



The following document contains information on Cypress products. Although the document is marked with the name “Spansion” and “Fujitsu”, the company that originally developed the specification, Cypress will continue to offer these products to new and existing customers.

Continuity of Specifications

There is no change to this document as a result of offering the device as a Cypress product. Any changes that have been made are the result of normal document improvements and are noted in the document history page, where supported. Future revisions will occur when appropriate, and changes will be noted in a document history page.

Continuity of Ordering Part Numbers

Cypress continues to support existing part numbers. To order these products, please use only the Ordering Part Numbers listed in this document.

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress (NASDAQ: CY) delivers high-performance, high-quality solutions at the heart of today’s most advanced embedded systems, from automotive, industrial and networking platforms to highly interactive consumer and mobile devices. With a broad, differentiated product portfolio that includes NOR flash memories, F-RAM™ and SRAM, Traveo™ microcontrollers, the industry’s only PSoC® programmable system-on-chip solutions, analog and PMIC Power Management ICs, CapSense® capacitive touch-sensing controllers, and Wireless BLE Bluetooth® Low-Energy and USB connectivity solutions, Cypress is committed to providing its customers worldwide with consistent innovation, best-in-class support and exceptional system value.

FR60Lite
32-BIT MICROCONTROLLER
MB91260B Series
HARDWARE MANUAL

FR60Lite

32-BIT MICROCONTROLLER

MB91260B Series

HARDWARE MANUAL

<p>Be sure to refer to the “Check Sheet” for the latest cautions on development.</p>
--

“Check Sheet” is seen at the following support page

URL : <http://www.fujitsu.com/global/services/microelectronics/product/micom/support/index.html>

“Check Sheet” lists the minimal requirement items to be checked to prevent problems beforehand in system development.

FUJITSU LIMITED

PREFACE

■ Purpose of this document and intended reader

We sincerely thank you for your continued use of Fujitsu semiconductor products.

The FR family is a line of single-chip microcontrollers based on a 32-bit high-performance RISC CPU and integrating a variety of I/O resources for embedded control applications which require high - performance, high-speed CPU processing.

The MB91260B is designed to be best suited for embedded applications which require high-performance processing power of the CPU, such as DVD players, printers, TV sets, and the PDP control.

The MB91260B series is a line of CPUs in the FR60Lite implemented by FR family.

This manual describes the functions and operations of the MB91260B Series for engineers who develop products using the MB91260B Series. Please read through this manual.

■ Trademarks

FR is the abbreviation of FUJITSU RISC CONTROLLER, which is a product of Fujitsu Limited.

■ License

Purchase of Fujitsu I²C components conveys a license under the Philips I²C Patent Rights to use, these components in an I²C system provided that the system conforms to the I²C Standard Specification as defined by Philips.

■ Sample Program

We provide sample programs free of charge to operate peripheral functions of the F²MC-16LX family. The programs can be used to check the operational specification and usage of our microcontroller device.

MPU/MCU Support Information

<http://www.fujitsu.com/global/services/microelectronics/product/micom/support/index.html>

Note: The sample programs are subject to change without notice. The software is designed to show the standard operation and usage of the product, therefore, it must be used after full evaluation before using it on your system. Moreover, we assume no liability for any damage resulting from or caused by the use of the programs.

■ Organization of this document

This manual consists of the following 18 chapters and appendix.

CHAPTER 1 OVERVIEW

This chapter provides basic information for understanding the MB91260B series as a whole, covering its features, block diagram, and functions.

CHAPTER 2 HANDLING DEVICES

This chapter provides precautions on caution of using devices the MB91260B series.

CHAPTER 3 CPU AND CONTROL UNITS

This chapter describes the basics of the architecture, specifications, and instructions of the MB91260B series of CPU cores to introduce their features.

CHAPTER 4 I/O PORTS

This chapter outlines the I/O ports and describes the configuration and functions of their registers.

CHAPTER 5 INTERRUPT CONTROLLER

This chapter outlines the interrupt control, describes its register configuration/functions and its operations, and gives an example of using the hold request cancel request.

CHAPTER 6 EXTERNAL INTERRUPT AND NMI CONTROLLER

This chapter describes the external interrupt and NMI controller, the configuration and functions of registers, and operation of the external interrupt and NMI controller.

