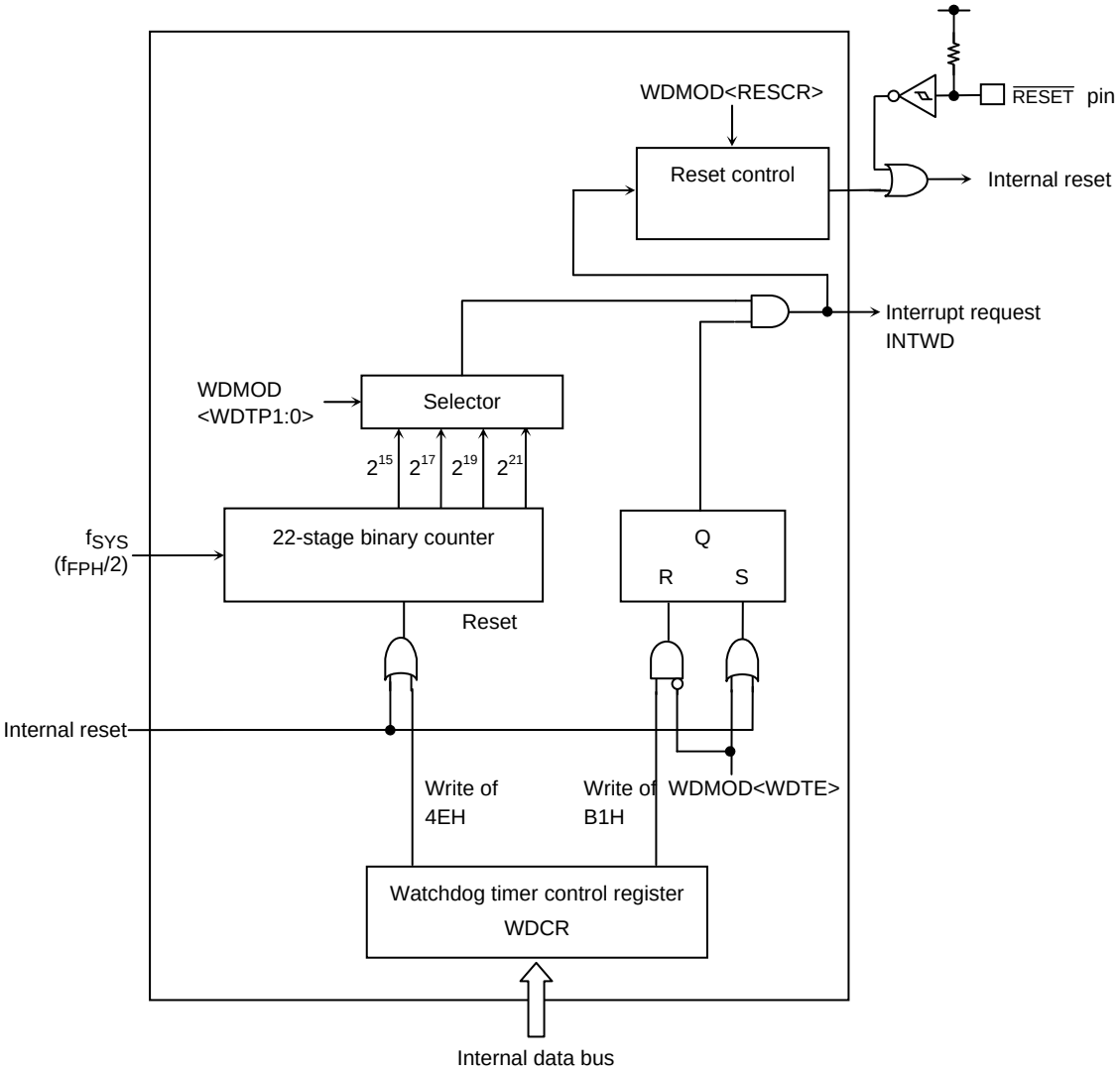


Figure 3.17.1 Watchdog Timer Block Diagram

Noise: Careful consideration must be given in designing a system because the watchdog timer may not be able to realize its full functionality due to external noise, etc.

3.17.2 Operational Description

The watchdog timer is a kind of timer that generates an interrupt request if it times out. The watchdog timer allows the user to program the time-out period in the <WDTP1:0> field in the WDMOD register. While the watchdog timer is enabled, it can be cleared to 0 by software at any time by writing a special clear code. If the CPU loses control of the system and fails to execute an instruction to clear the counter before it reaches the time-out time due to noise or other causes, the watchdog timer generates an INTWD interrupt. In response to the interrupt, the CPU jumps to a system recovery routine to regain control of the system.

The watchdog timer begins counting immediately after reset.

The watchdog timer halted in IDLE1 or STOP mode.

In IDLE2 mode, the <I2WDT> bit in the WDMOD determines whether or not the watchdog timer is disabled. Program the <I2WDT> bit as necessary before placing the TMP91CW40 in IDLE2 mode.

The watchdog timer contains a 22-stage binary counter clocked by the system clock f_{SYS}. The binary counter can output $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/2^{19}$ or $f_{SYS}/2^{21}$, which is selected by WDMOD<WDTP1:0>. When the watchdog timer counter overflows, a watchdog timer interrupt is generated as shown in Figure 3.17.2.

