FUJITSU SEMICONDUCTOR

CONTROLLER MANUAL

CM25-10147-1E

F^2MC-8L

8-BIT MICROCONTROLLER

MB89470 Series HARDWARE MANUAL

FUJITSU

F^2MC-8L

8-BIT MICROCONTROLLER

MB89470 Series

HARDWARE MANUAL

FUJITSU LIMITED

PREFACE

■ Objectives and Intended Reader

The MB89470 series has been developed as a general-purpose version of the F2MC*-8L family consisting of proprietary 8-bit, single-chip microcontrollers. The MB89470 series is applicable to home appliance as well as in a wide range of applications for consumer product.

This manual describes the functions and operation of the MB89470 series and is aimed at engineers using the MB89470 series of microcontrollers to develop actual products. See the F^2MC-8L MB89600 Series Programming Manual for details on the MB89470 instruction set.

■ Trademarks

F^2MC is the abbreviation of FUJITSU Flexible Microcontroller.

Other system and product names in this manual are trademarks of respective companies or organizations.

■ Structure of this manual

This manual contains the following 15 chapters:

CHAPTER 1 "OVERVIEW"

This chapter describes the features and basic specifications of the MB89470 series.

CHAPTER 2 "HANDLING DEVICE"

This chapter describes points to note when using the general-purpose single-chip microcontroller.

CHAPTER 3 "CPU"

This chapter describes the functions and operation of the CPU.

CHAPTER 4 "I/O PORTS"

This chapter describes the functions and operation of the I/O ports.

CHAPTER 5 "TIMEBASE TIMER"

This chapter describes the functions and operation of the timebase timer.

CHAPTER 6 "WATCHDOG TIMER"

This chapter describes the functions and operation of the watchdog timer.

CHAPTER 7 "WATCH PRESCALER"

This chapter describes the functions and operation of the watch prescaler.

CHAPTER 8 "8-BIT PWM TIMER"

This chapter describes the functions and operation of the 8-bit PWM timer.

CHAPTER 9 "PULSE WIDTH COUNT TIMER (PWC)"

This chapter describes the functions and operation of the pulse width count timer (PWC).

CHAPTER 10 "8/16-BIT TIMER/COUNTER"

This chapter describes the functions and operation of the 8/16-bit Timer/Counter.

CHAPTER 11 "EXTERNAL INTERRUPT 1 CIRCUIT (EDGE)"

This chapter describes the functions and operation of the external interrupt circuit.

CHAPTER 12 "EXTERNAL INTERRUPT 2 CIRCUIT (LEVEL)"

This chapter describes the functions and operation of the external interrupt 2 circuit (level).

CHAPTER 13 "A/D CONVERTER"

This chapter describes the functions and operation of the A/D converter.

CHAPTER 14 "UART/SIO"

This chapter describes the functions and operation of the UART/SIO.

CHAPTER 15 "BUZZER OUTPUT"

This chapter describes the functions and operation of the buzzer output.

APPENDIX

This appendix includes I/O maps, instruction lists, and other information.

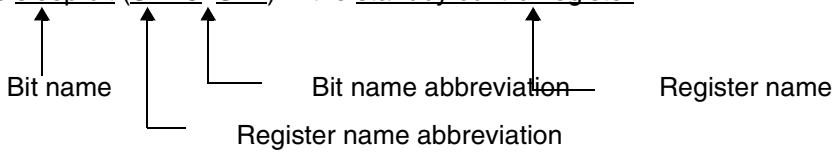
- The contents of this document are subject to change without notice.
  Customers are advised to consult with FUJITSU sales representatives before ordering.
- The information, such as descriptions of function and application circuit examples, in this document are presented solely for the purpose of reference to show examples of operations and uses of Fujitsu semiconductor device; Fujitsu does not warrant proper operation of the device with respect to use based on such information. When you develop equipment incorporating the device based on such information, you must assume any responsibility arising out of such use of the information. Fujitsu assumes no liability for any damages whatsoever arising out of the use of the information.
- Any information in this document, including descriptions of function and schematic diagrams, shall not be construed as license of the use or exercise of any intellectual property right, such as patent right or copyright, or any other right of Fujitsu or any third party or does Fujitsu warrant non-infringement of any third-party' s intellectual property right or other right by using such information. Fujitsu assumes no liability for any infringement of the intellectual property rights or other rights of third parties which would result from the use of information contained herein.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
  Please note that Fujitsu will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.

READING THIS MANUAL

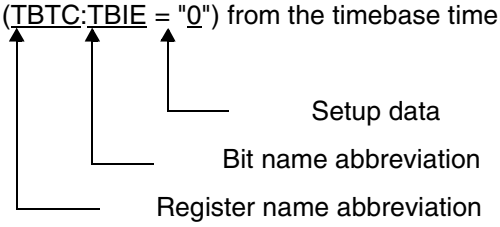
■ Naming conventions for register name and pin name

● Example for description of register name and bit name

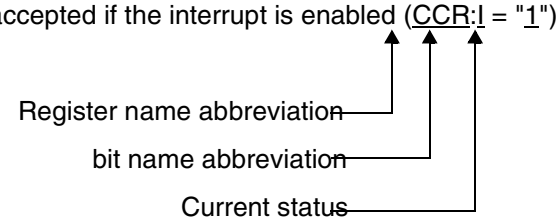
By writing "1" to the sleep bit (STBC: SLP) in the standby control register ...



Disable the interrupt request output (TBTC:TBIE = "0") from the timebase timer.



Interrupt is accepted if the interrupt is enabled (CCR:I = "1")



● Notations for shared pins

P01/AN1 pin

Many of the pins in the devices of this series are multi-function pins. (They can be switched between two or more functions under program control.) The multiple names of these pins (indicating their multiple functions) are separated by slant bars (/).

■ Development Tools and Other Resources Required for Development

The following items are required for developing using the MB89470 series.

Contact FUJITSU sales staff for the required development tools and other resources.

● Manuals required for development

[Check field]

- ☐ F^2MC-8L MB89470 Series Data Sheet
  (Provides electrical characteristics and various characteristic examples for the device.)
- ☐ F^2MC-8L MB89600 Series Programming Manual
  (manual including instructions for the F^2MC-8L family.
- * F^2MC Family Softune C Compiler Manual
  (required only if C language is used for development) (manual describing how to develop and activate programs in the C language)
- * F^2MC Family Softune Assembler Manual for V3
  (manual describing program development using the assembler language)
- * F^2MC Family Softune Linkage Kit Manual for V3
  (manual describing functions and operations of the linker, and library manager
- * F^2MC Family Softune Workbench Operation Manual
- * F^2MC Family Softune Workbench User's Manual
- * F^2MC Family Softune Command Reference Manual

Manuals marked with * are provided with the products.

In addition, manuals for products such as development tools are provided with the product.

● Tool required for development

[Check field]

- ☐ Softune (Workbench, C compiler, Assembler, Linkage kit)

For details, see the Product Guide.

❍ **Items required for evaluation using one-time PROM microcontrollers (when performing your own PROM programming)**

[Check field]

☐ MB89P475

☐ One-time PROM programmer
See the data sheet for details of recommended programmers.

● Development tools

[Check field]

☐ MB89PV470 (piggyback/evaluation device)

☐ Evaluation tool

Main unit	Pod	Probe
MB2141A + MB2144-508		MB2144-203

Check with the supplier when using a third party development environment.

● Reference material

- "F^{2}MC Development Tool Catalog"
- "Microcomputer Product Guide"

CONTENTS

CHAPTER 1
OVERVIEW

This chapter describes the features and basic specifications of the MB89470 series.

1.1 MB89470 Series Features

The MB89470 series has been developed as a general-purpose version of the F^{2}MC-8L family consisting of proprietary 8-bit, single-chip microcontrollers.
In addition to a compact instruction set, the microcontrollers contains a variety of peripheral functions such as 21-bit time-base timer, watch prescaler, PWC timer, PWM timer, 8/16-bit timer/counter, external interrupt 1 (edge), external interrupt 2 (level), 10-bit A/D converter, UART/SIO, buzzer, watchdog timer reset.
The MB89470 series is designed suitable for home appliance as well as in a wide range of applications for consumer product.

■ MB89470 series features

● Supports high speed operation at low voltage

Minimum instruction execution time: 0.32µs at 12.5MHz

● Package used

- QFP package and SH-DIP package for MB89P475, MB89475
- MQFP package for MB89PV470

● F^{2}MC-8L CPU core

Instructions set optimized for controller applications

- Multiplication and division instructions
- 16-bit operations
- Bit-test branch instructions
- Bit manipulation instructions, etc.

● Six timers sub-systems

- PWC timer (also ussable as a interval timer)
- PWM timer
- 8/16-bit timer/counter 11,12
- 8/16-bit timer/counter 21,22
- 21-bit timebase timer
- watch prescaler

● External interrupts systems

- Edge detection (Selectable edge) : 4 channels
- Low-level interrupt (Wake-up function) : 5 channels

- A/D converter (8 channels)
  - 10-bit successive approximation type
- UART / SIO
  - Synchronous/asynchronous data transfer capable
- Buzzer
  - 7 frequency type are selectable by software
- Low-power modes
  - Stop mode (Oscillation stops to minimize the current consumption.)
  - Sleep mode (The CPU stops to reduce the current consumption to approx. 1/3 of normal.)
  - Subclock mode (for dual clock product)
  - Watch mode (for dual clock product)
- Watch dog timer reset
- I/O ports
  - Max 39 channels

1.2 MB89470 Series Product Range

The MB89470 series contains 3 different models. Table 1.2-1 "MB89470 series product range" lists the product range and Table 1.2-2 "Common specifications for the MB89470 series" lists the common specifications.

■ MB89470 series product range

Table 1.2-1 MB89470 series product range

Parameter	Model number		
	MB89475	MB89P475	MB89PV470
Class	Mass Production Products (mask ROM product)	OTP	Piggy-back
ROM capacity	16K x 8-bit (internal ROM)	16K x 8-bit (internal PROM with read protection) *1	32K x 8-bit (external ROM) *2
RAM capacity	512 x 8 bits		1K x 8-bit
Low-power consumption (stand by modes)	Sleep mode, stop mode, subclock mode, watch mode		
Process	CMOS		
Operating voltage	2.2V to 5.5V	3.5V to 5.5V	2.7V to 5.5V

*1: Read protection is selected by part number. Detail please refer to "Appendix C Mask Options"
*2: Use MBM27C256A as the external ROM.

Table 1.2-2 Common specifications for the MB89470 series

Parameter		Specification
CPU functions		Number of instructions: 136 instructions Instruction bit length: 8-bit Instruction length : 1 to 3 bytes Data bit length : 1, 8, or 16-bit Minimum instruction execution time: 0.32 µs/12.5 MHz Minimum Interrupt processing time: 2.88 µs/12.5 MHz
Peripheral functions	Ports	Output only ports (N-Channel open drain): 7 pins Input-only ports: 3 pins (1 pin in product with dual clock) I/O ports (CMOS): 29 pins Total: 39 pins
	21-Bit Time-base timer	Interrupt period (0.66ms, 2.6 ms, 21.0 ms, 335.5 ms) at 12.5 MHz
	Watchdog timer	Reset period (167.8 ms to 335.5 ms) at 12.5 MHz Reset period (500 ms to 1000 ms) at 32.768 kHz for subclock
	Watch prescalar	17 bits Interrupt cycle: 31.25 ms, 0.25 s, 0.5 s, 1.00 s, 2.00 s, 4.00 s, / 32.768 kHz for subclock
	Pulse width count timer	2 channels 8-bit one-shot timer operation (supports underflow output, operating clock period: $1t_{inst}$, $4t_{inst}$, $32t_{inst}$, external) 8-bit reload timer operation (supports square wave output, operating clock period: $1t_{inst}$, $4t_{inst}$, $32t_{inst}$, external) 8-bit pulse width measurement operation (supports continuous measurement, H width, L width, rising edge to rising edge, falling edge to falling edge measurement and both edge measurement)
	PWM timer	8-bit reload timer operation (supports square wave output, operationg clock period: $1t_{inst}$, $8t_{inst}$, $16t_{inst}$, $64t_{inst}$) 8-bit resolution PWM operation
	8/16-bit timer/ counter Ch1	Can be operated either as a 2-channel 8-bit timer/counter (Timer 11 and Timer 12, each with its own independent operating clock cycle), or as one 16-bit timer/counter In Timer 11 or 16-bit timer/counter operation, event counter operation (external clock-triggered) and square wave output capable
	8/16-bit timer/ counter Ch2	Can be operated either as a 2-channel 8-bit timer/counter (Timer 21 and Timer 22, each with its own independent operating clock cycle), or as one 16-bit timer/counter In Timer 21 or 16-bit timer/counter operation, event counter operation (external clock-triggered) and square wave output capable
	External Interrupt	4 independent channels (selectable edge, interrupt vector, request flag) 5 channels (low level interrupt)
	A/D converter	10-bit resolution x 8 channels A/D conversion function (conversion time: 60 t_{inst}) Supports repeated activation by internal clock.
	UART/SIO 1, 2	Synchronous/asynchronous data transfer capable (Max. baud rate: 97.656 Kbps at 12.5 MHz) (7 and 8 bits with parity bit ; 8 and 9 bits without parity bit)
	Buzzer output	7 frequency types ($F_{CH}/2^{12}$, $F_{CH}/2^{11}$, $F_{CH}/2^{10}$, $F_{CH}/2^{9}$, $F_{CL}/2^{5}$, $F_{CL}/2^{4}$, $F_{CL}/2^{3}$,) are selectable by software.
Standby modes		Sleep mode, stop mode, subclock mode (dual clock product) and watch mode (dual clock product)
Process		CMOS

Note: t_{inst} is instruction cycle (execution time) which can be selected as 1/4, 1/8, 1/16, or 1/64 of main clock. See 3.6.2.

1.3 Differences between Products

This section describes the differences between the 3 products in the MB89470 series and lists points to note in product selection.

■ Product differences and points to note for product selection

Table 1.3-1 Package of corresponding products

Package	Model number		
	MB89475	MB89P475	MB89PV470
DIP-48P-M01	O	O	X
FPT-48P-M05	O	O	X
FPT-48P-M13	O	O	X
MQP-48C-P02	X	X	O

O: Avialable
X: Not Aviable
Note: For more information about each package, see section "1.6 Package Dimensions".

● Memory space

Before evaluating using the piggyback product, verify its differences from the product that will actually be used. Take particular care on the following points:

- The stack area, etc., is set at the upper limit of the RAM.

● Current consumption

- For the MB89PV470, add the current consumed by the EPROM mounted in the piggy-back socket.
- When operating at low speed, the current consumed by the one-time PROM products is greater than that for the mask ROM product. However, the current consumptions are roughly the same in sleep or stop mode.
- For more information, refer to "Data sheet".

● Oscillation stabilization time after power-on reset

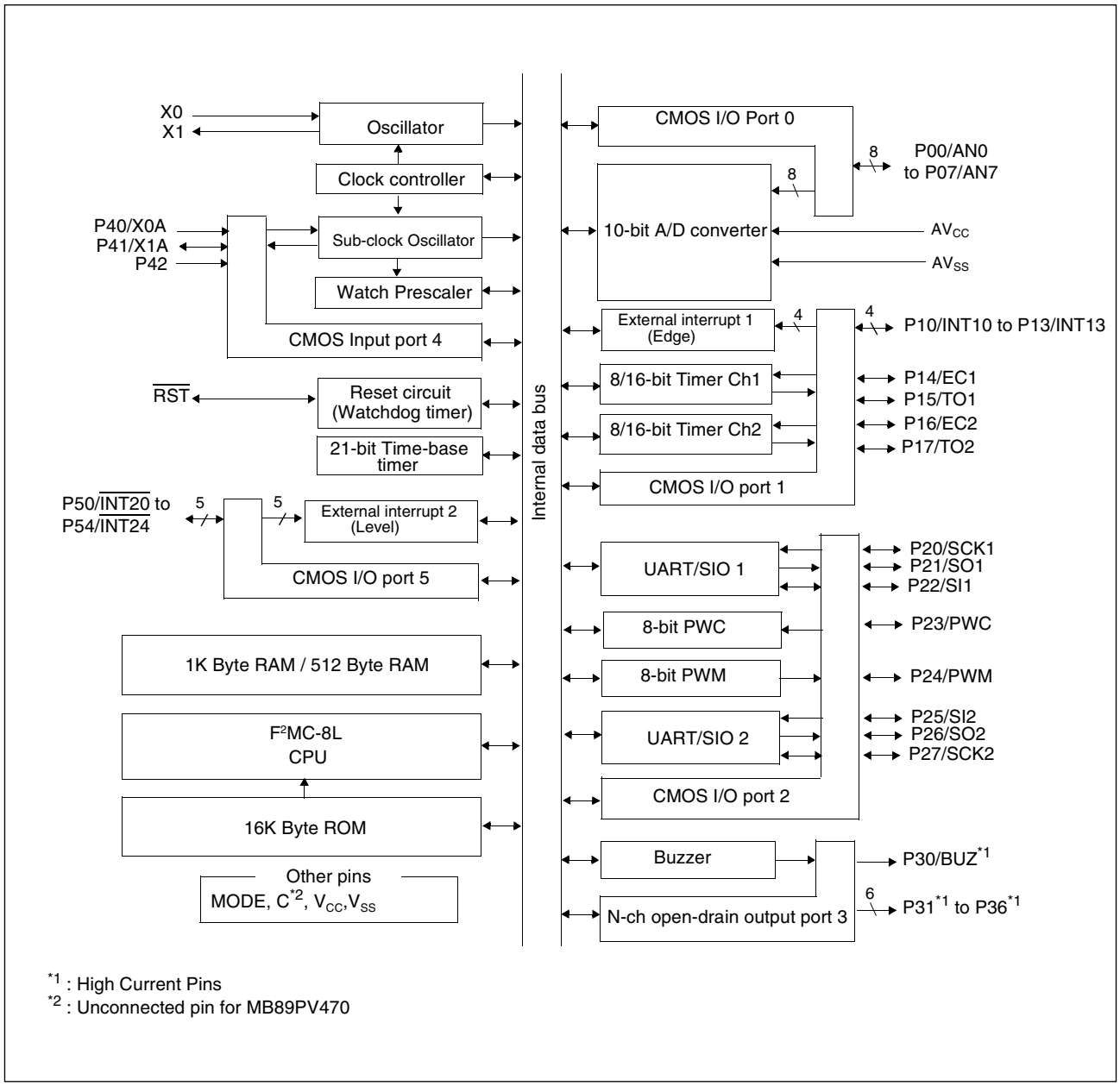
- For MB89PV470, there is no power-on stabilization time after power-on reset
- For MB89P475, there is power-on stabilization time after power-on reset
- For MB89475, the power-on stabilization time can be select.
- For more information, refer to “Appendix C “Mask Options””.

1.4 Block Diagram of MB89470 Series

Figure 1.4-1 "MB89470 series overall block diagram" shows the block diagram of the MB89470 series.

■ MB89470 series block diagram

Figure 1.4-1 MB89470 series overall block diagram



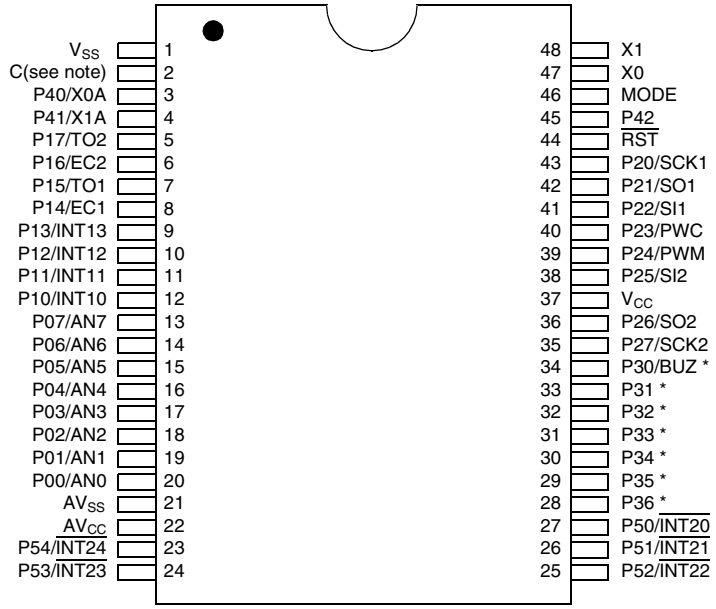
*1 : High Current Pins
*2 : Unconnected pin for MB89PV470

1.5 Pin Assignment

Figure 1.5-1 "DIP-48P-M01 pin assignment to Figure 1.5-3 "MQP-48C-P02 pin assignment" show the pin assignment diagrams for the MB89470 series.

■ DIP-48P-M01 pin assignment

Figure 1.5-1 DIP-48P-M01 pin assignment



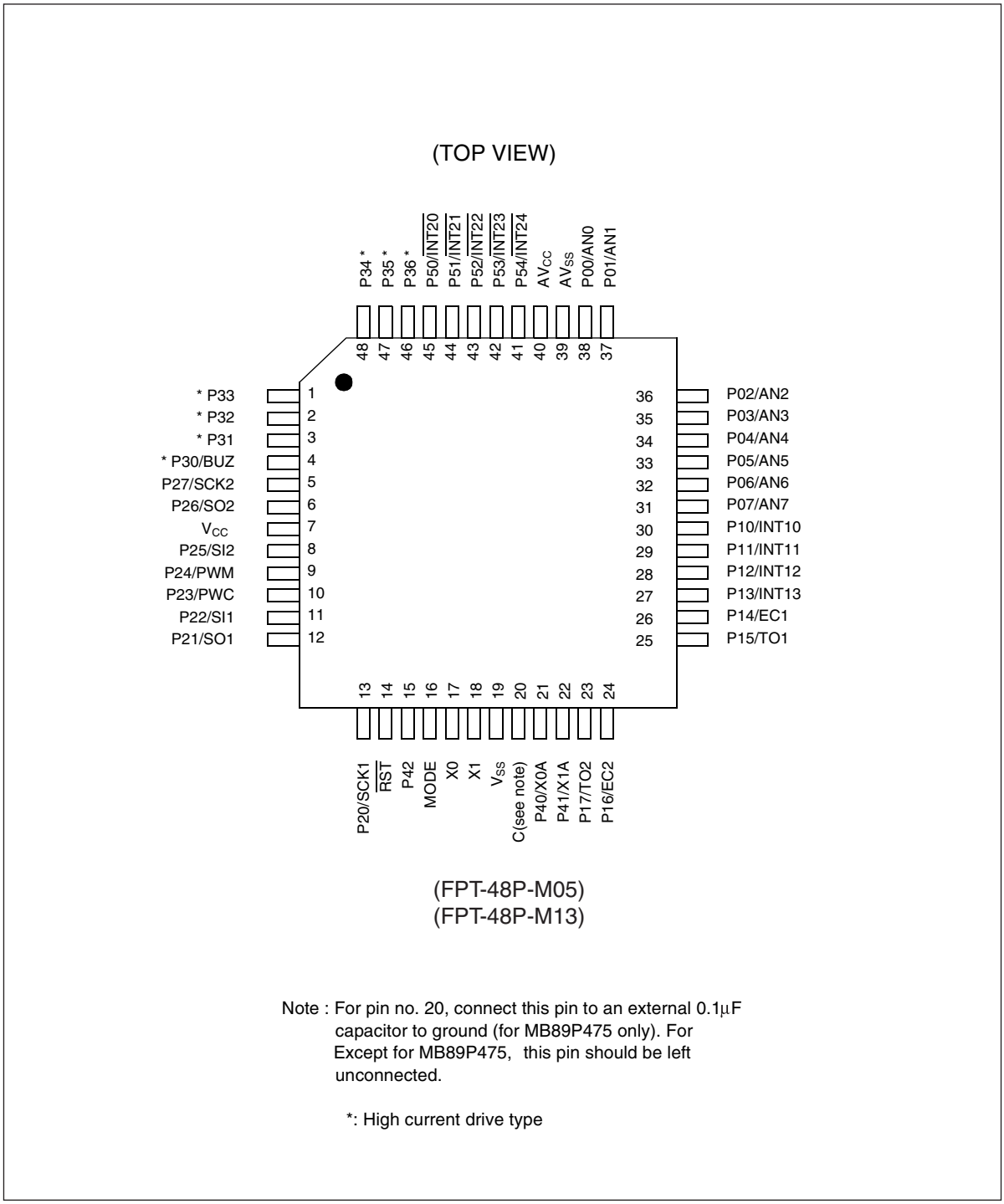
(DIP-48P-M01)

Note : For pin no. 2, connect this pin to an external 0.1μF capacitor to ground (for MB89P475 only). Except for MB89P475, this pin should be left unconnected.

* High current drive type

■ FPT-48P-M05 Pin Assignment

Figure 1.5-2 FPT-48P-M05 pin assignment

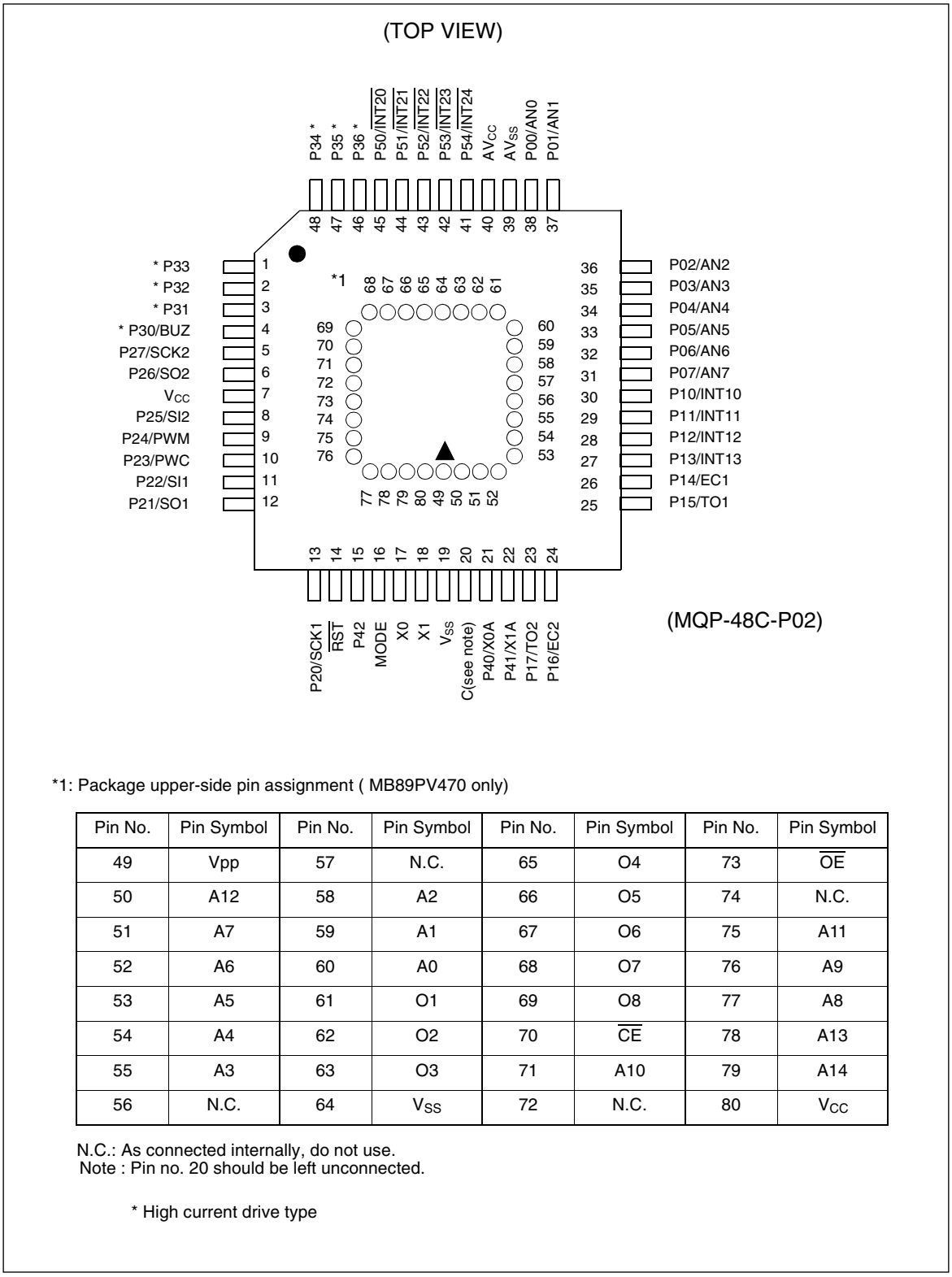


Note : For pin no. 20, connect this pin to an external 0.1μF capacitor to ground (for MB89P475 only). For Except for MB89P475, this pin should be left unconnected.

*: High current drive type

■ MQP-48C-P02 pin assignment

Figure 1.5-3 MQP-48C-P02 pin assignment



*1: Package upper-side pin assignment (MB89PV470 only)

Pin No.	Pin Symbol	Pin No.	Pin Symbol	Pin No.	Pin Symbol	Pin No.	Pin Symbol
49	Vpp	57	N.C.	65	O4	73	$\overline{OE}$
50	A12	58	A2	66	O5	74	N.C.
51	A7	59	A1	67	O6	75	A11
52	A6	60	A0	68	O7	76	A9
53	A5	61	O1	69	O8	77	A8
54	A4	62	O2	70	$\overline{CE}$	78	A13
55	A3	63	O3	71	A10	79	A14
56	N.C.	64	V_{SS}	72	N.C.	80	V_{CC}

N.C.: As connected internally, do not use.
Note : Pin no. 20 should be left unconnected.

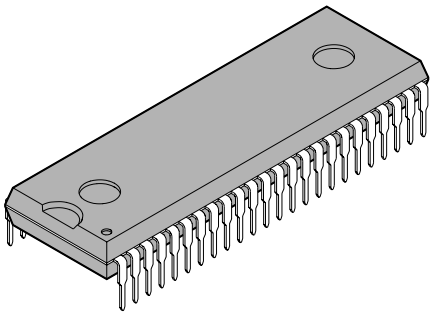
* High current drive type

1.6 Package Dimensions

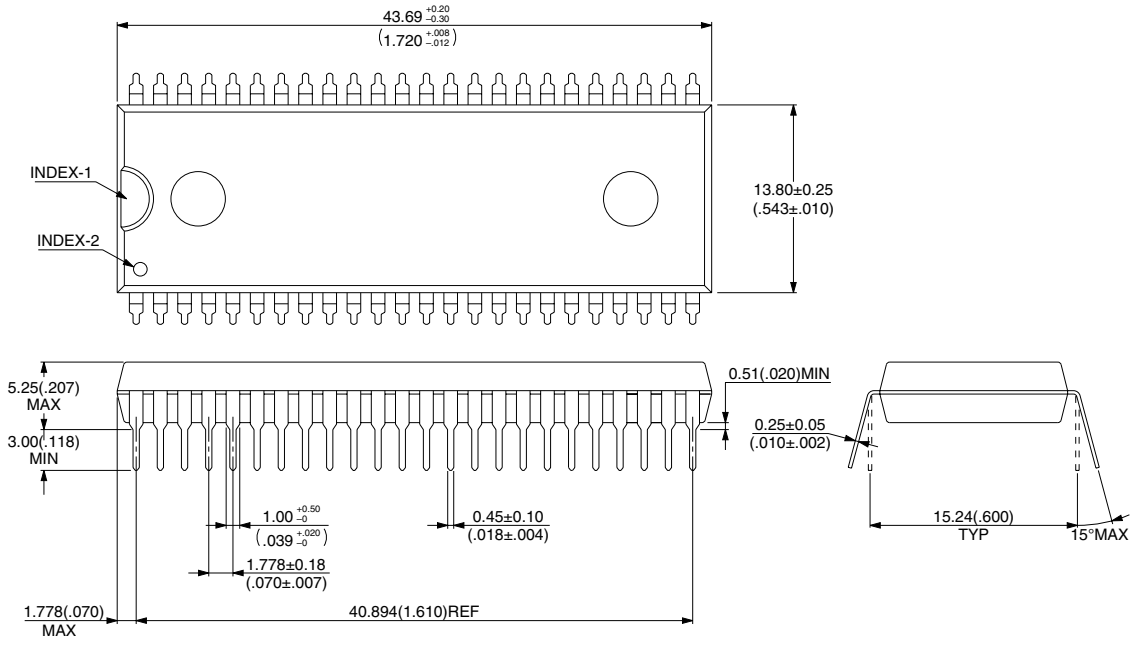
Three different packages are available for the MB89470 series.
Figure 1.6-1 "DIP-48P-M01 package dimensions" to Figure 1.6-4 "MQP-48C-P02 package dimensions" show the package dimensions.

■ DIP-48P-M01 package dimensions

Figure 1.6-1 DIP-48P-M01 package dimensions

48-pin plastic SH-DIP  (DIP-48P-M01)	Lead pitch	70 mil
	Row spacing	600 mil
	Sealing method	Plastic mold

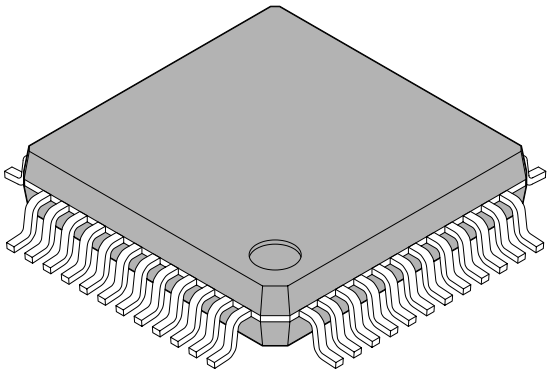
48-pin plastic SH-DIP
(DIP-48P-M01)



Dimensions in mm (inches).

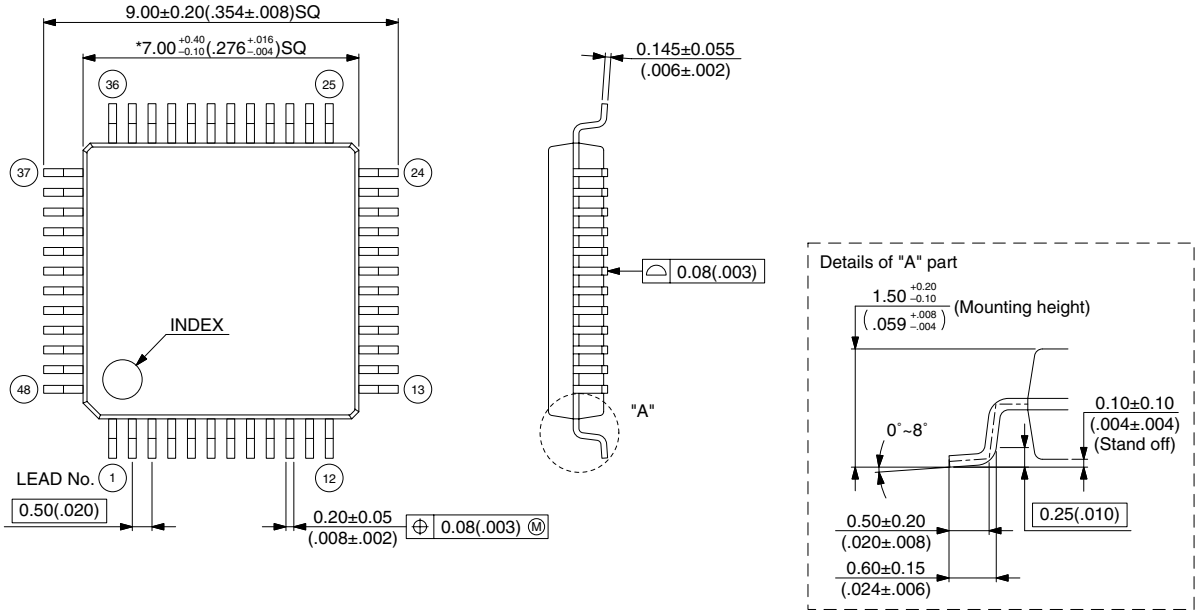
■ FPT-48P-M05 package dimensions

Figure 1.6-2 FPT-48P-M05 package dimensions

48-pin plastic LQFP		
 (FPT-48P-M05)	Lead pitch	0.50 mm
	Package width × package length	7.0 × 7.0 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm MAX
	Weight	0.17g
	Code (Reference)	P-LFQFP48-7 × 7-0.50

48-pin plastic LQFP
(FPT-48P-M05)

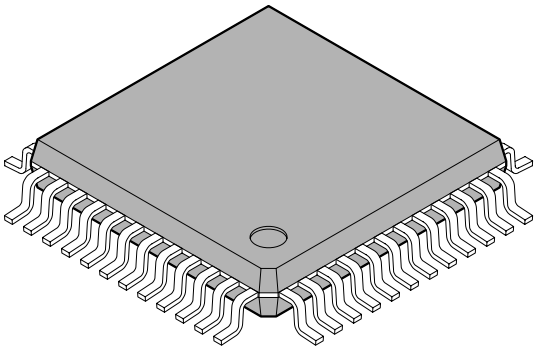
Note 1) * : These dimensions include resin protrusion.
Note 2) Pins width and pins thickness include plating thickness.
Note 3) Pins width do not include tie bar cutting remainder.



Dimensions in mm (inches).
Note: The values in parentheses are reference values.

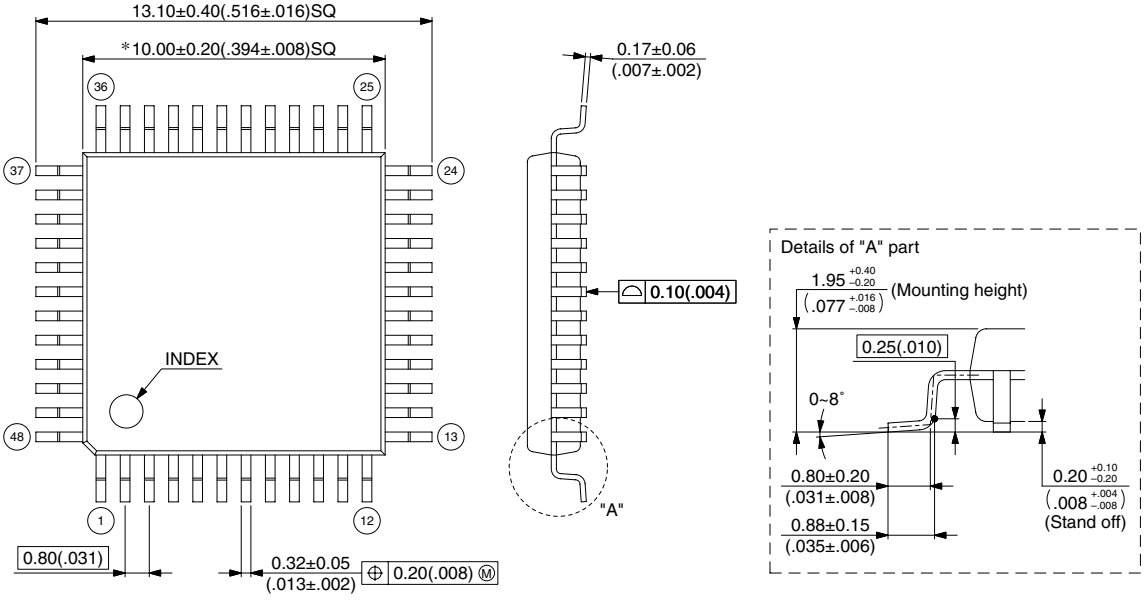
■ FPT-48P-M13 package dimensions

Figure 1.6-3 FPT-48P-M13 package dimensions

48-pin plastic QFP  (FPT-48P-M13)	Lead pitch	0.80 mm
	Package width × package length	10 × 10 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	2.35 mm MAX
	Code (Reference)	P-QFP44-10×10-0.80

48-pin plastic QFP
(FPT-48P-M13)

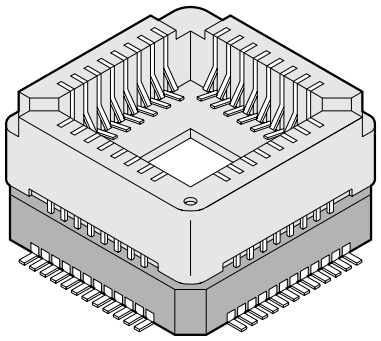
Note 1) * : These dimensions do not include resin protrusion.
Note 2) Pins width and pins thickness include plating thickness.
Note 3) Pins width do not include tie bar cutting remainder.



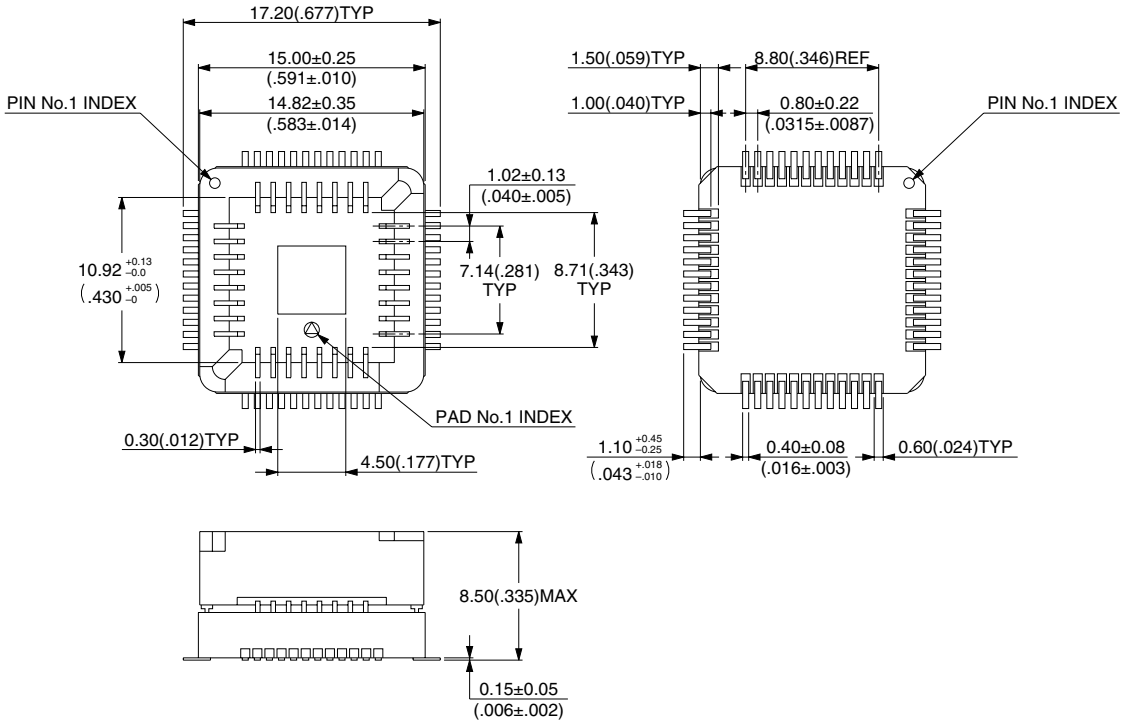
Dimensions in mm (inches).
Note: The values in parentheses are reference values.

■ MQP-48C-P02 package dimensions

Figure 1.6-4 MQP-48C-P02 package dimensions

48-pin ceramic MQFP  (MQP-48C-P02)	Lead pitch	0.8 mm
	Lead shape	Straight
	Motherboard material	Ceramic
	Mounted package material	Plastic

48-pin ceramic MQFP
(MQP-48C-P02)



Dimensions in mm (inches).

1.7 I/O Pins and Pin Functions

Table 1.7-1 "Pin descriptions" and Table 1.7-2 "Pin descriptions for the external EPROM socket (MB89PV470 only)" list the MB89470 series I/O pins and their functions. Table 1.7-3 "I/O circuit types" lists the I/O circuit types.
The letter in the I/O Circuit Type column of Table 1.7-1 "Pin descriptions" refers to the letter in the Circuit Class column of Table 1.7-3 "I/O circuit types".

■ I/O Pins and Pin Functions

Table 1.7-1 Pin descriptions (1/3)

Pin Number		Pin Name	I/O Circuit Type	Function
QFP/ MQFP *2	SDIP *1			
17	47	X0	A	Connection pins for a crystal or other oscillator. An external clock can be connected to X0. In this case, leave X1 open.
18	48	X1		
16	46	MODE	B	Input pins for setting the memory access mode. Connect directly to V_{SS}.
14	44	$\overline{RST}$	C	Reset I/O pin. The pin is a N-ch open-drain type with pull-up resistor and a hysteresis input. The pin outputs a "L" level when an internal reset request is present. Inputting an "L" level initializes internal circuits.
38 to 31	20 to 13	P00/AN0 to P07/ AN7	D	General-purpose I/O port. The pins are shared with the analog inputs for the A/D converter.
30 to 27	12 to 9	P10/INT10 to P13/INT13	E	General-purpose I/O port. A hysteresis input for INT10 to INT13. The pin is shared with an external interrupt 1 input.
26	8	P14/EC1	E	General-purpose I/O port. A hysteresis input for EC1. The pin is shared with the 8/16 bit timer 11 input.
25	7	P15/TO1	F	General-purpose I/O port. The pin is shared with the output of 8/16-bit timer 11.
24	6	P16/EC2	E	General-purpose I/O port. A hysteresis input for EC2. The pin is shared with the 8/16 bit timer 21 input.
23	5	P17/TO2	F	General-purpose I/O port. The pin is shared with the output of 8/16-bit timer 21.

Table 1.7-1 Pin descriptions (2/3)

Pin Number		Pin Name	I/O Circuit Type	Function
QFP/ MQFP *2	SDIP *1			
13	43	P20/SCK1	E	General-purpose I/O port. A hysteresis input for SCK1. The pin is shared with the clock I/O of UART/SIO 1.
12	42	P21/SO1	F	General-purpose I/O port. The pin is shared with the serial data output of UART/SIO 1.
11	41	P22/SI1	E	General-purpose I/O port. A hysteresis input for SI1. The pin is shared with the serial data input of UART/SIO 1.
10	40	P23/PWC	E	General-purpose I/O port. A hysteresis input for PWC. This pin is shared with PWC input.
9	39	P24/PWM	F	General-purpose input port. This pin is shared with PWM output.
8	38	P25/SI2	E	General-purpose I/O port. A hysteresis input for SI2. The pin is shared with the serial data input of UART/SIO 2.
6	36	P26/SO2	F	General-purpose I/O port. The pin is shared with the serial data output of UART/SIO 2.
5	35	P27/SCK2	E	General-purpose I/O port. A hysteresis input for SCK2. The pin is shared with the clock I/O of UART/SIO 2.
4	34	P30/BUZ	G	N-channel open-drain output. The pin is shared with buzzer output.
3 to 1, 48 to 46	33 to 28	P31 to P36	G	N-channel open-drain output.
21	3	P40/X0A	H	General-purpose input port. (single clock system)
			A	Connection pins for a crystal or other oscillator. (dual clock system) An external clock can be connected to X0A. In this case, leave X1A open.
22	4	P41/X1A	H	General-purpose input port. (single clock system)
			A	Connection pins for a crystal or other oscillator. (dual clock system) An external clock can be connected to X0A. In this case, leave X1A open.
15	45	P42	H	General-purpose input port.

Table 1.7-1 Pin descriptions (3/3)

Pin Number		Pin Name	I/O Circuit Type	Function
QFP/ MQFP *2	SDIP *1			
45 to 41	27 to 23	P50/$\overline{\text{INT20}}$ to P54/$\overline{\text{INT24}}$	E	General-purpose I/O port. A hysteresis input for $\overline{\text{INT20}}$ to $\overline{\text{INT24}}$. The pin is shared with an external interrupt 2 input.
20	2	C	–	Capacitor connection pin *3
7	37	V_{CC}	–	Power supply pin (+5V).
19	1	V_{SS}	–	Power supply pin (GND).
40	22	AV_{CC}	–	A/D converter power supply pin.
39	21	AV_{SS}	–	A/D converter power supply pin. Use at the same voltage level as V_{SS}.

*1: DIP-48P-M01
*2: FPT-48P-M05 / FPT-48P-M13 / MQP-48C-P02
*3: When MB89475 or MB89PV470 is used, this pin will become a N.C. pin without internal connection.
When MB89P475 is used, connect this pin to an external 0.1µF capacitor to ground.

Table 1.7-2 Pin descriptions for the external EPROM socket (MB89PV470 only)

Pin Numbe MQFP*	Pin Name	I/O	Function
49	V_{PP}	O	"H" level output pin
50 51 52 53 54 55 58 59 60	A12 A7 A6 A5 A4 A3 A2 A1 A0	O	Address output pins.
61 62 63	O1 O2 O3	I	Data input pins.
64	V_{SS}	O	Power supply pin (GND).
65 66 67 68 69	O4 O5 O6 O7 O8	I	Data input pins.
70	$\overline{CE}$	O	Chip enable pin for the ROM. Outputs "H" in standby mode.
71	A10	O	Address output pin.
73	$\overline{OE}$	O	Output enable pin for the ROM. Always outputs "L".
75 76 77 78 79	A11 A9 A8 A13 A14	O	Address output pins.
80	V_{CC}	O	Power supply pin for the EPROM.
56 57 72 74	N.C.	–	Internally connected pins. Always leave open.

*: MQP-48C-P02

Table 1.7-3 I/O Circuit Types (1/2)

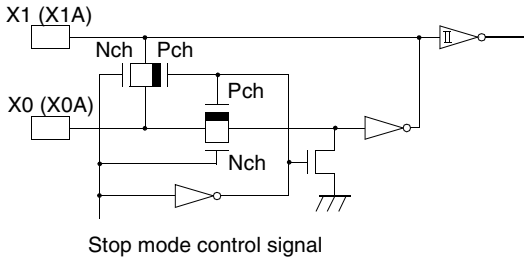
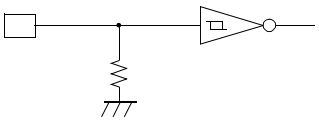
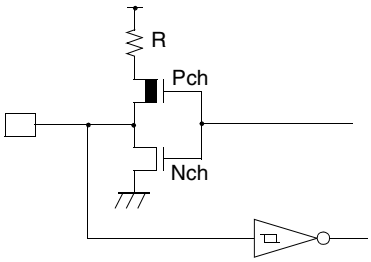
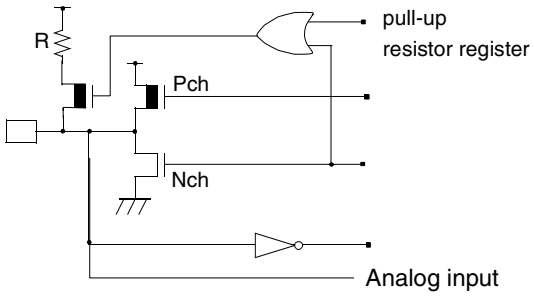
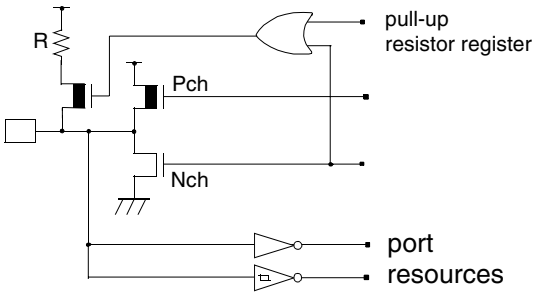
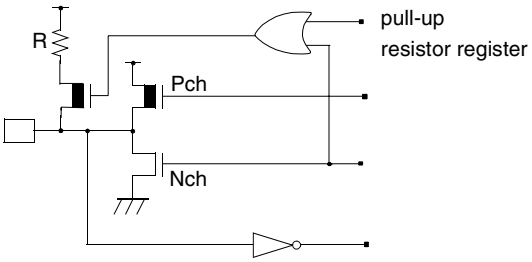
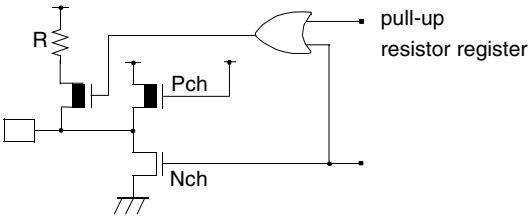
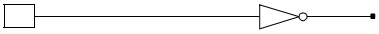
Circuit Class	Circuit	Remarks
A		• Oscillation feedback resistance is approx. 500 kΩ for main clock circuit and approx.5 MΩ for sub-clock circuit.
B		• Hysteresis input • The pull-down resistor is Approx. 50kΩ (no pull-down resistor in MB89P475)
C		• The pull-up resistance (P-channel) is Approx. 50kΩ • Hysteresis input
D		• CMOS output • CMOS input • Analog input • Selectable pull-up resistor Approx. 50 kΩ

Table 1.7-3 I/O Circuit Types (2/2)

Circuit Class	Circuit	Remarks
E		• CMOS output • CMOS input • Selectable pull-up resistor Approx. 50 kΩ
F		• CMOS output • CMOS input • Selectable pull-up resistor Approx. 50 kΩ
G		• N-channel open-drain output • Selectable pull-up resistor Approx. 50 kΩ
H		• CMOS input

CHAPTER 2
HANDLING DEVICES

This chapter describes points to note when using the general-purpose single-chip microcontroller.

2.1 Notes on Handling Devices

This section lists points to note regarding the power supply voltage, pins, and other device handling aspects.

■ Notes on handling devices

- Take great care not to exceed the maximum rated voltage (prevent latchup).

  Latchup may occur on CMOS ICs if voltage higher than V_{CC} or lower than V_{SS} is applied to input and output pins other than medium- to high-voltage pins, or if voltage higher than the ratings is applied between V_{CC} and V_{SS}.

  When latchup occurs, power supply current increases rapidly and might thermally damage elements. When using, take great care not to exceed the absolute maximum ratings.

  Also, take care to prevent the analog power supply (AV_{CC}) and analog input from exceeding the digital power supply (V_{CC}) when the analog system power supply is turned on and off.

- Stabilizing supply voltage is important.

  A rapid fluctuation of V_{CC} power supply voltage could cause malfunctions, even if it occurs within the operation assurance range of the voltage. As stabilization guidelines, it is recommended to control power so that V_{CC} ripple fluctuations (P-P value) will be less than 10% of the standard V_{CC} value at the commercial frequency (50 to 60 Hz) and the transient fluctuation rate will be less than 0.1 V/ms at the time of a momentary fluctuation such as when power is switched.

- Treatment of unused input pins

  Leaving unused input pins open could cause malfunctions. They should be connected to a pull-up or pull-down resistor.

- Treatment of N.C. pins

  Be sure to leave (internally connected) N.C. pins open.

- Note to Noise in the External Reset Pin ($\overline{RST}$)

  If the reset pulse applied to the external reset pin ($\overline{RST}$) does not meet the specifications, it may cause malfunctions. Use caution so that the reset pulse less than the specifications will not be fed to the external reset pin ($\overline{RST}$).

- Treatment of power supply pins on microcontroller with A/D converter

  Connect to be $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$ even if the A/D is not in use.

- Precautions when using an xternal clock

  Even when an external clock is used, oscillation stabilization delay time is required for power-on reset and wake-up from stop mode.

● MB89470 series C pin handling

The connection of C pin is shown below:

Model number	C pin connection
MB89475	N.C.
MB89P475	This pin must be connected to an external 0.1 μF capacitor to ground.
MB89PV470	N.C.

CHAPTER 3

CPU

This chapter describes the functions and operation of the CPU.

3.1 Memory Space

The microcontrollers of the MB89470 series offer a memory space of 64 Kbytes. The memory space contains the I/O area, RAM area, ROM area, and external area. The memory space contains areas used for special purposes such as the general-purpose registers and vector table.

■ Memory space structure

● I/O area (addresses: 0000_H to $007F_H$)

- Control registers and data registers for the internal peripheral functions are located in this area.
- As the I/O area is allocated within the memory space, I/O can be accessed in the same way as memory. High-speed access using direct addressing is available.

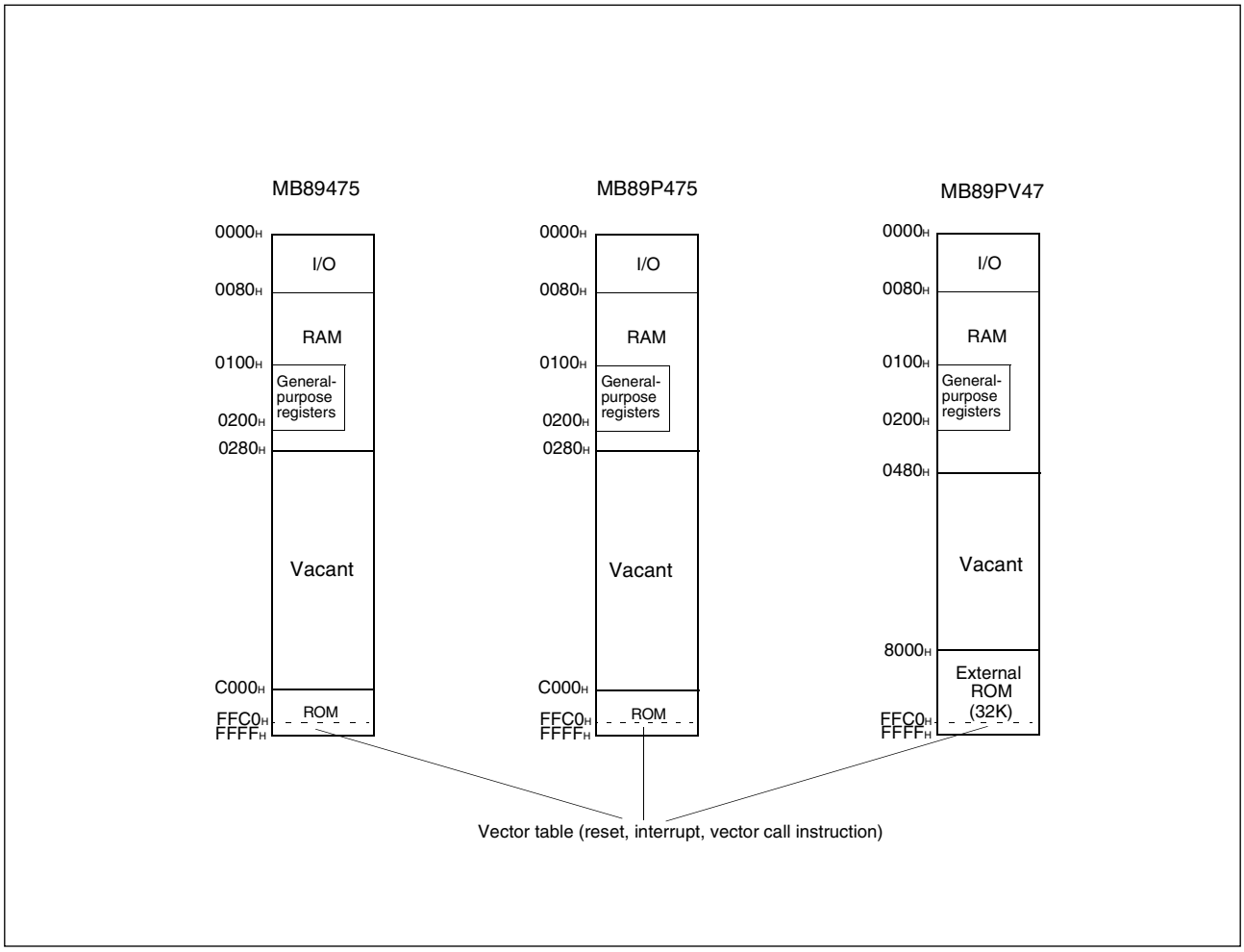
● RAM area

- Internal static RAM is provided as an internal data area.
- The internal RAM size differs between products.
- Addresses between 80_H and FF_H support high-speed access using direct addressing.
- Addresses between 100_H and $1FF_H$ can be used as the general-purpose register area
- The contents of RAM is indeterminate after a reset.

● ROM area

- Internal ROM is provided as an internal program area.
- The internal ROM size differs between products. Setting the memory access mode to internal ROM mode.
- Addresses between $FFC0_H$ and $FFFF_H$ are used for the vector table, etc.

Figure 3.1-1 Memory Map



3.1.1 Special Areas

In addition to the I/O area, the special purpose areas in the memory space include the general-purpose register area and the vector table area.

■ General-purpose register areas (addresses: 0100_H to $01FF_H$)

- Provides auxiliary registers for 8-bit arithmetic operation and transfer instructions.
- Allocated to a region of the RAM area. Can also be used as normal RAM.
- Using the area as general-purpose registers enables high-speed access by general-purpose register addressing using short instructions.

Reference:

See section 3.2.2, "Register Bank Pointer (RP)" and section 3.3, "General-purpose Registers" for details.

■ Vector table area (addresses: $FFC0_H$ to $FFFF_H$)

- Used as the vector table for the vector call instruction, interrupts, and resets.
- The vector table is allocated at the top of the ROM area. The start address of the corresponding processing routine is set as data at each vector table address.

Table 3.1-1 "Vector table" lists the vector table addresses referenced by the vector call instruction, interrupts, and resets.

Reference:

See Section 3.4, "Interrupts", Section 3.5, "Resets", and "(6) CALLV #vct" in Appendix B.2, "Special Instructions" for details.

Table 3.1-1 Vector table

Vector call instruction	Vector table address	
	Upper	Lower
CALLV #0	$FFC0_H$	$FFC1_H$
CALLV #1	$FFC2_H$	$FFC3_H$
CALLV #2	$FFC4_H$	$FFC5_H$
CALLV #3	$FFC6_H$	$FFC7_H$
CALLV #4	$FFC8_H$	$FFC9_H$
CALLV #5	$FFCA_H$	$FFCB_H$
CALLV #6	$FFCC_H$	$FFCD_H$
CALLV #7	$FFCE_H$	$FFCF_H$

Interrupts	Vector table address	
	Upper	Lower
IRQF	$FFDC_H$	$FFDD_H$
IRQE	$FFDE_H$	$FFDF_H$
IRQD	$FFE0_H$	$FFE1_H$
IRQC	$FFE2_H$	$FFE3_H$
IRQB	$FFE4_H$	$FFE5_H$
IRQA	$FFE6_H$	$FFE7_H$
IRQ9	$FFE8_H$	$FFE9_H$
IRQ8	$FFEA_H$	$FFEB_H$
IRQ7	$FFEC_H$	$FFED_H$
IRQ6	$FFEE_H$	$FFEF_H$
IRQ5	$FFF0_H$	$FFF1_H$
IRQ4	$FFF2_H$	FFF3H
IRQ3	$FFF4_H$	$FFF5_H$
IRQ2	$FFF6_H$	$FFF7_H$
IRQ1	$FFF8_H$	$FFF9_H$
IRQ0	$FFFA_H$	$FFFB_H$
Mode data	–	$FFFD_H$
Reset vector	$FFFE_H$	$FFFF_H$

3.1.2 Storing 16-bit Data in Memory

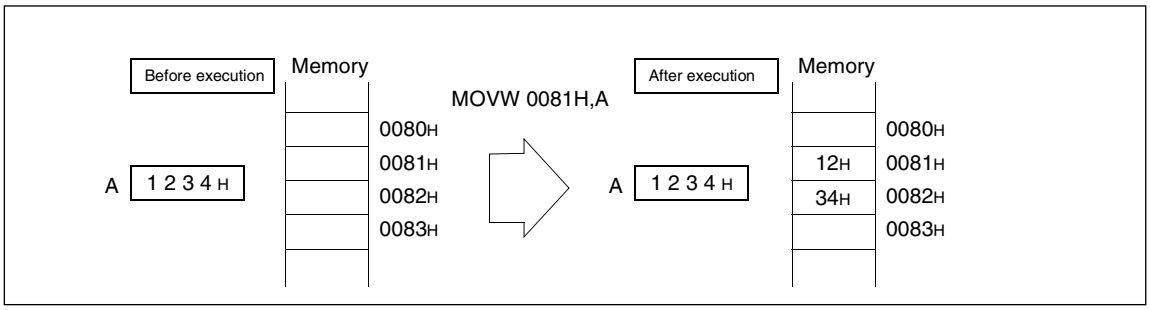
For 16-bit data and the stack, store the upper data in the lower memory address value.

■ Storing 16-bit data in RAM

When writing 16-bit data to memory, store the upper byte at the lower address and the lower byte at the next address. Handle reading of 16-bit data in the same way.

Figure 3.1-2 "Storing 16-bit data in memory" shows how 16-bit data is stored in memory.

Figure 3.1-2 Storing 16-bit data in memory



■ Storing 16-bit operands

The same byte order applies when specifying a 16-bit operand in an instruction. Store the upper byte at the address following the operation code (instruction) and the lower byte at the next address.

The byte ordering applies to both 16-bit immediate data and operands that specify a memory address.

Figure 3.1-3 "Byte order of 16-bit data in an instruction" shows how 16-bit data is stored in an instruction.

Figure 3.1-3 Byte order of 16-bit data in an instruction

```
[Example]   MOV   A,5678H          ; Extended address
            MOVW  A,#1234H         ; 16-bit immediate data
                  ⇩ After assembly
                  .
                  .
                  .
            X X X 0 H   XX XX
            X X X 2 H   60 56 78   ; Extended address
            X X X 5 H   E4 12 34   ; 16-bit immediate data
            X X X 8 H   XX
                  .
                  .
                  .
```

■ Storing 16-bit data on stack

The same byte order applies when saving 16-bit register data on the stack during an interrupt or similar. The upper byte is stored in the lower address.

3.2 Dedicated Registers

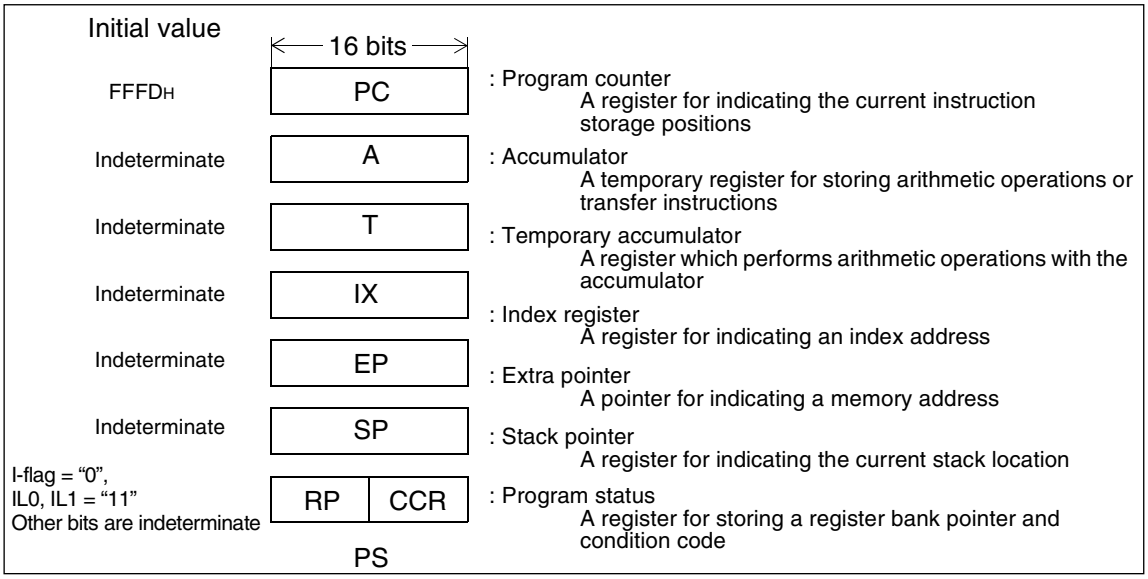
The dedicated registers in the CPU consist of the program counter (PC), two arithmetic operation registers (A and T), three address pointers (IX, EP, and SP), and the program status (PS). All registers are 16 bits.

■ Dedicated register configuration

The dedicated registers in the CPU consist of seven 16-bit registers. Some of these registers are also able to be used as 8-bit registers, using the lower 8 bits only.

Figure 3.2-1 "Dedicated register configuration" shows the structure of the dedicated registers.

Figure 3.2-1 Dedicated register configuration



■ Dedicated register functions

● Program counter (PC)

The program counter is a 16-bit counter that indicates the memory address of the instruction currently being executed by the CPU. Instruction execution, interrupts, resets, and similar update the contents of the program counter. The initial value during a reset is the read address of the mode data ($FFFD_H$).

● Accumulator (A)

The accumulator is a 16-bit arithmetic operation register. The accumulator is used to perform arithmetic operations and data transfers with data in memory or in other registers such as the temporary accumulator (T). The content of the accumulator can be treated as either word (16-bit) or byte (8-bit) data. Only the lower 8 bits (AL) of the accumulator are used for byte arithmetic operations or transfers. In this case, the upper 8 bits (AH) remain unchanged. The content of the accumulator after a reset is indeterminate.

● Temporary accumulator (T)

The temporary accumulator is an auxiliary 16-bit arithmetic operation register used to perform arithmetic operations with the data in the accumulator (A). The content of the temporary accumulator is treated as word data (16-bit) for word-length arithmetic operations with the accumulator and as byte data (8-bit) for byte-length arithmetic operations. For byte-length arithmetic operations, only the lower 8 bits of the temporary accumulator (TL) are used and the upper 8 bits (TH) are not used.

Executing a transfer instruction to transfer data to the accumulator (A) automatically transfer the previous content of the accumulator to the temporary accumulator. In this case also, a byte transfer leaves the upper 8 bits of the temporary accumulator (TH) unchanged. The content of the temporary accumulator after a reset is indeterminate.

● Index register (IX)

The index register is a 16-bit register used to hold the index address. The index register is used in conjunction with a single byte offset value (–128 to +127). Adding the sign-extended offset value to the index address generates the memory address for data access. The content of the index register after a reset is indeterminate.

● Extra pointer (EP)

The extra pointer is a 16-bit register used to hold a memory address for data access. The content of the extra pointer after a reset is indeterminate.

● Stack pointer (SP)

The stack pointer is a 16-bit register used to hold the address referenced during operations such as interrupts, subroutine calls, and the stack save and restore instructions. The value of the stack pointer during program execution is the address of the most recently saved data on the stack. The content of the stack pointer after a reset is indeterminate.

● Program status (PS)

The program status is a 16-bit control register. The upper 8 bits contain the register bank pointer (RP) which points to the address of the current general-purpose register bank.

The lower 8 bits contain the condition code register (CCR) which contains flags indicating the current CPU status. The two 8-bit registers which form the program status cannot be accessed independently (the program status can only be accessed by the MOVW A,PS and MOVW PS,A instructions).

Reference:

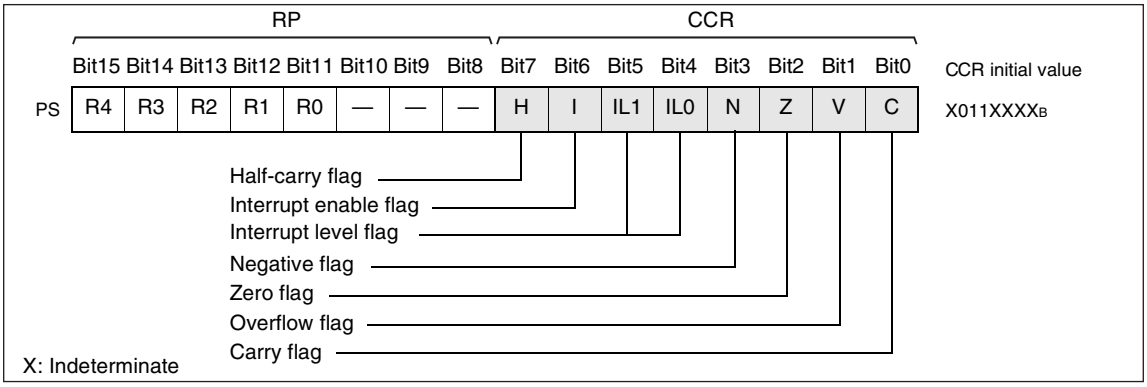
Refer to the F^2MC-8L MB89600 series Programming Manual for details on using the dedicated registers.

3.2.1 Condition Code Register (CCR)

The condition code register (CCR) located in the lower 8 bits of the program status (PS) consists of the C, V, Z, N, and H bits indicating the results of arithmetic operations and the contents of transfer data, and the I, IL1, and IL0 bits for control whether or not the CPU accepts interrupt requests.

■ Structure of condition code register (CCR)

Figure 3.2-2 Structure of condition code register



■ Arithmetic operation result bits

● Half-carry flag (H)

Set when a carry from bit 3 to bit 4 or a borrow from bit 4 to bit 3 occurs as a result of an arithmetic operation. Cleared otherwise. As this flag is for the decimal adjustment instructions, do not use this flag in cases other than addition or subtraction.

● Negative flag (N)

Set if the most significant bit (MSB) is set to 1 as a result of an arithmetic operation. Cleared otherwise.

● Zero flag (Z)

Set when an arithmetic operation results in 0. Cleared otherwise.

● Overflow flag (V)

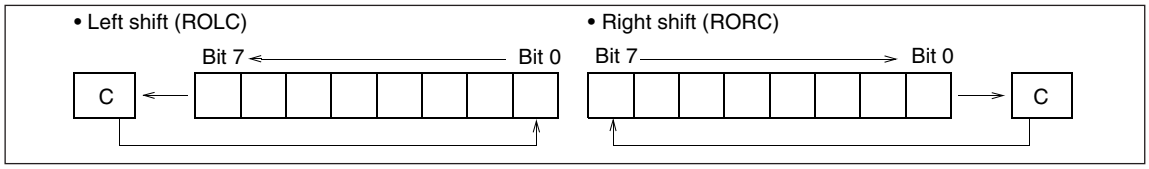
Set if the complement on 2 overflows as a result of an arithmetic operation. Cleared otherwise.

● Carry flag (C)

Set when a carry from bit 7 or borrow to bit 7 occurs as a result of an arithmetic operation. Cleared otherwise. Set to the shift-out value in case of a shift instruction.

Figure 3.2-3 "Change of carry flag by shift instruction" shows the change of the carry flag by a shift instruction.

Figure 3.2-3 Change of carry flag by shift instruction



Precaution:

The condition code register is part of the program status (PS) and cannot be accessed independently.

Note:

In practice, the flag bits are rarely fetched and used directly. Instead, the bits are used indirectly by instructions such as branch instructions (such as BNZ) or the decimal adjustment instructions (DAA, DAS). The content of the flags after a reset is indeterminate.

■ Interrupt acceptance control bit

● Interrupt enable flag (I)

Interrupt is enabled when this flag is set to "1" and the CPU accepts interrupt. Interrupt is prohibited when this flag is set to "0" and the CPU does not accept interrupt.

The initial value after a reset is "0".

Normal practice is to set the flag to "1" by the SETI instruction and clear to "0" by the CLRI instruction.

● Interrupt level bits (IL1, IL0)

These bits indicate the level of the interrupt currently being accepted by the CPU. The value is compared with the interrupt level setting registers (ILR1 to ILR4) which have a setting for each peripheral function interrupt request (IRQ0 to IRQF).

Given that the interrupt enable flag is enabled (I = "1"), the CPU only performs interrupt processing for interrupt requests with an interrupt level value that is less than the value of these bits.

Table 3.2-1 "Interrupt level" lists the interrupt level priorities. The initial value after a reset is "11_B".

Table 3.2-1 Interrupt level

IL1	IL0	Interrupt level	Priority
0	0	1	High ↕ Low (no interrupt)
0	1		
1	0	2	
1	1	3	

Note:

The interrupt level bits (IL1, IL0) are normally "11" when the CPU is not processing an interrupt (during main program execution).

Reference:

See Section 3.4, "Interrupts" for details on interrupts.

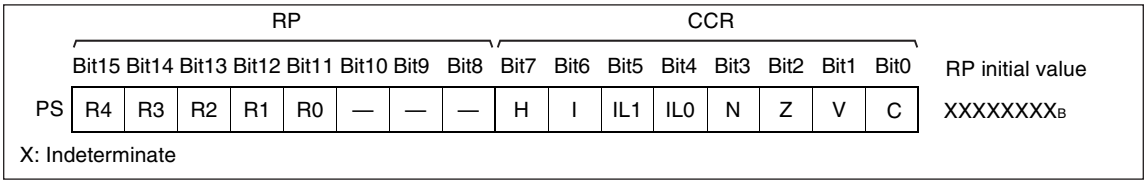
3.2.2 Register Bank Pointer (RP)

The register bank pointer (RP) located in the upper 8 bits of the program status (PS) indicates the address of the general-purpose register bank currently in use. The RP is converted to form the actual address in general-purpose register addressing.

■ Structure of register bank pointer (RP)

Figure 3.2-4 "Structure of register bank pointer" shows the structure of the register bank pointer.

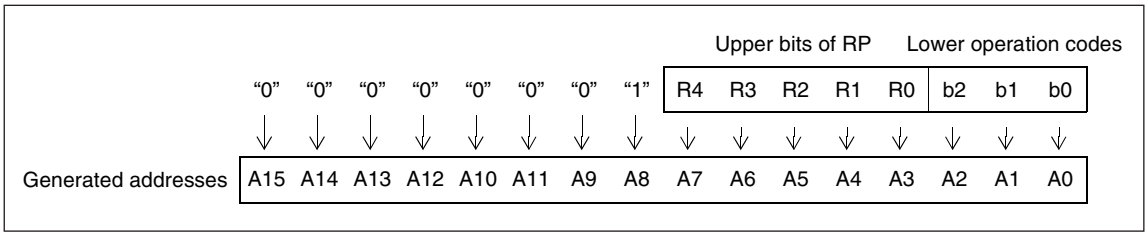
Figure 3.2-4 Structure of register bank pointer



The register bank pointer indicates the address of the register bank currently in use.

Figure 3.2-5 "Rule for conversion of actual addresses of general-purpose register area" shows the relationship between the pointer contents and the actual address is based on the conversion rule.

Figure 3.2-5 Rule for conversion of actual addresses of general-purpose register area



The register bank pointer points to the memory block (register bank) in the RAM area that is used for general-purpose registers. A total of 32 register banks are available. A register bank is specified by setting a value between 0 and 31 in the upper 5 bits of the register bank pointer. Each register bank contains 8-bit general-purpose registers. Registers are specified by the lower 3 bits of the operation codes.

Using the register bank pointer, the addresses 0100_H to $01FF_H$ can be used as the general-purpose register area. However, the available area is limited on some products if internal RAM only is used. The initial value after a reset is indeterminate.

Precautions:

- The register bank pointer is part of the program status (PS) and cannot be accessed independently.
- The register bank pointer(RP) must be set before using the general prupose register.

3.3 General-purpose Registers

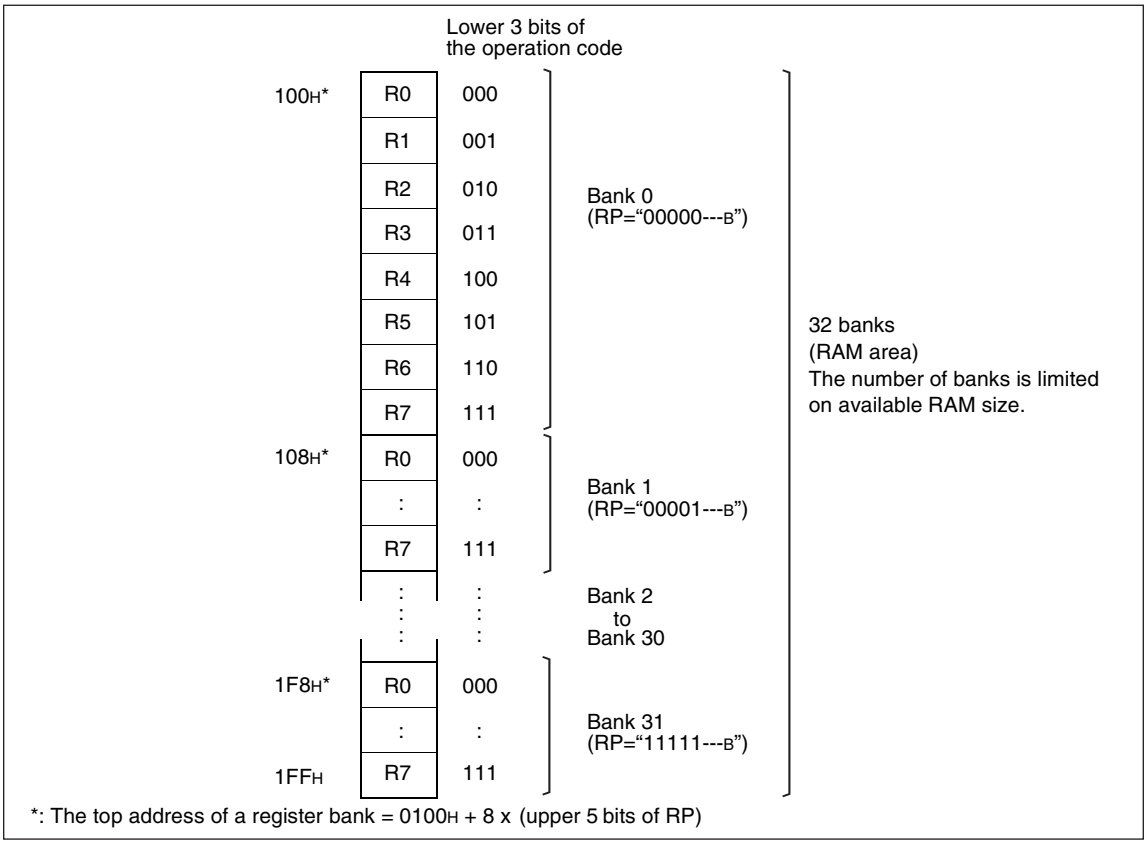
The general-purpose registers are a memory block made up of banks, with 8 x 8-bit registers per bank.
The register bank pointer (RP) is used to specify the register bank.
The function permits the use of up to 32 banks, but the number of banks that can actually be used depends on how much RAM the device has.
Register banks are valid for interrupt processing, vector call processing, and subroutine calls.

■ Structure of general-purpose registers

- The general-purpose registers are 8 bits and located in the register banks of the general-purpose register area (in RAM).
- One bank contains eight registers (R0 to R7) and up to a total of 32 banks.
- The register bank currently in use is specified by the register bank pointer (RP). The lower three bits of the operation code specify general-purpose register 0 (R0) to general-purpose register 7 (R7).

Figure 3.3-1 "Register bank structure" shows the register bank structure.

Figure 3.3-1 Register bank structure



Reference:

See Section 3.1.1 "Special Areas" for the general-purpose register area available for each product.

■ Features of general-purpose registers

General-purpose registers have the following features:

- RAM can be accessed at high-speed using short instructions (general-purpose register addressing).
- Registers are grouped in blocks in the form of register banks. This simplifies the process of saving register contents and dividing registers by function.

Dedicated register banks can be permanently assigned for each interrupt processing or vector call (CALLV #0 to #7) processing routine by general-purpose register. For example, register bank 4 interrupt 2.

For example, a particular interrupt processing routine only uses a particular register bank which cannot be written to unintentionally by other routines. The interrupt processing routine only needs to specify its dedicated register bank at the start of the routine to effectively save the general-purpose registers in use prior to the interrupt. Therefore, saving the general-purpose registers to the stack or other memory location is not necessary. This allows high-speed interrupt handling while maintaining simplicity.

Also, as an alternative to saving general-purpose registers in subroutine calls, register banks can be used to create reentrant programs (programs that do not use fixed addresses and can be entered more than once) usually made by the index register (IX).

Precaution:

If an interrupt processing routine changes the register bank pointer (RP), ensure that the program does not also change the interrupt level bits in the condition code register (CCR: IL1, IL0) when specifying the register bank.

3.4 Interrupts

The MB89470 series has 15 interrupt request input corresponding to peripheral functions. An interrupt level can be set independently.
If an interrupt request output is enabled in the peripheral function, an interrupt request from a peripheral function is compared with the interrupt level in the interrupt controller. The CPU performs interrupt operation according to how the interrupt is accepted. The CPU wakes up from standby modes, and returns to the interrupt or normal operation.

■ Interrupt requests from peripheral functions

Table 3.4-1 "Interrupt request and interrupt vector" lists the interrupt requests corresponding to the peripheral functions. On acceptance of an interrupt, execution branches to the interrupt processing routine. The contents of interrupt the vector table address corresponding to the interrupt request specifies the branch destination address for the interrupt processing routine.

An interrupt processing level can be for each interrupt request in the interrupt level setting registers (ILR1, ILR2, ILR3, ILR4). Three levels are available.

If an interrupt request with the same or lower level occurs during execution of an interrupt processing routine, the latter interrupt is not normally processed until the current interrupt processing routine completes. If interrupt request set the same level occur simultaneously, the highest priority is IRQ0.

Table 3.4-1 Interrupt request and interrupt vector

Interrupt request	Vector table address		Bit names of the interrupt level setting register	Simultaneously-generated same-level IRQ priority
	Upper	Lower		
IRQ0 (External interrupt(edge) INT10	$FFFA_H$	$FFFB_H$	L01, L00	High ↑ ↓ Low
IRQ1 (External interrupt(edge) INT11	$FFF8_H$	$FFF9_H$	L11, L10	
IRQ2 (External interrupt(edge) INT12	$FFF6_H$	$FFF7_H$	L21, L20	
IRQ3 (External interrupt(edge) INT13	$FFF4_H$	$FFF5_H$	L31, L30	
IRQ4 (External interrupt(level) 2)	$FFF2_H$	$FFF3_H$	L41, L40	
IRQ5 (UART/SIO 1)	$FFF0_H$	$FFF1_H$	L51, L50	
IRQ6 (UART/SIO 2)	$FFEE_H$	$FFEF_H$	L61, L60	
IRQ7 (8/16-Bit Timer Ch1)	$FFEC_H$	$FFED_H$	L71, L70	
IRQ8 (8/16-Bit Timer Ch2)	$FFEA_H$	$FFEB_H$	L81, L80	
IRQ9 (A/D converter)	$FFE8_H$	$FFE9_H$	L91, L90	
IRQA (PWM)	$FFE6_H$	$FFE7_H$	LA1, LA0	
IRQB (PWC)	$FFE4_H$	$FFE5_H$	LB1, LB0	
IRQC (Timebase timer)	$FFE2_H$	$FFE3_H$	LC1, LC0	
IRQD (Watch prescaler)	$FFE0_H$	$FFE1_H$	LD1, LD0	
IRQE (Vacancy)	$FFDE_H$	$FFDF_H$	LE1, LE0	
IRQF (Vacancy)	$FFDC_H$	$FFDD_H$	LF1, LF0	

3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3, ILR4)

The interrupt level setting registers (ILR1, ILR2, ILR3, ILR4) together contain 16 blocks of 2-bit data, with each data corresponding to an interrupt request from a peripheral function. The interrupt level for each interrupt is set in that interrupt's corresponding 2-bit data (interrupt level setting bits).