CHAPTER 7 REALOS-RELATED HARDWARE

REALOS-related hardware is used by the realtime OS. Therefore, when using REALOS, the hardware cannot be used with the user program.

CHAPTER 8 16-BIT RELOAD TIMER

This chapter describes the 16-bit reload timer, the configuration and functions of registers, and 16-bit reload timer operation.

CHAPTER 9 PPG (Programmable Pulse Generator)

This chapter describes the overview of the PPG (Programmable Pulse Generator) timer, the configuration and functions of registers, and the operation of the PPG timer.

CHAPTER 10 PWC (Pulse Width Count: Pulse Width Measurement)

This chapter explains the overview of the pulse width counter (PWC), the register configuration and functions and the counter operation.

CHAPTER 11 MULTIFUNCTIONAL TIMER

This chapter explains the overview of the multifunction timer, the configuration and functions of registers, and operation of the multifunction timer.

CHAPTER 12 U-TIMER (16-bit Timer for UART Baud Rate Generation)

This chapter describes the U-TIMER, the configuration and functions of registers, and U-TIMER operation.

CHAPTER 13 UART

This chapter describes the overview of the UART, the configuration and functions of registers, and UART operation.

CHAPTER 14 8/10-BIT A/D CONVERTER

This chapter describes the overview of the 8/10-bit A/D converter, the configuration and functions of registers, and the operation of the 8/10-bit A/D converter.

CHAPTER 15 MULTIPLICATION AND ADDITION CALCULATOR

This chapter explains the overview of the multiply and accumulate circuit, the configuration and functions of registers, and the macro (the definition and each instruction) of the multiply and accumulate circuit.

CHAPTER 16 DMAC (DMA Controller)

This chapter explains the overview of the DMA controller (DMAC), the configuration and functions of registers, and DMAC operation.

CHAPTER 17 FLASH MEMORY

This chapter describes an overview of flash memory and the configuration and functions of registers, access modes, automatic algorithm and sector protect operations.

CHAPTER 18 SERIAL PROGRAMMING CONNECTION

This chapter describes basic configuration of serial programming and examples of the connection.

APPENDIX

This appendix contains the following items: I/O map, interrupt vector, pin status list, notes when little endian area is used, instruction lists, and the precautions on handling.

- 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.

CONTENTS

CHAPTER 1	OVERVIEW	1
1.1	Overview	2
1.2	Block Diagram	4
1.3	Package Dimension	5
1.4	Pin Assignment	7
1.5	Pin Description	9
1.6	I/O Circuit Types	18
CHAPTER 2	HANDLING DEVICES	21
2.1	Handling Devices	22
CHAPTER 3	CPU AND CONTROL UNITS	25
3.1	Memory Space	26
3.2	Internal Architecture	28
3.3	Programming Model	33
3.4	Data Structure	40
3.5	Word Alignment	41
3.6	Memory Map	42
3.7	Branch Instructions	45
3.8	EIT (Exception/Interrupt/Trap)	48
3.8.1	Interrupt Level	49
3.8.2	ICR (Interrupt Control Register)	51
3.8.3	SSP (System Stack Pointer)	52
3.8.4	TBR (Table Base Register)	53
3.8.5	Multi-EIT Servicing	56
3.8.6	Operation	58
3.9	Operation Modes	62
3.10	Reset (Device Initialization)	64
3.11	Clock Generation Control	70
3.11.1	PLL Control	71
3.11.2	Oscillation Stabilization Wait Time and PLL Lock Wait Time	72
3.11.3	Clock Distribution	74
3.11.4	Clock Frequency Division	75
3.11.5	Block Diagram of the Clock Generation Control Unit	76
3.11.6	Registers in the Clock Generation Control Unit	77
3.11.7	Peripheral Circuits in the Clock Control Unit	90
3.12	Device Status Control	93
CHAPTER 4	I/O PORTS	101
4.1	Overview of I/O Ports	102
4.2	Registers of I/O Port	104
4.3	Analog Input Ports	110

