

Welcome to [E-XFL.COM](http://E-XFL.COM)

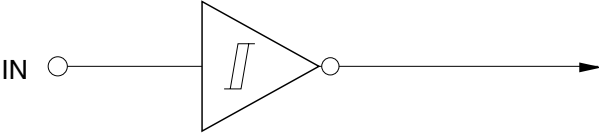
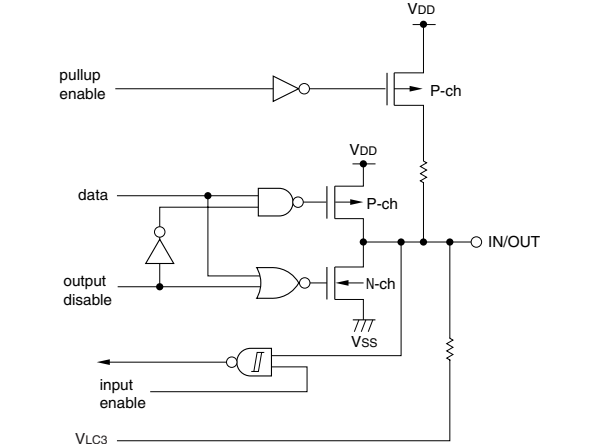
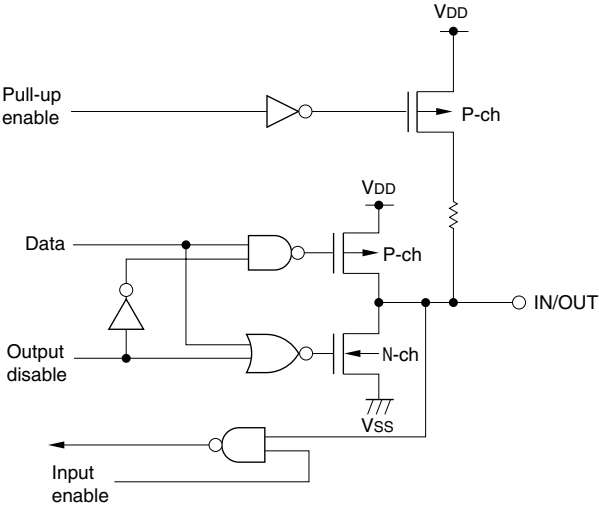
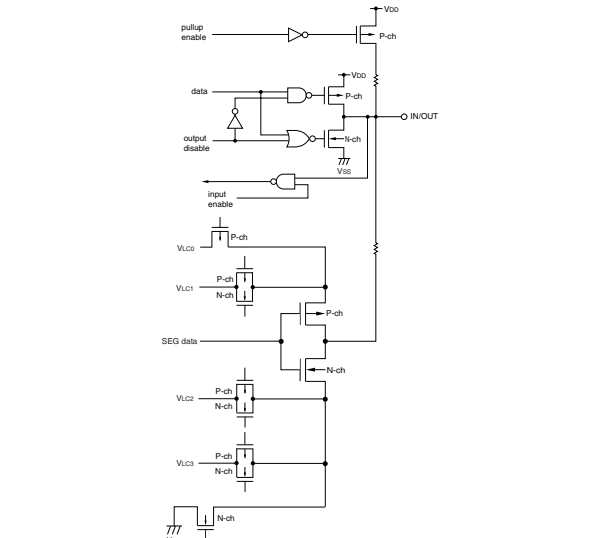
## What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "[Embedded - Microcontrollers](#)"

Details	
Product Status	Active
Core Processor	78K/0
Core Size	8-Bit
Speed	10MHz
Connectivity	3-Wire SIO, LINbus, UART/USART
Peripherals	LCD, LVD, POR, PWM, WDT
Number of I/O	34
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	52-LQFP
Supplier Device Package	-
Purchase URL	<a href="https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f0420gb-gag-ax">https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f0420gb-gag-ax</a>

Figure 2-1. Pin I/O Circuit List (1/2)

<p>Type 2</p>  <p>Schmitt-triggered input with hysteresis characteristics</p>	<p>Type 5-AO</p> 
<p>Type 5-AG</p> 	<p>Type 17-P</p> 
<p>Type 5-AH</p>	<p>Type 17-Q</p>

9

### 3.4 Operand Address Addressing

3.4.3 Direct addressing

[Function]

The memory to be manipulated is directly addressed with immediate data in an instruction word becoming an operand address.

[Operand format]

Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

MOV A, !0FE00H; when setting !addr16 to FE00H

Operation code	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	1	0	0	0	1	1	1	0	OP code
1	0	0	0	1	1	1	0			
	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	00H
0	0	0	0	0	0	0	0			
	<table><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	1	1	1	1	1	1	1	0	FEH
1	1	1	1	1	1	1	0			

[Illustration]