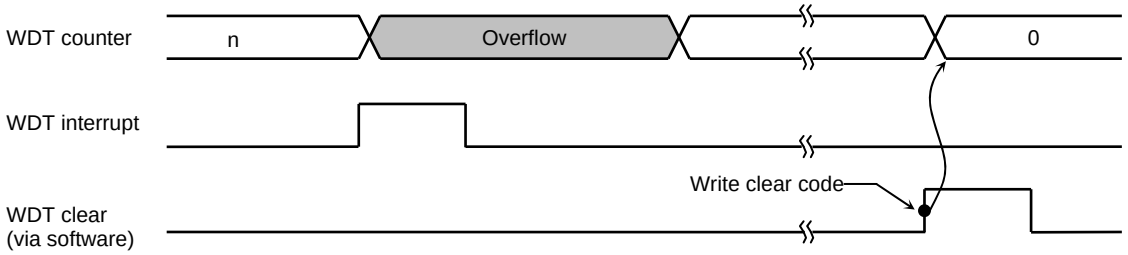


Figure 3.17.2 Normal Operation

It is also possible to reset the system when the watchdog timer counter overflows. In this case, a reset operation takes 22 to 29 states (1.63 to 2.15 μs when fc = 27 MHz) as shown in Figure 3.17.3. After a reset, the system clock f_{SYS} (1 cycle = 1 state) is fc/2.

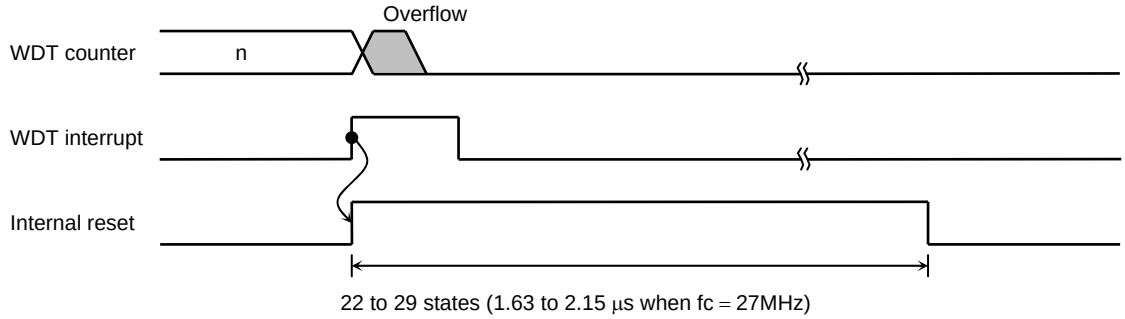


Figure 3.17.3 Reset Operation

3.17.3 Control Registers

The watchdog timer is controlled by two registers called WDMOD and WDCR.

(1) Watchdog timer mode register (WDMOD)

a. Time-out period <WDTP1:0>

This 2-bit field determines the duration of the watchdog timer time-out interval. A reset initializes WDMOD<WDTP1:0> to 00. Figure 3.17.4 shows possible time-out periods.

b. Watchdog timer enable/disable control <WDTE>

A reset initializes WDMOD<WDTE> to 1, enabling the watchdog timer. To disable the watchdog timer, the clearing of the <WDTE> bit must be followed by a write of a special key code (B1H) to the WDCR register. This protects the watchdog timer from being inadvertently disabled. To re-enable the watchdog timer, it is only necessary to set the <WDTE> bit to 1.

c. System reset <RESCR>

This bit is used to program the watchdog timer to generate a system reset when it reaches the time-out time. A reset initializes WDMOD<RESCR> = 0 so that a time-out does not cause a system reset.

(2) Watchdog timer control register (WDCR)

This register is used to disable the watchdog timer and to clear the watchdog timer's binary counter.

- Disabling the watchdog timer

The watchdog timer can be disabled by clearing WDMOD<WDTE> to 0 and then writing the disable code (B1H) to the WDCR register.

WDCR	← 0 1 0 0 1 1 1 0	Write the clear code (4EH).
WDMOD	← 0 – – X X – – 0	Clear <WDTE> to 0.
WDCR	← 1 0 1 1 0 0 0 1	Write the disable code (B1H).

- Enabling the watchdog timer

The watchdog timer can be enabled by setting WDMOD<WDTE> to 1.

- Clearing the watchdog timer counter

Writing the clear code (4EH) to the WDCR register clears the binary counter and causes the counter to start counting again.

WDCR	← 0 1 0 0 1 1 1 0	Write the clear code (4EH).

Note1: If the disable control is used, set the disable code (B1H) to WDCR after writing the clear code (4EH) once. (Please refer to setting example.)

Note2: If the watchdog timer setting is changed, change setting after setting to disable condition once.

WDMOD (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	R/W		R/W
After reset	1	0	0	0	0	0	0	0
Function	WDT control 0: Disable 1: Enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		Always write 0.	Always write 0.	IDLE2 0: Stop 1: Operate	1:Internally connects WDT out to the reset pin	Always write 0 .

Watchdog timer out control

0	–
1	Internally route the WDT time-out signal to the system reset.