CHAPTER 5	INTERRUPT CONTROLLER	113
5.1	Overview	114
5.2	Interrupt Control Registers	118
5.3	Operation of Interrupt Controller	120
CHAPTER 6	EXTERNAL INTERRUPT AND NMI CONTROLLER	129
6.1	Overview of External Interrupt/NMI Controller	130
6.2	Registers of External Interrupt/NMI Controller	132
6.3	Operation of External Interrupt/NMI Controller	134
CHAPTER 7	REALOS-RELATED HARDWARE	141
7.1	Delayed Interrupt Module	142
7.2	Register of Delayed Interrupt Module	143
7.3	Operation of Delayed Interrupt Module	144
7.4	Bit Search Module	145
7.5	Register of Bit Search Module	146
7.6	Operation of Bit Search Module	148
CHAPTER 8	16-BIT RELOAD TIMER	151
8.1	Overview	152
8.2	16-bit Reload Timer Block Diagram	153
8.3	16-bit Reload Timer Registers	154
8.4	Operation of 16-bit Reload Timer	158
CHAPTER 9	PPG (Programmable Pulse Generator)	163
9.1	Overview	164
9.2	Block Diagram	168
9.3	Register of PPG	172
9.4	Operation Explanation	177
CHAPTER 10	PWC (Pulse Width Count: Pulse Width Measurement)	183
10.1	Overview	184
10.2	Block Diagram	185
10.3	Register of PWC	186
10.4	Operation Explanation	192
CHAPTER 11	MULTIFUNCTIONAL TIMER	203
11.1	Overview	204
11.2	Block Diagram	206
11.3	Pins of Multifunctional Timer	212
11.4	Multifunctional Timer Register	213
11.4.1	Compare Clear Buffer Register (CPCLRBH, CPCLRBL) /Compare Clear Register (CPCLRH, CPCLRL)	218
11.4.2	Timer Data Register (TCDTH, TCDTL)	220
11.4.3	Timer State Control Register (TCCSH, TCCSL)	221
11.4.4	A/D Trigger Control Register (ADTRGC)	226

11.4.5	Output Compare Buffer Register (OCCPBH0 to OCCPBH5, OCCPBL0 to OCCPBL5) / Output Compare Register (OCCPH0 to OCCPH5, OCCPL0 to OCCPL5)	227
11.4.6	Compare Control Register (OCSH0 to OCSH5, OCSL0 to OCSL5)	229
11.4.7	Compare Mode Control Register (OCMOD)	234
11.4.8	Input Capture Data Registers (IPCPH0 to IPCPH3, IPCPL0 to IPCPL3)	236
11.4.9	Input Capture State Control/PPG Output Control Register (ICSH23, ICSL23, PICSH01, PICSL01)	237
11.4.10	16-bit Dead Timer Register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2)	244
11.4.11	16-bit Dead Timer Control Register (DTCR0 to DTCR2)	245
11.4.12	Waveform Control Register (SIGCR1, SIGCR2)	251
11.4.13	A/D Activation Compare Register (ADCOMP0, ADCOMP1, ADCOMP2, ADCOMPC)	254
11.5	Multifunctional Timer Interrupt	256
11.6	Operation of Multifunctional Timer	260
11.6.1	Operation of 16-bit Free-run Timer	261
11.6.2	Operation of 16-bit Output Compare	267
11.6.3	Operation of 16-bit Input Capture	277
11.6.4	Waveform Generator Operation	279
11.6.4.1	Operation of Timer Mode	283
11.6.4.2	Operation During Dead Time Timer Mode	285
11.6.4.3	DTTI Pin Control Operation	289
11.6.5	A/D Activation Compare Operation	291
11.7	Notes on Using Multifunctional Timer	292
11.8	Program Example of Multifunctional Timer	294
CHAPTER 12	U-TIMER (16-bit Timer for UART Baud Rate Generation)	297
12.1	Overview	298
12.2	Description of Registers	299
12.3	Description of Operation	302
CHAPTER 13	UART	303
13.1	Overview	304
13.2	Detail Description of Registers	307
13.3	Operation of UART	313
13.4	Example of Using the UART	319
13.5	Example of Setting Baud Rates and U-TIMER Reload Values	321
CHAPTER 14	8/10-BIT A/D CONVERTER	323
14.1	Overview	324
14.2	Configuration	325
14.3	Pin	328
14.4	Registers	330
14.4.1	A/D Channel Control Register (ADCH)	331
14.4.2	A/D Mode Setting Register (ADMD)	333
14.4.3	A/D Control Status Register (ADCS)	336
14.4.4	A/D Data Register (ADCD)	339
14.4.5	Analog Input Control Register (AICR)	340
14.5	Interrupt	341