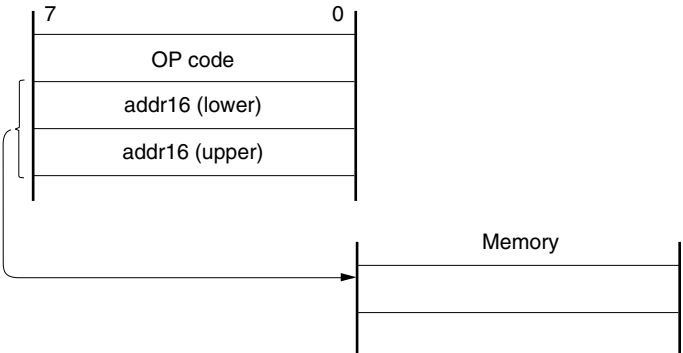
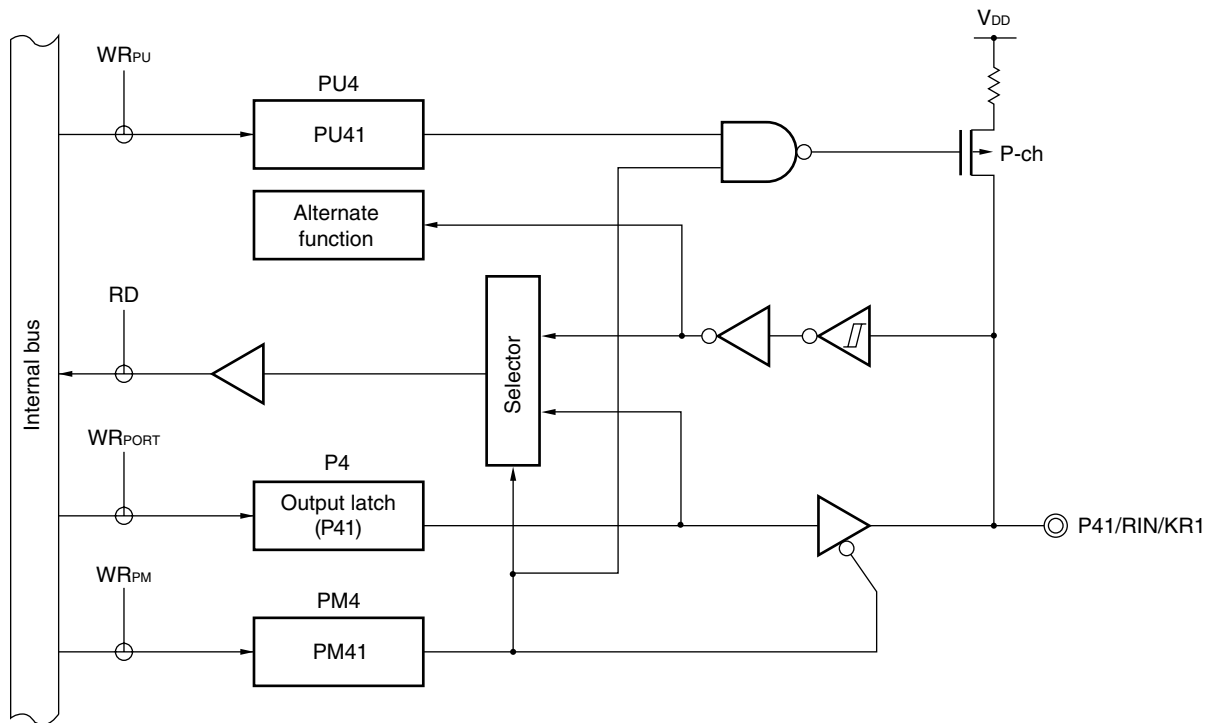


Figure 4-9. Block Diagram of P41



P4: Port register 4  
 PU4: Pull-up resistor option register 4  
 PM4: Port mode register 4  
 RD: Read signal  
 WR<sub>xx</sub>: Write signal

**(6) Port function register ALL (PFALL)**

This register sets whether to use pins P8, P10, P11, 14 and P15 as port pins (other than segment output pins) or segment output pins.

PFALL is set using a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets PFALL to 00H.

**Figure 4-25. Format of Port Function Register ALL (PFALL)**

Address: FFB6H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PFALL	0	PF15ALL	PF14ALL	0	PF11ALL	PF10ALL	0	PF08ALL

PFnALL	Port/segment output specification
0	Used as port (other than segment output)
1	Used as segment output

**Remark** n = 08, 10, 11, 14, 15

**(3) Internal low-speed oscillation clock (clock for watchdog timer)**

- **Internal low-speed oscillator**

This circuit oscillates a clock of  $f_{RL} = 240 \text{ kHz}$  (TYP.). After a reset release, the internal low-speed oscillation clock always starts operating.

Oscillation can be stopped by using the internal oscillation mode register (RCM) when “internal low-speed oscillator can be stopped by software” is set by option byte.

The internal low-speed oscillation clock cannot be used as the CPU clock. The following hardware operates with the internal low-speed oscillation clock.

- Watchdog timer
- 8-bit timer H1 (if  $f_{RL}$ ,  $f_{RL}/2^7$  or  $f_{RL}/2^9$  is selected as the count clock)
- LCD controller/driver (if  $f_{RL}/2^9$  is selected as the LCD source clock)

**Remark**  $f_{RL}$ : Internal low-speed oscillation clock frequency

**5.2 Configuration of Clock Generator**

The clock generator includes the following hardware.

**Table 5-1. Configuration of Clock Generator**

Item	Configuration
Control registers	Clock operation mode select register (OSCCTL) Processor clock control register (PCC) Internal oscillation mode register (RCM) Main OSC control register (MOC) Main clock mode register (MCM) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) Internal high-speed oscillation trimming register (HIOTRM)
Oscillators	X1 oscillator XT1 oscillator Internal high-speed oscillator Internal low-speed oscillator

**(8) Oscillation stabilization time select register (OSTS)**

This register is used to select the X1 clock oscillation stabilization wait time when the STOP mode is released.