IDLE2 control

0	Stop
1	Operation

Watchdog timer detection time at fc = 27 MHz, fs = 32.768 kHz

System Clock SYSCR1 <SYSCK>	Watchdog Timer Detection Time WDMOD<WDTP1:0>			
	00	01	10	11
1 (fs)	2.0 s	8.0 s	32.0 s	128.0 s
0 (fc)	2.43 ms	9.71 ms	38.84 ms	155.34 ms

Watchdog timer enable

0	Disable
1	Enable

Figure 3.17.4 Watchdog Timer Mode Register

WDCR (0301H)

Read-modify-write instructions cannot be used.

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
After reset	–							
Function	B1H: WDT disable code 4EH: WDT clear code							

Special code

B1H	Disable code
4EH	Clear code
Other	–

Figure 3.17.5 Watchdog Timer Control Register

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Supply voltage	Vcc	−0.5 to 4.0	V
Input voltage	VIN	−0.5 to Vcc + 0.5	V
Output current (per pin)	IOL (other than Port8)	2	mA
	IOL (Port8)	20	mA
Output current (per pin)	IOH	−2	mA
Output current (total)	Σ IOL (other than Port8)	60	mA
	Σ IOL (Port8)	80	mA
Output current (total)	Σ IOH	−80	mA
Power dissipation (Ta = 85°C)	PD	600	mW
Soldering temperature (10 s)	TSOLDER	260	°C
Storage temperature	TSTG	−65 to 150	°C
Operating temperature	TOPR	−40 to 85	°C

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

Solderability of lead free products

Test parameter	Test condition	Note
Solderability	(1) Use of Sn−37Pb solder Bath Solder bath temperature =230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	(2) Use of Sn−3.0Ag−0.5Cu solder bath Solder bath temperature =245°C, Dipping time = 5 seconds 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 = 8 to 27 MHz	fs = 30 to 34 kHz	2.7		3.6	V
			fc = 8 to 16 MHz		2.2			
Low-level input voltage	P0, P1, P2, P5, P62, P7, P8, P9, PA, PB	VIL1	Vcc ≥ 2.7 V		−0.3		0.3 Vcc	V
			Vcc < 2.7 V				0.2 Vcc	
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P60(INT0), P61(INT1)	VIL2	Vcc ≥ 2.7 V				0.25 Vcc	
			Vcc < 2.7 V				0.15 Vcc	
	AM0, AM1	VIL3	Vcc ≥ 2.7 V				0.3	
			Vcc < 2.7 V				0.3	
	X1	VIL4	Vcc ≥ 2.7 V				0.2 Vcc	
			Vcc < 2.7 V				0.1 Vcc	
High-level input voltage	P0, P1, P2, P5, P62, P7, P8, P9, PA, PB	VIH1	Vcc ≥ 2.7 V		0.7 Vcc		Vcc + 0.3	V
			Vcc < 2.7 V		0.8 Vcc			
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P60(INT0),P61(INT1)	VIH2	Vcc ≥ 2.7 V		0.75 Vcc			
			Vcc < 2.7 V		0.85 Vcc			
	AM0, AM1	VIH3	Vcc ≥2.7 V		Vcc − 0.3			
			Vcc < 2.7 V		Vcc − 0.3			
	X1	VIH4	Vcc ≥ 2.7 V		0.8 Vcc			
			Vcc < 2.7 V		0.9 Vcc			
Low-level output voltage		VOL	IOL = 1.6 mA	Vcc ≥ 2.7 V			0.45	V
			IOL = 0.4 mA	Vcc < 2.7 V			0.15 Vcc	
High-level output voltage		VOH	IOH = −400 μA	Vcc ≥ 2.7 V	Vcc − 0.3			V
Low-level output current (Port 8)		IOL	VOL = 1.0 V	Vcc ≥ 2.7 V			15	mA
			VOL = 1.0 V	Vcc ≥ 2.2 V			10	

Note: Ta = 25°C, Vcc = 3.0 V, unless otherwise noted.

DC Electrical Characteristics (2/2)

Parameter	Symbol	Condition	Min	Typ. (Note 1)	Max	Unit
Input leakage current	ILI	$0.0 \leq V_{IN} \leq Vcc$		0.02	±5	μA
Output leakage current	ILO	$0.2 \leq V_{IN} \leq Vcc - 0.2$		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	2.2		3.6	V
$\overline{RESET}$ pull-up resistor	RRST	Vcc ≥ 2.7 V	100		400	kΩ
		Vcc < 2.7 V	200		1000	
Pin capacitance	CIO	fc = 1 MHz			10	pF
Schmitt width $\overline{RESET}$, $\overline{NMI}$, INT0, INT1	VTH	Vcc ≥ 2.7 V	0.4			V
		Vcc < 2.7 V	0.3			
NORMAL (Note 2)	Icc	Vcc = 2.7 V to 3.6 V fc = 27 MHz		34	46	mA
IDLE2				25	34	
IDLE1				18	26	
NORMAL (Note 2)		Vcc = 2.2 V to 3.6 V fc = 16 MHz		15	21	mA
IDLE2				11	16	
IDLE1				8	11	
SLOW (Note 2)		Vcc = 2.2 V to 3.6 V fs = 32.768 kHz		30	75	μA
IDLE2				20	60	
IDLE1				13	45	
STOP		Vcc = 2.2 V to 3.6 V		1	10	μA

Note 1: Typical values are for when Ta = 25°C and Vcc = 3.0 V unless otherwise noted.