14.6	Operation Explanation	342
14.7	A/D Conversion Data Protection Function	346
14.8	Precautions on Using	347
CHAPTER 15	MULTIPLICATION AND ADDITION CALCULATOR	349
15.1	Overview	350
15.2	Register Description	355
15.3	Operation Explanation	359
15.4	Instruction Detail Explanation	363
CHAPTER 16	DMAC (DMA Controller)	367
16.1	Overview	368
16.2	Register Details Explanation	371
16.3	Operation Explanation	385
16.4	Setting Up Transfer Request	387
16.5	Transfer Sequence	388
16.6	Overview of DMA Transfer	390
16.7	Operation Flow	398
16.8	Data Path	400
CHAPTER 17	FLASH MEMORY	403
17.1	Overview of Flash Memory	404
17.2	Flash Memory Registers	409
17.2.1	Flash Memory Status Register (FLCR)	410
17.2.2	Flash Wait Register (FLWC)	412
17.3	Access Modes of Flash Memory	414
17.4	Starting the Flash Memory Automatic Algorithm	416
17.5	Automatic Algorithm Execution Status	420
17.6	Sector Protect Operation	426
CHAPTER 18	SERIAL PROGRAMMING CONNECTION	431
18.1	Overview	432
APPENDIX	437
APPENDIX A	I/O Map	438
APPENDIX B	Vector Table	446
APPENDIX C	Pin Status In Each CPU State	450
APPENDIX D	Notes When Little Endian Area Is Used	453
APPENDIX E	Instruction Lists	460
APPENDIX F	Precautions on Handling	475
INDEX	479

Main changes in this edition

Page	Changes (For details, refer to main body.)
i	■ Sample Program is added.
23	● Note on Operation in PLL Clock Mode is changed.
53	Table 3.8-3 Vector Table (1 / 3) is changed. (Instruction break exception → System-reserved) (Operand break trap → System-reserved)
66	● Watchdog reset is changed. (WPR (watchdog reset defer register) → CTBR (timebase counter clear register))
70	■ Selecting the Source Clock Signal is changed. (ϕ is the base clock that is generated from the source clock divided by two or by using the PLL oscillation. Therefore, the system base clock is a clock generated in the above-mentioned internal base clock generation. is added.)
76	Figure 3.11-1 Block Diagram of the Clock Generation Control Unit is changed. (The WPR register part of [Watchdog controller] is deleted.)
78	The table in [bit9, bit8] WT1, WT0 (Watchdog interval Time select) is changed. (WPR → CTBR)
78	■ STCR: Standby Control Register is changed.
83	■ CTBR: Timebase Counter Clear Register is changed. (Note, however, that the FF is cleared automatically when the CPU is not operating such as in the stop or sleep mode or during DMA transfer. If such a condition develops, therefore, a watchdog reset is deferred automatically. For details, see the section "3.11.7 Peripheral Circuits in the Clock Control Unit". is added.)
85	■ WPR: Watchdog Reset Defer Register in 3.11.6 Registers in the Clock Generation Control Unit is deleted.
90	[Deferring the generation of a watchdog reset] in ● Watchdog timer is changed. (WPR (watchdog reset defer register) → CTBR (timebase counter clear register))
96	● Sleep mode is changed.
97	● Stop mode is changed.
121	Table 5.3-1 Interrupt Sources, Interrupt Numbers, and Interrupt Levels (1 / 3) is changed. (Instruction break exception → System-reserved) (Operand break trap → System-reserved)
137	■ Precautions when Returning from STOP State Using External Interrupt is added.
138	■ Return Operation from STOP State is added.
304	■ Features of UART is changed. (● The DMAC interrupt source is cleared by the writing operation to the DRCL register. is deleted.)
305	■ Register List is changed. (DRCL part is deleted.)
312	■ DRCL is deleted.
318	■ Precautions on Usage is changed. (Write to the DRCL register before starting DMA transfer by an interrupt for the first time. is deleted.)