When the X1 clock is selected as the CPU clock, the operation waits for the time set using OSTS after the STOP mode is released.

When the internal high-speed oscillation clock is selected as the CPU clock, confirm with OSTC that the desired oscillation stabilization time has elapsed after the STOP mode is released. The oscillation stabilization time can be checked up to the time set using OSTC.

OSTS can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets OSTS to 05H.

**Figure 5-8. Format of Oscillation Stabilization Time Select Register (OSTS)**

Address: FFA4H After reset: 05H R/W

Symbol	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0		Oscillation stabilization time selection		
				$f_x = 2 \text{ MHz}$	$f_x = 5 \text{ MHz}$	$f_x = 10 \text{ MHz}$
0	0	1	$2^{11}/f_x$	1.02 ms	409.6 $\mu\text{s}$	204.8 $\mu\text{s}$
0	1	0	$2^{13}/f_x$	4.10 ms	1.64 ms	819.2 $\mu\text{s}$
0	1	1	$2^{14}/f_x$	8.19 ms	3.27 ms	1.64 ms
1	0	0	$2^{15}/f_x$	16.38 ms	6.55 ms	3.27 ms
1	0	1	$2^{16}/f_x$	32.77 ms	13.11 ms	6.55 ms
Other than above			Setting prohibited			

**Cautions 1.** To set the STOP mode when the X1 clock is used as the CPU clock, set OSTS before executing the STOP instruction.

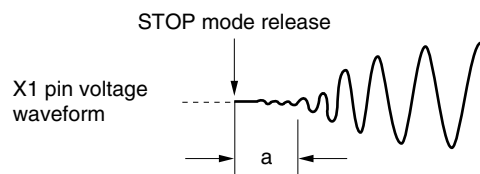
**2.** Do not change the value of the OSTS register during the X1 clock oscillation stabilization time.

**3.** The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- Desired OSTC oscillation stabilization time  $\leq$  Oscillation stabilization time set by OSTS

**Note,** therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

**4.** The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).



**Remark**  $f_x$ : X1 clock oscillation frequency



### 6.4.2 Square wave output operation

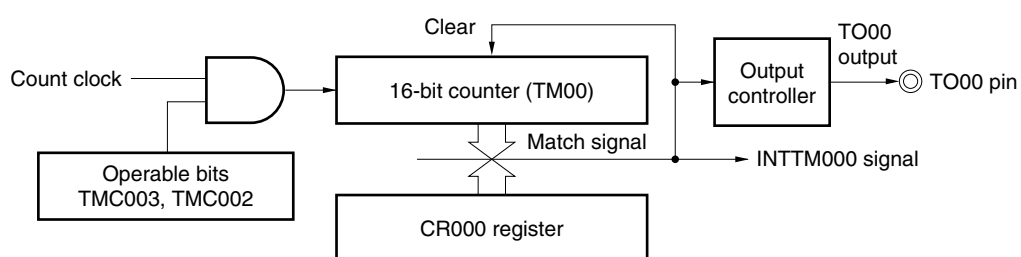
When 16-bit timer/event counter 00 operates as an interval timer (see 6.4.1), a square wave can be output from the TO00 pin by setting the 16-bit timer output control register 00 (TOC00) to 03H.

When TMC003 and TMC002 are set to 11 (count clear & start mode entered upon a match between TM00 and CR000), the counting operation is started in synchronization with the count clock.

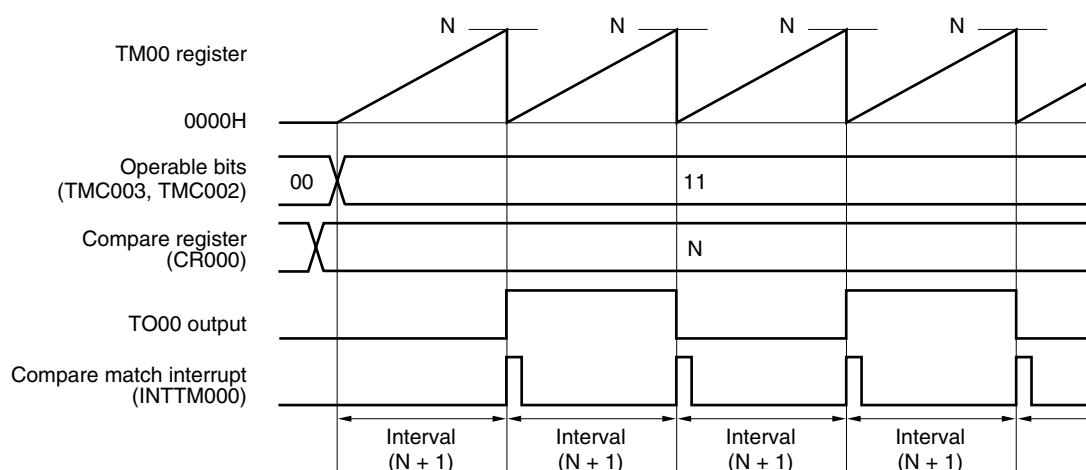
When the value of TM00 later matches the value of CR000, TM00 is cleared to 0000H, an interrupt signal (INTTM000) is generated, and TO00 output is inverted. This TO00 output that is inverted at fixed intervals enables TO00 to output a square wave.