Note 2: Icc measurement conditions (NORMAL, SLOW):

All functions are operating; output pins are input pins are fixed.

4.3 AD Conversion Electrical Characteristics

AVCC = VCC , AVSS = VSS

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)	VREFH	Vcc ≥ 2.7V	VCC−0.2V	VCC	VCC	V
		Vcc < 2.7 V	VCC	VCC	VCC	
Analog reference voltage (−)	VREFL	Vcc ≥ 2.7V	VSS	VSS	VSS+0.2V	
		Vcc < 2.7 V	VSS	VSS	VSS	
Analog input voltage	VAIN		VREFL		VREFH	
Analog current for analog reference voltage <VREFON> = 1	IREF (VREFL=0V)	Vcc ≥ 2.7V		0.94	1.35	mA
		Vcc < 2.7 V		0.65	0.90	
<VREFON> = 0		V_{CC} = 2.2V to 3.6V		0.02	5.0	μA
Total error (not including quantization error)	–	Vcc ≥ 2.7V		±1.0	±4.0	LSB
		Vcc < 2.7 V		±1.0	±4.0	

Note 1: 1 LSB = (VREFH – VREFL)/1024 [V]

Note 2: Minimum operating frequency

The operation of the AD converter is guaranteed only when the high-fequency oscillator (fc) is used and the clock selected with the clock gear is 4 MHz or higher (not guaranteed with fs).

Note 3: The supply current flowing through the AVCC pin is included in the VCC pin supply current (I_{CC}).

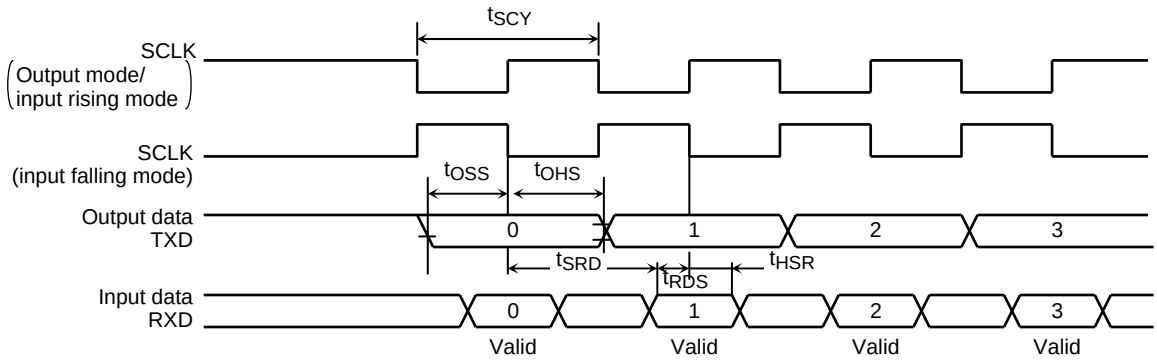
4.4 SIO Timing (I/O Interface Mode)

(1) SCLK input mode

Parameter	Symbol	Equation		16 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16X		1.0		0.59		μs
Output Data → SCLK rising /falling edge*	t_{OSS}	$t_{SCY}/2 - 4X - 110$ (V_{CC} = 2.7V to 3.6V)		140		38		ns
		$t_{SCY}/2 - 4X - 180$ (V_{CC} = 2.2V to 2.7V)		70		–		
SCLK rising /falling edge* → Output Data hold	t_{OHS}	$t_{SCY}/2 + 2X + 0$		625		370		ns
SCLK rising /falling edge* → Input Data hold	t_{HSR}	3X + 10		198		121		ns
SCLK rising /falling edge* → Valid Data hold	t_{SRD}		$t_{SCY} - 0$		1000		592	ns
Valid data input /falling edge* → SCLK rising /falling edge*	t_{RDS}	0		0		0		ns

(2) SCLK output mode

Parameter	Symbol	Equation		16 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16X	8192X	1.0	512	0.59	303	μs
Output Data → SCLK rising /falling edge*	t_{OSS}	$t_{SCY}/2 - 40$		460		256		ns
SCLK rising /falling edge* → Output Data hold	t_{OHS}	$t_{SCY}/2 - 40$		460		256		ns
SCLK rising /falling edge* → Input Data hold	t_{HSR}	0		0		0		ns
SCLK rising /falling edge* → Valid Data hold	t_{SRD}		$t_{SCY} - 1X - 180$		757		375	ns
Valid data input /falling edge* → SCLK rising /falling edge*	t_{RDS}	1X + 180		243		217		ns



Note 1: SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

Note 2: The values shown in the 27 MHz and 16 MHz columns are measured with t_{SCY} = 16X.

Note 3: In the above tables, 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 clock gear setting and whether the high-frequency or low frequency oscillator is used.