■ Structure of interrupt level setting registers (ILR1, ILR2, ILR3, ILR4)

Figure 3.4-1 Structure of interrupt level setting registers

Register	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
ILR1	007B$_H$	L31	L30	L21	L20	L11	L10	L01	L00	11111111$_B$
		W	W	W	W	W	W	W	W	
ILR2	007C$_H$	L71	L70	L61	L60	L51	L50	L41	L40	11111111$_B$
		W	W	W	W	W	W	W	W	
ILR3	007D$_H$	LB1	LB0	LA1	LA0	L91	L90	L81	L80	11111111$_B$
		W	W	W	W	W	W	W	W	
ILR4	007E$_H$	LF1	LF0	LE1	LE0	LD1	LD0	LC1	LC0	11111111$_B$
		W	W	W	W	W	W	W	W	

W: Write-only

Two bits of the interrupt level setting registers are allocated to each interrupt request. The value of the interrupt level setting bits in these registers sets the interrupt priority (interrupt levels 1 to 3).

The interrupt level setting bits are compared with the interrupt level bits in the condition code register (CCR: IL1, IL0).

The CPU does not accept interrupt requests set to interrupt level 3.

Table 3.4-2 "Interrupt level setting bit and interrupt level" shows the relationship between the interrupt level setting bits and the interrupt levels.

Table 3.4-2 Interrupt level setting bit and interrupt level

L01 to LF1	L00 to LF0	Request interrupt level	Priority
0	0	1	High ↕ Low (no interrupt)
0	1		
1	0	2	
1	1	3	

Note:

The interrupt level bits in the condition code register (CCR: IL1, IL0) are normally "11_B" during main program execution.

Precaution:

As the IRL1, ILR2, ILR3 and ILR4 registers are write-only, the bit manipulation instructions cannot be used.

3.4.2 Interrupt Processing

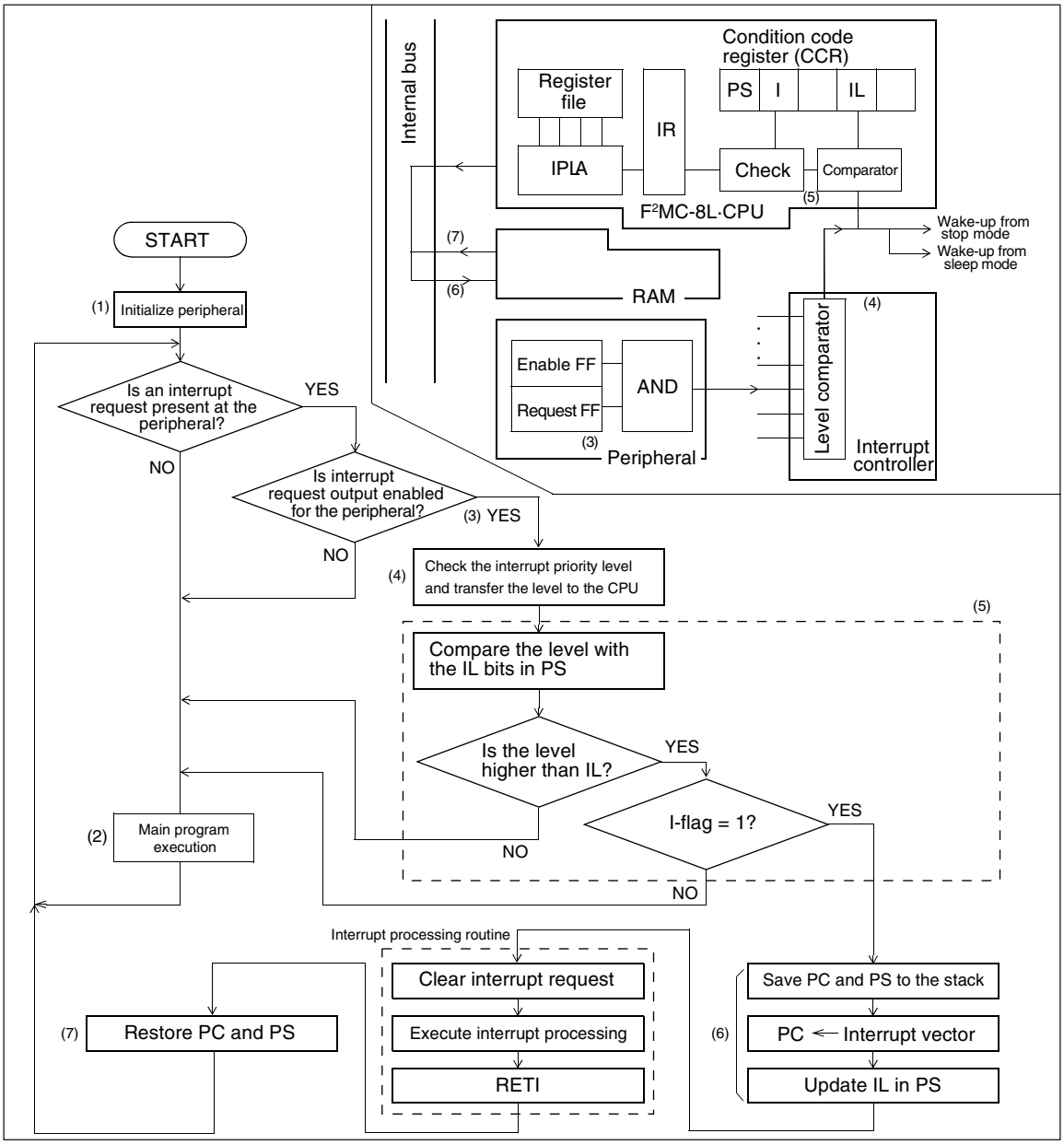
The interrupt controller transmits the interrupt level to the CPU when an interrupt request is generated by a peripheral function. If the CPU is able to receive the interrupt, the CPU temporarily halts the currently executing program and executes the interrupt processing routine.

■ Interrupt processing

The procedure for interrupt operation is performed in the following order: interrupt source generated at peripheral function, set the interrupt request flag bit (request FF), discriminate the interrupt request enable bit (enable FF), the interrupt level (ILR1, ILR2, ILR3, ILR4 and CCR: IL1, IL0), simultaneously generated interrupt requests with the same level, then check the interrupt enable flag (CCR: I).

Figure 3.4-2 "Interrupt processing" shows the interrupt processing.

Figure 3.4-2 Interrupt processing



1. After a reset, all interrupt requests are disabled.
   - Initialize the peripheral functions that are to generate interrupts in the peripheral function initialization program, set the interrupt levels in the appropriate interrupt level setting registers (ILR1, ILR2, ILR3, ILR4), and start peripheral function.
   - The interrupt level can be set to 1, 2 or 3. Level 1 is the highest priority, followed by level 2. Setting level 3 disables the interrupt for that peripheral function.
2. Execute the main program (for multiple interrupts, execute the interrupt processing routine).
3. The interrupt request flag bit (request FF) for a peripheral function is set to "1" when the peripheral function generates an interrupt source. If the interrupt request enable bit for the peripheral function is set to "enable" (enable FF = "1"), the peripheral function outputs the interrupt request to the interrupt controller.

4. The interrupt controller continuously monitors for interrupt requests from the peripheral functions and passes the interrupt level of the current interrupt request with the highest interrupt level to the CPU. The interrupt controller also evaluates the priority order if requests with the same level are present simultaneously.
5. If the interrupt level received by the CPU has a higher priority (a lower level value) than the level set in the interrupt level bits in the condition code register (CCR: IL1, IL0), the CPU checks the interrupt enable flag (CCR: I) and receives the interrupt if interrupts are enabled (CCR: I = "1").
6. The CPU saves the contents of the program counter (PC) and program status (PS) on the stack, reads the top address of the interrupt processing routine from the interrupt vector table for the interrupt, updates the interrupt level bits in the condition code register (CCR: IL1, IL0) with the received interrupt level, and starts execution of the interrupt processing routine.
7. Finally, on execution of the RETI instruction, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution from the instruction following the last instruction executed before the interrupt.

Precaution:

As the interrupt request flag bit of a peripheral function is not cleared automatically when an interrupt request is received, the bit must be cleared by the program (normally, by writing "0" to the interrupt request flag bit) at interrupt processing routine.

Reference:

An interrupt wakes up the CPU from standby mode (low-power consumption). See Section 3.7, "Standby Modes (Low-power Consumption)" for details.

Note:

If the interrupt request flag bit is cleared at the top of the interrupt processing routine, the peripheral function that has generated the interrupt becomes able to generate another interrupt during execution of the interrupt processing routine (resetting the interrupt request flag bit). However, the interrupts are not normally accepted until the current processing routine completes.

3.4.3 Multiple Interrupts

Multiple interrupts can be performed by setting different interrupt levels to the interrupt level setting register for two or more interrupt requests from peripheral functions.

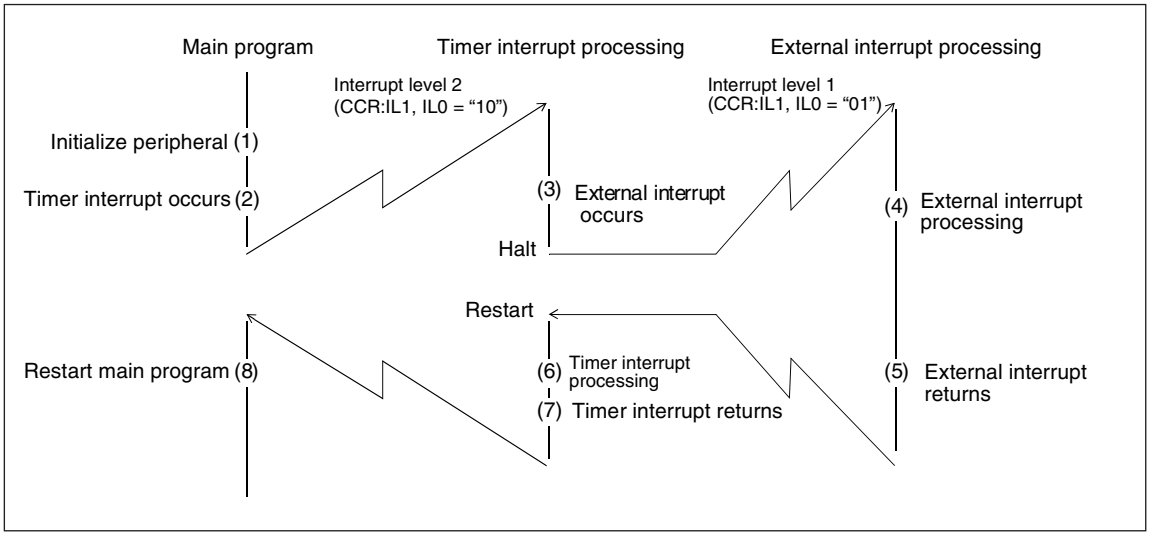
■ Multiple interrupts

If the interrupt request having the higher interrupt levels occurs during the interrupt processing routines, the CPU halts the current interrupt process and switches to accept the interrupt with the higher priority. Interrupt levels can be set in the range 1 to 3. However, the CPU does not accept interrupt requests set to interrupt level 3.

● Example of multiple interrupts

As an example of multiple interrupt processing, assume that an external interrupt has a higher priority than the timer interrupt. The timer interrupt is set to level 2 and the external interrupt is set to level 1. Figure 3.4-3 "Example of multiple interrupts" shows the processing when the external interrupt occurs during execution of timer interrupt processing.

Figure 3.4-3 Example of multiple interrupts



- During execution of timer interrupt processing, the interrupt level bits in the condition code register (CCR:IL1, IL0) are automatically set to the same value as the interrupt level setting register (ILR1, ILR2, ILR3, ILR4) corresponding to the timer interrupt (level 2 in this example). If the interrupt request set to higher interrupt level (level 1 in this example) occurs at this time, the interrupt processing has priority.
- To temporarily disable multiple interrupts during the timer interrupt, the interrupt enable flag in the condition code register is set to "interrupts disabled" (CCR: I = "0") or the interrupt level bits (IL1, IL0) set to "00_B".
- On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution of the interrupted program.

Restoring the program status (PS) returns the condition code register (CCR) to the value prior to the interrupt.

3.4.4 Interrupt Processing Time

The total time from the generation of an interrupt request until control passes to the interrupt processing routine is the sum of the time required to complete execution of the current instruction and the interrupt handling time (the time required to prepare for interrupt processing). The maximum time for this process is 30 instruction cycles.

■ Interrupt processing time

When an interrupt request occurs, the time until the interrupt is accepted and the interrupt processing routine is executed includes the interrupt request sampling time and the interrupt handling time.

● Interrupt request sampling time

Whether or not an interrupt request has occurred is determined by sampling and testing for interrupt requests during the final cycle of each instruction. Therefore, the CPU is unable to identify interrupt requests during execution of an instruction. The longest delay occurs when an interrupt request is generated immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles).

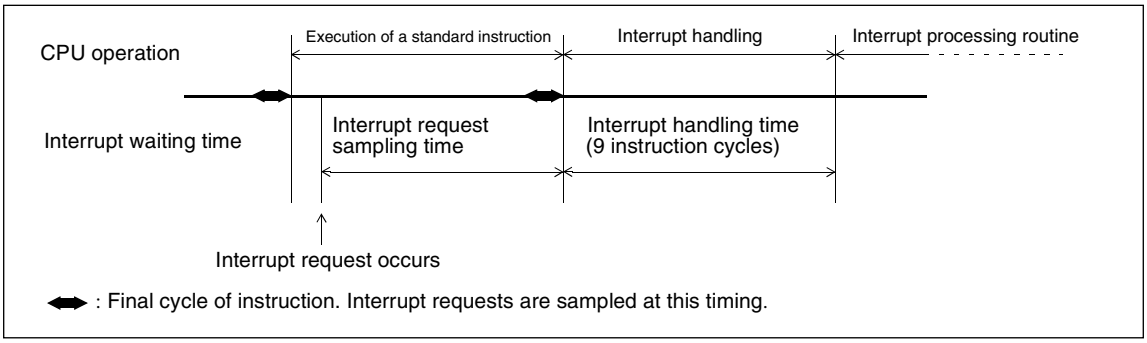
● Interrupt handling time

Nine instruction cycles are required to perform the following preparation for interrupt processing after the CPU accepts an interrupt request:

- Save the program counter (PC) and program status (PS).
- Set the top address of the interrupt processing routine (the interrupt vector) in the PC.
- Update the interrupt level bits (PS:CCR: IL1, IL0) in the program status (PS).

Figure 3.4-4 "Interrupt processing time" shows the interrupt processing time.

Figure 3.4-4 Interrupt processing time



The total interrupt processing time of 21 + 9 = 30 instruction cycles is required if an interrupt request occurs immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles). If, on the other hand, the program does not use the DIVU or MULU instructions, the maximum interrupt processing time is 6 + 9 = 15 instruction cycles.

Reference:

The time of one instruction cycle changes with the clock mode and the main clock frequency as selected by the "speed-shift" (gear) function. See Section 3.6, "Clocks" for details.

3.4.5 Stack Operation during Interrupt Processing

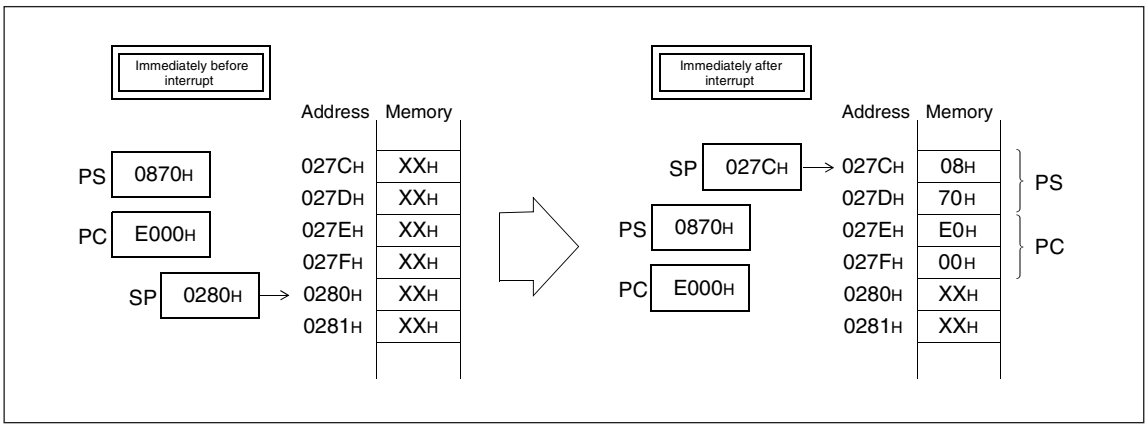
This section describes the saving of the register contents to the stack and restore operation during interrupt processing.

■ Stack operation at start of interrupt processing

The CPU automatically saves the current contents of the program counter (PC) and program status (PS) to the stack when an interrupt is accepted.

Figure 3.4-5 "Stack operation at start of interrupt processing" shows the stack operation at the start of interrupt processing.

Figure 3.4-5 Stack operation at start of interrupt processing



■ Stack operation at interrupt return

On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU performs the opposite processing to interrupt initiation, restoring first the program status (PS) and then the program counter (PC) from the stack. This returns the PS and PC to their states immediately prior to the start of the interrupt.

Precaution:

The CPU does not automatically save the accumulator (A) or temporary accumulator (T) contents to the stack. Use the PUSHW and POPW instructions to save and restore A and T contents to and from the stack.

3.4.6 Stack Area for Interrupt Processing

Interrupt processing execution uses the stack area in RAM. The contents of the stack pointer (SP) specifies the top address of the stack area.

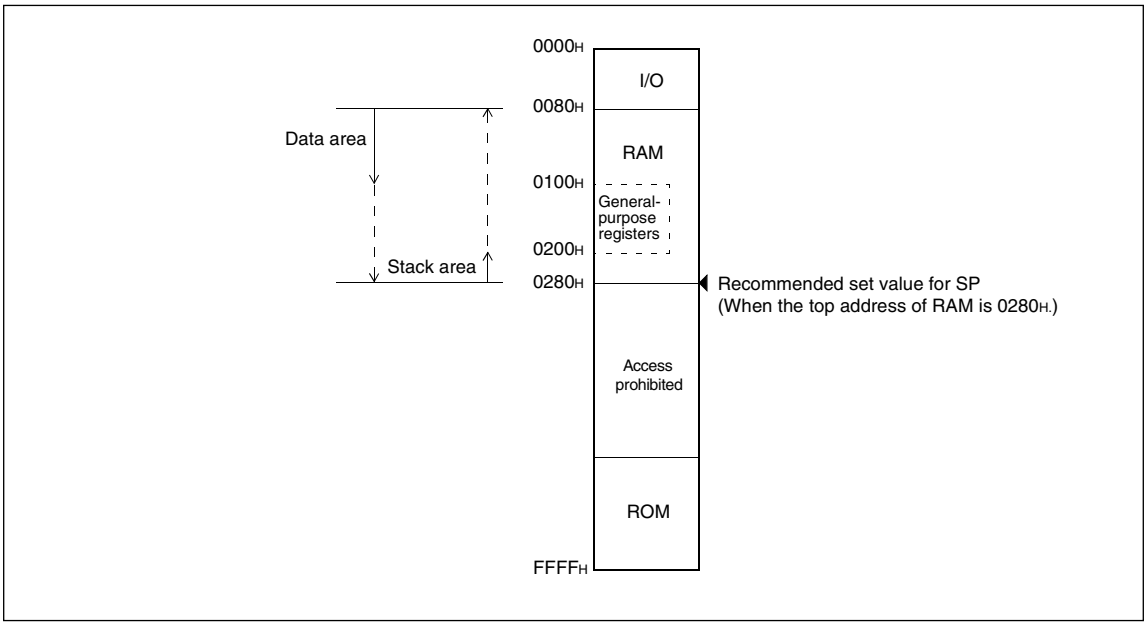
■ Stack area for interrupt processing

The subroutine call instruction (CALL) and vector call instruction (CALLV) use the stack area to save and restore the program counter (PC). The stack area is also used by the PUSHW and POPW instructions to temporarily save and restore registers.

- The stack area is located in RAM along with the data area.
- Initializing the stack pointer (SP) to the top address of RAM and allocating data areas upwards from the bottom RAM address is recommended.

Figure 3.4-6 "Stack area for interrupt processing" shows the example of stack area setting.

Figure 3.4-6 Stack area for interrupt processing



Note:

The stack area is used in the downward direction starting from a high address by functions such as interrupts, subroutine calls, and the PUSHW instruction. Instructions such as return instructions (RETI, RET) and the POPW instruction release stack area in the upward direction. Take care when the stack address is decreased by multiple interrupts or subroutine calls that the stack does not overlap the general-purpose register area or areas containing other data.

3.5 Resets

The MB89470 series supports the following four types of reset source:

- **External reset**
- **Software reset**
- **Watchdog reset**
- **Power-on reset**

The pin states during a reset or during reset operation depend on the operating mode.

■ Reset source

Table 3.5-1 Reset source

Reset source	Reset conditions
External reset	Set the external reset pin to the "L" level.
Software reset	Write "0" to the software reset bit in the standby control register (STBC: RST).
Watchdog reset	Watchdog timer overflow
Power-on reset	Power is turned on.

● External reset

Inputting an "L" level to the external reset pin ($\overline{RST}$) generates an external reset. Returning the reset pin to the "H" level disables the external reset.

When power is turned on or for external resets in stop mode, the reset operation is performed after the oscillation stabilization wait time has passed and the external reset is released.

The external reset pin can also function as a reset output pin.

● Software reset

Writing "0" to the software reset bit in the standby control register (STBC: RST) generates a four-instruction cycle reset.

● Watchdog reset

The watchdog reset generates a four-instruction cycle reset if data is not written to the watchdog timer control register (WDTC) within a fixed time after the watchdog timer starts.

● Power-on reset

Turning on the power generates a reset. The reset operation is performed after the oscillation stabilization wait time has passed.

■ Main clock oscillation stabilization wait time and the reset source

Whether there will be an oscillation stabilization wait time depends on the operating mode when reset occurs.

Following reset, operation always starts out in the normal main clock operating mode, regardless of the kind of reset it was, or the operating mode (standby mode) prior to reset. Therefore, if reset occurs while the main clock oscillator is stopped or in a stabilization wait time, the system will be in a "main clock oscillation stabilization reset" state, and a clock stabilization period will be provided.

In software or watchdog reset, if the reset occurs while the device is in main clock mode, no stabilization time is provided. If it occurs in the subclock mode, however, a stabilization time is provided since the main clock oscillation is stopped.

Table 3.5-2 shows the relationships between the reset sources and the main clock oscillation stabilization wait time.

Table 3.5-2 Reset source and oscillation stabilization wait time

Reset source	Operating state	Reset operation and main clock oscillation stabilization wait time
		Reset
External reset *	At power on, during stop mode	After the main clock oscillation stabilization wait time, if the external reset is waked up, reset is operated. "L" is output at $\overline{RST}$ pin during the main clock oscillation stabilization wait time.
Software and watchdog reset	Main clock mode	After 4-instruction-cycle reset occurs, reset is operated. "L" is output at $\overline{RST}$ pin during 4-instruction-cycle.
	Subclock mode	Reset is operated after the main clock oscillation stabilization wait time. "L" is output at $\overline{RST}$ pin during the main clock oscillation stabilization wait time.
Power-on reset		Device enters main clock oscillation stabilization wait time at power on. Reset is operated after wait time ends. "L" is output at $\overline{RST}$ pin during the main clock oscillation stabilization wait time.

* : No oscillation stabilization wait time is required for external reset while main clock mode is operating. Reset is operated after external reset is waked up.

3.5.1 Reset Flag Register (RSFR)

The reset flag register (RSFR) can be used to identify the reset generation sources when a reset occurs.

■ Reset flag register (RSFR)

Figure 3.5-1 Reset flag register (RSFR)

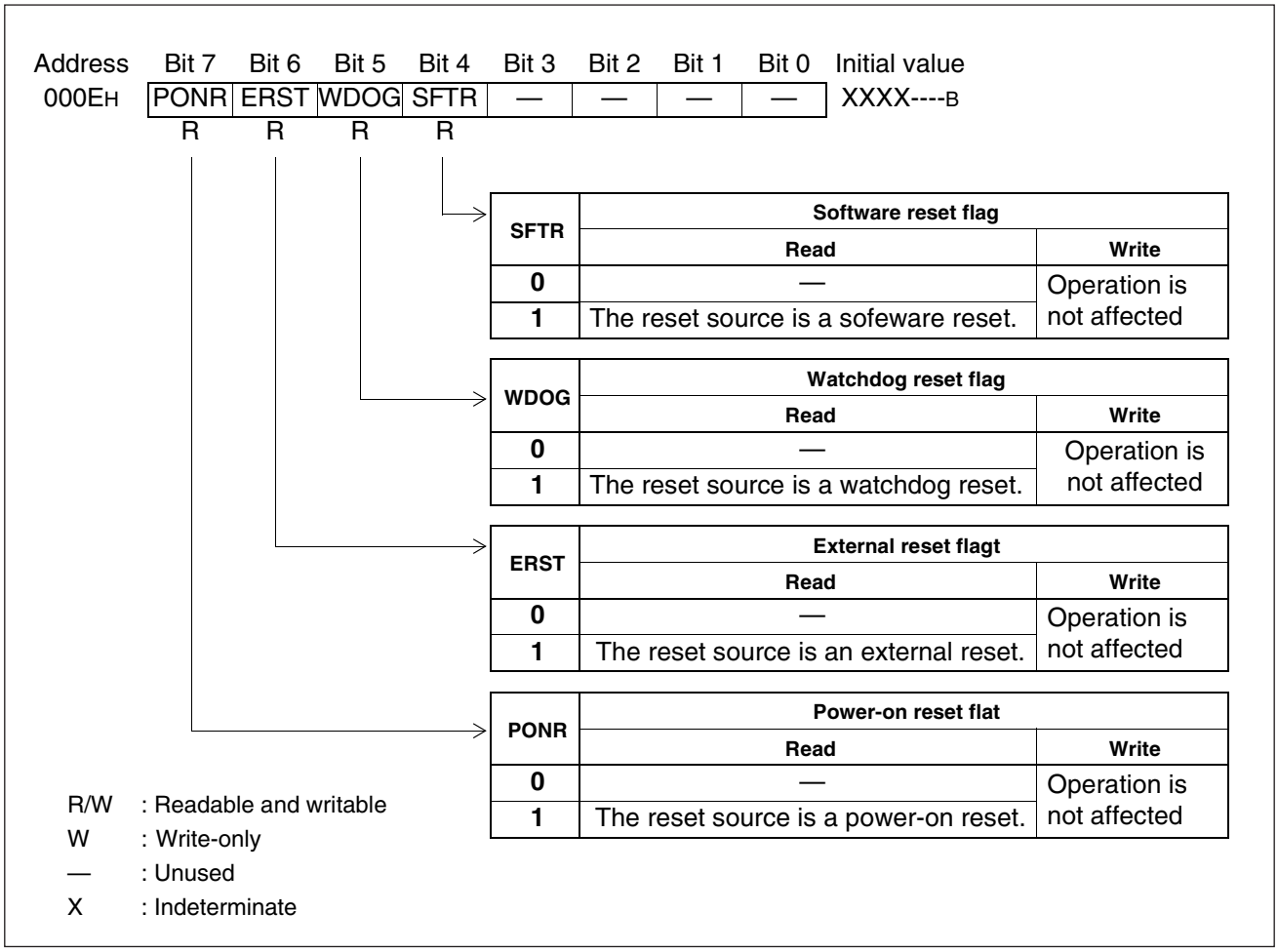


Table 3.5-3 Explanation of the Functions of Each Bit of the Reset Flag Register (RSFR) (Continued)

Bit		Function
Bit 7	PONR: Power-on reset flag	• "1" is set when a power-on reset occurs. • "1" is set after power-on. • This register is cleared to "0" by reading it. • Writing to this bit has no effect.
Bit 6	ERST: External reset flag	• "1" is set when an external reset occurs. • "1" is set to the software reset flag while retaining each reset flag if each reset flag is set before the external reset flag is set. • This register is cleared to "0" by reading it. • Writing to this bit has no effect.
Bit 5	WDOG: Watchdog reset flag	• "1" is set when a watchdog reset occurs. • "1" is set to the watchdog reset flag while retaining each reset flag if each reset flag is set before the watchdog reset flag is set. • This register is cleared to "0" by reading it. • Writing to this bit has no effect.
Bit 4	SFTR: Software reset flag	• "1" is set when a software reset occurs. • "1" is set to the software reset flag while retaining each reset flag if each reset flag is set before the software reset flag is set. • This register is cleared to "0" by reading it. • Writing to this bit has no effect.
Bit 3 Bit 2 Bit 1 Bit 0	Unused bit	• Values during a read operation are undefined. • Writing to this bit has no effect.

Note:

Each reset source flag is set when each reset source occurs. When each reset source flag register is read, each bit of the reset source flag register is cleared. Thus, determine the reset source by reading this register using the initialization routine after the reset.

3.5.2 External Reset Pin

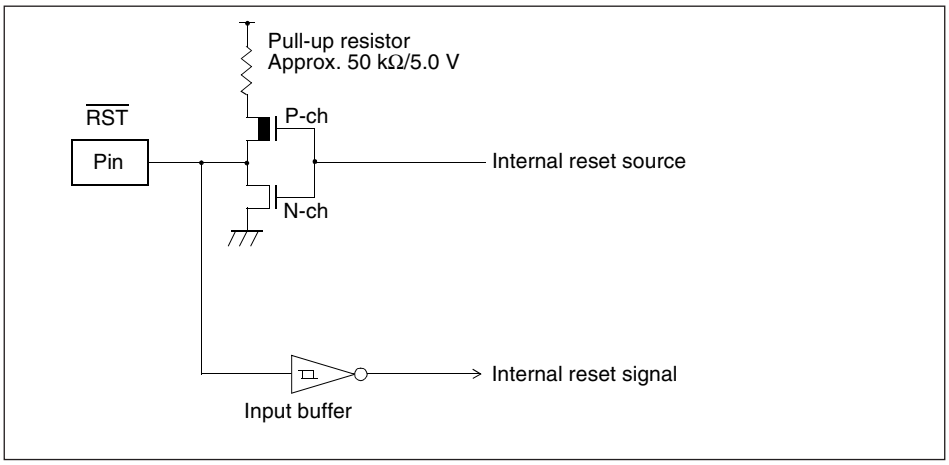
Inputting an “L” level to the external reset pin generates a reset. The pin outputs an “L” level when internal reset occurs.

■ Block diagram of external reset pin

The external reset pin ($\overline{RST}$) is a hysteresis input type and N-ch open-drain output type with a pull-up resistor.

Figure 3.5-2 "Block diagram of external reset pin" shows the block diagram of the external reset pin.

Figure 3.5-2 Block diagram of external reset pin



■ External reset pin functions

Inputting an "L" level to the external reset pin ($\overline{RST}$) generates an internal reset signal.

The pin outputs an "L" level when internal reset occurs or during the oscillation stabilization wait time due to an external reset. Software reset, watchdog reset, and power-on reset are classed as internal reset sources.

Precaution:

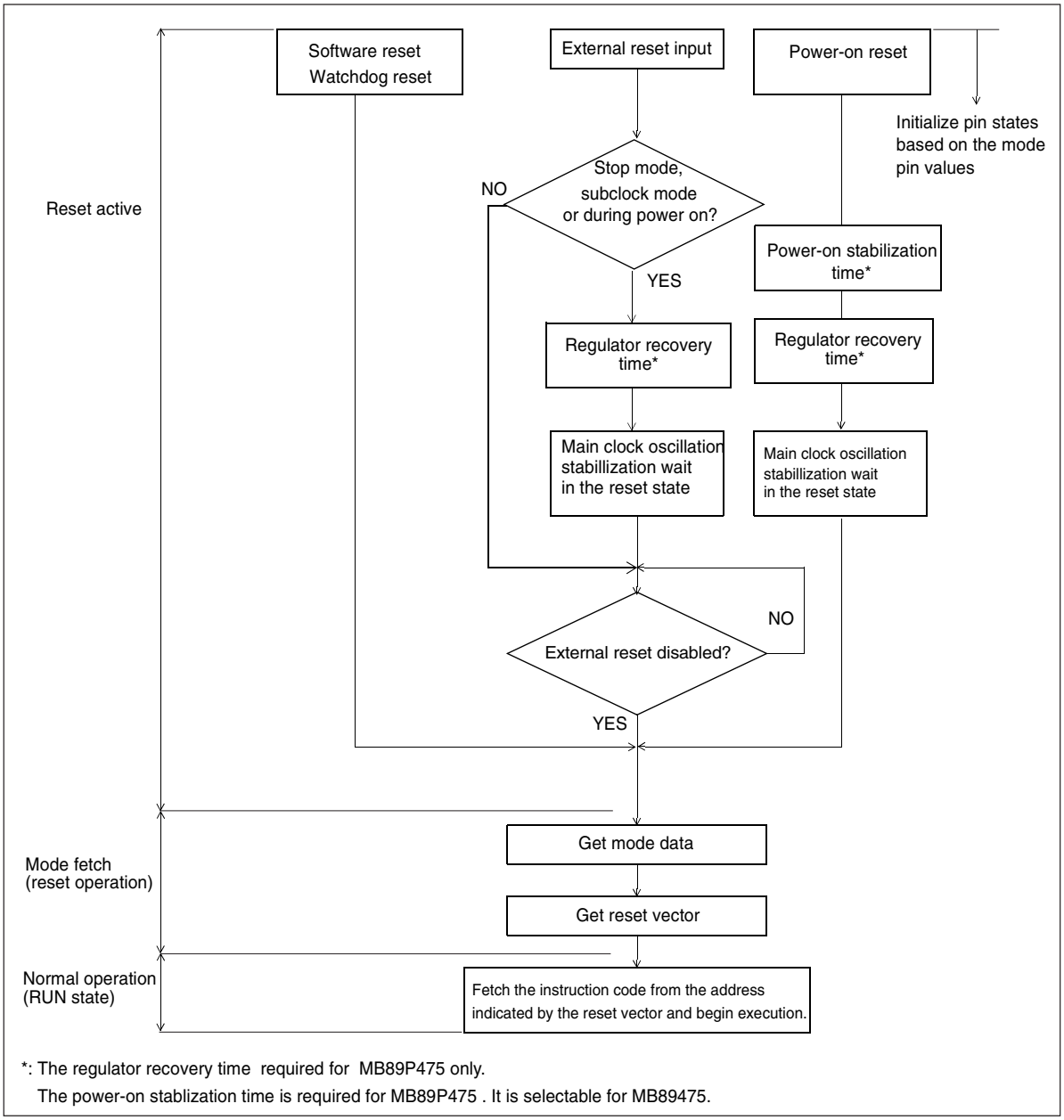
- The external reset input accepts asynchronous with the internal clock. Therefore, initialization of the internal circuit requires a clock. Especially when an external clock is used, a clock is needed to be input at the reset.
- If the reset pulse applied to the external reset pin ($\overline{RST}$) does not meet the specifications, it may cause malfunctions. Use caution so that the reset pulse less than the specifications will not be fed to the external reset pin ($\overline{RST}$).

3.5.3 Reset Operation

When the CPU wakes up from a reset, the CPU selects the read address of the mode data and reset vector according to the mode pin settings, then performs a mode fetch. The mode fetch is performed after the oscillation stabilization wait time has passed when power is turned on, or when the CPU wakes up from stop mode by a reset. If reset occurs during a write to RAM, the contents of the RAM address cannot be assured.

■ Overview of reset operation

Figure 3.5-3 Reset operation flow diagram



*: The regulator recovery time required for MB89P475 only.
The power-on stablization time is required for MB89P475 . It is selectable for MB89475.

■ Mode pins

The MB89470 series devices are single-chip mode devices. The mode pins (MODE) must be tied to V_{SS}.

Do not change the mode pin settings, even after the reset has been completed.

■ Mode fetch

When the CPU wakes up from a reset, it reads the mode data and reset vector from internal ROM.

● Mode data (address: $FFFD_H$)

Always set the mode to "00_H" (single-chip mode).

● Reset vector (address: $FFFE_H$ (upper), $FFFF_H$ (lower))

Contains the address where execution is to start after completion of the reset. The CPU starts executing instructions from the address contained in the reset vector.

■ Oscillation stabilization wait reset state

The reset operation for a power-on reset or external reset in stop mode starts after the oscillation stabilization wait time determined by the time-based timer. If the external reset input has not been released when the wait time completes, the reset operation does not start until the external reset is released.

As the oscillation stabilization wait time is also required when an external clock is used, a reset requires that the external clock is input.

The main clock oscillation stabilization wait time is timed by the timebase timer.

■ Effect of reset on RAM contents

The contents of RAM are unchanged before and after a reset other than power-on reset. If an external reset is input close to a write timing, however, the contents of the write address cannot be assured. For this reason, all RAM locations being used should be initialized following reset.

■ Regulator recovery time

MB89P475 series contains an on-chip voltage regulator for the internal 3V circuits. In power-down mode or in the stop mode, the internal voltage regulator is turned off to minimize power consumption. The Regulator Recovery Time is the the time for the internal voltage regulator to resume normal operation.

The Regulator Recovery Time is 20µ second long, and it is required only in MB89P475.

■ Power-on stabilization time

When power-on, the on-chip voltage regulator requires a stabilization time in additional to the oscillation stabilization time. This additional wait time is called power-on Stabilization Time. The external power supply voltage must reach the mininum operation voltage within the Power-On stabilization time. For product without regulator, it is selectable.

The Power-On Stabilization Time is required in MB89P475 and selectable in MB89475. It depends on the oscillating clock, and it is $2^{17}/F_{CH}$ long (F_{CH} is the main clock operating frequency).

3.5.4 Pin States during Reset

Reset initialized the pin states.

■ Pin states during reset

When a reset source occurs, with a few exceptions, all I/O pins (peripheral pins) go to the high-impedance state and the mode data is read from internal ROM

■ Pin states after reading mode data

With a few exceptions, the I/O pins remain in the high-impedance state immediately after reading the mode data.

Precaution:

For devices connected to pins that change to high-impedance state when a reset source occurs take care that malfunction does not occur due to the change in the pin states.

Reference:

See Appendix E, "MB89470 Series Pin States" for pin states at times other than a reset.

3.6 Clocks

The clock generator is provided with two oscillators. By connecting with external resonators, the two circuits generate the high speed main clock and low speed subclock source oscillators. Alternatively, externally generated clock inputs can be used.
Clock controller controls the speed and supply of the dual-clock signals according to the clock and standby modes.

■ Clock supply map

Oscillation of a clock and its supply to the CPU and peripheral circuit (peripheral functions) are controlled by the clock controller. As shown in the map, operating clocks fed to the CPU and peripheral circuits are affected by main clock/subclock switching (clock mode), main clock speed switching (speed-shift function), and standby modes (sleep/stop/watch).

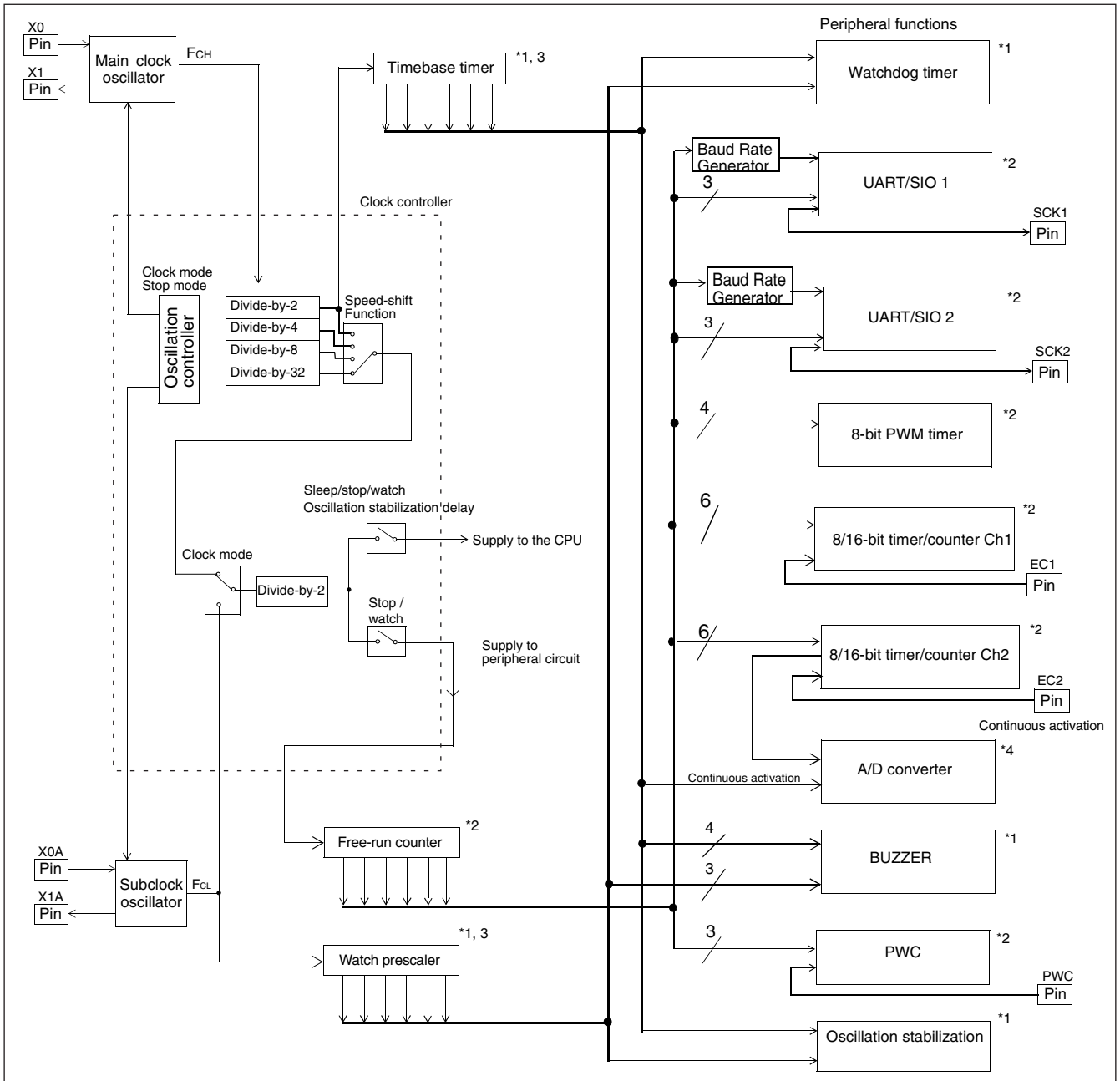
Divide-by-n output derived from the free-run counter clocked by the peripheral circuit clock is supplied to the peripheral functions.

Divide-by-n outputs from the timebase timer and watch prescaler are also supplied to the peripheral functions.

These clocks, however, are not affected by the speed-shift function, etc. The timebase timer is clocked by the output of the main clock source oscillator after it is fed through a divide-by-n circuit, and the watch prescaler is clocked directly by the subclock oscillator.

Figure 3.6-1 "Clock supply map" shows the clock supply map.

Figure 3.6-1 Clock supply map



FCH: Main clock source oscillator
FCL: Subclock source oscillator

*1: Not affected by clock mode, speed-shift function, etc.
*2: Operating speed, etc., affected by clock mode or speed-shift function.
*3: Stops operating if its clock source (main or subclock oscillator) stops.
*4: Timebase timer output can be selected in continuous A/D conversion operating mode.
In other modes, clock speed is affected by clock mode and speed-shift function.

3.6.1 Clock Generator

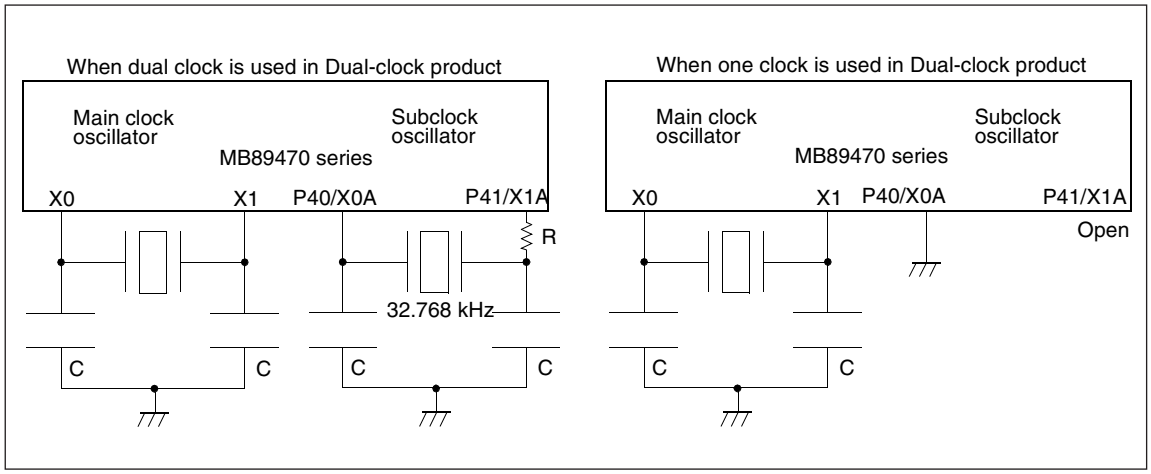
Enable and stop of the main clock oscillations is controlled by normal mode and stop modes.

■ Clock generator

● Crystal or ceramic resonator

Connect as shown in Figure 3.6-2 "Connection example for a crystal or ceramic resonator".

Figure 3.6-2 Connection example for a crystal or ceramic resonator



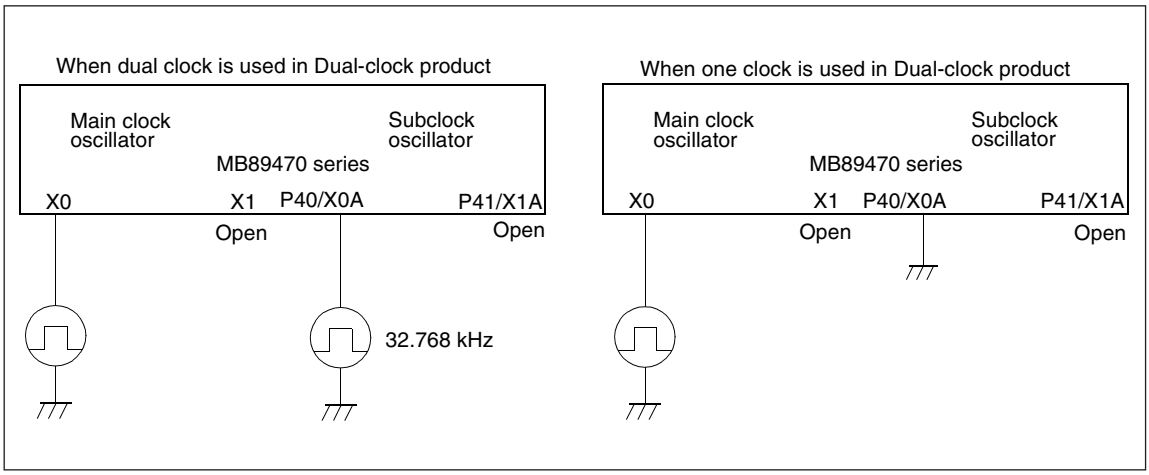
Note:

A piezoelectric resonator (FAR series) that contains the external capacitors can also be used. See Data Sheet for details.

● External clock

Connect the external clock to the X0 pin and leave X1 pin open, as shown in Figure 3.6-3 "Connection example for external clock". To use an external subclock source, connect the external clock to the X0A pin and leave the X1A pin open.

Figure 3.6-3 Connection example for external clock



● Single-clock product

For single-clock product, P40/X0A and P41/X1A function as input-only port. When they are not used, the input to the pin should be fixed at a certain level to prevent current leakage.

3.6.2 Clock Controller

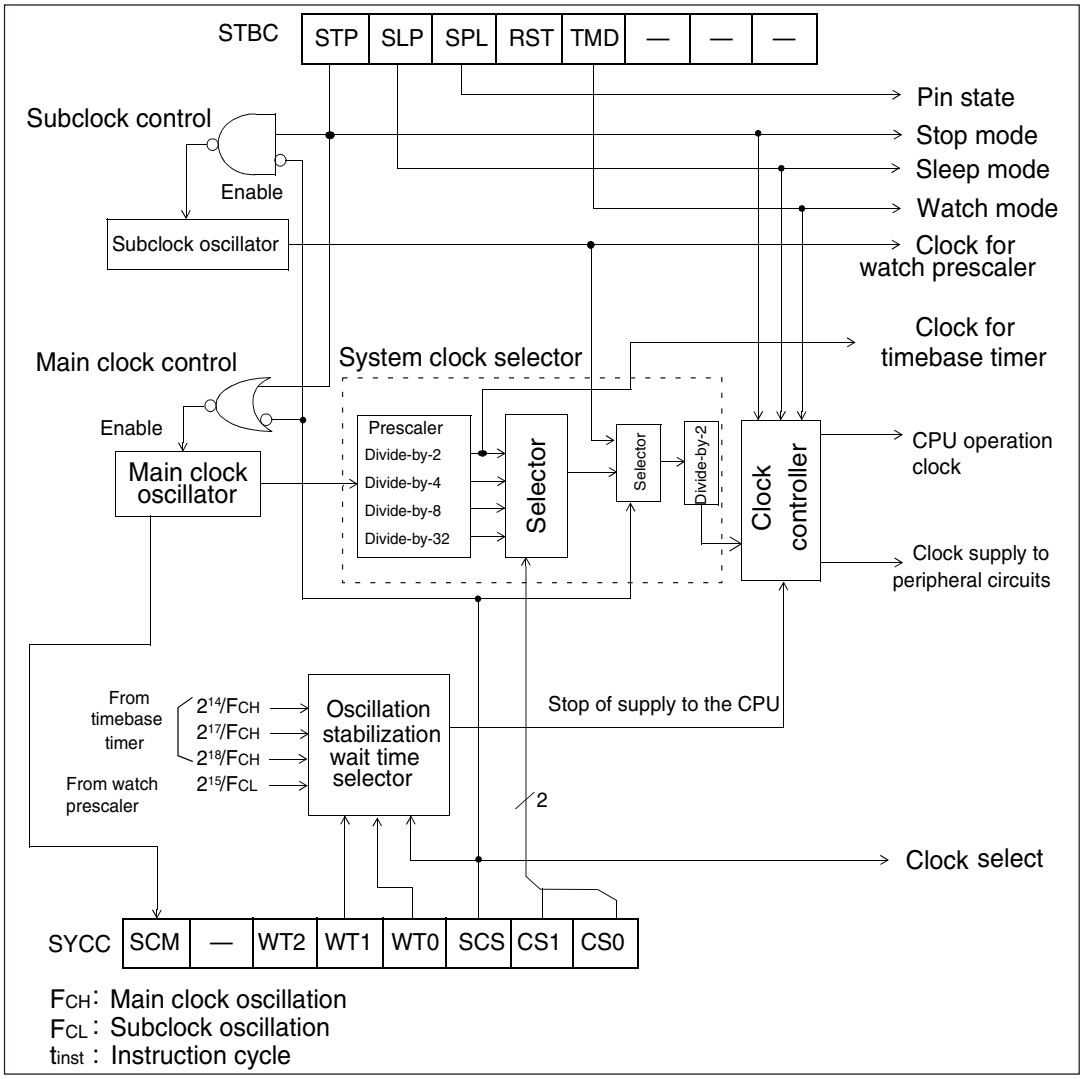
The clock controller contains the following seven blocks:

- **Main clock oscillator**
- **Subclock oscillator**
- **System clock selector**
- **Clock controller**
- **Oscillation stabilization wait time selector**
- **System clock control register (SYCC)**
- **Standby control register (STBC)**

■ Block diagram of clock controller

Figure 3.6-4 "Block diagram of clock controller" shows the block diagram of the clock controller.

Figure 3.6-4 Block diagram of clock controller



- **Main clock oscillator**

  The main clock oscillator is stopped in main-stop modes.

- **Subclock oscillator**

  The subclock oscillator is normally running except in sub-stop mode.

- **System clock selector**

  The system clock selector selects one of four clocks: one of four divided clocks derived from the main clock master clock oscillator.

- **Clock controller**

  This circuit controls the supply of operating clocks to the CPU and peripheral circuits, selecting the clock based on the active mode: normal (RUN), or standby (sleep/stop/watch) mode.

  Supply of the clock to the CPU is stopped until the clock supply stop signal in the oscillation stabilization wait time selector is released.

- **Oscillation stabilization wait time selector**

  This register selector selects a wait time from among three main clock oscillation stabilization times timed by the timebase timer oscillation stabilization time as the clock supply stop signal to the CPU based on the normal mode, standby mode and reset.

- **SYCC register**

  The SYCC register is used to select the speed of the main clock, and the main clock oscillation stabilization wait time, and to check the status of these selections.

- **STBC register**

  This register controls from normal operation (RUN) to the standby modes, sets the pin states in the stop mode, and initiates software reset.

3.6.3 System Clock Control Register (SYCC)

The system clock control register (SYCC) controls main clock/subclock switching, main clock speed selection, and oscillation stabilization wait time selection.

■ Structure of system clock control register (SYCC)

Figure 3.6-5 Structure of system clock control register (SYCC)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0007_H	SCM	—	WT2	WT1	WT0	SCS	CS1	CS0	X-1MM100B
	R		R/W	R/W	R/W	R/W	R/W	R/W	

CS1	CS0	Main clock speed select bit Instruction cycle time (F_{CH} = 12.5 MHz)
0	**0**	$64/F_{CH}$ (5.12µs)
0	**1**	$16/F_{CH}$ (1.28 µs)
1	**0**	$8/F_{CH}$ (0.64µs)
1	**1**	$4/F_{CH}$ (0.32µs)

SCS	System clock select bit
0	Selects subclock (32 kHz) mode
1	Selects main clock mode

WT1	WT0	Oscillation stabilization wait time select bit Main clock oscillation stabilization wait time per selected timebase timer output (F_{CH} = 12.5MHz)
0	**0**	Reserved
0	**1**	Approx. $2^{14}/F_{CH}$ (Approx. 1.3 ms)
1	**0**	Approx. $2^{17}/F_{CH}$ (Approx. 10.5 ms)
1	**1**	Approx. $2^{18}/F_{CH}$ (Approx. 21.0 ms)

WT2	Reserved bit
1	Always write one

SCM	System clock monitor bit
0	Subclock (main clock stopped or in oscillation stabilization wait time)
1	Main clock

R/W : Readable and writable
R : Read-only
— : Unused bit
X : Indeterminate
M : Per option selection
: Initial value
F_{CH} : Main clock oscillation (12.5MHz)

Table 3.6-1 System clock control register (SYCC) bits

Bit		Function
Bit 7	SCM: System clock monitor bit	• Indicates the current clock mode (operating clock). • "0" indicates subclock mode (main clock is stopped or in the oscillation stabilization wait time to go to main clock mode). • "1" indicates main clock mode. **Note:** This is a read-only bit. Writing to it has no effect.
Bit 6	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on operation.
Bit5	Reserved bit	• This bit must always set to 1
Bit 4 Bit 3	WT1, WT0: Oscillation stabilization wait time select bits	• Select main clock oscillation stabilization wait time. • Selected wait time applies if external interrupt causes "wakeup" from main-stop mode (transition to normal (run) mode). • Initial value of these bits is an option selection. Accordingly, when an oscillation stabilization wait time is provided at reset, the wait time will be as selected by the option. **Note:** These bits should not be changed at the same time switching from subclock to main clock (SCS = 1 --> 0). Before changing the bits, first check the SCM bit to verify that the device is not currently in the stabilization wait time.
Bit 2	SCS: System clock select bit	• Specifies the clock mode. • Writing "0" to this bit sets the CPU changing from main clock to subclock mode. • Writing "1" to this bit causes the device to go from subclock to main clock mode after the oscillation stabilization wait time set by WT1 and WT0 bits. **Note:** If the single clock option is selected, this bit has no function. It should be set to "1".
Bit 1 Bit 0	CS1, CS0: Main clock speed select bits	• These bits select the clock speed for the main clock mode. • Four different speeds can be set for CPU and peripheral function operating clocks (speed-shift function). The clock that clock the timebase timer is not affected by these bits.

■ Instruction cycle (t_{inst})

Instruction cycle (minimum execution time) can be selected as
1/4, 1/8, 1/16, or 1/64 of the main clock. The selection is made by main clock speed select bits (CS1 and CS0) of the SYCC register.

With main clock mode, and the highest clock speed selected (SYCC: SCS = 1, CS1 = 1, CS0 = 1), and with a main clock source oscillation (F_{CH}) of 12.5 MHz, the instruction cycle is $4/F_{CH}$ = approximately 0.32 µs.

With subclock mode selected (SCS = 0), and with a subclock source oscillation (F_{CL}) of 32.768 kHz, the instruction cycle is $2/F_{CL}$= approximately 61.0 µs.

3.6.4 Clock Modes

The clock modes consists of main clock mode and subclock mode.
In the main clock mode, the primary operating clock is provided by the main clock oscillator. The speed of the operating clock is selected by switching between one of four clocks obtained by dividing the output of the main clock oscillator (speed-shift function).
In the subclock mode, the main clock oscillator is stopped, and operating clocks are provided solely by the subclock.

■ Clock mode operating states

Table 3.6-2 Clock mode operating states

Clock mode	Main clock speed SYCC register (SYCC: CS1, CS0)		Standby mode	Clock Oscillator		Operating clocks for various sections				Non-reset event triggering exit from stanby
				Main	Sub	CPU	Timebase timer	Peripherals	Watch prescaler	
Main clock mode	(1.1)	Fast ↕ Slow	RUN	Oscillate	Oscillate	$F_{CH}/4$	$F_{CH}/2$	$F_{CH}/4$	F_{CL}	IRQ
			Sleep			Stop				
			Stop	Stop			Stop	Stop		External interrupt
	(1.0)		RUN	Oscillate	Oscillate	$F_{CH}/8$	$F_{CH}/2$	$F_{CH}/8$	F_{CL}	IRQ
			Sleep			Stop				
			Stop	Stop			Stop	Stop		External interrupt
	(0.1)		RUN	Oscillate	Oscillate	$F_{CH}/16$	$F_{CH}/2$	$F_{CH}/16$	F_{CL}	IRQ
			Sleep			Stop				
			Stop	Stop			Stop	Stop		External interrupt
	(0.0)		RUN	Oscillate	Oscillate	$F_{CH}/64$	$F_{CH}/2$	$F_{CH}/64$	F_{CL}	IRQ
			Sleep			Stop				
			Stop	Stop			Stop	Stop		External interrupt
Subclock mode	–		RUN	Stop	Oscillate	$F_{CL}/2$	Stop *1	$F_{CL}/2$	F_{CL}	IRQ
			Sleep			Stop				
			Stop		Stop			Stop	Stop	External interrupt
			Watch Mode	Stop	Oscillate	Stop	Stop *1	Stop	F_{CL}	External or watch interrupt

F_{CH}: Main clock source oscillation (12.5 MHz)
F_{CL}: Subclock source oscillation
*1: Since the timebase timer is derived from the main clock, it stops in subclock mode.

Reference:

See Section 3.7, "Standby Modes" for a description of the standby modes.

■ Speed-shift (main clock speed-switching) function

One of four main clock frequencies can be selected by writing the appropriate values between "00_B" and "11_B" to main clock speed select bits of the system clock control register (SYCC: CS1, CS0).

The switch-selected clock signal provides the operating clock for the CPU and peripheral circuits. The timebase timer and watch prescaler, however, is not affected by the speed-shift (gear) function.

A slower main clock speed reduces power consumption.

■ Operation of main clock mode

The main clock and the subclock oscillators both run in the "main-run" mode (the normal main clock operating mode). The watch prescaler runs on the subclock, but the CPU, timebase timer, and other peripheral circuits all use the main clock.

When operating in main clock mode, the speed-shift function can be used to select a main clock speed. This selection affects all circuits that are clocked by the main clock except for the timebase timer. By specifying a standby mode, you can also go to "main-sleep," or "main-stop" mode.

When the device is reset, the system always starts out in "main-run" mode regardless of how the reset was initiated. (Each operating mode exited by reset.)

● Changing from main clock mode to subclock mode

Writing "0" to the system clock select bit in the system clock control register (SYCC:SCS) changes the CPU from the main clock to subclock mode. You can determine which clock is currently being used by checking the system clock monitor bit of the same register (SYCC: SCM).

Precaution:

If you go to subclock mode immediately after power on, write the software so as to provide a longer subclock oscillation stabilization wait time than that defined by the watch prescaler.

■ Operation of subclock mode

In the normal subclock operating mode ("sub-run" mode), the system runs on the subclock only. The main clock oscillator is stopped. Using the low speed subclock reduces power consumption.

Other than the timebase timer, all functions operate the same in subclock mode as they do in main clock mode. If standby mode is specified while operating in subclock mode, the device goes to "sub-sleep," "sub-stop," or "watch" mode.

● Returning to main clock mode from subclock mode

Writing "1" to the system clock select bit in the system clock control register (SYCC:SCS) returns to main clock mode from subclock mode.

Operation from the main clock, however, will not start until after the main clock oscillation stabilization wait time has passed. One of three wait times can be selected by setting the oscillator stabilization wait time select bits of the system clock control register (SYCC: WT1, WT0).

Precaution:

Do not change the oscillation stabilization wait time select bits (SYCC: WT1, WT0) at the same time you switch from subclock to main clock mode (SYCC: SCS = 1), or during the oscillator stabilization wait time. Always check the system clock monitor bit to verify that the main clock is the active operating clock (SYCC: SCM = 1) before changing these bits.

The device always enters the oscillator stabilization wait time when the system is returned from subclock mode to main clock mode by reset.

3.6.5 Oscillation Stabilization Wait Time

When the system transitions to main-run mode from a state in which the main clock is stopped (such as at power-on, and in main-stop and subclock modes, etc.), a wait time is required for oscillation to stabilize before operation starts.
Similarly, a subclock oscillation stabilization wait time is required when exiting the sub-stop mode, because the subclock oscillator is stopped in that mode.

■ Oscillation stabilization wait time

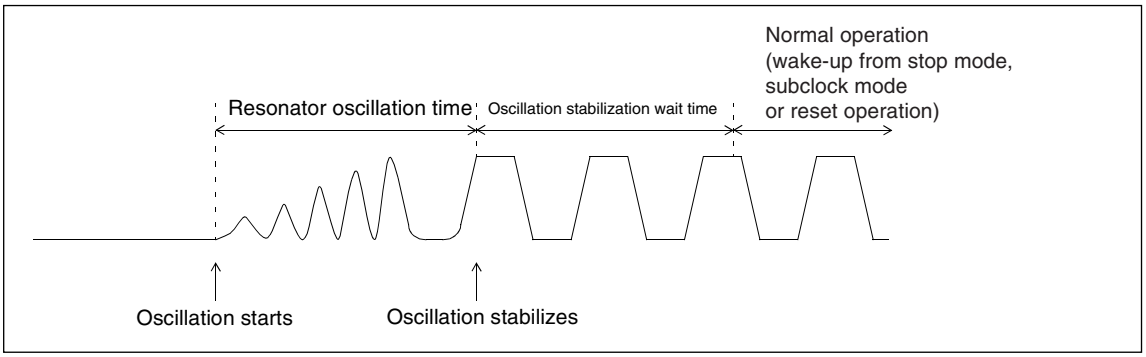
After starting, ceramic, crystal, and other resonators typically require the time between several milliseconds and several tens of milliseconds to stabilize at their fixed oscillation frequency.

Therefore, operation of the CPU and other functions is disabled when oscillation first starts and no clock signal is supplied to the CPU and peripheral functions until the oscillation stabilization wait time has passed and the oscillation has sufficiently stabilized.

The time required for oscillation to stabilize depends on the resonator type (crystal, ceramic, etc.) connected to the clock generator. Consequently, it is necessary to select an oscillation stabilization wait time that matches the type of oscillator being used.

Figure 3.6-6 "Operation of oscillator after starting oscillation" shows the operation of an oscillator after starting oscillation.

Figure 3.6-6 Operation of oscillator after starting oscillation



■ Regulator recovery time

MB89470 series contains an on-chip voltage regulator for the internal 3 V circuits. In power-down mode or in the stop mode, the internal voltage regulator is turned off to minimize power consumption. The Regulator Recovery Time is the the time for the internal voltage regulator to resume normal operation.

The Regulator Recovery Time is 20 μ second long, and it is required only in MB89P475.

■ Power-on stabilization time

When power-on, the on-chip voltage regulator requires a stabilization time in addition to the oscillation stabilization time. This additional wait time is called Power-On Stabilization Time. The external power supply voltage must reach the mininum operation voltage within the Power-On stabilization time.

The Power-On Stabilization Time is required in MB89P475 and MB89475. It depends on the oscillating clock, and it is $2^{17}/ F_{CH}$ long (F_{CH} is the main clock operating frequency).

■ Wait time for different operation for different product

Table 3.6-3 "Operating states while waiting oscillation delay" shows the operating states while waiting for the oscillation stabilization delay.

Table 3.6-3 Operating states while waiting oscillation delay

Operating Mode	Oscillation	CPU	Timebase timer	Watchdog timer	Other Peripherals	Pin	Cancel Method
Oscillation stabilization wait time	Start	Halted	Operating	Operating	Operating	Hold *	–
Power-On stabilization time	Started		Halted	Halted	Halted		
Regulator recovery time	Halted						

* : During an oscillation stabilization delay initiated by an external interrupt, pins go to their states prior to entering stop mode. Pins go to their reset states during an oscillation stabilization delay initiated by a reset.
(See Section 3.5.4 "Pin States During a Reset".)

Table 3.6-4 Wait time for different action

Operation	MB89475	MB89P475	MB89PV470
Power-on reset	Power-On stab. time (selectable) + Osc. stab. time	Regulator recovery time + Power-On stab. time + Osc. stab. time	Osc. stab. time
External reset in stop/ subclock/watch mode or external interrupt trigger recover from main stop mode	Osc. stab. time	Regulator recovery time + Osc. stab. time	Osc. stab. time

■ Main clock oscillation stabilization wait time

When first starting operation in main clock mode after a state in which the main clock oscillator is stopped, a wait time is required for oscillation to stabilize. This wait time starts when the timebase timer starts counting up from its cleared state, and ends when the count overflows at the specified bit.

● Oscillation stabilization wait time during operation

A time length must be selected for the oscillation stabilization wait time when an external interrupt takes the system from main-stop mode back to main-run mode, or when changing from subclock to main clock mode. One of three possible wait times can be selected, using the oscillator stabilization wait time select bits of the system clock control register (SYCC: WT1, WT0).

● Oscillation stabilization wait time at reset

The oscillation stabilization wait time at reset (the initial values of WT1 and WT0) is selected as an option setting.

Products require an oscillation stabilization wait time when exit from stop mode is triggered by reset in subclock mode, power-on reset, or external reset.

Table 3.6-5 "Main clock startup conditions vs. oscillation stabilization wait time" shows the relationships between the conditions in which main clock mode operation is started and oscillation stabilization wait time

Table 3.6-5 Main clock startup conditions vs. oscillation stabilization wait time

Main clock mode startup conditions	At power-on	During subclock mode		Exit from main-stop		Transition from subclock to main clock mode (SYCC: SCS *1 =1)
		External reset	Software reset and watchdog reset	External reset	External interrupt	
Oscillation stabilization wait time selection	Option setting				SYCC: WT1, WT0 *2	

*1: System clock select bit of system clock control register
*2: Oscillation stabilization wait time select bits of system clock control register

■ Subclock oscillation stabilization wait time

When an external interrupt returns the system from sub-stop (subclock oscillator stopped) to sub-run mode (thus starting the subclock oscillator), a set subclock oscillation stabilization wait time is provided. (This set wait time is equal to $2^{15}/F_{CL}$, where F_{CL} is the subclock oscillator frequency.)

The subclock oscillation stabilization wait time is also entered at power-on. Therefore, if you go to subclock mode after power on, you should insert a software delay, to provide a longer wait time before starting this transition than the subclock oscillation stabilization wait time alone.

The subclock oscillation stabilization wait time starts when the watch prescaler starts counting up from the cleared state, and ends when it overflows.

3.7 Standby Modes (Low-Power Consumption)

**The standby modes consist of sleep mode, stop mode, and watch mode.
From both main and subclock clock modes, standby modes are changed to sleep mode, stop mode, or watch mode by setting the standby control register (STBC).
From main clock mode, you can go only to sleep or stop mode, but from subclock mode you can go any of the three standby modes. Standby mode reduces the power consumption by stopping the operation of the CPU and peripheral functions.
This section describes the relationship between standby mode and clock mode, and the operation of various sections during standby.**

■ Standby modes

Watch mode reduce the power consumption by lowering the frequencies of the clocks that run the CPU and peripheral circuits. You can do this by switching from main clock to subclock mode, or by using the speed-shift function to select a lower main clock frequency. Standby mode reduce the power consumption, however, by stopping the clock signal supply to the CPU via clock controller (sleep mode), by stopping the clock signal supply to the CPU and peripheral circuits (watch mode), or by stopping the source oscillator itself (stop mode).

● Main-sleep mode

Main-sleep mode stops the CPU and watchdog timer, but operate the peripheral functions except watch prescaler by the main clock. (Certain functions can still run on the subclock.)

● Sub-sleep mode

Sub-sleep mode stops the main clock oscillator, CPU, watchdog timer, and timebase timer, but operate the peripheral functions on the subclock.

● Main-stop mode

Main-stop mode stops the CPU and peripheral functions. The main clock oscillator is stopped, but the subclock oscillator keeps running. Everything is shut down except external interrupt servicing, watch prescaler counter operation, and some functions that operate from the subclock.

● Sub-stop mode

Sub-stop mode stops all chip functions except the external interrupt, and stops the main clock and subclock oscillations.

● Watch mode

You can only go to watch mode from the subclock clock mode. All functions are shut down except the watch prescaler (watch interrupt), external interrupts, and some functions that operate from the subclock.

3.7.1 Operating States in Standby Modes

This section describes the operating states of the CPU and peripheral functions in standby modes.

■ Operating states during standby modes

Table 3.7-1 Operating states of the CPU and peripherals in standby modes

Function		Main clock mode			Subclock mode			
		RUN	Sleep	Stop	RUN	Sleep	Stop	Watch
Main clock		Operating	Operating	Stop	Stop	Stop	Stop	Stop
Subclock		Operating	Operating	Operating	Operating	Operating	Stop	Operating
CPU	Instructions	Operating	Stop	Stop	Operating	Stop	Stop	Stop
	ROM	Operating	Hold	Hold	Operating	Hold	Hold	Hold
	RAM							
Peripheral functions	I/O ports	Operating	Hold	Hold	Operating	Hold	Hold	Hold
	Watch prescaler	Operating	Operating	Operating *1	Operating	Operating	Stop	Operating
	Timebase timer	Operating	Operating	Stop	Stop	Stop	Stop	Stop
	8/16-bit timer/ counter 11/12, 21/22	Operating	Operating	Stop	Operating	Operating	Stop	Stop
	UART/SIO 1,2	Operating	Operating	Stop	operating	Operating	Stop	Stop
	8-bit PWM timers	Operating	Operating	Stop	Operating	Operating	Stop	Stop
	A/D converter	Operating	Operating	Stop	Operating	Operating	Stop	Stop
	External Interrupts 1 and 2	Operating	Operating	Operating	Operating	Operating	Operating	Operating
	Watchdog timer	Operating	Stop	Stop	Operating *2	Stop	Stop	Stop
	PWC timer	Operating	Operating	Stop	Operating	Operating	Stop	Stop
	BUZZER	Operating	Operating	Operating *2	Operating *2	Operating *2	Stop	Operating *2

*1: Watch prescaler counts but does not generate watch interrupts.
*2: Can be operated if watch prescaler output is selected as its operating clock.

● Pin states in standby mode

Almost all I/O pins will either keep the state they were placed in by the pin state control bit of the standby control register (STBC: SPL) just prior to going to the stop or watch mode, or will go to the high impedance state. This is true regardless of the clock mode.

Reference:

See Appendix E, "MB89470 Series Pin States." for pin states in a standby mode.

3.7.2 Sleep Mode

This section describes the operations of sleep mode.

■ Operation of sleep mode

● Changing to sleep mode

Sleep mode stops the CPU operating clock. The CPU stops while maintaining all register contents, RAM contents at their values immediately prior to entering sleep mode. However, peripheral functions except the watchdog timer continue to operate.

Writing "1" to the sleep bit in the standby control register (STBC: SLP) changes the CPU to sleep mode. If an interrupt request is generated when "1" is written to the SLP bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to sleep mode. (The CPU does not change to sleep mode even after completion of the interrupt processing.)

● Wake-up from sleep mode

A reset or an interrupt from a peripheral function wakes up the CPU from sleep mode.

If a reset occurs during sub-sleep mode on a product, the reset operation starts after the main clock oscillation stabilization wait time.

The reset operation also initializes the pin states.

- If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function or an external interrupt circuit during sleep mode, the CPU wakes up from sleep mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1 and IL0) in the CPU.
- The normal interrupt operation is performed after wake-up from sleep mode. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before changing to sleep mode.

3.7.3 Stop Mode

This section describes the operations of stop mode.

■ Operation of stop mode

● Changing to stop mode

Stop mode stops the source oscillation. Most functions stop while maintaining all register and RAM contents at their value immediately before changing to stop mode.

If the system is in main clock mode, the main clock oscillation stops. Except the external interrupt circuit, however, the CPU and other peripheral functions stop operating.

Writing "1" to the stop bit in the standby control register (STBC: STP) changes the CPU to stop mode. At this time, external pin states are held if the pin state specification bit (STBC: SPL) is "0". If SPL is "1", external pins go to the high-impedance state. (Pins with the pull-up resistor go to the "H" level.)

If an interrupt request is generated when "1" is written to the STP bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to stop mode. (The CPU does not assume stop mode even after completion of the interrupt processing.)

Prohibit interrupt request out from the timebase timer (TBTC: TBIE = "0") before changing to stop mode in main clock mode as necessary. Similarly, prohibit watch prescaler interrupt request output from the watch prescaler (WPCR: WIE = 0) before changing to stop mode in subclock mode.

● Wake-up from stop mode

A reset and an external interrupt wakes up the CPU from stop mode.

If reset occurs during stop mode on a product, the reset operation starts after the main clock oscillation stabilization wait time. The reset initializes pin states.

If an interrupt request with an interrupt level higher than "11" occurs from an external interrupt circuit during stop mode, the CPU wakes up from stop mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1, IL0) in the CPU. Only external interrupt requests can occur during stop mode because peripheral functions are stopped.

After wake-up from stop mode, the normal interrupt operation is performed after the oscillation stabilization wait time has passed. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering stop mode.

Some peripheral functions restart from mid-operation when the CPU wakes up from stop mode by an external interrupt. The first interval time from the interval timer function, for example, is indeterminate. Therefore, initialize all peripheral functions after wake-up from stop mode.

Precaution:

Only interrupt requests from external interrupt circuits can be used to wake up from stop mode by an interrupt.

3.7.4 Watch Mode

This section describes the operations of watch mode.

■ Operation of watch mode

● Changing to watch mode

Watch mode stops the clocks that clock the CPU and the main peripheral functions. You can go to watch mode only from subclock mode (in which the main clock oscillation is stopped).

Prior to going to watch mode, registers are saved and the contents of RAM are held. All chip functions other than watch prescaler (watch prescaler interrupt), external interrupt circuit, and certain functions that run off of the subclock stop. Accordingly, data can be held with extremely small power consumption.

Writing "1" to the watch bit in the standby control register (STBC: TMD) changes the CPU to watch mode.

This can be done, however, only when the system clock select bit of the system clock control register (SYCC: SCS) is "0" (subclock mode active).

When you go to watch mode, external pin states are held if the pin state specification bit in the standby control register (STBC: SPL) is "0". If SPL is "1", external pins go the high-impedance state. (Pins with a pull-up resistor go to the "H' level)

If an interrupt request is generated when "1" is written to the TMD bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to watch mode. (The CPU does not assume watch mode even after completion of the interrupt processing.)

● Wake-up from watch mode

A reset, a watch prescaler interrupt or an external interrupt wakes up CPU from watch mode.

If a reset occurs during watch mode on a product, the reset operation starts after the main clock oscillation stabilization wait time.

The reset initializes pin states.

If an interrupt request with an interrupt level higher than "11" occurs from a watch prescaler or an external interrupt circuit during watch mode, the CPU wakes up from watch mode, regardless of the interrupt enable flag (CCR:I) and interrupt level bits ((CCR: IL1, IL0) in the CPU. Only watch prescaler or external interrupt requests can occur during watch mode because most of the peripheral functions except watch prescaler are stopped.

After wake-up from stop mode, the normal interrupt operation is performed. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering watch mode.

Some peripheral functions restart from mid-operation when the CPU wakes up from watch mode. The first interval time from the interval timer function, for example, is indeterminate. Therefore,initialize all peripheral functions after wake-up from watch mode.

3.7.5 Standby Control Register (STBC)

The standby control register (STBC) controls the changing to sleep mode, stop mode, sets the pin states in stop mode, and initiates software resets.

■ Standby control register (STBC)

Figure 3.7-1 Standby control register (STBC)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0008H	STP	SLP	SPL	RST	TMD	—	—	—	00010XXXB
	R/W	R/W	R/W	W	R/W				

TMD	Watch bit	
	Valid only in subclock mode (SYCC: SCS = 0)	
	Read	Write
0	Reading always returns “0”.	No effect on operation
1	—	Changing to watch mode

RST	Software reset bit	
	Read	Write
0	—	Generates a reset signal for four instruction cycles.
1	Reading always returns “1”.	No effect on operation

SPL	Pin state specification bit	
	Read	Write
0	External pins hold their states prior to entering stop mode or watch mode.	
1	External pins go to high-impedance state on entering stop mode or watch mode.	

SLP	Sleep bit	
	Read	Write
0	Reading always returns “0”.	No effect on operation
1	—	Change to sleep mode.

STP	Stop bit	
	Read	Write
0	Reading always returns “0”.	No effect on operation
1	—	Change to stop mode.

R/W : Readable and writable
W : Write-only
— : Unused
X : Indeterminate
: Initial value

Table 3.7-2 Standby control register (STBC) bits

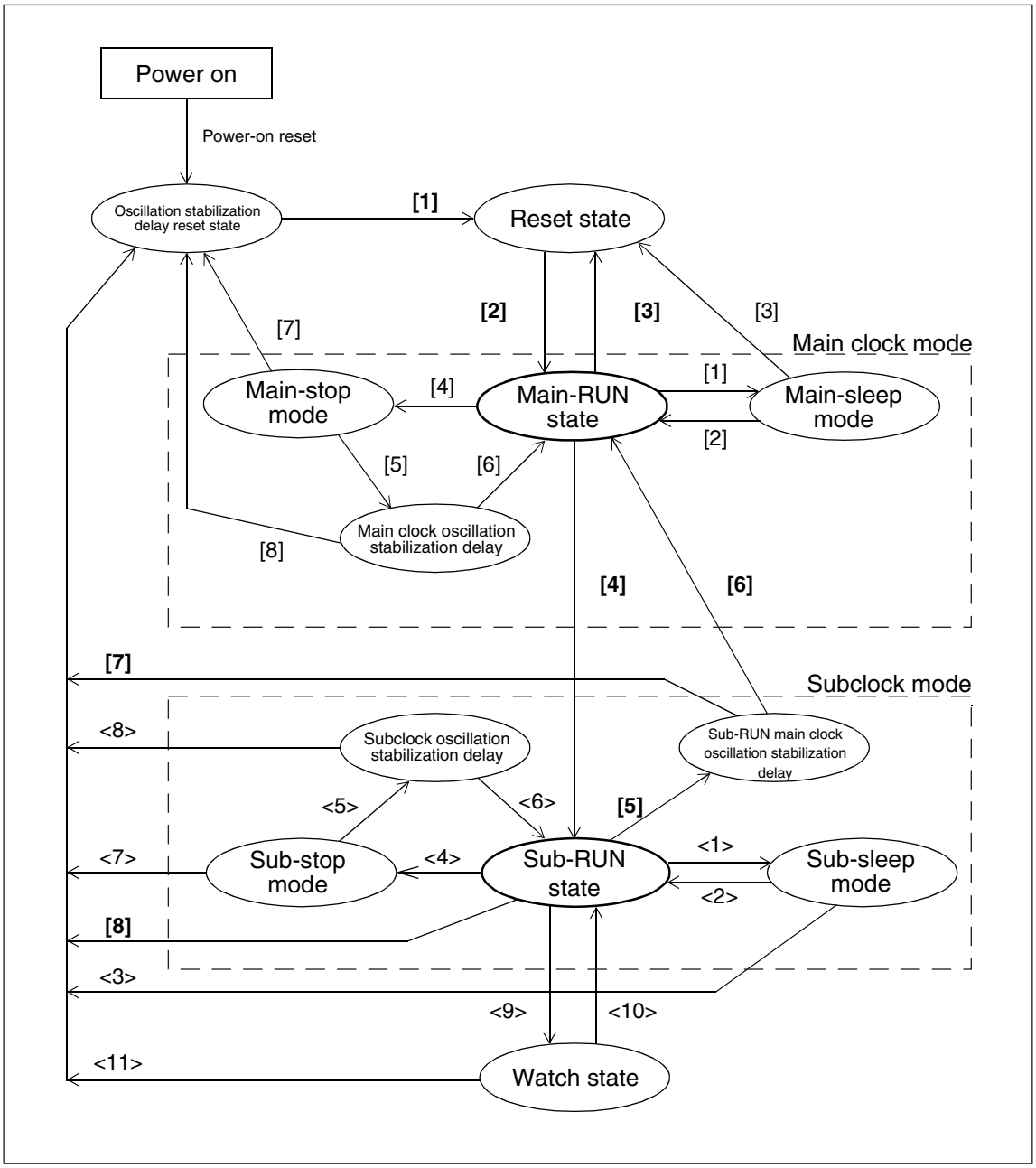
Bit		Function
Bit 7	STP: Stop bit	• Sets the CPU changing to stop mode. • Writing "1" to this bit sets the CPU changing to stop mode. • Writing "0" to this bit has no effect on operation. • Reading this bit always returns "0".
Bit 6	SLP: Sleep bit	• Sets the CPU changing to sleep mode. • Writing "1" to this bit sets the CPU changing to sleep mode. • Writing "0" to this bit has no effect on operation. • Reading this bit always returns "0".
Bit 5	SPL: Pin state specification bit	• Specifies the states of the external pins during stop mode and watch mode. • Writing "0" to this bit specifies that external pin hold their states (levels) on changing to stop mode or watch mode. • Writing "1" to this bit specifies that external pins to go to high impedance state on entering stop mode or watch mode. (Pin with a pull-up resistor go to "H" level.) • Initialized to "0" by a reset.
Bit 4	RST: Software reset bit	• Specifies a software reset. • Writing "0" to this bit generates an internal reset source for four instruction cycles. • Writing "1" to this bit has no effect on operation. • Reading this bit always returns "1".
Bit 3	TMD: Watch bit	• Sets the CPU changing to watch mode. • A write to this bit is valid only in subclock mode (SYCC: SCS = 0). • Writing "1" to this bit sets the CPU changing to watch mode. • Writing "0" to this bit has no effect on operation. • Reading this bit always returns "0".
Bit 2 Bit 1 Bit 0	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on operation.

3.7.6 State Transition Diagram 1 (Two-clock)

This section shows the state transition diagrams for two-clock option products.

■ State transition diagram 1 (Two-clock)

Figure 3.7-2 State transition diagram 1 (Two-clock)



● Changing to/wake-up from clock modes (non-standby modes)

Table 3.7-3 Changing to/wake-up from clock modes (options: two clocks)

State transition	Conditions/events required to transition
Changing to main-RUN state (normal main clock mode) after power-on reset	[1] Main clock oscillation stabilization wait time complete. (Timebase timer output) [2] Wake up from reset input.
Reset in main-RUN state	[3] Have external, software, or watchdog reset.
Changing from main-RUN state to sub-RUN state	[4] SYCC: SCS=0 *1
Changing from sub-RUN state back to main-RUN state	[5] SYCC: SCS=1 [6] Main clock oscillation stabilization wait time complete. (Can be checked by looking at SYCC: SCM) [7] Have external, software, or watchdog reset.
Reset in sub-RUN state	[8] Have external, software, or watchdog reset.

SYCC: System clock control register

*1: Changing to sub-RUN state at power-on occurs after the subclock oscillation stabilization wait time complete.