- Remarks**
1. For the setting of I/O pins, see 6.3 (6) **Port mode register 3 (PM3)**.
  2. For how to enable the INTTM000 signal interrupt, see **CHAPTER 19 INTERRUPT FUNCTIONS**.

**Figure 6-16. Block Diagram of Square Wave Output Operation**



**Figure 6-17. Basic Timing Example of Square Wave Output Operation**



**Figure 6-45. Example of Register Settings for One-Shot Pulse Output Operation (2/2)**

**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

**(f) 16-bit capture/compare register 000 (CR000)**

This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR000, an interrupt signal (INTTM000) is generated and the TO00 output level is inverted.

**(g) 16-bit capture/compare register 010 (CR010)**

This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR010, an interrupt signal (INTTM010) is generated and the TO00 output level is inverted.

**Caution** Do not set the same value to CR000 and CR010.

#### 6.4.9 External 24-bit event counter operation

16-bit timer/event counter 00 can be operated to function as an external 24-bit event counter, by connecting 16-bit timer/event counter 00 and 8-bit timer/event counter 52 in cascade, and using the external event counter function of 8-bit timer/event counter 52.

It operates as an external 24-bit event counter, by counting the number of external clock pulses input to the TI52 pin via 8-bit timer counter 52 (TM52), and counting the signal which has been output upon a match between the TM52 count value and 8-bit timer compare register 52 (CR52 = FFH<sup>Note</sup>) via 16-bit timer counter 00 (TM00).

When using 16-bit timer/event counter 00 as an external 24-bit event counter, external event input enable can be controlled via 8-bit timer counter H2 output.

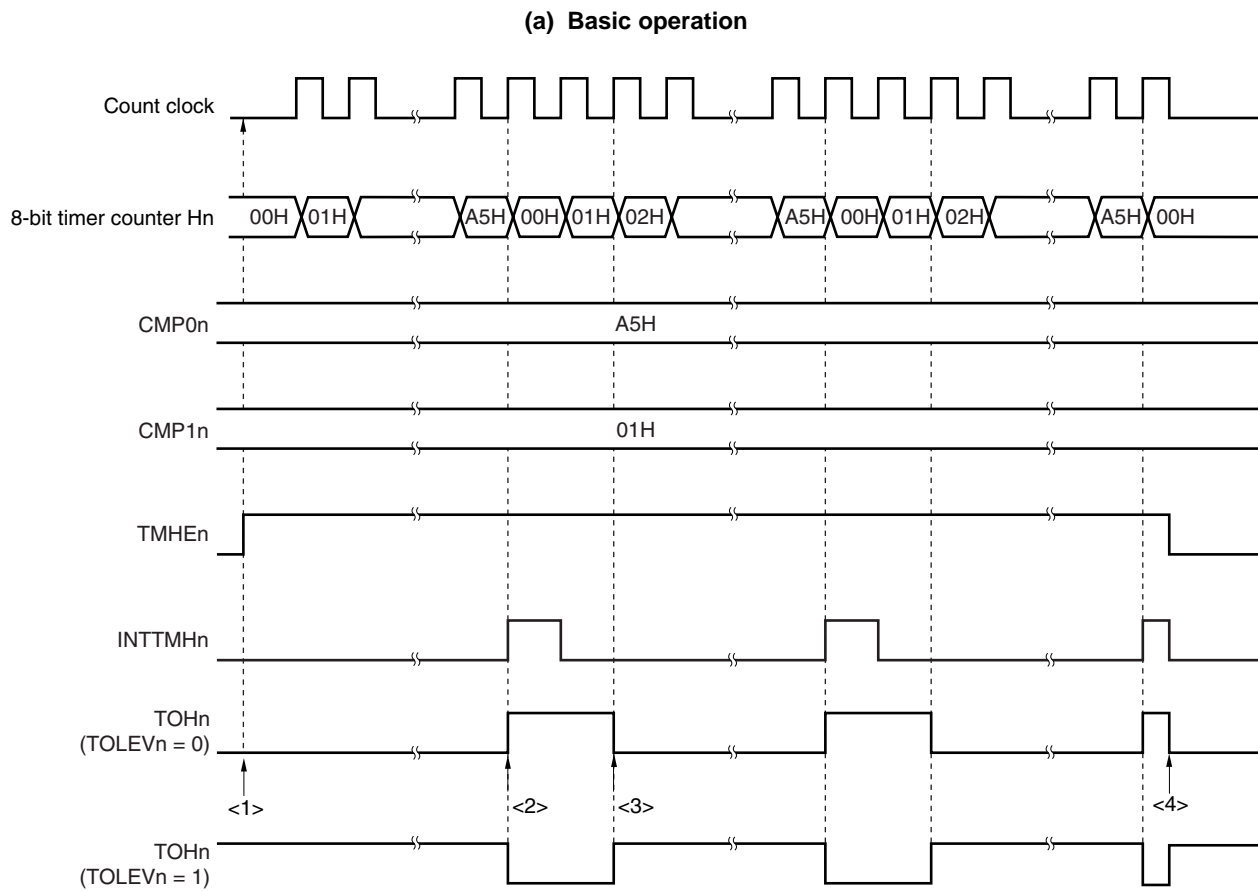
The valid edge of the input to the TI52 pin can be specified by timer clock selection register 52 (TCL52) of 8-bit timer counter 52 (TM52). Also, input enable for TM52 external event input can be controlled via 8-bit timer counter H2 output, by setting bit 2 (ISC2) of the input switch control register (ISC) to "1".