4.5 Timer/Counter Input (ECIN) Characteristics

Parameter	Symbol	Condition		Min	Typ.	Max	Unit
Timer/counter input (ECIN1 to ECIN3 input)	t_{TC1}	Frequency measurement mode VDD =2.7 to 3.6 V	Count on a single edge	–	–	fc/2 (fc/2 = max. 8MHz)	MHz
			Count on both edges				
		Frequency measurement mode VDD =2.2 to 2.7 V	Count on a single edge	–	–		
			Count on both edges				

4.6 Interrupts

(1) $\overline{NMI}$, INT0 and INT1 interrupts

Parameter	Symbol	Equation		16 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width for $\overline{NMI}$, INT0, INT1	t_{INTAL}	4X + 40		290		188		ns
High pulse width for $\overline{NMI}$, INT0, INT1	t_{INTAH}	4X + 40		290		188		ns

Note 1: Xc represents the cycle period of the high-frequency oscillator clock (fc).

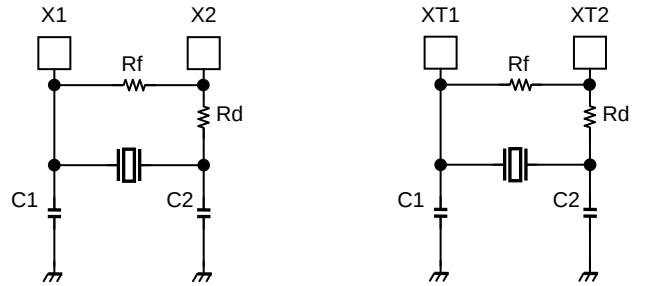
Note 2: In the above table, the letter x represents the f_{FPH} cycle period, which is half the system clock (f_{SYS}) cycle period used in the CPU core. The f_{FPH} cycle period varies depending on the clock gear setting and whether the high-frequency or low frequency oscillator is used.

4.7 Recommended Crystal Oscillation Circuit

TMP91CW40FG is evaluated by below oscillator vender. When selecting external parts, make use of this information.

Note: Total loads value of oscillator is sum of external loads (C1 and C2) and floating loads of actual assemble board. There is a possibility of miss-operating using C1 and C2 value in below table. When designing board, it should design minimum length pattern around oscillator. And we recommend that oscillator evaluation try on your actual using board.

(1) Connection example



High-frequency oscillator Low-frequency oscillator

(2) Recommended ceramic oscillator: Murata Manufacturing Co., Ltd. (JAPAN)

For up-to-date information, please refer to the following URL:

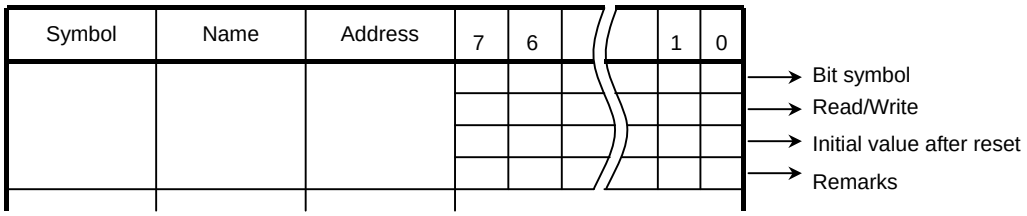
http://www.murata.co.jp/

5. Table of SFRs

The special function registers (SFRs) include the I/O ports and peripheral control registers allocated to the 4-Kbyte address space from 000000H to 000FFFH.

(1) I/O ports
(2) Interrupt control
(3) Clock gear
(4) UART/serial channel
(5) AD converter
(6) Watchdog timer
(7) Real time clock
(8) Melody/Alarm generator
(9) Divider output
(10) Key-on wakeup
(11) LCD driver
(12) Program patch logic
(13) 8-bit timer
(14) 16-bit timer

Table layout



Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these register.

Example: When setting bit0 only of the register PxCR, the instruction "SET 0, (PxCR)" cannot be used. The LD (Transfer) instruction must be used to write all eight bits.

Read/write

R/W: Both read and write are possible.
R: Only read is possible.
W: Only write is possible.
W*: Both read and write are possible (when this bit is read as 1).
Prohibit RMW: Read-modify-write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read-modify-write instructions.)
R/W*: Read-modify-write is prohibited when controlling the pull-up resistor.

表 5.1 SFR Address Map

[1] PORT

Address	Name
0000H	P0
1H	P1
2H	P0CR
3H	
4H	P1CR
5H	
6H	P2
7H	
8H	P2CR
9H	
AH	
BH	
CH	
DH	P5
EH	
FH	

Address	Name
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	Name
0020H	PACR
1H	PAFC
2H	
3H	
4H	PB
5H	PBCR
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	P7FC2
EH	
FH	ODE

[2] INTC

Address	Name
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	Name
0090H	INTE0AD
1H	INTE1ALM0
2H	INTEALM12
3H	INTEALM34
4H	INTETMR56
5H	INTETMR78
6H	INTETMR12
7H	
8H	
9H	INTETMR3
AH	INTES0
BH	INTES1
CH	INTERTC
DH	INTES2
EH	INTES3
FH	

Address	Name
00A0H	INTETC01
1H	INTETC23
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[3] CGEAR

Address	Name
00E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	SYSCR3
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