● Changing to/wake-up from standby modes

Table 3.7-4 Changing to/wake-up from standby modes (options: two clocks)

State transition	Conditions/events required to transition	
	Main clock mode	Subclock mode
Changing to sleep mode	[1] STBC: SLP = 1	<1> STBC: SLP = 1
Wake-up from sleep mode	[2] Interrupt (any) [3] External reset	<2> Interrupt (any) <3> External reset
Changing to stop mode	[4] STBC: STP = 1	<4> STBC: STP = 1
Wake-up from stop mode	[5] External interrupt [6] Main clock oscillation stabilization wait time complete. (Have timebase timer output.) [7] External reset [8] External reset (during oscillation stabilization wait time)	<5> External interrupt <6> Subclock oscillation stabilization wait time ends.(Have watch prescaler output.) <7> External reset <8> External reset (during oscillation stabilization wait time)

Table 3.7-4 Changing to/wake-up from standby modes (options: two clocks)

State transition	Conditions/events required to transition	
	Main clock mode	Subclock mode
Changing to watch mode	–	<9> STBC: TMD = 1 *
Wake-up from watch mode	–	<10> External or watch interrupt <11> External reset

STBC: Standby control register
*: Changing to watch mode is possible only from sub-RUN state (SYCC: SCS = 0).

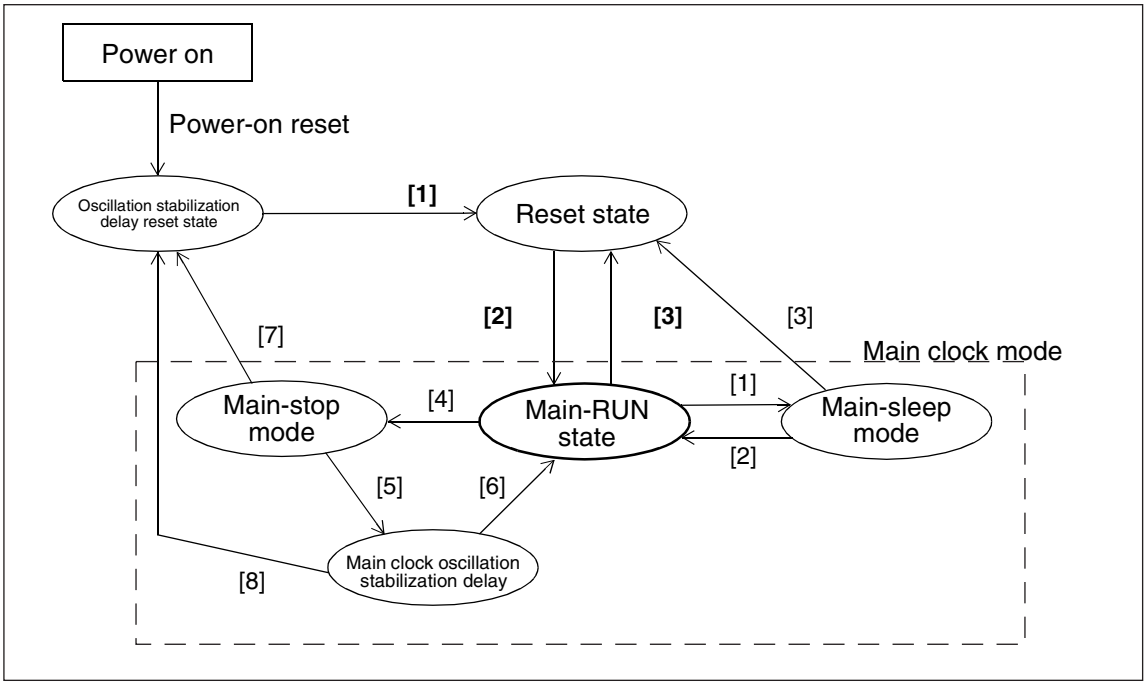
Neither software nor watchdog resets can occur during standby because the CPU and watchdog timer are both stopped.

3.7.7 State Transition Diagram 2 (One-clock Option)

This section shows the state transition diagrams for one-clock option products. There are no subclock or watch modes when one clock is used.

■ State transition diagram 2 (one-clock option)

Figure 3.7-3 State transition diagram 2



● Changing to normal state (RUN) and reset

Table 3.7-5 Changing to main clock mode run state and reset (one-clock)

State transition	Conditions/events required to transition
Changing to normal state (RUN) after power-on	[1] Main clock oscillation stabilization wait time complete. (Timebase timer output.) [2] Wake-up from Reset input.
Reset in RUN state	[3] Have external, software, or watchdog reset.

● Changing to/wake-up from standby mode

Table 3.7-6 Changing to/wake-up from standby mode (one-clock option)

State transition	Conditions/events required to transition
Changing to sleep mode	[1] STBC: SLP=1
Wake-up from sleep mode	[2] Interrupt [3] External reset
Changing to stop mode	[4] STBC: STP=1
Wake-up from stop mode	[5] External interrupt [6] Main clock oscillation stabilization wait time complete. (Timebase timer output.) [7] External reset [8] External reset (during oscillation stabilization wait time)

STBC: Standby control register

3.7.8 Notes on Using Standby Modes

The CPU does not change to a standby mode if an interrupt request occurs from a peripheral function when a standby mode is set in the standby control register. (STBC) Also, if an interrupt is used to wake up from a standby mode to the normal operating state, the operation after wake-up differs depending on whether or not the interrupt request is accepted.

■ Changing to a standby mode and interrupts

If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function to the CPU, writing "1" to the stop bit (STP),or sleep bit (SLP) in the standby control register (STBC) is ignored. Therefore, the CPU does not change to a standby mode. (The CPU also does not change to the standby mode after completing interrupt processing.)

This does not depend on whether or not the CPU accepts the interrupt.

Even if the CPU is currently performing interrupt processing, the interrupt request flag bit is cleared and, if no other interrupt request is present, the device can change to the standby mode.

■ Wake-up from standby mode by interrupt

If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function or others during sleep or stop mode, the CPU wakes up from a standby mode. This does not depend on whether or not the CPU accepts the interrupt.

After wake-up from a standby mode, the CPU performs the normal interrupt operations. If the level set in the interrupt level setting register (ILR1 to ILR4) corresponding to the interrupt request is higher than the interrupt level bits in the condition code register (CCR: IL1, IL0), and if the interrupt enable flag is enabled (CCR: I = "1"), the CPU branches to the interrupt processing routine. If the interrupt is not accepted, operation restarts from the instruction following the instruction that activated a standby mode.

To prevent control from branching to an interrupt processing routine after wake-up, take measures such as disabling interrupts before setting a standby mode.

■ Notes on setting standby mode

When setting the standby control register (STBC) to go to a standby mode, make the settings in accordance with Table 3.7-7 "Standby control register low-power consumption mode settings". The order of precedence as to which mode will be activated if more than one bit is set to "1" is "stop" mode, "watch" mode, and "sleep" mode. Other factors being equal, it is best to set "1" for just one bit.

Also avoid going to stop, sleep, or watch mode immediately after switching from subclock to main clock mode (SYCC: SCS=0 --> 1). First verify that the clock monitor bit (SYCC: SCM) of the system control register is "1," then make the standby mode change.

Note that you cannot go to the watch standby mode when operating in main clock mode. (A write to the TMD bit will be ignored.)

Table 3.7-7 Standby control register low-power consumption mode settings

STBC register			Mode
STP (Bit7)	SLP (Bit 6)	TMD (Bit 3)	
0	0	0	Normal
0	0	1	Watch
0	1	0	Sleep
1	0	0	Stop

■ Oscillation stabilization wait time

As the oscillator that provides the source oscillation is stopped during stop mode in both main clock mode and subclock mode, a wait time is required for oscillation to stabilize after the oscillator restarts operation.

In main clock mode, the main clock oscillation stabilization wait time is selected from one of four possible wait times defined by the timebase timer. In subclock mode, the subclock oscillation stabilization wait time is defined by the watch prescaler.

In main clock mode, if the interval time set for the timebase timer is less than the oscillation stabilization wait time, the timebase timer generates an interval timer interrupt request before the end of the oscillation stabilization wait time. To prevent this, disable the interrupt request output for the timebase timer (TBTC: TBIE ="0") before changing to stop mode in main clock mode as necessary.

Selection of a watch prescaler interrupt interval shorter than the oscillation stabilization wait time will similarly cause the watch interrupt request to be generated during the oscillation stabilization wait time.To prevent this, disable the watch interrupt request output for the watch prescaler (WPCR: WIE =0) before changing to stop mode in subclock mode as necessary.

3.8 Memory Access Mode

In the MB89470 series, the only memory access mode is the single-chip mode.

■ Single-chip mode

In single-chip mode, the device uses internal RAM and ROM only. Therefore, the CPU can access no areas other than the internal I/O area, RAM area, and ROM area (internal access).

■ Mode pins (MODE)

Always set the mode pins, MODE, to V_{SS}.

At reset, reads the mode data and reset vector from internal ROM.

Do not change the mode pin settings, even after completion of the reset (i.e. during normal operation).

Table 3.8-1 "Mode pin setting" lists the mode pin settings.

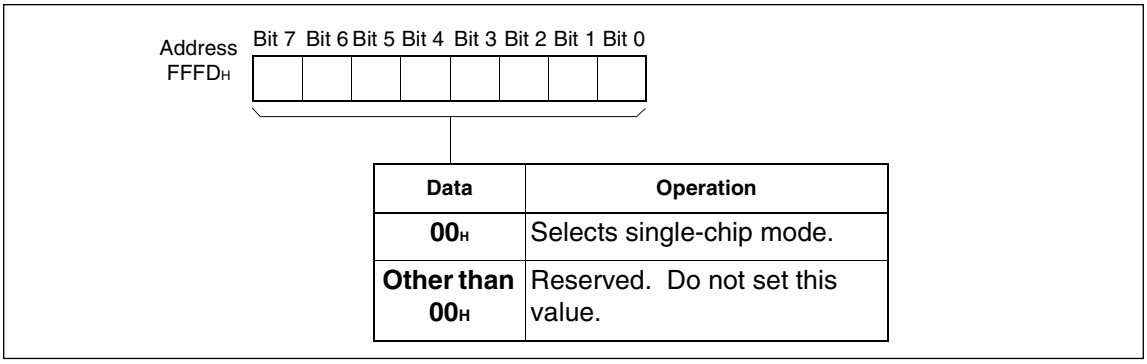
Table 3.8-1 Mode pin setting

Pin state MODE	Description
V_{SS}	Reads the mode data and reset vector from internal ROM.
V_{CC}	Prohibited settings

■ Mode data

Always set the mode data in internal ROM to "00_H" to select single-chip mode.

Figure 3.8-1 Mode data structure



■ Operation for selecting the memory access mode

Selecting mode other than single-chip mode is not possible.

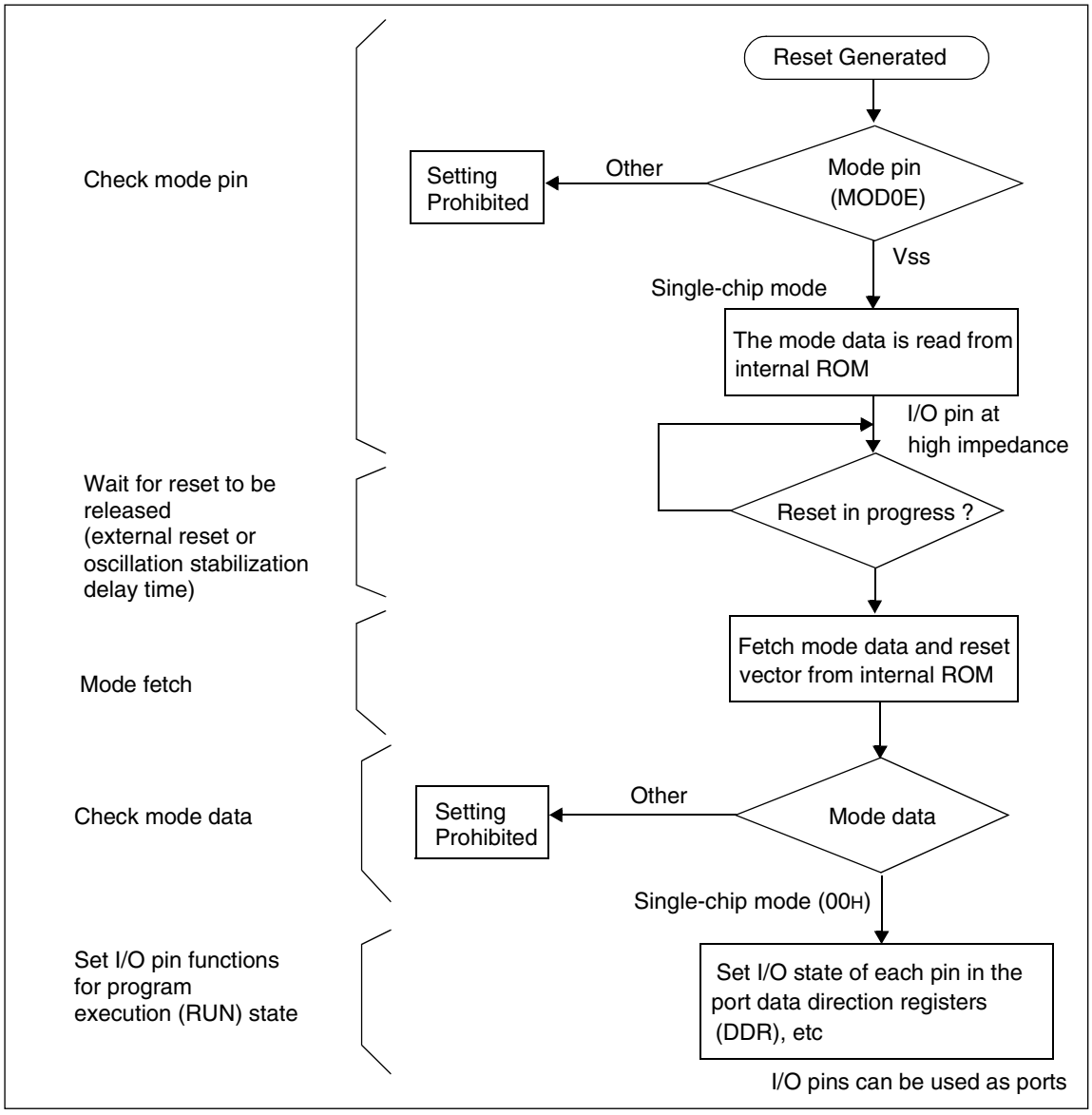
Table 3.8-2 "Mode pin and mode data" shows the mode pin and mode data.

Table 3.8-2 Mode pin and mode data

Memory Access Mode	Mode Pin(MODE)	Mode Data
Single-chip Mode	V_{SS}	00_H
Other modes	Setting prohibited	Setting prohibited

Figure 3.8-2 "Operation for selecting the memory access mode" shows the operation for selecting the memory access mode.

Figure 3.8-2 Operation for selecting the memory access mode



CHAPTER 4
I/O PORTS

This chapter describes the functions and operation of the I/O ports.

4.1 Overview of I/O Ports

The I/O ports consist of 39 pins general-purpose I/O ports.
The ports also serve as peripherals (I/O pins of peripheral functions).

■ I/O port functions

The functions of the I/O ports are to output data from the CPU via the I/O pins and to fetch signals input to the I/O pins into the CPU. Input and output are performed via the port data registers (PDR). Also, for certain ports the direction of each I/O pin can be individually set to either input or output for each bit by the port data direction register (DDR).

The following lists the functions of each port and the peripheral with which the ports also serve as.

- Port 0: General-purpose I/O port. Also serves as peripherals
  (AN0 to AN7)
- Port 1: General-purpose I/O port. Also serves as peripherals
  (INT10 to INT13, EC1, TO1, EC2, TO2)
- Port 2: General-purpose I/O port. Also serves as peripherals
  (SCK1, SO1, SI1, PWC, PWM, SI2, SO2, SCK2)
- Port 3: N-channel open-drain output port. Also serves as peripherals
  (BUZ)
- Port 4: General-purpose input port. Also serves sub-clock oscillator I/O pins
  (X0A, X1A)
- Port 5: General-purpose I/O port. Also serves as peripherals
  ($\overline{\text{INT20}}$ to $\overline{\text{INT24}}$)

Table 4.1-1 "Port function" lists the functions of each port and Table 4.1-2 "Port registers" lists the registers for each port.

Table 4.1-1 Port function

Port	Pin name	Input type	Output type	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Port0	P00/AN0 to P07/AN7	CMOS (resource analog)	CMOS push-pull	General-purpose I/O port	P07	P06	P05	P04	P03	P02	P01	P00
				Peripherals	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
Port1	P10/INT10 to P17/TO2	CMOS (resource hysteresis)	CMOS push-pull	General-purpose I/O port	P17	P16	P15	P14	P13	P12	P11	P10
				Peripherals	TO2	EC2	TO1	EC1	INT13	INT12	INT11	INT10
Port2	P20/SCK1 to P27/SCK2	CMOS (resource hysteresis)	CMOS push-pull	General-purpose I/O port	P27	P26	P25	P24	P23	P22	P21	P20
				Peripherals	SCK2	SO2	SI2	PWM	PWC	SI1	SO1	SCK1
Port3	P30/BUZ to P36	–	N-ch open-drain	General-purpose output port	–	P36	P35	P34	P33	P32	P31	P30
				Peripherals	–	–	–	–	–	–	–	BUZ
Port4	P40/X0A to P42	CMOS	–	General-purpose input port	–	–	–	–	–	P42	P41	P40
				Peripherals	–	–	–	–	–	–	X1A	X0A
Port5	P50/$\overline{\text{INT20}}$ to P54/$\overline{\text{INT24}}$	CMOS (resource hysteresis)	CMOS push-pull	General-purpose I/O port	–	–	–	P54	P53	P52	P51	P50
				Peripherals	–	–	–	$\overline{\text{INT24}}$	$\overline{\text{INT23}}$	$\overline{\text{INT22}}$	$\overline{\text{INT21}}$	$\overline{\text{INT20}}$

Table 4.1-2 Port registers

Register	Read/Write	Address	Initial value
Port 0 data register (PDR0)	R/W	0000_H	$XXXXXXXX_B$
Port 0 data direction register (DDR0)	W*	0001_H	00000000_B
Port 0 pull-up resistor control register (PURC0)	R/W	0070H	11111111_B
A/D input enable register	R/W	0024H	11111111_B
Port 1 data register (PDR1)	R/W	0002_H	$XXXXXXXX_B$
Port 1 data direction register (DDR1)	W*	0003_H	00000000_B
Port 1 pull-up resistor control register (PURC1)	R/W	0071H	11111111_B
Port 2 data register (PDR2)	R/W	0004_H	00000000_B
Port 2 data direction register (DDR2)	R/W	0006_H	00000000_B
Port 2 pull-up resistor control register (PURC2)	R/W	0072H	11111111_B
Port 3 data register (PDR3)	R/W	$000C_H$	-1111111_B
Port 3 pull-up resistor control register (PURC3)	R/W	0073H	-1111111_B
Port 4 data register (PDR4)	R	000DH	$-----XXX_B$
Port 5 data register (PDR5)	R/W	0010H	$---XXXXX_B$
Port 5 data direction register (DDR5)	R/W	0011_H	$---00000_B$
Port 5 pull-up resistor control register (PURC5)	R/W	0075H	$---11111_B$

* : Bit manipulation instructions cannot be used on DDR0, DDR1
R/W: Readable and writable
R: Read-only
W: Write-only
X: Indeterminate
–: Unused

4.2 Port 0

Port 0 is general-purpose I/O port that also serves as resource signal I/O pins. This section principally describes the port functions when operating as general-purpose I/O ports.
The section describes the port structure and pins, the pin block diagram, and the registers for port 0.

■ Structure of port 0

Port 0 consists of five components respectively.

- General-purpose I/O pins / Analog input (P00/AN0, P01/AN1, P02/AN2, P03/AN3, P04/AN4, P05/AN5, P06/AN6, P07/AN7)
- Port 0 data register (PDR0)
- Port 0 data direction register (DDR0)
- Port 0 pull-up resistor control register (PURC0)
- A/D input enable register (ADER)

■ Port 0 pins

Port 0 consists of eight I/O pins of a CMOS input and CMOS output type.

As the I/O pins are shared with resource I/O, the pins cannot be used as general-purpose I/O ports when the corresponding resource is used.

Table 4.2-1 Port 0 pins

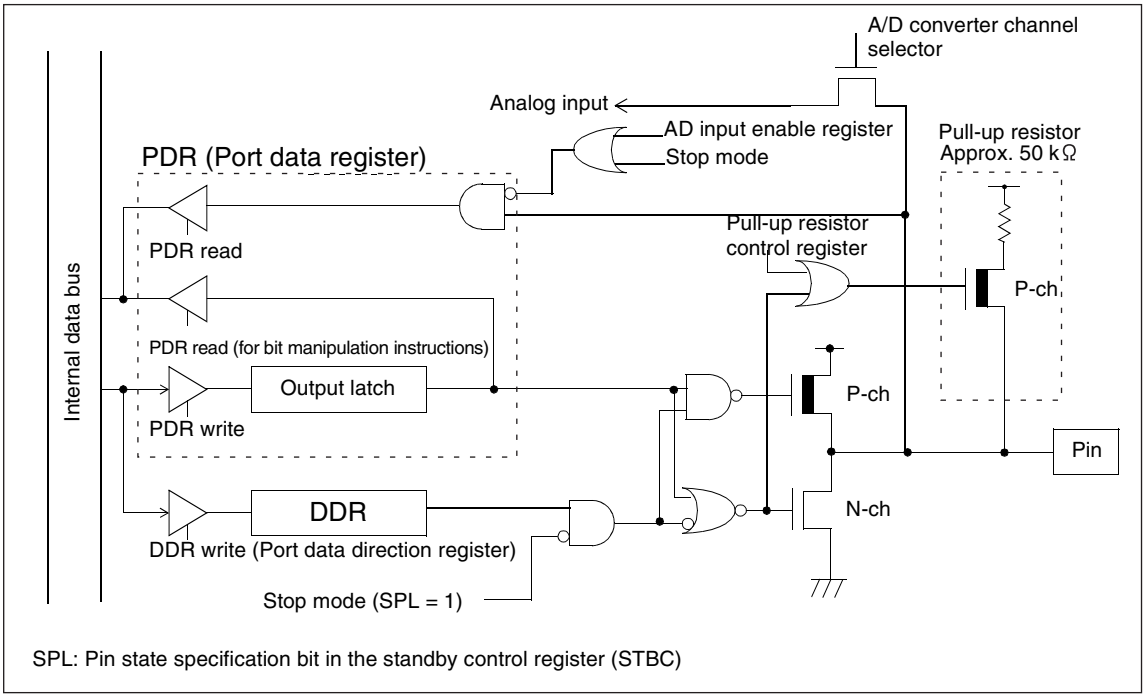
Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port0	P00/AN0 to P07/AN7	P00 to P07 General-purpose I/O	AN0 to AN7	CMOS (resource analog)	CMOS	D

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 0 pin

Figure 4.2-1 Block diagram of Port 0 pin



Precaution:

When a port is used as a normal input port, peripheral circuit operation that uses the same pins must be disabled.

■ Port 0 registers

The port 0 registers consist of PDR0, DDR0, PURC0 and ADER. Each bit in these registers has a one-to-one relationship with port 0 pin respectively.

Table 4.2-2 "Correspondence between pin and register for Port 0" shows the correspondence between pins and registers for port 0.

Table 4.2-2 Correspondence between pin and register for Port 0

Port	Correspondence between register bit and pin								
Port 0	PDR0, DDR0, PURC0, ADER	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	P07	P06	P05	P04	P03	P02	P01	P00

4.2.1 Port 0 Registers (PDR0, DDR0, PURC0, ADER)

This section describes the port 0 registers.

■ Port 0 register functions

● Port 0 data register (PDR0)

The PDR0 register hold the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Note:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Port 0 data direction register (DDR0)

The DDR0 register set the direction (input or output) for each pin (bit).

Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

Precaution:

As the DDR0 register is write-only, the bit manipulation instructions (SETB and CLRB) cannot be used.

● Settings as an analog input

When port 0 pins are used as analog signal input pins, write "0" to the corresponding bits of DDR0 to set the pin as input pin and write "1" to the corresponding bits of ADER to set the pin as analog input pin.

Table 4.2-3 "Port 0 register function" lists the functions of the port 0 register.

Table 4.2-3 Port 0 register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 0 data register (PDR0)	0	Pin state is the "L" level.	Sets "0" to the output latch. Outputs an "L" level to the pin if the pin functions as an output port.	R/W	0000_H	$XXXXXXXX_B$
	1	Pin state is the "H" level.	Sets "1" to the output latch. Outputs an "H" level to the pin if the pin functions as an output port.			
Port 0 data direction register (DDR0)	0	Reading is not permitted (write-only).	Disables output transistor and sets the pin as an input pin.	W	0001_H	00000000_B
	1		Enables output transistor and sets the pin as output pin.			

R/W: Readable and writable
W: Write-only
X: Indeterminate

● AD input enable register (ADER)

The ADER register set the funciton for each pin.

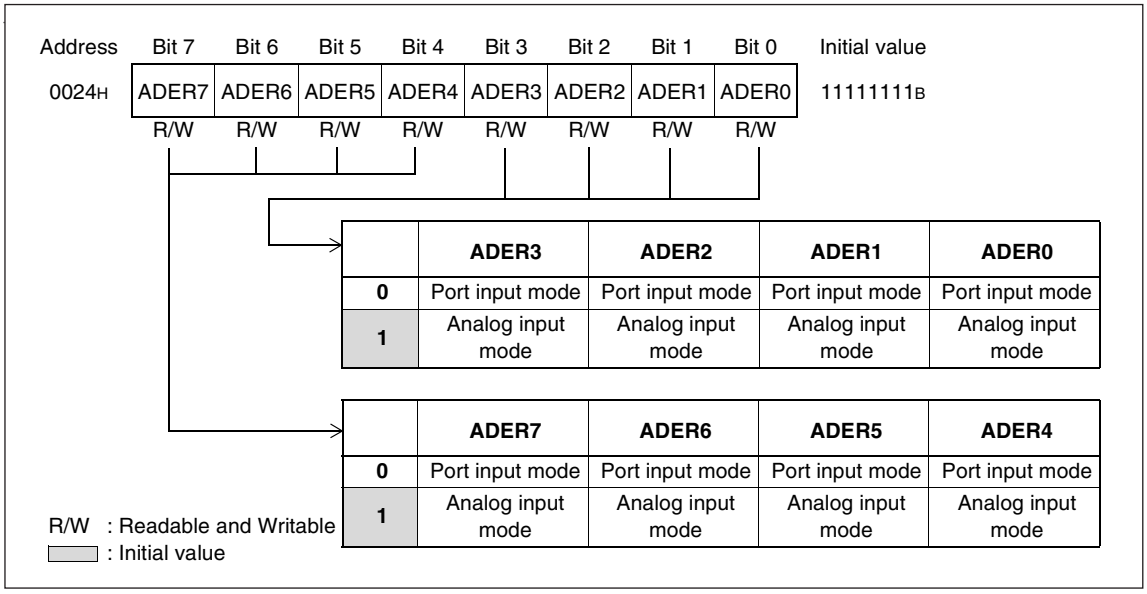
Setting "1" to the bit corresponding to a port (pin) sets the pin as an analog input pin. Setting "0" sets the pin as a general purpose I/O port.

Note:

To set the pin as analog input pin, the corresponding bitin DDR0 also should be set to "1".

Figure 4.2-2 "AD input enble register (ADER)" list the ADER setting.

Figure 4.2-2 AD input enble register (ADER)



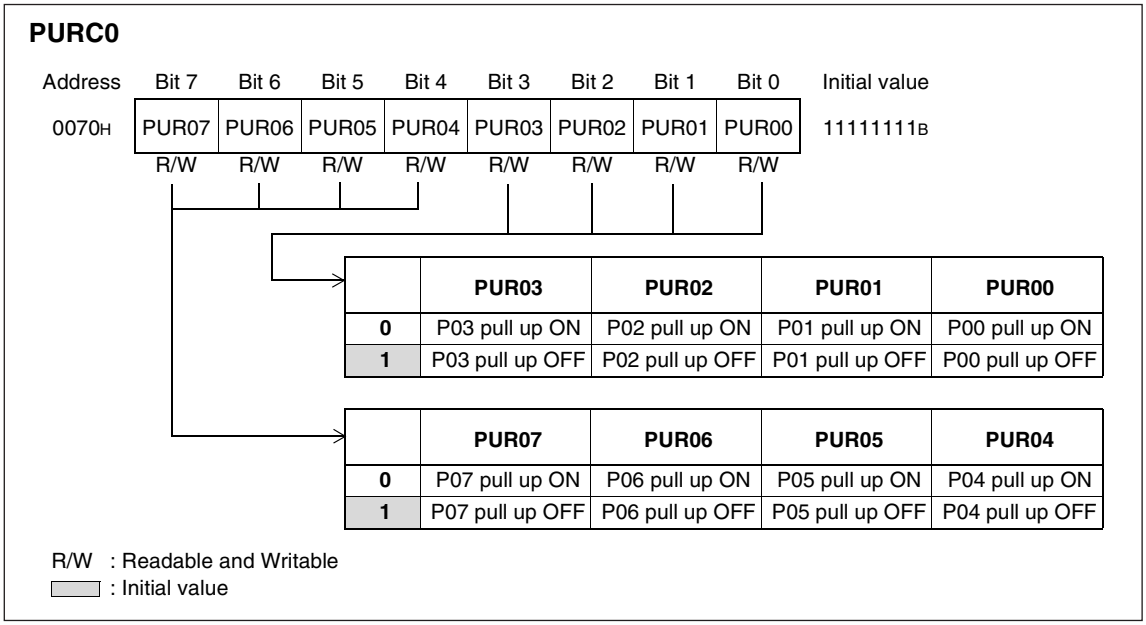
● Port 0 pull-up resistor control register (PURC0)

By using pull-up resistor control register (PURC0), it is possible to turn on/off pull-up resistor bit .

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1). The states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

Figure 4.2-3 "Pull-up resistor control register setting (PURC0)" lists the function of Port 0 pull-up contorl resistor.

Figure 4.2-3 Pull-up resistor control register setting (PURC0)



4.2.2 Operation of Port 0

This section describes the operations of the port 0.

■ Operation of Port 0

● Operation as an output port

- Setting the corresponding DDR0 register bit to "1" sets a pin as an output port.
- When a pin is set as an output port, its output transistor is enabled and the pin outputs the data stored in the output latch.
- Writing data to the PDR0 registers stores the data in the output latch and outputs the data directly to the pin.
- Reading the PDR0 register returns the pin value.

● Operation as an input port

- Setting the corresponding DDR0 and ADER register bit to "0" sets a pin as an input port.
- When a pin is set as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Writing data to the PDR0 registers stores the data in the output latch but does not output the data to the pin.
- Reading the PDR0 register returns the pin value.

● Operation as an A/D analog input

- Set the DDR0 bit that corresponds to the analog input pin to "0" to turn the port to input port.
- Set the ADER corresponding bit to "1" to set the port as analog input port.

● Operation at reset

- Resetting the CPU initializes the DDR0 register values to "0". This sets the output transistors "OFF" (all pins become input ports) and sets the pins to the high-impedance state.
- The PDR0 register is not initialized by a reset. Therefore, to use as output ports, the output data must be set in the PDR0 registers before setting the corresponding DDR0 register bits to output mode.
- The ADER register values are initialized to "1" by a reset. Therefore, to use as general/purpose I/O ports, the corresponding bit in ADER should be cleared to "0".

● Operation in stop mode

The pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the DDR0 register values.

Table 4.2-4 "Port 0 pin state" lists the port 0 pin states.

Table 4.2-4 Port 0 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Main-stop mode (SPL = 1) sub-stop mode (SPL = 1) watch mode (SPL = 1)	Reset
P00 to P07	General-purpose I/O ports	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Note:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistors are all "OFF".

4.3 Port 1

Port 1 is a general-purpose I/O port that also serves as resource signal I/O pins. This section principally describes the port functions when operating as general-purpose I/O ports.
Also the section describes the port structure and pins, the pin block diagram, and the registers for port 1.

■ Structure of Port 1

Port 1 consists of four components respectively.

- General-purpose I/O pins / External interrupt (P10/INT10, P11/INT11, P12/INT12, P13/INT13), general-purpose I/O pins / 8/16 bit timer input (P14/EC1, P16/EC2), general-purpose I/O pins / 8/16 bit timer output (P15/TO1, P17/TO2)
- Port 1 data register (PDR1)
- Port 1 data direction register (DDR1)
- Port 1 pull-up resistor contorl register (PURC1)

■ Port 1 pins

Port 1 consists of eight I/O pins of a CMOS input and CMOS output type.

As the I/O pins are shared with resource I/O, the pins cannot be used as general-purpose I/O ports when the corresponding resource is used

Table 4.3-1 Port 1 pins

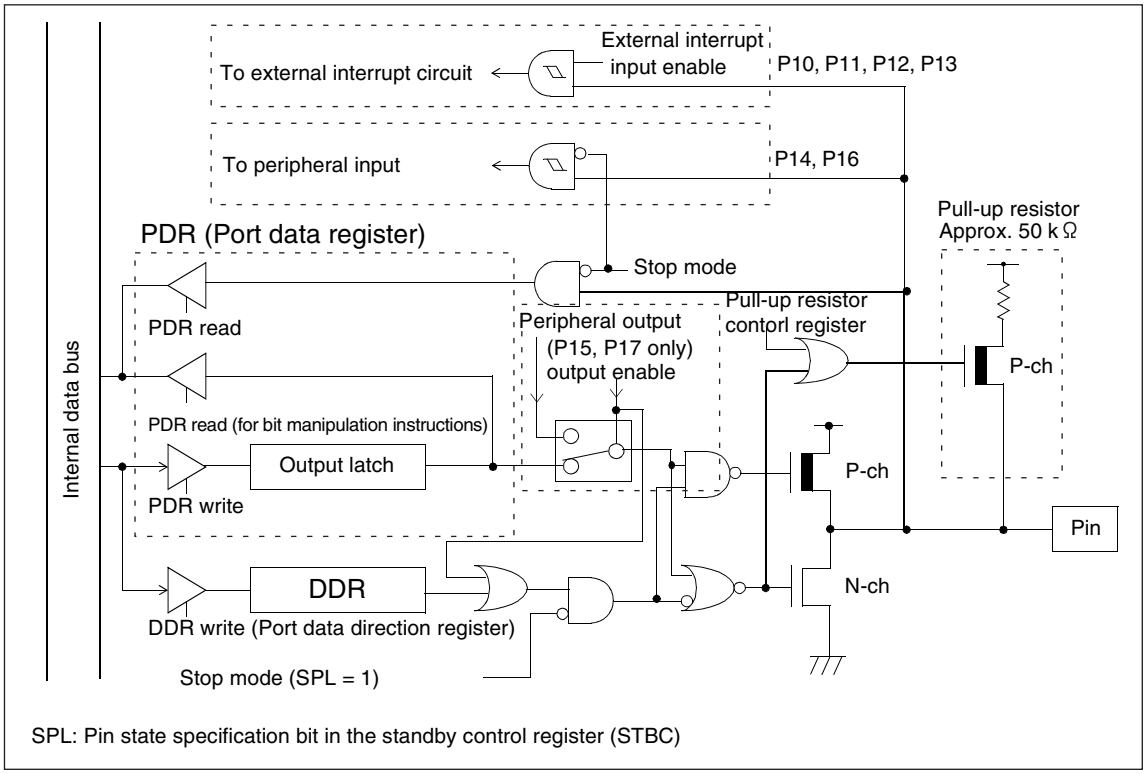
Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port1	P10/INT10 to P13/ INT13, P14/EC1, P16/ EC2	P10 - P13, P14, P16 General purpose I/O	INT10 to INT13, EC1, EC2	CMOS (resource hysteresis)	CMOS	E
	P15/TO1, P17/TO2	P15, P17 General purpose I/O	TO1, TO2	CMOS	CMOS	F

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 1 pin

Figure 4.3-1 Block diagram of Port 1 pin



Precaution:

When a port is used as a normal input port, peripheral circuit operation that uses the same pins must be disabled.

■ Port 1 registers

The port 1 registers consist of PDR1, DDR1 and PURC1. Each bit in these registers has a one-to-one relationship with port 1 pin respectively.

Table 4.3-2 "Correspondence between pin and register for Port 0" shows the correspondence between pins and registers for port 1.

Table 4.3-2 Correspondence between pin and register for Port 0

Port	Correspondence between register bit and pin								
Port 1	PDR1, DDR1, PURC1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	P17	P16	P15	P14	P13	P12	P11	P10

4.3.1 Port 1 Registers (PDR1, DDR1, PURC1)

This section describes the port 1 registers.

■ Port 1 register functions

● Port 1 data register (PDR1)

The PDR1 register hold the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Note:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Port 1 data direction register (DDR1)

The DDR1 register set the direction (input or output) for each pin (bit).

Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

Precaution:

As the DDR1 register is write-only, the bit manipulation instructions (SETB and CLRB) cannot be used.

● Settings as peripheral output

To use a peripheral that has an output pin, set the peripheral output enable bit for that pin to the "enable" state. As can be seen in the block diagram, the peripheral has precedence over the general-purpose port for use of the output pin. Once the peripheral output is enabled, the states set in the PDR1 and DDR1 registers are no longer valid, and do not affect the data output by the peripheral, or the enabling of the output.

● Settings as a peripheral input

To use a peripheral that has a Port 1 pin as an input pin, set that pin as an input port. The output latch data for that pin will no longer be valid.

Table 4.3-3 "Port 1 register function" lists the functions of the port 1 register.

Table 4.3-3 Port 1 register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 1 data register (PDR1)	0	Pin state is the "L" level.	Sets "0" to the output latch. Outputs an "L" level to the pin if the pin functions as an output port.	R/W	0002_H	$XXXXXXXX_B$
	1	Pin state is the "H" level.	Sets "1" to the output latch. Outputs an "H" level to the pin if the pin functions as an output port.			
Port 1 data direction register (DDR1)	0	Reading is not permitted (write-only).	Disables output transistor and sets the pin as an input pin.	W	0003_H	00000000_B
	1		Enables output transistor and sets the pin as output pin.			

R/W: Readable and writable
W: Write-only
X: Indeterminate

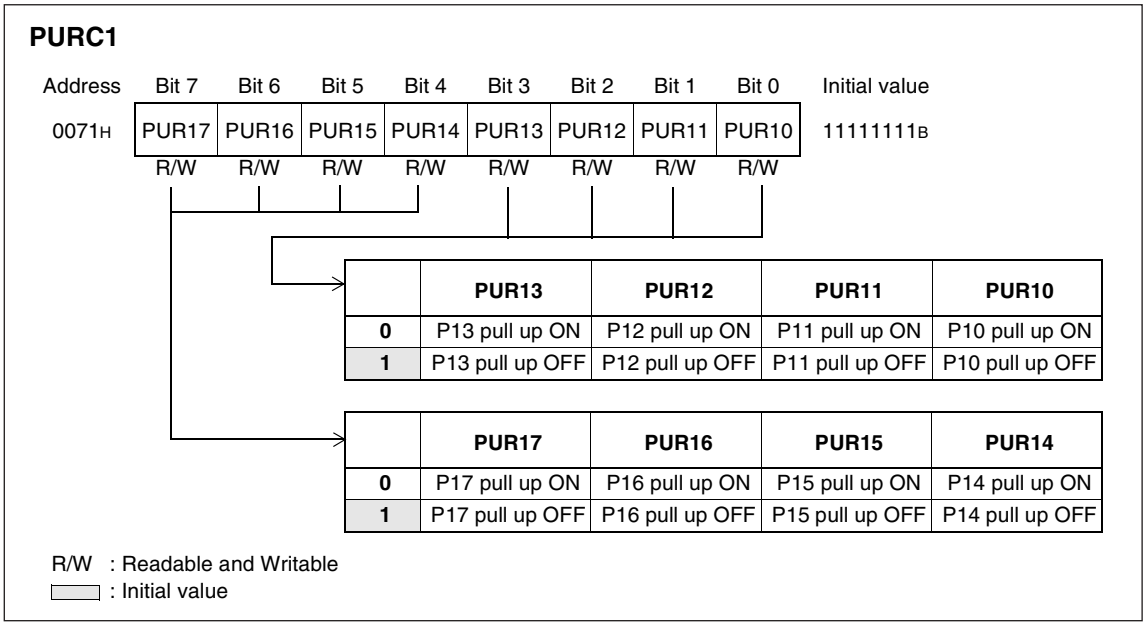
● Port 1 pull-up resister control register (PURC1)

By using pull-up resistor control register (PURC1), it is possible to turn on/off pull-up resistor bit .

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1). The states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

Figure 4.3-2 "Pull-up resistor control register setting (PURC1)" lists the function of Port 1 pull-up contorl resistor.

Figure 4.3-2 Pull-up resistor control register setting (PURC1)



4.3.2 Operation of Port 1

This section describes the operations of the Port 1.

■ Operation of Port 1

● Operation as an output port

- Setting the corresponding DDR1 register bit to "1" sets a pin as an output port.
- When a pin is set as an output port, its output transistor is enabled and the pin outputs the data stored in the output latch.
- Writing data to the PDR1 registers stores the data in the output latch and outputs the data directly to the pin.
- Reading the PDR1 register returns the pin value.

● Operation as an input port

- Setting the corresponding DDR1 register bit to "0" sets a pin as an input port.
- When a pin is set as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Writing data to the PDR1 registers stores the data in the output latch but does not output the data to the pin.
- Reading the PDR1 register returns the pin value.

● Operation as a peripheral output

- If a peripheral output enable bit is set to "enable", the corresponding pin becomes a peripheral output.
- As the pin value can be read even if the peripheral output is enabled, the peripheral output value can be read via the PDR1 register.

● Operation as a peripheral input

- A port pin is set as a peripheral input by setting the corresponding DDR1 register bit to "0".
- Reading the PDR1 register returns the pin value, regardless of whether or not the peripheral is using the input pin.

● Operation at reset

- Resetting the CPU initializes the DDR1 register values to "0". This sets the output transistors "OFF" (all pins become input ports) and sets the pins to the high-impedance state.
- The PDR1 registers are not initialized by a reset. Therefore, to use as output ports, the output data must be set in the PDR1 registers before setting the corresponding DDR1 register bits to output mode.

● Operation in stop mode

The pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the DDR1 register values.

Table 4.3-4 "Port 1 pin state" lists the port 1 pin states.

Table 4.3-4 Port 1 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Main-stop mode (SPL = 1) sub-stop mode (SPL = 1) watch mode (SPL = 1)	Reset
P10 to P17	General-purpose I/O ports	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Note:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistors are all "OFF".

4.4 Port 2

Port 2 is general-purpose I/O ports that also serve as resource signal I/O pins. This section principally describes the port functions when operating as general-purpose I/O ports.
The section describes the port structure and pins, the pin block diagram, and the registers for port 2.

■ Structure of Port 2

Port 2 consist of four components.

- General-purpose I/O pin / UART/SIO 1 I/O (P20/SCK1, P21/SO1, P22/SI1)), general-urpose I/O pin / PWC input (P23/PWC), general-purpose I/O pin / PWM output (P24/PWM), general-purpose I/O pin / UART/SIO 2 I/O (P25/SI2, P26/SO2, P27/SCK2).
- Port 2 data register (PDR2)
- Port 2 data direction register (DDR2)
- Port 2 pull-up resistor control register (PURC2)

■ Port 2 pins

Port 2 consists of eight I/O pins of a CMOS input and CMOS output type.

As the I/O pins are shared with resource I/O, the pins cannot be used as general-purpose I/O ports when the corresponding resource is used.

Table 4.4-1 Port 2 pins

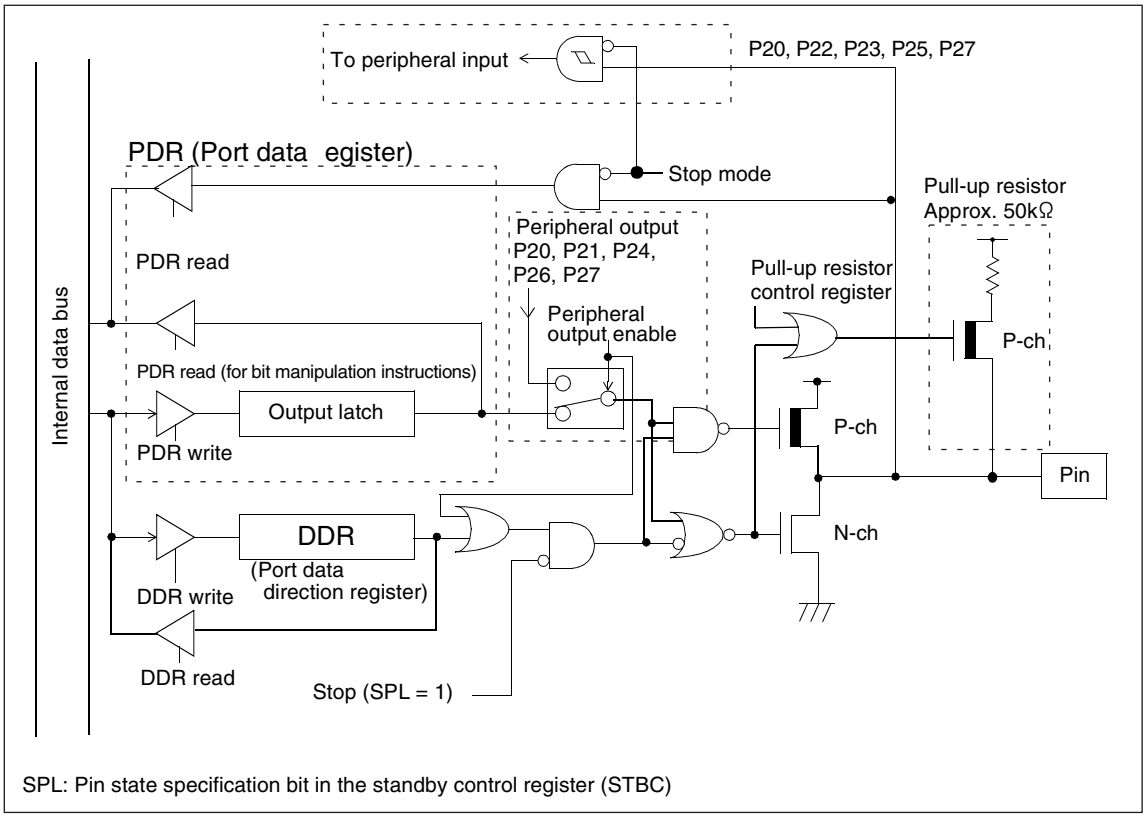
Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 2	P20/SCK1, P22/SI1, P23/PWC, P25/SI2, P27/SCK2	P20, P22, P23, P25, P27 General-purpose I/O	SCK1, SI1, PWC, SI2, SCK2	CMOS (resource hysteresis)	CMOS	E
	P21/SO1, P24/PWM, P26/SO2	P21, P24, P26 General-purpose I/O	SO1, PWM, SO2	CMOS	CMOS	F

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 2 pin

Figure 4.4-1 Block diagram of Port 2 pin



Precaution:

When a port is used as a normal input port, peripheral circuit operation that uses the same pins must be disabled.

■ Port 2 registers

The port 2 registers consist of PDR2, DDR2 and PURC2. Each bit in these registers has a one-to-one relationship with port 2 pin. Table 4.4-2 "Correspondence between pin and register for Port 2" shows the correspondence between pins and registers for port 2.

Table 4.4-2 Correspondence between pin and register for Port 2

Port	Correspondence between register bit and pin								
Port2	PDR2, DDR2, PURC2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	P27	P26	P25	P24	P23	P22	P21	P20

4.4.1 Port 2 Registers (PDR2, DDR2, PURC2)

This section describes the port 2 registers.

■ Port 2 register functions

● Port 2 data register (PDR2)

The PDR2 register holds the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Note:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Port 2 data direction register (DDR2)

The DDR2 register sets the direction (input or output) for each pin (bit).

Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

● Settings as peripheral output

To use a peripheral that has an output pin, set the peripheral output enable bit for that pin to the "enable" state. As can be seen in the block diagram, the peripheral has precedence over the general-purpose port for use of the output pin. Once the peripheral output is enabled, the states set in the PDR2 and DDR2 registers are no longer valid, and do not affect the data output by the peripheral, or the enabling of the output.

● Settings as a peripheral input

To use a peripheral that has a Port 2 pin as an input pin, set that pin as an input port. The output latch data for that pin will no longer be valid.

Table 4.4-3 "Port 2 register function" lists the functions of port 2 registers.

Table 4.4-3 Port 2 register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 2 data register (PDR2)	0	Pin state is the "L" level.	Sets "0" to the output latch. Outputs an "L" level to the pin if the pin functions as an output port.	R/W	0004_H	00000000_B
	1	Pin state is the "H" level.	Sets "1" to the output latch. Outputs an "H" level to the pin if the pin functions as an output port.			
Port 2 data direction register (DDR2)	0	Input port	Disables output transistor and sets the pin as an input pin.	R/W	0006_H	00000000_B
	1	Output port	Enables output transistor and sets the pin as output pin.			

R/W: Readable and writable

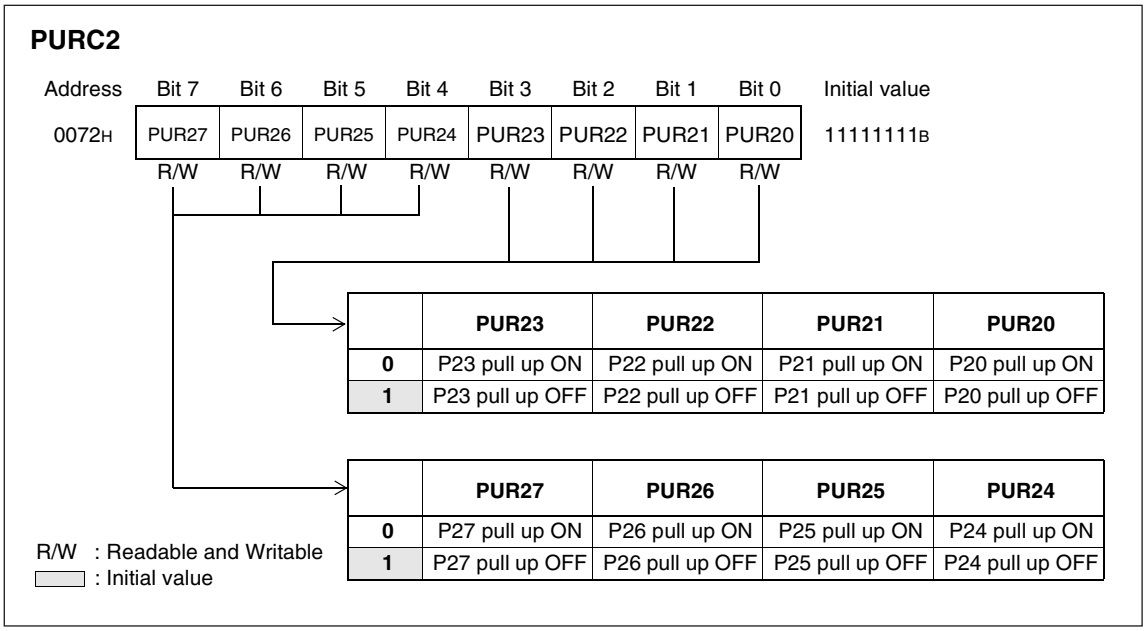
● Port 2 pull-up resistor control register (PURC2)

By using pull-up resistor control register (PURC2), it is possible to turn on/off pull-up resistor bit .

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1). The states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

Figure 4.4-2 "Pull-up resistor control register setting (PURC2)" lists the function of Port 2 pull-up contorl resistor.

Figure 4.4-2 Pull-up resistor control register setting (PURC2)



	PUR23	PUR22	PUR21	PUR20
0	P23 pull up ON	P22 pull up ON	P21 pull up ON	P20 pull up ON
1	P23 pull up OFF	P22 pull up OFF	P21 pull up OFF	P20 pull up OFF

	PUR27	PUR26	PUR25	PUR24
0	P27 pull up ON	P26 pull up ON	P25 pull up ON	P24 pull up ON
1	P27 pull up OFF	P26 pull up OFF	P25 pull up OFF	P24 pull up OFF

4.4.2 Operation of Port 2

This section describes the operations of the port 2.

■ Operation of Port 2

● Operation as an output port

- Setting the corresponding DDR2 register bit to "1" sets a pin as an output port.
- When a pin is set as an output port, its output transistor is enabled and the pin outputs the data stored in the output latch.
- Writing data to the PDR2 registers stores the data in the output latch and outputs the data directly to the pin.
- Reading the PDR2 register returns the pin value.

● Operation as an input port

- Setting the corresponding DDR2 register bit to "0" sets a pin as an input port.
- When a pin is set as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Writing data to the PDR2 registers stores the data in the output latch but does not output the data to the pin.
- Reading the PDR2 register returns the pin value.

● Operation as a peripheral output

- If a peripheral output enable bit is set to "enable," the corresponding pin becomes a peripheral output.
- As the pin value can be read even if the peripheral output is enabled, the peripheral output value can be read via the PDR2 register.

● Operation as a peripheral input

- A port pin is set as a peripheral input by setting the corresponding DDR2 register bit to "0".
- Reading the PDR2 register returns the pin value, regardless of whether or not the peripheral is using the input pin.

● Operation at reset

- Resetting the CPU initializes the DDR2 register values to "0". This sets the output transistors "OFF" (all pins become input ports) and sets the pins to high-impedance state.

● Operation in stop mode

The pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the DDR2 register values.

Table 4.4-4 "Port 2 pin state" lists the port 2 pin states.

Table 4.4-4 Port 2 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Main-stop mode (SPL = 1) sub-stop mode (SPL = 1) watch mode (SPL = 1)	Reset
P20 to P27	General-purpose I/O ports/ peripheral I/O	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Note:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistors are all "OFF".

4.5 Port 3

Port 3 is a general-purpose output port that also serves as the resource signal output pin. This section principally describes the port functions when operating as a general-purpose output port.
The section describes the port structure and pins, the pin block diagram, and the port registers for port 3.

■ Structure of Port 3

Port 3 consists of the following three components:

- General-purpose output port / buzzer output pin (P30/BUZ, P31-P36)
- Port 3 data register (PDR3)
- Port 3 pull-up resister control register (PURC3)

■ Port 3 pins

Port 3 consists of seven N-channel open-drain output pins. P30 is also used as signal output pin for buzzer. While it is being used by buzzer this pin cannot be used as the general-purpose output port.

Table 4.5-1 "Port 3 pins" lists the port 3 pins.

Table 4.5-1 Port 3 pins

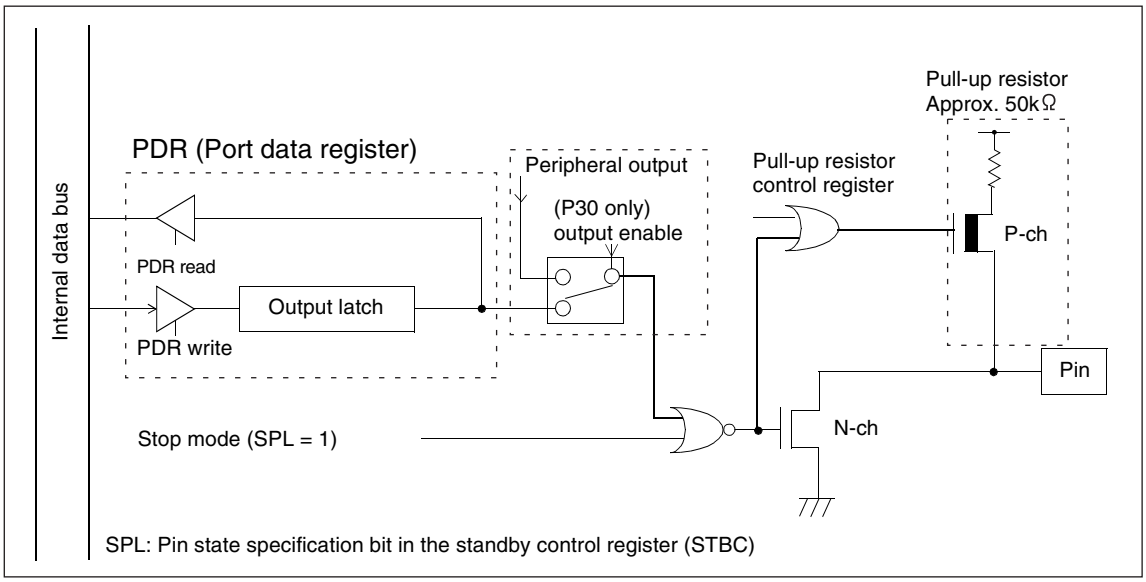
Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 3	P30/BUZ	General-purpose output	Buzzer output	–	N-ch open-drain	G
	P31 - P36		–			

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 3 pin

Figure 4.5-1 Block diagram of Port 3 pin



■ Port 3 registers

The port 3 registers consist of PDR3 and PURC3.

Each bit in these registers has a one-to-one relationship with a port 3 bit and port 3 pin.

Table 4.5-2 "Correspondence between pin and register for Port 3" shows the correspondence between pins and registers for port 3.

Table 4.5-2 Correspondence between pin and register for Port 3

Port	Correspondence between register bit and pin								
Port 3	PDR3, PURC3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	–	P36	P35	P34	P33	P32	P31	P30

4.5.1 Port 3 Registers (PDR3, PURC3)

This section describes the port 3 registers.

■ Port 3 register functions

● Port 3 data register (PDR3)

The PDR3 register holds the pin states. Therefore, when used as an output port that is not a peripheral output, it reads out as the same state ("0" or "1") as that of the output data latch; and when it is a resource output port, the output latch state is read out instead of the pin state.

● Settings as a peripheral output

To use a peripheral that has an output pin, set the peripheral output enable bit for that pin to the "enable" state. As can be seen in the block diagram, the peripheral has precedence over the general-purpose port for use of the output pin. Once the peripheral output is enabled, the states set in the PDR3 register are no longer valid, and do not affect the data output by the peripheral, or the enabling of the output.

Table 4.5-3 "Port 3 register function" lists the functions of the port 3 registers.

Table 4.5-3 Port 3 register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 3 data register (PDR3)	0	Output latch value is “0”	Sets "0" to the output latch. Outputs an "L" level to the pin.	R/W	$000C_H$	-1111111_B
	1	Output latch value is “1”	Sets "1" to the output latch. The pin is set to high impedence. *			

R/W: Readable and writable
–: Unused

*: Pins with pull-up resistor selected go to the pulled-up state.

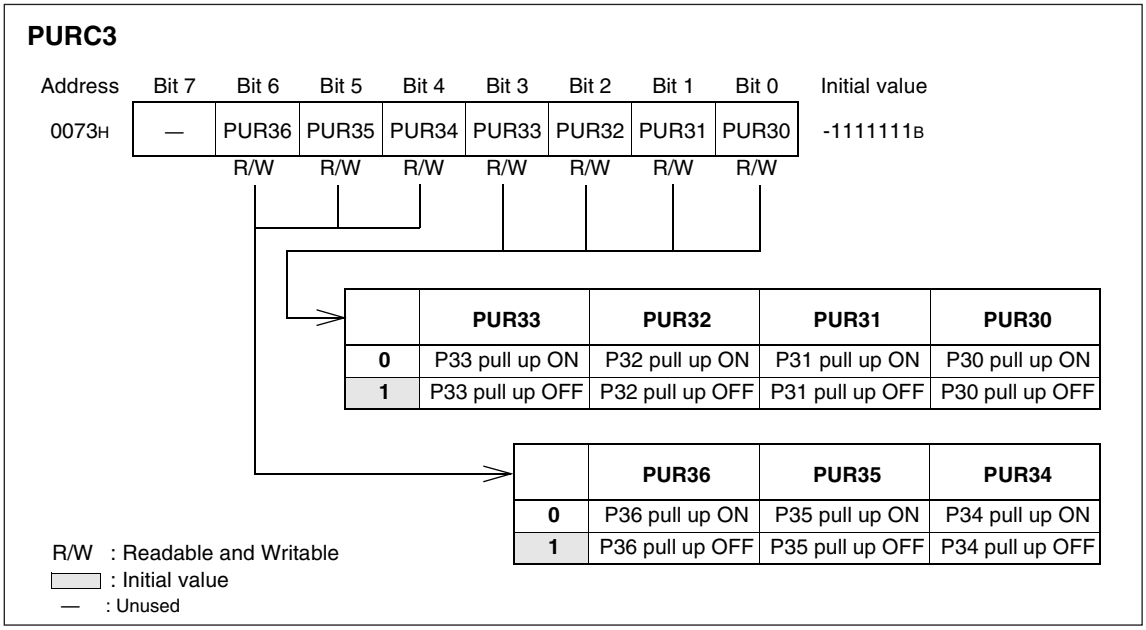
● Port 3 pull-up resistor control register (PURC3)

By using pull-up resistor control register (PURC3), it is possible to turn on/off pull-up resistor bit .

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1). The states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state

Figure 4.5-2 "Pull-up resistor register setting (PURC3)" lists the function of Port 3 pull-up contorl resistor.

Figure 4.5-2 Pull-up resistor register setting (PURC3)



4.5.2 Operation of Port 3

This section describes the operations of the port 3.

■ Operation of Port 3

● Operation as an output port

- Writing data to the PDR3 register stores the data in the output latch. When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and the pin goes to the high-impedance state. If a pull-up is set to the output pin, the pin goes to the pull-up state when the output latch value is "1".
- Reading the PDR3 register returns output latch value.

● Operation as a peripheral output

- If a peripheral output enable bit is set to "enable," the corresponding pin becomes a peripheral output.
- As the pin value cannot be read as port 3 is output port, the peripheral output value cannot be read via the PDR3 register.

● Operation at reset

- Resetting the CPU initializes the PDR3 register value to "1". This sets output transistors "OFF" and sets the pins to the high-impedance state.

● Operation in stop mode

- The pins go to the high-impedance state, if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the PDR3 register value.

Table 4.5-4 "Port 3 pin state" lists the port 3 pin states.

Table 4.5-4 Port 3 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Main-stop mode (SPL = 1) sub-stop mode (SPL = 1) watch mode (SPL = 1)	Reset
P30/BUZ, P31 to P36	General-purpose outpout port/ peripheral output	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Note:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

4.6 Port 4

Port 4 is an general purpose input ports that also serve as connection pins for a crystal or other oscillator in dual clock product. This section principally describes the port functions when operating as general-purpose input ports.
The section describes the port structure and pins, the pin block diagram, and the registers for port 4.

■ Structure of Port 4

Port 4 consists of two components.

- General-purpose input pins/connection pins for crystal or other oscillator (P40/X0A, P41/X1A), general-purpose input pin (P42)
- Port 4 data register (PDR4)

■ Port 4 pins

Port 4 consists of three input pins of CMOS input type. When P40 , P41 are used as external clock connection, they cannot be used as input pins.

Table 4.6-1 Port 4 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 4	P40/X0A, P41/X1A, P42	P40/X0A, P41/X1A external clock conneciton	X0A, X1A	CMOS	–	A
		P42 General purpose input ports	–	CMOS	–	H

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 4 pin

Figure 4.6-1 Block diagram of Port 4 pin (except P41/X1A)

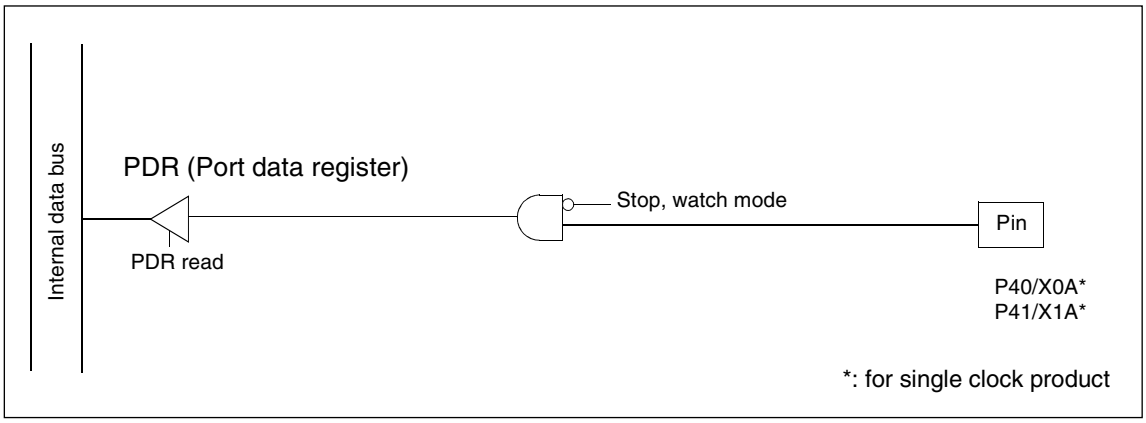
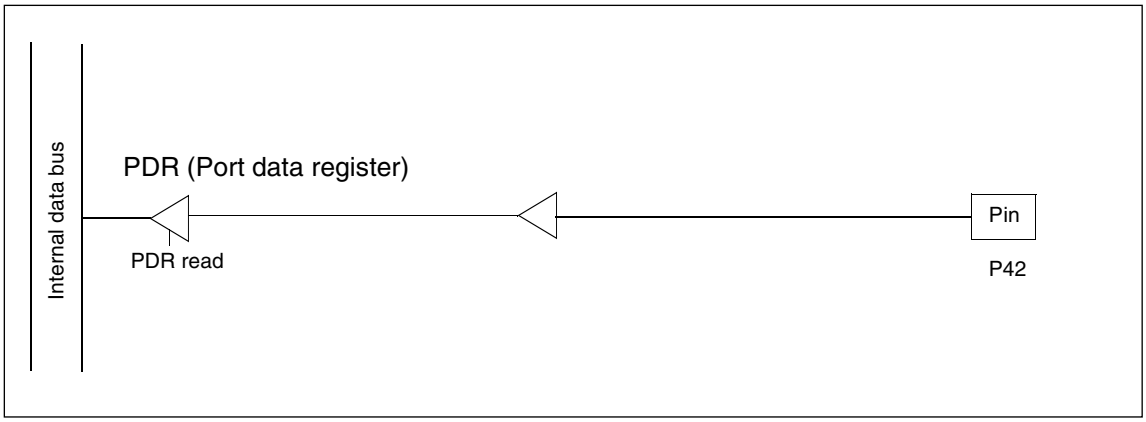


Figure 4.6-2 Block diagram of Port 4 pin (except P41/X1A)



Note:

Input buffer is not fixed in stop and watch modes. Please make sure that intermadiate potential is not input.

■ Port 4 register

The port 4 register consist of data register PDR4. Each bit in these register has a one-to-one relationship with port 4.

Table 4.6-2 "Correspondence between pin and register for Port 4" shows the correspondence between pins and registers for port 4.

Table 4.6-2 Correspondence between pin and register for Port 4

Port	Correspondence between register bit and pin								
Port4	PDR4	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	–	–	–	–	–	P42	P41	P40

4.6.1 Port 4 Register (PDR4)

This section describes the port 4 register.

■ Port 4 register functions

● Port 4 data register (PDR4)

The PDR4 register holds the pin states. It is an input port and cannot be read as the output latch state.

Table 4.6-3 "Port 4 register function" lists the functions of the port 4 register.

Table 4.6-3 Port 4 register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 4 data register (PDR4)	0	Pin state is the "L" level	–	R	$000D_H$	$-----XXX_B$
	1	Pin state is the "H" level	–			

R: Read only
X: Underterminate
–: Unused

4.6.2 Operation of Port 4

This section describes the operations of the port 4.

■ Operation of Port 4

● Operation as an input port

- Reading the PDR4 register returns the pin value.

● Operation at reset

- The PDR4 register is not initilized by a reset.

Table 4.6-4 "Port 4 pin state" lists the port 4 pin states.

Table 4.6-4 Port 4 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Reset
P40/X0A, P41/X1A, P42	General-purpose I/O ports	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

4.7 Port 5

Port 5 is a general-purpose I/O port that also serve as resource signal I/O pins. This section principally describes the port functions when operate as general-purpose I/O ports.
The section describes the port structure and pins, the pin block diagram, and the registers for port 5.

■ Structure of Port 5

Port 5 consist of three components.

- General-purpose I/O pins / external interrupt 2 input (P50/$\overline{\text{INT20}}$ - P54/$\overline{\text{INT24}}$)
- Port 5 data register (PDR5)
- Port 5 pull-up resistor control register (PURC5)

■ Port 5 pins

Port 5 consists of five I/O pins of a CMOS I/O type.

Table 4.7-1 Port 5 pins

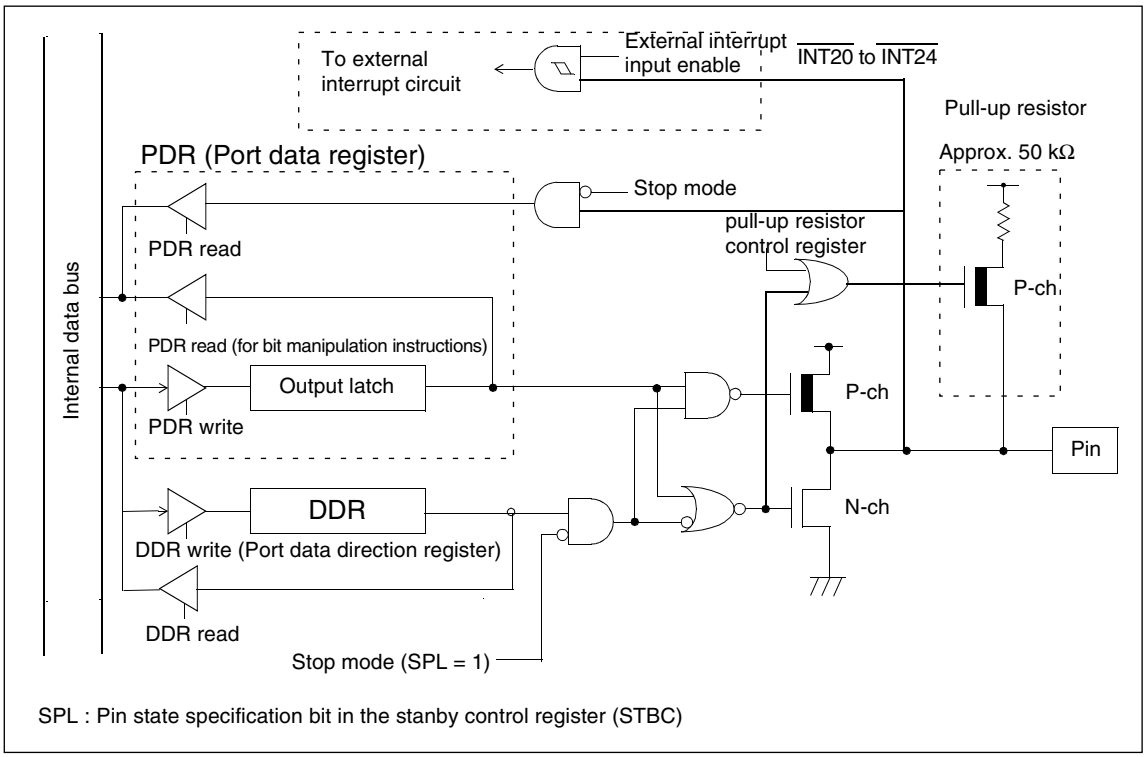
Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 5	P50/$\overline{\text{INT20}}$ to P54/$\overline{\text{INT24}}$	P50 to P54 General-purpose I/O pin	$\overline{\text{INT20}}$ to $\overline{\text{INT24}}$	COMS (resource hysteresis)	CMOS	E

Reference:

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of Port 5 pin

Figure 4.7-1 Block diagram of Port 5 pin



The port 5 registers consist of PDR5, DDR5 and PURC5. Each bit in these register has a one-to-one relationship with port 5. Table 4.7-2 "Correspondence between pin and register for Port 5" shows the correspondence between pins and registers for port 5.

Table 4.7-2 Correspondence between pin and register for Port 5

Port	Correspondence between register bit and pin								
Port 5	PDR5, DDR5, PURC5	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	–	–	–	P54	P53	P52	P51	P50

4.7.1 Port 5 Registers (PDR5)

This section describes the port 5 register.

■ Port 5 register functions

● Port 5 data register (PDR5)

The PDR5 register hold the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Note:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Port 5 data direction register (DDR5)

The DDR5 register set the direction (input or output) for each pin (bit).

Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

● Settings as a peripheral input

To use a peripheral that has a Port 5 pin as an input pin, set that pin as an input port. The output latch data for that pin will no longer be valid.

Table 4.7-3 "Port 5 register function" lists the functions of the port 5 register.

Table 4.7-3 Port 5 register function

Register	Data	Read	Wirte	Read/ Write	Address	Initial value
Port 5 data register (PDR5)	0	Pin state is the "L" level.	Sets "0" to the output latch. Outputs and "L" level to the pin if the pin functions as an output port.	R/W	0010_H	---XXXXX$_B$
	1	Pin state is the "H" level.	Sets "1" to the output latch. Outputs and "H" level to the pin if the pin functions as an output port.			
Port 5 data direction register (DDR5)	0	Input port.	Disables output transistor and sets the pin as an input pin.	R/W	0011_H	---00000$_B$
	1	Output port	Enables output transistor and sets the pin as output pin.			

R/W: Readable and writable
X: Indeterminate
-: Unused

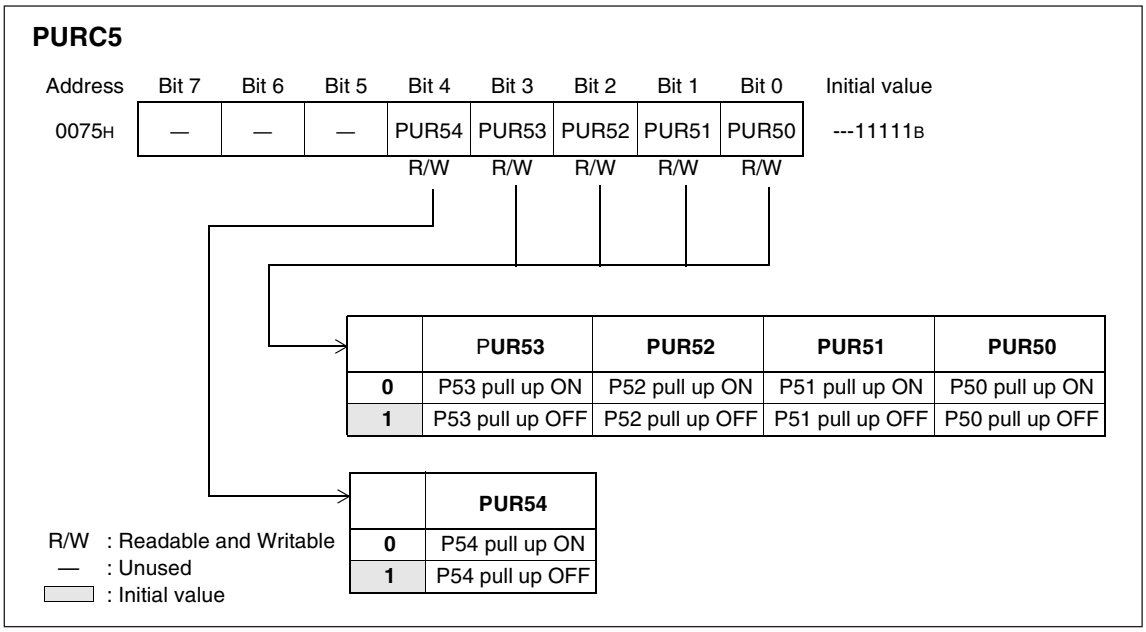
● Port 5 pull-up resistor control registers (PURC5)

By using pull-up resistor control register (PURC5), it is possible to turn on/off pull-up resistor bit .

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1). The states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

Figure 4.7-2 "Pull-up resistor control register setting (PURC5)" lists the function of Port 5 pull-up contorl resistor.

Figure 4.7-2 Pull-up resistor control register setting (PURC5)



	PUR53	PUR52	PUR51	PUR50
0	P53 pull up ON	P52 pull up ON	P51 pull up ON	P50 pull up ON
1	P53 pull up OFF	P52 pull up OFF	P51 pull up OFF	P50 pull up OFF

	PUR54
0	P54 pull up ON
1	P54 pull up OFF

4.7.2 Operation of Port 5

This section describes the operations of the port 5.

■ Operation of Port 5

● Operation as an output port

- Setting the corresponding DDR5 register bit to "1" sets a pin as an output port.
- When a pin is set as an output port, its output transistor is enabled and the pin outputs the data stored in the output latch.
- Writing data to the PDR5 registers stores the data in the output latch and outputs the data directly to the pin.
- Reading the PDR5 register returns the pin value.

● Operation as an input port

- Setting the corresponding DDR5 register bit to "0" sets a pin as an input port.
- When a pin is set as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Writing data to the PDR5 registers stores the data in the output latch but does not output the data to the pin.
- Reading the PDR5 register returns the pin value.

● Operation as a peripheral input

- A port pin is set as a peripheral input by setting the corresponding DDR5 register bit to "0".
- Reading the PDR5 register returns the pin value, regardless of whether or not the peripheral is using the input pin.

● Operation at reset

- Resetting the CPU initializes the DDR5 register values to "0". This sets the output transistors "OFF" (all pins become input ports) and sets the pins to the high-impedance state.
- The PDR5 register is not initialized by a reset. Therefore, to use as output ports, the output data must be set in the PDR5 register before setting the corresponding DDR5 register bits to output mode.

● Operation in stop mode

The pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the DDR5 register values. Table 4.7-4 "Port 5 pin state" lists the port 5 pin states.

Table 4.7-4 Port 5 pin state

Pin name	Normal operation main-sleep mode main-stop mode (SPL=0) sub-sleep mode sub-stop mode (SPL=0) watch mode (SPL=0)	Main-stop mode (SPL = 1) sub-stop mode (SPL = 1) watch mode (SPL = 1)	Reset
P50/$\overline{\text{INT20}}$ to P54/ $\overline{\text{INT24}}$	General-purpose I/O ports external interrupt input	Hi-Z / external interrupt input	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Note:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

4.8 Program Example for I/O Ports

This section gives an example program using the I/O ports.

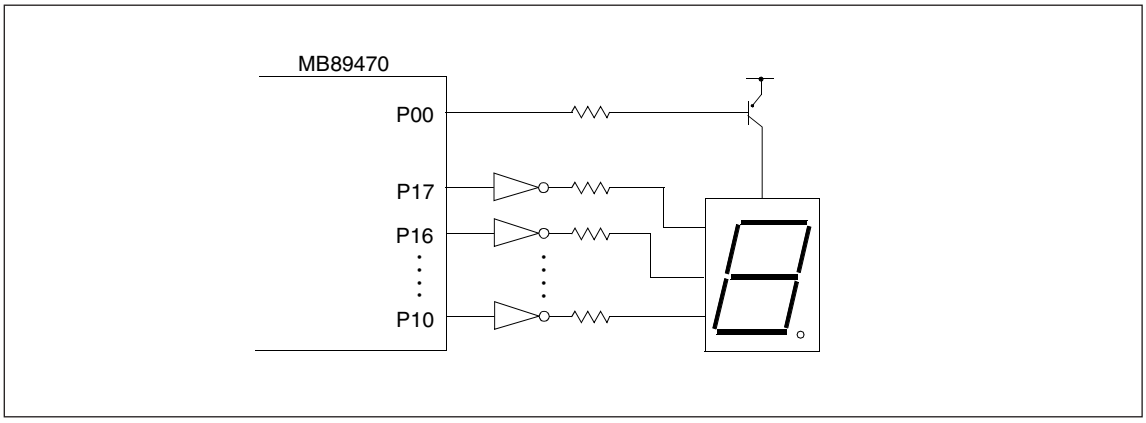
■ Program Example for I/O Ports

● Processing description

- Port 0 and port 1 are used to illuminate all elements of seven segment LED (eight segments if the decimal point is included).
- The P00 pin is used for the anode common pin of the LED and the P10 to P17 pins operate as the segment pins.

Figure 4.8-1 "Connection example for an eight segment LED" shows the connection example for an eight segment LED.

Figure 4.8-1 Connection example for an eight segment LED



● Coding example

```
PDR0  EQU  0000H              ; Address of the Port 0 data register
DDR0  EQU  0001H              ; Address of the Port 0 data direction register
PDR1  EQU  0002H              ; Address of the Port 1 data register
DDR1  EQU  0003H              ; Address of the Port 1 data direction register
;---------------Main program-----------------------------------------------
      CSEG                    ; [CODE SEGMENT]
       :
      CLRB PDR0:0             ; Set P00 to the "L" level.
      MOV  PDR1, #11111111B   ; Set all port 1 pins to the "H" level.
      MOV  DDR0, #11111111B   ; Set P00 as an output (#xxxxxxx1B).
      MOV  DDR1, #11111111B   ; Set all port 1 pins as outputs.
       :
      ENDS
;-------------------------------------------------------------------------------
      END
```

CHAPTER 5

TIMEBASE TIMER

This chapter describes the functions and operation of the timebase timer.

5.1 Overview of Timebase Timer

The timebase timer provides interval timer functions. Four different interval times can be selected. The timebase timer uses a 21-bit free-run counter which counts-up in sync with the internal count clock (divide-by-two the main clock source oscillation). The timebase timer also provides the timer output for the oscillation stabilization wait time , the operating clock for the watchdog timer and buzzer, continuous activation for the A/D converter.
The timebase timer stops operating in modes in which the main clock oscillator is stopped.

■ Interval timer function

The interval timer function generates repeated interrupts at fixed time intervals.

- The timer generates an interrupt each time the interval timer bit overflows on the timebase timer counter.
- The interval timer bit (interval time) can be selected from four different settings.

Table 5.1-1 "Timebase timer interval time" lists the available interval time for the timebase timer.

Table 5.1-1 Timebase timer interval time

Internal count clock cycle	Interval time
$2/F_{CH}$ (0.16μs)	$2^{13}/F_{CH}$ (approx. 0.66 ms)
	$2^{15}/F_{CH}$ (approx. 2.62 ms)
	$2^{18}/F_{CH}$ (approx. 21.0 ms)
	$2^{22}/F_{CH}$ (approx. 335.5 ms)

F_{CH}: Main clock oscillation frequency
The values enclosed in parentheses ( ) are for a 12.5 MHz main clock source oscillation.

■ Clock supply function

The clock supply function provides the timer output used for the main clock oscillation stabilization wait time (three values), and operation clock for some peripheral functions.

Table 5.1-2 "Clocks supplied by timebase timer" lists the cycles of the clocks that the timebase timer supplies to various peripherals.

Table 5.1-2 Clocks supplied by timebase timer

Clock destination	Clock cycle	Remarks
Main clock oscillation stabilization wait time	$2^{14}/F_{CH}$ (approx. 1.31 ms)	Selected by the oscillator stabilization wait time select bit of the system clock control register (SYCC: WT1, WT0), which is in the clock control section.
	$2^{17}/F_{CH}$ (approx. 10.5 ms)	
	$2^{18}/F_{CH}$ (approx. 21.0 ms)	
Watchdog timer	$2^{21}/F_{CH}$ (approx. 167.8 ms)	Count-up clock for the watchdog timer
Buzzer output	$2^{9}/F_{CH}$ to $2^{12}/F_{CH}$ (approx. 41.0 to 327.7 µs)	See Chapter 15 "Buzzer Output".
A/D converter	$2^{8}/F_{CH}$ (approx. 20.5 µs)	Frame cycle clock

F_{CH}: Main clock oscillation frequency

The values enclosed in parentheses ( ) are for a 12.5 MHz main clock source oscillation.

Reference:

The oscillation stabilization wait time should be used as a guide line since the oscillation cycle is unstable immediately after oscillation starts.

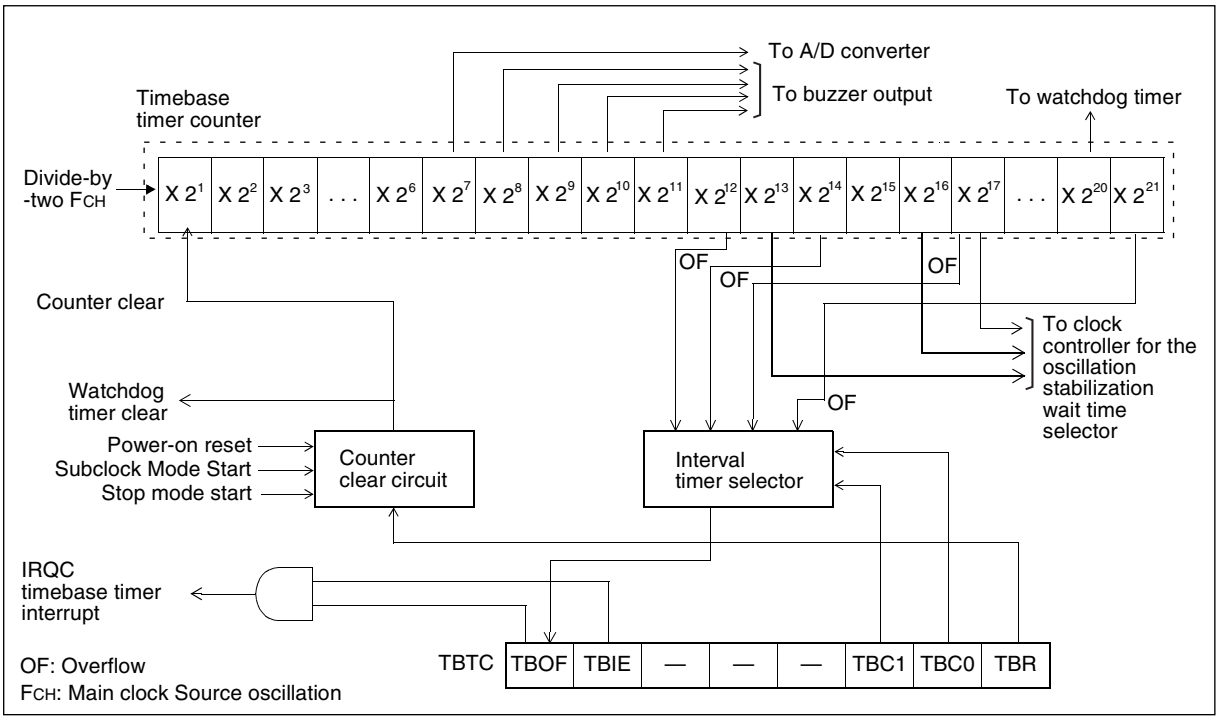
5.2 Block Diagram of Timebase Timer

The timebase timer consists of the following four blocks:

- **Timebase timer counter**
- **Counter clear circuit**
- **Interval timer selector**
- **Timebase timer control register (TBTC)**

■ Block diagram of timebase timer

Figure 5.2-1 Block diagram of timebase timer



● Timebase timer counter

A 21-bit up-counter that uses the divide-by-two main clock source oscillation as a count clock. The counter stops when the main clock oscillator is stopped.