Count operation using 8-bit timer 52 output as the count clock is started, by setting bits 2, 1, and 0 (PRM002, PRM001, and PRM000) of prescaler mode register 00 (PRM00) of 16-bit timer/event counter 00 to "1", "1", and "1" (TM52 output is selected as a count clock), and bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) to "1" and "1" (count clear & start mode entered upon a match between TM00 and CR000). TM00 is cleared to "0" and an interrupt request signal (INTTM000) is generated upon a match between the TM00 count value and 16-bit timer compare register 000 (CR000) value.

Subsequently, INTTM000 is generated upon every match between the TM00 and CR000 values.

**Note** When operating 16-bit timer/event counter 00 as an external 24-bit event counter, the 8-bit timer compare register 52 (CR52) value must be set to FFH. Also, the TM52 interrupt request signal (INTTM52) must be masked (TMMK52 = 1).

Figure 8-14. Operation Timing in PWM Output Mode (1/4)



- <1> The count operation is enabled by setting the TMHEn bit to 1. Start 8-bit timer counter Hn by masking one count clock to count up. At this time, PWM output outputs an inactive level.
- <2> When the values of 8-bit timer counter Hn and the CMP0n register match, an active level is output. At this time, the value of 8-bit timer counter Hn is cleared, and the INTTMHn signal is output.
- <3> When the values of 8-bit timer counter Hn and the CMP1n register match, an inactive level is output. At this time, the 8-bit counter value is not cleared and the INTTMHn signal is not output.
- <4> Clearing the TMHEn bit to 0 during timer Hn operation sets the INTTMHn signal to the default and PWM output to an inactive level.

**Remark** n = 0 to 2, however, TOH0 and TOH1 only for TOHn

**(13) Watch error correction register (SUBCUD)**

This register is used to correct the count value of the sub-count register (RSUBC).

SUBCUD can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-14. Format of Watch Error Correction Register (SUBCUD)**

Address: FF82H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SUBCUD	DEV	F6	F5	F4	F3	F2	F1	F0

DEV	Setting of watch error correction timing
0	Corrects watch error when the second digits are at 00, 20, or 40.
1	Corrects watch error only when the second digits are at 00.

F6	Setting of watch error correction method
0	Increases by $\{(F5, F4, F3, F2, F1, F0) - 1\} \times 2$ .
1	Decreases by $\{(/F5, /F4, /F3, /F2, /F1, /F0) + 1\} \times 2$ .
When $(F6, F5, F4, F3, F2, F1, F0) = (*, 0, 0, 0, 0, 0, *)$ , the watch error is not corrected. /F5 to /F0 are the inverted values of the corresponding bits (000011 when 111100).	

**(14) Alarm minute register (ALARMWM)**

This register is used to set minutes of alarm.

ALARMWM can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

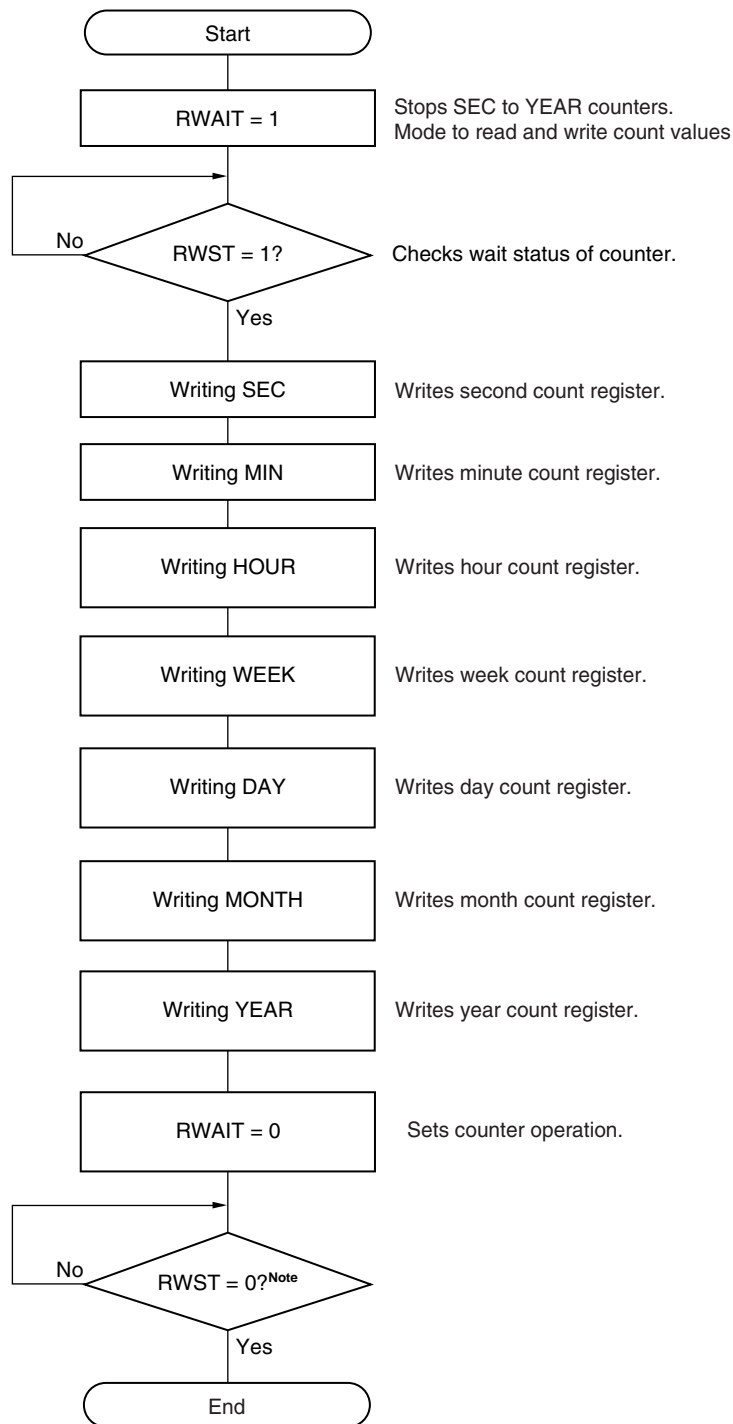
**Caution** Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is set, the alarm is not detected.