表 5.2 SFR Address Map

[4] UART/SIO

Address	Name
0200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	

Address	Name
0210H	SC2BUF
1H	SC2CR
2H	SC2MOD0
3H	BR2CR
4H	BR2ADD
5H	SC2MOD1
6H	
7H	
8H	SC3BUF
9H	SC3CR
AH	SC3MOD0
BH	BR3CR
CH	BR3ADD
DH	SC3MOD1
EH	
FH	

[5]AD converter

Address	Name
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	Name
02B0H	ADMOD0
1H	ADMOD1
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] WDT

Address	Name
0300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[7]RTC

Address	Name
0320H	SECR
1H	MINR
2H	HOURR
3H	DAYR
4H	DATER
5H	MONTHR
6H	YEARR
7H	PAGER
8H	RESTR
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[8] MLD

Address	Name
0330H	ALM
1H	MELALMC
2H	MELFL
3H	MELFH
4H	ALMINT
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] DVO

Address	Name
0340H	TBTCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] KWI

Address	Name
03A0H	KWIEN
1H	KWICR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

表 5.3 SFR Address Map

[11] LCDD

Address	Name
03D0H	LCDCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	LCDCR2
FH	

Address	Name
03E0H	LCDREG0
1H	LCDREG1
2H	LCDREG2
3H	LCDREG3
4H	LCDREG4
5H	LCDREG5
6H	LCDREG6
7H	LCDREG7
8H	LCDREG8
9H	LCDREG9
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
03F0H	LCDREG10
1H	LCDREG11
2H	LCDREG12
3H	LCDREG13
4H	LCDREG14
5H	LCDREG15
6H	LCDREG16
7H	LCDREG17
8H	LCDREG18
9H	LCDREG19
AH	
BH	
CH	
DH	
EH	
FH	

[12] Program patch logic

Address	Name
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	Name
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	Name
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	

[13] 8-bit timer

Address	Name
0900H	TC5CR1
1H	TC6CR1
2H	TC5CR2
3H	TC6CR2
4H	TTREG5
5H	TTREG6
6H	
7H	
8H	PWREG5
9H	PWREG6
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0910H	TC7CR1
1H	TC8CR1
2H	TC7CR2
3H	TC8CR2
4H	TTREG7
5H	TTREG8
6H	
7H	
8H	PWREG7
9H	PWREG8
AH	
BH	
CH	
DH	
EH	
FH	

[14] 16-bit timer

Address	Name
0940H	TREG1AL
1H	TREG1AH
2H	
3H	TREG1B
4H	TC1CR1
5H	TC1CR2
6H	TC1SR
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

表 5.4 SFR Address Map

Address	Name
0950H	TREG2AL
1H	TREG2AH
2H	
3H	TREG2B
4H	TC2CR1
5H	TC2CR2
6H	TC2SR
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0960H	TREG3AL
1H	TREG3AH
2H	
3H	TREG3B
4H	TC3CR1
5H	TC3CR2
6H	TC3SR
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

6. Port Section Equivalent Circuit Diagrams

- Reading the circuit diagrams

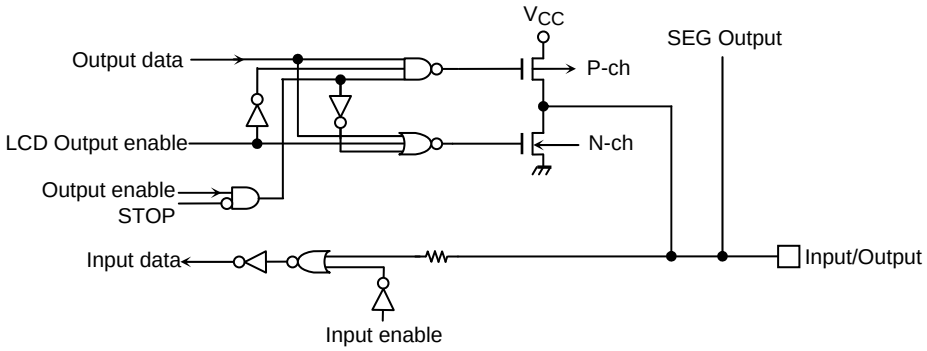
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The dedicated signal is described below.

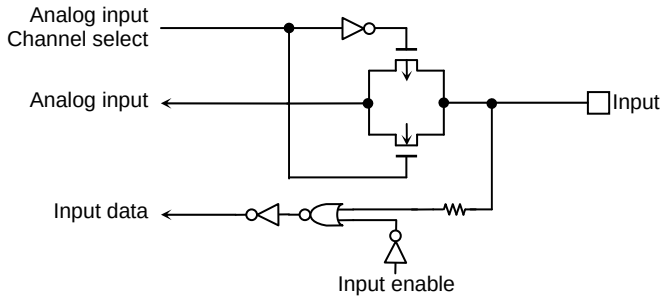
STOP : This signal becomes active 1 when the HALT mode setting register is set to the STOP mode (SYSCR2<HALTM1:0> = "01") and the CPU executes the HALT instruction. When the drive enable bit SYSCR2<DRVE> is set to "1", however STOP remains at "0".