● Counter clear circuit

In addition to being cleared by setting the TBTC register (TBR = "0"), the counter is cleared when device changes to main stop (STBC: STP = "1"), to subclock (SYCC: SCS = "0") and by power-on reset.

● Interval timer selector

Selects one of four operating timebase timer counter bits as the interval timer bit. An overflow on the selected bit triggers an interrupt.

● TBTC register

The TBTC register is used to select the interval timer bit, clear the counter, control interrupts, and check the state of the timebase timer.

5.3 Timebase Timer Control Register (TBTC)

The timebase timer control register (TBTC) is used to select the interval times bit, clear the counter, control interrupts, and check the state of the timebase timer.

■ Timebase timer control register (TBTC)

Figure 5.3-1 Timebase timer control register (TBTC)

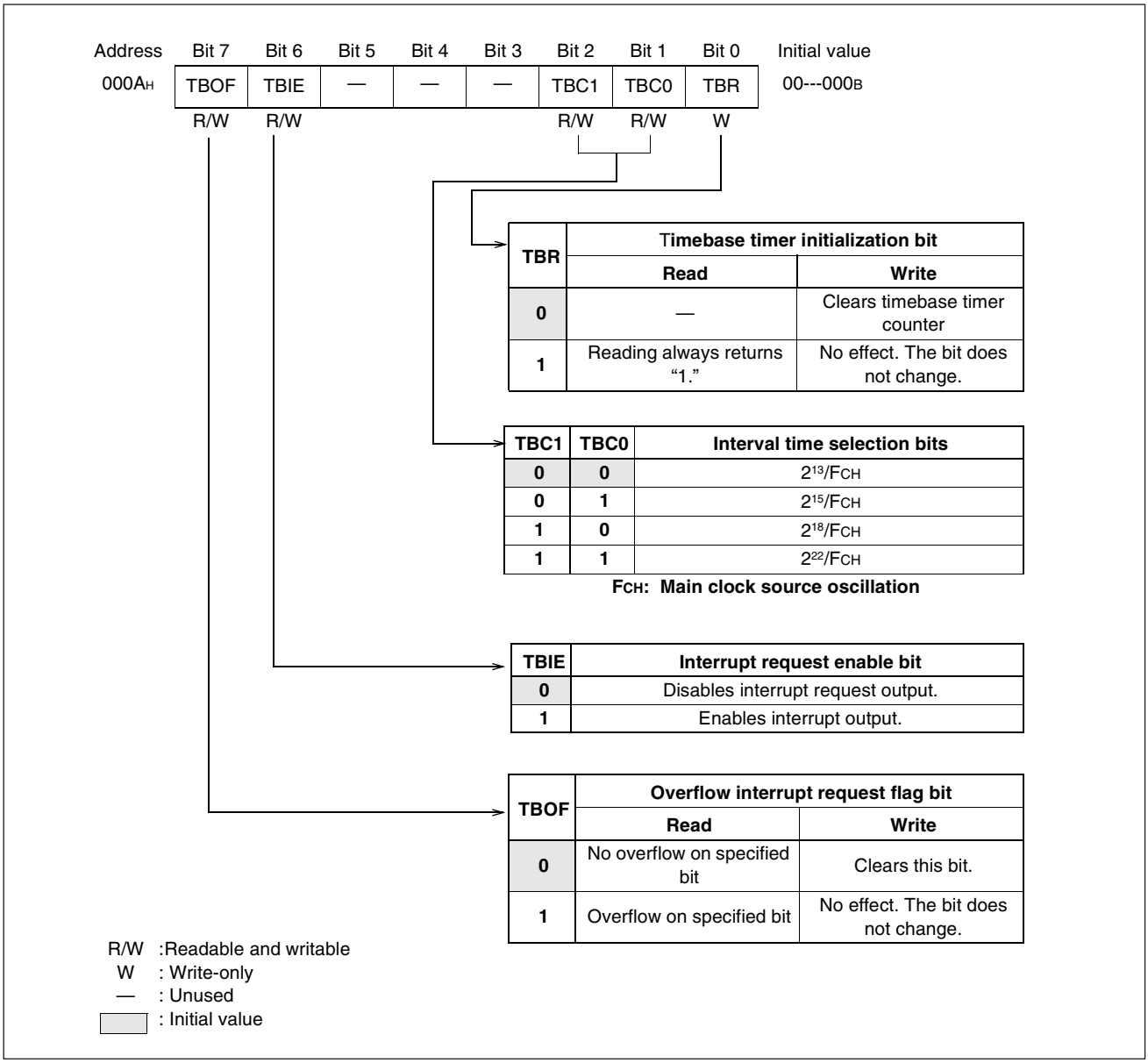


Table 5.3-1 Timebase timer control register (TBTC) bits

Bit		Function
Bit 7	TBOF: Overflow interrupt request flag bit	• This bit is set to "1" when counter overflow occurs on the specified bit of the timebase timer counter. • An interrupt request is output when both this bit and the interrupt request enable bit (TBIE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 6	TBIE: interrupt request	• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the overflow interrupt request flag bit (TBOF) are "1".
Bit 5 Bit 4 Bit 3	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 2 Bit 1	TBC1, TBC0: Interval time selection bits	• These bits select the cycle of the interval timer. • These bits select which bit of the timebase timer counter to use as the interval timer bit. • Four different interval times can be selected.
Bit 0	TBR: Timebase timer initialization bit	• This bit clears the timebase timer counter. • Writing "0" to this bit clears the counter to "000000_H". • Writing "1" has no effect and does not change the bit value. **Note:** The read value is always "1".

5.4 Timebase Timer Interrupt

The timebase timer can generate an interrupt request when an overflow occurs on the specified bit of the timebase counter (for the interval timer function).

■ Interrupts for interval timer function

The counter counts-up on the internal count clock. When an overflow occurs on the selected interval timer bit, the overflow interrupt request flag bit (TBTC: TBOF) is set to "1". At this time, an interrupt request (IRQC) to the CPU is generated if the interrupt request enable bit is enabled (TBTC: TBIE = "1"). Write "0" to the TBOF bit in the interrupt processing routine to clear the interrupt request. The TBOF bit is set when at the specified counter bit overflows, regardless of the TBIE bit value.

Note:

When enabling an interrupt request output (TBIE = "1") after wake-up from a reset, always clear the TBOF bit (TBOF = "0") at the same time.

References:

- An interrupt request is generated immediately if the TBOF bit is "1" when the TBIE bit is changed from disabled to enabled ("0" --> "1").
- The TBOF bit is not set if the counter is cleared (TBTC: TBR = "0") at the same time as an overflow on the specified bit occurs.

■ Oscillation stabilization wait time and timebase timer interrupt

If the interval time is set shorter than the main clock oscillation stabilization wait time, an interval interrupt request from the timebase timer (TBTC: TBOF = "1") is generated at the time when the main clock mode starts operation. In this case, disable the timebase timer interrupt (TBTC: TBIE = "0") when changing to a mode in which the main clock oscillation is stopped (main stop mode and subclock mode).

■ Register and vector table for timebase timer interrupts

Table 5.4-1 Register and vector table for timebase timer interrupt

Interrupt	Interrupt level settings register			Vector table address	
	Register	Set bit		Upper	Lower
IRQC	ILR4 ($007E_H$)	LC1 (Bit 1)	LC0 (Bit 0)	$FFE2_H$	$FFE3_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupt.

5.5 Operation of Timebase Timer

The timebase timer has the interval timer function and the clock supply function for some peripherals.

■ Operation of interval timer function (timebase timer)

Figure 5.5-1 "Interval timer function settings" shows the settings required to operate the interval timer function.

Figure 5.5-1 Interval timer function settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TBTC	TBOF	TBIE	—	—	—	TBC1	TBC0	TBR
	0	1				◎	◎	0

◎ : Used bit
1 : Set “1”.
0 : Set “0”.

Provided the main clock is oscillating, the timebase timer counter continues to count-up in sync with the internal count clock (divide-by-two main clock source oscillation).

After being cleared (TBR = "0"), the counter restarts counting-up from zero. The timebase timer sets the overflow interrupt request flag bit (TBOF) to "1" when an overflow occurs on the interval timer bit. Consequently, the timebase timer generates interrupt requests at fixed intervals (the selected interval time), based on the time that the counter is cleared.

■ Operation of clock supply function

The timebase timer is also used as a timer to generate the main clock oscillation stabilization wait time. The time from when the timebase timer counter is cleared and starts counting-up until an overflow occurs on the oscillation stabilization wait time bit is the oscillation stabilization wait time. One of three possible delay times is selected by the oscillation stabilization wait time bits of the system clock control register (SYCC: WT1, WT0).

The timebase timer also provides the clock for the watchdog timer, buzzer output, A/D converter. Clearing the timebase timer counter affects the operation of the continuous activation cycle of the A/D converter and the buzzer output.

If the timebase timer output is selected by the watchdog timer counter (WDTC: CS = "0"), it will also be cleared at the same time if the timebase timer counter is cleared due to changing to main stop (STBC: STP = "1").

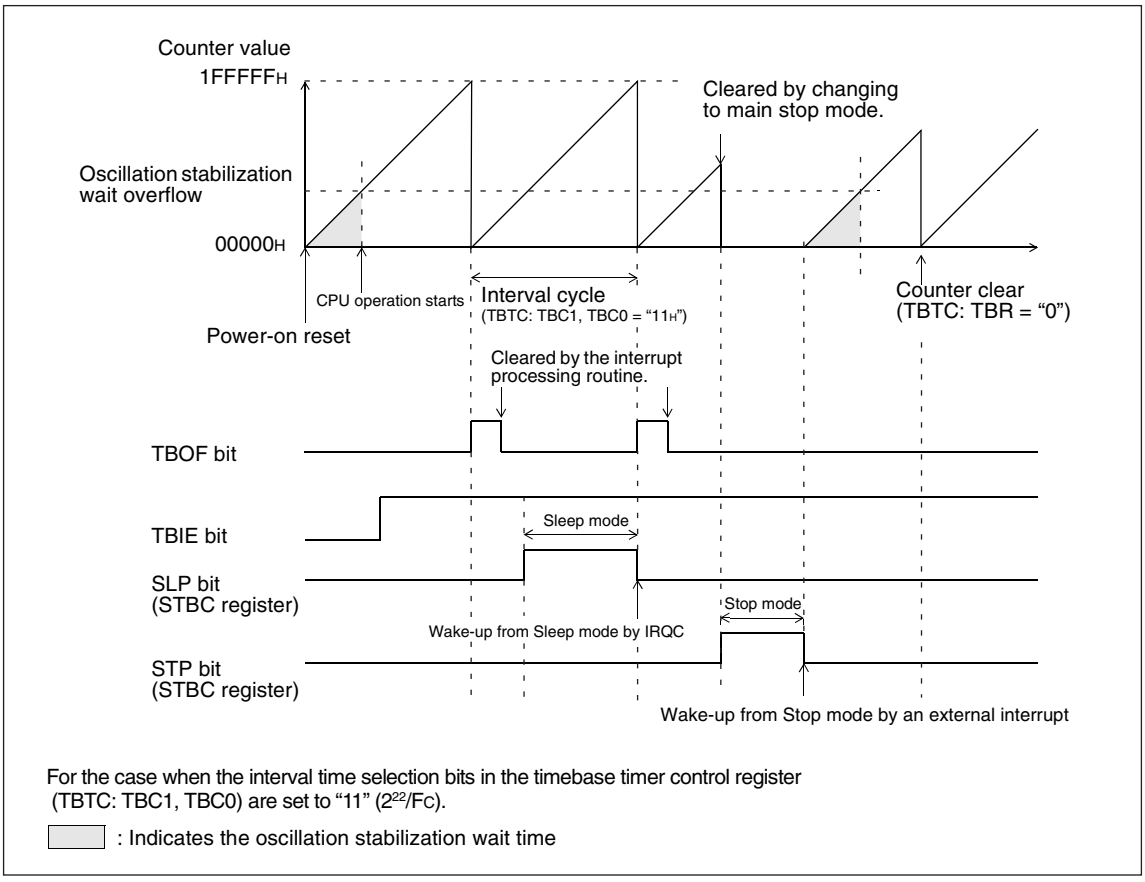
■ Operation of timebase timer

The state of following operations are shown in Figure 5.5-2 "Operation of timebase timer".

- A power-on reset occurs.
- Changes to sleep mode during operation of the interval timer function in the main clock mode.
- Changes to main stop mode.
- A counter clear request occurs.

The timebase timer is cleared by changing to subclock and main stop modes, and stops operation. The timebase timer counts the oscillation stabilization wait time after wake-up from subclock and main stop modes.

Figure 5.5-2 Operation of timebase timer



5.6 Notes on Using Timebase Timer

This section lists points to note when using the timebase timer.

■ Notes on using timebase timer

● Notes on setting bits by program

The system cannot recover from interrupt processing if the overflow interrupt request flag bit (TBTC: TBOF) is "1" and the interrupt request enable bit is enabled (TBTC: TBIE = "1"). Always clear the TBOF bit.

● Clearing timebase timer

In addition to being cleared by the timebase timer initialization bit (TBTC: TBR = "0"), the timer is cleared whenever the main clock oscillation stabilization wait time is required. When the timebase timer is selected as a count clock of the watchdog timer, clearing the timebase timer also clears the watchdog timer.

● Using as timer for oscillation stabilization wait time

As the main clock source oscillation is stopped when the power is turned on during main-stop mode, the timebase timer provides the oscillation stabilization wait time after the oscillator starts.

An appropriate oscillation stabilization wait time must be selected for the type of resonator connected to the main clock oscillator (clock generator).

See Section 3.6.5 "Oscillation Stabilization Wait Time".

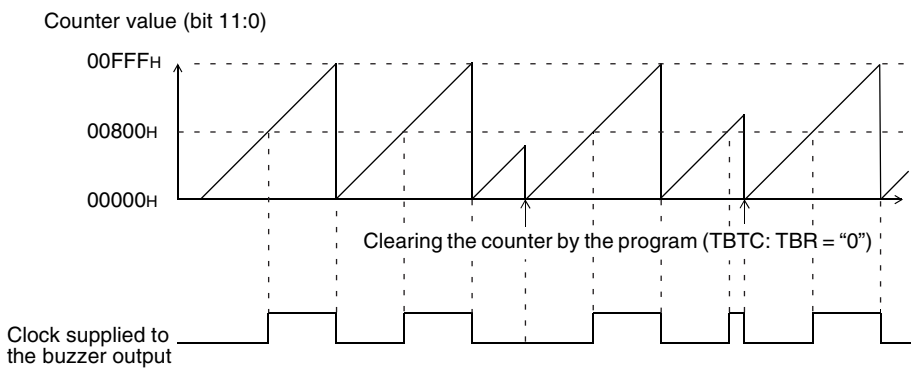
● Notes on peripheral functions that provided a clock supply from timebase timer

In modes in which the main clock source oscillation is stopped, the timebase timer also stops, and the counter is cleared.

As the clock derived from the timebase timer restarts output from its initial state when the timebase timer counter is cleared, the “H” level may be shorter or the "L" level longer by a maximum of half cycle. The clock of the watchdog timer also restarts output from its initial state.

Figure 5.6-6 shows the effect on the buzzer output of clearing the timebase timer.

Figure 5.6-1 Effect on buzzer output of clearing timebase timer



For the case when the buzzer selection bits in the buzzer register (BUZR: BZ2, BZ1, BZ0) are set to “001”.
(Divide-by-4096 source oscillation, approximately 3.052 kHz output at 12.5 MHz operation)

5.7 Program Example for Timebase Timer

This section gives a program example for the timebase timer.

■ Program example for timebase timer

● Processing description

- Generates repeated interval timer interrupts at $2^{18}/F_{CH}$ (F_{CH}: Main clock source oscillation) intervals. At this time, the interval time is approximately 21.0 ms (at 12.5 MHz operation).

● Coding example

```
TBTC  EQU   0000AH            ; Address of the timebase timer control register

TBOF  EQU   TBTC:7            ; Define the interrupt request flag bit.

ILR4  EQU   007EH             ; Address of the interrupt level setting register 3

INT_V DSEG  ABS               ; [DATA SEGMENT]
      ORG   0FFE2H
IRQC  DW    WARI              ; Set interrupt vector.
INT_V ENDS
;-----Main program-----------------------------------------------------------
      CSEG                    ; [CODE SEGMENT]
                              ; Stack pointer (SP) etc. are already initialized.
       :
      CLRI                    ; Disable interrupts.
      MOV   ILR4,#11111101B   ; Set interrupt level (level 1).
      MOV   TBTC,#01000100B   ; Clear interrupt request flag enable interrupt
                                request output, select 2^18/FCH, and clear timebase
                                timer.
      SETI                    ; Enable interrupts.
       :
;-----Interrupt program------------------------------------------------------
WARI  CLRB  TBOF              ; Clear interrupt request flag.
      PUSHW A
      XCHW  A,T
      PUSHW A
       :
      User processing
       :
      POPW  A
      XCHW  A,T
      POPW  A
      RETI
      ENDS
;---------------------------------------------------------------------------
      END
```

CHAPTER 6
WATCHDOG TIMER

This chapter describes the functions and operation of the watchdog timer.

6.1 Overview Watchdog Timer

The Watchdog timer is a 1-bit counter that uses, as its count clock source, either the timebase timer derived from the main clock, or the watch prescaler derived from the subclock. The watchdog timer resets the CPU if not cleared within a fixed time after activation.

■ Watchdog timer function

The watchdog timer is a counter provided to guard against program runaway. Once activated, the counter must be repeatedly cleared within a fixed time interval. If the program becomes trapped in an endless loop or similar and does not clear the counter within the fixed time, the watchdog timer generates a four-instruction cycle watchdog reset to the CPU.

Either the timebase timer output or the watch prescaler output can be selected as the watchdog timer count clock.

Table 6.1-1 "Watchdog timer interval time" lists the watchdog timer interval times. If not cleared, the watchdog timer generates a watchdog reset at a time between the minimum and maximum times listed. Clear the counter within the minimum time given in the table.

Table 6.1-1 Watchdog timer interval time

	Count clock	
	Timebase timer output (main clock oscillator frequency at 12.5 MHz)	Watch prescaler output (subclock oscillator frequency at 32.768 kHz)
Minimum time	Approx. 168 ms*1	500 ms*2
Maximum time	Approx. 336 ms	1000 ms

*1: Divide-by-two the main clock source oscillation (F_{CH}) x timebase timer count value (2^{20}).

*2: The time of a clock cycle at the subclock oscillator frequency (F_{CL}) x watch prescaler count (2^{14}).

See Section 6.4"Watchdog Timer Operation" for the details on the minimum and maximum time of the watchdog timer interval times.

Note:

The watchdog timer counter is cleared whenever the device changes to sleep or stop watch mode. Operation halts until the device returns to normal operation (RUN state).

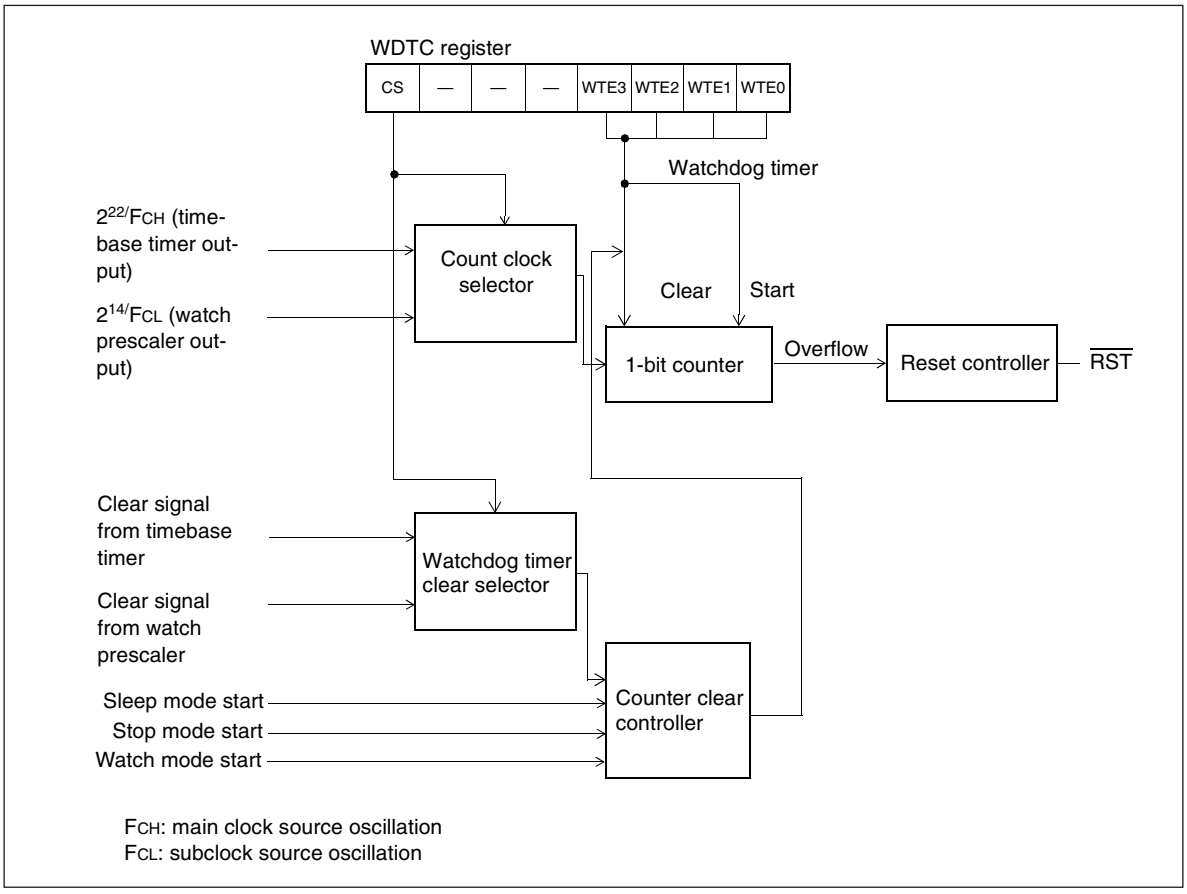
6.2 Block Diagram of Watchdog Timer

The Watchdog timer consists of the following six blocks:

- **Count clock selector**
- **Watchdog timer counter**
- **Reset controller**
- **Watchdog timer clear selector**
- **Counter clear controller**
- **Watchdog timer control register (WDTC)**

■ Block diagram of watchdog timer

Figure 6.2-1 Block diagram of watchdog timer



● Count clock selector

The count clock selector selects the count clock for the watchdog timer counter. Either the timebase timer counter output or the watch prescaler output can be selected as the count clock.

- Watchdog timer counter (1-bit counter)

  A 1-bit counter that uses the timebase timer output or the watch prescaler output as a count clock

- Reset controller

  Generates a reset signal to the CPU when an overflow occurs on the watchdog timer counter.

- Watchdog timer clear selector

  The clear selector selects a watchdog timer clear signal from either the timebase timer or watch prescaler at the same time as the count clock selector selects a clock. (It selects the clear signal from the selected clock source.)

- Counter clear controller

  Controls clearing and halting the operation of watchdog timer counter.

- WDTC register

  The WDTC register is used to select the count clock, and to activate or clear the watchdog timer counter. As the register is write-only, the bit manipulation instructions cannot be used.

6.3 Watchdog Timer Control Register (WDTC)

The Watchdog timer control register (WDTC) is used to activate or clear the watchdog timer.

■ Watchdog timer control register (WDTC)

Figure 6.3-1 Watchdog timer control register (WDTC)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0009H	CS	—	—	—	WTE3	WTE2	WTE1	WTE0	0---XXXXB
	R/W				W	W	W	W	

WTE3	WTE2	WTE1	WTE0	Watchdog timer control bits
0	1	0	1	• Activate the watchdog timer (when writing for the first time after a reset). • Clear the watchdog timer (when writing for the second and subsequent times after a reset).
Other than the above				No operation

CS	Watchdog timer control bits
0	Cycle time of timebase timer output (2^{22}/FCH*1)
1	Cycle time of watch prescaler output (2^{14}/FCL*2)

R/W : Readable and Writable
W : Write-only
— : Unused
X : Indeterminate
□ : Initial value

Note: As the bit 3 ~ bit 0 are write-only, the bit manipulation instructions cannot be used.
*1: FCH: main clock source oscillation
*2: FCL: subclock source oscillation

Table 6.3-1 Watchdog timer control register (WDTC) bits

Bit		Function
Bit 7	CS: Count clock select bit	At watchdog timer startup, selects the watchdog timer count clock. Selects either the timebase timer output or the watch prescaler output as the count clock. **Reference:** When using the subclock mode, always select the watch prescaler output. Make the count clock selection when the watchdog timer is started. Once the timer is started, do not change the count clock. Bit operation instructions cannot be used.
Bit 6 Bit 5 Bit 4	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on operation.
Bit 3 Bit 2 Bit 1 Bit 0	WTE3, WTE0: Watchdog timer control bits	Writing "0101_B" to these bits activates (when writing for the first time after a reset) or clears (when writing for the second and subsequent times after a reset) the watchdog timer. Writing a value other than "0101_B" has no effect on the operation. **Note:** The read value is "1111_B". The bit manipulation instructions cannot be used.

6.4 Operation of Watchdog Timer

The watchdog timer generates a watchdog reset when the watchdog timer counter overflows.

■ Operation of watchdog timer

● Activating watchdog timer

- The watchdog timer is activated by writing "0101_B" to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the first time after a reset.
  The count clock select bit (WDTC: CS) is written to the desired state in the same write operation.
- Once activated, the watchdog timer cannot be stopped other than by a reset.

● Clearing watchdog timer

- The watchdog timer counter is cleared by writing "0101_B" to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the second or subsequent times after a reset.
- If the counter is not cleared within the interval time of the watchdog timer, the counter overflows and the watchdog timer generates an internal reset signal for four-instruction cycles.

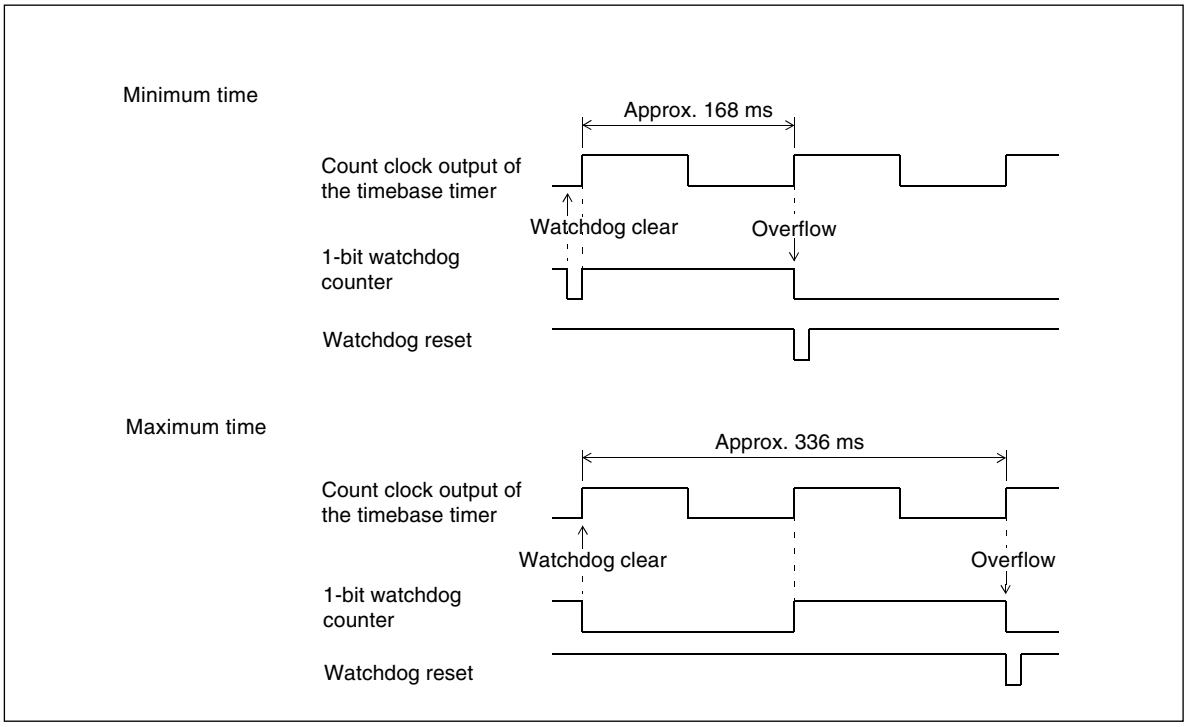
● Interval time of watchdog timer

The interval time changes depending on when the watchdog timer is cleared.

Figure 6.4-1"Watchdog timer clear and interval time" shows the relationship between the watchdog timer clear timing and the interval time.

The indicated times apply if the timebase timer output is selected as the count clock, and the main clock source oscillation is 12.5 MHz.

Figure 6.4-1 Watchdog timer clear and interval time



6.5 Notes on Using Watchdog Timer

This section lists points to note when using the watchdog timer.

■ Notes on Using Watchdog Timer

● Stopping watchdog timer

Once activated, the watchdog timer cannot stop until a reset generates.

● Count clock selection

The count clock select bit (WDTC: CS) can only be changed during activating or clearing the watchdog timer. You can change it by writing the desired state to the count clock select bit (WDTC: CS) at the same time as you write "0101_B" to the watchdog control bits (WDTC: WTE3 to WTE0) to activate or clear the watchdog timer. Therefore, the CS bit cannot be changed by a bit operation instruction. Do not change the CS bit after activating the timer.

In the subclock mode, the main clock source oscillation is stopped, which means that the timebase timer also stops. For the watchdog timer to operate in subclock mode, then, the watch prescaler must have been selected in advance as the count clock (WDTC: CS = 1).

● Clearing watchdog timer

- Clearing the counter being used as a count clock of the watchdog timer (timebase timer or watch prescaler) also simultaneously clears the watchdog timer counter.
- The watchdog timer counter is cleared on changing to sleep, stop or watch mode.

● Notes on programming

When writing a program in which the watchdog timer is repeatedly cleared in the main loop, the processing time for the main loop including interrupt processing must be less than the minimum watchdog timer interval time.

● Operation in subclock mode

If the watchdog reset signal is generated in subclock mode, operation will start in main clock mode after an oscillation stabilization delay time. Therefore, if the device has the reset signal output option, a reset signal will be output during the oscillation stabilization delay time.

6.6 Program Example for Watchdog Timer

This section gives a program example for the watchdog timer.

■ Program Example for Watchdog Timer

● Processing description

- Selects the watch prescaler as the count clock and activates the watchdog timer immediately after the program.
- Clears the watchdog timer in each loop of the main program.
- The processing time for the main loop, including interrupt processing, must be less than the minimum interval time of the watchdog timer (approximately 500 ms at 32.768 kHz operation).

● Coding example

```
WDTC    EQU    0009H          ; Address of the watchdog timer control register
WDT_CLR EQU    10000101B


VECT    DSEG   ABS            ; [DATA SEGMENT]
        ORG    0FFFEH
RST_V   DW     PROG           ; Set reset vector.
VECT    ENDS
;-----Main program-------------------------------------------------------------
        CSEG                  ; [CODE SEGMENT]
PROG                          ; Initialization routine after a reset
        MOVW   SP,#0280H      ; Set initial value of stack pointer
                                (for interrupt processing).
         :
        Initialization of peripheral functions (interrupts), etc.
         :
INIT    MOV    WDTC,#WDT_CLR  ; Activate the watchdog timer.

         :
MAIN    MOV    WDTC,#WDT_CLR  ; Clear the watchdog timer.
         :
        User processing (interrupt processing may occur during this cycle)
         :
        JMP    MAIN           ; The loop must be executed in less than the
                                minimum interval time of the watchdog timer.
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 7

WATCH PRESCALER

This chapter describes the functions and operation of the watch prescaler.

7.1 Overview Watch Prescaler

The Watchdog prescaler provides interval timer functions. Six different interval times can by selected. The watch prescaler uses a 17-bit free-run counter which counts-up in sync with a subclocl generated by the clock generator.
The watch prescaler also provides the timer output for the subclock oscillation stabilization wait time and the operating clock for watchdog timer and buzzer output.

■ Interval timer function (watch literrupt)

- The interval timer function generates repeated interrupts at fixed intervals with the subclock used as the count clock.
- Interrupts are generated by watch prescaler interval timer divided clock outputs.
- The interval timer divided clock output (interval time) can be selected from different settings.
- The watch prescaler counter can be cleared.

Table 7.1-1 "Watch prescaler interval time" lists the available interval times for the watch prescaler.

Table 7.1-1 Watch prescaler interval time

Subclock Cycle Time	Interval time
$1/F_{CL}$ (approx. 30.5 μs)	$2^{10}/F_{CL}$ (31.25 ms)
	$2^{13}/F_{CL}$ (0.25 s)
	$2^{14}/F_{CL}$ (0.50 s)
	$2^{15}/F_{CL}$ (1.00 s)
	$2^{16}/F_{CL}$ (2.00 s)
	$2^{17}/F_{CL}$ (4.00 s)

F_{CL}: Subclock source oscillation
The values enclosed in parentheses ( ) are for a 32.768 kHz subclock source oscillation.

Note:

The watch prescaler cannot be used in devices in which a single clock option has been selected.

■ Clock supply function

The watch prescaler has the following clock supply functions:

- The timer output used for the subclock oscillation stabilization wait time (one value)
- The clock used for the watchdog timer (one value)
- The clock used for the buzzer output (three values)

Table 7.1-2 "Clocks supplied by watch prescaler" lists the cycles of the clocks that the watch prescaler supplies to various peripherals.

Table 7.1-2 Clocks supplied by watch prescaler

Subclock destination	Subclock cycle	Remarks
Subclock oscillation stabilization wait time	$2^{15}/F_{CL}$ (1.00 s)	Do not switch to the subclock mode during the oscillator stabilization wait time.
Watchdog timer	$2^{14}/F_{CL}$ (0.50 s)	count-up clock for the watchdog timer
Buzzer output	$2^{3}/F_{CL}$ to $2^{5}/F_{CL}$ (approx. 0.24 to 0.98 ms)	See Chapter 15 "Buzzer Output".

F_{CL}: Subclock source oscillation
The values enclosed in parentheses ( ) are for a 32.768 kHz subclock source oscillation.

Reference:

The oscillation stabilization wait time should be used as a guideline since the oscillation cycle is unstable immediately after oscillation starts.

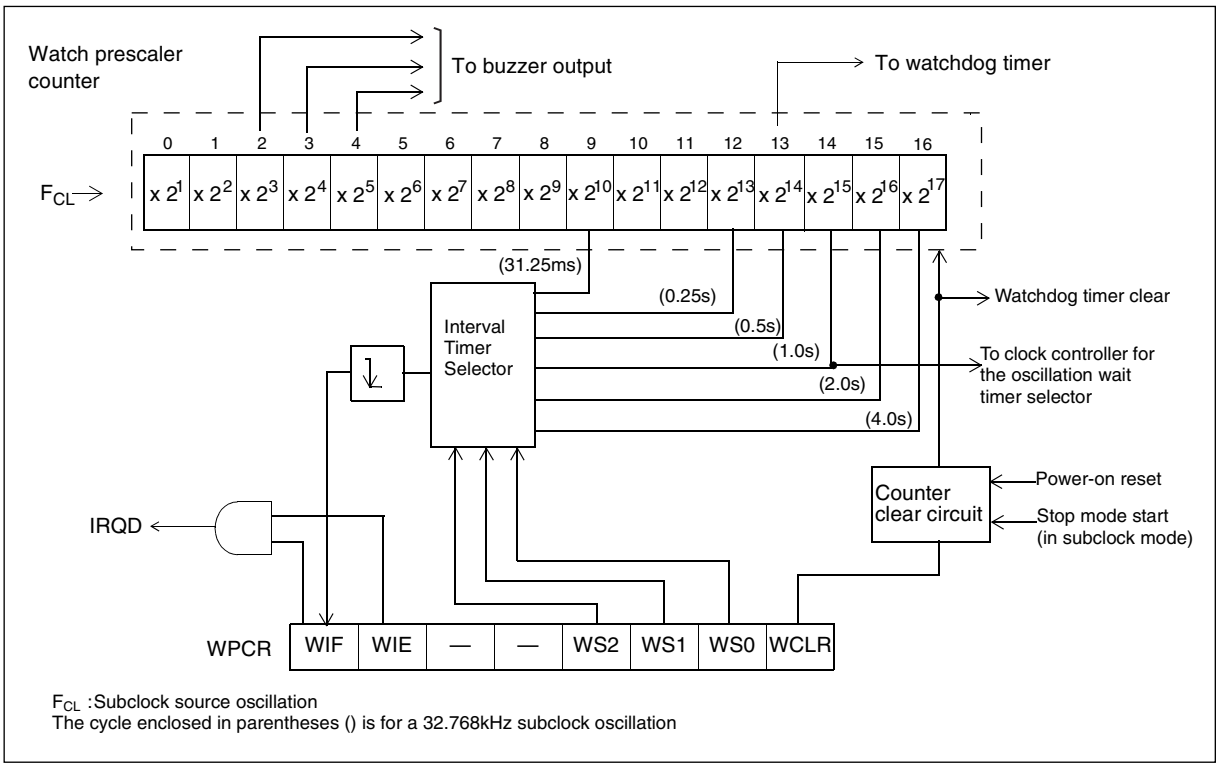
7.2 Block Diagram of Watch Prescaler

The Watch prescaler consists of the following four blocks:

- **Watch prescaler counter**
- **Counter clear circuit**
- **Interval timer selector**
- **Watch prescaler control register (WPCR)**

■ Block diagram of watch prescaler

Figure 7.2-1 Block diagram of watch prescaler



● Watch prescaler counter

A 17-bit up-counter that uses the subclock source oscillation clock as its count clock.

● Counter clear circuit

In addition to being cleared by setting the WPCR register (WCLR = "0"), the counter is cleared when the device changes to sub-stop mode (STBC : STP = "1") and by power-on reset.

● Interval timer selector

This circuit selects one of six divided clock outputs of the watch prescaler counter as the interval timer output. The falling edge of the selected output is the event that triggers the watch interrupt.

● WPCR register

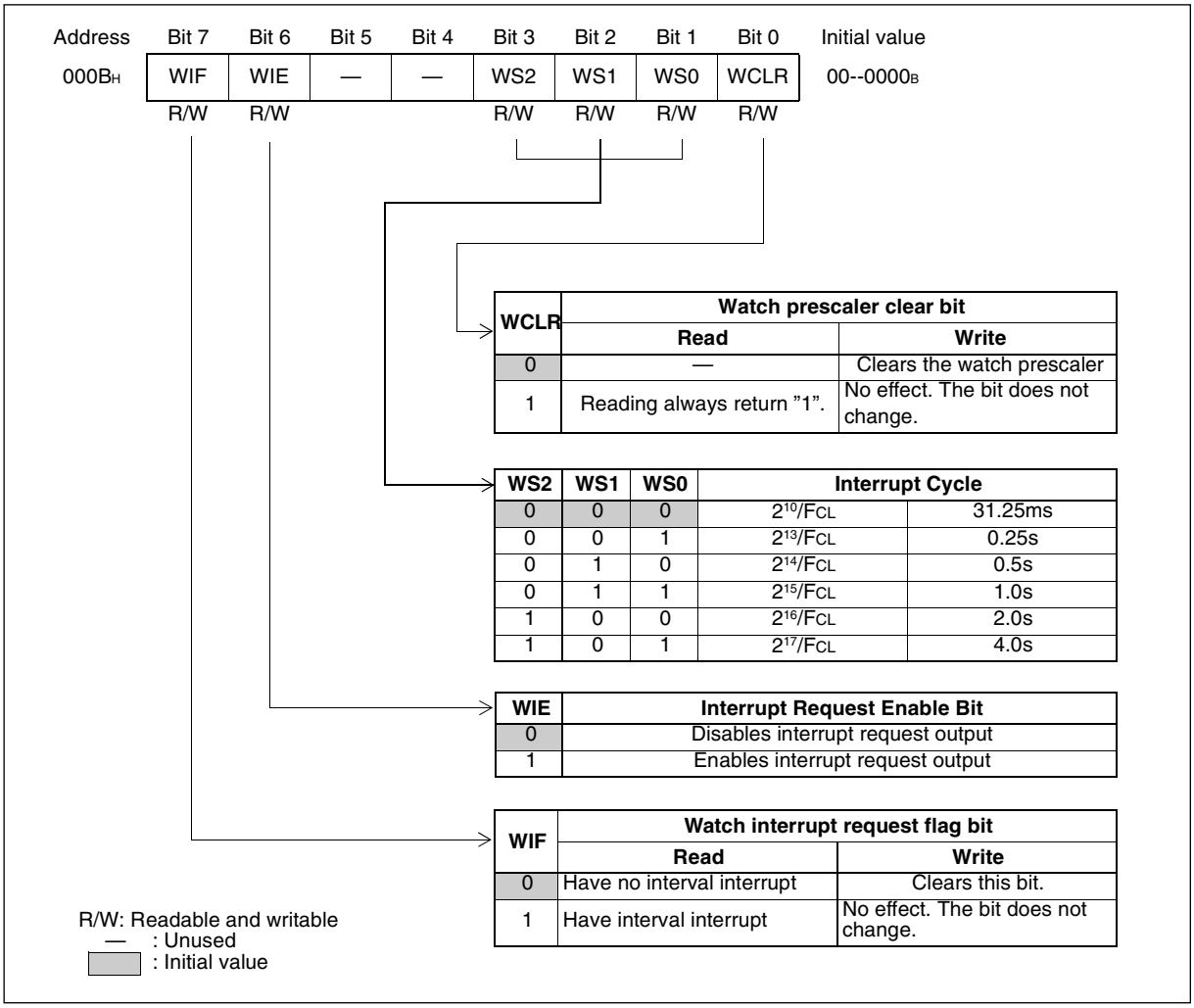
The WPCR register is used to select the interval time bit, clear the counter, control interrupts, and check the state of the watch prescaler.

7.3 Watch Prescaler Control Register (WPCR)

The Watch prescaler control register (WPCR) is used to select the interval timer bit, clear the counter, control interrupts, and check state of the watch prescaler.

■ Watch prescaler control register (WPCR)

Figure 7.3-1 Watch prescaler control register (WPCR)



WCLR	Watch prescaler clear bit	
	Read	Write
0	—	Clears the watch prescaler
1	Reading always return ”1”.	No effect. The bit does not change.

WS2	WS1	WS0	Interrupt Cycle	
0	0	0	$2^{10}/F_{CL}$	31.25ms
0	0	1	$2^{13}/F_{CL}$	0.25s
0	1	0	$2^{14}/F_{CL}$	0.5s
0	1	1	$2^{15}/F_{CL}$	1.0s
1	0	0	$2^{16}/F_{CL}$	2.0s
1	0	1	$2^{17}/F_{CL}$	4.0s

WIE	Interrupt Request Enable Bit
0	Disables interrupt request output
1	Enables interrupt request output

WIF	Watch interrupt request flag bit	
	Read	Write
0	Have no interval interrupt	Clears this bit.
1	Have interval interrupt	No effect. The bit does not change.

Table 7.3-1 Watch prescaler control register (WPCR) bits

Bit		Function
Bit 7	WIF: Watch interrupt request flag bit	• Set to "1" by the falling edge of the selected interval timer divided output. • An interrupt request is output when both this bit and the interrupt request enable bit (WIE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 6	WIE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the watch interrupt request flag bit (WIF) are "1".
Bit 5 Bit 4	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 3 Bit 2 Bit 1	WS2, WS1, WS0: Watch interrupt interval time selection bits	• Select interval timer cycle. • Specify which bit of the watch prescaler counter (or which divided output) will be used for the interval timer. • One of six interval times may be selected.
Bit 0	WCLR: Watch prescaler clear bit	• Bit used to clear the watch prescaler counter. • Writing "0" to this bit clears the counter to 00000_H. Writing "1" has no effect and does not change the bit value. **Note:** The read value is always "1".

7.4 Watch Prescaler Interrupt

The watch prescaler generates an interrupt request at the falling edge of the specific divided output (interval timer function).

■ Interrupts for interval timer function (watch interrupt)

The watch prescaler counter counts up, clocked by the subclock source oscillation. Unless the system is in main-stop mode, the watch interrupt request flag is set to "1" (WPCR: WIF = 1) at the end of the selected time interval. At this time, an interrupt request (IRQD) to the CPU is generated if the interrupt request enable bit is enabled (WPCR: WIE = "1"). Write "0" to the WIF bit in the interrupt processing routine to clear the interrupt request. The WIF bit is set when the specified divided output falls, regardless of the WIE bit value.

Note:

When enabling an interrupt request output (WIE = "1") after wake-up from a reset, always clear the WIF bit (WIF = "0") at the same time.

References:

An interrupt request is generated immediately if the WIF bit is "1" when the WIF bit is changed from disabled to enabled ("0" --> "1").

The WIF bit is not set if the counter cleared (WPCR: WCLR = "0") at the same time as an overflow on the specified bit occurs.

■ Oscillation stabilization wait time and watch interrupt

If the interval time is set shorter than the subclock oscillation stabilization wait time, an watch interrupt request from the watch prescaler (WPCR: WIF = "1") is generated at the time when CPU wakes up from sub-stop mode by an external interrupt. In this case, disable the watch prescaler interrupt (WPCR: WIE = "0") when changing to sub-stop mode.

■ Register and vector table for watch prescaler interrupt

Table 7.4-1 "Register and vector for watch prescaler interrupt" lists the register and vector table for watch prescaler interrupt.

Table 7.4-1 Register and vector for watch prescaler interrupt

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQD	ILR4 ($007E_H$)	LD1 (bit3)	LD0 (bit2)	$FFE0_H$	$FFE1_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupt operations.

7.5 Operation of Watch Prescaler

The watch prescaler has the interval timer function and the clock supply function.

■ Operation of interval timer function (watch prescaler)

Figure 7.5-1 "Interval timer function settings" shows the settings required to operate the interval timer function.

Figure 7.5-1 Interval timer function settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPCR	WIF	WIE	—	—	WS2	WS1	WS0	WCLR
	0	1			◎	◎	◎	0

◎: Used bit
1: Set “1”
0: Set “0”

Provided the subclock is oscillating, the watch prescaler 17-bit counter continues to count-up using the subclock as its count clock.

After being cleared (WCLR = "0"), the counter restarts counting-up from "00000_H". When the counter reaches a full count of "$1FFFF_H$", the next count takes it "00000_H", and it continues to count-up. As the count proceeds, a falling edge will eventually occur at the selected divided clock output. At this time, unless the system is in main clock stop mode, the watch prescaler sets the watch interrupt request flag bit (WIF) to "1". Consequently, the watch prescaler generates interrupt requests at fixed intervals (the selected interval time), based on the time that the counter is cleared.

■ Operation of clock supply function

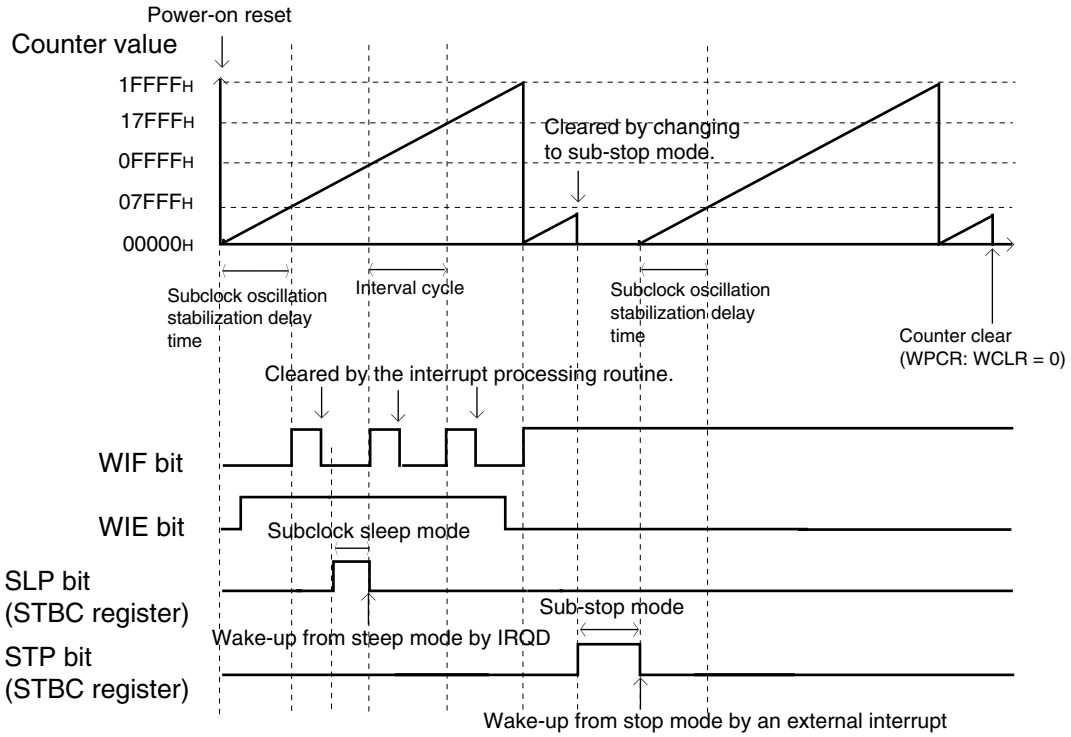
The watch prescaler is also used as a timer to generate the subclock oscillation stabilization wait time. The time between the counter cleared state and the falling edge of the MSB output is used the subclock oscillation stabilization wait time ($2^{15}/F_{CL}$, where F_{CL} is subclock source oscillation).

The watch prescaler also provides the clock for the watchdog timer and buzzer output. Clearing the watch prescaler counter affects the operation of the buzzer output. When the watch prescaler is selected as the clock source for the watchdog timer (WDTC: CS = 1) both counters are cleared simultaneously.

■ Operation of watch prescaler

Figure 7.5-2 "Operation of watch prescaler " shows counter states when the interval timer is operating in subclock mode and the system goes into the sleep and stop modes, and when there is a counter clear request.

Figure 7.5-2 Operation of watch prescaler



For the case when the interrupt interval time selection bits in the watch prescaler control register (WPCR: WS2, WS1, WS0) are “set to 011” (2^{15}/FCL).

7.6 Notes on Using Watch Prescaler

This section lists points to note when using the watch prescaler. The watch prescaler cannot be used in devices in which the single-clock option has been selected.

■ Notes on using watch prescaler

● Notes on setting bits by program

The system cannot recover from interrupt processing if the interrupt request flag bit (WPCR : WIF) is "1" and the interrupt request enable bit is enabled (WPCR: WIE = "1"). Always clear the WIF bit.

● Clearing Watch prescaler

In addition to being cleared by the watch prescaler clear bit (WPCR: WCLR = "0"), the watch prescaler is cleared wherever the subclock oscillation stabilization wait time is required.

When the watch prescaler is selected as a count clock of the watchdog timer (WDTC: CS = "1"), clearing the watch prescaler also clears the watchdog timer.

● Using as timer for oscillator stabilization delay time

As the subclock source oscillation is stopped when the power is turned on and during sub-stop mode, the watch prescaler provides the oscillation stabilization wait time after the oscillator starts. Do not switch clock modes from main clock to subclock during this delay time (immediately after power on, etc.)

The subclock oscillation stabilization wait time is fixed.

See Section 3.6.5 "Oscillation Stabilization Wait Time" for details.

● Notes on watch interrupt

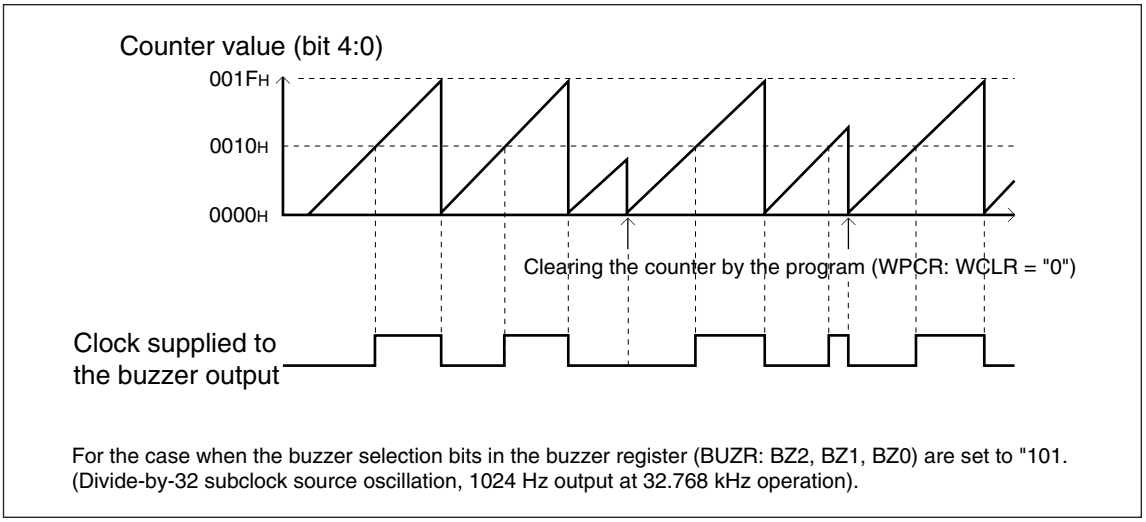
In main-stop mode, the watch prescaler counter operates, but no interrupt requests are generated.

● Notes on peripheral functions that provides a clock supply from watch prescaler

As the clock derived from the watch prescaler restarts output from the its initial state when the watch prescaler counter is cleared, the "H" level may be shorter or the "L" level longer by a maximum of half cycle. The clock of the watchdog timer also restarts output from its initial state. However, as the watchdog timer counter is cleared at the same time, the watchdog timer operates in normal cycle.

Figure 7.6-1 "Effect on buzzer output due to clearing of watch prescaler" shows the effect on the buzzer output of clearing the watch prescaler.

Figure 7.6-1 Effect on buzzer output due to clearing of watch prescaler



For the case when the buzzer selection bits in the buzzer register (BUZR: BZ2, BZ1, BZ0) are set to "101. (Divide-by-32 subclock source oscillation, 1024 Hz output at 32.768 kHz operation).

7.7 Program Example for Watch Prescaler

This section gives program example for the watch prescaler.

■ Program example for the watch prescaler

● Processing description

Generates repeated watch interrupts at $2^{15}/F_{CL}$ (F_{CL} = subclock source oscillation) intervals. At this time, the interval time is 1 second (at 32.768 kHz operation).

● Coding example

```
WPCR    EQU    000BH           ; Address of watch prescaler control register
WIF     EQU    WPCR:7          ; Define the watch interrupt request flag bit.
ILR4    EQU    007EH           ; Address of the interrupt level setting register 2
INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFE0H
IRQD    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        MOV    ILR4,#11111011B ; Set interrupt priority (level 2).
        MOV    WPCR,#01000110B ; Clear interrupt request flag, enable interrupt
                                 request output, select 2^15/FCL, and clear watch
                                 prescaler.
        SETI                   ; Enable interrupts.
         :
;-----Interrupt program--------------------------------------------------------
WARI    CLRB   WIF             ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 8

8-BIT PWM TIMER

This chapter describes the functions and operation of the 8-bit PWM timer.

8.1 Overview of 8-bit PWM Timer

The 8-bit PWM timer can be selected to function as either an interval timer or PWM timer with 8-bit resolution. The interval timer function count-up in sync with one of four internal count clocks. Therfore, an 8-bit interval timer time can be set and the output can be used to generate variable frequency square waves. Also, the 8-bit PWM timer can be used as a D/A converter by connecting the PWM output to low pass filter.

■ Interval timer function (Square wave output function)

The interval timer function generates repeated interrupts at variable time intervals.

Also, as the 8-bit PWM timer can invert the output level of the pin (PWM) each time an interrupt is generated, the 8-bit PWM timer can output a variable frequency square waves.

- The interval timer can operate with a cycle among 1 and 2^8 times the count clock cycle.
- The count clock can be selected from four different clocks.

Table 8.1-1 "Interval time and square wave output range" lists the range for the interval time and square wave output.

Table 8.1-1 Interval time and square wave output range

	Count clock cycle		Interval time	Square wave output (Hz)
1	Internal count clock	1 t_{inst}	1 t_{inst} to 2^8 t_{inst}	$1/(2\ t_{inst})$ to $1/(2^9\ t_{inst})$
2		8 t_{inst}	2^3 t_{inst} to 2^{11} t_{inst}	$1/(2^4\ t_{inst})$ to $1/(2^{12}\ t_{inst})$
3		16 t_{inst}	2^4 t_{inst} to 2^{12} t_{inst}	$1/(2^5\ t_{inst})$ to $1/(2^{13}\ t_{inst})$
4		64 t_{inst}	2^6 t_{inst} to 2^{14} t_{inst}	$1/(2^7\ t_{inst})$ to $1/(2^{15}\ t_{inst})$

t_{inst}: Instruction cycle (affected by clock mode, etc.)

Reference:

[Calculation example for the interval time and square wave frequency]

In this example, the main clock oscillation frequency (F_{CH}) is 12.5 MHz, the PWM compare register (COMR) value is set to "DD_H (221)", and the count clock cycle is set to 1 t_{inst}. In this case, the interval time and the frequency of the square wave output from the PWM pin (where the PWM timer operates continuously and the value of the COMR register is constant) are calculated as follows.

Assume that the main clock mode and its highest clock speed has been selected via the system clock control register (SYCC: SCS = 1, CS1 = 1, CS0 = 1, 1 instruction cycle = $4/F_{CH}$.

Interval time = (1 x 4/FCH) x (COMR register value + 1)
= (4/12.5 MHz) x (221 + 1)
= 71.04 µs

Output frequency = FCH / (1 x 8 x (COMR register value + 1))
= 12.5 MHz / (8 x (221 + 1))
≒ 7.04 kHz

■ PWM timer function

The PWM timer function has 8-bit resolution and can control the "H" and "L" widths of one cycle.

- As the resolution is 1/256, pulses can be output with duty ratios of between 0 and 99.6%.
- The cycle of the PWM wave can be selected from four types.
- The PWM timer can be used as a D/A converter by connecting the output to a low pass filter.

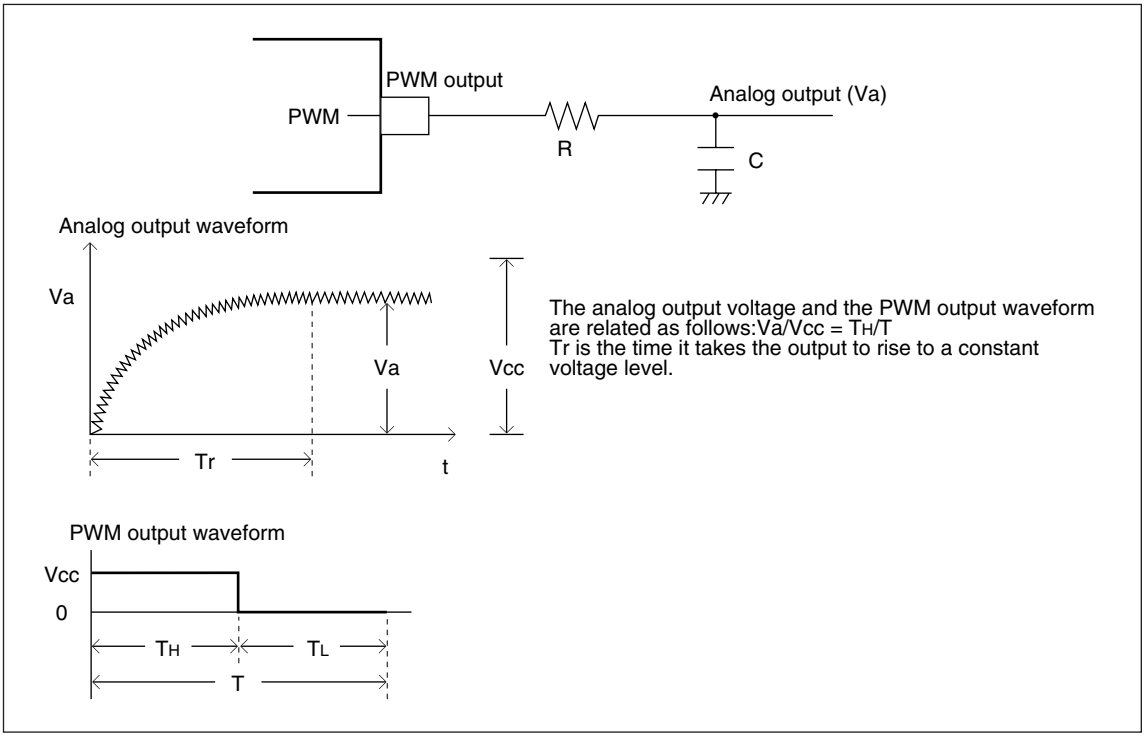
Table 8.1-2 "Available PWM wave cycle for PWM timer function" lists the available PWM wave cycles for the PWM timer function. Figure 8.1-1"Example D/A converter configuration using PWM output and low pass filter" shows an example D/A converter configuration.

Table 8.1-2 Available PWM wave cycle for PWM timer function

	1	2	3	4
	Internal count clock			
Count clock cycle	1 t_{inst}	8 t_{inst}	16 t_{inst}	64 t_{inst}
PWM wave cycle	2^8 t_{inst}	2^{11} t_{inst}	2^{12} t_{inst}	2^{14} t_{inst}

t_{inst}: Instruction cycle (affected by clock mode, etc.)

Figure 8.1-1 Example D/A converter configuration using PWM output and low pass filter



Reference:

Interrupt requests are not generated during operation of the PWM function.

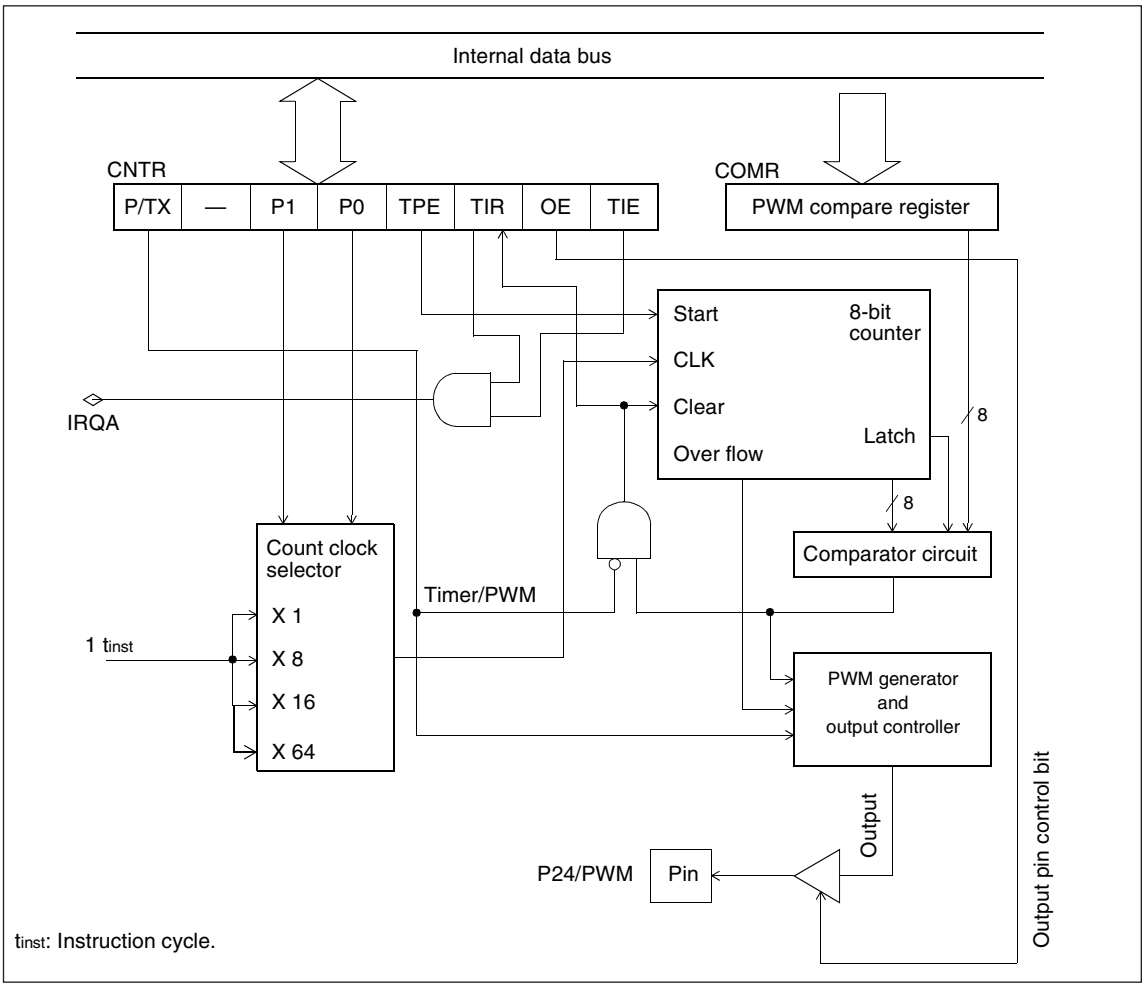
8.2 Block Diagram of 8-bit PWM Timer

The 8-bit PWM timer consists of the following six blocks:

- **Count clock selector**
- **8-bit counter**
- **Comparator circuit**
- **PWM generator and output controller**
- **PWM compare register (COMR)**
- **PWM control register (CNTR)**

■ Block diagram of 8-bit timer

Figure 8.2-1 Block diagram of 8-bit PWM timer



● Count clock selector

Selects a count-up clock for the 8-bit counter from the four internal count clocks

● 8-bit counter

The 8-bit counter counts-up on the count clock selected by the count clock selector.

● Comparator circuit

The comparator circuit has a latch to hold the COMR register value. The circuit latches the COMR register value when the 8-bit counter value is "00_H". The comparator circuit compares the 8-bit counter value with the latched COMR register value, and detects when a match occurs.

● PWM generator and output controller

When a match is detected during interval timer operation, an interrupt request is generated and, if the output pin control bit (CNTR: OE) is "1", the output controller inverts the output level of the PWM pin. At the same time, the 8-bit counter is cleared.

When a match is detected during PWM timer operation, the PWM generator changes the output level of the PWM pin from "H" to "L". The pin is set back to the "H" level when the next overflow occurs on the 8-bit counter.

● COMR register

The COMR register is used to set the value that is compared with the value of the 8-bit counter.

● CNTR register

The CNTR register is used to select the operating mode, enable or disable operation, set the count clock, control interrupts, and check the PWM status.

Setting the operation to PWM timer mode (P/TX = "0") disables clearing of the 8-bit counter and generation of interrupt requests IRQA when the comparator circuit detects a match.

8.3 Structure of 8-bit PWM Timer

This seciton describes the pin, pin block diagram, register source, and interrupt of the 8-bit PWM timer.

■ 8-bit PWM timer pin

The 8-bit PWM timer uses the P24/PWM pin. This pin can function either as a general-purpose I/O port (P24) or as the interval timer or PWM timer output.

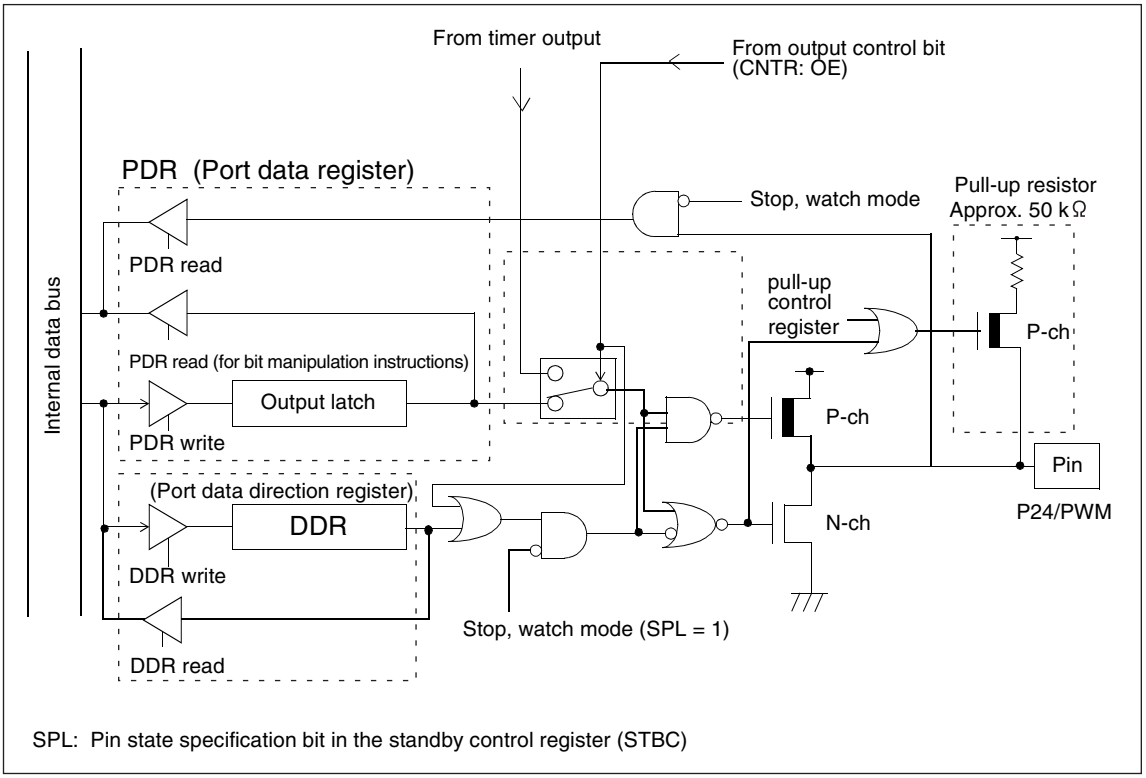
PWM:When the interval timer function is selected, the square waves are output to this pin.

When the PWM timer function is selected, the pin outputs the PWM wave.

Setting the output pin control bit (CNTR: OE) to "1" makes Pin P24/PWM the output-only pin for 8-Bit PWM Timer. Once this has been done, the pin performs its PWM function regardless of the state of the port data register output latch data (PDR2: Bit 4).

■ Block diagram of 8-bit PWM timer pin

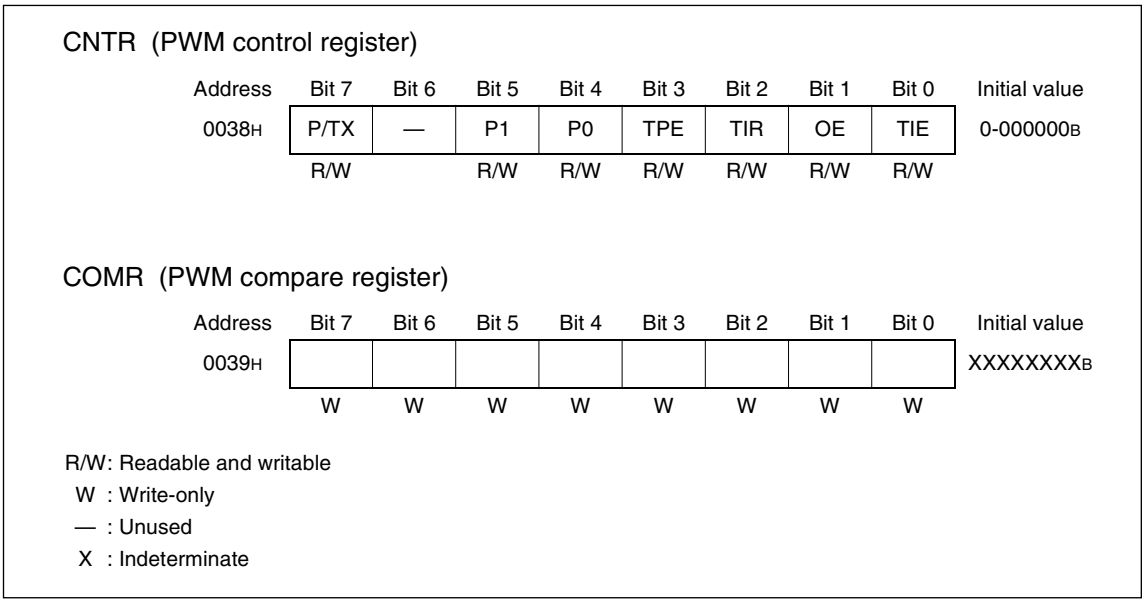
Figure 8.3-1 Block diagram of 8-bit PWM timer pin



SPL: Pin state specification bit in the standby control register (STBC)

■ 8-bit PWM timer registers

Figure 8.3-2 8-bit PWM1 timer registers



Note:

As the PWM compare register (COMR) is write-only, the bit manipulation instructions can not be used.

8.3.1 PWM Control Register (CNTR)

The PWM control register (CNTR) is used to select the operating mode of the 8-bit PWM timer (interval timer operation or PWM timer operation), enable or disable operation, select the count clock, control interrupts, and check the state of the 8-bit PMM timer.

■ PWM control register (CNTR)

Figure 8.3-3 PWM control register (CNTR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0038H	P/TX	—	P1	P0	TPE	TIR	OE	TIE	0-000000B
	R/W		R/W	R/W	R/W	R/W	R/W	R/W	

TIE	Interrupt request enable bit
0	Disables interrupt request output.
1	Enables interrupt request output.

OE	Output pin control bit
0	Functions as a general-purpose port (P24).
1	Functions as the interval timer/PWM timer output pin (PWM).

TIR	Interrupt request flag bit		
	Read		Write
	Interval timer function	PWM timer function	
0	Counter value and set value do not match.	No change	Clears this bit.
1	Counter value and set value match.		No effect. The bit does not change.

TPE	Counter operation enable bit
0	Stops count operation.
1	Starts count operation.

P1	P0	Clock selection bits	
0	0	Internal count clock	1 t_{inst} *
0	1		8 t_{inst} *
1	0		16t_{inst} *
1	1		64 t_{inst} *

P/TX	Operating mode selection bit
0	Operates as an interval timer.
1	Operates as a PWM timer.

* : t_{inst} : Instruction cycle

R/W : Readable and writable
— : Unused
X : Indeterminate
□ : Initial value

Table 8.3-1 PWM control register (CNTR) bit

Bit		Function
Bit 7	P/TX: Operating mode selection bit	• This bit switches between the interval timer function (P/TX = "0") and PWM timer function (P/TX = "1"). **Reference:** Write to this bit when the counter operation is stopped (TPE = "0"), interrupts are disabled (TIE = "0"), and the interrupt request flag bit is cleared (TIR = "0").
Bit 6	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 5 Bit 4	P1, P0: Clock selection bit	• These bits select the count clock for the interval timer function and PWM timer function. • These bits can select the count clock from four internal count clocks. **Reference:** Do not change P1 and P0 when the counter is operating (TPE = "1").
Bit 3	TPE: Counter operation enable bit	• This bit activates or stops operation of the PWM timer function and interval timer function. • Writing "1" to this bit starts the count operation. Writing "0" to this bit stops the count and clears the counter to "00H".
Bit 2	TIR: Interrupt request flag bit	• For the interval timer function: This bit is set to "1" when the counter and PWM compare register (COMR) values match. An interrupt request is output to the CPU when both this bit and the interrupt request enable bit (TIE) are "1". • For the PWM timer function: Interrupt requests are not generated. • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 1	OE: Output pin control bit	• The P24/PWM1 pin functions as a general-purpose port (P24) when this bit is set to "0", and a dedicated pin (PWM) when this bit is set to "1". • The PWM pin outputs a square wave when the interval timer function is selected and a PWM waveform when the PWM timer function is selected.
Bit 0	TIE: Interrupt request enable bit	• This bit enables or disables interrupt request output to the CPU. An interrupt request is output when both this bit and the interrupt request flag bit (TIR) are "1".

8.3.2 PWM Compare Register (COMR)

The PWM compare register (COMR) sets the interval time for the interval timer function. The register value sets the “H” width of the pulse for the PWM timer function.

■ PWM compare register (COMR)

Figure 8.3-4 "PWM compare register (COMR)" shows the bit structure of the PWM compare register.

As the register is write-only, bit manipulation instructions cannot be used.

Figure 8.3-4 PWM compare register (COMR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0039H									XXXXXXXXB
	W	W	W	W	W	W	W	W	

W: Write-only
X: Indeterminate

● Interval timer operation

This register is used to set the value to be compared with the counter value. The register specifies the interval time.

The counter is cleared when the counter value matches the value set in this register, and the interrupt request flag bit is set to "1" (CNTR : TIR = "1").

If data is written to the COMR register during counter operation, the new value applies from the next cycle (after the next match is detected).

Reference:

The COMR setting for interval timer operation can be calculated using the following formula.
The instruction cycle time is affected by the clock mode, and the speed-shift selection.
COMR register value = interval time/(count clock cycle x instruction speed-shift cycle) - 1

● PWM timer operation

This register is used to set the value to be compared with the counter value. The register therefore sets the "H" width of the pulse.

The PWM pin outputs an "H" level until the counter value matches the value set in this register. From the match until the counter value overflows, the PWM pin outputs an "L" level.

If data is written to the COMR register during counter operation, the new value applies from the next cycle (after the next overflow).

In PWM timer operation, the COMR setting and the PWM cycle time can be calculated using the following formulas. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)
COMR register value = duty ratio (%) x 256
PWM wave cycle = count clock cycle x instruction cycle x 256

8.4 8-bit PWM Timer Interrupts

The 8-bit PWM timer can generate an interrupt request when a match is detected between the counter value and PWM compare register value for the interval timer function. Interrupt requests are not generated for the PWM timer function.
8-bit PWM timer generates the IRQA as an interrupt request.

■ Interrupts for interval timer function

The counter value is counted-up from "00_H" on the selected count clock. When the counter value matches the PWM compare register (COMR) value, the interrupt request flag bit (CNTR: TIR) is set to "1".

At this time, an interrupt request to the CPU is generated if the interrupt request enable bit is enabled (CNTR: TIE = "1"). Write "0" to the TIR bit in the interrupt processing routine to clear the interrupt request.

The TIR bit is set to "1" when the counter value matches the set value, regardless of the value of the TIE bit.

Reference:

The TIR bit is not set if the counter is stopped (CNTR: TPE = "0") at the same time as the counter value matches the COMR register value.

An interrupt request is generated immediately if the TIR bit is "1" when the TIE bit is changed from disabled to enabled ("0" --> "1").

■ Registers and vector tables for 8-bit PWM timer interrupts

Table 8.4-1 Registers and vector tables for 8-bit PWM timer interrupts

	Interrupt	Interrupt level setting register			Vector table address	
		Register	Setting bits		Upper	Lower
8-bit PWM timer	IRQA	ILR3 ($007D_H$)	LA1 (Bit 5)	LA0 (Bit 4)	$FFE6_H$	$FFE7_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupt operation.

8.5 Operation of Interval Timer Function

This seciton describes the operation of the interval timer function of the 8-bit PWM timer.

■ Operation of interval timer function

Figure 8.5-1 "Interval timer function settings" shows the settings required to operate as an interval timer function.

Figure 8.5-1 Interval timer function settings

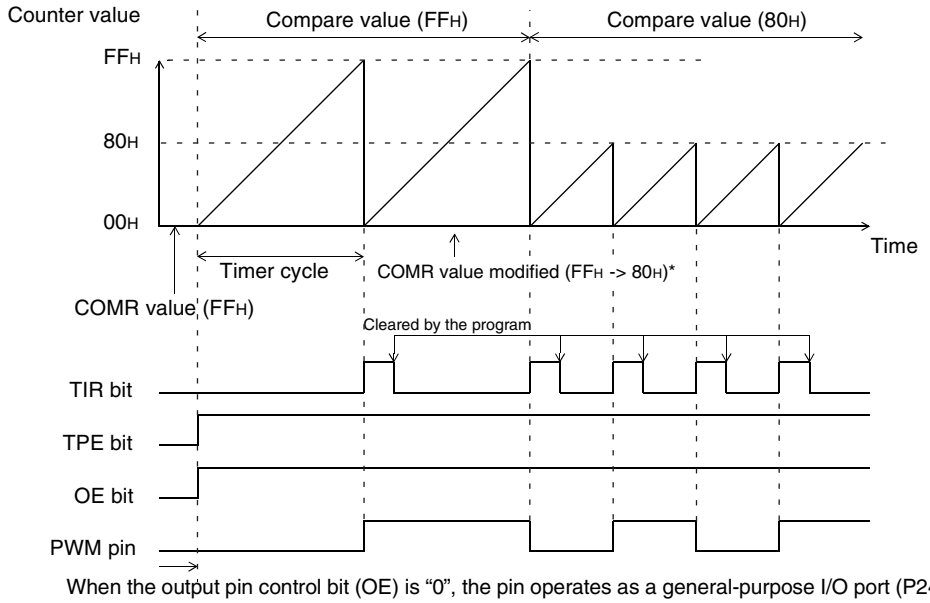
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CNTR	P/TX	—	P1	P0	TPE	TIR	OE	TIE
	0		◎	◎	1	◎	◎	◎
COMR	Sets interval time (compare value).							

◎: Used bit
1 : Set “1”.
0 : Set “0”.

On activation, the counter starts counting-up from "00_H" on the rising edge of the selected count clock. When the counter value matches the value set in the COMR register (compare value), the PWM timer inverts the level of the output pin (PWM) on the next rising edge of the count clock, clears the counter, sets the interrupt request flag bit (CNTR: TIR = "1"), and restarts counting from "00_H".

Figure 8.5-2 "Operation of 8-bit PWM timer" shows the operation of the 8-bit PWM timer.

Figure 8.5-2 Operation of 8-bit PWM timer



When the output pin control bit (OE) is “0”, the pin operates as a general-purpose I/O port (P24).

*: If the PWM compare register (COMR) value is modified during counter operation, the new value is used from the next cycle.

Note:

Do not change the count clock cycle (CNTR: P1, P0) during operation of the interval timer function (CNTR: TPE = "1").

References:

Setting the COMR register value to "00_H" causes the PWM pin output to be inverted with the cycle of the selected count clock.

When the counter is stopped (CNTR: TPE = "0") while the interval timer function is selected, the PWM pin outputs an "L" level.

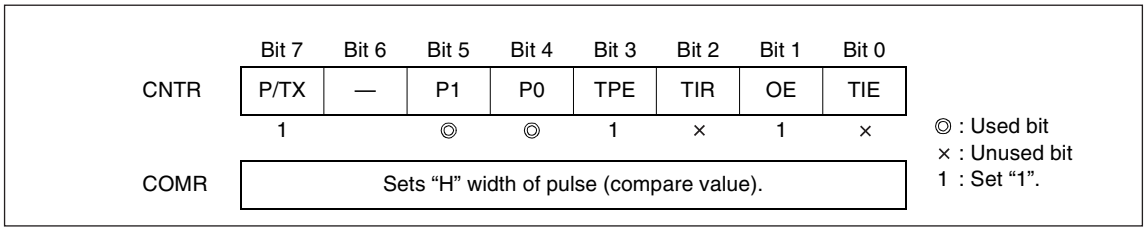
8.6 Operation of PWM Timer Function

This section describes the operation of the PWM timer function of the 8-bit PWM timer.

■ Operation of PWM timer function

Figure 8.6-1 "PWM timer function settings" shows the settings required to operate as the PWM timer function.

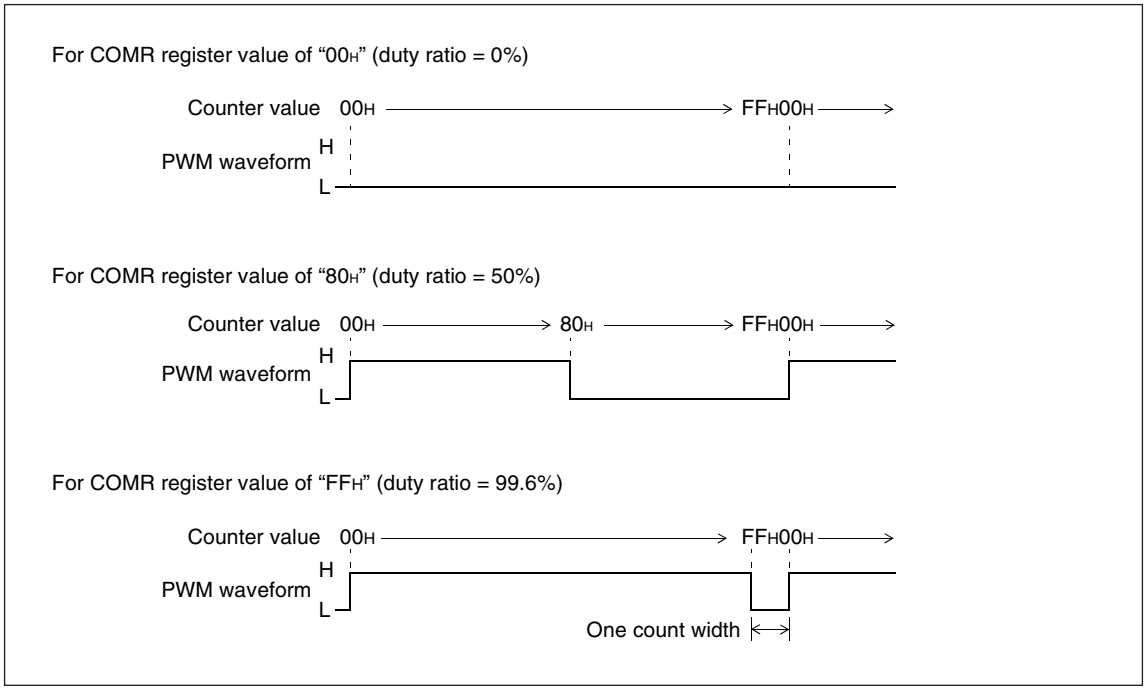
Figure 8.6-1 PWM timer function settings



On activation, the counter starts counting-up from "00_H" on the rising edge of the selected count clock. The PWM pin (PWM) outputs (PWM waveform) an "H" level until the counter value matches the value set in the COMR register. From the match until the counter value overflows (FF_H --> 00_H), the PWM pin outputs an "L" level.