**Figure 9-15. Format of Alarm Minute Register (ALARMWM)**

Address: FF86H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ALARMWM	0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

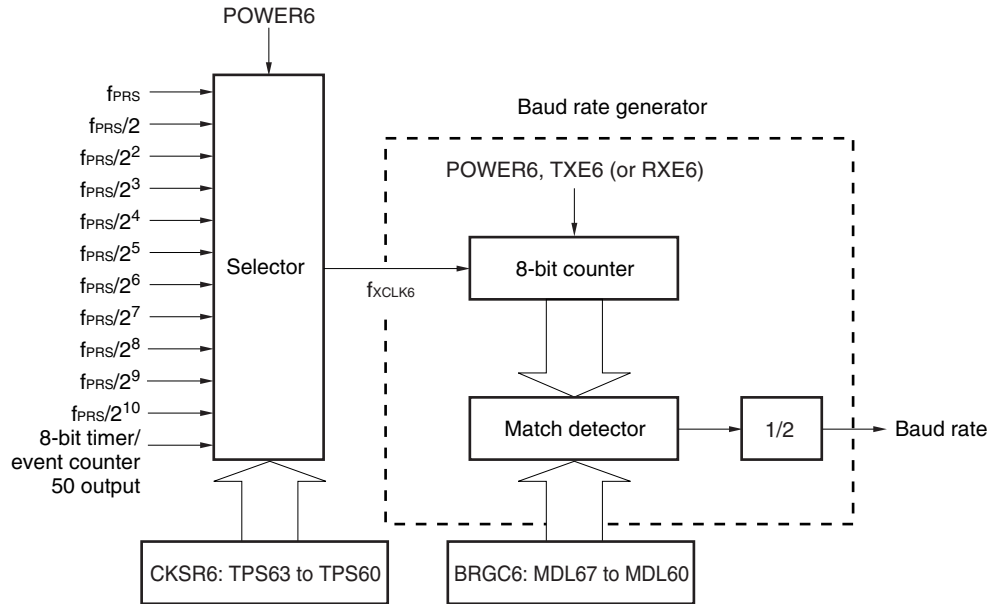
Figure 9-20. Procedure for Writing Real-Time Counter



**Note** Be sure to confirm that RWST = 0 before setting STOP mode.

**Caution** Complete the series of operations of setting RWAIT to 1 to clearing RWAIT to 0 within 1 second.

**Remark** SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR may be written in any sequence.  
All the registers do not have to be set and only some registers may be written.

**Figure 14-26. Configuration of Baud Rate Generator**

**Remark** **POWER6:** Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)

**TXE6:** Bit 6 of ASIM6

**RXE6:** Bit 5 of ASIM6

**CKSR6:** Clock selection register 6

**BRGC6:** Baud rate generator control register 6

## (2) Generation of serial clock

A serial clock to be generated can be specified by using clock selection register 6 (CKSR6) and baud rate generator control register 6 (BRGC6).

The clock to be input to the 8-bit counter can be set by bits 3 to 0 (TPS63 to TPS60) of CKSR6 and the division value ( $f_{XCLK6}/4$  to  $f_{XCLK6}/255$ ) of the 8-bit counter can be set by bits 7 to 0 (MDL67 to MDL60) of BRGC6.

## CHAPTER 15 SERIAL INTERFACE CSI10

### 15.1 Functions of Serial Interface CSI10

Serial interface CSI10 has the following two modes.

#### (1) Operation stop mode

This mode is used when serial communication is not performed and can enable a reduction in the power consumption.

For details, see **15.4.1 Operation stop mode**.

#### (2) 3-wire serial I/O mode (MSB/LSB-first selectable)

This mode is used to communicate 8-bit data using three lines: a serial clock line ( $\overline{\text{SCK10}}$ ) and two serial data lines (SI10 and SO10).

The processing time of data communication can be shortened in the 3-wire serial I/O mode because transmission and reception can be simultaneously executed.

In addition, whether 8-bit data is communicated with the MSB or LSB first can be specified, so this interface can be connected to any device.

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.

For details, see **15.4.2 3-wire serial I/O mode**.

### 15.2 Configuration of Serial Interface CSI10

Serial interface CSI10 includes the following hardware.