- The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

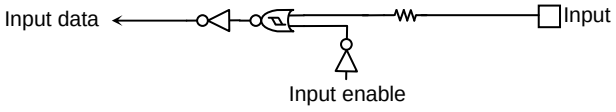
■ P0 (SEG24~SEG31), P1 (SEG16~SEG23), P2 (SEG8~SEG15), PB (SEG32~SEG39)



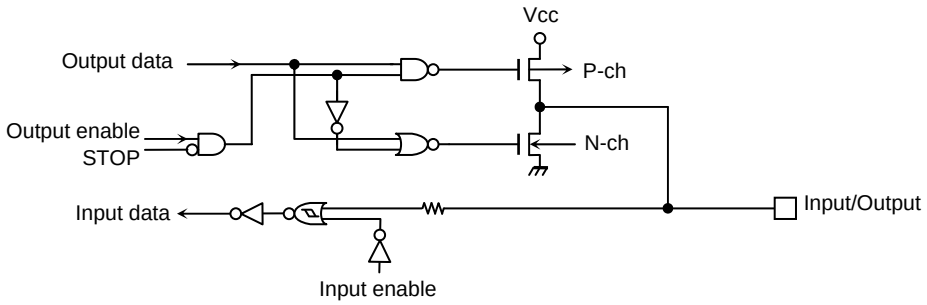
■ P5 (AN0~AN3/KWI0~KWI3)



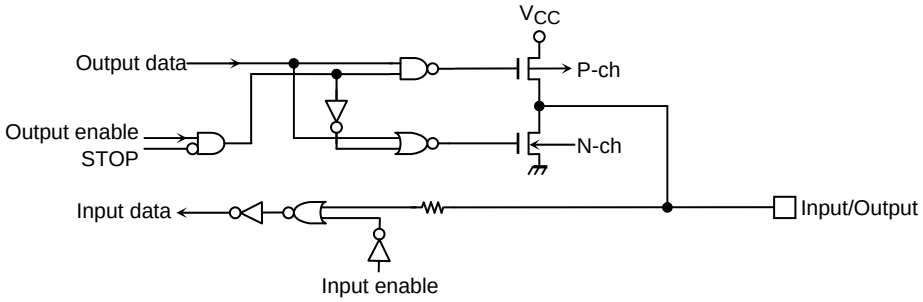
■ P60 (INT0)



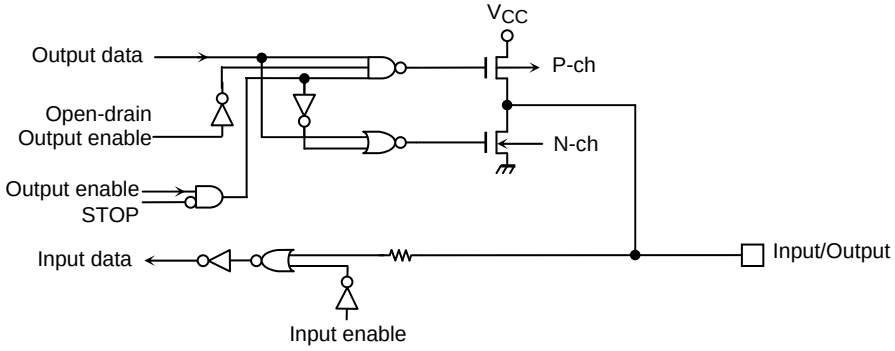
■ P61 (INT1)



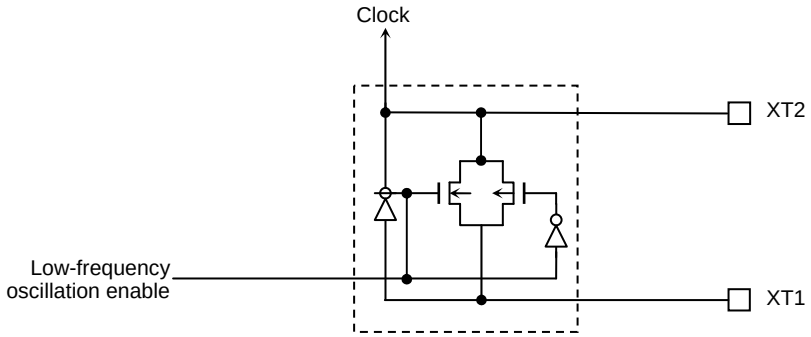
■ P62(ALARM), P70~P75(ECNT1~ECNT3, ECIN1~ECIN3), P91(RXD0), P92(SCLK0/CTS0), P94 (RXD1), P95(SCLK1/CTS1), PA1(RXD2), PA2(SCLK2/CTS2), PA4(RXD3), PA5(SCLK3/CTS3)



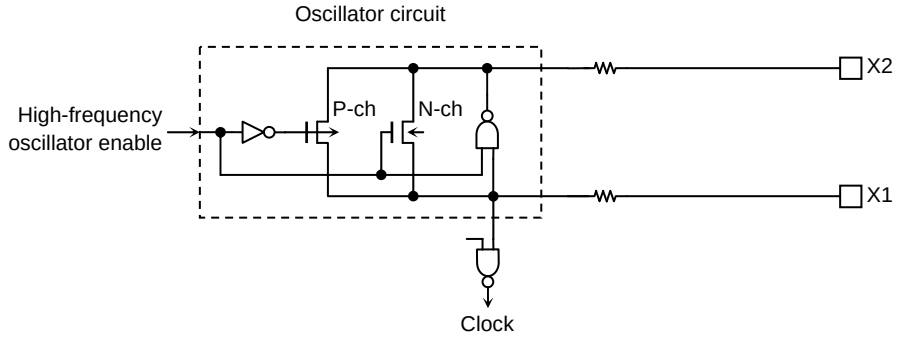
■ P80~P83(TC5OUT~TC8OUT), P90(TXD0), P93(TXD1), PA0(TXD2), PA3(TXD3)