Figure 8.6-2 "Example of PWM waveform output (PWM pin)" shows the PWM waveforms output from the PWM pin.

Figure 8.6-2 Example of PWM waveform output (PWM pin)



Note:

Do not change the count clock cycle (CNTR: P1, P0) during operation of the PWM timer function (CNTR: TPE = "1").

Refernce:

When the PWM timer function is selected, the PWM pin maintains its existing level when the counter is stopped (CNTR: TPE = "0").

8.7 States in Each Mode during 8-bit PWM Timer Operation

This section describes the operation of the 8-bit PWM timer when the device changes to sleep or stop mode or an operation halt request occurs during operation

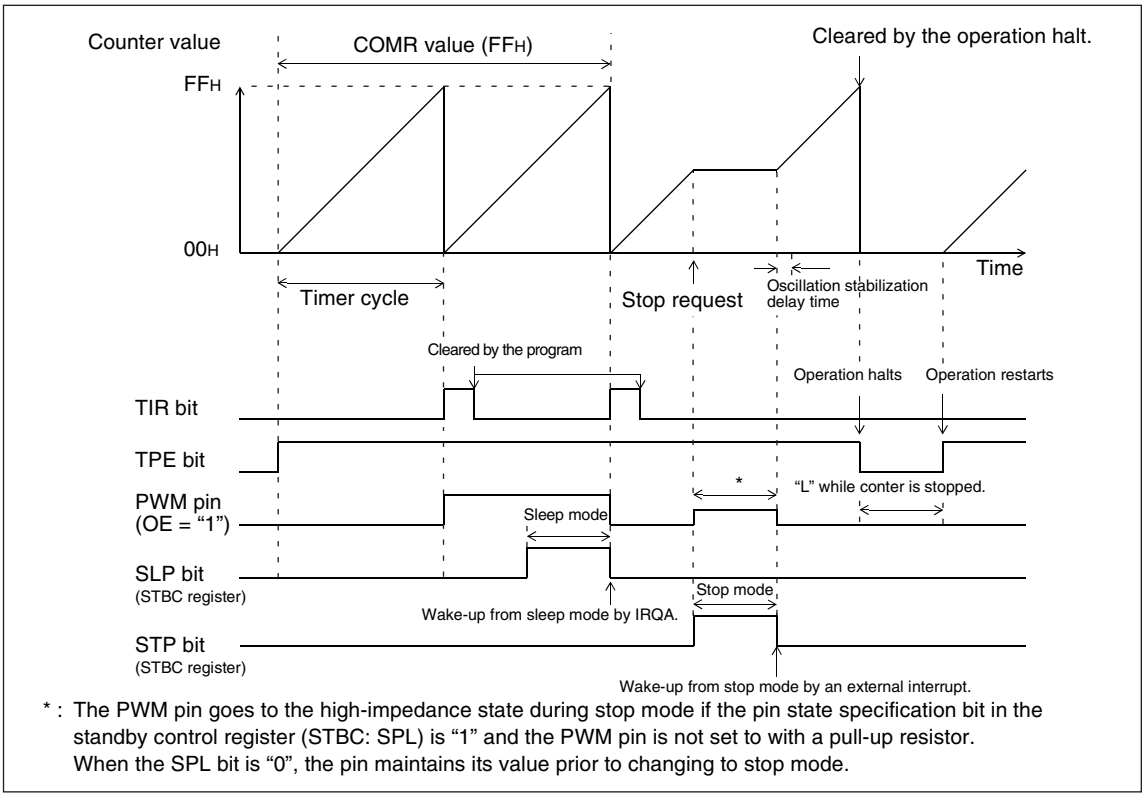
■ Operation during standby mode or operation halt

Figure 8.7-1 "Counter operation during standby modes or operation halt (for interval timer function)" and Figure 8.7-2 "Operation during standby modes or operation halt (for PWM timer function)" show the counter value states when the device changes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or PWM timer function.

The counter halts and maintains its current value when the device changes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. Therefore, the first interval time or PWM wave cycle does not match the set value. Always initialize the 8-bit PWM timer after wake-up from stop mode.

● For interval timer function

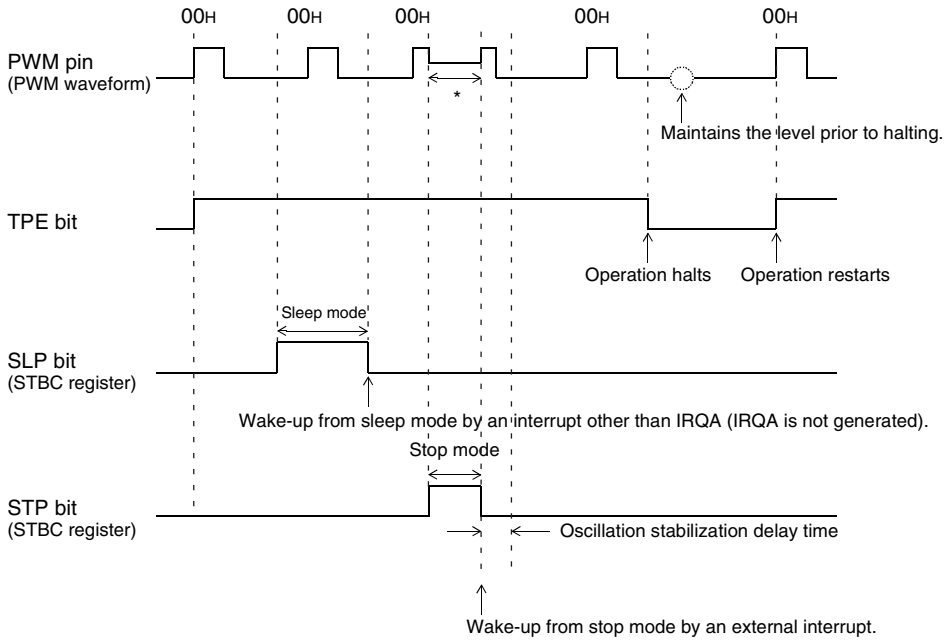
Figure 8.7-1 Counter operation during standby modes or operation halt (for interval timer function)



* : The PWM pin goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is "1" and the PWM pin is not set to with a pull-up resistor. When the SPL bit is "0", the pin maintains its value prior to changing to stop mode.

● For PWM timer function

Figure 8.7-2 Operation during standby modes or operation halt (for PWM timer function)



* : The PWM pin (PWM) goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is “1” and the PWM pin is not set to with a pull-up resistor.
When the SPL bit is “0”, the pin maintains its value prior to changing to stop mode.

8.8 Notes on Using 8-bit PWM Timer

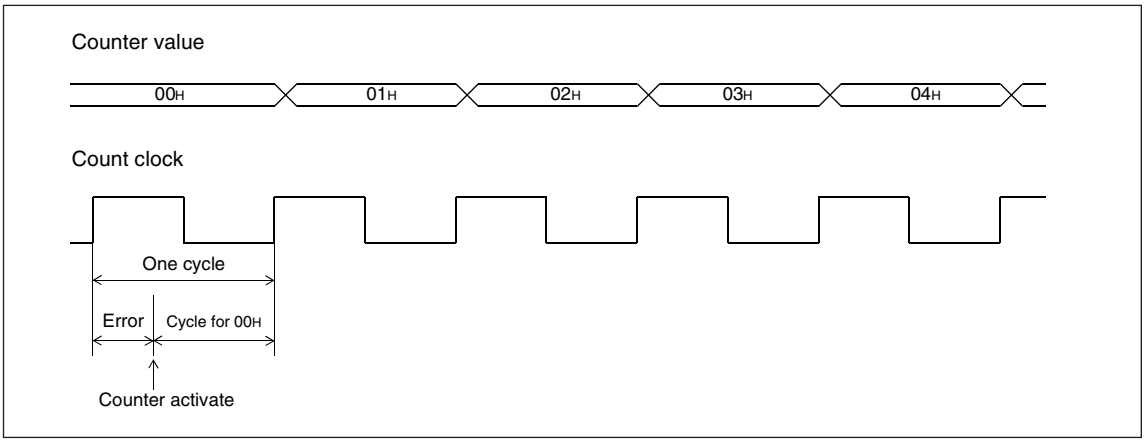
This section lists points to note when using the 8-bit PWM timer.

■ Notes on using 8-bit PWM timer

● Error

Activating the counter by program is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter until a match with the PWM compare register (COMR) is detected may be shorter than the theoretical time by a maximum of one cycle of the count clock. Figure 8.8-1 "Error on starting counter operation" shows the error that occurs on starting counter operation.

Figure 8.8-1 Error on starting counter operation



● Notes on setting by program

- Do not change the count clock cycle (CNTR: P1, P0) when the interval timer function or PWM timer function is operating (CNTR: TPE = "1").
- Stop the counter (CNTR: TPE = "0"), disable interrupts (TIE = "0"), and clear the interrupt request flag (TIR = "0") before switching between the interval timer function and PWM timer function (CNTR: P/TX).
- Interrupt processing cannot return if the interrupt request flag bit (CNTR: TIR) is "1" and the interrupt request enable bit is enabled (CNTR: TIE = "1"). Always clear the TIR bit.
- The TIR bit is not set if the counter is disabled (TPE = "0") at the same time as the counter and COMR register values match.

8.9 Program Example for 8-bit PWM Timer

This section gives program examples for the 8-bit PWM timer.

■ Program example for interval timer function

● Processing description

- Generates repeated interval timer interrupts at 5 ms intervals.
- Outputs a square wave to the PWM pin that inverts after each interval time.
- With a main clock master oscillation F_{CH} of 12.5 MHz, and the highest speed clock selected by the speed-shift function (1 instruction cycle time = $4/F_{CH}$), the COMR register is set for an interval time of approximately 5 ms. (An internal clock period of 64 t_{inst} is selected as the count clock.) The COMR register setting is calculated as follows:

  COMR register value = 5 ms/(64 x 4/12.5 MHz) - 1 = 243.0 ($0F3_H$)

● Coding example

```
CNTR    EQU    0038H          ; Address of the PWM control register
COMR    EQU    0039H          ; Address of the PWM compare register

TPE     EQU    CNTR:3         ; Define the counter operation enable bit.
TIR     EQU    CNTR:2         ; Define the interrupt request flag bit.

ILR3    EQU    007DH          ; Address of the interrupt level setting register 3

INT_V   DSEG   ABS            ; [DATA SEGMENT]
        ORG    0FFE6H
IRQA    DW WARI               ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------------
        CSEG                  ; [CODE SEGMENT]
                              ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                  ; Disable interrupts.
        CLRB   TPE            ; Stop counter operation.
        MOV    ILR3,#11011111B ; Set interrupt level (level 1).
        MOV    COMR,#0F3H     ; Value compared with the counter value (interval time)
        MOV    CNTR,#00111011B ; Operate interval timer, select 64 tinst,
                                start counter operation, clear interrupt request flag,
                                enable PWM pin output, enable interrupt request output.
        SETI                  ; Enable interrupts.
         :
;-----Interrupt program--------------------------------------------------------------
WARI    CLRB   TIR            ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T            ; Save A and T.
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T            ; Restore A and T.
        POPW   A
        RETI
        ENDS
;-----------------------------------------------------------------------------------
        END
```

■ Program example for PWM timer function

● Processing description

- Generates a PWM wave with a duty ratio of 50%. Then, changes the duty ratio to 25%.
- Does not generate interrupts.
- For a 12.5 MHz source oscillation, (F_{CH}), and the highest speed clock selected by the speed-shift function (1 instruction cycle time = $4/F_{CH}$), selecting the interval 16 t_{inst} count clock gives a PWM wave cycle of 16 x 4/12.5 MHz x 256 ≒ 1.31 ms.
- The following shows the COMR register value required for a duty ratio of 50%:
  COMR register value = 50/100 x 256 = 128 (080_H)

● Coding example

```
CNTR    EQU    0038H           ; Address of the PWM control register
COMR    EQU    0039H           ; Address of the PWM compare register

TPE     EQU    CNTR:3          ; Define the counter operation enable bit.
;-----Main program-------------------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
         :
        CLRB   TPE             ; Stop counter operation.
        MOV    COMR,#80H       ; Set "H" width of pulse. Duty ratio = 50%
        MOV    CNTR,#10101010B ; Operate PWM timer, select 16 tinst,
                                 start counter operation, clear interrupt request flag,
                                 enable PWM pin output, and disable interrupt request
                                 output.
         :
         :
        MOV    COMR,#40H       ; Change the duty ratio to 25% (effective from the next
                                 PWM wave cycle).
         :
        ENDS
;-----------------------------------------------------------------------------------------
        END
```

CHAPTER 9

PULSE WIDTH COUNT TIMER (PWC)

This chapter describes the functions and operation of the pulse width count timer (PWC).

9.1 Overview of Pulse Width Count Timer

The pulse width count timer (PWC) can be selected to function as either an interval timer or the pulse width measurement. The interval timer function counts down in sync with one of three internal clocks . The pulse width measurement function measures the width of pulses input to an external pin.

■ Interval timer function

The interval timer function generates repeated interrupts at variable time intervals.

- The interval timer can operate with a cycle among 1 and 2^8 times the internal count clock cycle.
- The internal count clock can be selected from three different clocks.
- Two operating modes are available: reload timer mode (continuous operation) and one-shot mode (one-time operation).

Table 9.1-1 "Interval time range" lists the available interval time and square wave output ranges.

Table 9.1-1 Interval times and Square wave output range

Internal count clock cycle	Interval time	Square wave output(Hz)
1 t_{inst}	1 t_{inst} to 2^8 t_{inst}	$1/(2\ t_{inst})$ to $1/(2^9\ t_{inst})$
4 t_{inst}	2^2 t_{inst} to 2^{10} t_{inst}	$1/(2^3\ t_{inst})$ to $1/(2^{11}\ t_{inst})$
32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}	$1/(2^6\ t_{inst})$ to $1/(2^{14}\ t_{inst})$

t_{inst}: Instruction cycle (affected by clock mode, etc.)

The following shows an example of the interval time.

For an 12.5 MHz source oscillation (F_C), a PWC reload buffer register (PLBR) value of "DD_H (221)", and a count clock cycle of one instruction cycle, the interval time is calculated as follows:

Interval time = (1 x 4/F_{CH}) x (PLBR register value)
= (4/12.5 MHz) x 221
= 70.7 µs

PLBR register value of "00_H" is assumed as 256.

■ Pulse width measurement function

The pulse width measurement function can measure the "H" width, "L" width, and one-cycle width of pulses input from an external pin (PWC pin).

- The PWC can perform continuous pulse width measurement.
- The measurement speed (internal count clock) can be selected from three different speeds.
- The width of long input pulses can be measured using an interrupt processing routine.

Table 9.1-2 "Available pulse width measured by pulse width measurement function" lists the available pulse widths measured by the pulse width measurement function.

Table 9.1-2 Available pulse width measured by pulse width measurement function

Internal count clock cycle	Interval time
1 t_{inst}	1 t_{inst} to 2^8 t_{inst}
4 t_{inst}	2^2 t_{inst} to 2^{10} t_{inst}
32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}

t_{inst}: Instruction cycle (affected by clock mode, etc.)

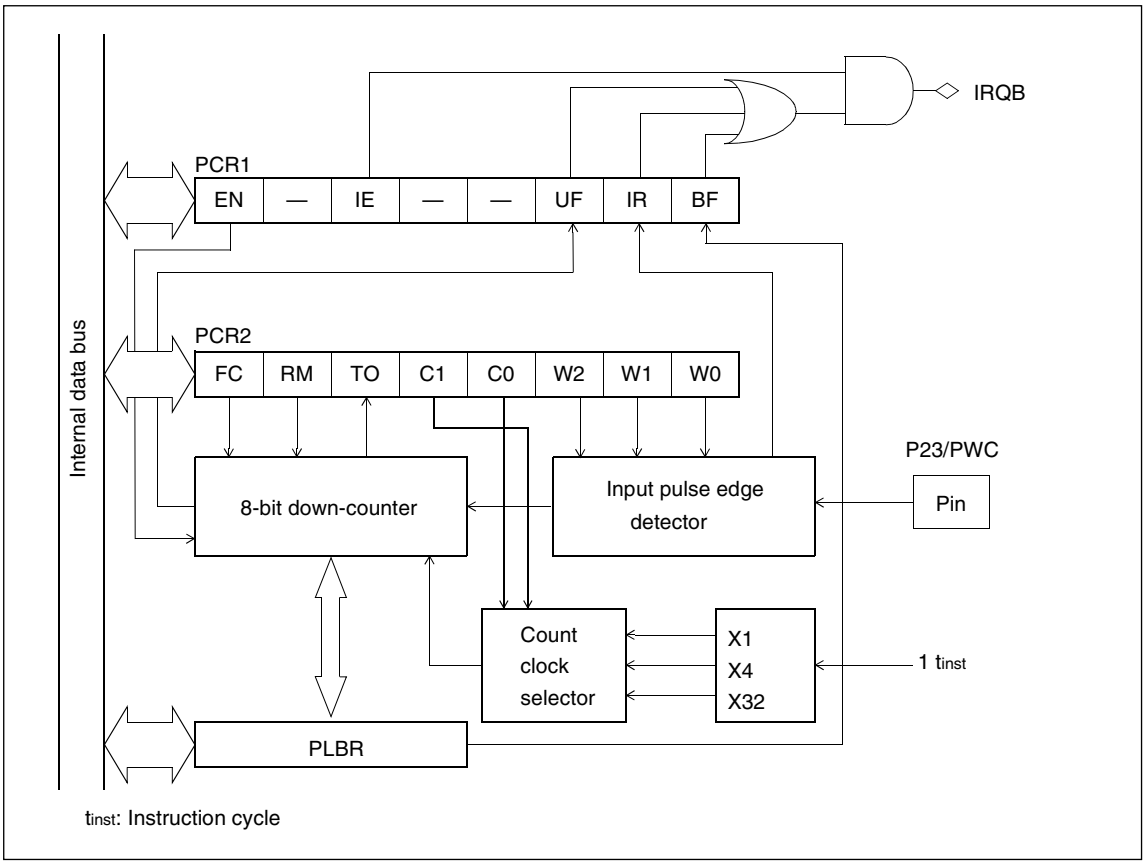
9.2 Block Diagram of Pulse Width Count Timer

The pulse width count timer consists of the following nine blocks:

- **Count clock selector**
- **8-bit down counter**
- **Input pulse edge detector**
- **PWC reload buffer register (PLBR)**
- **PWC pulse width control register 1 (PCR1)**
- **PWC pulse width control register 2 (PCR2)**

■ Block diagram of pulse width count timer

Figure 9.2-1 Block diagram of pulse width count timer



- Count clock selector

  Count clock selector selects out three kinds of iternal count clock and the selected clock is used for 8-bit down counter.

- 8-bit down counter

  The 8-bit down counter starts to counting from the value set in the PWC reload buffer register (PLBR) when operating as an interval timer, and from FF_H when performing pulse width measurement. When an underflow (00_H --> FF_H) occurs, the counter inverts the timer output bit (PCR2: TO).

- Input pulse edge detector

  Operates when the pulse width measurement function is selected, and starts or stops the 8-bit down counter when an edge input from the PWC pin matches the edge specified by the PWC pulse width control register 2 (PCR2).

- PLBR register

  When operating in reload timer mode of the interval timer function, the PLBR register value is re-loaded to the counter and the count continues whenever a counter value underflow (00_H --> FF_H) occurs.

  When performing pulse width measurement, the value of the 8-bit down counter is transferred to the PLBR register when measurement completes.

- PCR1 and PCR2 register

  These registers are used to select the function, set operating conditions, enable or disable operation, control interrupts, and to check the PWC status.

9.3 Structure of Pulse Width Count Timer

This section describes the pins, pin block diagram, registers, and interrupt source of the pulse width count timer.

■ Pulse width count timer pin

The pulse width count timer uses the P23/PWC pin. This pin can function either as CMOS general-purpose I/O port (P23), or as the measured pulse input (PWC).

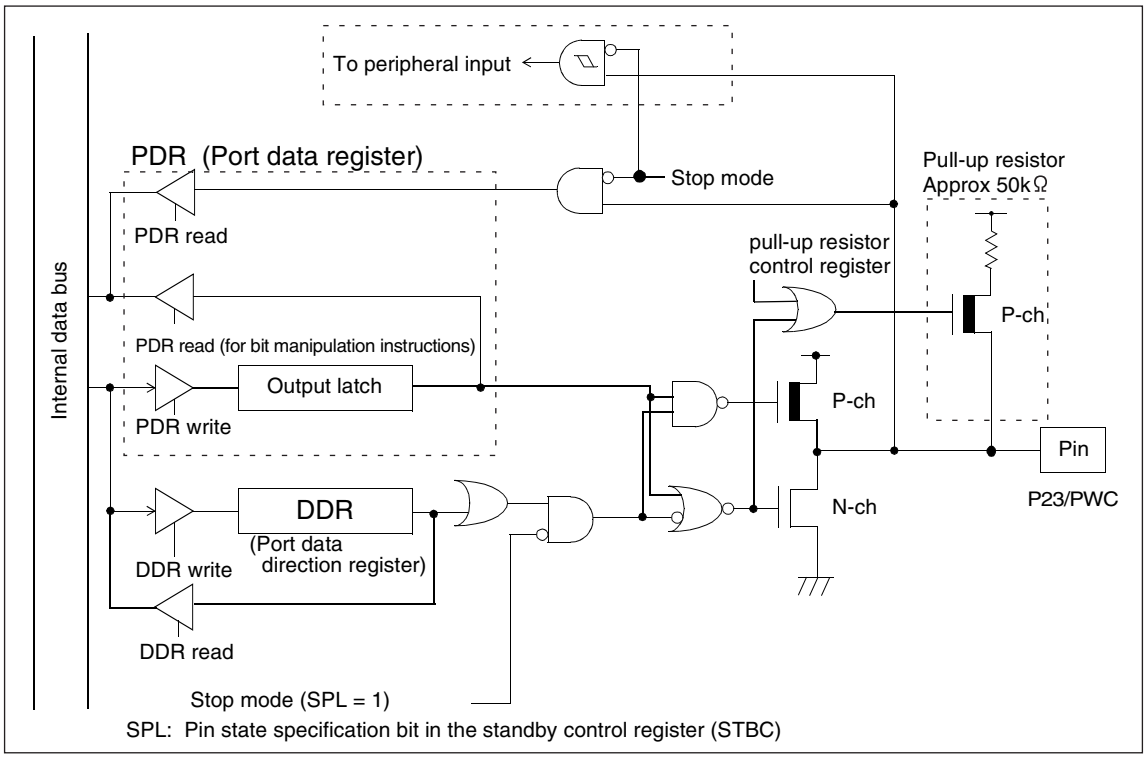
PWC:

The pulse width measurement function measures the pulse widths input to this pin.

Set the pin as an input port in the port data direction register (DDR2: bit 3 = "0") when using as the PWC pin for the pulse width measurement function.

■ Block diagram of pulse width count timer pin

Figure 9.3-1 Block diagram of pulse width count timer pin



Reference:

When pull-up resistor is selected in pull-up resistor control register in stop mode (SPL=1), the states of these pins are in "H" level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

■ Pulse width count timer registers

Figure 9.3-2 Pulse width count timer registers

PCR1 (PWC pulse width control register 1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0034H	EN	—	IE	—	—	UF	IR	BF	0-0--000B
	R/W		R/W			R/W	R/W	R	

PCR2 (PWC pulse width control register 2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0035H	FC	RM	TO	C1	C0	W2	W1	W0	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

PLBR (PWC reload buffer register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value	
0036H									XXXXXXXXB	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		.....For the interval timer function
	R	R	R	R	R	R	R	R		.....For the pulse width measurement function

R/W: Readable and writable
R : Read-only
— : Unused
X : Indeterminate

■ Pulse width count timer interrupt source

IRQB:

For both the interval timer and pulse width measurement function, the PWC generates an interrupt request if interrupt request output is enabled (PCR1: IE = "1") when the counter value underflows (00_H --> FF_H).

Also, for the pulse width measurement function, the PWC also generates an interrupt request for the pulse width measurement function if interrupt request output is enabled (PCR1: IE = "1") when pulse width measurement completes or a pulse width measurement value remains in the PLBR register.

9.3.1 PWC Pulse Width Control Register 1 (PCR1)

The PWC pulse width control register 1 (PCR1) is used to enable or disable functions, control interrupts and check the state of the pulse width count timer.

■ PWC pulse width control register 1 (PCR1)

Figure 9.3-3 PWC pulse width control register 1 (PCR1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0034H	EN	—	IE	—	—	UF	IR	BF	0-0--000B
	R/W		R/W			R/W	R/W	R	

BF	Buffer full flag bit
0	No pulse width measurement value
1	Pulse width measurement value

IR	Measurement completion interrupt request flag bit	
	Read	Write
0	Pulse width measurement not complete.	Clears this bit.
1	Pulse width measurement complete.	No effect. The bit does not change.

UF	Underflow (00H -> FFH) interrupt request flag bit	
	Read	Write
0	No underflow (00H -> FFH) on counter	Clears this bit.
1	Underflow (00H -> FFH) on counter	No effect. The bit does not change.

The UF, IR, and BF bits are interrupt request flag bits.

IE	Interrupt request enable bit
0	Disables interrupt request output.
1	Enables interrupt request output.

EN	Counter operation enable bit
0	Disables/stops operation.
1	Enables/starts operation.

R/W: Readable and writable
R : Read-only
— : Unused
□ : Initial value

Table 9.3-1 PWC pulse width control register 1 (PCR1) bits

Bit		Function
Bit 7	EN: Counter operation enable bit	• For the interval timer function: Writing "1" to this bit starts the counter to counting-down from the PWC reload buffer register (PLBR) value. Writing "0" to this bit stops the counter operation. • For the pulse width measurement function: Writing "1" to this bit enables measurement. The counter starts to counting-down from "FF_H" on detection of the specified edge on the measurement pulse. Writing "0" to this bit stops the counter operation. **Note:** If operation is disabled (EN = "0") during measurement in pulse width measurement mode, the counter stops but the value is not transferred to the PLBR register. Restarting operation (EN = "1") sets the counter value to "FF_H" then enables operation.
Bit 6	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 5	IE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and one or more of the interrupt request flag bits (UF, IR, and BF) are "1".
Bit 4 Bit 3	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 2	UF: Underflow (00_H --> FF_H) interrupt request flag bit	• This bit is set to "1" when the counter underflow (00_H --> FF_H) occurs. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value. **Notes:** • When the interval timer function is active, the PWC inverts the timer output bit (PCR2: TO) if the counter underflow (00_H --> FF_H) occurs. In reload timer mode, counting-down continues from the PLBR register value. In one-shot timer mode, the counter operation automatically stops (EN = "0"). • If the counter underflow (00_H --> FF_H) occurs while measuring a long input pulse in the pulse width measurement function, this bit is set to "1" and counter operation continues.
Bit 1	IR: Measurement completion interrupt request flag bit	• For the pulse width measurement function: This bit is set to "1" when the pulse width measurement is completed. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value. • For the interval timer function: The bit has no meaning.
Bit 0	BF: Buffer full flag bit	• For the pulse width measurement function: This bit is an interrupt request flag and is set to "1" when a measurement value is present in the PLBR register. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". This bit is set to "1" when pulse width measurement completes and cleared to "0" when the measurement value is read from the PLBR register. This bit is read-only. The write value has no meaning and has no effect on the operation. • For the interval timer function: This bit has no meaning.

9.3.2 PWC Pulse Width Control Register 2 (PCR2)

The PWC pulse width control register 2 (PCR2) is used to select the operating mode (pulse width measurement or interval timer operation, etc.), select the count clock, set the measured pulse (measurement edges), and check the timer output state of the pulse width count timer.

■ PWC pulse width control register 2 (PCR2)

Figure 9.3-4 PWC pulse width control register 2 (PCR2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0035H	FC	RM	TO	C1	C0	W2	W1	W0	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

W2	W1	W0	Measured pulse selection bits Only applies to the pulse width measurement function (FC = "1")
0	0	0	"H" pulse (rising edge to falling edge)
0	0	1	"L" pulse (falling edge to rising edge)
0	1	0	Rising edge to rising edge (one cycle)
0	1	1	Falling edge to falling edge (one cycle)
1	1	0	High level and rising to rising

C1	C0	Count clock selection bits
0	0	1 tinst
0	1	4 tinst
1	0	32 tinst
1	1	Do not use this setting

tinst: Instruction cycle

TO	Timer output bit	
	Read	Write
0	Reads the current output value.	Can be used to set the output value when the counter is stopped.
1		

RM	Reload mode selection bit Only applies to the interval timer function (FC = "0").
0	Reload timer mode
1	One-shot timer mode

FC	Operating mode selection bit
0	Operates as an interval timer function.
1	Operates as a pulse width measurement function.

R/W: Readable and writable
: Initial value

Table 9.3-2 PWC pulse width control register 2 (PCR2) bits

Bit		Function
Bit 7	FC: Operating mode selection bit	• This bit switches between the interval timer function (FC = "0") and pulse width measurement function (FC = "1"). **Note:** When using the pulse width measurement function (FC = "1"), set the P23/PWC pin as an input port.
Bit 6	RM: Reload mode selection bit	• For the interval timer function: This bit selects reload timer mode (RM = "0") or one-shot timer mode (RM = "1"). • For the pulse width measurement function: This bit has no meaning.
Bit 5	TO: Timer output bit	• The value of this bit is inverted each time a counter value underflow (00_H --> FF_H) occurs. • By counting the number of times this bit is inverted (number of underflow (00_H --> FF_H) occurs), pulse widths longer than 2^8 × the cycle of the selected count clock can be measured.
Bit 4 Bit 3	C1, C0: Count clock selection bits	• These bits select the count clock for the interval timer function and pulse width measurement function. • Three internal count clocks can be selected.
Bit 2 Bit 1 Bit 0	W2, W1, W0: Measured pulse selection bits	• For the pulse width measurement function: These bits select which pulse edges to use as the start and end conditions for pulse measurement. Five types of pulse width or cycle can be selected. • For the interval timer function: These bits have no meaning. Do not use these setting with invalid values.

Note:

Do not modify the PCR2 register while the counter is operating (PCR1: EN = "1").

9.3.3 PWC Reload Buffer Register (PLBR)

The PWC reload buffer register (PLBR) functions as a reload register for the interval timer function and as a measurement value storage register for the pulse width measurement function.

■ PWC reload buffer register (PLBR)

Figure 9.3-5 "PWC reload buffer register (PLBR)" shows the bit structure of the PWC reload buffer register.

Figure 9.3-5 PWC reload buffer register (PLBR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0036H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	.....For the interval timer function
	R	R	R	R	R	R	R	R	.....For the pulse width measurement function

R/W: Readable and writable
R : Read-only
X : Indeterminate

● For interval timer function

The register functions as a reload register, specifying the interval time.

The counter starts to counting-down from the set value written in this register when counter operation is enabled (PCR1: EN = "1").

In reload timer mode, the PLBR register value is reloaded to the counter and the counter continues counting-down when a counter value underflows (00_H --> FF_H). If a value is written to the PLBR register during counter operation, the new value applies from the next time the counter is reloaded due to an underflow (00_H --> FF_H).

Reference:

The setting value of the PLBR register for the interval timer function is calculated as follows:

PLBR register value = interval time/(count clock cycle x instruction cycle)

● For pulse width measurement function

The register is used to store the pulse width measurement value.

The counter value is transferred to this register when pulse width measurement completes on detection of the edge specified for measurement completion.

At this time, the buffer full flag bit (PCR1: BF) and the measurement completion interrupt request flag bit (PCR1: IR) are set to "1". Reading this register clears the BF bit to "0".

The register is read-only if the pulse width measurement function is selected.

Reference:

The pulse width for the pulse width measurement function is calculated based on the PLBR register value as follows:

Pulse width = (256 - PLBR register value) x count clock cycle x instruction cycle

9.4 Pulse Width Count Timer Interrupts

The pulse width count timer has the following two interrupts:

- **Counter value underflow (00_H --> FF_H) for the interval timer function**
- **Measurement completion and buffer full for the pulse width measurement function**

■ Interrupt for the interval timer function

The counter counts down from the set value on the selected internal count clock. When an underflow occurs, the underflow (00_H --> FF_H) interrupt request flag bit (PCR1: UF) is set to "1". At this time, an interrupt request (IRQB) to the CPU is generated if the interrupt request enable bit is enabled (PCR1: IE = "1"). Write "0" to the UF bit in the interrupt processing routine to clear the interrupt request.

References:

- The UF bit is not set if the counter is stopped (PCR1: EN = "0") at the same time as the counter value underflows (00_H --> FF_H).
- An interrupt request is generated immediately if the UF bit is "1" when the IE bit is changed from disabled to enabled ("0" --> "1").

■ Interrupt for pulse width measurement function

When the specified measurement completion edge is detected, the measurement completion interrupt request flag bit (PCR1: IR) and the buffer full flag bit (PCR1: BF) are set to "1". Also, when a counter underflow (00_H --> FF_H) occurs due to measurement of a long pulse, the UF bit is set to "1". At this time, an interrupt request (IRQB) to the CPU is generated if the interrupt request enable bit is enabled (PCR1: IE = "1"). Write "0" to the IR and UF bit in the interrupt processing routine to clear the interrupt request. Also read the PWC reload buffer register (PLBR) to clear the BF bit to "0".

References:

- The IR and BF bit are not set if the counter is stopped (PCR1: EN = "0") at the same time as the specified measurement completion edge is detected.
- An interrupt request is generated immediately if the IR, BF, or UF bit is "1" when the IE bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for pulse width count timer interrupt

Table 9.4-1 Register and vector table for pulse width count timer interrupt

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQB	ILR3 ($007D_H$)	LB1 (Bit 7)	LB0 (Bit 6)	$FFE4_H$	$FFE5_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupt.

9.5 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the pulse width count timer.

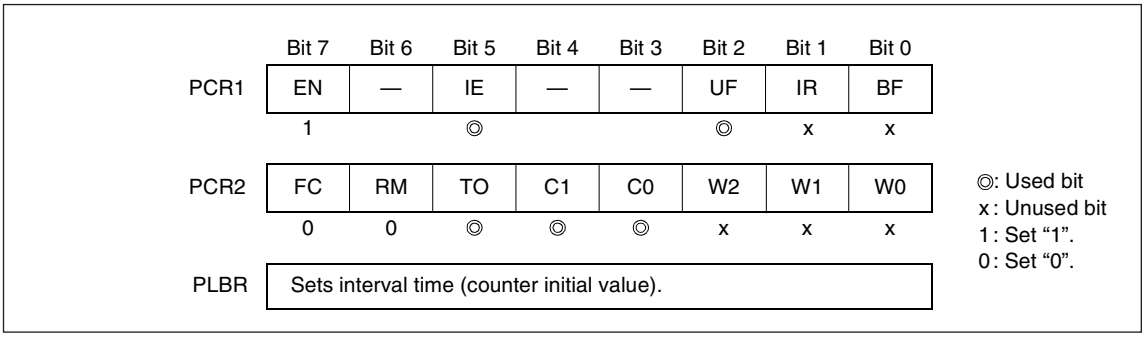
■ Operation of interval timer function

The interval timer function can operate as a continuous timer (reload timer mode), or as a timer that operates for one timer-cycle and then stops (one-shot mode).

● Reload timer mode

Figure 9.5-1 "Interval timer function (reload timer mode) settings" shows the settings required to operate in reload timer mode.

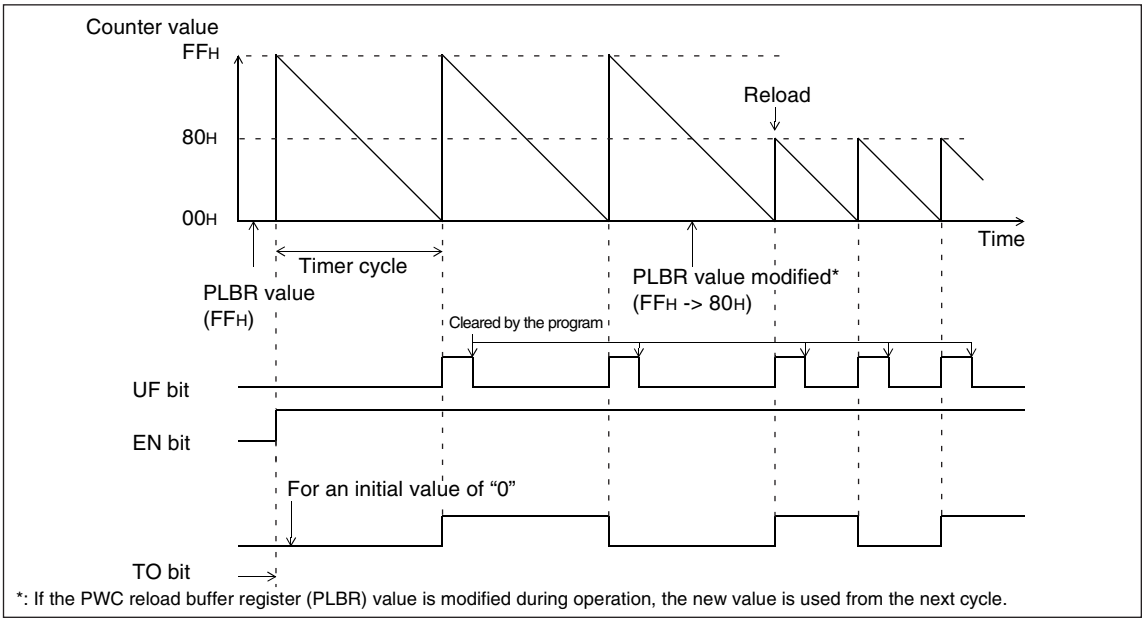
Figure 9.5-1 Interval timer function (reload timer mode) settings



On activation, the PLBR register value is loaded to the counter and the counter starts to counting-down on the rising edge of the selected count clock. When the counter value underflows (00_H --> FF_H), the PWC inverts the timer output bit (PCR2: TO) value, reloads the PLBR register value to the counter, and sets the underflow (00_H --> FF_H) interrupt request flag bit (PCR1: UF = "1") on the next rising edge of the count clock.

Figure 9.5-2 "Operation in reload timer mode" shows the operation in reload timer mode.

Figure 9.5-2 Operation in reload timer mode



Reference:

Setting the PLBR register value to "01_H" causes the TO bit to be inverted after each count clock cycle.

● One-shot timer mode

Figure 9.5-3 "Interval timer function (one-shot timer mode) settings" shows the settings required to operate in one-shot timer mode.

Figure 9.5-3 Interval timer function (one-shot timer mode) settings

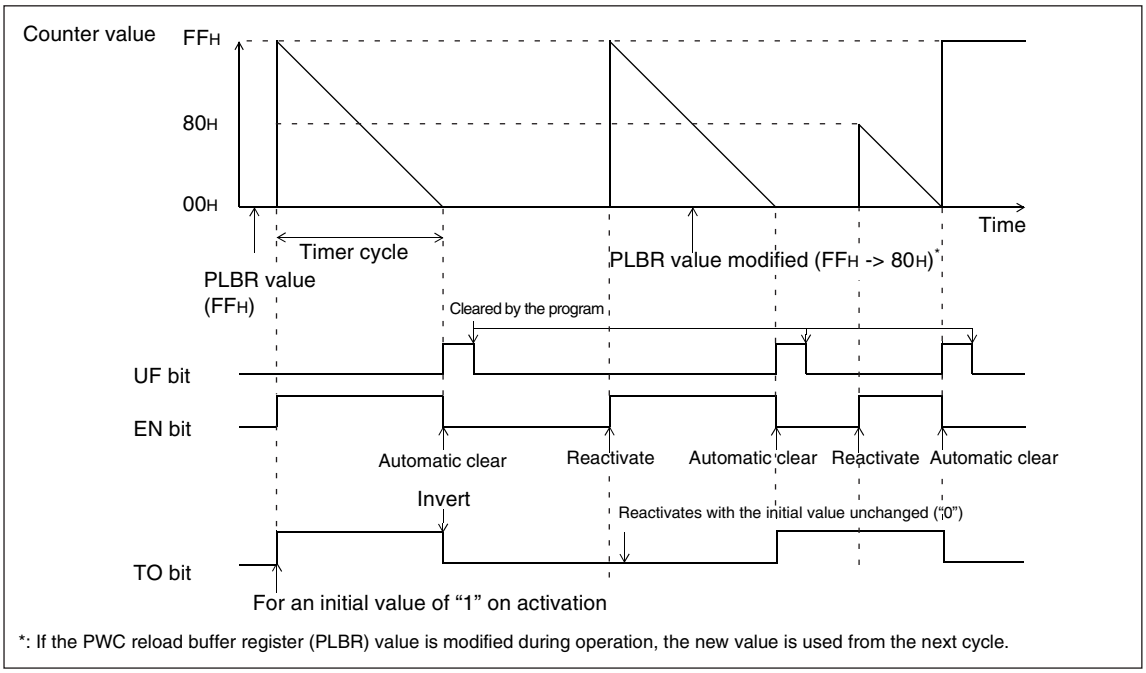
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCR1	EN	—	IE	—	—	UF	IR	BF
	1		◎			◎	×	×
PCR2	FC	RM	TO	C1	C0	W2	W1	W0
	0	1	◎	◎	◎	×	×	×
PLBR	Sets interval time (counter initial value).							

◎: Used bit
×: Unused bit
1: Set “1”.
0: Set “0”.

On activation, the PLBR register value is loaded to the counter and the counter starts to counting-down on the rising edge of the selected count clock. When the counter value underflows (00_H --> FF_H), the counter inverts the timer output bit (PCR2: TO) value, automatically clears the counter operation enable bit (PCR1: EN = "0") to stop counter operation, and sets the underflow (00_H --> FF_H) interrupt request flag bit (PCR1: UF = "1") on the next rising edge of the count clock.

Figure 9.5-4 "Operation in one-shot timer mode" shows the operation in one-shot timer mode.

Figure 9.5-4 Operation in one-shot timer mode



Note:

Do not modify PCR2 when the counter is operating (PCR1: EN = "1").

References:

- The UF bit is set to "1" if counter underflows (00_H --> FF_H), regardless of the value of the interrupt request enable bit (PCR1: IE).
- When the counter is stopped (PCR1: EN = "0") while the interval timer function is selected, the TO bit maintains the value it had immediately before the counter stopped.

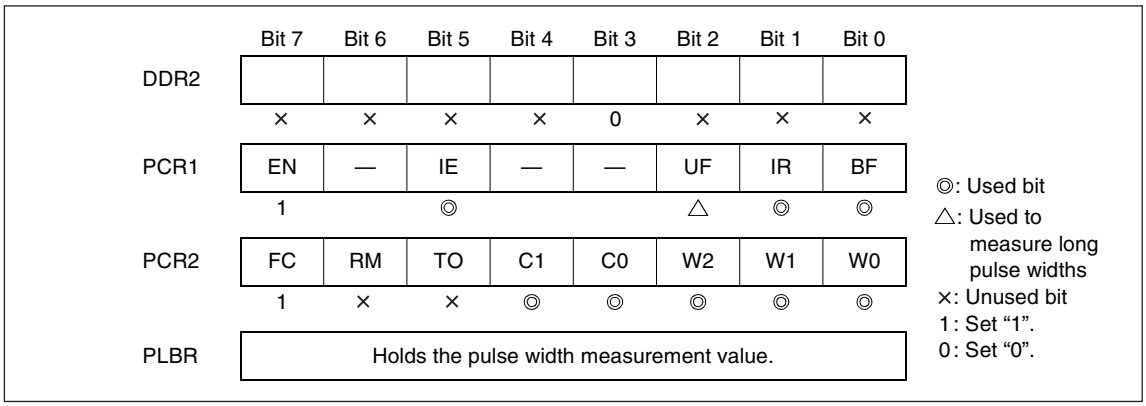
9.6 Operation of Pulse Width Measurement Function

This section describes the operations of the pulse width measurement function of the pulse width count timer.

■ Operation of pulse width measurement function

Figure 9.6-1 "Pulse width measurement function settings" shows the settings required to operate as the pulse width measurement function.

Figure 9.6-1 Pulse width measurement function settings

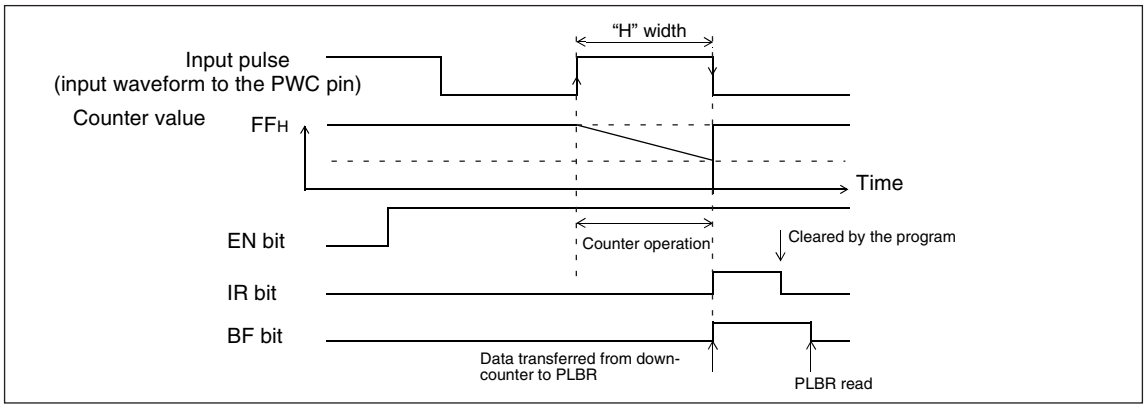


When counter operation is enabled, the counter starts to counting-down from "FF_H" when a measurement start edge is detected on the pulse input to the PWC pin. (For "H" level measurement, the counter starts measurement from the next rising edge if the input is already "H".)

On detection of the measurement completion edge, the current down-counter value is transferred to the PWC reload buffer register (PLBR), the measurement completion interrupt request flag bit (PCR1: IR) and buffer full flag bit (PCR1: BF) are both set to "1", and counter operation is re-enabled. (The function supports continuous pulse width measurement and so can be used like an input capture.)

Figure 9.6-2 "Example of "H" width measurement using pulse width measurement function" shows the operation when the measured pulse selection bits (PCR2: W2, W1, W0) are set to "000_B" ("H" width measurement).

Figure 9.6-2 Example of "H" width measurement using pulse width measurement function



Notes:

- If the previous PLBR register value has not been read during continuous pulse width measurement, the PWC leaves the BF bit set to "1" and maintains the previous measurement value. In this case, the new measurement value is lost.
- Do not modify the PCR1/PCR2 register during pulse width measurement (PCR1: EN = "1").

■ Measuring long pulse widths

To measure pulse widths longer than 2^8 times the cycle of the selected count clock, it is necessary to count the number of counter underflows (00_H --> FF_H) by software in the interrupt processing routine. Counting by software requires a buffer in RAM (a software counter) to hold the number of counter underflows (00_H --> FF_H).

After initializing the software counter and enabling counter operation, the counter starts to count down from "FF_H" when a measurement start edge is detected on the pulse input to the PWC pin.

An interrupt request is generated on detection of the measurement completion edge or when the counter underflows (00_H --> FF_H). Check the measurement completion interrupt request flag bit (PCR1: IR) and underflow (00_H --> FF_H) interrupt request flag bit (PCR1: UF) in the interrupt processing routine. If the UF bit is "1", write "0" to the UF bit to clear the interrupt request and increment the software counter (the PWC counter continues to operate).

When the IR bit is "1", calculate the pulse width (including underflows (including underflows (00_H --> FF_H) from the values of the software counter and PWC reload buffer register (PLBR).

When the PLBR register value is "00_H", calculate as 256.

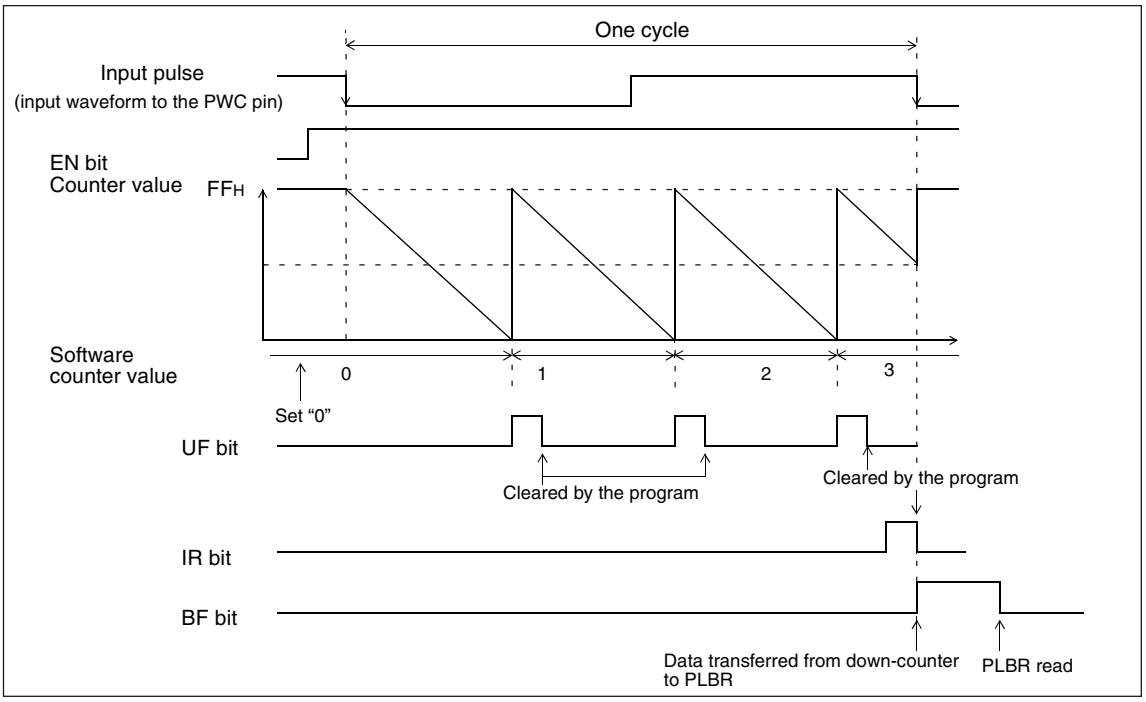
● Calculating the width of long pulses

Pulse width = [(256 - PLBR register value) + (number of counter underflows (00_H --> FF_H) x 256)] x one-cycle width of count clock

Calculate the pulse width before the next underflow (00_H --> FF_H) occurs. The correct measurement value may not be able to be calculated after the next underflow (00_H --> FF_H) occurs.

Figure 9.6-3 "Measuring long pulse width" shows the operation when the measured pulse selection bits (PCR2: W1, W0) are set to "11_B" (falling edge to falling edge).

Figure 9.6-3 Measuring long pulse width



9.7 States in Each Mode during Pulse Width Count Timer Operation

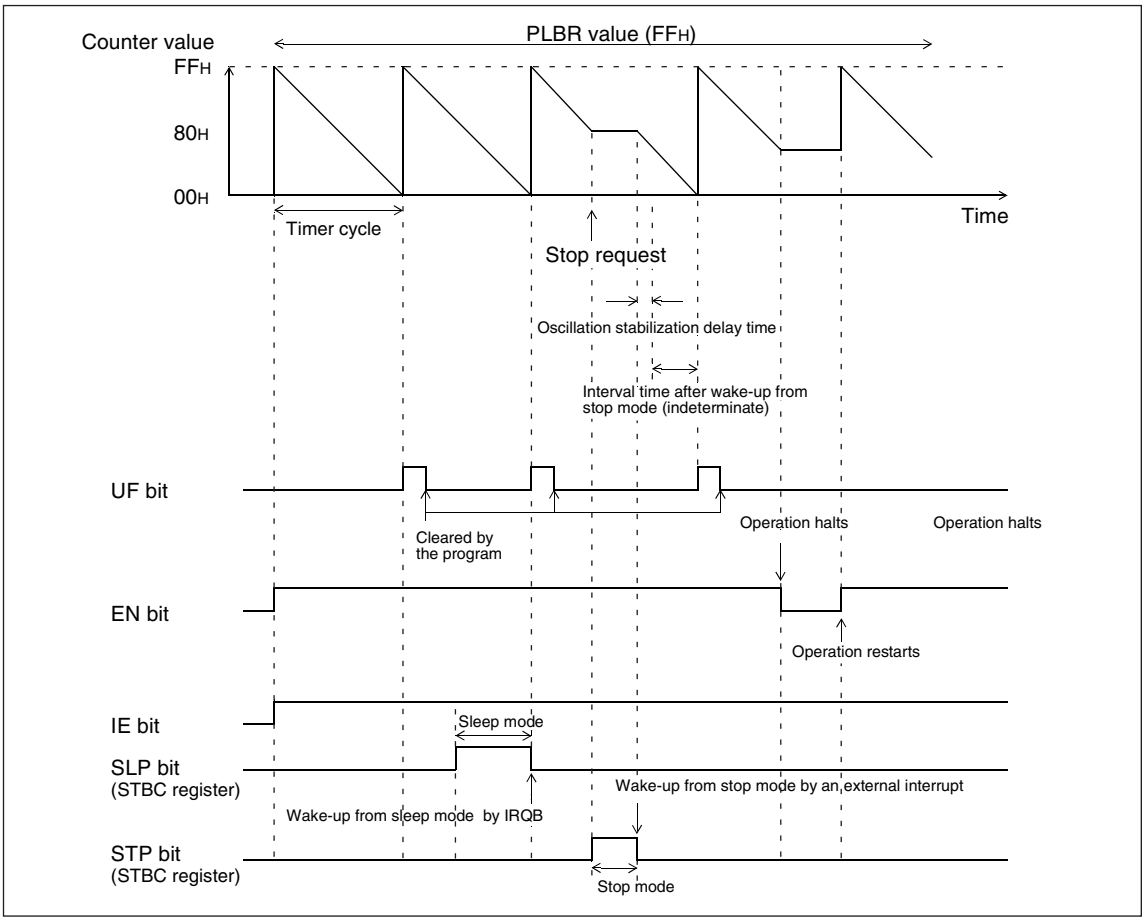
This section describes the operation of the pulse width count timer when the device goes to sleep or stop mode, or an operation halt request occurs during operation.

■ Operation during standby mode or operation halt

Figure 9.7-1 "Counter operation during standby mode or operation halt" shows the counter value state when the device goes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or pulse width measurement function.

The counter halts and maintains its current value when the device goes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. Therefore, the first interval time or pulse width measurement is not correct value. Always initialize the pulse width count timer after wake-up from stop mode.

Figure 9.7-1 Counter operation during standby mode or operation halt



9.8 Notes on Using Pulse Width Count Timer

This section lists points to note when using the pulse width count timer.

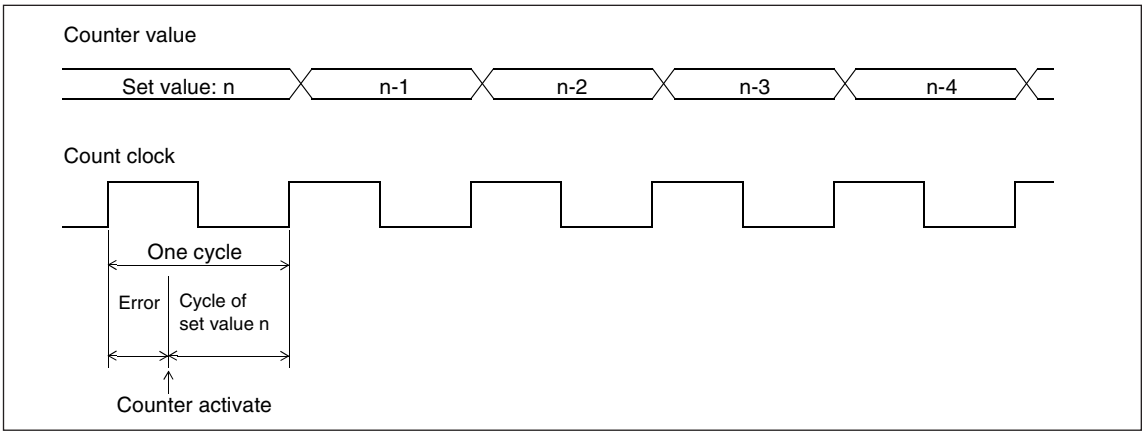
■ Notes on using pulse width count timer

● Error

When using the interval timer function, activating the counter by program is not synchronized with the start of counting down using the selected internal count clock. Therefore, the time from activating the counter until an underflow occurs may be shorter than the theoretical time by a maximum of one cycle of the count clock.

Figure 9.8-1 "Error on starting counter operation" shows the error that occurs on starting counter operation.

Figure 9.8-1 Error on starting counter operation



● Notes on setting by program

- Do not modify the contents of the PWC pulse width control register 2 (PCR2) when the interval timer function or pulse width measurement function is operating (PCR1: EN = "1").
- Stop the counter (EN = "0"), disable interrupts (IE = "0"), and clear the interrupt request flag bits (UF, IR, BF = "0") in the PCR1 register before switching between the interval timer function and pulse width measurement function (PCR2: FC).
- Interrupt processing cannot return if the interrupt request flag bit (PCR1: UF, IR, or BF) is "1" and the interrupt request enable bit is enabled (PCR1: IE = "1"). Always clear the interrupt request flag bit.
- If a previous measurement value has not been read when performing continuous pulse width measurement for pulse width measurement function, new measurement values are not transferred to the PWC reload buffer register (PLBR). The PLBR maintains the previous value. Always read the measurement value before the next underflow (00_H --> FF_H) when measuring long pulse widths.
- The interrupt request flag bit (PCR1: UF, IR, or BF) is not set if the counter is disabled (PCR1: EN = "0") at the same time as an interrupt source is generated.

9.9 Program Example for Timer Function of Pulse Width Count Timer

This section gives two program examples for the timer function of the pulse width count timer.

■ Program example 1 for interval timer function (reload timer mode)

● Processing description

- Generates repeated interval timer interrupts at 3 ms intervals (reload timer mode).
- The TO bit will be inverted after each interval time cycle. The initial value of TO bit is "0" level.
- The following shows the PLBR register value that results in an interval time of approximately 3 ms for a 12.55 MHz source oscillation frequency. The count clock is 32 t_{inst} (t_{inst}: divide-by-four source oscillation when the highest main clock is selected).

  PLBR register value = 3 ms/(32 x 4/12.5 MHz) = 293.0 (125_H)

● Coding example

```
PCR1    EQU    0034H          ; Address of the PWC pulse width control register 1
PCR2    EQU    0035H          ; Address of the PWC pulse width control register 2
PLBR    EQU    0036H          ; Address of the PWC reload buffer register

EN      EQU    PCR1:7         ; Define the counter operation enable bit.
IE      EQU    PCR1:5         ; Define the interrupt request enable bit.
UF      EQU    PCR1:2         ; Define the underflow (00H→FFH) interrupt
                                request flag bit.
ILR3    EQU    007DH          ; Address of the interrupt level setting register 3

INT_V   DSEG   ABS            ; [DATA SEGMENT]
        ORG    0FFE4H
IRQB    DW     WARI           ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                  ; [CODE SEGMENT]
                              ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                  ; Disable interrupts.
        CLRB   EN             ; Stop counter operation.
        CLRB   IE             ; Disable interrupt request output.
        MOV    ILR3,#01111111B ; Set interrupt level (level 1).
        MOV    PLBR,#125H     ; Counter reload value (interval time)
        MOV    PCR2,#00001000B ; Select interval timer function, reload timer mode,
                                initial output value of the TO bit, and 32 tinst.
        MOV    PCR1,#11100000B ; Start counter operation, enable interrupt
                                request output, clear underflow (00H→FFH)
                                interrupt request flag, clear measurement
                                completion interrupt request flag (bit 1).
        SETI                  ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   UF             ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

■ Program example 2 for interval timer function (one-shot timer mode)

● Processing description

- Generates a single 3 ms interval timer interrupt (one-shot timer mode).
- The TO bit is initialized to "1" and inverted after the interval time.
- The following shows the PLBR register value that results in an interval time of approximately 3 ms for a 12.5 MHz source oscillation frequency. The count clock is 32 t_{inst} (t_{inst}: divide-by-four source oscillation when the highest main clock is selected).

  PLBR register value = 3 ms/(32 x 4/12.5 MHz) = 293.0 (125_H)

● Coding example

```
PCR1    EQU    0034H           ; Address of the PWC pulse width control register 1
PCR2    EQU    0035H           ; Address of the PWC pulse width control register 2
PLBR    EQU    0036H           ; Address of the PWC reload buffer register

EN      EQU    PCR1:7          ; Define the counter operation enable bit.
IE      EQU    PCR1:5          ; Define the interrupt request enable bit.
UF      EQU    PCR1:2          ; Define the underflow (00H→FFH) interrupt
                                 request flag bit.
ILR3    EQU    007DH           ; Address of the interrupt level setting register 3

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFE4H
IRQB    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   EN              ; Stop counter operation.
        CLRB   IE              ; Disable interrupt request output.
        MOV    ILR3,#01111111B ; Set interrupt level (level 1).
        MOV    PLBR,#125H      ; Counter reload value (interval time)
        MOV    PCR2,#01110000B ; Select interval timer function, one-shot timer
                                 mode, initial output value of the TO bit, and
                                 32 tinst.
        MOV    PCR1,#11100000B ; Start counter operation, enable interrupt
                                 request output, clear underflow (00H→FFH)
                                 interrupt request flag, clear measurement
                                 completion interrupt request flag (bit 1).
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   UF              ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

9.10 Program Example for Pulse Width Measurement Function

This section gives a program example for the pulse width measurement function of the pulse width count timer.

■ Program example for pulse width measurement function

● Processing description

- Measures the "H" width of a pulse input to the PWC pin (pulse width measurement function).
- Generates an interrupt when pulse width measurement completes and continues measurement.
- The following shows the relationship between the PLBR register value and the measured pulse width when the source oscillation is 12.5 MHz and the 4 t_{inst} (t_{inst}: divide-by-four source oscillation when the highest main clock is selected).

  Pulse width = (256 – PLBR register value) x 4 x 4/12.5 MHz (measurement range: 1.28 µs to 327.7 µs)