**Table 15-1. Configuration of Serial Interface CSI10**

Item	Configuration
Controller	Transmit controller Clock start/stop controller & clock phase controller
Registers	Transmit buffer register 10 (SOTB10) Serial I/O shift register 10 (SIO10)
Control registers	Serial operation mode register 10 (CSIM10) Serial clock selection register 10 (CSIC10) Port function register 1 (PF1) Port mode register 1 (PM1) Port register 1 (P1)



### (3) Remote controller receive control register (RMCN)

This register is used to enable/disable remote controller reception and to set the noise elimination width, clock internal division, input invert signal, and source clock.

RMCN is set with a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets RMCN to 00H.

**Figure 18-5. Format of Remote Controller Receive Control Register (RMCN)**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
RMCN	RMEN	NCW	PRSEN	RMIN	0	0	RMCK1	RMCK0	FF9AH	00H	R/W

RMEN	Control of remote controller receive operation
0	Disable remote controller reception
1	Enable remote controller reception

NCW	Noise elimination width control signal
0	Eliminate noise less than $1/f_{REMPRS}$
1	Eliminate noise less than $2/f_{REMPRS}$

PRSEN	Internal clock division control signal
0	Clock not divided internally ( $f_{REMPRS} = f_{REM}$ )
1	Clock internally divided into two ( $f_{REMPRS} = f_{REM}/2$ )

RMIN	Remote controller input invert signal
0	Input positive phase
1	Input negative phase

RMCK1	RMCK0	Selection of source clock ( $f_{REM}$ ) of remote controller counter
0	0	$f_{PRS}/2^6$ (156.25 kHz)
0	1	$f_{PRS}/2^7$ (78.125 kHz)
1	0	$f_{PRS}/2^8$ (39.063 kHz)
1	1	$f_{SUB}$ (32.768 kHz)

**Cautions** 1. Always set bits 2 and 3 to 0.

2. To change the values of NCW, PRSEN, RMIN, RMCK1, and RMCK0, disable remote controller reception (RMEN = 0) first.

**Remarks** 1.  $f_{REM}$ : Source clock of remote controller counter (selected by bits 0 and 1 (RMCK0 and RMCK1))

2.  $f_{REMPRS}$ : Operation clock inside remote controller receiver

3.  $f_{PRS}$ : Peripheral hardware clock frequency

4.  $f_{SUB}$ : Oscillation frequency of subsystem clock

5. The parenthesized values apply to operation at  $f_{PRS} = 10$  MHz and  $f_{SUB} = 32.768$  kHz.

Table 22-2. Hardware Statuses After Reset Acknowledgment (1/4)

Hardware		After Reset Acknowledgment <sup>Note 1</sup>
Program counter (PC)		The contents of the reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined <sup>Note 2</sup>
	General-purpose registers	Undefined <sup>Note 2</sup>
Port registers (P1 to P4, P8, P10 to P12, P14, P15) (output latches)		00H
Port mode registers (PM1 to PM4, PM8, PM10 to PM12, PM14, PM15)		FFH
Pull-up resistor option registers (PU1, PU3, PU4, PU8, PU10 to PU12, PU14, PU15)		00H
Port function register (PF1)		00H
Port function register (PF2)		00H
Port function register (PFALL)		00H
Internal memory size switching register (IMS)		CFH <sup>Note 3</sup>
Clock operation mode select register (OSCCTL)		00H
Processor clock control register (PCC)		01H
Internal oscillation mode register (RCM)		80H
Main OSC control register (MOC)		80H
Main clock mode register (MCM)		00H
Oscillation stabilization time counter status register (OSTC)		00H
Oscillation stabilization time select register (OSTS)		05H
Internal high-speed oscillation trimming register (HIOTRM)		10H
16-bit timer/event counters 00	Timer counters 00 (TM00)	0000H
	Capture/compare registers 000, 010 (CR000, CR010)	0000H
	Mode control registers 00 (TMC00)	00H
	Prescaler mode registers 00 (PRM00)	00H
	Capture/compare control registers 00 (CRC00)	00H
	Timer output control registers 00 (TOC00)	00H
8-bit timer/event counters 50, 51, 52	Timer counters 50, 51, 52 (TM50, TM51, TM52)	00H
	Compare registers 50, 51, 52 (CR50, CR51, CR52)	00H
	Timer clock selection registers 50, 51, 52 (TCL50, TCL51, TCL52)	00H
	Mode control registers 50, 51, 52 (TMC50, TMC51, TMC52)	00H

- Notes**
1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.
  2. When a reset is executed in the standby mode, the pre-reset status is held even after reset.
  3. The initial values of the internal memory size switching register (IMS) after a reset release are constant (IMS = CFH) in all the 78K0/LD3 products, regardless of the internal memory capacity. Therefore, after a reset is released, be sure to set the following values for each product.

Flash Memory Version (78K0/LD3)	IMS	ROM Capacity	Internal High-Speed RAM Capacity
μPD78F0420, 78F0430	42H	8 KB	512 bytes
μPD78F0421, 78F0431	04H	16 KB	768 bytes
μPD78F0422, 78F0432	C6H	24 KB	1 KB
μPD78F0423, 78F0433	C8H	32 KB	

## 26.5 Connection of Pins on Board

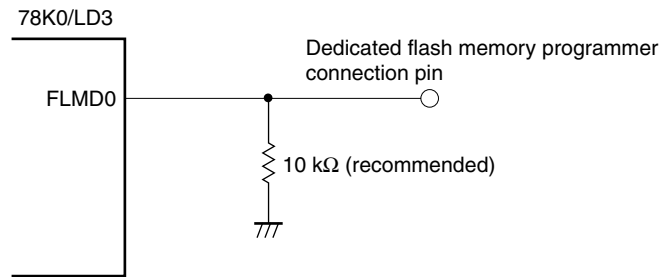
To write the flash memory on-board, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be handled as described below.