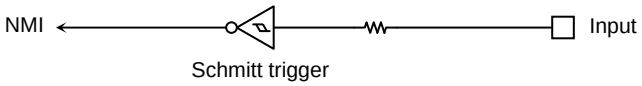
■ XT1, XT2



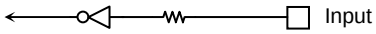
■ X1, X2



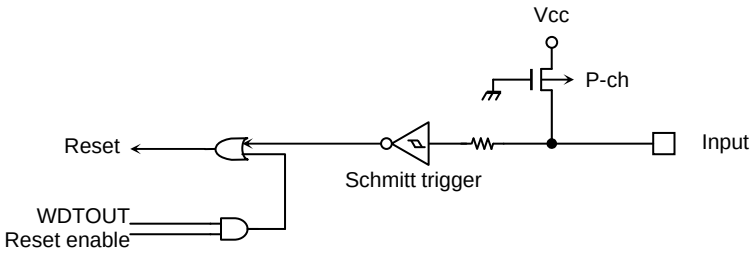
■ $\overline{\text{NMI}}$



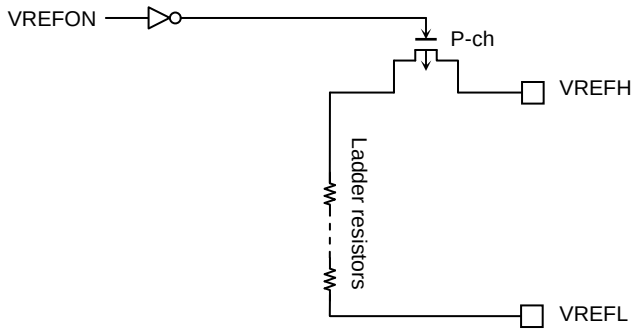
■ AM0~AM1



■ $\overline{\text{RESET}}$



■ VREFH, VREFL



7. Points to Note and Restrictions

(1) Notation

a. The notation for built-in I/O registers is as follows register symbol <Bit symbol>

e.g.) TC5CR1<TC5S> denotes bit TC5CR1 of register TC5S.

b. Read-modify-write instructions

An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

Example 1: SET 3, (TC5CR1) ... Set bit3 of TC5CR1.

Example 2: INC 1, (100H) ... Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900

Exchange instruction

EX (mem), R

Arithmetic operations

ADD	(mem), R/#	ADC	(mem), R/#
SUB	(mem), R/#	SBC	(mem), R/#
INC	#3, (mem)	DEC	#3, (mem)

Logic operations

AND	(mem), R/#	OR	(mem), R/#
XOR	(mem), R/#		

Bit manipulation operations

STCF	#3/A, (mem)	RES	#3, (mem)
SET	#3, (mem)	CHG	#3, (mem)
TSET	#3, (mem)		

Rotate and shift operations

RLC	(mem)	RRC	(mem)
RL	(mem)	RR	(mem)
SLA	(mem)	SRA	(mem)
SLL	(mem)	SRL	(mem)
RLD	(mem)	RRD	(mem)

c. fc, fs, f_{FPH}, f_{SYS} and one state

The clock frequency input on pins X1 and X2 is called fc. The clock frequency input on pins XT1 and XT2 is called fs.

The clock selected by SYSCR1<SYSCK> is called f_{FPH}. The clock frequency give by f_{FPH} divided by 2 is called f_{SYS}.

One cycle of f_{SYS} is referred to as one state.

(2) Points of note

a. AM0 and AM1 pins

This pin is connected to the DVCC pin. Do not alter the level when the pin is active.

b. EMU0 and EMU1

Open pins.

c. HALT mode (IDLE1)

When the HALT instruction is executed in IDLE1 mode (in which only the oscillator operates), the internal Special timer for CLOCK operate. When necessary, stop the circuit by setting RTCCR<RTCRUN> to "0", before the HALT instructions is executed.

d. Warm-up counter

The warm-up counter operates when STOP mode is released, even if the system is using an external oscillator. As a result a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

e. Watchdog timer

The watchdog timer starts operation immediately after a reset is released. When the watchdog timer is not to be used, disable it.

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

f. AD converter

The string resistor between the VREFH and VREFL pins can be cut by a program so as to reduce power consumption. When STOP mode is used, disable the resistor using the program before the HALT instruction is executed.

g. CPU (Micro DMA)

Only the LDC cr, r and LDC r, cr instructions can be used to access the control registers in the CPU (e.g., the transfer source address register (DMASn)).

h. Undefined SFR

The value of an undefined bit in an SFR is undefined when read.

i. POP SR instruction

Please execute the POP SR instruction during DI condition.

8. Package

LQFP100-P-1414-0.50F

Unit: mm