● Coding example

```
DDR2    EQU    0006H           ; Address of the PORT2 data direction register
PCR1    EQU    0034H           ; Address of the PWC pulse width control register 1
PCR2    EQU    0035H           ; Address of the PWC pulse width control register 2
PLBR    EQU    0036H           ; Address of the PWC reload buffer register

EN      EQU    PCR1:7          ; Define the counter operation enable bit.
IE      EQU    PCR1:5          ; Define the interrupt request enable bit.
IR      EQU    PCR1:1          ; Measurement completion interrupt request flag bit.
BF      EQU    PCR1:0          ; Buffer full flag bit.

ILR3    EQU    007DH           ; Address of the interrupt level setting register 3

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFE4H
IRQB    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        MOV    DDR2,#00000000B ; Set P23/PWC pin as an input.
        CLRI                   ; Disable interrupts.
        CLRB   EN              ; Stop counter operation.
        CLRB   IE              ; Disable interrupt request output.
        CLRB   BF              ; Clear buffer full flag (PCR1: BF)
        MOV    ILR3,#01111111B ; Set interrupt level (level 1).
        MOV    PCR2,#10001000B ; Select the pulse width measurement function,
                                 4 tinst, and "H" pulse.
        MOV    PCR1,#10100000B ; Enable counter operation, enable interrupt
                                 request output, clear underflow (00H→FFH)
                                 interrupt request flag (bit 2), clear measurement
                                 completion interrupt request flag (IR).
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   IR              ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
        MOV    A,PLBR          ; Read pulse width measurement value and clear
                                 BF flag.
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

CHAPTER 10

8/16-BIT TIMER/COUNTER

This chapter describes the functions and operation of the 8/16-bit Timer/Counter.

10.1 Overview of 8/16-bit Timer/Counter

The 8/16-bit timer/counter is made up of two 8-bit timers (Timer 11/21 and Timer 12/22) that can be used separately (8-bit mode) or connected in cascade to form one counter (16-bit mode).
Timer 11/21 can be selected to function as either an interval timer or a counter. The interval timer function counts up in sync with one of three interval count clocks. The counter function counts up by a clock input to the external pin.
Timer 12/22 functions as an interval timer clocked by one of three internal count clocks. In the 16 bit mode, it is connected in series with Timer 11/21. The output can be used to generate variable frequency square wave output.

■ Interval timer function

The interval timer function generates repeated interrupts at variable intervals. Also, as the 8/16-bit timer/counter can invert the output level of the pin (TO1/TO2 pin) each time an interval time is generated, the 8/16-bit timer/counter can output variable frequency square waves (Timer 11/21 in 8 bit mode, or 16 bit mode).

- In 8-bit mode, Timer 11/21 and Timer 12/22 operate as two independent interval timers, each of which can count time intervals ranging from the clock period (the time of one clock cycle) to 2^8 times the clock period.
- In 16-bit mode, the two counters form a single 16-bit timer, with Timer 11/21 containing the LSBs and Timer 12/22 the MSBs. The interval timer can operate with a cycle among 1 and 2^{16} times the internal count clock cycle.
- The count clock can be selected from three different internal clocks. (An external clock can be selected for Timer 11/21, but it will then function as a counter).

Table 10.1-1 "Timer 11/21 interval times and square wave frequencies in 8-bit mode" to Table 10.1-3 "Interval times and square wave frequencies in 16-bit mode" list the interval time and square wave output ranges for the various modes.

Table 10.1-1 Timer 11/21 interval times and square wave frequencies in 8-bit mode

Count clock cycle		Interval time	Square wave output range (Hz)
Internal count clock	2 t_{inst}	2 t_{inst} to 2^9 t_{inst}	$1/(2^2\ t_{inst})$ to $1/(2^{10}\ t_{inst})$
	32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}	$1/(2^6\ t_{inst})$ to $1/(2^{14}\ t_{inst})$
	512 t_{inst}	2^9 t_{inst} to 2^{17} t_{inst}	$1/(2^{10}\ t_{inst})$ to $1/(2^{18}\ t_{inst})$
External clock	1 t_{ext}	1 t_{ext} to 2^8 t_{ext}	$1/(2\ t_{ext})$ to $1/(2^9\ t_{ext})$

Table 10.1-2 Timer 12/22 interval times in 8-bit mode

Count clock cycle		Interval time
Internal count clock	2 t_{inst}	2 t_{inst} to 2^9 t_{inst}
	32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}
	512 t_{inst}	2^9 t_{inst} to 2^{17} t_{inst}

Table 10.1-3 Interval times and square wave frequencies in 16-bit mode

Count clock cycle		Interval time	Square wave output range (Hz)
Internal count clock	2 t_{inst}	2 t_{inst} to 2^{17} t_{inst}	$1/(2^2\ t_{inst})$ to $1/(2^{18}\ t_{inst})$
	32 t_{inst}	2^5 t_{inst} to 2^{21} t_{inst}	$1/(2^6\ t_{inst})$ to $1/(2^{22} t_{inst})$
	512 t_{inst}	2^9 t_{inst} to 2^{25} t_{inst}	$1/(2^{10}\ t_{inst})$ to $1/(2^{26}\ t_{inst})$
External clock	1 t_{ext}	1 t_{ext} to 2^{16} t_{ext}	$1/(2\ t_{ext})$ to $1/(2^{17}\ t_{ext})$

t_{inst}: Instruction cycle (affected by clock mode, etc.)
t_{ext}: External clock period

Reference:

[Calculation example for the interval time and square wave frequency:]

In this example, the main clock source oscillation (F_{CH}) is 12.5 MHz, the Timer 11/21 data register (T1/3DR) value is set to "DD_H(221)", and the count clock cycle is set to the 8-bit mode operation at 2 t_{inst}. In this case, the Timer 11/21 interval time and frequency of square wave output from the TO1/TO2 pin are calculated as follows.

Assume that the highest clock speed has been selected via the system clock control register (SYCC: CS1 = 1, CS0 = 1) (1 instruction cycle = $4/F_{CH}$).

Interval time = (2 x 4/FCH) x (T1/3DR register value + 1)
= (8/12.5 MHz) x (221 + 1)
= 142.08 µs

Output frequency = FCH/(2 x 8 x (T1/3DR register value + 1))
= 12.5 MHz/(16 x (221 + 1))
= 3.52 kHz

■ Counter function

The counter function counts rising edges of an external count clock applied to the external pin (EC1, EC2 pin). Since the external clock can be selected only for Timer 11/21, the counter function operates in either the 8-bit Timer 11/21 or 16 bit mode.

- The counter counts up, clocked by external clocks. When the count equals the set value, it generates an interrupt request in both 8-bit and 16-bit mode and inverts the level being output at the TO1/TO2 pin .
- In the 8 bit mode, Timer 11/21 can count as high as 2^8.
- In the 16 bit mode, the function counts as high as 2^{16}.
- By injecting an external clock having a set period, the counter function can be used the same way as the interval timer function.

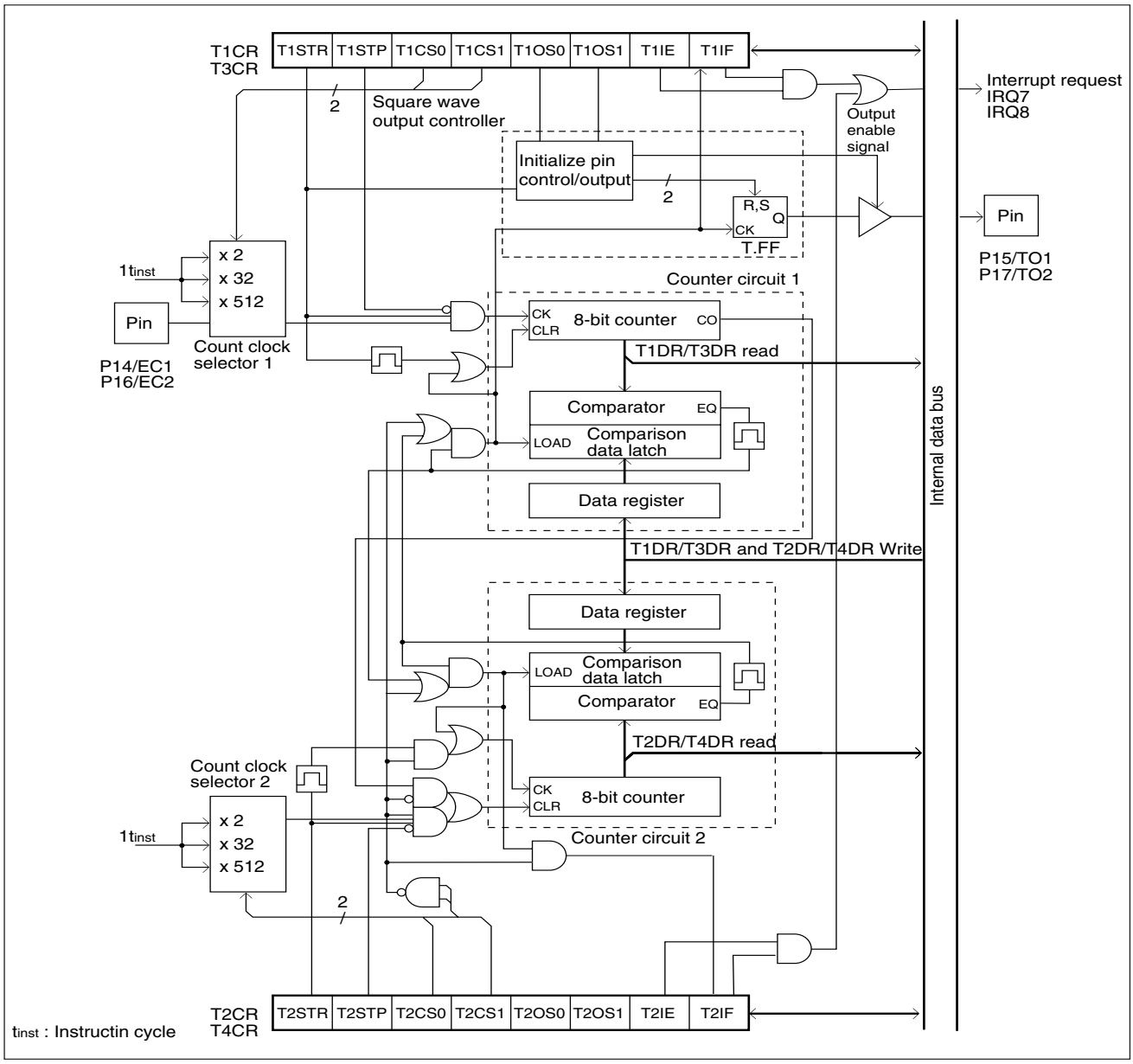
10.2 Block Diagram of 8/16-bit Timer/Counter

The 8/16-bit timer/counter consists of the following five blocks:

- **Count clock selectors 1 and 2**
- **Counter circuits 1 and 2**
- **Square wave output controller**
- **Timer data registers (T1DR, T2DR, T3DR, T4DR)**
- **Timer control registers (T1CR, T2CR, T3CR, T4CR)**

■ Block diagram of 8/16-bit timer/counter

Figure 10.2-1 Block diagram 8/16-bit timer/counter



● Count clock selectors 1, 2

This circuit selects an input clock. In the 8-bit timer 11/21 and 16 bit modes, count clock selector 1 selects one of four clocks: three internal clocks, and an external clock. In the 8-bit mode, count Clock Selector 2 selects one of three internal clocks only.

● Counter circuit 1, 2

Counter circuit 1 and counter circuit 2 are each made up of an 8-bit counter, a comparator, a comparison data latch, and a data register (T1/3DR or T2/4DR).

In each counter circuit, the 8-bit counter is an up-counter clocked by the selected count clock. The comparator compares the count in the counter with the value in the comparison data latch. When it detects a match, it clears the counter, and loads the contents of the data register into the comparison data latch.

In the 8 bit-mode, the two counter circuits operate independently as Timer 11/21 and Timer 12/22. In the 16-bit mode, the two circuits are connected in series to form a single 16-bit counter with counter circuit 1 forming the low (8 LSBs) end of the counter, and counter circuit 2 at the high (8 MSBs) end.

● Square wave output control circuit

An interrupt request is generated when the comparator detects a match in the 8-bit timer mode or the 16-bit mode.

At this time, if the square wave output is enabled, the output control circuit inverts the level output at the TO1/TO2 pin. (only for 8-bit timer 11/21 in 8-bit mode or 16-bit mode).

The circuit can also initialize the output level to have the output square wave start out in a specific state ("H" or "L") (only for 8-bit timer 11/21 in 8-bit mode in 8-bit mode or 16-bit mode).

● T1DR/T3DR and T2DR/T4DR registers

The value to be compared with the count in the counter is set by writing the desired value into these registers. They can be read to determine the current counter values.

● T1CR/T3CR and T2CR/T4CR registers

These registers are used to select the function, to enable or disable operation, control interrupts, and check the timer/counter status.

10.3 Structure of 8/16-bit Timer/Counter

This section describes the pins, pin block diagram, registers, and interrupt source of the 8/16-bit Timer/Counter.

■ 8/16-bit timer/counter pins

The 8/16-bit timer/counter 11, 12 uses the P14/EC1 pin and P15/TO1 pin. The 8/16-bit timer/counter 21, 22 uses the P16/EC2 and P17/TO2 pin. The P14/EC1, P16/EC2 pin can function as a general-purpose I/O port (P14, P16) or external clock input pin of timer(EC1, EC2). The P15/TO1, P17/TO2 pin can function as a general-purpose I/O port (P15, P17) or the square wave output pin of timer(TO1, TO2).

EC1 and EC2:

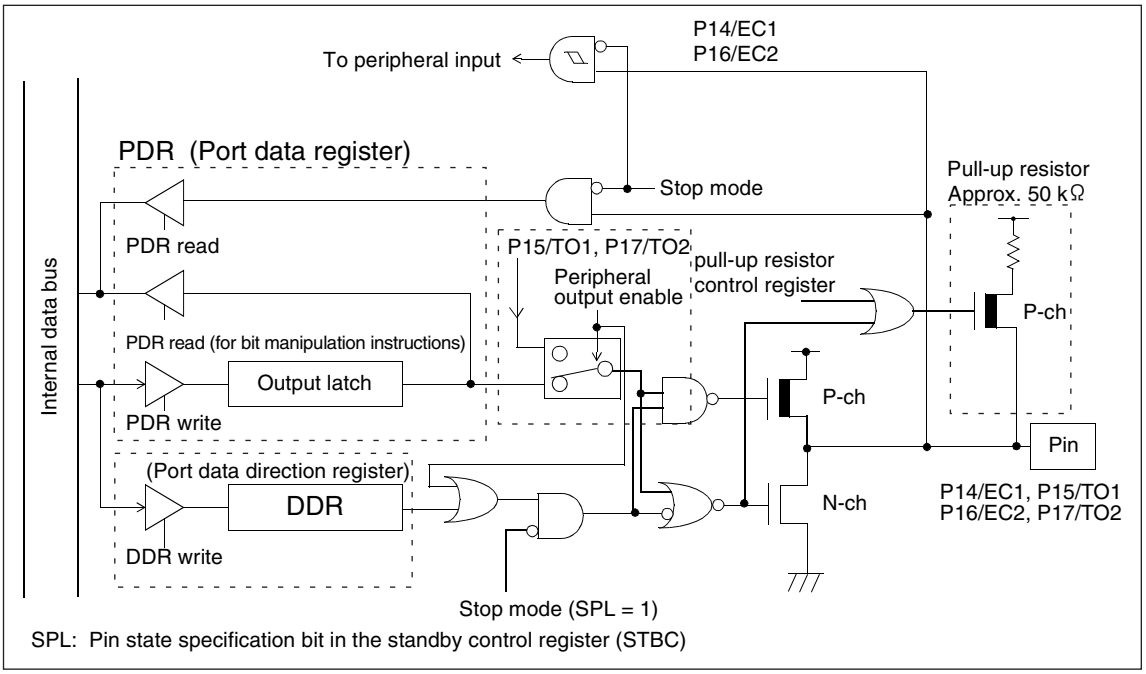
In the 8-bit timer 11/21 or 16-bit mode, if external clock input (counter function) is selected (T1/3CR: T1CS1 = 1, T1CS0 = 1), the counter counts the external clocks applied to this pin. The P14/EC1, P16/EC2 pin sets the pin as an input port in the port data direction register DDR1: bit 4, bit 6 = "0") when using as the EC1 and EC2 pin.

TO1 and TO2:

In the 8-bit timer 11/21 or 16-bit mode, a square wave is output at this pin. Enabling square wave output (T1/3CR: T1OS1, T1OS0 = 11_B) automatically sets the P15/TO1, P17/TO2 pin as an output pin, regardless of the port data direction register (DDR1: bit 5, bit 7) value, and sets the pin to function as the TO1, TO2 pin.

■ Block diagram of 8/16-bit timer counter pins

Figure 10.3-1 Block diagram of 8/16-bit timer/counter pins



SPL: Pin state specification bit in the standby control register (STBC)

■ 8/16-bit timer/counter registers

Figure 10.3-2 8/16-bit timer/counter registers

T1/3CR (Timer 11/21 control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0019H 0015H	T1IF	T1IE	T1OS1	T1OS0	T1CS1	TS1CS0	T1STP	T1STR	000000X0B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

T2/4CR (Timer 12/22 control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0018H 0014H	T2IF	T2IE	T2OS1	T2OS0	T2CS1	T2CS0	T2STP	T2STR	000000X0B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

T1/3DR (Timer 11/21 data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001BH 0017H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

T2/4DR (Timer 12/22 data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001AH 0016H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X : Indeterminate

■ 8/16-bit timer/counter interrupt source

IRQ7:

In the interval timer and counter functions for timer 11 in 8-bit mode or 16 bit mode, timer 12 in 8-bit mode, if the interrupt request output is enabled, an IRQ7 interrupt request will be generated when the count in the counter equals the value set in the data register. Interrupt request outputs are enabled by setting the proper timer control register bit (T1/3CR: T1IE = 1 in the 8-bit Timer 11, T2/4CR: T2IE = 1 in the 8-bit Timer 12 mode or 16-bit mode).

IRQ8:

In the interval timer and counter functions for timer 21 in 8-bit mode or 16 bit mode, timer 22 in 16-bit modes, if the interrupt request output is enabled, an IRQ8 interrupt request will be generated when the count in the counter equals the value set in the data register. Interrupt request outputs are enabled by setting the proper timer control register bit (T3CR: T3IE = 1 in the 8-bit Timer 21, T4CR: T4IE = 1 in the 8-bit Timer 22 mode or 16-bit mode).

10.3.1 Timer 11/21 Control Register (T1/3CR)

In the 8-bit Timer 11/21 and in the 16-bit mode, the Timer 11/21 Control Register (T1/3CR) is used to select functions, to enable/disable operation, to control interrupts, and to check status. In the 8-bit mode, the Timer 21/22 control register (T2/4CR) must still be initialized, even only Timer 11/21 is used.

■ Timer 11/21 control register (T1/3CR)

Figure 10.3-3 Timer 11/21 control register (T1/3CR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0019H 0015H	T1IF	T1IE	T1OS1	T1OS0	T1CS1	T1CS0	T1STP	T1STR	000000X0B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

T1STR	Timer activation bit
0	Stops counter
1	Clears counter, then starts it

T1STP	Timer stop bits
0	Causes counter to resume counting without clearing
1	Temporarily stops counter

T1CS1	T1CS0	Count clock selection Bits (tinst : instruction cycle)
0	0	2 tinst
0	1	32 tinst
1	0	512 tinst
1	1	External clock

T1OS1	T1OS0	Square wave output control bits
0	0	Use pin for general-purpose port
0	1	Set data to output square wave “L” state
1	0	Set data to output square wave “H” state
1	1	Output set level at square wave output pin (TO1/TO2)*

T1IE	Interrupt request enable bit
0	Disable interrupt request output
1	Enable interrupt request output

T1IF	Interrupt request Flag Bit	
	Read	Write
0	No counter match	clears this bit
1	Have counter match	No effect: The bit does not change.

R/W : Readable and writable
X : Indeterminate
: Initial value

*: The square wave output pin will go to the level corresponding to data set when T1STR is “0.”

Table 10.3-1 Timer 11/21 control register (T1/3CR) bits

Bit		Function
Bit 7	T1IF: Interrupt request flag bit	• 8 bit-mode: Set to "1" when the count in the timer 11/21 counter matches the value set in the T1/3DR, the timer 11/21 data register (comparison data latch). • 16-bit mode: Set to "1" when the counts in the timer 11/21 and timer 12/22 counters match the values set in the T1/3DR and T2/4DR registers, respectively. • An interrupt request is output when both this bit and the interrupt request enable bit (T1IE) are "1". • Writing "0"clears this bit. writing "1" has no effect and does not change the bit value.
Bit 6	T1IE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. • An interrupt request is output when both this bit and the interrupt request flag bit (T1IF) are "1".
Bit 5 Bit 4	T1OS1 and T1OS0: square wave output control bits	• P15/TO1 or P17/TO2 is a general-purpose I/O port pin (P15, P17) if both of these bits are "00_B." If either bit is "1," it is the square wave output pin (TO1 or TO2). • If written to "01_B," or "10_B," the initialize data will be set in the square wave output controller, but the corresponding level will not be output to the TO1 or TO2 pin. • If both bits are "11_B," and the function is in the stop timer state (T2STR= 0), the TO1 or TO2 pin is set to a level corresponding to the initialize data.
Bit 3 Bit 2	T1CS1 and T1CS0: Clock source selection bits	• Selects the count clock to be supplied to the counter. • Selects one of three internal clocks, or an external clock. • When both bits are "11_B," Timer 11/21 operates as a counter with the external clock is selected as the count clock. **Note:** If external clock input is selected (T1CS1, T1CS0 = 11_B), Set the P14/EC1, P16/EC2 pin as an input port pin.
Bit 1	T1STP: Timer stop bit	• This bit is used to temporarily stop the counter. • Writing this bit to "1" temporarily stops the counter. Writing it to "0" when the timer in startup state (T1STR = 1), restarts the counter where it left off.
Bit 0	T1STR: Timer activation bit	Starts and stops timer. Changing this bit from "0" to "1" clears the counter. At this time, if the timer is in the continuous operation mode (T1STP = 0), the counter starts (counts up, clocked by the selected count clock). Writing this bit to "0" stops the counter. In the 16 bit mode, both Timer 11/21 and Timer 12/22 are cleared at timer start (T1STP = 0 --> 1).

Note:

Before using 8/16-bit timer/counter Timer 11/21 only in 8 bit mode, first set the timer count clock selection bits of the Timer 21/22 control register (T2/4CR: T2CS1, T2CS0) to some state other than "11_B". Operating in this mode without making this register setting could result in faulty operation.

10.3.2 Timer 12/22 Control Register (T2/4CR)

In the 8 bit mode, the Timer 12/22 Control Registers (T2/4CR) are used to select functions, to enable/disable operation, to control interrupts, and to check states. In the 16 bit mode, although the function is controlled by the Timer 11/21 control register (T1/2CR), the Timer 12/22 control register (T2/4CR) must still be set.

■ Timer 12/22 control register (T2/4CR)

Figure 10.3-4 Timer 12/22 control register (T2/4CR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0018H 0014H	T2IF	T2IE	T2OS1	T2OS0	T2CS1	T2CS0	T2STP	T2STR	000000X0B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

T2STR	Timer activation bit
0	Stops counter
1	Clears counter, then starts it

T2STP	Timer stop bit
0	Causes counter to resume counting without clearing
1	Temporarily stops counter

T2CS1	T2CS0	Count clock selection bits
0	**0**	2 t_{inst}
0	**1**	32 t_{inst}
1	**0**	512 t_{inst}
1	**1**	16-bit mode

T2OS1	T2OS0	Unused bits
0	**0**	These two bit must set to 00B. Do not set to other value.
0	**1**	
1	**0**	
1	**1**	

T2IE	interrupt request enable bit
0	Disables interrupt request output
1	Enables interrupt request output

T2IF	interrupt request flag bit	
	Read	Write
0	No counter match	Clears this bit
1	Have counter match	No effect: The bit does not change.

R/W : Readable and writable
X : Indeterminate
□ : Initial value

Table 10.3-2 Timer 12/22 control register (T2/4CR) bits

Bit		Function
Bit 7	T2IF: Interrupt request flag bit	• Set to "1" when the count in the timer 12/22 counter matches the value set in the T2/4DR, the Timer 12/22 data register (comparison data latch). • An interrupt request is output when both this bit and the interrupt request enable bit (T2IE) are "1". • Writing "0"clears this bit. writing "1" has no effect and does not change the bit value. **Note:** In the 16-bit mode, the T1IF bit is the valid interrupt request flag, and the T2IF bit has no effect.
Bit 6	T2IE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. • An interrupt request is output when both this bit and the interrupt request flag bit (T2IF) are "1". **Note:** In the 16 bit mode, the T1IE bit is the valid interrupt request enable bit, and the T2IE bit has no effect.
Bit 5 Bit 4	T2OS1 and T2OS0: Unused bits	• These two bit must set to 00_B. • Do not set to other value
Bit 3 Bit 2	T2CS1 and T2CS0: Clock Source Selection Bits	• Selects the count clock to be supplied to the counter. • Selects one of three internal clocks. • Setting to "11_B" selects the 16-bit mode. **Note:** In 16-bit mode, T1CS1 and T1CS0 select the clock. T2CS1 and T2CS0 serve only to select the 16 bit-mode.
Bit 1	T2STP: Timer stop Bit	• This bit is used to temporarily stop the counter. • Writing this bit to "1" temporarily stops the counter. Writing it to "0" when the timer start bit (T2STR) is "1," restarts the counter where it left off. **Note:** In 16-bit mode, T1STP is the stop bit, and T2STP has no effect.
Bit 0	T2STR: Timer activation bit	• Starts and stops timer. • Changing this bit from "0" to "1" clears the counter. At this time, if the T2STP bit is "0," the counter starts (counts up, clocked by selected count clock). Writing this bit to "0" stops the counter. **Note:** In 16-bit mode, T1STR is the start bit, and T2STR has no effect.

Note:

When using timer 12/22 in the 16-bit mode, set T2CS1 and T2CS0 to "11_B", then use the T1/3CR register to control the circuit.

10.3.3 Timer 11/21 Data Register (T1/3DR)

The Timer 11/21 data register (T1/3DR) are used to set all or part of the interval time or counter value, and to read out all or part of the counter value, depending on the mode and function being used. In 8-bit mode, it sets the Timer 11/21 interval time (interval timer function) or counter value (counter function), and reads out the counter value. In 16-bit mode, it sets the 8 LSBs of the 16-bit timer interval (interval timer function) or counter value (counter function), and reads out the counter value.

■ Timer 11/21 data register (T1/3DR)

The value set into this register is compared with the counter value (count). If you read the register, you get the current counter value. The register setting cannot be read out.

Figure 10.3-5 "Timer 11/21 data register (T1/3DR)" shows the bit structure of the Timer 11/21 data register.

Figure 10.3-5 Timer 11/21 data register (T1/3DR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001BH 0017H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X: Indeterminate

● 8-bit mode (Timer 11/21)

The value in this register is compared with the count in the timer 11/21 counter. For the interval timer function it sets the interval time, and for the counter function, it sets the count to be detected. When count operation enabled (T1/3CR: T1STR = 0 --> 1, T1STP = 0), the value in the T1/3DR register is loaded into the comparison data latch, and the counter starts counting up.

When the counter counts up to where it matches the value in the comparison data latch, the value in the T1/3DR register is re-loaded into the comparison data latch, and the counter is cleared and continues to count.

Since the comparison data latch is reloaded when a match is detected, if a new value is loaded into the T1/3DR register while the counter is counting, the new value will not take effect until the next count cycle (after a match is detected in the current cycle).

Reference:

The T1/3DR setting for interval timer operation can be calculated using the following formula. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)
T1/3DR register value = interval time/(count clock cycle x instruction cycle) - 1

● 16-bit mode

The value in this register is compared with the counter value for the lower 8 bits

(LSBs) of the 16-bit timer. In the interval timer function, this sets the lower 8 bits of the interval time setting, and in the counter function, the lower 8 bits of the count to be detected. The contents of the T1/3DR register are loaded into the lower 8 bits of the comparison data latch when the counter first starts operating and when a match is detected in the 16-bit count. Therefore, if a new value is loaded into the T1/3DR register while the 16-bit counter is counting, the new value will not take effect until after the next match is detected.

For information on T1/3DR settings in the interval timer mode, refer to Section 10.3.4 "Timer 12/22 Data Register (T2/4DR)".

10.3.4 Timer 12/22 Data Register (T2/4DR)

The Timer 12/22 data register (T2/4DR) is used to set all or part of the interval time or counter value, and to read out all or part of the counter value, depending on the mode and function being used. In 8-bit mode, it sets the Timer 12/22 interval time (interval timer function) or counter value (counter function), and reads out the counter value. In 16-bit mode, it sets the 8 MSBs of the 16-bit timer interval (interval timer function) or counter value (counter function), and reads out the counter value.

■ Timer 12/22 data register (T2/4DR)

The value set into this register is compared with the counter value (count). If you read the register, you get the current counter value. The register setting cannot be read out.

Figure 10.3-6 "Timer 12/22 data register (T2/4DR)" shows the bit structure of the Timer 12/22 data register.

Figure 10.3-6 Timer 12/22 data register (T2/4DR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001AH 0016H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X: Indeterminate

● 8-bit mode (Timer 12/22)

The value in this register is compared with the count in the timer 12/22 counter. For the interval timer function, it sets the interval time, and for the counter function, it sets the count to be detected. The value in the T2/4DR register is reloaded into the comparison data latch when counter operation starts, and when a match is detected.

If a new value is loaded into the T2/4DR register while the counter is counting, the new value will not take effect until the next count cycle (after a match is detected in the current cycle).

Reference:

The T2/4DR setting for interval timer operation can be calculated using the following formula. (The instruction cycle time is affected by the clock mode, and speed-shift selection.)
T2/4DR register value = interval time/(count clock cycle x instruction cycle time) -1

● 16-bit mode

The value in this register is compared with upper 8 bits (MSBs) of the 16-bit timer. In the interval timer function, this sets the upper 8 bits of the interval time setting, and in the counter function, the upper 8 bits of the count to be detected. The contents of the T2/4DR register are loaded into the upper 8 bits of the comparison data latch when the counter first starts operating and when a match is detected in the 16-bit count. Therefore, if a new value is loaded into the T2/4DR register while the 16-bit counter is counting, the new value will not take effect until after the next match is detected. In the 16 bit mode, the operation of the counter is controlled by the Timer 11/21 control register (T1/3CR).

Reference:

In the interval timer function, the T1/3DR and T2/4DR register settings can be calculated from the following formula. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)

16-bit data value = interval time/(count clock cycle x instruction cycle) - 1

The 8 MSBs of the 16-bit data value are the T2/4DR setting, and the 8 LSBs are the T1/3DR setting.

10.4 8/16-bit Timer/Counter Interrupt

In the 8/16-bit timer/counter, interrupt conditions are satisfied (if interrupts are enabled) when the counter matches the data register. This is true for both the interval timer and counter functions.

■ 8/16-bit timer/counter interrupt

Table 10.4-1 "8/16-bit timer/counter interrupt control bits and interrupts" lists the 8/16-bit timer/counter interrupts, interrupt request flags and IRQ output enable bits.

Table 10.4-1 8/16-bit timer/counter interrupt control bits and interrupts

	8-bit mode		16-bit mode
Timer 11/21 and Timer 12/22	Timer 11/21	Timer 12/22	Timer 11/21 + Timer 12/22
Interrupt request flag bit	T1/3CR:T1IF	T2/4CR:T2IF	T1/3CR:T1IF
interrupt request enable bit	T1/3CR:T1IE	T2/4CR:T2IE	T1/3CR:T1IE
Interrupt source	8-bit counter matches T1/3DR	8-bit counter matches T2/4DR	16-bit counter matches T1/3DR+T2/4DR

In 8-bit mode, 8/16-bit timer/counter interrupt requests are generated independently for Timer 11/21 and Timer 12/22. In 16-bit mode, the interrupt request is generated only for Timer 11/21, but basic operation is the same. Interrupt operation will therefore be described only for Timer 11/21 in 8-bit mode.

● 8-bit mode timer 11/21 interrupt operation

The counter counts up from "00_H", clocked by the selected count clock. When the count in the counter matches the value in the comparison data latch (corresponding to the value in timer data register T1/3DR), the interrupt request flag bit is set to "1" (T1/3CR: T1IF).

At this time, an interrupt request (IRQ7, 8) to the CPU is generated if the interrupt request enable bit is enabled (T1/3CR: T1IE="1"). Write "0" to the T1IF bit in the interrupt processing routine to clear the interrupt request.

The T1IF bit is set to "1" when the counter value matches the set value, regardless of the value of the T1IE bit.

References:

The T1IF bit is not set if the counter is stopped (T1/3CR: T1STR = "0") at the same time as the counter value matches the T1/3DR register.

An interrupt request is generated immediately if the T1IF bit is "1" when the T1IE bit is changed from disabled to enabled ("0" --> "1").

■ Registers and Vector Table for 8/16-bit Timer/Counter (11, 12, 21, 22) Interrupt

Table 10.4-2 Registers and vector table for 8/16-bit timer/counter interrupt

Interrupt	Interrupt level settings register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ7 (Timer 11 or Timer 12)	ILR2 ($007C_H$)	L71 (Bit 7)	L70 (Bit 6)	$FFEC_H$	$FFED_H$
IRQ8 (Timer 21 or Timer 22)	ILR3 ($007D_H$)	L81 (Bit 1)	L80 (Bit 0)	$FFEA_H$	$FFEB_H$

See section 3.4.2 "Interrupt Processing" for details on the interrupt operation.

10.5 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the 8/16-bit timer/counter (11, 12, 21, 22).

■ Operation of interval timer function

● 8-bit mode

Figure 10.5-1 "Interval timer function (timer 11/21) settings" shows the settings required to operate Timer 11/21 as the interval timer function in the 8-bit mode.

Figure 10.5-1 Interval timer function (timer 11/21) settings

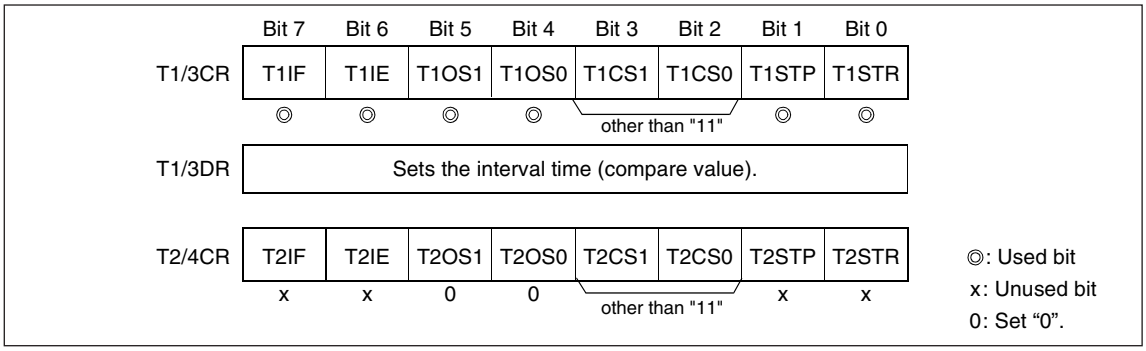
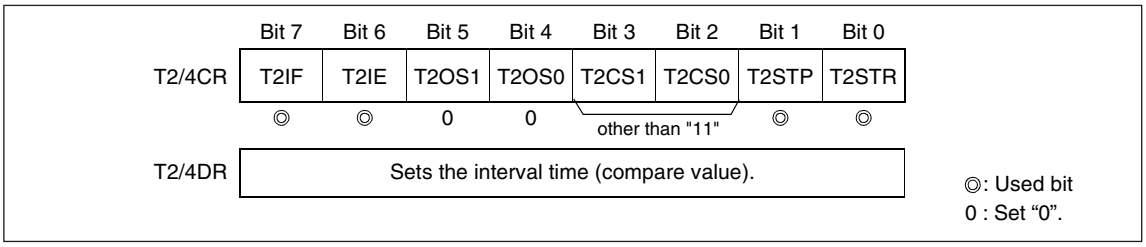


Figure 10.5-2 "Interval timer function (timer 12/22) settings" shows the settings required to operate Timer 12/22 as the interval timer function in the 8-bit mode.

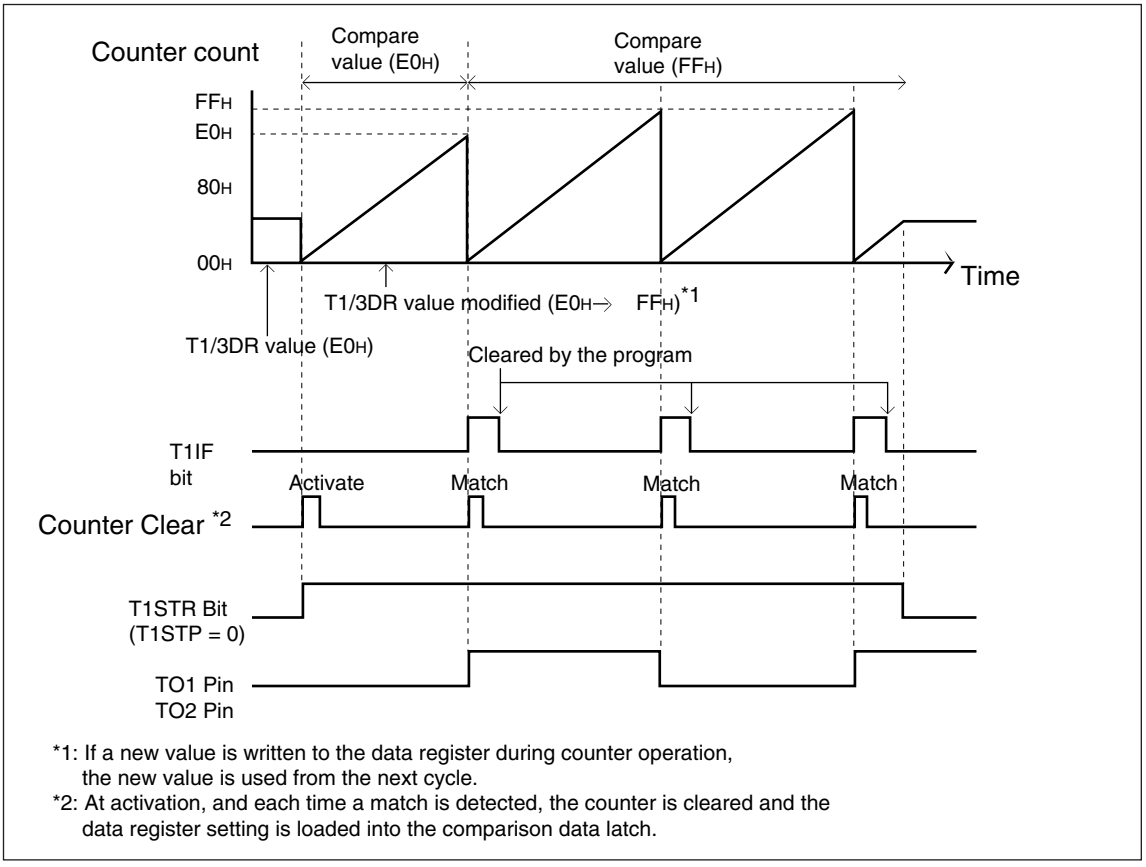
Figure 10.5-2 Interval timer function (timer 12/22) settings



On activation in 8-bit mode, the counter starts counting-up from "00_H", on the rising edge of the selected count clock. Eventually, the count in the counter will match the value set in the data register (comparison data latch). When this occurs, the timer control register interrupt request flag bit (T1/3CR: T1IF) is set to "1", and the counter starts counting-up again from "00_H". If using timer 11/21, the output of the square wave output control circuit is inverted when a match is detected; and if square wave output is enabled (T1/3CR: T1OS1, T1OS0 = values other than "00_B"), a square wave is output at the TO1 or TO2 pin.

Figure 10.5-3"Operation of interval timer (timer 12/22)" shows the interval timer function operation in the 8 bit mode.

Figure 10.5-3 Operation of interval timer (timer 12/22)

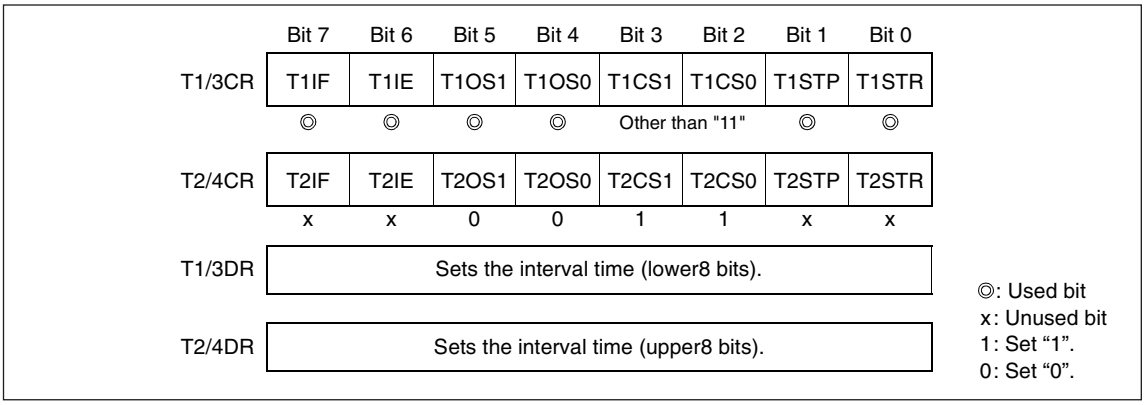


*1: If a new value is written to the data register during counter operation, the new value is used from the next cycle.
*2: At activation, and each time a match is detected, the counter is cleared and the data register setting is loaded into the comparison data latch.

● 16-bit mode

Figure 10.5-4"Interval timer function settings (16-bit mode) " shows the settings required to operate the interval timer function in the 16-bit mode.

Figure 10.5-4 Interval timer function settings (16-bit mode)



In 16-bit mode, the timer 11/21 control register (T1/3CR) controls the timer. The timer 12/22 control register (T2/4CR) must, however, still be initialized. The data to be compared with the 16-bit counter is set in both data registers: the upper 8 bits in T2/4DR and the lower 8 bits in T1/3DR. All 16 bits of the counter are cleared simultaneously. All other operation in the 16-bit mode is the same as Timer 11/21 operation in 8-bit mode.

10.6 Operation of Counter Function

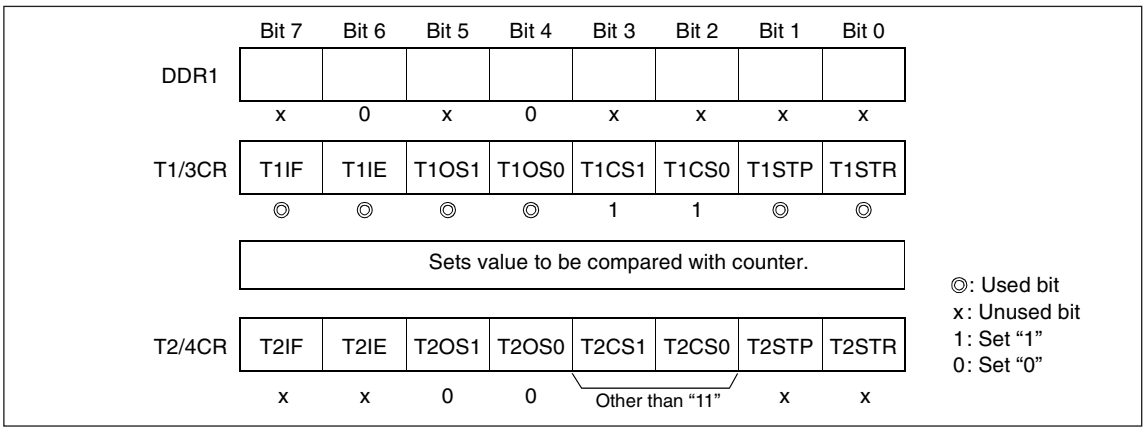
This section describes the operations of the counter function of the 8/16-bit timer/counter.

■ Operation of counter function

● 8-bit mode

Figure 10.6-1 "Counter function settings (8-bit mode) " shows the settings required to operate the timer 11/21 as the counter function in the 8-bit mode.

Figure 10.6-1 Counter function settings (8-bit mode)

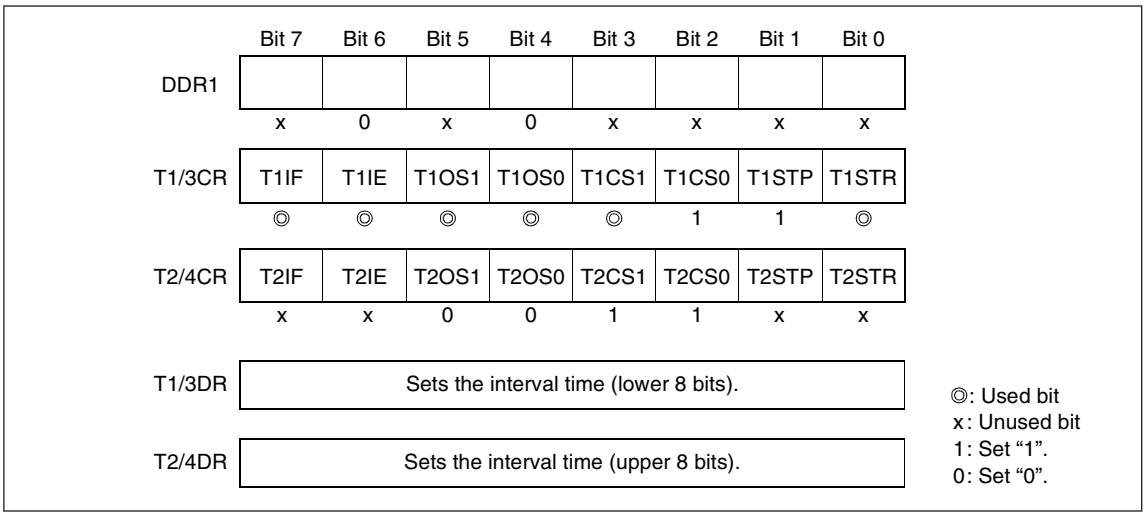


Counter operation in the 8-bit mode is the same as interval timer operation of timer 11/21 in 8-bit mode, except that an external clock is used in lieu of the internal clock.

● 16-bit mode

Figure 10.6-2"Counter function settings (16-bit mode) " shows the settings required to operate the counter function in the 16-bit mode.

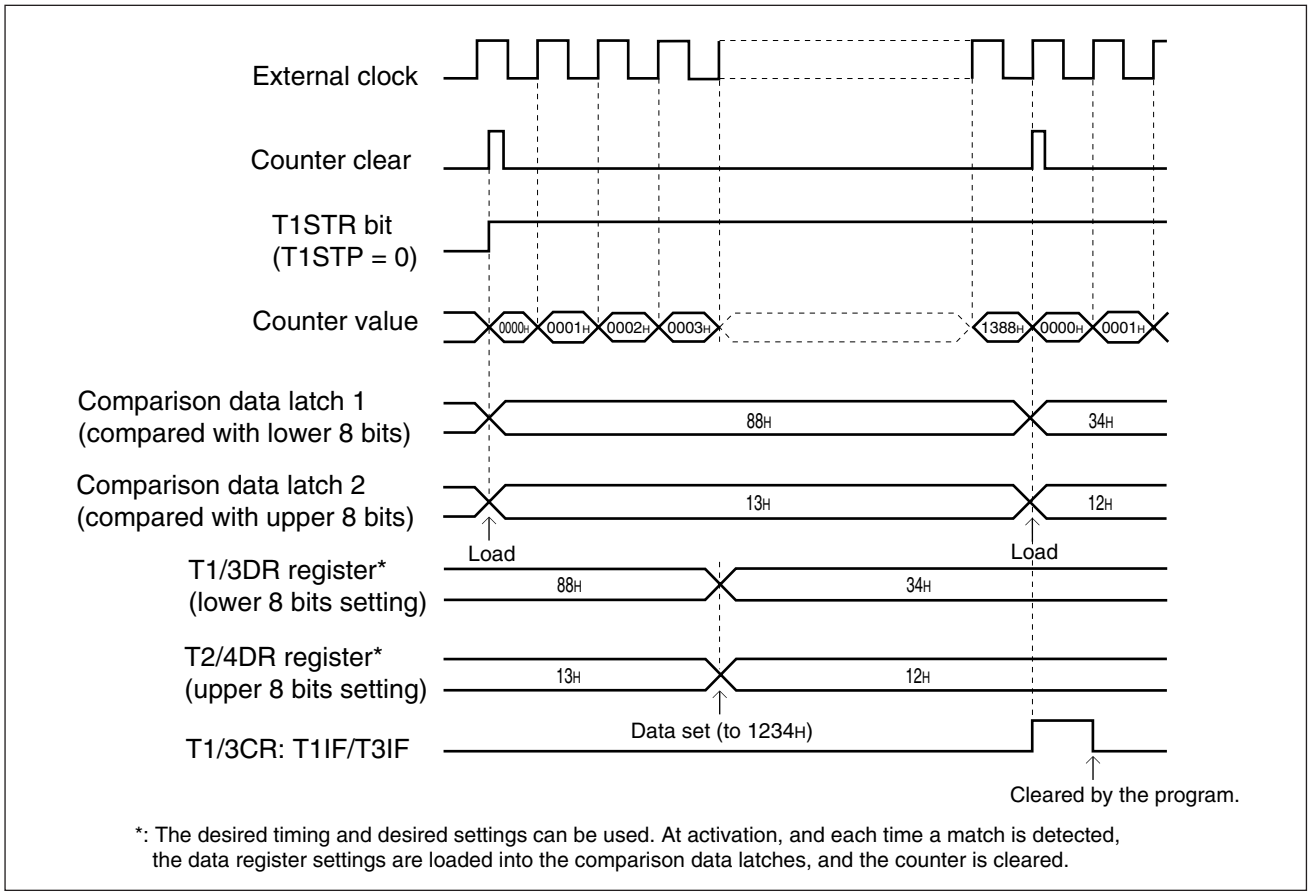
Figure 10.6-2 Counter function settings (16-bit mode)



Counter operation in the 16-bit mode is the same as interval timer operation in 16-bit mode, except that an external clock is used in lieu of the internal clock.

Figure 10.6-3"Operation of counter function in 16-bit mode" shows the counter function operation in the 16-bit mode.

Figure 10.6-3 Operation of counter function in 16-bit mode



Note:

When the counter value during operation is read out in 16-bit mode, always read it twice, and verify that a proper value is got before using it.

10.7 Operation of the Square Wave Output Initial Setting Function

The Square wave output can be set to the desired initial value using the timer 11/21 control register (T1/3CR).

■ Operation of square wave output initial setting function

The square wave output can be set to the desired initial value by the program, but this can be done only when the timer operation is stopped (T1/3CR: T1STR = 0).

Figure 10.7-1 "Square wave output initial setting equivalent circuit" shows an equivalent circuit for the square wave output control circuit initial setting. To perform the initial setting, follow the procedure in Table 10.7-1 "Square wave output initial setting procedure (T1/3CR register)". The operation of the square wave output when this is done is as shown in Figure 10.7-2 "Square wave output initial setting operation".

Figure 10.7-1 Square wave output initial setting equivalent circuit

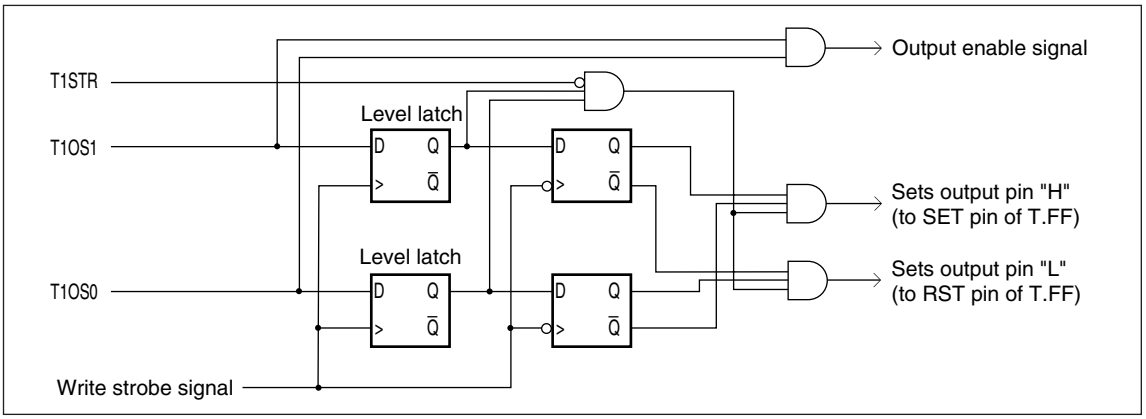


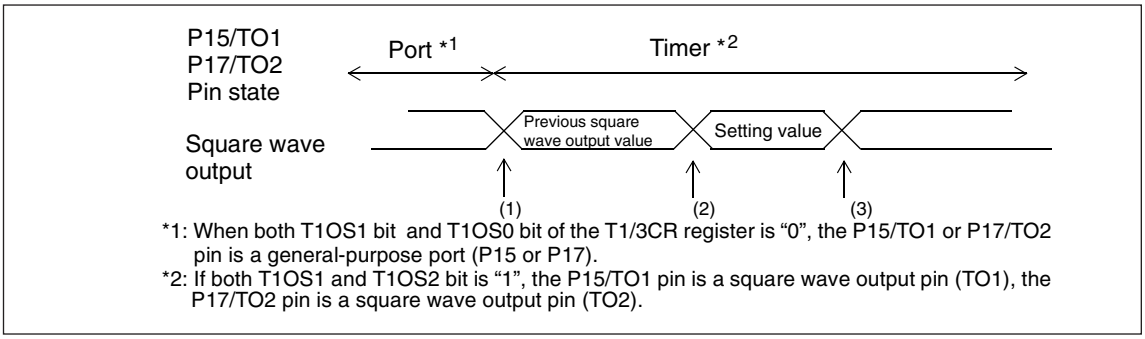
Table 10.7-1 Square wave output initial setting procedure (T1/3CR register)

Step	Settings and Operation
(1)	To set the square wave output pin (TO1/TO2) "L", set the square wave output control bits (T1/3CR: T1OS1, T1OS0) first to "01_B", then to "11_B". To set the TO1 pin "H", set the bits to "10_B", then "11_B." **Reference:** Until the bits are written to "11_B", the circuit simply holds the latched value, and the square wave output, TO1 or TO2 pin level remains in its current or previous state.
(2)	If the square wave output control bits (T1OS1, T1OS0) are written to "11B" and the timer operation stopped (T1STR = 0), the TO1 pin will output the level corresponding to the level latch value (initial value). This can also be accomplished by setting T1OS1, T1OS0, and T1STR simultaneously. If the timer activation bit is set (T1STR= 1), the counter will start.

Table 10.7-1 Square wave output initial setting procedure (T1/3CR register)

Step	Settings and Operation
(3)	The square wave output is inverted each time the counter value matches the data register settings.

Figure 10.7-2 Square wave output initial setting operation



*1: When both T1OS1 bit and T1OS0 bit of the T1/3CR register is “0”, the P15/TO1 or P17/TO2 pin is a general-purpose port (P15 or P17).
*2: If both T1OS1 and T1OS2 bit is “1”, the P15/TO1 pin is a square wave output pin (TO1), the P17/TO2 pin is a square wave output pin (TO2).

10.8 Operation of 8/16-bit Timer/Counter Stop and Restart

This section describes the operation of stop and restart operation functions of the 8/16-bit timer/counter.

■ Timer stop and restart

Operation is described for timer 11/21 only. Timer 12/22, however, operates the same way.

Timer 11/21 is stopped and restarted using the timer 11/21 control register stop and start bits (T1/3CR: T1STP and T1STR).

- To start the counter after clearing it, with "0" in T1STR bit

Set T1STP, T1STR bit to "01_B". On the T1STR bit rising edge, the counter will be cleared and start counting.

- To temporarily stop the counter and then resume counting (without clearing the counter)

First stop the counter by setting T1STP, T1STR to "11_B", then set T1STP, T1STR to "01_B" to resume counting where you left off.

Table 10.8-1 "Timer stop and restart" lists the timer states for each T1STP, T1STR bit, and operation when the timer is activated from that state (T1STP, T1STR = 01_B).

Table 10.8-1 Timer stop and restart

T1STP (T2STP)	T1STR (T2STR)	Timer State	Timer operation when counter is activated (T1STP, T1STR="01_B") from the state shown at the left
0	0	Counter stopped	Counter cleared and starts counting
0	1	Counter operating	Counter keeps on operating as-is.
1	0	Counter stopped	Counter cleared and starts counting
1	1	Counter temporarily stopped	Counter resumes counting without being cleared

10.9 States in Each Mode during 8/16-bit Timer/Counter Operation

This section describes the operation of the 8/16-bit timer/counter when the device changes to sleep or stop mode or an operation halt request occurs during operation.

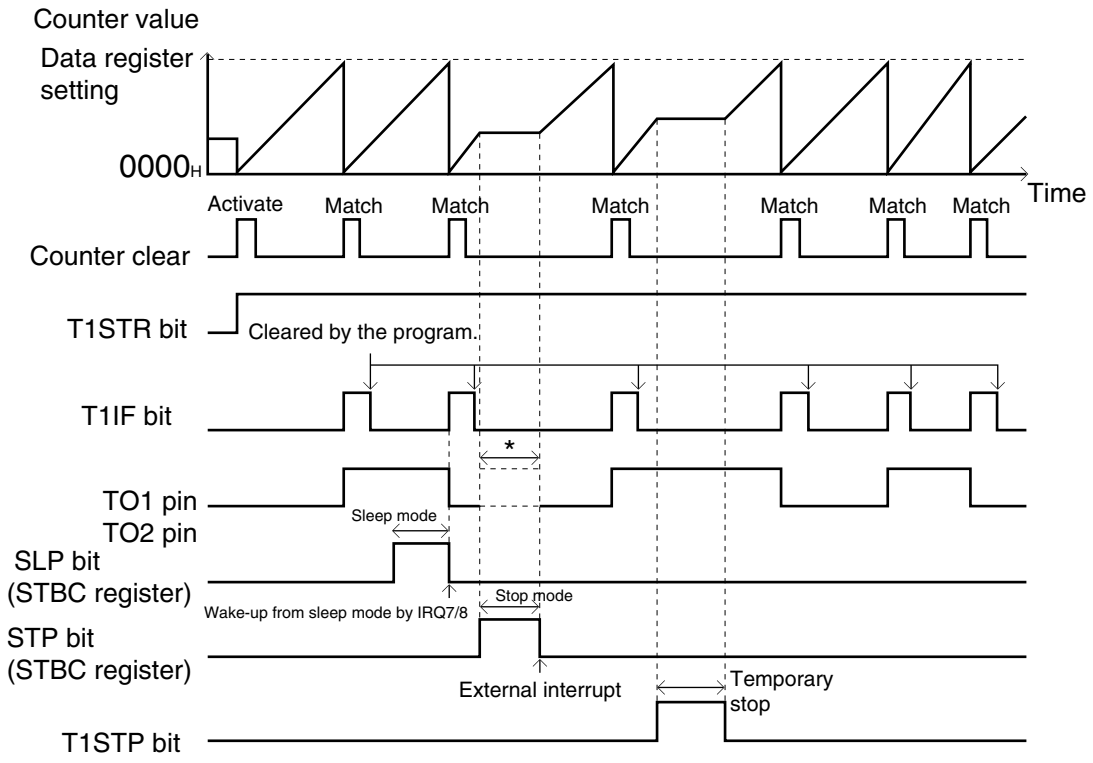
■ Operation during standby mode, or operation halt

Figure 10.9-1 "Counter operation during standby mode, or operation halt" shows the counter value state when the device changes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or counter function (for timer 11/21).

The counter halts and maintains its current value when the device changes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. therefore, the first interval time or external clock count is not correct value. Always initialize the 8/16-bit timer/counter after wake-up from stop mode.

Operation when entering or exiting watch mode (STBC: TMD = 1) is the same as when entering or exiting stop mode. When the counter is stopped temporarily (T1STP = 1), it holds the count it had when it was stopped. When it is restarted (T1STP= 0), it resumes counting from the count at which it was stopped.

Figure 10.9-1 Counter operation during standby mode, or operation halt



*: The TO1 or TO2 pin goes to the high-impedance state during stop mode if the pin state specification bit in the standby control regisater (STBC: SPL) is "1" and the TO1 or TO2 pin is not set to with a pull-up resistor.
When the SPL bit is "0", the pin maintains its value prior to changing to stop mode.

10.10 Notes on Using 8/16-bit Timer/Counter

This section lists points to note when using the 8/16-bit timer/counter. Those points can apply on Timer 11/21 and 12/22.

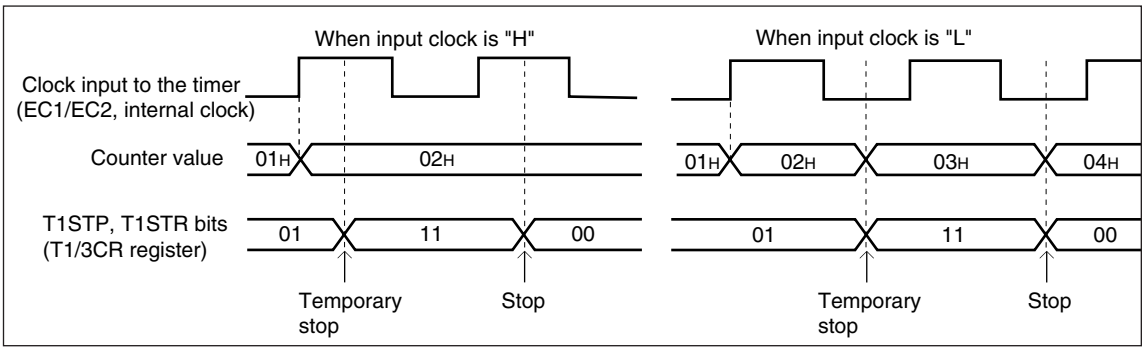
■ Notes on using 8/16-bit timer/counter

● Notes when counter is stopped

This information is described for timer 11/21, but the same information applies to timer 12/22.

As shown in Figure 10.10-1 "Operation when timer stop bit is used", if the clock is "L" when T1STP temporarily stops the timer, the count will be incremented by 1. This may also occur if the input clock is "L" after a temporary stop, and the T1STP and T1STR bits are both written to "00_B" simultaneously. When using the T1STP bit to temporarily stop the counter, first read out the counter value; then write T1STP bit to "0".

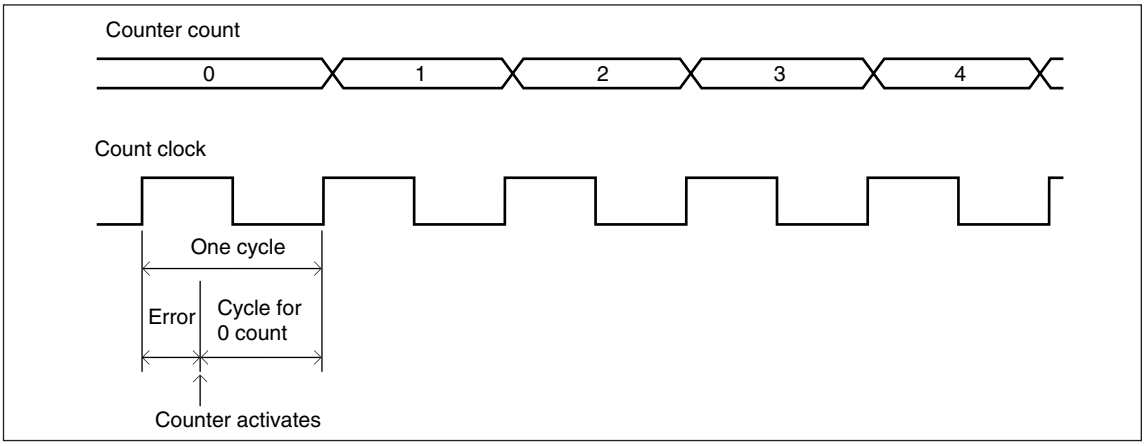
Figure 10.10-1 Operation when timer stop bit is used



● Error

Activating of by program 8/16-bit timer/counter is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter to match the register setting may be shorter than the theoretical time by a maximum of one cycle of the count clock. Figure 10.10-2 "Error on starting counter operation" shows the error that occurs on standing counter operation.

Figure 10.10-2 Error on starting counter operation



● Using one 8-bit channel

When 8/16-bit timer/counter timer 11/21 only is used in the 8-bit mode, before doing so, first set the timer count clock select bits of the timer 12/22 control register (T2/4CR: T2CS1, T2CS0) to some state other than "11_B". Failure to do so may result in faulty operation.

● Notes on setting by program

- When the 8/16-bit timer/counter Timer 11/21 only is used in the 16-bit mode, the timer 12/22 control register count clock select bits (T2/4CR: T2CS1, T2CS0) should always be set to "11_B", and bits 5 and 4, the unused bits (T2/4CR: T2OS1, T2OS0) to "00_B".
- In 16-bit mode, when the counter value is read out during operation, always read it twice and verify that a proper value is got before using it.
- While the timer is operating (T1/3CR: T1STR = 1), performing the initial state setting will not immediately cause the square wave output level to change. The output state will be initialized when the timer stops.
- Interrupt processing cannot return if the interrupt request flag bit (T1/3CR: T1IF, T2/4CR: T2IF) is "1" and the interrupt request enable bit is enabled (T1/3CR: T1IE= "1", T2/4CR: T2IE= "1"). Always clear the interrupt request flag bit.
- The interrupt request flag bit (T1/3CR: T1IF or T2/4CR: T2IF) is not set if the counter is disabled by the timer start bit (T1/3CR: T1STR=0 or T2/4CR: T2CR =0) at the same time as an interrupt source is generated.

10.11 Program Examples for 8/16-bit Timer/Counter

This section gives a program examples for 8/16-bit timer/counter.

■ Program example for interval timer function

● Processing description

- Using timer 11 only, in 8-bit mode, generates repeated interval timer interrupts at 20 ms intervals.
- Outputs a square wave to the TO1 pin that inverts after each interval time.
- With a main clock master oscillation F_{CH} of 12.5 MHz, and the highest speed main clock selected by the speed-shift function (1 instruction cycle time = $4/F_{CH}$), and with an internal clock period of 512 t_{inst} selected as the count clock, the T1DR setting for an interval of approximately 20 ms is calculated as follows:

  T1DR register value = 20 ms/(512 x 4/12.5 MHz) - 1 = 122.07 ($7A_H$)

● Coding example

```
T2CR    EQU    0018H           ;Address of the Timer 12 control register
T1CR    EQU    0019H           ;Address of the Timer 11 control register
T2DR    EQU    001AH           ;Address of the Timer 12 data register
T1DR    EQU    001BH           ;Address of the Timer 11 data register

T1IF    EQU    T1CR:7          ;Define the timer 11 interrupt request flag bit.

ILR2    EQU    007CH           ;Address of the interrupt level setting register 2

INT_V   DSEG   ABS
        ORG    0FFEAH
IRQ7    DW     WARI            ;Set interrupt vector.
        ENDS
;----------Main program------------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
                               ;Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ;Disable interrupts.
        MOV    ILR2,#10111111B ;Set interrupt priority to level 2.
        MOV    T2CR,#00000010B ;Clear timer 12 interrupt request flag, disable
                                interrupt request output, set other than 16-bit
                                mode, stop operation.
        MOV    T1CR,#00011000B ;Clear timer 11 interrupt request flag, initialize
                                square wave output "L", select 512 tinst, and
                                stop operation.
        MOV    T1DR,#7AH       ;Set value compared with the counter value
                                (interval time).
        MOV    T1CR,#11111001B ;Timer 11 interrupt request, clear counter, and
                                start timer.
        SETI                   ;Enable CPU interrupts.
         :
         :
;----------Interrupt Program-------------------------------------------------------
WARI    CLRB   T1IF            ;Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
 ;-------------------------------------------------------------------------------
        END
```

■ Program example for pulse counter function

● Processing description

- Using timer 11 and timer 12 in 16-bit mode, count external clocks input to the EC1 pin, and generate an interrupt once for each 5000 clocks (1388_H).
- Shows a sample program (READ16) for reading out the count in the 16-bit counter, while the counter is counting.

● Coding example

```
DDR1    EQU    0003H           ;Address of the Port 1 data direction register
T2CR    EQU    0018H           ;Address of the Timer 12 control register
T1CR    EQU    0019H           ;Address of the Timer 11 control register
T2DR    EQU    001AH           ;Address of the Timer 12 data register
T1DR    EQU    001BH           ;Address of the Timer 11 data register
T1IF    EQU    T1CR:7          ;Define the timer 11 interrupt request flag bit.
ILR2    EQU    007CH           ;Address of the interrupt level setting register 3
NT_V    DSEG   ABS
        ORG    0FFEAH
IRQ7    DW     WARI            ;Set interrupt vector.
ENDS
;----------Main program-------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
                               ;Stack pointer (SP) etc. are already initialized.
        MOV    DDR1,#00000000B ;Set EC pin as an input.
        CLRI                   ;Disable interrupts.
        MOV    ILR2,#10111111B ;Set interrupt level 2.
        MOV    T1DR,#088H      ;Set lower 8 bits of counter comparison value.
        MOV    T2DR,#013H      ;Set upper 8 bits of counter comparison value.
        MOV    T2CR,#00001100B ;Set timer 12 to 16-bit mode.
        MOV    T1CR,#01001101B ;Clear timer 11 interrupt request flag, enable
                                interrupt request output, set P15/TO1 as
                                general-purpose port (P15), select external clock,
                                clear counter, and start operation.
        SETI                   ;Enable CPU interrupts.
;----------Data read subroutine-----------------------------------------------
 READ16 MOVW   A,T2DR          ;16-bit read, T1DR + T2DR.
        MOVW   A,T2DR          ;16-bit read, T1DR + T2DR, save old value in
                                T register.
        CMPW   A               ;Check first and second reads, compare A and
                                T registers.
        BEQ    RET16           ;If match, return.
        XCHW   A,T
        INCW   A               ;Old value + 1
        CMPW   A
        BNE    READ16          ;If mismatch, read again.
RET16   RET
         :
;----------Interrupt program--------------------------------------------------
WARI    CLRB   T1IF            ;Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User process
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 11

EXTERNAL INTERRUPT 1 CIRCUIT (EDGE)

This chapter describes the functions and operation of the external interrupt circuit.

11.1 Overview of the External Interrupt 1 Circuit

The external interrupt circuit detects edges on the signals input to the four external interrupt pins and generates the corresponding interrupt requests to the CPU.

■ Functions of the external interrupt circuit

The function of the external interrupt circuit is to detect specified edges on signals input to the external interrupt pins and to generate interrupt requests to the CPU. These interrupts can cancel standby mode and return the device to the normal operating state (main-run state).

External interrupt pins: 4 pins (P10/INT10 - P13/INT13)

External interrupt sources: Input of a specified edge (rising edge or falling edge) on the signal input to an external interrupt pin.

Interrupt control: Output of external interrupt requests is enabled or disabled by the interrupt request enable bits in external interrupt control registers 1 and 2 (EIC1, EIC2).

Interrupt flags: Detection of specified edges sets the external interrupt request flag bits in external interrupt control registers 1 and 2 (EIC1, EIC2).

Interrupt request: Separate interrupt requests are generated for each external interrupt source (IRQ0, IRQ1, IRQ2, IRQ3).

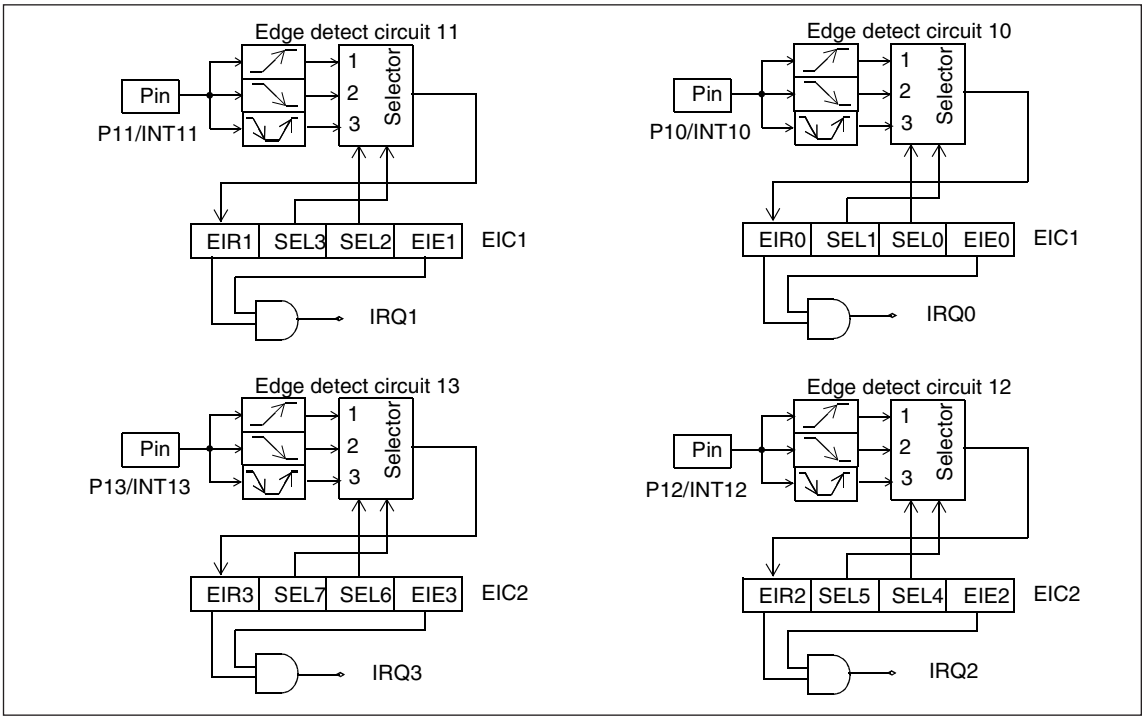
11.2 Block Diagram of the External Interrupt 1 Circuit

The external interrupt circuit consists of four blocks with the same functions.
Each block contains the following two elements:

- **Edge detect circuits (10 - 13)**
- **External interrupt control registers 1, 2 (EIC1, EIC2)**

■ Block diagram of the external interrupt 1 circuit

Figure 11.2-1 Block diagram of the external interrupt 1 circuit



● Edge detect circuit

If the polarity of an edge on the input signal to one of the external interrupt pins (INT10 - INT13) matches the edge polarity specified for the pin in the EIC1 or EIC2 register (SEL0 - SEL7), the edge detect circuit sets the corresponding external interrupt request flag bit (EIR0 - EIR3) to "1".

● EIC1 and EIC2 registers

The EIC1 and EIC2 registers are used for operations such as edge selection, enabling or disabling interrupt requests, and checking interrupt requests.

11.3 Structure of the External Interrupt 1 Circuit

This section describes the pins, pin block diagram, registers, and interrupt sources of the external interrupt circuit.

■ External interrupt 1 circuit pins

The external interrupt circuit has four external interrupt pins.The external interrupt pins can function either as external interrupt inputs (hysteresis inputs) or general I/O ports.

Although the P10/INT10 - P13/INT13 pins continuously function as external interrupt input, the external interrupt circuit does not output interrupts if output of interrupt requests is disabled for the pin. The pin states can be read directly from the port data register (PDR1) at any time.

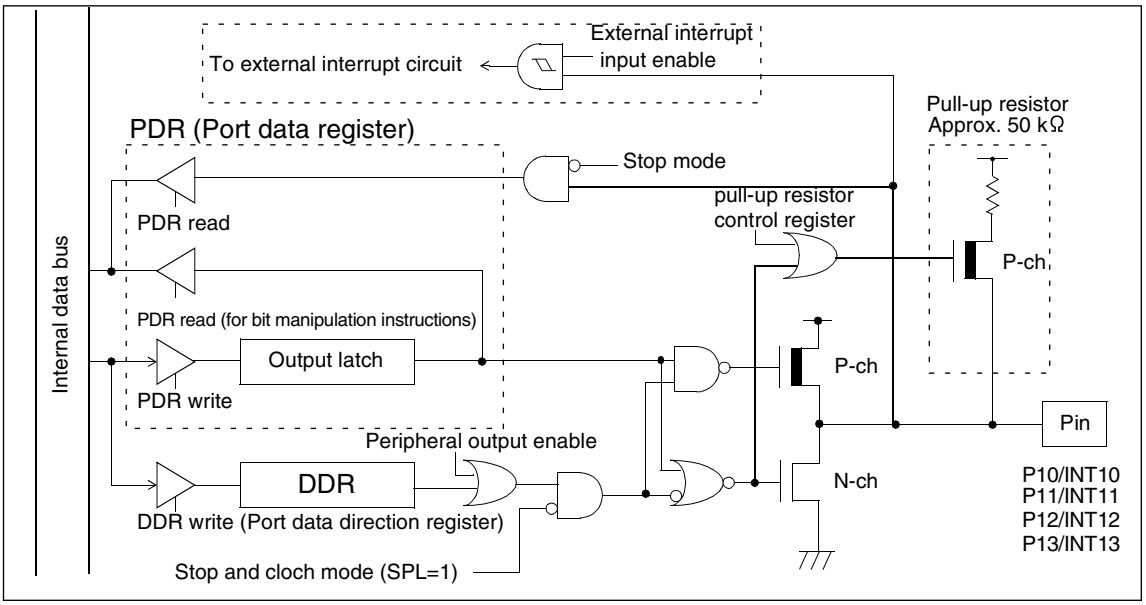
Table 11.3-1 External interrupt 1 circuit pins

External Interrupt Pin	When Used as an External Interrupt Input (Output of Interrupt Requests Enabled)	When Used as an Input-Only Port (Output of Interrupt Requests Disabled)
P10/INT10	INT10 (EIC1: EIE0=1)	P10 (EIC1: EIE0=0)
P11/INT11	INT11 (EIC1: EIE1=1)	P11 (EIC1: EIE1=0)
P12/INT12	INT12 (EIC2: EIE2=1)	P12 (EIC2: EIE2=0)
P13/INT13	INT13 (EIC2: EIE3=1)	P13 (EIC2: EIE3=0)

INT10 - INT13: The external interrupt circuit generates the interrupt request for the pin when an edge of the specified polarity is input.

■ Block diagram of the external interrupt 1 circuit (EDGE) pins

Figure 11.3-1 Block diagram of the external interrupt 1 circuit (EDGE) pins



Reference:

When pull-up resistor is selected in pull-up register in stop and clock mode (SPL=1). The states of these pins are in “H” level (pull-up state) rather than high impedance. However, during reset, pull-up is unavailable and will be in high impedance state.

■ External interrupt circuit 1 registers

Figure 11.3-2 External interrupt 1 circuit registers

EIC1 (External interrupt control register 1)

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0030H	EIR1	SEL3	SEL2	EIE1	EIR0	SEL1	SEL0	EIE0	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	← IRQ1 INT11 →				← IRQ0 INT10 →				

EIC2 (External interrupt control register 2)

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0031H	EIR3	SEL7	SEL6	EIE3	EIR2	SEL5	SEL4	EIE2	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	← IRQ3 INT13 →				← IRQ2 INT12 →				

R/W : Readable and writable

■ External interrupt 1 circuit interrupt sources

IRQ0:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT10 when output of interrupt requests is enabled.

IRQ1:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT11 when output of interrupt requests is enabled.

IRQ2:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT12 when output of interrupt requests is enabled.

IRQ3:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT13 when output of interrupt requests is enabled.

11.3.1 External Interrupt Control Register 1 (EIC1)

External interrupt control register 1 (EIC1) is used to select the edge polarity and to control interrupts for external interrupt pins INT10, INT11.

■ External interrupt control register 1 (EIC1)

Figure 11.3-3 External interrupt control register 1 (EIC1)

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0030H	EIR1	SEL3	SEL2	EIE1	EIR0	SEL1	SEL0	EIE0	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

EIE0	INT10 Interrupt request enable bit
0	Disable output of interrupt requests.
1	Enable output of interrupt requests.

SEL1	SEL0	INT10 Edge polarity selection bits
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edge

EIR0	INT10 External interrupt request flag bit	
	Read	Write
0	The specified edge has not been input.	Clear this bit.
1	The specified edge has been input.	No effect. The bit does not change.

EIE1	INT11 Interrupt request enable bit
0	Disable output of interrupt requests.
1	Enable output of interrupt requests.

SEL3	SEL2	INT11 Edge polarity selection bits
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edge

EIR1	INT11 External interrupt request flag bit	
	Read	Write
0	The specified edge has not been input.	Clear this bit.
1	The specified edge has been input.	No effect. The bit does not change.

R/W : Readable and writable
: Initial value

Table 11.3-2 External interrupt control register 1 (EIC1) bits

Bit		Function
bit7	EIR1: INT11 External interrupt request flag bit	• This bit is set to "1" when the edge selected by INT11 edge polarity selection (SEL3, SEL2) is input to external interrupt pin INT11. • An interrupt request is output when both this bit and INT11 interrupt request enable bit (EIE1) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
bit6 bit5	SEL3, SEL2: INT11 Edge polarity mode selection bits	• Control the mode of the input edge polarity of INT11 pin • Writing "00_B" uses no edge detection, "01_B" uses rising-edge mode, "10_B" uses falling-edge mode or "11_B" uses both-edge mode. • Always write "0" into EIR1 when changing this bit.
bit4	EIE1: INT11 Interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is output when both this bit and INT11 external interrupt request flag bit (EIR1) are "1".
bit3	EIR0: INT10 External interrupt request flag bit	• This bit is set to "1" when the edge selected by INT10 edge polarity selection (SEL1,SEL0) is input to external interrupt pin INT10. • An interrupt request is output when both this bit and INT10 interrupt request enable bit (EIE0) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
bit2 bit1	SEL1, SEL0: INT10 Edge polarity mode selection bits	• Control the mode of the input edge polarity of INT10 pin • Writing "00_B" uses no edge detection, "01_B" uses rising-edge mode, "10_B" uses falling-edge mode or "11_B" uses both-edge mode. • Always write "0" into EIR0 when changing this bit.
bit0	EIE0: INT10 Interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is output when both this bit and INT10 external interrupt request flag bit (EIR0) are "1".

11.3.2 External Interrupt Control Register 2 (EIC2)

External interrupt control register 2 (EIC2) is used to select the edge polarity and to control interrupts for external interrupt pins INT12, INT13.

■ External interrupt control register 2 (EIC2)

Figure 11.3-4 External interrupt control register 2 (EIC2)

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0031H	EIR3	SEL7	SEL6	EIE3	EIR2	SEL5	SEL4	EIE2	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

EIE2	INT12 Interrupt request enable bit
0	Disable output of interrupt requests.
1	Enable output of interrupt requests.

SEL5	SEL4	INT12 Edge polarity selection bits
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edge

EIR2	INT12 External interrupt request flag bit	
	Read	Write
0	The specified edge has not been input.	Clear this bit.
1	The specified edge has been input.	No effect. The bit does not change.

EIE3	INT13 Interrupt request enable bit
0	Disable output of interrupt requests.
1	Enable output of interrupt requests.

SEL7	SEL6	INT 13 Edge polarity selection bits
0	0	No edge detection
0	1	Rising edge
1	0	Falling edge
1	1	Both edge

EIR3	INT13 External interrupt request flag bit	
	Read	Write
0	The specified edge has not been input.	Clear this bit.
1	The specified edge has been input.	No effect. The bit does not change.

R/W : Readable and writable
□ : Initial value

Table 11.3-3 External interrupt control register 2 (EIC2) bits

Bit		Function
bit7	EIR3: INT13 External interrupt request flag bit	• This bit is set to "1" when the edge selected by INT13 edge polarity selection (SEL7,SEL6) is input to external interrupt pin INT13. • An interrupt request is output when both this bit and INT13 interrupt request enable bit (EIE3) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
bit6 bit5	SEL7, SEL6: INT13 Edge polarity mode selection bits	• Control the mode of the input edge polarity of INT13 pin • Writing "00_B" uses no edge detection, "01_B" uses rising-edge mode, "10_B" uses falling-edge mode or "11_B" uses both-edge mode. • Always write "0" into EIR3 when changing this bit.
bit4	EIE3: INT13 Interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is output when both this bit and INT13 external interrupt request flag bit (EIR3) are "1".
bit3	EIR2: INT12 External interrupt request flag bit	• This bit is set to "1" when the edge selected by INT12 edge polarity selection (SEL5, SEL4) is input to external interrupt pin INT12. • An interrupt request is output when both this bit and INT12 interrupt request enable bit (EIE2) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
bit2 bit1	SEL5, SEL4: INT12 Edge polarity mode selection bits	• Control the mode of the input edge polarity of INT12 pin • Writing "00_B" use no edge detection, "01_B" uses rising-edge mode, "10_B" uses falling-edge mode or "11_B" uses both-edge mode. • Always write "0" into EIR2 when changing this bit.
bit0	EIE2: INT12 Interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is output when both this bit and INT12 external interrupt request flag bit (EIR2) are "1".

11.4 External Interrupt 1 Circuit Interrupts

The external interrupt circuit can generate interrupt requests when it detects a specified edge on the signal input to an external interrupt pin.

■ Interrupts when the external interrupt 1 circuit is operating

On detecting a specified edge on an external interrupt input, the external interrupt circuit sets the corresponding external interrupt request flag bit (EIC1, EIC2 : EIR0 - EIR3) to "1". An interrupt request to the CPU (IRQ0 - IRQ3) is generated at this time if the corresponding interrupt request enable bit is enabled (EIC1, EIC2,: EIE0 - EIE3 = "1"). Always write "0" to the corresponding external interrupt request flag bit in the interrupt processing routine to clear the interrupt request.

Note:

When enabling interrupts (EIE0 - EIE3 = “1”) after exit of a reset, always clear the corresponding external interrupt request flag bit (EIR0 - EIR3 = “0”) at the same time.

Interrupt processing cannot return if the external interrupt request flag bit is “1” and the interrupt request enable bit is enabled. Always clear the external interrupt request flag bit.

Reference:

Cancelling stop mode using an interrupt is only possible using the external interrupt circuit.

- An interrupt request is generated immediately if the external interrupt request flag bit is "1" when the interrupt request enable bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for the external interrupt circuit interrupts

Table 11.4-1 Register and vector table for external interrupt 1 circuit interrupts

Interrupt	Interrupt Level Setting Register			Vector Table Address	
	Register	Setting Bits		Upper	Lower
IRQ0	ILR1 ($007B_H$)	L01 (bit1)	L00 (bit0)	$FFFA_H$	$FFFB_H$
IRQ1	ILR1 ($007B_H$)	L11 (bit3)	L10 (bit2)	$FFF8_H$	$FFF9_H$
IRQ2	ILR1 ($007B_H$)	L21 (bit5)	L20 (bit4)	$FFF6_H$	$FFF7_H$
IRQ3	ILR1 ($007B_H$)	L31 (bit7)	L30 (bit6)	$FFF4_H$	$FFF5_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupts.

■ Notes when changing edge polarity selection

When changing the edge polarity for INT10 to INT13, always write "0" into the corresponding EIR bits. This will prevent accidentally creating an interrupt.

11.5 Operation of the External Interrupt 1 Circuit

The external interrupt circuit can detect a specified edge on a signal input to an external interrupt pin.

■ Operation of the external interrupt 1 circuit

Figure 11.5-1 "External interrupt 1 circuit settings" shows the settings required to operate the external interrupt 1 circuit

Figure 11.5-1 External interrupt 1 circuit settings

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
EIC1	EIR1	SEL3	SEL2	EIE1	EIR0	SEL1	SEL0	EIE0
	◎	◎	◎	◎				
EIC2	EIR3	SEL7	SEL6	EIE3	EIR2	SEL5	SEL4	EIE2

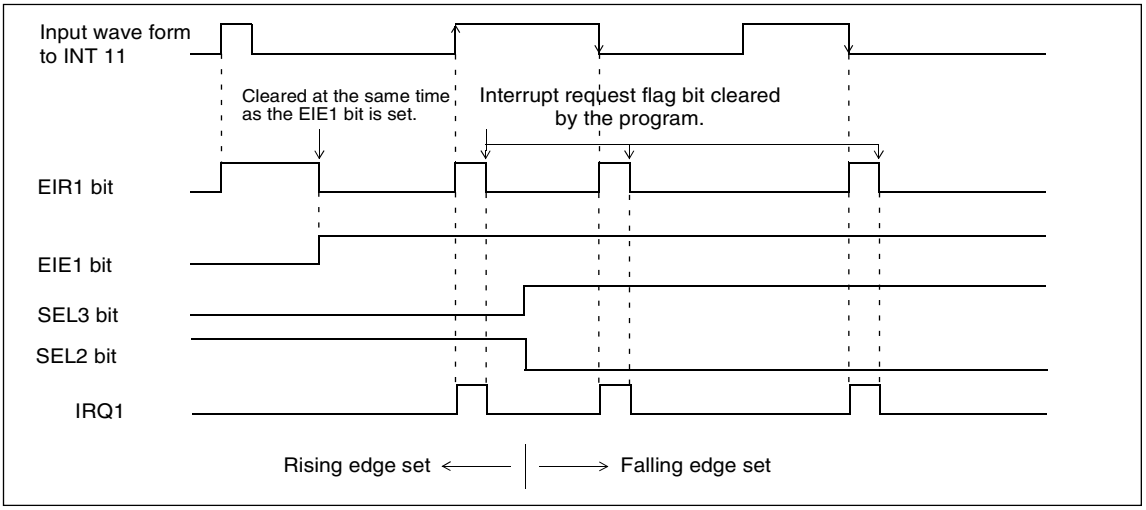
◎ : Used bit

If the polarity of an edge on the input signal to one of the external interrupt pin (INT11) matches the edge polarity specified for the pin in the external interrupt control register (EIC1: SEL2, SEL3), the external interrupt circuit sets the external interrupt request flag bit (EIC1: EIR1) to "1".

The external interrupt request flag bit is set when the edge polarity match occurs, regardless of the value of the interrupt request enable bit (EIC1: EIE1).

Figure 11.5-2 "External interrupt 1 circuit operation (INT11)" shows the operation when an external interrupt is input to the INT11 pin.

Figure 11.5-2 External interrupt 1 circuit operation (INT11)



Refernce:

The pin state can be read directly from the port data register (PDR1), even when used as an external interrupt input.

11.6 Program Example for the External Interrupt 1 Circuit

This section gives a program example for the external interrupt circuit.

■ Program example for the external interrupt 1 circuit

● Processing description

- Generates interrupts on detecting a rising edge on pulses input to the INT11 pin.

● Coding example

```
EIC1    EQU    0030H           ;External interrupt control register 1

EIR1    EQU    EIC1:7          ;Defines the external interrupt request flag bit.
SEL2    EQU    EIC1:5          ;Defines the edge polarity selection bit.
EIE1    EQU    EIC1:4          ;Defines the interrupt request enable bit.

ILR1    EQU    007BH           ;Set interrupt level setting register 1.

INT_V   DSEG   ABS             ;[DATA SEGMENT]
        ORG    0FFF8H
IRQ1    DW     WARI            ;Set interrupt vector (INT11).
INT_V   ENDS
;-----Main program-----------------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
                               ;Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ;Disable interrupts.
        CLRB   EIR1            ;Clear interrupt request flag.
        MOV    ILR0,#11110111B ;Set interrupt level (level 1).
        SETB   SEL2            ;Select rising edge.
        SETB   EIE1            ;Enable output of interrupt requests.
        SETI                   ;Enable interrupts.
   :
;-----Interrupt processing routine-------------------------------------------------
WARI    CLRB   EIR1            ;Clear INT1 interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;----------------------------------------------------------------------------------
        END
```

CHAPTER 12

EXTERNAL INTERRUPT 2 CIRCUIT (LEVEL)

This chapter describes the functions and operation of the external interrupt 2 circuit (level).

12.1 Overview of External Interrupt 2 Circuit

The external interrupt circuit 2 detects the level of the signals input to the five external interrupt pins and generates the interrupt requests to the CPU.

■ External interrupt 2 circuit function (level detection)

The external interrupt 2 circuit function detects the signals of the "L" levels input to the external interrupt pins and to generate interrupt request to the CPU. These interrupts can wake up the CPU from standby mode and change the device to the normal operating state (main-run or sub-run mode).

External interrupt pins: 5 pins (P50/$\overline{\text{INT20}}$ to P54/$\overline{\text{INT24}}$)

External interrupt sources: "L" level signal input to an external interrupt pin.

Interrupt control: Enables or disables to input external interrupt controlled by external interrupt 2 control register (EIE2)

Interrupt flag: IRQ flag bit of external interrupt 2 flag register (EIF2). Flag set when there is an IRQ.

Interrupt request: IRQ4 is generated if any enabled external interrupt pin goes LOW.

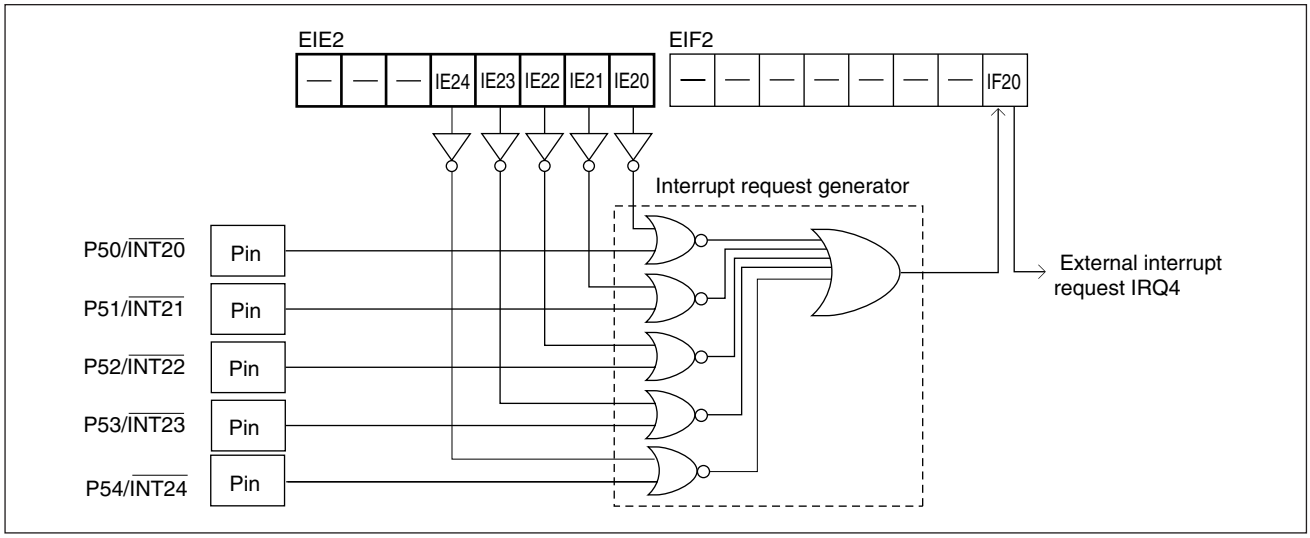
12.2 Block Diagram of External Interrupt 2 Circuit

The external interrupt circuit 2 consists of the following three blocks:

- **Interrupt request generator**
- **External interrupt 2 control register (EIE2)**
- **External interrupt 2 flag register (EIF2)**

■ Block diagram of external interrupt circuit 2

Figure 12.2-1 Block diagram of external interrupt 2 circuit



● Interrupt request generator

The interrupt request generator generates CPU interrupt requests based on signals input at external interrupt pins ($\overline{INT20}$ to $\overline{INT24}$) and the external interrupt enable bits.

● EIE2 register

External interrupt input enable bits (IE20 to IE24) enable/disable "L" level signals input at the corresponding external interrupt input pins.

● EIF2 register

The interrupt request flag bit of this register (IF20) is used to hold (and clear) interrupt request signals.

12.3 Structure of External Interrupt 2 Circuit

This section describes the pins, pin block diagram, registers, and interrupt sources of the external interrupt 2 circuit.

■ External interrupt 2 circuit pins

The external interrupt 2 circuit uses five external interrupt pins.

The external interrupt pins can function either as external interrupt inputs (hysteresis inputs) or as general-purpose I/O ports.

When P50/$\overline{\text{INT20}}$ to P54/$\overline{\text{INT24}}$ pins are set as inputs in the port 5 data direction register (DDR5), and the corresponding external interrupt inputs are enabled in the external interrupt 2 control register (EIE2) they operate as external interrupt input pins ($\overline{\text{INT20}}$ to $\overline{\text{INT24}}$). When they are being used as the input port, the pin states can be read from the port data register (PDR5) at any time.

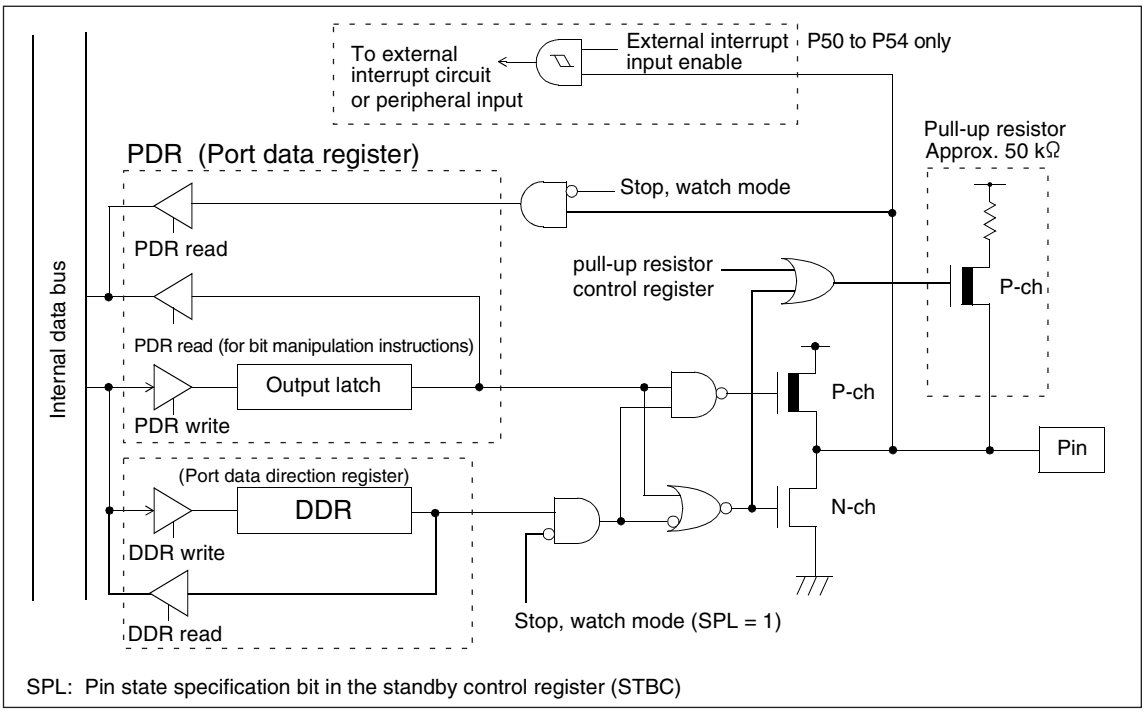
Table 12.3-1 "External interrupt 2 circuit pins" lists the external interrupt 2 circuit pins.

Table 12.3-1 External interrupt 2 circuit pins

External interrupt pin	When used as an external interrupt input (Interrupt Input Enabled)	When used as General-purpose I/O port (Interrupt Input Disabled)
P50/$\overline{\text{INT20}}$	$\overline{\text{INT20}}$ (EIE2:IE20=1)	P50 (EIE2:IE20=0)
P51/$\overline{\text{INT21}}$	$\overline{\text{INT21}}$ (EIE2:IE21=1)	P51 (EIE2:IE21=0)
P52/$\overline{\text{INT22}}$	$\overline{\text{INT22}}$ (EIE2:IE22=1)	P52 (EIE2:IE22=0)
P53/$\overline{\text{INT23}}$	$\overline{\text{INT23}}$ (EIE2:IE23=1)	P53 (EIE2:IE23=0)
P54/$\overline{\text{INT24}}$	$\overline{\text{INT24}}$ (EIE2:IE24=1)	P54 (EIE2:IE24=0)

■ Block diagram of external interrupt 2 circuit pins

Figure 12.3-1 Block diagram of external interrupt 2 circuit pins



Reference:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF". However, during reset pull-up is unavailable and will be Hi-Z.

■ External interrupt 2 circuit registers

Figure 12.3-2 External interrupt 2 circuit registers

EIE2 (External interrupt 2 control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0032H	—	—	—	IE24	IE23	IE22	IE21	IE20	XXX00000B
				R/W	R/W	R/W	R/W	R/W	

EIF2 (External interrupt 2 flag register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0033H	—	—	—	—	—	—	—	IF20	XXXXXXX0B
								R/W	

R/W : Readable and writable
— : Unused
X : Indeterminate

■ External interrupt 2 circuit interrupt sources

IRQ4:

IRQ4 is generated if any one of external interrupt pins $\overline{INT20}$ to $\overline{INT24}$ goes to "L" with a "1" in the external interrupt input enable bit for that pin.

12.3.1 External Interrupt 2 Control Register (EIE2)

The external interrupt 2 control register (EIE2) is used to enable/disable input of external interrupt pins ($\overline{INT20}$ to $\overline{INT24}$).

■ External interrupt 2 control register (EIE2)

Figure 12.3-3 External interrupt 2 control register (EIE2)

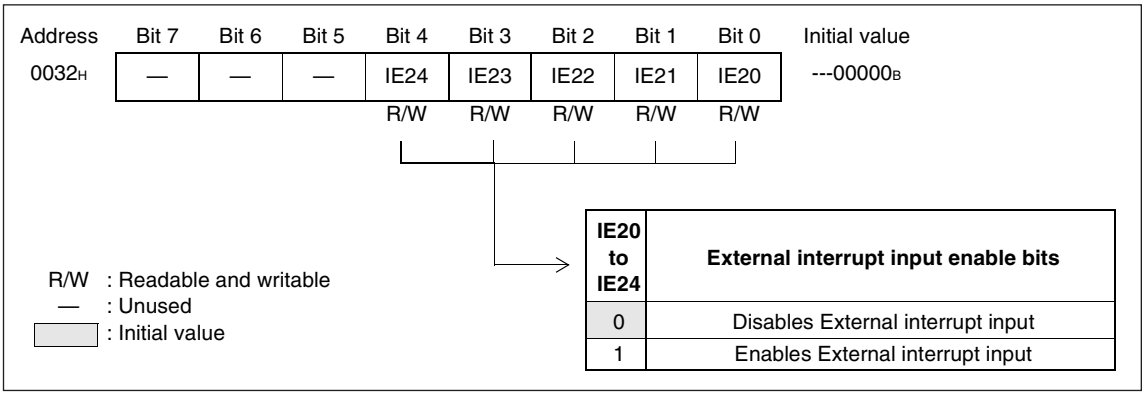


Table 12.3-2 External interrupt 2 control register (EIE2) bits vs. pins

	Bit	Pin
Bit 7	-	-
Bit 6	-	-
Bit 5	-	-
Bit 4	IE24	$\overline{INT24}$
Bit 3	IE23	$\overline{INT23}$
Bit 2	IE22	$\overline{INT22}$
Bit 1	IE21	$\overline{INT21}$
Bit 0	IE20	$\overline{INT20}$

Table 12.3-3 External interrupt 2 control register (EIE2) bits

Bit		Function
Bit 7 Bit 6 Bit 5	Unused bits	• The read value is undefined. • Writing has no effect on the operation.
Bit 4 Bit 3 Bit 2 Bit 1 Bit 0	IE23 to IE20: External interrupt enable bits	• These bits enable/disable input of external interrupts at external interrupt pins $\overline{\text{INT20}}$ to $\overline{\text{INT24}}$. • Setting these bits to"1" puts the corresponding pin into its external interrupt input mode, and enables input of external interrupts at the pin. • Conversely, a"0" in the bit allows the pin to function in its general-purpose port mode and inhibits input of interrupts at the pin. **Reference:** • When using a pin for external interrupts, set it as an input by writing its bit in the port 5 data direction register (DDR5) to"0". • The state of the pin can always be read directly out of the port 5 data register (PDR5) regardless of the sate of this external interrupt enable bit.

12.3.2 External Interrupt 2 Flag Register (EIF2)

External Interrupt 2 flag register (EIF2) is used to hold the IRQ state when a level interrupt has been detected, and clear the interrupt.

■ External interrupt 2 flag register (EIF2)

Figure 12.3-4 External interrupt 2 flag register (EIF2)

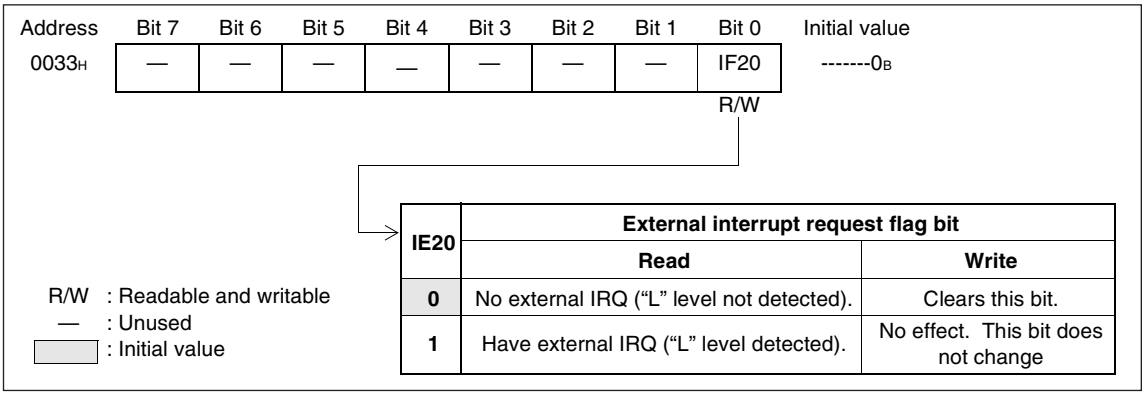


Table 12.3-4 External interrupt 2 flag register (EIF2) bits

Bit		Function
Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1	Unused Bits	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 0	IF20: External interrupt request flag bit	• This bit is set to "1" when a "L" is detected at an enabled external input pin ($\overline{INT20}$ to $\overline{INT24}$). • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value. **Reference:** Writing "0" to the external interrupt enable bits of the external interrupt 2 control register (EIE2: IE20 to IE24) simply disables the corresponding external interrupt input; it does not clear the interrupt request. IRQ4 will continue to be sent to the CPU until it is cleared by writing "0" to the IF20 bit.

12.4 External Interrupt 2 Circuit Interrupt

The external interrupt 2 circuit interrupt trigger event is the detection of a “L” level at the external interrupt pin.

■ Interrupts for external interrupt 2 circuit operation

If a "L" is detected at an enabled external interrupt pin, the external interrupt request flag bit (EIF2: IF20) is set to "1," and an interrupt request (IRQ4) to the CPU is generated. Write "0" to the IF20 bit in the interrupt processing routine to clear the interrupt request.

Once the external interrupt request flag bit (IF20) is set to "1," IRQ4 continues to be asserted as long as the flag set. Disabling the interrupt input by writing the IE bit (IE20 to IE24) of the EIE2 register to "0" will not clear the interrupt request. Always clear the IF20 bit.

Also, if the external interrupt pin stays "L", writing "0" to the IF20 bit without disabling the external interrupt input will not clear the interrupt either, because IF20 will immediately be set again by the "L" pin. After an interrupt request is generated, then, either the input must be disabled, or the external IRQ signal de-asserted.

Note:

When enabling interrupts of CPU after wake-up from a reset, clear the IF20 bit in advance.

Reference:

Wake-up from stop mode by an interrupt is possible using only the external interrupt circuit 1 and 2.

■ Register and vector table for external interrupt 2 circuit interrupts

Table 12.4-1 Registers and vector table for external interrupt 2 circuit interrupts

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ4	ILR2 ($007C_H$)	L41 (Bit 1)	L40 (Bit 0)	$FFF2_H$	$FFF3_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupts operation.