### 26.5.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the  $V_{DD}$  write voltage is supplied to the FLMD0 pin. An FLMD0 pin connection example is shown below.

**Figure 26-7. FLMD0 Pin Connection Example**



### 26.5.2 Serial interface pins

The pins used by each serial interface are listed below.

**Table 26-4. Pins Used by Each Serial Interface**

Serial Interface	Pins Used
CSI10	SO10, SI10, $\overline{SCK10}$
UART6	TxD6, RxD6

To connect the dedicated flash memory programmer to the pins of a serial interface that is connected to another device on the board, care must be exercised so that signals do not collide or that the other device does not malfunction.

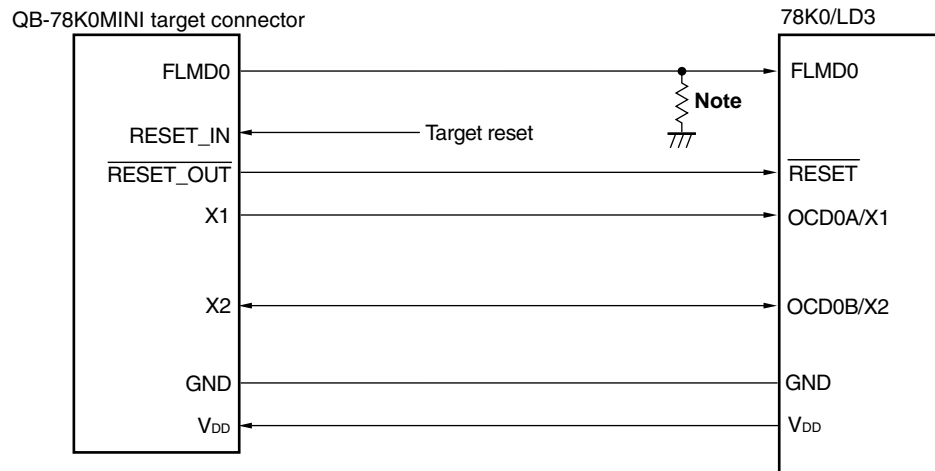
## CHAPTER 27 ON-CHIP DEBUG FUNCTION

### 27.1 Connecting QB-78K0MINI to 78K0/LD3

The 78K0/LD3 uses the  $V_{DD}$ , FLMD0,  $\overline{\text{RESET}}$ , OCD0A/X1, OCD0B/X2, and  $V_{SS}$  pins to communicate with the host machine via an on-chip debug emulator (QB-78K0MINI).

**Caution** The 78K0/LD3 has an on-chip debug function. Do not use this product for mass production because its reliability cannot be guaranteed after the on-chip debug function has been used, given the issue of the number of times the flash memory can be rewritten. NEC Electronics does not accept complaints concerning this product after the on-chip debug function has been used.

**Figure 27-1. Connection Example of QB-78K0MINI and 78K0/LD3  
(When OCD0A/X1 and OCD0B/X2 Are Used)**

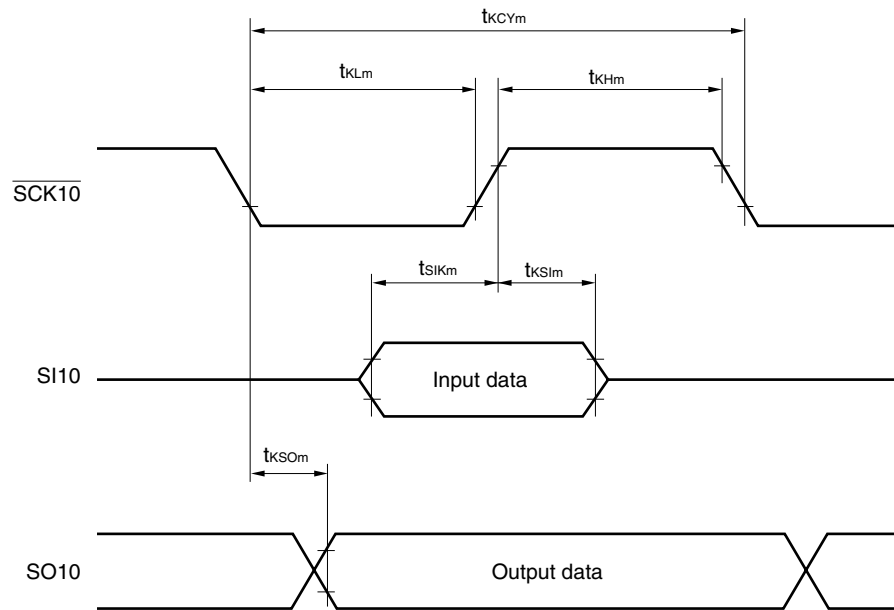


**Note** Make pull-down resistor 470  $\Omega$  or more (10 k $\Omega$ : recommended).

**Caution** Input the clock from the OCD0A/X1 pin during on-chip debugging.

## Serial Transfer Timing

CSI10:

Remark  $m = 1, 2$