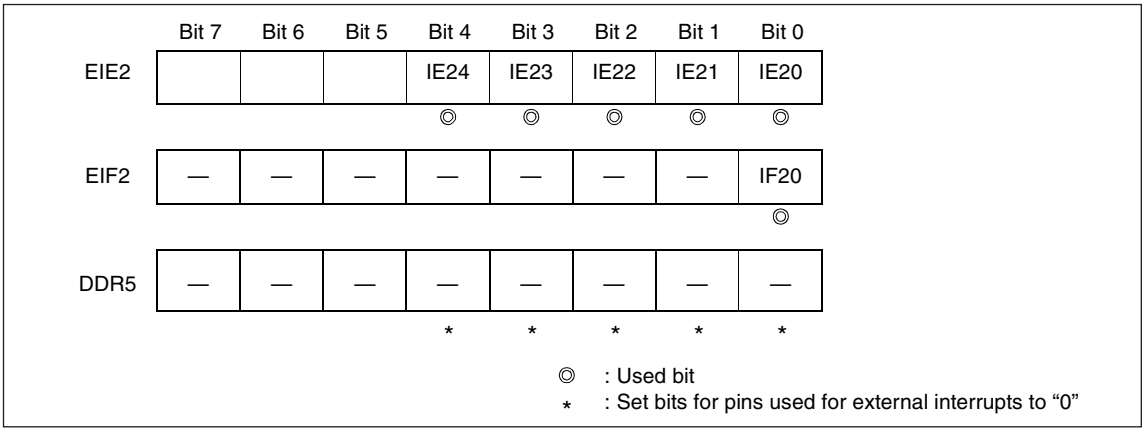
12.5 Operation of External Interrupt 2 Circuit

The external interrupt 2 circuit sends an interrupt request to the CPU when it detects a “L” at one of its external interrupt pins.

■ Operation of external interrupt 2 circuit

Figure 12.5-1 "External interrupt 2 circuit settings" shows the settings required to operate the external interrupt 2 circuit .

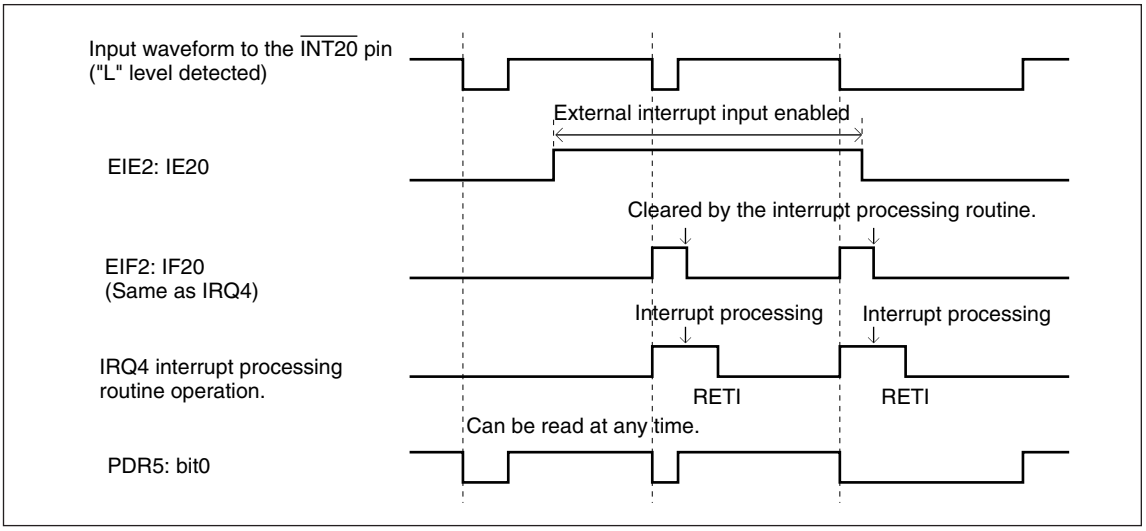
Figure 12.5-1 External interrupt 2 circuit settings



If a "L" is applied to one of the $\overline{INT20}$ to $\overline{INT24}$ pins with the corresponding external interrupt input enable bit (IE20 to IE24) in the "enable" state, the circuit sends an IRQ4 interrupt request to the CPU.

Figure 12.5-2 "Operation of external interrupt 2 ($\overline{\text{INT20}}$)" shows external interrupt 2 circuit operation (for a signal received at INT20).

Figure 12.5-2 Operation of external interrupt 2 ($\overline{\text{INT20}}$)



Reference:

The pin state can be read directly from the port data register (PDR5) even when the pin is being used as an external interrupt input.

12.6 Program Example for External Interrupt 2 Circuit

This section gives a Program example for the external interrupt 2 circuit.

■ Program example for external interrupt 2 circuit

● Processing description

- Generates interrupts on detecting a "L" input the $\overline{\text{INT20}}$ pin.

● Coding example

```
DDR5    EQU    0011H           ;Address of port 5 data direction register
EIE2    EQU    0032H           ;Address of external interrupt 2 control register
EIF2    EQU    0033H           ;Address of external interrupt 2 flag register

IF20    EQU    EIF2:0          ;Define the external interrupt request flag bit.

ILR2    EQU    007CH           ;Address of the set interrupt level setting
                                register 2
INT_V   DSEG   ABS             ;[DATA SEGMENT]
        ORG    0FFF2H
IRQ4    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-----------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
                               ;Stack pointer (SP) etc. are already initialized.
         :                     ;
        CLRI                   ;Disable interrupts.
        CLRB   IF20            ;Clear external interrupt request flag.
        MOV    ILR2,#11111110B ;Set interrupt priority (level 2).
        MOV    DDR5,#00000000B ;Set P50/INT20 pin as input.
        MOV    EIE2,#00000001B ;Enable external interrupt input at INT20 pin.
        SETI                   ;Enable interrupts.
         :
;-----Interrupt processing routine-------------------------------------------
WARI    MOV    EIE2,#00000000B ;Disable external interrupt input at INT20 pin.
        CLRB   IF20            ;Clear external interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;-----------------------------------------------------------------------------
        END
```

CHAPTER 13

A/D CONVERTER

This chapter describes the functions and operation of the A/D converter.

13.1 Overview of A/D Converter

The A/D converter can function as an 10-bit successive approximation type A/D converter. The function selects one input signal from the eight analog input pin channels and can be activated either by software, by an internal clock, or by 8/16-bit timer 3.

■ A/D Conversion Function

The A/D conversion function converts the analog voltage (input voltage) input to an analog input pin to an 10-bit digital value.

- Selects one input from eight analog input pins.
- Conversion speed is 60 instruction cycles (19.2µs for a 12.5 MHz source oscillation).
- Generates an interrupt when A/D conversion completes.
- Conversion completion can also be determined by software.

The following methods are available to activate A/D conversion:

- Activation by software
- Continuous activation by a timebase timer output (divide-by-2^8main clock source oscillation)
- Continuous activation by 8/16-bit timer 3 output.

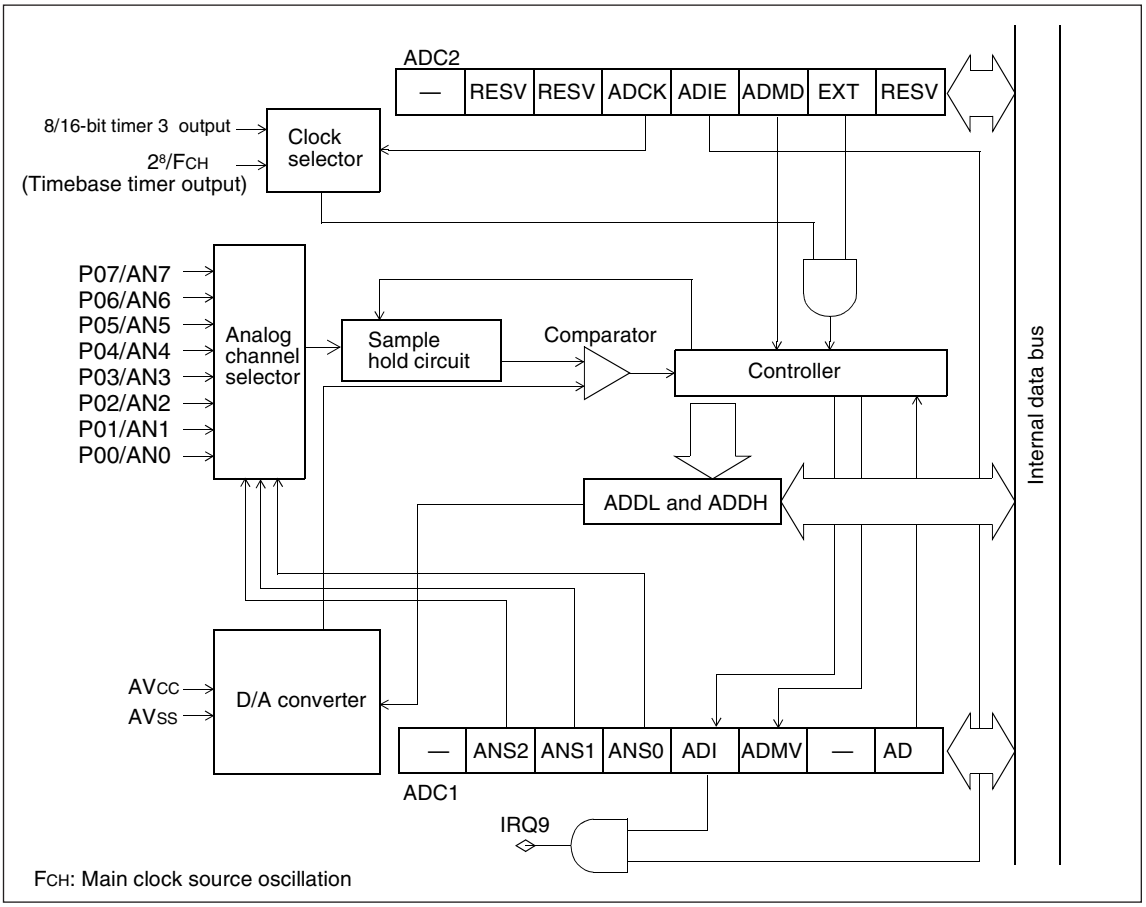
13.2 Block Diagram of A/D Converter

The A/D converter consists of the following nine blocks:

- **Clock selector (input clock selector for A/D converter activation)**
- **Analog channel selector**
- **Sample hold circuit**
- **D/A converter**
- **Comparator**
- **Controller**
- **A/D data register (ADDL and ADDH)**
- **A/D control register 1 (ADC1)**
- **A/D control register 2 (ADC2)**

■ Block diagram of A/D converter

Figure 13.2-1 Block diagram of A/D converter



- **Clock selector**

  Selects the clock used to activate the A/D conversion when continuous activation is enabled (ADC2: EXT = "1").

- **Analog channel selector**

  Selects one of the eight analog input channels.

- **Sample hold circuit**

  Holds the input voltage selected by the analog channel selector. The circuit samples and holds the input voltage immediately after the A/D conversion is activated. This allows A/D conversion to proceed without being affected by input voltage fluctuation.

- **D/A converter**

  Generates the voltage corresponding to the value set in the ADDL and ADDH register.

- **Comparator**

  Compares the sampled and held input voltage with the output voltage of the D/A converter, and determines which voltage is higher or lower.

- **Controller**

  For the A/D conversion function, the controller successively determines the value of each bit of the ADDL and ADDH register, starting from the most significant bit and proceeding to the least significant bit, based on the greater-than/less-than signal from the comparator. When conversion is complete, the circuit sets the interrupt request flag bit (ADC1: ADI).

- **ADDL and ADDH register**

  Stores the A/D conversion result for the A/D conversion function.

- **ADC1 register**

  The ADC1 register is used to enable or disable each function, select the analog input pin, check status, and control interrupts.

- **ADC2 register**

  The ADC2 register is used to select the input clock, enable or disable interrupts, and select functions.

■ A/D converter power supply voltage

● AV_{CC}

The A/D converter power supply pin. Use at the same voltage as V_{CC}. When high A/D conversion resolution is required, take measures to ensure that the noise on V_{CC} is not present on AV_{CC}, or use a separate power supply. Connect this pin to the power supply, even if the A/D converter is not used.

● AV_{SS}

The A/D converter ground pin. Use at the same voltage as V_{SS}. When high A/D conversion accuracy is required, take measures to ensure that the noise on V_{SS} is not present on AV_{SS}. Connect this pin to ground (GND), even if the A/D converter is not used.

13.3 Structure of A/D Converter

This section describes the pins, pin block diagrams, registers, and an interrupt source for the A/D converter.

■ A/D converter pins

The A/D converter function uses the P00/AN0 to P07/AN7 pins. These pins can function as either general-purpose I/O port (P00 to P07), or the analog input pins (AN0 to AN7).

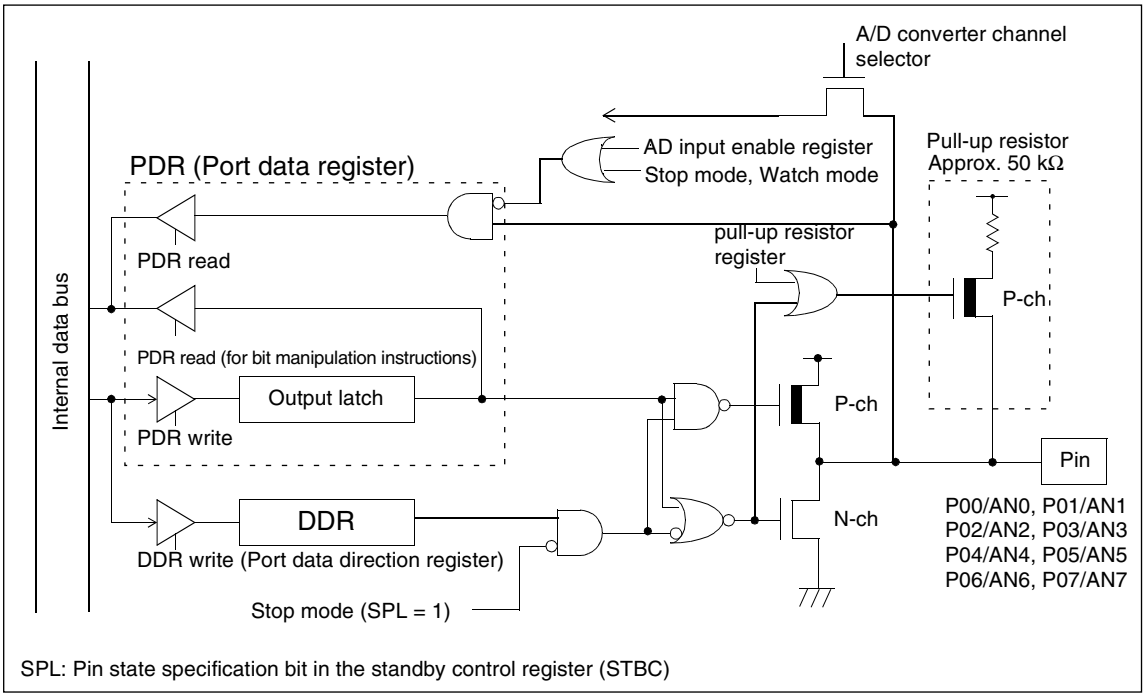
● AN0 to AN7:

The analog voltages to be converted (A/D conversion function) are applied to these pins.

To select the analog input function for one of these pins, you set the corresponding bit of the port data direction register (DDR0) to "1," to turn off the port output transistor, then set the corresponding bit of the A/D input enable register (ADER) to "1" and set the analog input channel select bits (ADC1: ANS0 to ANS2) to select the pin as the analog input channel. Pins that are not needed for analog inputs can still be used as general I/O port pins, even while the A/D converter is being used.

■ Block diagram of A/D converter pin

Figure 13.3-1 Block diagram of P00/AN0 to P07/AN7 pins



SPL: Pin state specification bit in the standby control register (STBC)

■ A/D converter registers

Figure 13.3-2 A/D converter registers

ADC1 (A/D control register 1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0020H	—	ANS2	ANS1	ANS0	ADI	ADMV	RESV	AD	-00000X0B
		R/W	R/W	R/W	R/W	R	R/W	R/W	

ADC2 (A/D control register 2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0021H	—	RESV	RESV	ADCK	ADIE	ADMD	EXT	RESV	-0000001B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	

ADDH (A/D data register H)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0022H	—	—	—	—	—	—	D9	D8	------XXB
							R	R	

ADDL (A/D data register L)

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Initial value
0023H	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXXB
	R	R	R	R	R	R	R	R	

R/W : Readable and writable
R : Read-only
— : Unused
X : Indeterminate

■ A/D converter interrupt source

IRQ9:

The A/D converter generates an interrupt request if an interrupt request output is enabled (ADC2: ADIE = "1") when A/D conversion completes.

13.3.1 A/D Control Register 1 (ADC1)

A/D control register 1 (ADC1) is used to enable or disable the functions, select the analog input pin, and check the status of the A/D converter.

■ A/D control register 1 (ADC1)

Figure 13.3-3 A/D control register 1 (ADC1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0020H	—	ANS2	ANS1	ANS0	ADI	ADMV	RESV	AD	-00000X0B
		R/W	R/W	R/W	R/W	R	R/W	R/W	

AD	A/D converter activation bit
	Only applies when software activation is specified (ADC2: EXT = "0").
0	Does not activate the A/D conversion
1	Activates the A/D conversion

RESV	Reserved bit
	Always write "0" to this bit

ADMV	Conversion-in-progress flag bit
0	Conversion not currently in progress
1	Conversion currently in progress

ADI	Interrupt request flag bit	
	Read	Write
0	Conversion not complete.	Clears this bit.
1	Conversion completes.	No effect. The bit does not change.

ANS2	ANS1	ANS0	Analog input channel selection bits
0	0	0	P00/AN0 pin
0	0	1	P01/AN1 pin
0	1	0	P02/AN2 pin
0	1	1	P03/AN3 pin
1	0	0	P04/AN4 pin
1	0	1	P05/AN5 pin
1	1	0	P06/AN6 pin
1	1	1	P07/AN7 pin

R/W : Readable and writable
R : Read-only
— : Unused
: Initial value

Table 13.3-1 A/D control register 1 (ADC1) bits

Bit		Function
Bit 7	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 6 Bit 5 Bit 4	ANS2 to ANS0: Analog input channel selection bits	• These bits select which of the AN0 to AN7 pins to use as the analog input pin. When using software activation (ADC2: EXT = "0"), these bits can be modified at the same time as activating the A/D conversion (AD = "1"). **Note:** Disable general-purpose port output corresponding to the analog input pin, and set it as analog input pin by ADER. Do not modify these bits when the ADMV bit is set to "1". **Reference:** Pins not used as analog inputs can be used as general-purpose ports.
Bit 3	ADI: Interrupt request flag bit	• For the A/D conversion function: This bit is set to "1" when the A/D conversion is completed. • An interrupt request is output when both this bit and the interrupt request enable bit (ADC2: ADIE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 2	ADMV: Conversion-in-progress flag bit	• This bit indicates whether or not the A/D converter is currently performing a conversion. The bit is set to "1" when a conversion is in progress. **Reference:** This bit is read-only. The write value has no meaning and has no effect on the operation.
Bit 1	RESV: Reserved bit	**Note:** Always write "0" to this bit.
Bit 0	AD: A/D converter activation bit	• This bit activates the A/D conversion by software. • Writing "1" to this bit activates the A/D conversion when continuous activation is not specified (ADC2: EXT = "0"). **Notes:** • Writing "0" to this bit does not stop the A/D conversion. The read value is always "0". • This bit has no meaning when continuous activation is specified.

13.3.2 A/D Control Register 2 (ADC2)

A/D control register 2 (ADC2) is used to select the A/D converter functions, select the input clock, enable or disable interrupts and continuous activation, and check the status of the A/D converter.

■ A/D control register 2 (ADC2)

Figure 13.3-4 A/D Control register 2 (ADC2)

Address	Bit 7	Bit 6	Bit5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0021H	—	RESV	RESV	ADCK	ADIE	ADMD	EXT	RESV	-0000001B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	

RESV	Reserved bit
	Always write “1” to this bit.

EXT	Continuous activation enable bit
0	Activates by the AD bit in the ADC1 register.
1	Activates continuously by the clock selected in the ADCK bit.

ADMD	Function selection bit
0	A/D conversion function
1	prohibited

ADIE	Interrupt request enable bit
0	Disables interrupt request output.
1	Enables interrupt request output.

ADCK	Input clock selection bit
	Only applies when continuous activation is enabled (EXT = “1”).
0	Timebase timer output (divide-by-2^8 source oscillation)
1	8/16-bit timer 3 output

RESV	Reserved bit
	Always write “0” to these bits.

R/W : Readable and writable
— : Unused
: Initial value

Table 13.3-2 A/D control register 2 (ADC2) bits

Bit		Function
Bit 7	Unused bits	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 6 Bit 5	Reserved bits	**Note:** Always write "0" to these bits.
Bit 4	ADCK: Input clock selection bit	• This bit selects the input clock used to activate the A/D conversion when continuous activation is specified (EXT = "1"). Setting this bit to "0" selects the timebase timer output (divide-by-2^8 main clock source oscillation). Setting this bit to "1" selects output of 8/16-bit timer 3.
Bit 3	ADIE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. • An interrupt request is output when both this bit and the interrupt request flag bit (ADC1: ADI) are "1".
Bit 2	ADMD: Function selection bit	**Note:** Always write "0" to this bit.
Bit 1	EXT: Continuous activation enable bit	• This bit selects whether to activate the A/D conversion by software or to operate continuously synchronized with an input clock. • Setting this bit to "0" enables software activation by the A/D converter activation bit (ADC1: AD). Setting this bit to "1" enables continuous activation on the rising edge of the clock selected in the input clock selection bit (ADC2: ADCK).
Bit 0	RESV: Reserved bit	**Notes:** • Always write "1" to this bit. • The read value is always "1".

13.3.3 A/D Data Registers (ADDL and ADDH)

These registers stores in the 10 bits A/D conversion result for the A/D conversion function. (lower 8-bits in ADDL and upper 2-bits in ADDH)

■ A/D data registers (ADDL and ADDH)

Figure 13.3-5 "A/D data registers (ADDL and ADDH)" shows the bit structure of the A/D data registers (ADDL and ADDH).

Figure 13.3-5 A/D data registers (ADDL and ADDH)

ADDH

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0022_H	—	—	—	—	—	—	D9	D8	------XX_B
							R	R	

ADDL

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
0023_H	D7	D6	D5	D4	D3	D2	D1	D0	$XXXXXXXX_B$
	R	R	R	R	R	R	R	R	

R : Read-only
— : Unused
X : Indeterminate

● For A/D conversion function

The conversion result is decided approximately 60 instruction cycles after A/D conversion is activated. The data of conversion is stored in these registers. The values of these registers are indeterminate while A/D conversion is in progress. These registers are READ-ONLY for the A/D conversion function.

13.4 A/D Converter Interrupt

The A/D converter has the following interrupt:

- **Conversion completion for the A/D conversion function**

■ Interrupt for A/D conversion function

When A/D conversion completes, the interrupt request flag bit (ADC1: ADI) is set to "1". At this time, an interrupt request (IRQ9) to the CPU is generated if the interrupt request enable bit is enabled (ADC2: ADIE = "1"). Write "0" to the ADI bit in the interrupt processing routine to clear the interrupt request.

The ADI bit is set after completion of A/D conversion, regardless of the ADIE bit value.

Reference:

An interrupt request is generated immediately if the ADI bit is "1" when the ADIE bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for A/D converter interrupt

Table 13.4-1 Register and vector table for A/D converter interrupt

Interrupt	Interrupt level settings register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ9	ILR3 ($007D_H$)	L91 (Bit 3)	L90 (Bit 2)	$FFE8_H$	$FFE9_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupt.

13.5 Operation of A/D Converter

The A/D conversion functions of the A/D converter can be activated by software or can be activated continuously.

■ Activating A/D conversion function

● Software activation

Figure 13.5-1 "A/D Conversion function (software activation) settings" shows the settings required for software activation of the A/D conversion function.

Figure 13.5-1 A/D Conversion function (software activation) settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADC1	—	ANS2	ANS1	ANS0	ADI	ADMV	—	AD
		◎	◎	◎	◎	◎	0	1
ADC2	—	RESV	RESV	ADCK	ADIE	ADMD	EXT	RESV
		0	0	x	◎	0	0	1
ADDL	Stores the A/D conversion result.							
ADDH	Stores the A/D conversion result.							

◎ : Used bit
x : Unused bit
1 : Set “1”.
0 : Set “0”.

On activation, the A/D converter starts the operation of the A/D conversion function. The A/D conversion function can be reactivated while conversion is in progress.

● Continuous activation

Figure 13.5-2 "A/D Conversion function (continuous activation) settings" shows the settings required for continuous activation of the A/D conversion function.

Figure 13.5-2 A/D Conversion function (continuous activation) settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADC1	—	ANS2	ANS1	ANS0	ADI	ADMV	—	AD
		◎	◎	◎	◎	◎	0	x
ADC2	—	RESV	RESV	ADCK	ADIE	ADMD	EXT	RESV
		0	0	◎	◎	0	1	1
ADDL	Stores the A/D conversion value.							
ADCH	Stores the A/D conversion value.							

◎ : Used bit
x : Unused bit
1 : Set “1”.
0 : Set “0”.

When continuous activation is enabled, the rising edge of the selected input clock activates the A/D conversion, starting operation of the A/D conversion function. When continuous activation is disabled (ADC2: EXT = "0"), continuous activation halts but software activation is available.

■ Operation of A/D conversion function

The following describes the operation of the A/D converter. The A/D conversion requires approximately 60 instruction cycles from activation to completion.

1. On activation, A/D conversion sets the conversion-in-progress flag bit (ADC1: ADMV = "1") and connects the sample hold circuit to the specified analog input pin.
2. The internal sample hold capacitor captures the voltage at the analog input pin for approximately 16 instruction cycles. The capacitor holds the voltage until the A/D conversion completes.
3. The comparator compares the voltage captured by the sample hold capacitor with the A/D converter reference voltage starting from the most significant bit (MSB) and ending with the least significant bit (LSB), and transfers each bit sequentially to the ADDL and ADDH register.
4. When the complete result has been transferred to the ADDL and ADDH register, the conversion-in-progress flag bit is cleared (ADC1: ADMV = "0") and the interrupt request flag bit is set (ADC1: ADI = "1").

13.6 Notes on Using A/D Converter

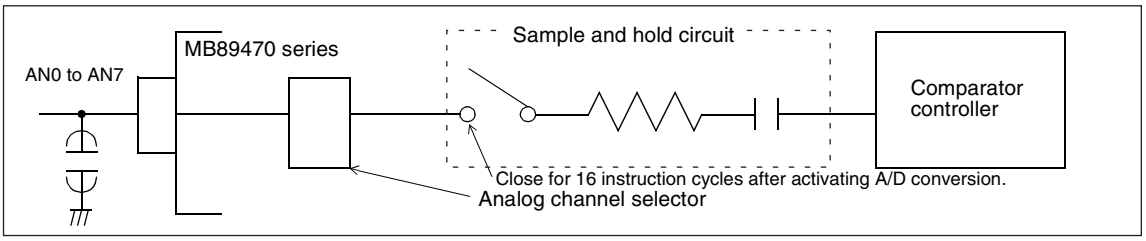
This section lists points to note when using the A/D converter.

■ Notes on using A/D converter

● Input impedance of analog input pins

The A/D converter contains a sample hold circuit as shown in Figure 13.6-1 "Analog input equivalent circuit"to fetch analog input voltage into the sample hold capacitor for 16 instruction cycles after activating A/D conversion For this reason, if the output impedance of the external circuit for the analog input is high, analog input voltage might not stabilize within the analog input sampling period. Therefore, it is recommended to keep the output impedance of the external circuit low (below 10 kΩ). Note that if the impedance cannot be kept low, it is recommended to connect an external capacitor of about 0.1 µF for the analog input pin.

Figure 13.6-1 Analog input equivalent circuit



● Notes on setting by program

- For the A/D conversion function, the ADDL and ADDH registers maintain previous values until the next A/D conversion is activated. However, the content of the ADDL and ADDH registers becomes indeterminate immediately after activating A/D conversion.
- Do not re-select the analog input channel (ADC1: ANS2 to ANS0) while the A/D conversion is operating. Particularly, when continuous activation is enabled, only perform such operations after disabling continuous activation (ADC2: EXT = "0") and waiting for the conversion-in-progress flag bit (ADC1: ADMV) to go to "0".
- A reset or activation of stop mode stops the A/D converter and initializes all registers.
- Interrupt processing cannot return if the interrupt request flag bit (ADC1: ADI) is "1" and the interrupt request enable bit is enabled (ADC2: ADIE = "1"). Always clear the ADI bit.

● Note on interrupt requests

The interrupt request flag bit (ADC1: ADI) is not set if A/D conversion is reactivated (ADC1: AD = "1") at the same time as the previous A/D conversion completes.

● Turn-on sequence for A/D converter power supply and analog inputs

Always apply the A/D converter power supply (AV_{CC}, AV_{SS}) and analog inputs (AN0 to AN7) at the same time or after turning on the digital power supply (V_{CC}).

Similarly, when power supply is turned off, always turn off the A/D converter power supply (AV_{CC}, AV_{SS}) and analog inputs (AN0 to AN7) at the same time or before turning off the digital power supply (V_{CC}).

Take care that AV_{CC}, AV_{SS}, and the analog inputs do not exceed the digital power supply voltage when turning the A/D converter power supply on or off.

● Conversion time

A/D conversion function conversion time is affected by oscillator frequency, and main clock speed (speed-shift function).

● Continuous activation input clock

The timebase timer output, which can be selected as input clock, is not affected by the speed-shift function. Note also that the cycle time is affected (for one cycle) when the timebase timer is cleared.

13.7 Program Example for A/D Converter

This section gives program examples for the A/D conversion function of the10-bit A/D converter.

■ Program example for A/D conversion function

● Processing description

- Performs software-activated A/D conversion of the analog voltage input to the AN0 pin. The example does not use interrupts and detects conversion completion within the program loop.

● Coding example

```
DDR0    EQU    0001H           ;Port 0 data direction
ADC1    EQU    0020H           ;A/D control register 1
ADC2    EQU    0021H           ;A/D control register 2
ADDL    EQU    0023H           ;A/D data register L
ADDH    EQU    0022H           ;A/D data register H
ADER    EQU    0024H           ;A/D input enable register
AN0     EQU    DDR0:0          ;Define the AN0 analog input pin.
ADI     EQU    ADC1:3          ;Define the interrupt request flag bit.
ADMV    EQU    ADC1:2          ;Define the conversion-in-progress flag bit.
AD      EQU    ADC1:0          ;A/D converter activation bit (software activation)
EXT     EQU    ADC2:1          ;Define the continuous activation enable bit.
;-----Main program-------------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
         :
        CLRB   AN0             ;Set P00/AN0 pin as an analog input pin (AN0).
        SETB   ADER:0
        CLRI                   ;Disable interrupts.
        CLRB   EXT             ;Disable continuous activation.
AD_WAIT
        BBS    ADMV,AD_WAIT    ;Loop to check that the A/D converter is stopped.
        MOV    ADC1,#00000000B ;Select analog input channel 0 (AN0),
                                clear interrupt request flag, and do not activate
                                software.
        MOV    ADC2,#00000001B ;Disable interrupt request output, select the A/D
                                conversion function,and select software activation
                                by the AD bit.
        SETI                   ;Enable interrupts.
         :
        SETB   AD              ;Activate by software
AD_CONV
        BBS    ADMV,AD_CONV    ;Loop to delay until A/D conversion completes
                                (approx. 24  s at 12.5 MHz).
        CLRB   ADI             ;Clear interrupt request flag.
        MOV    A,ADDL          ;Read A/D conversion data.(Lower 8 bits)
        MOV    A,ADDH          ;Read A/D conversion data (upper 2 bits)
         :
         :
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 14
UART/SIO

This chapter describes the functions and operation of the UART/SIO.

14.1 Overview of UART/SIO

The UART/SIO is a general purpose data communication interface. The UART/SIO supports both synchronous and asynchronous mode operation and transmits variable-length serial data. The transmission format is the "NRZ" system and the transmission data rate is configurable by setting the proprietary baud rate generator, external clocks, internal timers.

■ UART/SIO Function

The UART/SIO communicates with other CPUs and peripheral devices by sending/receiving serial data (serial input/output).

- The full-duplex double buffer embedded in the device enables full-duplex bi-directional communication.
- Users can configure the UART/SIO to the synchronous transfer mode or asynchronous transfer mode.
- Internal baud rate generator allows users to select a baud rate from 14 different speeds. The baud rate is also configured by setting external clock inputs.
- The variable data length system allows users to set the data length at 7 to 8 bit non-parity or 8 to 9-bit parity mode (See Table 14.1-1 "UART operating mode").
- The data transmission format is based on the NRZ (Non Return to Zero) system.

Table 14.1-1 UART operating mode

Operating Mode	Data Length		Transfer Mode	Stop Bit Length
	non-parity	parity		
0	7	8	Asynchronous	1 bit or 2 bits
	8	9		
1	8		Synchronous	-

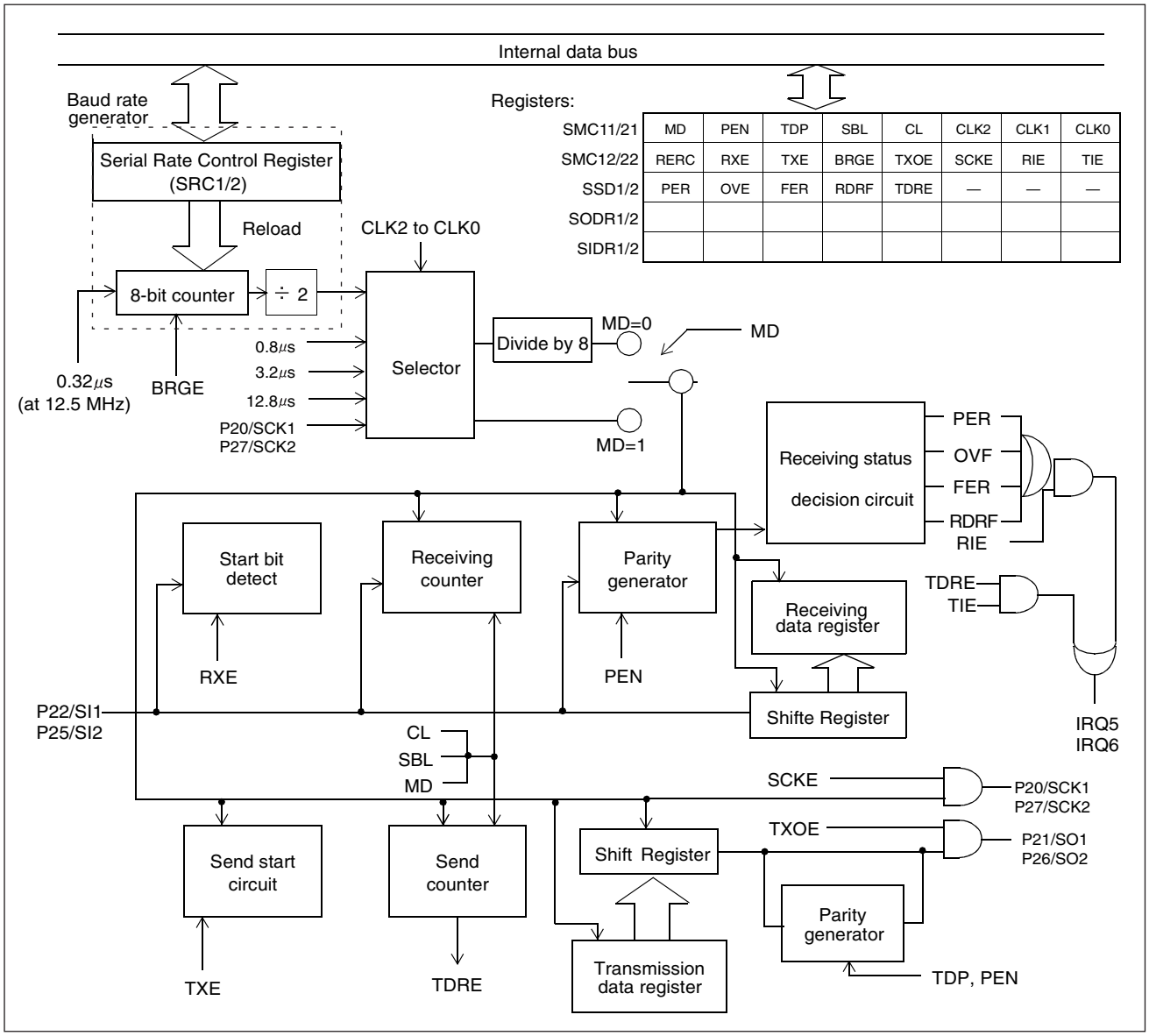
14.2 Structure of UART/SIO

The UART/SIO consists of the following blocks:

- **Serial mode control register 1 (SMC11/21)**
- **Serial mode control register 2 (SMC12/22)**
- **Serial status and data register (SSD1/2)**
- **Serial rate control register (SRC1/2)**
- **Serial input data register (SIDR1/2)**
- **Serial output data register (SODR1/2)**

■ Block Diagram of UART/SIO

Figure 14.2-1 Block diagram of UART/SIO



- Serial Mode Control Registers (SMC11/ 21 and SMC12/22)

  These registers controls operating modes in the UART/SIO. The register sets transfer mode (synchronous/ asychronous), parity/non-parity, stop bit length, data length, transimission data rate, enable/disable of UART/SIO operation, enable/disable of UART/SIO serial clock output, enable/disable of serial output, enble/disable of interrupt request to CPU.

- Serial Rate Control Register (SRC1/2)

  This register controls the data transmission rate (baud rate). The register selects transfer rate generated by the baud rate generator.

- Serial Status and Data Register (SSD1/2)

  This register shows UART/SIO sending/receiving, error status, as well as receives parity.

- Serial Input Data Register (SIDR1/2)

  This register holds received data. Serial data received is converted and stored in the register. When the data length is set to 7 bits, bit 7 does not have meaning.

- Serial Output Data Register (SODR1/2)

  This register sets sending data. The data written in this register is converted to the serial data and output. When the data length is set to be 7 bits, bit 7 does not have meaning.

14.3 UART/SIO Pins

This section describes the pins and pin block diagram of UART/SIO.

■ UART/SIO Pins

The pins for the UART/SIO function are shift clock input/output pin (P20/SCK1, P27/SCK2), serial data output pin (P21/SO1, P26/SO2) and serial data input pin (P22/SI1, P25/SI2).

P20/SCK1, P27/SCK2:

This pin function either as a general purpose input/output port (P20/P27) or a clock input output pin (hysteresis input) for the UART/SIO.(SCK1/SCK2)

When clock output is enabled (SMC12/SMC22: SCKE=1), this pin function as clock output pin (SCK1/SCK2) irrespective of settings on corresponding port direction register. In this case, do not select an external clock (SMC11/21: CLK2, CLK1, CLK0 not 100_B).

To use the port as a UART/SIO clock input pin, disable the clock output (SMC12/SMC22: SCKE = 0) and configure the port as an input port by setting a corresponding port direction register bit (DDR2: bit0/bit7 = 0). In this case, be sure to select an external clock (SMC11/21: CLK2, CLK1,CLK0 = 100_B).

P21/SO1, P26/SO2:

This pin function either as general purpose input/output port (P21/P26) or serial data output pin of the UART/SIO (SO1/SO2).

When serial data output is enabled (SMC12/SMC22: TXOE=1), this pin function as serial data output pin of the UART/SIO irrespective of settings on corresponding port direction register.

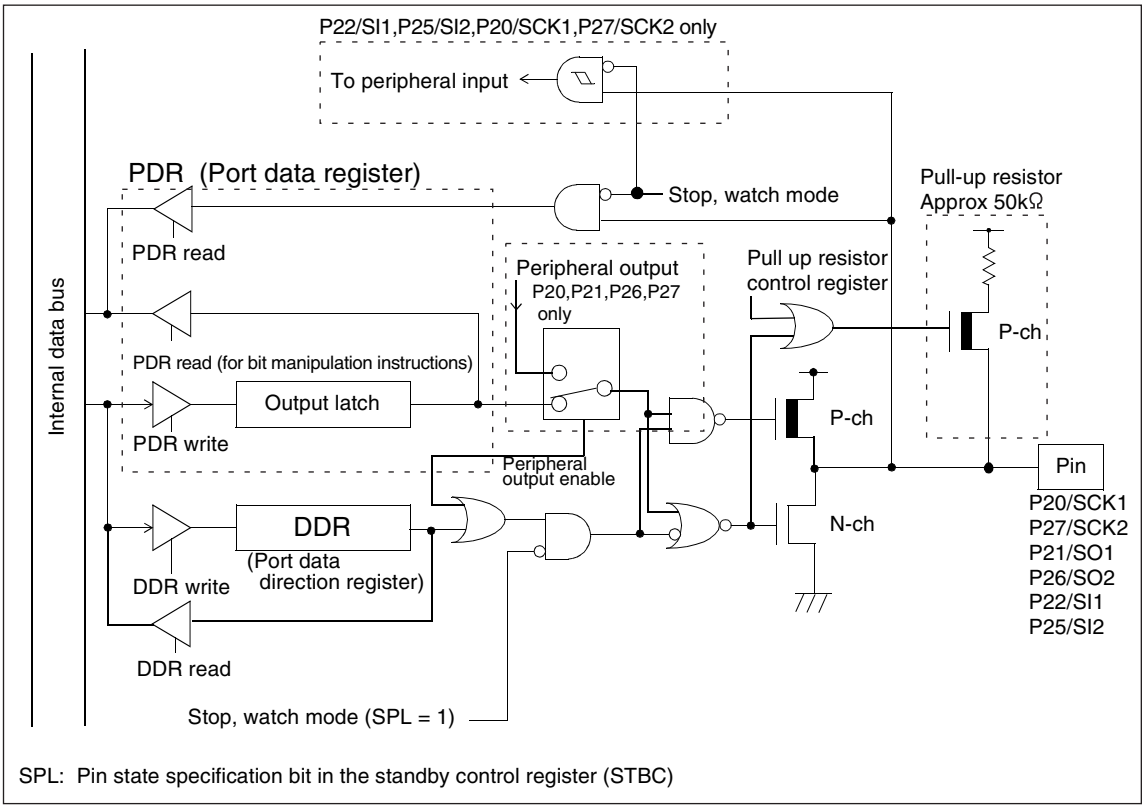
P22/SI1, P25/SI2:

This pin function either as general purpose input/output port (P22/P25) or serial data input pin (hysteresis input) of the UART/SIO (SI1/SI2).

To use the port as a UART/SIO serial data input pin, configure the port as input port by setting a corresponding bit of the port data direction register (DDR2: bit2/bit5 = 0).

■ Block Diagram of UART/SIO Pin

Figure 14.3-1 Block diagram of UART/SIO pin



SPL: Pin state specification bit in the standby control register (STBC)

Reference:

When pull up resistor is selected in pull-up resistor control register in stop and watch mode (SPL=1). The states of these pins are in "H" level (pull up state) rather than high impedance. However, during reset, pull up is unavailable and will be in high impedance state.

14.4 UART/SIO Registers

This section describes the registers of the UART/SIO.

■ UART registers

Figure 14.4-1 UART/SIO registers

SMC11/SMC21 (Serial mode control register 1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0026H 002BH	MD	PEN	TDP	SBL	CL	CLK2	CLK1	CLK0	00000000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SMC12/SMC22 (Serial mode control register 2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0027H 002CH	RERC	RXE	TXE	BRGE	TXOE	SCKE	RIE	TIE	00000000B
	W	R/W	R/W	W	R/W	R/W	R/W	R/W	

SRC1/2 (Serial rate control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
002AH 002FH									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SSD1/2 (Serial status and data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0028H 002DH	PER	OVE	FER	RDRF	TDRE	—	—	—	00001---B
	R	R	R	R	R				

SIDR1/2 (Serial input data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0029H 002EH									XXXXXXXXB
	R	R	R	R	R	R	R	R	

SODR1/2 (Serial output data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0029H 002EH									XXXXXXXXB
	W	W	W	W	W	W	W	W	

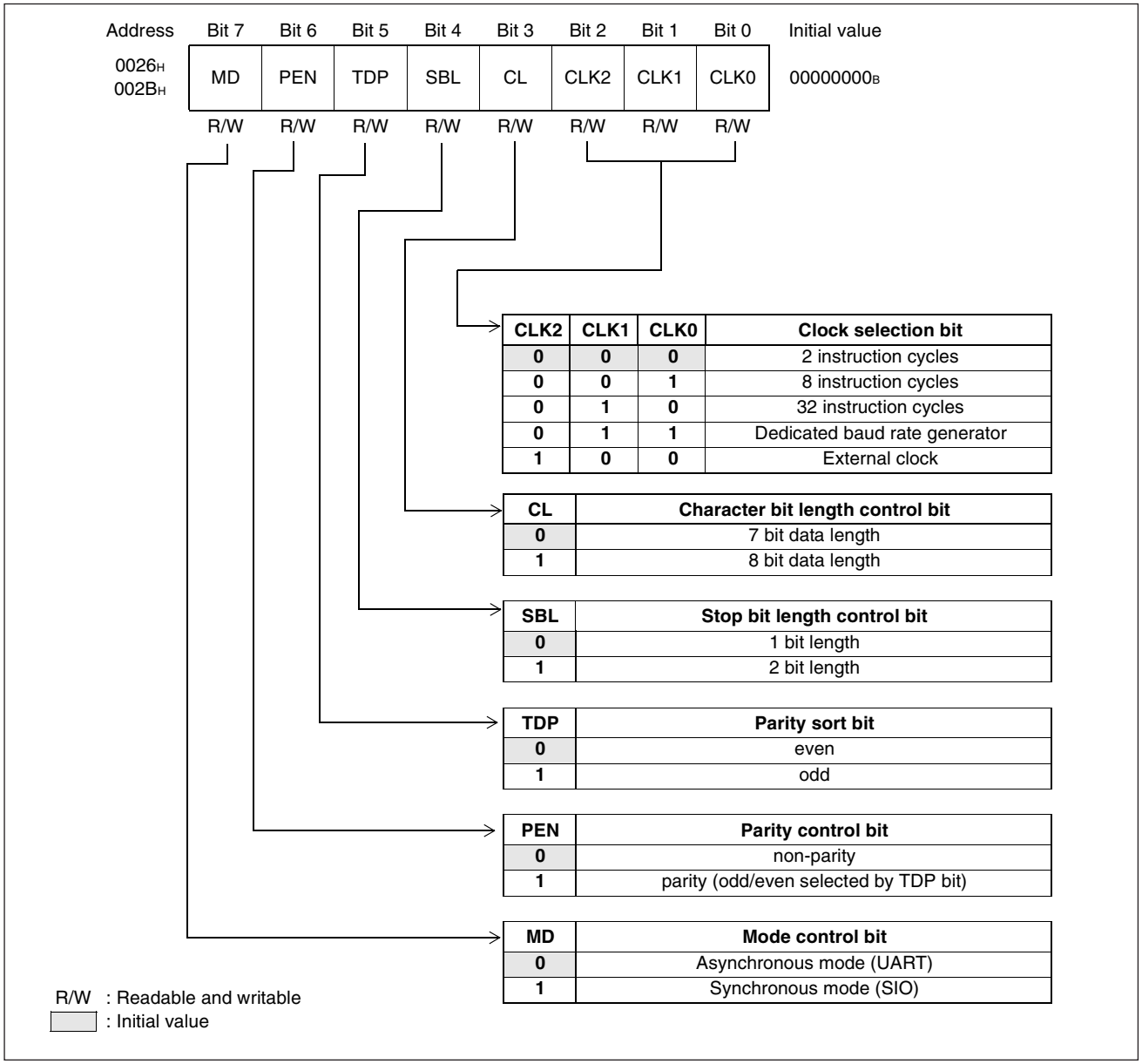
R/W : Readable and writable
R : Read-only
W : Write-only
— : Unused
X : Indeterminate

14.4.1 Serial Mode Control Register 1 (SMC11/21)

Serial Mode Control Register 1 (SMC11/21) sets transfer mode (synchronous/ asynchronous), stop bit length, data length, parity/non-parity and selects the clock.

■ Serial mode control register 1 (SMC11/21)

Figure 14.4-2 Serial mode control register 1 (SMC11/21)



CLK2	CLK1	CLK0	Clock selection bit
0	0	0	2 instruction cycles
0	0	1	8 instruction cycles
0	1	0	32 instruction cycles
0	1	1	Dedicated baud rate generator
1	0	0	External clock

CL	Character bit length control bit
0	7 bit data length
1	8 bit data length

SBL	Stop bit length control bit
0	1 bit length
1	2 bit length

TDP	Parity sort bit
0	even
1	odd

PEN	Parity control bit
0	non-parity
1	parity (odd/even selected by TDP bit)

MD	Mode control bit
0	Asynchronous mode (UART)
1	Synchronous mode (SIO)

Table 14.4-1 Serial mode control register 1 (SMC11/21) Bits

Bit		Function
Bit 7	MD: Mode control bit	• This bit selects the UART/SIO operating mode. In asynchronous mode, the UART/SIO operates on the serial clock divided by 8. In synchronous mode, it operates on the selected serial clock.
Bit 6	PEN: Parity control bit	• In asynchronous mode operation, this bit sets whether there is parity data or not
Bit 5	TDP: Parity sort bit	• In asynchronous mode operation, this bit decides which sort of parity data during serial transmission. During serial receiving, it checks parity data.
Bit 4	SBL: Stop bit length control bit	• This bit determines the stop bit length in asynchronous mode operation. - In serial transmissions, a stop bit of the bit length specified is appended. - In serial receiving, a stop bit is recognized as in a 1-bit length regardless of the value set here.
Bit 3	CL: Character bit length control bit	• This bit sets the character bit length in asynchronous mode operation. • This bit is set to 1 in synchronous mode operation.
Bit 2 Bit 1 Bit 0	CLK2, CLK1, CLK0: Clock selection bits	• These bits select the serial clock.

14.4.2 Serial Mode Control Register 2 (SMC12/22)

Serial Mode Control Register 2 (SMC12/22) sets the serial data output enable to pins, clock output enable to pins, transmission/receiving interrupt enable/disable and baud rate generator start/stop bit.

■ Serial mode control register 2 (SMC12/22)

Figure 14.4-3 Serial mode control register 2 (SMC12/22)

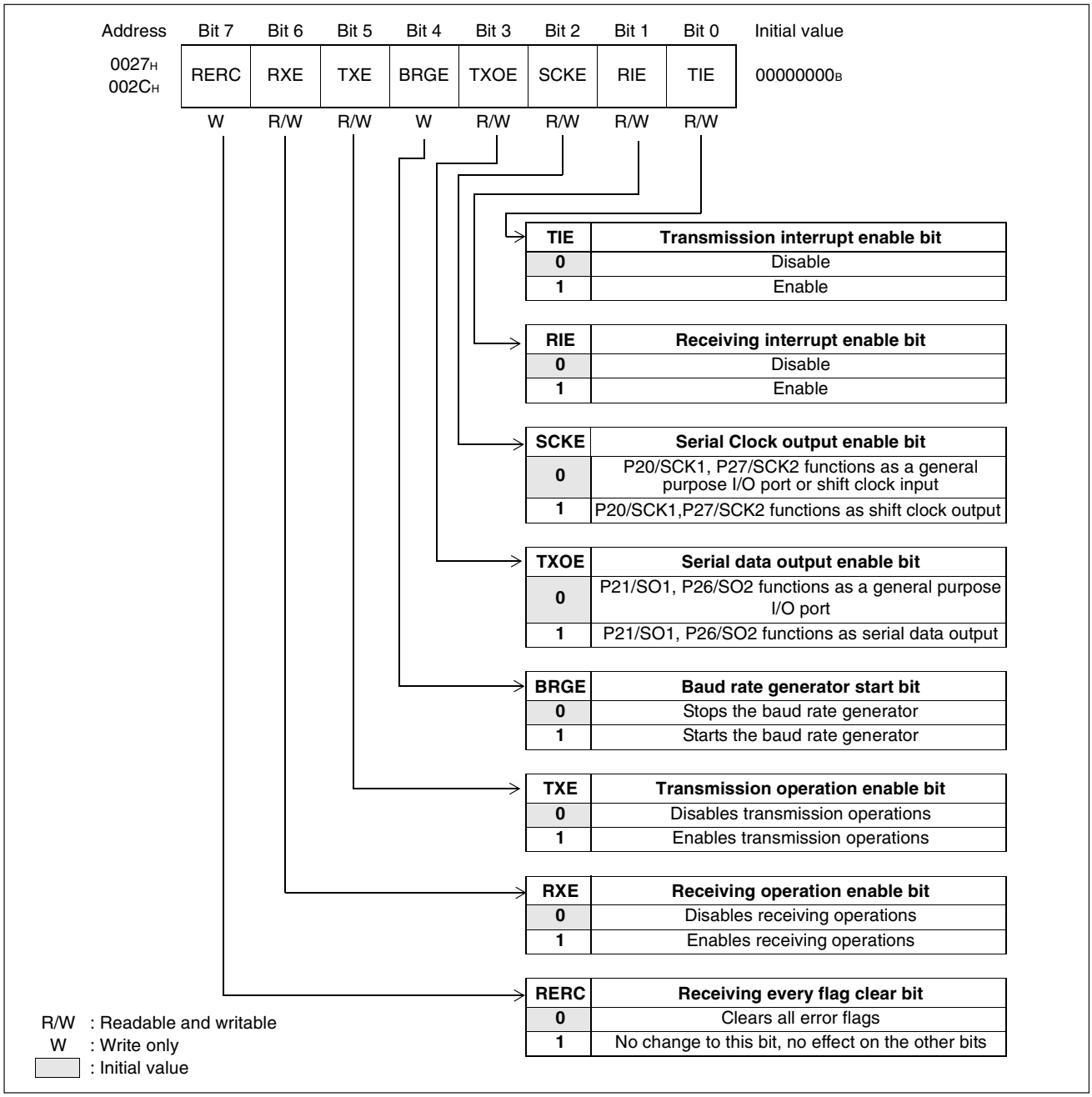


Table 14.4-2 Serial mode control register 2 (SMC12/22) bits

Bit		Function
Bit 7	RERC: Receiving every flag clear bit	• Writing "0" to this bit clears all error flag(PER, OVE and FER) in the SSD1/2 register.This bit is always "1" when read.
Bit 6	RXE: Receiving operation enable bit	• This bit enables reception of serial data. The receiving operation is stopped by writing "0" to this bit after receiving the current serial data, and disabled thereafter.
Bit 5	TXE : Transmission operation enable bit	• This bit enables transmission of serial data. The transmission operation is stopped by writing "0" to this bit after transmitting the current serial data, then disabled thereafter.
Bit 4	BRGE: Baud rate generator start bit	• This bit starts the baud rate generator
Bit 3	TXOE: Serial data output enable bit	• The P21/SO1, P26/SO2 pin functions as a general-purpose port (P21, P26) when this bit is set to "0" and as the serial data output pn (SO1, SO2) when set to "1". **Reference:** The pin functions as the (SO1, SO2) pin when serial data output is enabled (TXOE = "1"), regardless of the state of the general-purpose port P21, P26.
Bit 2	SCKE: Serial clock output enable bit	• This bit contorls shift clock input and output. • The P20/SCK1, P27/SCK2 pin function as the shift clcok input pin when this bit is set to "0" and as the shift clock output pin when set to "1". **Note:** • Set the P20/SCK1, P27/SCK2 pin as an input port when using this pin as the shift clock input. Also, selects external shift clock operation in the shift clock selection bits (SMC11/SMC21: CLK2, CLK1, CLK0 = "100_B") • When using this pin as internal shift clock output (SCKE = "1"), select internal shift clock operation (SMC11/SMC21: CLK2, CLK1, CLK0 = other than "100_B") **Reference:** • The pin functions as the SCK1, SCK2 output pin when shift clock is enabled (SCKE = "1") regardless of the state of the general-purpose port (P20, P27). • Set to shift clock input operation (SCKE = "0") when using this pin as a general-purpose port (P20, P27).
Bit 1	RIE: Receiving interrupt enable bit	• This bit enables receiving interrupts. If the RDRF bit is "1" or if any error flag is "1", a receiving interrupt is immediately generated once receiving interrupts are enabled.
Bit 0	TIE: Transmission interrupt enable bit	• This bit enables transmission interrupts. If the TDRE bit is "1", a transmission interrupt is immediately generated once transmission interrupts are enabled.

14.4.3 Serial Status and Data Register (SSD1/2)

The serial status and data register (SSD1/2) is used to set and monitor transmit/receive operations, error status.

■ Serial status and rate register (SSD1/2)

Figure 14.4-4 Serial status and data register (SSD1/2)

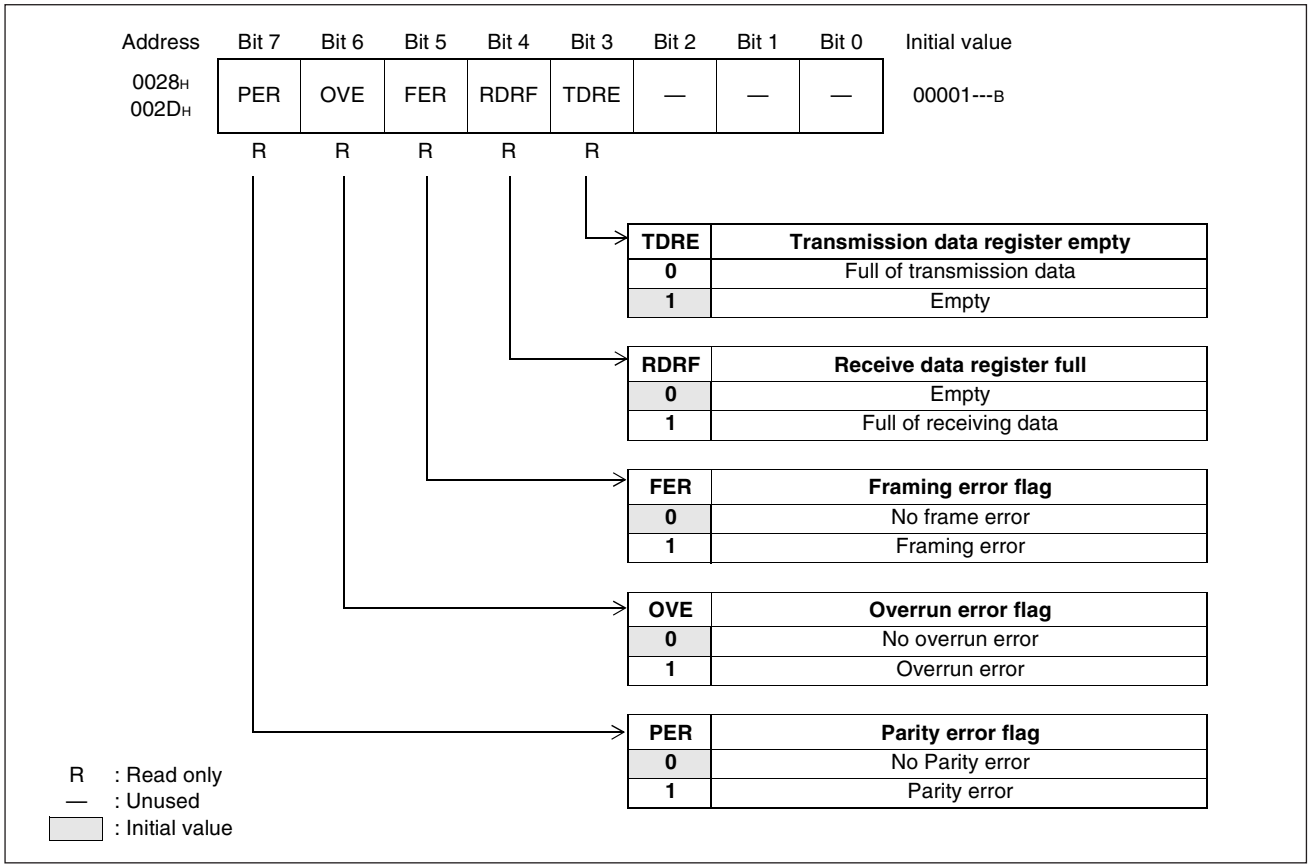


Table 14.4-3 Serial status and data register (SSD1/2) bits

Bit		Function
Bit7	PER: Parity error flag	• This bit is set when a parity error is generated during receiving. This bit can be cleared by writing "0" to the RERC bit of the SMC12/SMC22 register. When this flag is set, SIDR1/2 data becomes invalid. If the PER bit is set when the RIE bit of the SMC12/SMC22 register is "1", a receiving interrupt request is generated
Bit6	OVE: Overrun error flag	• This bit is set when an overrun error is generated during receiving. This bit can be cleared by writing "0" to the RERC bit of the SMC12/SMC22 register. When this flag is set, SIDR1/2 data becomes invalid. If the OVE bit is set when the RIE bit of the SMC12/SMC22 register is "1", a receiving interrupt request is generated.
Bit5	FER: Framing error flag	• This bit is set when a framing error is generated during receiving. This bit can be cleared by writing "0" to the RERC bit of the SMC12/SMC22 register. When this flag is set, SIDR1/2 data becomes invalid. If the FER bit is set when the RIE bit of the SMC12/SMC22 register is "1", a receiving interrupt request is generated.
Bit4	RDRF: Receive data register full	• This flag represents the status of the receiving data register SIDR1/2. This flag is set when receiving data is loaded into the SIDR1/2 register. It is cleared when the SIDR1/2 register is read. If the RDRF bit is set when the RIE bit of the SMC12/SMC22 register is "1", a receiving interrupt request is generated
Bit3	TDRE: Transmission data register empty	• This flag represents the status of the transmission data register SODR1/2. This flag is cleared when transmission data is written into the SODR1/2 register. It is set when the data is loaded into the transmission shift register and transmission begins. If the TDRE bit is set when the TIE bit of SMC12/SIM22 register is "1", a transmission interrupt request is generated
Bit 2 Bit 1 Bit 0	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.

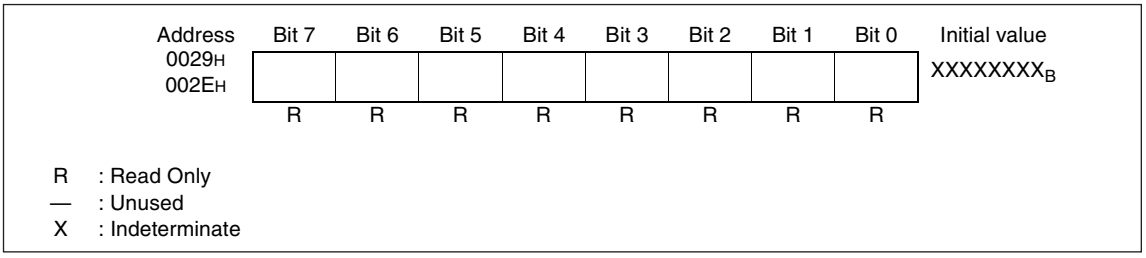
14.4.4 Serial Input Data Register (SIDR1/2)

The serial input data register (SIDR1/2) is used to input (receive) serial data.

■ Serial input data register (SIDR1/2)

Figure 14.4-5 "Serial input data register (SIDR1/2)" shows the bit allocations of the serial input data register.

Figure 14.4-5 Serial input data register (SIDR1/2)



This register stores received data. Serial data sent to the serial data input pin (SI) is converted in the shift register and stored in this register.

If received data is normally set in this register, the receive data flag bit (RDRF) is set to "1", and a receive interrupt request occurs if it is enabled. When the interrupt request is detected, check the RDRF bit in an interrupt processing or in a program. If a receive data is stored on this register, read this register, and then the RDRF flag is cleared automatically.

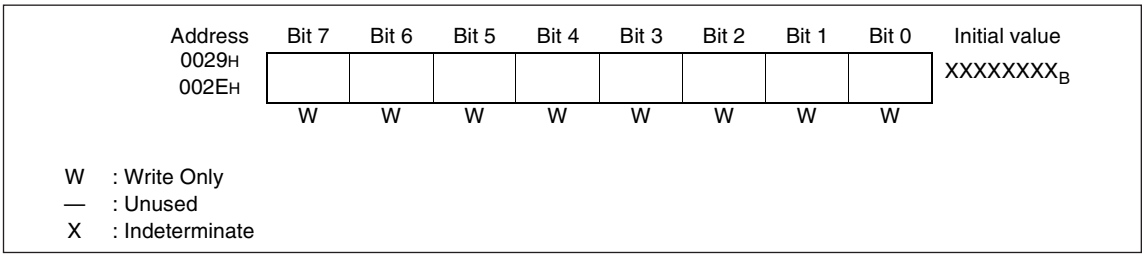
14.4.5 Serial Output Data Register (SODR1/2)

The serial output data register (SODR1/2) is used to output (transmit) serial data.

■ Serial output data register (SODR1/2)

Figure 14.4-6 "Serial output data register (SODR1/2)" shows the bit allocations of the serial output data register.

Figure 14.4-6 Serial output data register (SODR1/2)



When transmission is enabled, writing transmit data to this register will transfer the transmit data to the transmit register. The transmit data is converted on the transmit shift register and sent to the serial data output pin (SO1/2).

Setting transmit data in the SODR1/2 register sets the transmit data flag to "0". After the transmit data is transferred to the transmit shift register, the transmit data flag is set to "1" and the SODR1/2 is ready for the next data. If transmit interrupt request is enabled, interrupt occurs. Write next transmission data when interrupt occurs or transmit data flag bit is set to "1".

When the data rate length is set to 7 bits, bit 7 does not have meaning.

14.4.6 Serial rate control register (SRC1/2)

The serial rate control register (SRC1/2) is to set the UART/SIO transmission speed (baud rate).

■ Serial rate control register (SRC1/2)

Figure 14.4-7 "Serial rate control register (SRC1/2)" shows the bit allocations of the serial rate control register.

Figure 14.4-7 Serial rate control register (SRC1/2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
002AH 002FH									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X : Indeterminate

When the clock selection bits, CLK2 to CLK0 of SMC11/SMC21, are set to "011", the dedicated baud rate generator is selected as the serial clock.

Also, be sure to write to this register only when the UART/SIO operations are stopped.

14.5 UART/SIO Interrupts

The UART/SIO has five flags related to interrupts - three error flag bits (PER, OVE, FER), a receive data flag bit (RDRF) and a transmit data flag bit (TDRE). Except the TDRE flag which is related to data transmission, all other four flags are related to data reception.

■ Transmit interrupt

When transmission is enabled, writing transmit data to this SODR1/2 register transfers the transmit data to the transmit register. The transmit data is converted on the transmit shift register and sent from the serial data transmit pin (SO1/2).

When the UART/SIO is ready to accept next data, the TDRE is set to "1", and an interrupt request (IRQ5/6) is output to the CPU if transmit interrupt request is enabled (SMC12/22: TIE=1).

■ Receive interrupt

When the data is input normally to the stop bit(s), the RDRF is set to "1".

When an over run error or a framing error or parity error occurs, each corresponding error flag bit is set to "1".

These bits are set when the stop bit(s) are detected, and an interrupt request (IRQ5/6) to the CPU is generated if receive interrupt is enabled (SMC12/22: RIE=1).

■ Registers and vector tables for UART/SIO interrupts

Table 14.5-1 Registers and vector tables for UART/SIO interrupts

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ5	ILR2 ($007C_H$)	L51 (Bit3)	L50 (Bit2)	$FFF0_H$	$FFF1_H$
IRQ6	ILR2 ($007C_H$)	L61 (Bit5)	L60 (Bit4)	$FFEE_H$	$FFEF_H$

See Section 3.4.2 "Interrupt Processing"" for details on the interrupt operation.

14.6 Operation of UART/SIO

This section describes the operation of the UART/SIO.
The UART/SIO has a serial communication function (operation mode 0 and 1).

■ Operation of UART/SIO

● Operation mode

The UART/SIO has 2 operation modes. The mode 0 and 1 are standard serial transmission modes in which a data type from 7 bit data length/parity to 8 bit data length/non-parity is selected (See Table 14.1-1 "UART operating mode").

14.7 Operation of mode 0

The operating mode 0 provides an asynchronous mode operation.

■ Operation of operating mode 0

The serial clock is selected by the bits CLK2 to CLK0 of the SMC11/SMC21 register. Selection can be made from five sources: three internal clock outputs, one external clock output, and one baud rate generator output. Always be sure to input a clock when the external clock has been selected.

In clock asynchronous mode, the shift clock selected by the bits CLK2 to CLK0 is divided by eight for use as the baud rate generator. Data transfer in the range -2% to +2% of the selected baud rate is possible.

Figure 14.7-1 Transmit operation in mode 0 (CLK2, CLK1, CLK0 not equal to 011_B)

$$\text{Baud rate} = \frac{1}{8 \times \boxed{\text{Clock selction from CLK2~CLK0}}} \quad \text{(bps)}$$

Figure 14.7-2 Transmit operation in mode 0 (CLK2, CLK1, CLK0 = 011_B)

$$\text{Baud rate} = \frac{1}{8 \times 2 \times n \times \boxed{\begin{array}{l} 4/F_{CH} \\ 8/F_{CH} \\ 16/F_{CH} \\ 64/F_{CH} \end{array}}} \quad \text{(bps)}$$

speed-shift selection

F_{CH} : Frequency being used
n : Value of serial rate control register SRC1/2

Table 14.7-1 Baud rate setting

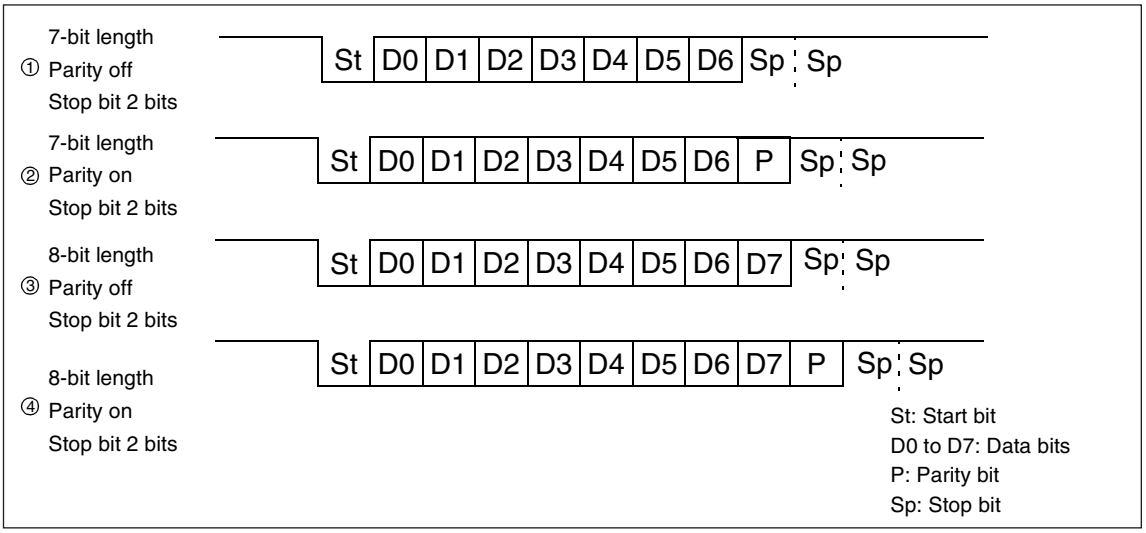
Frequency (F_{CH})	12.5MHz	8MHz	7.3728MHz	4.9152MHz
Instruction cycle	$4/F_{CH}$ (0.32 µs)	$4/F_{CH}$ (0.5 µs)	$4/F_{CH}$ (0.54 µs)	$4/F_{CH}$ (0.81 µs)
Baud rate	97656 (n=2)	62500 (n=2)	-	76800 (n=1)
	48828 (n=4)	31250 (n=4)	38400 (n=3)	38400 (n=2)
	13020 (n=15)	10417 (n=12)	19200 (n=6)	19200 (n=4)
	12207 (n=16)	9615 (n=13)	9600 (n=12)	9600 (n=8)
	6104 (n=32)	4807 (n=26)	4800 (n=24)	4800 (n=16)
	3004 (n=65)	2403 (n=52)	2400 (n=48)	2400 (n=32)
	1502 (n=130)	1201 (n=104)	1200 (n=96)	1200 (n=64)
	-	600 (n=208)	600 (n=192)	600 (n=128)
	-	-	-	300 (n=0)

■ Transmission data format

The UART/SIO uses only data of the NRZ (Non-Return to Zero) format. The diagram below shows this data format. In the diagram a stop bit length of two bits is used.

As shown in the diagram below, transfer data must begin from a start bit (Low level), and the data bit length specified by the LSB first is transferred and then ends on a stop bit (High level). Transfer data is assumed as high level when idle.

Figure 14.7-3 Transfer data format

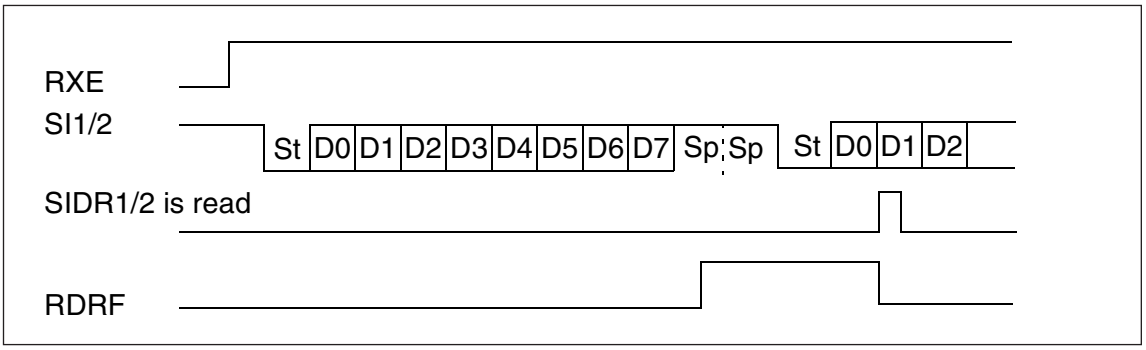


■ Receiving operation in asynchronous mode

Bits CLK2 to CLK0 of the SMC11/21 register select the baud rate clock. For information on the baud rate clock, see Figure 14.7-1 "Transmit operation in mode 0 (CLK2, CLK1, CLK0 not equal to 011_B)" and Table 14.1-1 "Baud rate setting". Receiving is enabled when the RXE bit of the SMC11/21 register is set to "1". Receiving operation starts at the first falling edge (detection of a start bit) of input data. Once receiving operation has ended, the RDRF bit of the SSD1/2 register is set to "1", and receiving data is loaded into the SIDR1/2 register. Also, if the RDRF bit is set to "1" when the RIE bit is "1", a receiving interrupt is sent to the CPU. If any one of three errors (PER, OVE, or FER) is present when receiving ends, the RDRF bit will not be set to "1", and receiving data will not be loaded into SIDR1/2. Therefore, the value in the SIDR1/2 register will be the former data received. Also, as long as the RXE bit is not set to "0", receiving operation will continue when a start bit is detected even if an error flag is present.

Writing a "0" to the RXE bit of the SMC12/22 register during receiving operation will disable any further receiving operation once current data receiving has ended.

Figure 14.7-4 Receiving operation in asynchronous mode.

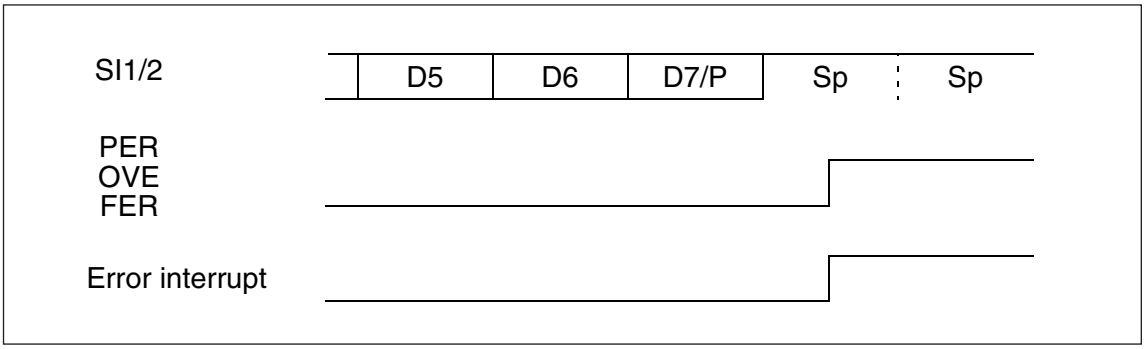


■ Receiving errors in asynchronous mode

Three types of error detection are performed in asynchronous mode. The three errors are: parity error, overrun error, and framing error. When an error is detected, a "1" is set in the corresponding PER, OVE, or FER bit of the SSD1/2 register.

These errors are detected as described below when receiving ends. When any of these errors is detected, the RDRF bit will not be set, and receiving data will not be loaded into the SIDR1/2 register. Therefore the value in the SIDR1/2 register will be last data received. Also, these error flags can all be cleared by writing "0" into the RERC bit of the SMC12/22 register.

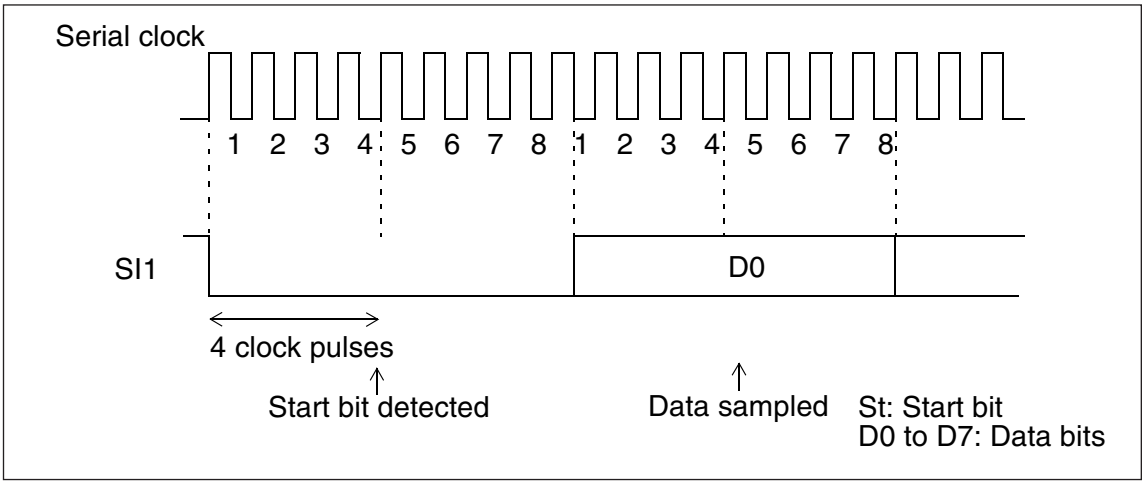
Figure 14.7-5 Set timing for receive errors



■ Detecting the start bit during receiving operation

The start bit is recognized when a Low level is present for four clock pulses of the selected serial clock (clock generator output, and so on.) after the first falling edge of input data. After the start bit is detected, data is sampled at the rising edge of the fifth clock pulse of the serial clock

Figure 14.7-6 Start bit detection

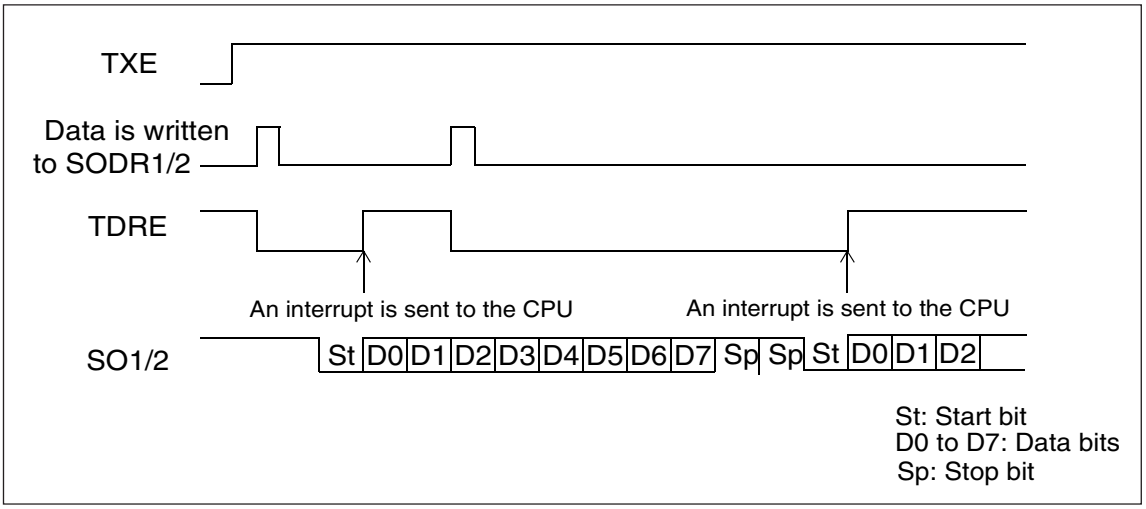


■ Transmission operation in clock asynchronous mode

When the TXE bit of the SMC12/22 register is set to "1", the TDRE bit of the SSDR1/2 register is cleared, and transmission operation begins once transmission data is written into the SODR1/2 register. The TDRE bit of the SSDR2 register is set once transmission data output begins after the data in the SODR1/2 register is loaded into the shift register. Writing data to the SODR1/2 register during transmission operation (when the TDRE bit is "1") causes the TDRE bit to be cleared, and transmission operation to continue after current transmission operation for the specified bit length has been terminated.

Also, writing "0" to the TXE bit of the SMC12/22 register during transmission operation disables transmission operation once current transmission for the specified bit length ends on condition that the SODR1/2 register is empty (the TDRE bit is "1"). If there is data in the SODR1/2 register (TDRE bit is "0"), transmission operation will be disabled once data in the SODR1/2 register has been transmitted.

Figure 14.7-7 Transmission operation in asynchronous mode



14.8 Operation of mode 1

The operating mode 1 provides a synchronous mode operation.

■ Operation of operating mode 1

In synchronous mode, the CLK2 to CLK0 bits of the SMC11/21 register select the clock from one of five source: three internal clocks, an external clock, or the baud rate generator. Shift operation are then performed using the selected clock as a shift clock. Be sure to set the SCKE bit of SMC12/SMC22 to "0" when an external clock is input. Also, be sure to set the SCKE bit to "1" if outputting an internal clock or baud rate generator as the shift clock.

The following equation is show the calculating the baud rate in synchronous mode.

Figure 14.8-1 Transmit operation in mode 1 (CLK2, CLK1, CLK0 = 011_B)

$$\text{Baud rate} = \frac{1}{2 \times n \times \boxed{\begin{matrix} 4/F_{CH} \\ 8/F_{CH} \\ 16/F_{CH} \\ 64/F_{CH} \end{matrix}}} \quad \text{(bps)}$$

speed-shift selection

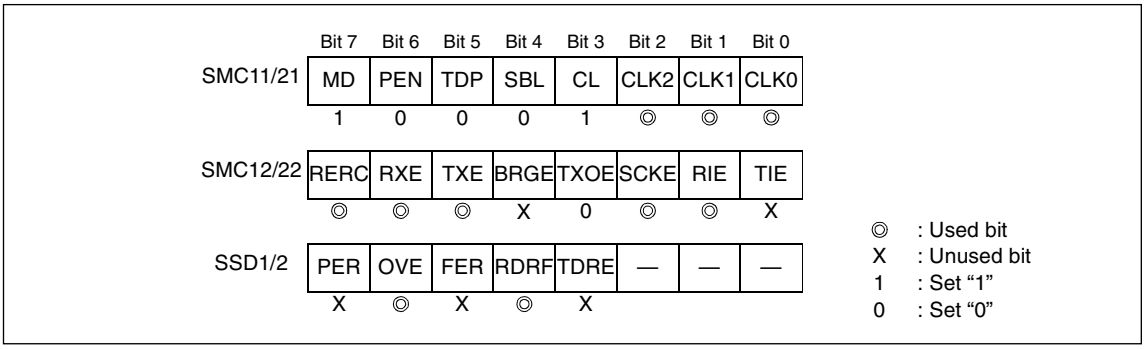
F_{CH} : Frequency being used
n : Value of serial rate control register SRC1/2

Figure 14.8-2 Transmit operation in mode 1 (CLK2, CLK1, CLK0 not 011_B)

$$\text{Baud rate} = \frac{1}{\boxed{\text{Clock selction from CLK2~CLK0}}} \quad \text{(bps)}$$

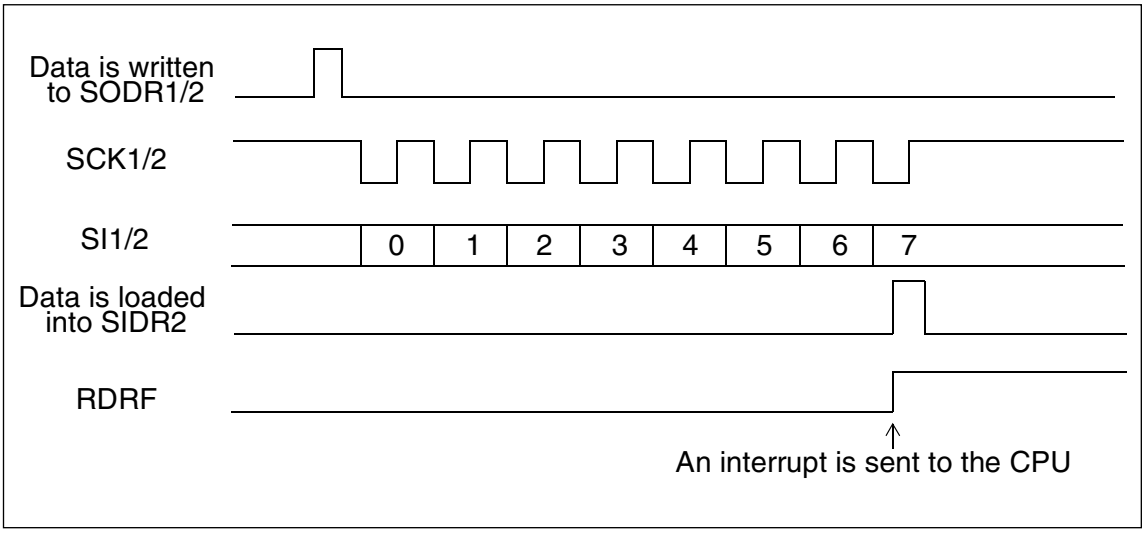
■ 8-bit receiving operation

Figure 14.8-3 Setting of 8-bit receiving operation in synchronous mode



Receiving operation can be enabled by setting the TXE, RXE bits to "11". Operation starts when data is written to the SODR1/2 register. Receiving operation is performed in synchronization with the rising edge of the shift clock. Once receiving the 8-bit data has ended, shift register data is loaded into the SIDR1/2 register, the RDRF flag is set to "1", and, if the RIE bit is "1", an interrupt request is sent to the CPU. If an overrun error has occurred when receiving has ended, data will not be loaded into the SIDR1/2 register. Writing "0" to the RXE bit during receiving operations will stop the reception operations once the current 8-bit data reception has ended. Be sure that serial clock input is always High level when serial operation is stopped (regardless of the value of the RXE bit).

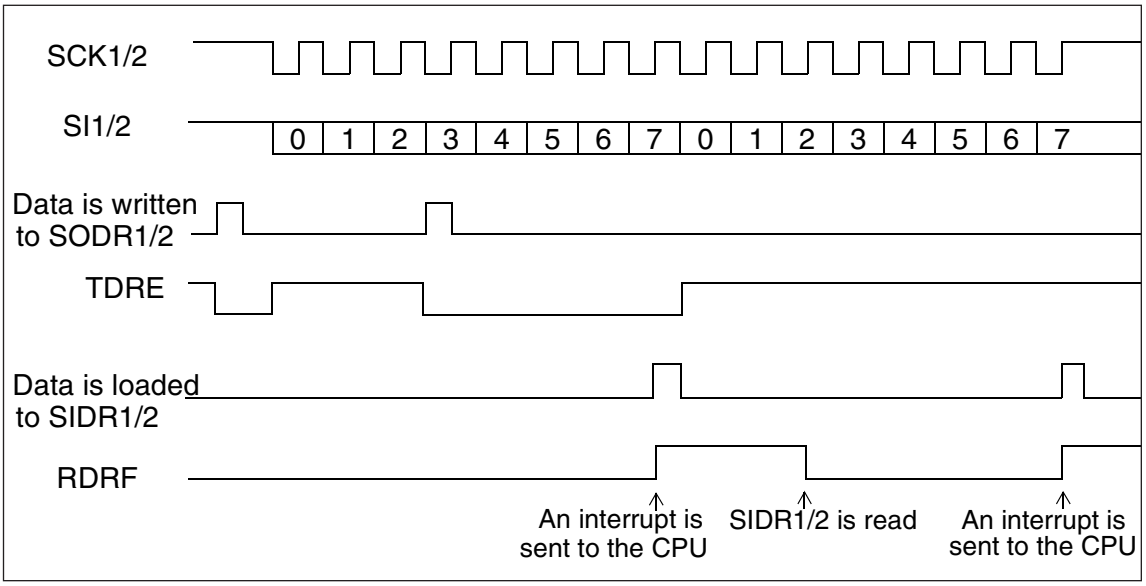
Figure 14.8-4 8 bit receiving operation in synchronous mode



■ Continuous receiving operations

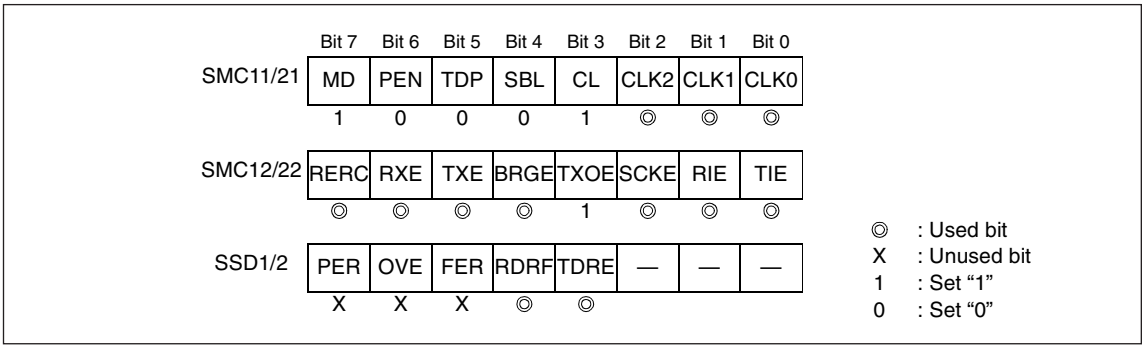
In synchronous mode, it is possible to perform continuous receiving operations in addition to the 8-bit data receiving. In addition to the bits used during the 8-bit receiving, the TIE bit of the SMC12/22 register and the TDRE bit of the SSD1/2 register are also used for this type of operation. Receiving operation is enabled by setting the TXE,RXE bits to "11". This operation starts when data is written to the SODR1/2 register. Receiving operation is performed in synchronization with the rising edge of the shift clock. Once shift operation begins, the TDRE bit is set to "1", and an interrupt is sent to the CPU if the TIE bit is "1". The next shift operation can be enabled by writing data to the SODR1/2 register before the current 8-bit shift operation ends, and this allows receiving operation to continue even after receiving the 8-bit data. After receiving 8-bit data, the data in the shift register is loaded into the SIDR1/2 register and the RDRF flag is set to "1", and then if the RIE bit is "1", an interrupt request is sent to the CPU. If an overrun error occurs after receiving the data, the data will not be loaded into the SIDR1/2 register, therefore the contents of the SIDR1/2 register will be last data received. Also, reading the SIDR1/2 register clears the receiving interrupt (RDRF). Subsequent receive operations can be stopped by writing "0" to the RXE bit. Writing "0" to the RXE bit during receiving operation stops receiving operation after receiving the current 8-bit data.

Figure 14.8-5 Continuous receiving operations in synchronous mode



■ 8-bit transmission operation

Figure 14.8-6 Setting of 8-bit transmission operation

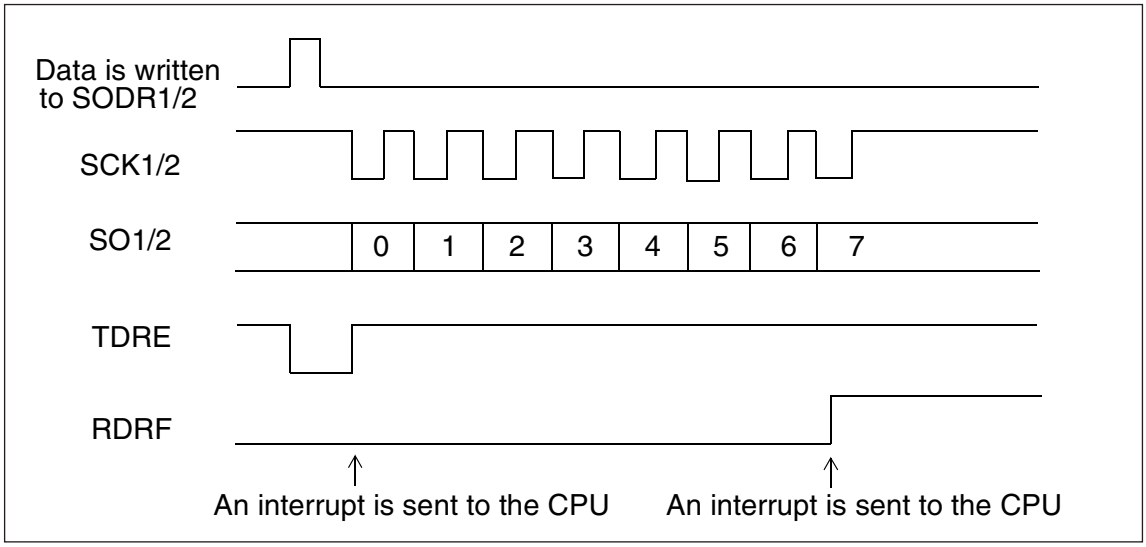


The transmission operation can be activated by writing the SODR1/2 register after setting the TXE,RXE bits to "11". When transmission operation starts, the data written in the SODR1/2 register is loaded into the shift register and shift operation begins. Once SODR1/2 register data is loaded into the shift register, the TDRE flag is set to "1" and, if the TIE bit is "1", an interrupt request is sent to the CPU. Serial data output can be enabled by setting TXOE to "1", then Output is made in synchronization with the falling edge of the shift clock.

Writing "0" to the TXE bit during transmission operation stops transmission operation once current 8-bit data is transmitted.

Also, after transmitting the 8-bit data, the RDRF bit is automatically set to "1", and an interrupt is sent to the CPU if the RIE bit is "1". Data transmission begins from bit 0 and ends at bit 7. Be sure to set the serial clock input to High level when serial operation is stopped (regardless of the value of the TXE bit).

Figure 14.8-7 8-bit transmission operation

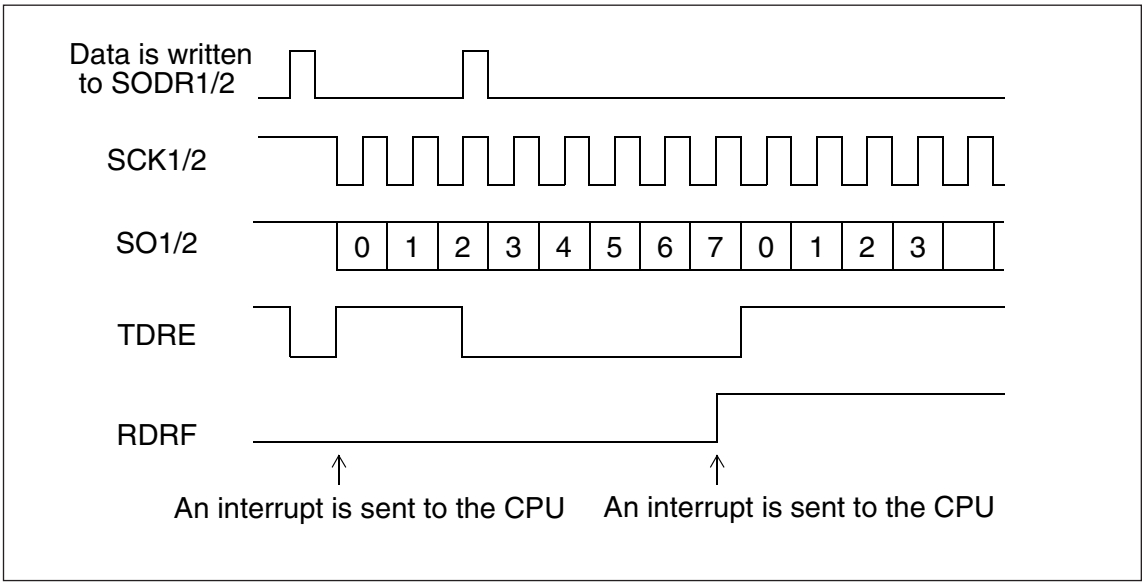


■ Continuous transmission operations

In synchronous mode, it is possible to perform continuous transmission operations in addition to transmitting the 8-bit data. The transmission operation is enabled by setting the TXE,RXE bits to "11", and then, start when data is written to the SODR1/2 register. Once transmission operation starts, data written in the SODR1/2 register is loaded into the shift register and shift operation begins. When data in the SODR1/2 register is loaded into the shift register, the TDRE flag is set to "1", and, if the TIE bit is "1", an interrupt is sent to the CPU.

The continuous operations are performed by writing the next data to be transmitted into the SODR1/2 register (SODR1/2 register is empty) during current transmission operation while the TDRE bit is "1". The TDRE bit is cleared by writing to SODR1/2, and then the data written in the SODR1/2 register is loaded into the shift register and transmission operation continues after transmitting the current 8-bit data. Subsequent transmission operation can be stopped by writing "0" to the TXE bit. Writing "0" to the TXE bit during transmission operation when the SODR1/2 register is empty (TDRE bit is "1") will stop transmission operation after transmitting the current 8-bit. Also, when there is data in the SODR1/2 register (TDRE bit is "0"), transmission operation will stop after data in the SODR1/2 register has been transmitted. Finally, when the 8-bit data transmission ends, the RDRF bit is set to "1", and, if the RIE bit is "1", an interrupt is sent to the CPU.

Figure 14.8-8 Continuous transmission operation in synchronous mode



CHAPTER 15
BUZZER OUTPUT

This chapter describes the functions and operation of the buzzer output.

15.1 Overview of Buzzer Output

The buzzer output can select from seven different output frequencies (square waves) and can be used for applications such as sounding a buzzer to confirm key input. The function uses the same output pin as the remote control transmit output.

■ Buzzer output function

The buzzer output function outputs a signal (square wave) suitable for applications such as sounding a buzzer to confirm an operation.

- For buzzer output, one of seven output frequencies can be selected, or the output disabled.
- Four divide-by-n outputs are supplied from the timebase timer and three from the watch prescaler, for selection as the buzzer output signal.

Reference:

Since divided outputs of the timebase timer and watch prescaler are fed as the buzzer output signal, the buzzer output will be affected when the signal source selected for it (timebase timer or watch prescaler) is cleared.

Note:

Since the timebase timer stops when the main clock oscillator stops (during subclock mode), do not select the divided output of the timebase timer as the buzzer output when subclock mode is used.

Similarly, do not select the watch prescaler as the buzzer source in a chip in which the single clock option is selected.

Table 15.1-1 "Output frequency" lists the seven output frequencies (square waves) that can be selected for the buzzer output function.

Table 15.1-1 Output frequency

Clock supply source	Buzzer output cycle	Square wave output (Hz)
Timebase Timer	$2^{12}/F_{CH}$	$F_{CH}/2^{12}$ (3.052 kHz)
	$2^{11}/F_{CH}$	$F_{CH}/2^{11}$ (6.104 kHz)
	$2^{10}/F_{CH}$	$F_{CH}/2^{10}$ (12.207 kHz)
	$2^{9}/F_{CH}$	$F_{CH}/2^{9}$ (24.414 kHz)
Watch Prescaler	$2^{5}/F_{CL}$	$F_{CL}/2^{5}$ (1.024 kHz)
	$2^{4}/F_{CL}$	$F_{CL}/2^{4}$ (2.048 kHz)
	$2^{3}/F_{CL}$	$F_{CL}/2^{3}$ (4.096 kHz)

F_{CH}: Main clock oscillation frequency

F_{CL}: Subclock oscillation frequency

The frequencies enclosed in parentheses ( ) are for F_{CH} = 12.5 MHz, and F_{CL} = 32.768 kHz.

Reference:

Calculation example for output frequency

For a 12.5 MHz main clock source oscillation and if the buzzer register (BUZR) selects a timebase timer divided output of $F_{CH}/2^{10}$ (BZ2, BZ1, BZ0 = 011_B), the output frequency of the BUZ pin is calculated as follows:

Output frequency = $F_{CH} / 2^{10}$
= 12.5 MHz / 1024
≒ 12.207 kHz

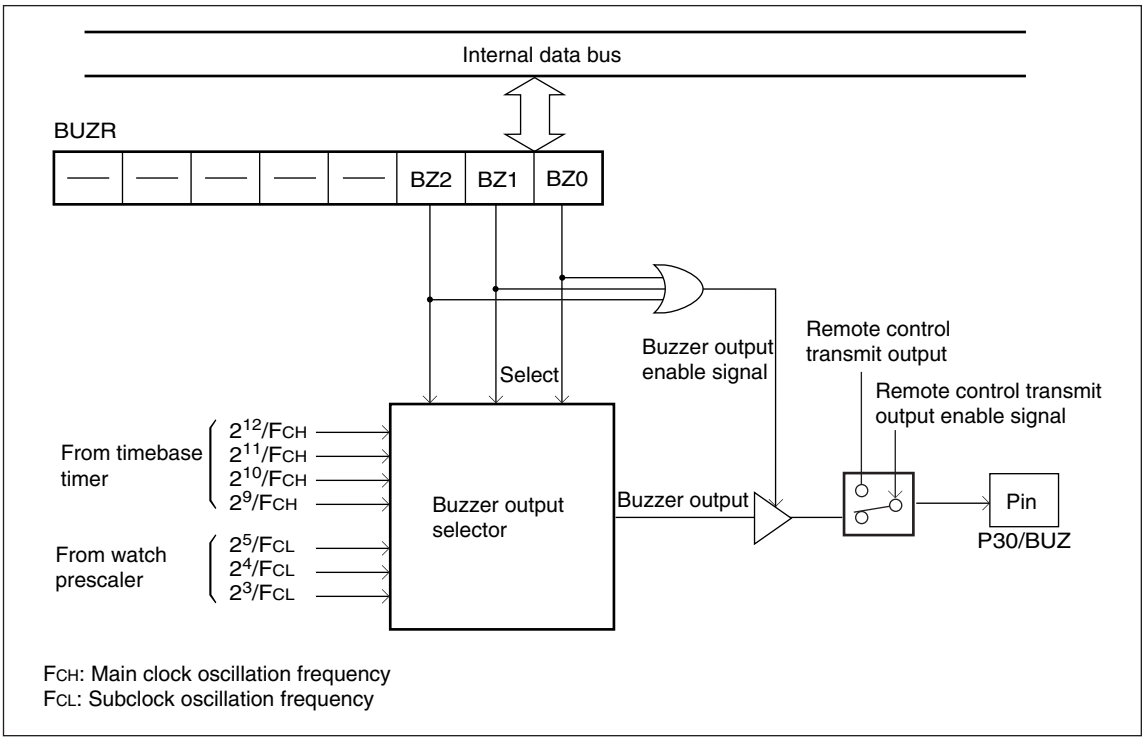
15.2 Block Diagram of Buzzer Output

The buzzer output consists of the following two blocks:

- **Buzzer output selector**
- **Buzzer output register (BUZR)**

■ Block diagram of buzzer output

Figure 15.2-1 Block diagram of buzzer output



● Buzzer output selector

Selects one of the four frequencies output from the timebase timer or three frequencies output from the watch prescaler.

● BUZR register

The BUZR register to set the buzzer output frequency and enable buzzer output.

Buzzer output is enabled if an output frequency is specified (BUZR: BZ2, BZ1, BZ0 = other than "000_B") in the BUZR register.

15.3 Structure of Buzzer Output

This section describes the pin, pin block diagram, and register of the buzzer output.

■ Buzzer output pin

The buzzer output uses the P30/BUZ pin. This pin can function either as an output-only port (P30) or the buzzer output pin (BUZ).

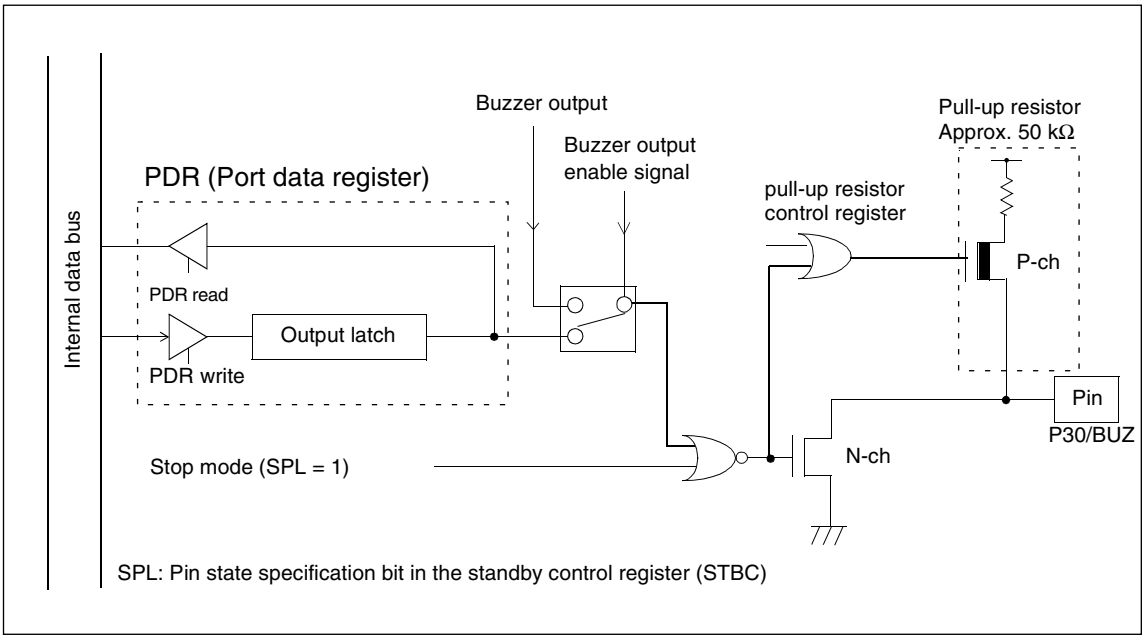
BUZ:

This pin outputs a buzzer square wave with the specified frequency.

Setting a buzzer output frequency in the buzzer output register (BUZR: BZ2, BZ1, BZ0 = other than "000_B") automatically sets the P30/BUZ pin as the BUZ pin.

■ Block diagram of buzzer output pin

Figure 15.3-1 Block diagram of P30/BUZ pin



■ Buzzer output register

Figure 15.3-2 Buzzer output register

BUZR(Buzzer Register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
000F$_H$	—	—	—	—	—	BZ2	BZ1	BZ0	-----000$_B$
						R/W	R/W	R/W	

R/W : Readable and writable
— : Unused
X : Indeterminate

15.3.1 Buzzer Register (BUZR)

The buzzer register (BUZR) is used to select the buzzer output frequency and also enables buzzer output.

■ Buzzer register (BUZR)

Figure 15.3-3 Buzzer register (BUZR)

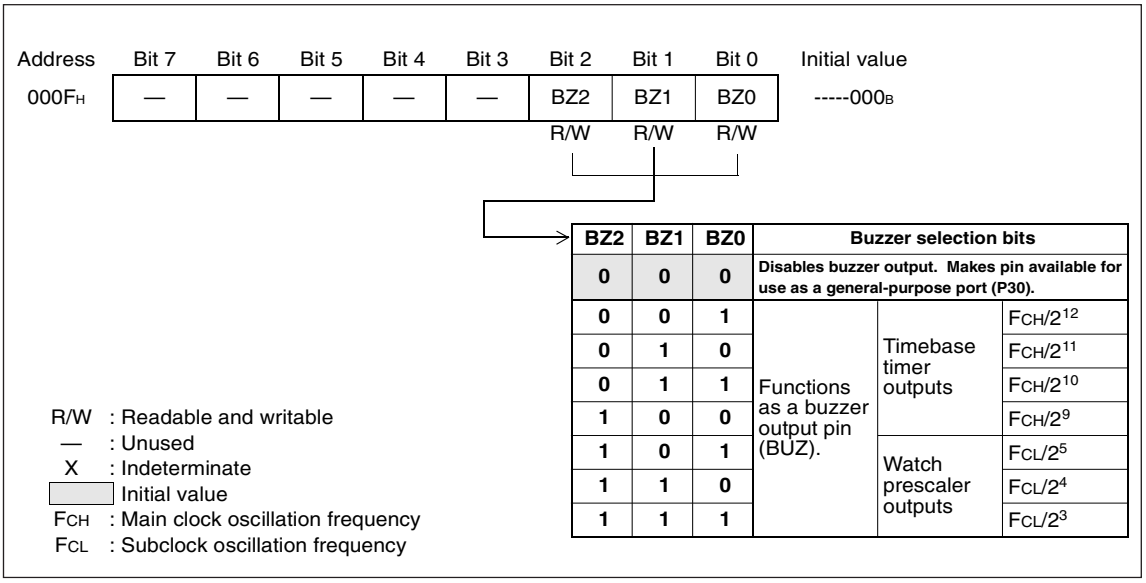


Table 15.3-1 Buzzer register (BUZR) bits

Bit		Function
Bit 7 Bit 6 Bit 5 Bit 4 Bit 3	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 2 Bit 1 Bit 0	BZ2, BZ1, BZ0: Buzzer selection bits	• These bits selects buzzer output and enable output. • Setting "000_B" disables the buzzer output and sets the pin as a general-purpose output port (P30). Setting other than "000_B" sets the pin as the buzzer output (BUZ) pin and outputs a square wave of the selected frequency. • Selects one of four divided outputs from the timebase timer or three from the timeclock prescaler. **References:** • Do not select a timebase timer division output in subclock mode. • The subclock oscillator operates in the main-stop mode. Therefore, if the pin state specification bit (STBC: SPL) is "0", the buzzer output can be used even in main-stop mode by selecting one of the watch prescaler divided-by-n outputs (BZ2, BZ1, BZ0 = 101_B to 111_B).

15.4 Program Example for Buzzer Output

This section gives a program example for the buzzer output.

■ Program example for buzzer output

● Processing description

- Output a buzzer output of approximately 3.051 kHz to the BUZ pin, then turn the buzzer output "OFF".
- For a 12.5 MHz main clock source oscillation and selecting $F_{CH}/2^{12}$ (F_{CH}: main clock oscillation), the buzzer output frequency is as follows:

Buzzer output frequency = 12.5 MHz / 2^{12}
= 12.5 MHz / 4096
= 3.052 kHz

● Coding example

```
BUZR    EQU    000FH           ;Buzzer register
;-----Main program-------------------------------------------------------------
        CSEG                   ;[CODE SEGMENT]
         :
BUZON   MOV    BUZR,#00000001B ;Buzzer output "ON".

         :
         :
         :
BUZOFF  MOV    BUZR,#00000000B ;Buzzer output "OFF".
         :
        ENDS
;-------------------------------------------------------------------------------
        END
```

APPENDIX

This appendix includes I/O maps, instruction lists, and other information.

APPENDIX A I/O Map

Table A-1 "I/O map" lists the addresses of the registers of used by the internal peripheral functions of the MB89470 series.

■ I/O map

Table A-1 I/O Map (1/3)

Address	Register name	Register Description	Read/Write	Initial value
00_H	PDR0	Port 0 data register	R/W	$XXXXXXXX_B$
01_H	DDR0	Port 0 data direction register	W*	00000000_B
02_H	PDR1	Port 1 data register	R/W	$XXXXXXXX_B$
03_H	DDR1	Port 1 data direction register	W*	00000000_B
04_H	PDR2	Port 2 data register	R/W	00000000_B
05_H	(Reserved)			
06_H	DDR2	Port 2 data direction register	R/W	00000000_B
07_H	SYCC	System clock control register	R/W	$X\text{-}1MM100_B$
08_H	STBC	Standby control register	R/W	$00010XXX_B$
09_H	WDTC	Watchdog timer control register	W*	$0\text{---}XXXX_B$
$0A_H$	TBTC	Timebase timer control register	R/W	$00\text{---}000_B$
$0B_H$	WPCR	Watch prescaler control register	R/W	$00\text{--}0000_B$
$0C_H$	PDR3	Port 3 data register	R/W	-1111111_B
$0D_H$	PDR4	Port 4 data register	R	$\text{-----}XXX_B$
$0E_H$	RSFR	Reset flag register	R	$XXXX\text{----}_B$
$0F_H$	BUZR	Buzzer register	R/W	$\text{-----}000_B$
10_H	PDR5	Port 5 data register	R/W	$\text{---}XXXXX_B$
11_H	DDR5	Port 5 direction register	R/W	$\text{---}00000_B$
12_H to 13_H	(Reserved)			
14_H	T4CR	Timer 22 control register	R/W	$000000X0_B$
15_H	T3CR	Timer 21 control register	R/W	$000000X0_B$

Table A-1 I/O Map (2/3)

Address	Register name	Register Description	Read/Write	Initial value
16_H	T4DR	Timer 22 data register	R/W	$XXXXXXXX_B$
17_H	T3DR	Timer 21 data register	R/W	$XXXXXXXX_B$
18_H	T2CR	Timer 12 control register	R/W	$000000X0_B$
19_H	T1CR	Timer 11 control register	R/W	$000000X0_B$
$1A_H$	T2DR	Timer 12 data register	R/W	$XXXXXXXX_B$
$1B_H$	T1DR	Timer 11 data register	R/W	$XXXXXXXX_B$
$1C_H$ to$1E_H$	(Reserved)			
$1F_H$	(Reserved)			
20_H	ADC1	A/D control register 1	R/W	$-00000X0_B$
21_H	ADC2	A/D control register 2	R/W	-0000001_B
22_H	ADDH	A/D data register (Upper byte)	R	$------XX_B$
23_H	ADDL	A/D data register (Lower byte)	R	$XXXXXXXX_B$
24_H	ADER	A/D input enable register	R/W	11111111_B
25_H	(Reserved)			
26_H	SMC11	UART/SIO serial mode control register 11	R/W	00000000_B
27_H	SMC12	UART/SIO serial mode control register 12	R/W	00000000_B
28_H	SSD1	UART/SIO serial status/data register 1	R/W	$00001---_B$
29_H	SIDR1/ SODR1	UART/SIO serial data register 1	R/W*	$XXXXXXXX_B$
$2A_H$	SRC1	UART/SIO serial rate control register 1	R/W	$XXXXXXXX_B$
$2B_H$	SMC21	UART/SIO serial mode control register 21	R/W	00000000_B
$2C_H$	SMC22	UART/SIO serial mode control register 22	R/W	00000000_B
$2D_H$	SSD2	UART/SIO serial status/data register 2	R/W	$00001---_B$
$2E_H$	SIDR2/ SODR2	UART/SIO serial data register 2	R/W*	$XXXXXXXX_B$
$2F_H$	SRC2	UART/SIO serial rate control register 2	R/W	$XXXXXXXX_B$
30_H	EIC1	External interrupt 1 control register 1	R/W	00000000_B
31_H	EIC2	External interrupt 1 control register 2	R/W	00000000_B
32_H	EIE2	External interrupt 2 enable register	R/W	$---00000_B$

Table A-1 I/O Map (3/3)

Address	Register name	Register Description	Read/Write	Initial value
33_H	EIF2	External interrupt 2 flag register	R/W	-------0_B
34_H	PCR1	PWC control register 1	R/W	0-0--000_B
35_H	PCR2	PWC control register 2	R/W	00000000_B
36_H	PLBR	PWC reload buffer register	R/W	$XXXXXXXX_B$
37_H	(Reserved)			
38_H	CNTR	PWM timer control register	R/W	0-000000_B
39_H	COMR	PWM timer compare register	W*	$XXXXXXXX_B$
$3A_H$ to $6F_H$	(Reserved)			
70_H	PURC0	Port 0 pull-up resistor control register	R/W	11111111_B
71_H	PURC1	Port 1 pull-up resistor control register	R/W	11111111_B
72_H	PURC2	Port 2 pull-up resistor control register	R/W	11111111_B
73_H	PURC3	Port 3 pull-up resistor control register	R/W	-1111111_B
74_H	(Reserved)			
75_H	PURC5	Port 5 pull-up resistor control register	R/W	---11111_B
76_H	(Reserved)			
77_H	(Reserved)			
78_H	(Reserved)			
79_H	(Reserved)			
$7A_H$	(Reserved)			
$7B_H$	ILR1	Interrupt level setting register 1	W*	11111111_B
$7C_H$	ILR2	Interrupt level setting register 2	W*	11111111_B
$7D_H$	ILR3	Interrupt level setting register 3	W*	11111111_B
$7E_H$	ILR4	Interrupt level setting register 4	W*	11111111_B
$7F_H$	(Reserved)			

*: Bit manipulation instruction cannot be used.

- Read/write access symbols

  R/W: Readable and writable

  R: Read-only

  W: Write-only

- Initial value symbols

  0: The initial value of this bit is "0"

  1: The initial value of this bit is "1"

  X: The initial value of this bit is undefined

  M: The initial value of this bit is determined by mask option.

  -: Unused

  Note:

  Do not use vacancies.

APPENDIX B Overview of Instructions

Appendix B describes the instructions used by the F^{2}MC-8L.

B.1 Overview of F^{2}MC-8L Instructions

The F^{2}MC-8L supports 140 types of instructions.

■ Symbols used with Instructions

Table B.1-1 "Symbols in the instruction list" lists the symbols used in the instruction code descriptions in Appendix B.

Table B.1-1 Symbols in the instruction list

Symbol	Meaning
dir	Direct address (8 bits)
off	Offset (8 bits)
ext	Extended address (16 bits)
#vct	Vector table number (3 bits)
#d8	Immediate data (8 bits)
#d16	Immediate data (16 bits)
dir:b	Bit direct address (8 bits:3 bits)
rel	Branch relative address (8 bits)
@	Register indirect addressing (examples: @A, @IX, @EP)
A	Accumulator (8 or 16 bits, which are determined depending on the instruction being used)
AH	Higher 8 bits of the accumulator (8 bits)
AL	Lower 8 bits of the accumulator (8 bits)
T	Temporary accumulator (8 or 16 bits, which are determined depending on the instruction being used)
TH	Higher 8 bits of the temporary accumulator (8 bits)
TL	Lower 8 bits of the temporary accumulator (8 bits)
IX	Index register (16 bits)
EP	Extra pointer (16 bits)
PC	Program counter (16 bits)
SP	Stack pointer (16 bits)
PS	Program status (16 bits)
dr	Either accumulator or index register (16 bits)
CCR	Condition code register (8 bits)
RP	Register bank pointer (5 bits)

Table B.1-1 Symbols in the instruction list (Continued)

Symbol	Meaning
Ri	General-purpose register (8 bits, i = 0 to 7)
X	X is immediate data (8 or 16 bits, which are determined depending on the instruction being used).
(X)	The content of X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).
((X))	The address indicated by the X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).

■ Explanation of Items in Instruction Set Table

Table B.1-2 Explanation of Items in Instruction Set Table

Item	Explanation
Mnemonic	Assembler notation of an instruction
~	Number of instructions. A single instruction cycle consists of two machine cycles.
#	Number of bytes
Operation	Operation of an instruction
TL, TH, AH	A content change when each of the TL, TH and AH instruction is executed. Symbols in the column indicate the following: • "-" indicates no change. • dH is the 8 upper bits of operation description data. • AL and AH must become the contents of AL and AH immediately before the instruction is executed. • 00 becomes 00.
N, Z, V, C	An instruction of which the corresponding flag will change. If + is written in this column, the relevant instruction will change its corresponding flag.
OP code	Code of an instruction. If an instruction is more than one code, it is written according to the following rule: Example: 48 to 4F <-- This indicates 48, 49, ... 4F.

B.2 Addressing

The F^{2}MC-8L has the following ten addressing modes:

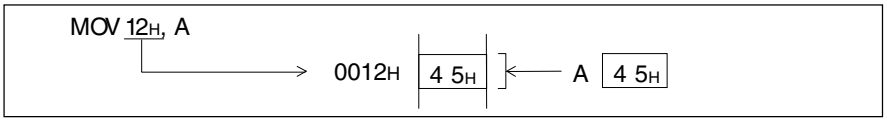
- **Direct addressing**
- **Extended addressing**
- **Bit direct addressing**
- **Index addressing**
- **Pointer addressing**
- **General-purpose register addressing**
- **Immediate addressing**
- **Vector addressing**
- **Relative addressing**
- **Inherent addressing**

■ Explanation of addressing

● Direct addressing

Direct addressing is indicated by dir in the instruction list. This addressing is used to access the area between 0000_H and $00FF_H$. In this addressing mode, the higher byte of the address is 00_H and the lower byte is specified by the operand. Figure B.2-1 "Example of direct addressing" shows an example.

Figure B.2-1 Example of direct addressing

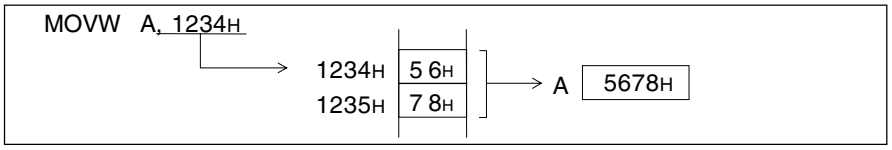


● Extended addressing

Extended addressing is indicated by ext in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the first operand specifies the higher byte of the address, and the second operand specifies the lower byte.

Figure B.2-2 "Example of Extended Addressing" shows an example.

Figure B.2-2 Example of extended addressing

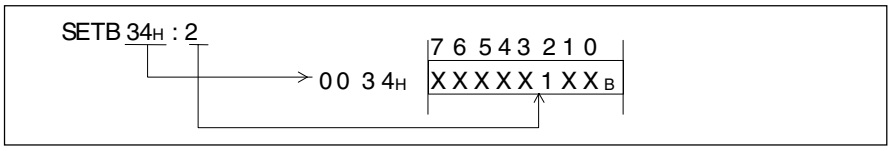


● Bit direct addressing

Bit direct addressing is indicated by dir:b in the instruction list. This addressing is used to access a particular bit in the area between 0000_H and $00FF_H$. In this addressing mode, the higher byte of the address is 00_H and the lower byte is specified by the operand. The bit position at the address is specified by the lower three bits of the operation code.

Figure B.2-3 "Example of bit direct addressing" shows an example.

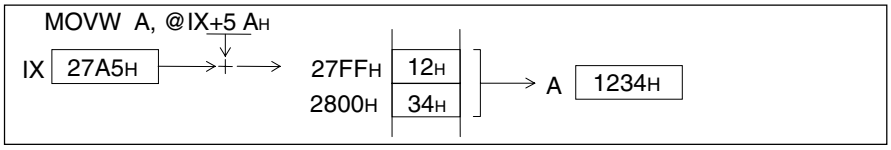
Figure B.2-3 Example of bit direct addressing



● Index addressing

Index addressing is indicated by @IX+off in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is the value resulting from sign-extending the contents of the first operand and adding them to IX (index register). Figure B.2-4 "Example of index addressing" shows an example.

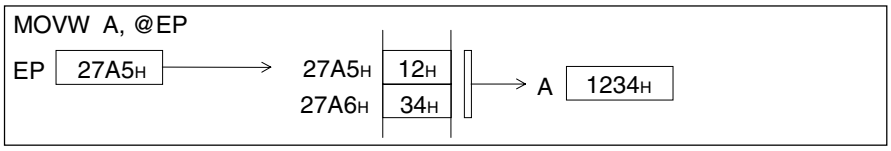
Figure B.2-4 Example of index addressing



● Pointer addressing

Pointer addressing is indicated by @EP in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is contained in EP (extra pointer). Figure B.2-5 "Example of pointer addressing" shows an example.

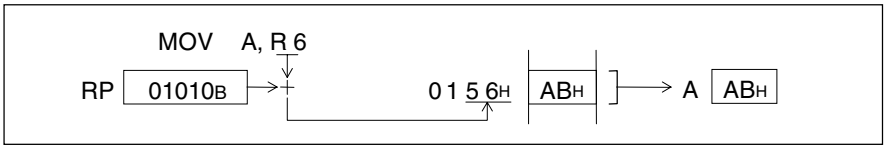
Figure B.2-5 Example of pointer addressing



● General-purpose register addressing

General-purpose register addressing is indicated by Ri in the instruction list. This addressing is used to access a register bank in the general-purpose register area. In this addressing mode, the higher byte of the address is always 01 and the lower byte is specified based on the contents of RP (register bank pointer) and the lower three bits of the operation code. Figure B.2-6 "Example of general-purpose register addressing" shows an example.

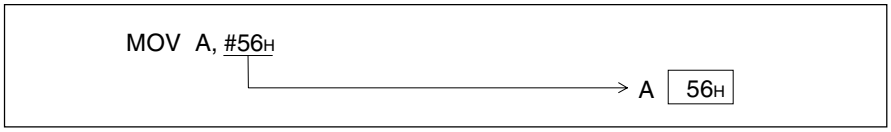
Figure B.2-6 Example of general-purpose register addressing



● Immediate addressing

Immediate addressing is indicated by #d8 in the instruction list. This addressing is used when immediate data is required. In this addressing mode, the operand is used as immediate data. Whether the data is specified in bytes or words is determined by the operation code. Figure B.2-7 "Example of immediate addressing" shows an example.

Figure B.2-7 Example of immediate addressing



● Vector addressing

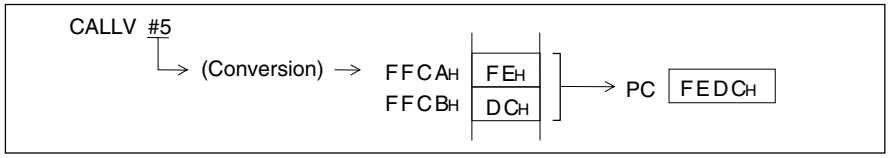
Vector addressing is indicated by vct in the instruction list. This addressing is used to branch to a subroutine address stored in the vector table. In this addressing mode, vct information is contained in the operation codes, and the corresponding table addresses are created as shown in Table B.2-1 "Vector table addresses corresponding to vct".

Table B.2-1 Vector table addresses corresponding to vct

#vct	Vector table address (higher address:lower address of branch destination)
0	$FFC0_H$: $FFC1_H$
1	$FFC2_H$: $FFC3_H$
2	$FFC4_H$: $FFC5_H$
3	$FFC6_H$: $FFC7_H$
4	$FFC8_H$: $FFC9_H$
5	$FFCA_H$: $FFCB_H$
6	$FFCC_H$: $FFCD_H$
7	$FFCE_H$: $FFCF_H$

Figure B.2-8 "Example of vector addressing" shows an example.

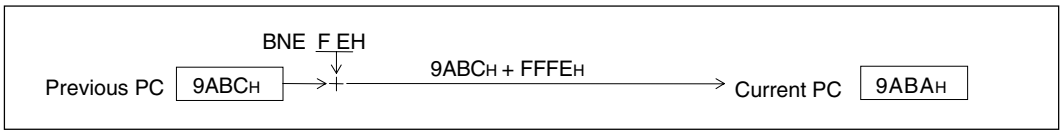
Figure B.2-8 Example of vector addressing



● Relative addressing

Relative addressing is indicated by rel in the instruction list. This addressing is used to branch to within the area between the address 128 bytes higher and that 128 bytes lower relative to the address contained in the PC (program counter). In this addressing mode, the result of a signed addition of the contents of the operand to the PC is stored in the PC. Figure B.2-9 "Example of relative addressing" shows an example.

Figure B.2-9 Example of relative addressing

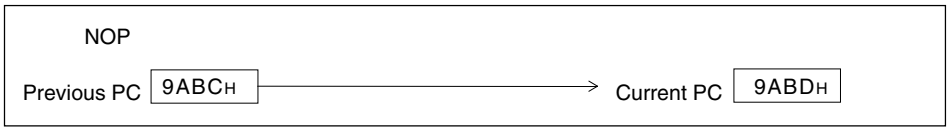


In this example, a branch to the address of the BNE operation code occurs, thus resulting in an infinite loop.

● Inherent addressing

Inherent addressing is indicated as the addressing without operands in the instruction list. This addressing is used to perform the operation determined by the operation code. In this addressing mode, different operations are performed via different instructions. Figure B.2-10 "Example of inherent addressing" shows an example.

Figure B.2-10 Example of inherent addressing



B.3 Special Instructions

This section describes the special instructions used for other than addressing.

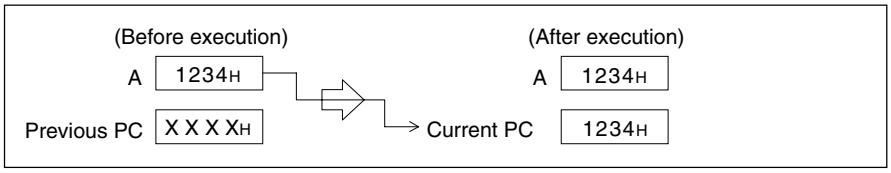
■ Special instructions

● JMP @A

This instruction sets the contents of A (accumulator) to PC (program counter) as the address, and causes a branch to that address. One of the N branch destination addresses is selected from a table, and then transferred to A. The instruction can be executed to perform N-branch processing.

Figure B.3-1 "JMP @A" shows a summary of the instruction.

Figure B.3-1 JMP @A

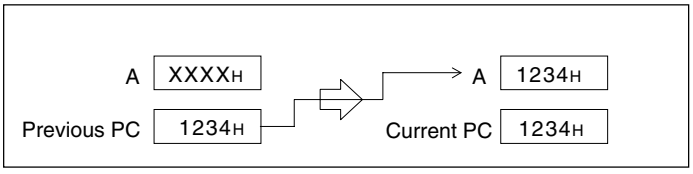


● MOVW A, PC

This instruction performs the operation which is the reverse of that performed by JMP @A. That is, the instruction stores the contents of PC in A. When the instruction is executed in the main routine, so that a specific subroutine is called, whether A contains a predetermined value can be checked by the subroutine. This can be used to determine that the branch source is not any unexpected section of the program and to check for program runaway.

Figure B.3-2 "MOVW A, PC" shows a summary of the instruction.

Figure B.3-2 MOVW A, PC



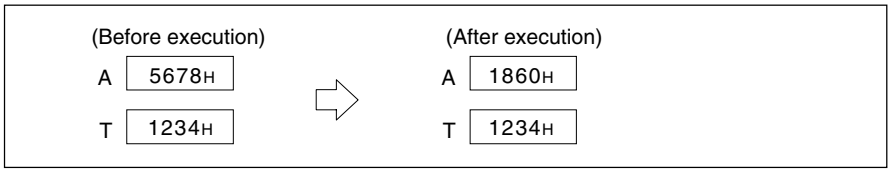
After the MOVW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of MOVW A, PC. Accordingly, Figure B.3-2 "MOVW A, PC" shows that A contains 1234_H, which is the address of the operation code of the instruction that follows MOVW A, PC.

● MULU A

This instruction performs an unsigned multiplication of AL (lower eight bits of the accumulator) and TL (lower eight bits of the temporary accumulator), and stores the 16-bit result in A. The contents of T (temporary accumulator) do not change. The contents of AH (higher eight bits of the accumulator) and TH (higher eight bits of the temporary accumulator) before execution of the instruction are not used for the operation. The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a multiplication.

Figure B.3-3 "MULU" shows a summary of the instruction.

Figure B.3-3 MULU



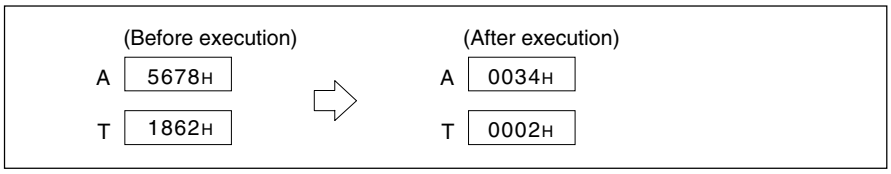
● DIVU A

This instruction divides the 16-bit value in T by the unsigned 8-bit value in AL, and stores the 8-bit result and the 8-bit remainder in AL and TL, respectively. A value of 0 is set to both AH and TH. The contents of AH before execution of the instruction are not used for the operation. An unpredictable result is produced from data that results in more than eight bits. In addition, there is no indication of the result having more than eight bits. Therefore, if it is likely that data will cause a result of more than eight bits, the data must be checked to ensure that the result will not have more than eight bits before it is used.

The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a division.

Figure B.3-4 "DIVU A" shows a summary of the instruction.

Figure B.3-4 DIVU A

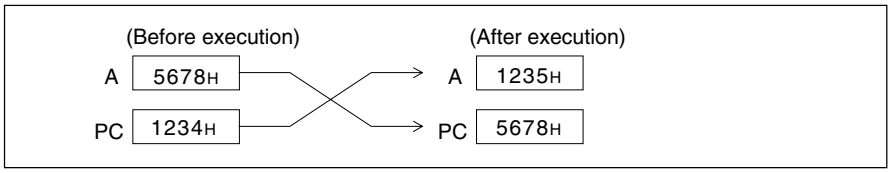


● XCHW A, PC

This instruction swaps the contents of A and PC, resulting in a branch to the address contained in A before execution of the instruction. After the instruction is executed, A contains the address that follows the address of the operation code of XCHW A, PC. This instruction is effective especially when it is used in the main routine to specify a table for use in a subroutine.

Figure B.3-5 "XCHW A, PC" shows a summary of the instruction.

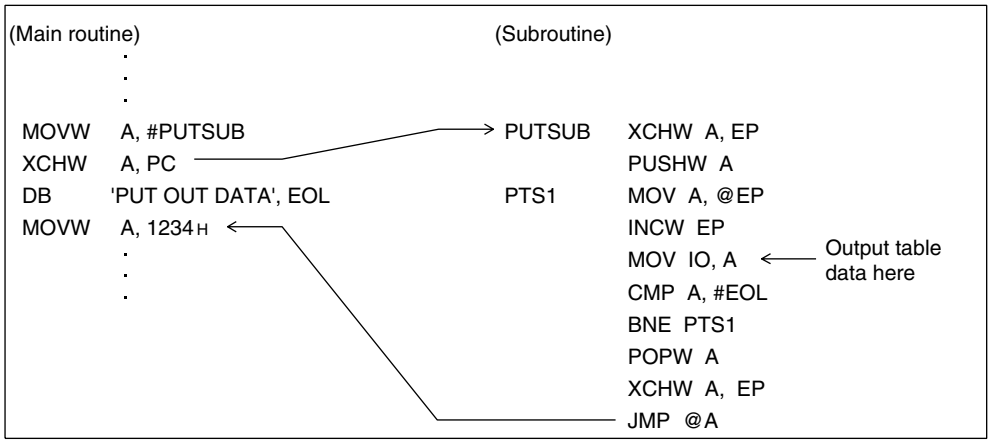
Figure B.3-5 XCHW A, PC



After the XCHW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of XCHW A, PC. Accordingly, Figure B.3-5 "XCHW A, PC" shows that A contains 1235_H, which is the address of the operation code of the instruction that follows XCHW A, PC. This is why 1235_H is stored instead of 1234_H.

Figure B.3-6 "Example of using XCHW A, PC" shows an assembly language example.

Figure B.3-6 Example of using XCHW A, PC

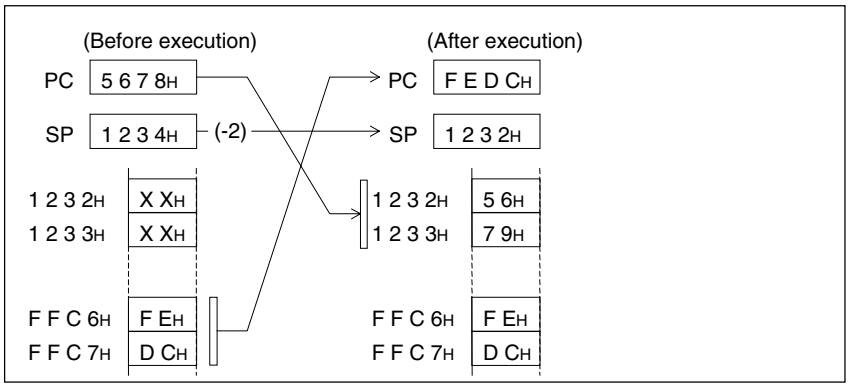


● CALLV #vct

This instruction is used to branch to a subroutine address stored in the vector table. The instruction saves the return address (contents of PC) in the location at the address contained in SP (stack pointer), and uses vector addressing to cause a branch to the address stored in the vector table. Because CALLV #vct is a 1-byte instruction, the use of this instruction for frequently used subroutines can reduce the entire program size.

Figure B.3-7 "Example of executing CALLV #3" shows a summary of the instruction.

Figure B.3-7 Example of executing CALLV #3



After the CALLV #vct instruction is executed, the contents of PC saved on the stack area are the address of the operation code of the next instruction, rather than the address of the operation code of CALLV #vct. Accordingly, Figure B.3-7 "Example of executing CALLV #3" shows that the value saved in the stack (1232_H and 1233_H) is 5679_H, which is the address of the operation code of the instruction that follows CALLV #vct (return address).

B.4 Bit Manipulation Instructions (SETB, CLRB)

Some bits of peripheral function registers include bits that are read by a bit manipulation instruction differently than usual.

■ Read-modify-write operation

By using these bit manipulation instructions, only the specified bit in a register or RAM location can be set to 1 (SETB) or cleared to 0 (CLRB). However, as the CPU operates on data in 8-bit units, the actual operation (read-modify-write operation) involves a sequence of steps: 8-bit data is read, the specified bit is changed, and the data is written back to the location at the original address.

Table B.4-1 "Bus operation for bit manipulation instructions" shows bus operation for bit manipulation instructions.

Table B.4-1 Bus operation for bit manipulation instructions

CODE	MNEMONIC	TO	Cycle	Address bus	Data bus	$\overline{RD}$	$\overline{WR}$	RMW
A0 to A7	CLRB dir:b	4	1	N+1	dir	0	1	0
			2	dir address	Data	0	1	1
A8 to AF	SETB dir:b		3	dir address	Data	1	0	0
			4	N+2	Next instruction	0	1	0

■ Read operation upon the execution of bit manipulation instructions

For some I/O ports and for the interrupt request flag bits, the value to be read differs between a normal read operation and a read-modify-write operation.

● I/O ports (during a bit manipulation)

From some I/O ports, an I/O pin value is read during a normal read operation, while an output latch value is read during a bit manipulation. This prevents the other output latch bits from being changed accidentally, regardless of the I/O directions and states of the pins.

● Interrupt request flag bits (during a bit manipulation)

An interrupt request flag bit functions as a flag bit indicating whether an interrupt request exists during a normal read operation. However, 1 is always read from this bit during a bit manipulation. This prevents the flag from being cleared accidentally by a value of 0 which would otherwise be written to the interrupt request flag bit when another bit is manipulated.

B.5 F²MC-8L Instructions

Table B.5-1 "Transfer instructions" to Table B.5-4 "Other instructions" list the instructions used with the F²MC-8L.

■ Transfer instructions

Table B.5-1 Transfer instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	MOV dir, A	3	2	(dir)<--(A)	-	-	-	-	-	-	-	45
2	MOV @IX+off, A	4	2	((IX)+off)<--(A)	-	-	-	-	-	-	-	46
3	MOV ext, A	4	3	(ext)<--(A)	-	-	-	-	-	-	-	61
4	MOV @EP, A	3	1	((EP))<--(A)	-	-	-	-	-	-	-	47
5	MOV Ri, A	3	1	(Ri)<--(A)	-	-	-	-	-	-	-	48 to 4F
6	MOV A, #d8	2	2	(A)<--d8	AL	-	-	+	+	-	-	04
7	MOV A, dir	3	2	(A)<--(dir)	AL	-	-	+	+	-	-	05
8	MOV A, @IX+off	4	2	(A)<--((IX)+off)	AL	-	-	+	+	-	-	06
9	MOV A, ext	4	3	(A)<--(ext)	AL	-	-	+	+	-	-	60
10	MOV A, @A	3	1	(A)<--((A))	AL	-	-	+	+	-	-	92
11	MOV A, @EP	3	1	(A)<--((EP))	AL	-	-	+	+	-	-	07
12	MOV A, Ri	3	1	(A)<--(Ri)	AL	-	-	+	+	-	-	08 to 0F
13	MOV dir, #d8	4	3	(dir)<--d8	-	-	-	-	-	-	-	85
14	MOV @IX+off, #d8	5	3	((IX)+off)<--d8	-	-	-	-	-	-	-	86
15	MOV @EP, #d8	4	2	((EP))<--d8	-	-	-	-	-	-	-	87
16	MOV Ri, #d8	4	2	(Ri)<--d8	-	-	-	-	-	-	-	88 to 8F
17	MOVW dir, A	4	2	(dir)<--(AH), (dir+1)<--(AL)	-	-	-	-	-	-	-	D5
18	MOVW @IX+off, A	5	2	((IX)+off)<--(AH), ((IX)+off+1)<--(AL)	-	-	-	-	-	-	-	D6
19	MOVW ext, A	5	3	(ext)<--(AH), (ext+1)<--(AL)	-	-	-	-	-	-	-	D4
20	MOVW @EP, A	4	1	((EP))<--(AH), ((EP)+1)<--(AL)	-	-	-	-	-	-	-	D7
21	MOVW EP, A	2	1	(EP)<--(A)	-	-	-	-	-	-	-	E3
22	MOVW A, #d16	3	3	(A)<--d16	AL	AH	dH	+	+	-	-	E4
23	MOVW A, dir	4	2	(AH)<--(dir), (AL)<--(dir+1)	AL	AH	dH	+	+	-	-	C5

Table B.5-1 Transfer instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
24	MOVW A, @IX+off	5	2	(AH)<--((IX)+off), (AL)<--((IX)+off+1)	AL	AH	dH	+	+	-	-	C6
25	MOVW A, ext	5	3	(AH)<--(ext), (AL)<--(ext+1)	AL	AH	dH	+	+	-	-	C4
26	MOVW A, @A	4	1	(AH)<--((A)), (AL)<--((A)+1)	AL	AH	dH	+	+	-	-	93
27	MOVW A, @EP	4	1	(AH)<--((EP)), (AL)<--((EP)+1)	AL	AH	dH	+	+	-	-	C7
28	MOVW A, EP	2	1	(A)<--(EP)	-	-	dH	-	-	-	-	F3
29	MOVW EP, #d16	3	3	(EP)<--d16	-	-	-	-	-	-	-	E7
30	MOVW IX, A	2	1	(IX)<--(A)	-	-	-	-	-	-	-	E2
31	MOVW A, IX	2	1	(A)<--(IX)	-	-	dH	-	-	-	-	F2
32	MOVW SP, A	2	1	(SP)<--(A)	-	-	-	-	-	-	-	E1
33	MOVW A, SP	2	1	(A)<--(SP)	-	-	dH	-	-	-	-	F1
34	MOV @A, T	3	1	((A))<--(T)	-	-	-	-	-	-	-	82
35	MOVW @A, T	4	1	((A))<--(TH), ((A)+1)<--(TL)	-	-	-	-	-	-	-	83
36	MOVW IX, #d16	3	3	(IX)<--d16	-	-	-	-	-	-	-	E6
37	MOVW A, PS	2	1	(A)<--(PS)	-	-	dH	-	-	-	-	70
38	MOVW PS, A	2	1	(PS)<--(A)	-	-	-	+	+	+	+	71
39	MOVW SP, #d16	3	3	(SP)<--d16	-	-	-	-	-	-	-	E5
40	SWAP	2	1	(AH)<-- -->(AL)	-	-	AL	-	-	-	-	10
41	SETB dir:b	4	2	(dir):b <--1	-	-	-	-	-	-	-	A8 to AF
42	CLRB dir:b	4	2	(dir):b <--0	-	-	-	-	-	-	-	A0 to A7
43	XCH A, T	2	1	(AL)<-- -->(TL)	AL	-	-	-	-	-	-	42
44	XCHW A, T	3	1	(A)<-- -->(T)	AL	AH	dH	-	-	-	-	43
45	XCHW A, EP	3	1	(A)<-- -->(EP)	-	-	dH	-	-	-	-	F7
46	XCHW A, IX	3	1	(A)<-- -->(IX)	-	-	dH	-	-	-	-	F6
47	XCHW A, SP	3	1	(A)<-- -->(SP)	-	-	dH	-	-	-	-	F5
48	MOVW A, PC	2	1	(A)<--(PC)	-	-	dH	-	-	-	-	F0

Note:

In automatic transfer to T during byte transfer to A, AL is transferred to TL.

If an instruction has two or more operands, they are assumed to be saved in the order indicated by MNEMONIC.

■ Arithmetic instructions

Table B.5-2 Arithmetic operation instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	ADDC A, Ri	3	1	(A)<--(A)+(Ri)+C	-	-	-	+	+	+	+	28 to 2F
2	ADDC A, #d8	2	2	(A)<--(A)+d8+C	-	-	-	+	+	+	+	24
3	ADDC A, dir	3	2	(A)<--(A)+(dir)+C	-	-	-	+	+	+	+	25
4	ADDC A, @IX+off	4	2	(A)<--(A)+((IX)+off)+C	-	-	-	+	+	+	+	26
5	ADDC A, @EP	3	1	(A)<--(A)+((EP))+C	-	-	-	+	+	+	+	27
6	ADDCW A	3	1	(A)<--(A)+(T)+C	-	-	dH	+	+	+	+	23
7	ADDC A	2	1	(AL)<--(AL)+(TL)+C	-	-	-	+	+	+	+	22
8	SUBC A, Ri	3	1	(A)<--(A)-(Ri)-C	-	-	-	+	+	+	+	38 to 3F
9	SUBC A, #d8	2	2	(A)<--(A)-d8-C	-	-	-	+	+	+	+	34
10	SUBC A, dir	3	2	(A)<--(A)-(dir)-C	-	-	-	+	+	+	+	35
11	SUBC A, @IX+off	4	2	(A)<--(A)-((IX)+off)-C	-	-	-	+	+	+	+	36
12	SUBC A, @EP	3	1	(A)<--(A)-((EP))-C	-	-	-	+	+	+	+	37
13	SUBCW A	3	1	(A)<--(T)-(A)-C	-	-	dH	+	+	+	+	33
14	SUBC A	2	1	(AL)<--(TL)-(AL)-C	-	-	-	+	+	+	+	32
15	INC Ri	4	1	(Ri)<--(Ri)+1	-	-	-	+	+	+	-	C8 to CF
16	INCW EP	3	1	(EP)<--(EP)+1	-	-	-	-	-	-	-	C3
17	INCW IX	3	1	(IX)<--(IX)+1	-	-	-	-	-	-	-	C2
18	INCW A	3	1	(A)<--(A)+1	-	-	dH	+	+	-	-	C0
19	DEC Ri	4	1	(Ri)<--(Ri)-1	-	-	-	+	+	+	-	D8 to DF
20	DECW EP	3	1	(EP)<--(EP)-1	-	-	-	-	-	-	-	D3
21	DECW IX	3	1	(IX)<--(IX)-1	-	-	-	-	-	-	-	D2
22	DECW A	3	1	(A)<--(A)-1	-	-	dH	+	+	-	-	D0
23	MULU A	19	1	(A)<--(AL)x(TL)	-	-	dH	-	-	-	-	01
24	DIVU A	21	1	(A)<--(T)/(AL), MOD -->(T)	dL	00	00	-	-	-	-	11
25	ANDW A	3	1	(A)<--(A) ∧ (T)	-	-	dH	+	+	R	-	63
26	ORW A	3	1	(A)<--(A) ∨ (T)	-	-	dH	+	+	R	-	73
27	XORW A	3	1	(A)<--(A) ∀ (T)	-	-	dH	+	+	R	-	53
28	CMP A	2	1	(TL)-(AL)	-	-	-	+	+	+	+	12

Table B.5-2 Arithmetic operation instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
29	CMPW A	3	1	(T)-(A)	-	-	-	+	+	+	+	13
30	RORC A	2	1	→ C --> A ┐	-	-	-	+	+	-	+	03
31	ROLC A	2	1	┌ C <-- A ←	-	-	-	+	+	-	+	02
32	CMP A, #d8	2	2	(A)-d8	-	-	-	+	+	+	+	14
33	CMP A, dir	3	2	(A)-(dir)	-	-	-	+	+	+	+	15
34	CMP A, @EP	3	1	(A)-((EP))	-	-	-	+	+	+	+	17
35	CMP A, @IX+off	4	2	(A)-((IX)+off)	-	-	-	+	+	+	+	16
36	CMP A, Ri	3	1	(A)-(Ri)	-	-	-	+	+	+	+	18 to 1F
37	DAA	2	1	decimal adjust for addition	-	-	-	+	+	+	+	84
38	DAS	2	1	decimal adjust for subtraction	-	-	-	+	+	+	+	94
39	XOR A	2	1	(A)<--(AL) ∀ (TL)	-	-	-	+	+	R	-	52
40	XOR A, #d8	2	2	(A)<--(AL) ∀ d8	-	-	-	+	+	R	-	54
41	XOR A, dir	3	2	(A)<--(AL) ∀ (dir)	-	-	-	+	+	R	-	55
42	XOR A, @EP	3	1	(A)<--(AL) ∀ ((EP))	-	-	-	+	+	R	-	57
43	XOR A, @IX+off	4	2	(A)<--(AL) ∀ ((IX)+off)	-	-	-	+	+	R	-	56
44	XOR A, Ri	3	1	(A)<--(AL) ∀ (Ri)	-	-	-	+	+	R	-	58 to 5F
45	AND A	2	1	(A)<--(AL) ∧ (TL)	-	-	-	+	+	R	-	62
46	AND A, #d8	2	2	(A)<--(AL) ∧ d8	-	-	-	+	+	R	-	64
47	AND A, dir	3	2	(A)<--(AL) ∧ (dir)	-	-	-	+	+	R	-	65
48	AND A, @EP	3	1	(A)<--(AL) ∧ ((EP))	-	-	-	+	+	R	-	67
49	AND A, @IX+off	4	2	(A)<--(AL) ∧ ((IX)+off)	-	-	-	+	+	R	-	66
50	AND A, Ri	3	1	(A)<--(AL) ∧ (Ri)	-	-	-	+	+	R	-	68 to 6F
51	OR A	2	1	(A)<--(AL) ∨ (TL)	-	-	-	+	+	R	-	72
52	OR A, #d8	2	2	(A)<--(AL) ∨ d8	-	-	-	+	+	R	-	74
53	OR A, dir	3	2	(A)<--(AL) ∨ (dir)	-	-	-	+	+	R	-	75

Table B.5-2 Arithmetic operation instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
54	OR A, @EP	3	1	(A)<--(AL) ∨ ((EP))	-	-	-	+	+	R	-	77
55	OR A, @IX+off	4	2	(A)<--(AL) ∨ ((IX)+off)	-	-	-	+	+	R	-	76
56	OR A, Ri	3	1	(A)<--(AL) ∨ (Ri)	-	-	-	+	+	R	-	78 to 7F
57	CMP dir, #d8	5	3	(dir)-d8	-	-	-	+	+	+	+	95
58	CMP @EP, #d8	4	2	((EP))-d8	-	-	-	+	+	+	+	97
59	CMP @IX+off, #d8	5	3	((IX)+off)-d8	-	-	-	+	+	+	+	96
60	CMP Ri, #d8	4	2	(Ri)-d8	-	-	-	+	+	+	+	98 to 9F
61	INCW SP	3	1	(SP)<--(SP)+1	-	-	-	-	-	-	-	C1
62	DECW SP	3	1	(SP)<--(SP)-1	-	-	-	-	-	-	-	D1

■ Branch instructions

Table B.5-3 Branch instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	BZ/BEQ rel	3	2	if Z=1 then PC<--PC+rel	-	-	-	-	-	-	-	FD
2	BNZ/BNE rel	3	2	if Z=0 then PC<--PC+rel	-	-	-	-	-	-	-	FC
3	BC/BLO rel	3	2	if C=1 then PC<--PC+rel	-	-	-	-	-	-	-	F9
4	BNC/BHS rel	3	2	if C=0 then PC<--PC+rel	-	-	-	-	-	-	-	F8
5	BN rel	3	2	if N=1 then PC<--PC+rel	-	-	-	-	-	-	-	FB
6	BP rel	3	2	if N=0 then PC<--PC+rel	-	-	-	-	-	-	-	FA
7	BLT rel	3	2	if V ∀ N=1 then PC<--PC+rel	-	-	-	-	-	-	-	FF
8	BGE rel	3	2	if V ∀ N=0 then PC<--PC+rel	-	-	-	-	-	-	-	FE
9	BBC dir:b, rel	5	3	if (dir:b)=0 then PC<--PC+rel	-	-	-	-	+	-	-	B0 to B7
10	BBS dir:b, rel	5	3	if (dir:b)=1 then PC<--PC+rel	-	-	-	-	+	-	-	B8 to BF
11	JMP @A	2	1	(PC)<--(A)	-	-	-	-	-	-	-	E0
12	JMP ext	3	3	(PC)<--ext	-	-	-	-	-	-	-	21
13	CALLV #vct	6	1	vector call	-	-	-	-	-	-	-	E8 to EF
14	CALL ext	6	3	subroutine call	-	-	-	-	-	-	-	31
15	XCHW A, PC	3	1	(PC)<--(A), (A)<--(PC)+1	-	-	dH	-	-	-	-	F4
16	RET	4	1	return from subroutine	-	-	-	-	-	-	-	20
17	RETI	6	1	return from interrupt	-	-	-	restore				30

■ Other instructions

Table B.5-4 Other instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	PUSHW A	4	1		-	-	-	-	-	-	-	40
2	POPW A	4	1		-	-	dH	-	-	-	-	50
3	PUSHW IX	4	1		-	-	-	-	-	-	-	41
4	POPW IX	4	1		-	-	-	-	-	-	-	51
5	NOP	1	1		-	-	-	-	-	-	-	00
6	CLRC	1	1		-	-	-	-	-	-	R	81
7	SETC	1	1		-	-	-	-	-	-	S	91
8	CLRI	1	1		-	-	-	-	-	-	-	80
9	SETI	1	1		-	-	-	-	-	-	-	90

B.6 Instruction map

Table B.6-1 "F²MC-8L instruction map" shows the F²MC-8L instruction map.

■ Instruction map

Table B.6-1 F²MC-8L instruction map

H / L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SWAP	RET	RETI	PUSHW A	POPW A	MOV A, ext	MOVW A, PS	CLRI	SETI	CLRB dir : 0	BBC dir : 0, rel	INCW A	DECW A	JMP @A	MOVW A, PC
1	MULU A	DIVU A	JMP addr16	CALL addr16	PUSHW IX	POPW IX	MOV ext, A	MOVW PS, A	CLRC	SETC	CLRB dir : 1	BBC dir : 1, rel	INCW SP	DECW SP	MOVW SP, A	MOVW A, SP
2	ROLC A	CMP A	ADDC A	SUBC A	XCH A, T	XOR A	AND A	OR A	MOV @A, T	MOV A, @A	CLRB dir : 2	BBC dir : 2, rel	INCW IX	DECW IX	MOVW IX, A	MOVW A, IX
3	RORC A	CMPW A	ADDCW A	SUBCW A	XCHW A, T	XORW A	ANDW A	ORW A	MOVW @A, T	MOVW A, @A	CLRB dir : 3	BBC dir : 3, rel	INCW EP	DECW EP	MOVW EP, A	MOVW A, EP
4	MOV A, #d8	CMP A, #d8	ADDC A, #d8	SUBC A, #d8		XOR A, #d8	AND A, #d8	OR A, #d8	DAA	DAS	CLRB dir : 4	BBC dir : 4, rel	MOVW A, ext	MOVW ext, A	MOVW A, #d16	XCHW A, PC
5	MOV A, dir	CMP A, dir	ADDC A, dir	SUBC A, dir	MOV dir, A	XOR A, dir	AND A, dir	OR A, dir	MOV dir, #d8	CMP dir, #d8	CLRB dir : 5	BBC dir : 5, rel	MOVW A, dir	MOVW dir, A	MOVW SP, #d16	XCHW A, SP
6	MOV A, @IX+d	CMP A, @IX+d	ADDC A, @IX+d	SUBC A, @IX+d	MOV @IX+d, A	XOR A, @IX+d	AND A, @IX+d	OR A, @IX+d	MOV @IX+d,#d8	CMP @IX+d,#d8	CLRB dir : 6	BBC dir : 6, rel	MOVW A, @IX+d	MOVW @IX+d, A	MOVW IX, #d16	XCHW A, IX
7	MOV A, @EP	CMP A, @EP	ADDC A, @EP	SUBC A, @EP	MOV @EP, A	XOR A, @EP	AND A, @EP	OR A, @EP	MOV @EP#, d8	CMP @EP#, d8	CLRB dir : 7	BBC dir : 7, rel	MOVW A, @EP	MOVW @EP, A	MOVW EP, #d16	XCHW A, EP
8	MOV A, R0	CMP A, R0	ADDC A, R0	SUBC A, R0	MOV R0, A	XOR A, R0	AND A, R0	OR A, R0	MOV R0, #d8	CMP R0, #d8	SETB dir : 0	BBS dir : 0, rel	INC R0	DEC R0	CALLV #0	BNC rel
9	MOV A, R1	CMP A, R1	ADDC A, R1	SUBC A, R1	MOV R1, A	XOR A, R1	AND A, R1	OR A, R1	MOV R1, #d8	CMP R1, #d8	SETB dir : 1	BBS dir : 1, rel	INC R1	DEC R1	CALLV #1	BC rel
A	MOV A, R2	CMP A, R2	ADDC A, R2	SUBC A, R2	MOV R2, A	XOR A, R2	AND A, R2	OR A, R2	MOV R2, #d8	CMP R2, #d8	SETB dir : 2	BBS dir : 2, rel	INC R2	DEC R2	CALLV #2	BP rel
B	MOV A, R3	CMP A, R3	ADDC A, R3	SUBC A, R3	MOV R3, A	XOR A, R3	AND A, R3	OR A, R3	MOV R3, #d8	CMP R3, #d8	SETB dir : 3	BBS dir : 3, rel	INC R3	DEC R3	CALLV #3	BN rel
C	MOV A, R4	CMP A, R4	ADDC A, R4	SUBC A, R4	MOV R4, A	XOR A, R4	AND A, R4	OR A, R4	MOV R4, #d8	CMP R4, #d8	SETB dir : 4	BBS dir : 4, rel	INC R4	DEC R4	CALLV #4	BNZ rel
D	MOV A, R5	CMP A, R5	ADDC A, R5	SUBC A, R5	MOV R5, A	XOR A, R5	AND A, R5	OR A, R5	MOV R5, #d8	CMP R5, #d8	SETB dir : 5	BBS dir : 5, rel	INC R5	DEC R5	CALLV #5	BZ rel
E	MOV A, R6	CMP A, R6	ADDC A, R6	SUBC A, R6	MOV R6, A	XOR A, R6	AND A, R6	OR A, R6	MOV R6, #d8	CMP R6, #d8	SETB dir : 6	BBS dir : 6, rel	INC R6	DEC R6	CALLV #6	BGE rel
F	MOV A, R7	CMP A, R7	ADDC A, R7	SUBC A, R7	MOV R7, A	XOR A, R7	AND A, R7	OR A, R7	MOV R7, #d8	CMP R7, #d8	SETB dir : 7	BBS dir : 7, rel	INC R7	DEC R7	CALLV #7	BLT rel

APPENDIX C Mask Options

This appendix lists the mask options for the MB89470 series.

■ Mask options

Table C-1 Mask options

No.	Part number	MB89475	MB89P475	MB89PV470
	Specifying procedure	Specify when ordering masking	Setting not possible	Setting not possible
1	Selection of clock mode • Single clock mode • Dual clock mode	Selectable	101/102: Single clock 201/202: Dual clock	101: Single clock 201: Dual clock
2	Selection of OTPROM content protection feature • No protection feature • With protection feature	--	101/201: No protection 102/202: with protection	--
3	Selection of oscillation stabilization time (OSC) • The initial value of the oscillation stabilization time for the main clock can be set by selecting the values of the WTM1 and WTM0 bits on the right.	Selectable OSC = 1: $2^{14}/F_{CH}$ OSC = 2: $2^{17}/F_{CH}$ OSC = 3: $2^{18}/F_{CH}$	Fixed to oscillation stabilization time of $2^{18}/F_{CH}$	Fixed to oscillation stabilization time of $2^{18}/F_{CH}$
4	Selection of power-on stabilization time • Nil • $2^{17}/F_{CH}$	Selectable	Fixed to power-on stabilization time of $2^{17}/F_{CH}$	Fixed to nil

■ Type of Order

Table C-2 Type of order

Type	Package	Remark
MB89475PFV MB89P475PFV-101 MB89P475PFV-201	48-pin plastic LQFP (FPT-48P-M05)	101: Single clock 201: Dual clock
MB89475PFM MB89P475PFM-101 MB89P475PFM-201	48-pin plastic QFP (FPT-48P-M13)	
MB89475P-SH MB89P475P-SH-101 MB89P475P-SH-201	48-pin plastic SH-DIP (DIP-48P-M01)	
MB89PV470CF-101 MB89PV470CF-201	48-pin ceramic MQFP (MQP-48C-P02)	

APPENDIX D Programming Specifications for One-Time PROM And EPROM Microcontroller

This appendix describes the programming specifications for one-time PROM and EPROM microcontroller.

D.1 Programming PROM

In MB89P475, there is a built-in PROM which can be programmed with Serial programmer by using dedicated adapter. Optionally, it can also be programmed with a general-purpose EPROM programmer by using dedicated adapter.

■ One-time PROM product

There is a one-time PROM product, MB89P475 in this series.

There is a built-in PROM in MB89P475. The built-in PROM can be programmed with Serail programmer by using dedicated adaptor. Please refer Section D.2 "Programming One-time PROM Microcontroller with Serial Programmer" for detail.

The PROM can also be programmed with a parallel OTPROM programmer by using the dedicated adaptor. Please refer Section D.3 "Programming One-time PROM Microcontroller with Parallel Programmer" for detail.

The following show the one-time PROM product with different option.

Table D.1-1 One-time PROM programming option

Product	Programmable with Serial Programmer	Programmable with Parallel Programmer
MB89P475-101 MB89P475-201	Yes	Yes
MB89P475-102 MB89P475-202	Yes	No

■ Piggyback/evaluation device

There is a piggyback/evaluation product MB89PV470 in this series.

There is no built-in PROM in MB89PV470.

The EPROM used by MB89PV470 can be programmed with a general-purpose EPROM programmer by using the dedicated adaptor. Please refer Section D.4 "Programming EPROM for Piggyback/Evaluation Device" for detail.

D.2 Programming One-time PROM Microcontroller with Serial Programmer

This section describes programming One-time PROM microcontroller with serial programmer.

■ Programming the OTPROM

To program the OTPROM using MCU programmer MB91919-001.

Inquiry : Fujitsu Microelectronics Asia Pte Ltd. : TEL (65)-2810770 FAX (65)-2810220

■ Programming adaptor for OTPROM

To program the OTPROM using MCU programmer MB91919-001, use the programming adapter listed below.

Package	Compatible socker adaptor
DIP-48P-M01	MB91919-805+MB91919-800
FPT-48P-M05	MB91919-806+MB91919-800
FPT-48P-M13	MB91919-807+MB91919-800

Inquiries : Fujitsu Microelectronics Asia Pte Ltd. : TEL (65)-2810770 FAX (65)-2810220

■ OTPROM content protection

For product with OTPROM content protection feature (MB89P475-102, MB89P475-202), OTPROM content can be read using serial programmer if the OTPROM content protection mechanism is not activated.

One predefined area of the OTPROM ($FFFC_H$) is assigned to be used for preventing the read access of OTPROM content. If the protection code "00_H" is written in this address ($FFFC_H$), the OTPROM content cannot be read by any serial programmer.

Reference:

The program written into the OTPROM cannot be verified once the OTPROM protection code is written ("00_H" in $FFFC_H$). It is advised to write the OTPROM protection code at last.

■ Programming yield

All bits cannot be programmed at Fujitsu shipping test to a blanked OTPROM microcomputer, due to its nature. For this reason, a programming yield of 100% cannot be assured at all times.

D.3 Programming One-time PROM Microcontroller with Parallel Programmer

This section describes programming One-time PROM microcontroller with parallel programmer.

■ OTP programming adaptor and recommended ROM programmer

Package name	Applicable adapter model		Recommended writer maker and writer	
	Fujitsu Microelectronics Asia Pte Ltd.	Sunhayato Co., Ltd	Ando Electric Co., Ltd.	Fujitsu MCU programmer
				MB91919-001
DIP-48P-M01	MB91919-601	ROM2-48SD-32DP-8LA	AF9708* AF9700* AF9723*	Available
FPT-48P-M05	MB91919-602	ROM2-48LQF-32DP-8LA2		
FPT-48P-M13	MB91919-603	ROM2-48QF-32DP-8LA2		

* : For the writer and its version, contact Flah Support Group, Inc.
Inquiries : Fujitsu Microelectronics Asia Pte Ltd. : TEL (65)-2810770
Sunhayato Corp.: TEL (81)-33984-7791
FAX (81)-33971-0535
E-mail : adapter@sunhayato.co.jp

Flah Support Group, Inc: FAX :(81)-53-428-8377
E-mail : support@j-fsg.co.jp

■ Writing data to the OTPROM

1. Set the OTPROM writer for the CU50-OTP (device code: cdB6DC).
2. Load the program data to the OTPROM writer.
3. Write data using the OTPROM writer.

■ Programming yield

All bits cannot be programmed at Fujitsu Shipping test to a blanked OTPROM microcontroller, due to its nature. For this reason, a programming yield of 100 % cannot be assured at all times.

■ OTPROM content protection

This feature is available in MB89P475-102 and MB89P475-202 only. It is not available in product which can be programmed by parallel programmer, i.e MB89P475-101 and MB89P475-201.

D.4 Programming EPROM for Piggyback/Evaluation Device

This section describes programming EPROM for piggyback/evaluation device.

■ EPROM for use

MBM27C256A-20TVM

■ Programming socket adaptor

To program to the PROM using an EPROM programmer, use the socket adaptor (manufacturer: Sunhayato Corp.) listed below.

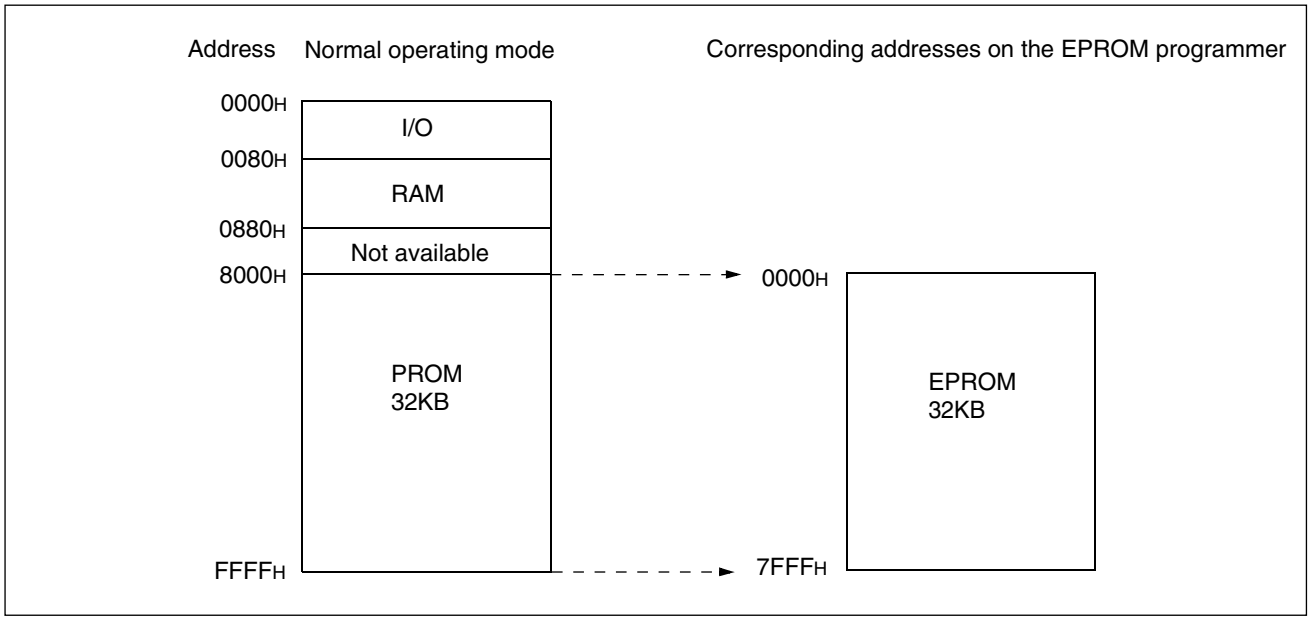
Table D.4-1 Programming socket adaptor

Package	Adaptor socket part number
LCC-32 (Rectangle)	ROM-32LC-28DP-S

Inquiries: Sunhayato Corp.: TEL (81)-33984-7791
FAX (81)-33971-0535
E-mail : adapter@sunhayato.co.jp

■ Memory space

Figure D.4-1 Memory map of piggyback/evaluation device



■ Programming to EPROM

1. Set the EPROM programmer to the MBM27C256A.
2. Load program data into the EPROM programmer at 0000_H to $7FFF_H$.
3. Program to 0000_H to $7FFF_H$ with the EPROM programmer.

APPENDIX E MB89470 Series Pin States

This section describes the pin states of the MB89470 series in each mode.

■ Pin states in each mode

Table E-1 Pin states in each mode

Pin name	Normal operation	Sleep mode	Stop mode SPL="0"	Stop mode SPL="1"	During reset
X0	Oscillator input	Oscillator input	Hi-Z	Hi-Z	Oscillator input
X1	Oscillator output	Oscillator output	output "H"	Output "H"	Oscillator output
MODE	Mode input	Mode input	Mode input	Mode input	Mode input
$\overline{RST}$	Reset I/O	Reset input	Reset input	Reset input	Reset I/O
P00/AN0 to P07/AN7	Port or AD input	Hold / peripheral I/O	Hold	"H" (if pull-up) Hi-Z (if not pull-up)	Hi-Z
P10/INT10 to P13/INT13	Port or external interrupt 1 input	Hold/external interrupt 1 input	Hold/external interrupt 1 input	"H" (if pull-up) Hi-Z (if not pull-up) / external interrupt 1 input	Hi-Z
P14/EC1	Port / peripheral I/O	Hold / peripheral I/O	Hold	"H" (if pull-up) Hi-Z (if not pull-up)	Hi-Z
P15/TO1					
P16/EC2					
P17/TO2					
P20/SCK1					
P21/SO1					
P22/SI1					
P23/PWC					
P24/PWM					
P25/SI2					
P26/SO2					
P27/SCK2					
P30/BUZ					
P31 to P36					
P40/X0A	Port or Oscillator input	Port or Oscillator input	Port or Hi-Z	Port or Hi-Z	Port or Oscillator input
P41/X1A	Port or Oscillator output	Port or Oscillator output	Port or output "H"	Port or output "H"	Port or Oscillator output
P42	Port	Port	Port	Port	Port
P50/$\overline{INT20}$ to P54/$\overline{INT24}$	Port or external interrupt 2 input	Hold/external interrupt 2 input	Hold/external interrupt 2 input	"H" (if pull-up) Hi-Z (if not pull-up) / external interrupt 2 input	Hi-Z

INDEX

The index follows on the next page.
This is listed in alphabetic order.

Index

F

G

H

I

L

M

O

P

R

S

T

U

V

W

CM25-10147-1E

FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL

F^2MC-8L
8-BIT MICROCONTROLLER
MB89470 Series
HARDWARE MANUAL

July 2003 the first edition

Published **FUJITSU LIMITED** Electronic Devices

Edited Business Promotion Dept.