Page	Changes (For details, refer to main body.)
362	Notes: is added.
445	Appendix Table A-1 I/O Map is changed. (*3: These are reserved registers. These access are prohibited. is added.)
447	Appendix Table B-1 Vector Table (1 / 3) is changed. (Instruction break exception → System reserved) (Operand break trap → System reserved)
475	● Low-power consumption mode is changed.
439, 443	The sections of the following registers in Appendix Table A-1 I/O Map are changed. (DRCL0, DRCL1, DRCL2, WPR)

CHAPTER 1

OVERVIEW

This chapter provides basic information for understanding the MB91260B series as a whole, covering its features, block diagram, and functions.

- 1.1 Overview
- 1.2 Block Diagram
- 1.3 Package Dimension
- 1.4 Pin Assignment
- 1.5 Pin Description
- 1.6 I/O Circuit Types

1.1 Overview

The MB91260B series is a line of Fujitsu's general-purpose 32-bit RISC microcontrollers designed for embedded control applications which require high-speed processing. The MB91260B contains an FR60Lite CPU core compatible with the FR family of CPUs.

■ Features

● FR60Lite CPU

- 32-bit RISC, load/store architecture, five pipelines
- Maximum operating frequency: 33 MHz (oscillation frequency of 4.192 MHz multiplied by 8 (by PLL clock multiplication))
- 16-bit fixed-length instructions (basic instructions)
- Instruction execution speed: 1 instruction per cycle
- Instructions including memory-to-memory transfer, bit manipulation, and barrel shift instructions: Instructions suitable for embedded applications
- Function entry and exit instructions and register multi-load/store instructions: Instructions compatible with the C language
- Register interlock function: Facilitating assembly-language coding
- Built-in multiplier with instruction-level support
- Signed 32-bit multiplication: 5 cycles
- Signed 16-bit multiplication: 3 cycles
- Interrupts (PC/PS saving): 6 cycles (16 priority levels)
- Harvard architecture enabling simultaneous execution of both program access and data access
- Instruction-compatible with the FR family

● Built-in peripheral resources

- Built-in ROM capacity and type
 MASK ROM: 128 KB (MB91263B), 256 KB (MB91264B)
 FLASH ROM: 256 KB (MB91F264B)
- Built-in RAM capacity: 8 KB
- A/D converters (successive approximation type)
 Resolution: 10 bits: 8 channels \times 1 unit, 2 channels \times 2 units
 Conversion time: 1.2 μ s (minimum conversion time at a system clock of 33 MHz)
 1.35 μ s (minimum conversion time at a system clock of 20 MHz)
- External interrupt inputs: 10 channels
- Bit search module (used by REALOS)
 This feature searches for the first bit position causing a 1-to-0 transition in the MSB (most significant bit) of a word.

- UART (full duplex, double buffering): 3 channels
 - Choice of enabling/disabling parity
 - Choice of clock asynchronous (start-stop) or synchronous communication
 - Dedicated baud rate timer (U-TIMER) integrated for each channel
 - Capable of using an external clock as a transfer clock
 - Detection of parity, frame, and overrun errors
- 8/16-bit PPG timer: 16 channels (in 8-bit mode) / 8 channels (in 16-bit mode)
- 16-bit reload timer: 3 channels (cascade mode available, with no output of reload timer 0)
- 16-bit free-running timer: 1 channel
- 16-bit PWC timer: 2 channels
- Input capture unit: 4 channels (interlocking with the free-running timer)
- Output compare unit: 6 channels (interlocking with the free-running timer)
- Waveform generator
 - Capable of generating various waveforms using the output compare unit's output, 16-bit PPG timer 0, and 16-bit dead timer.
- Multiplier-accumulator
 - RAM: Instruction RAM 256×16 bits
 - XRAM 64×16 bits
 - YRAM 64×16 bits
 - Executes a multiply-accumulate operation ($16 \text{ bits} \times 16 \text{ bits} + 40 \text{ bits}$) in one instruction cycle.
 - Extracts the operation result by rounding from 40 bits to 16 bits.
- DMAC (DMA Controller): 5 channels
 - Capable of starting transfer of built-in peripheral interrupts by means of software
- Watchdog timer
- Low-power consumption modes
 - Sleep and stop modes

● Others

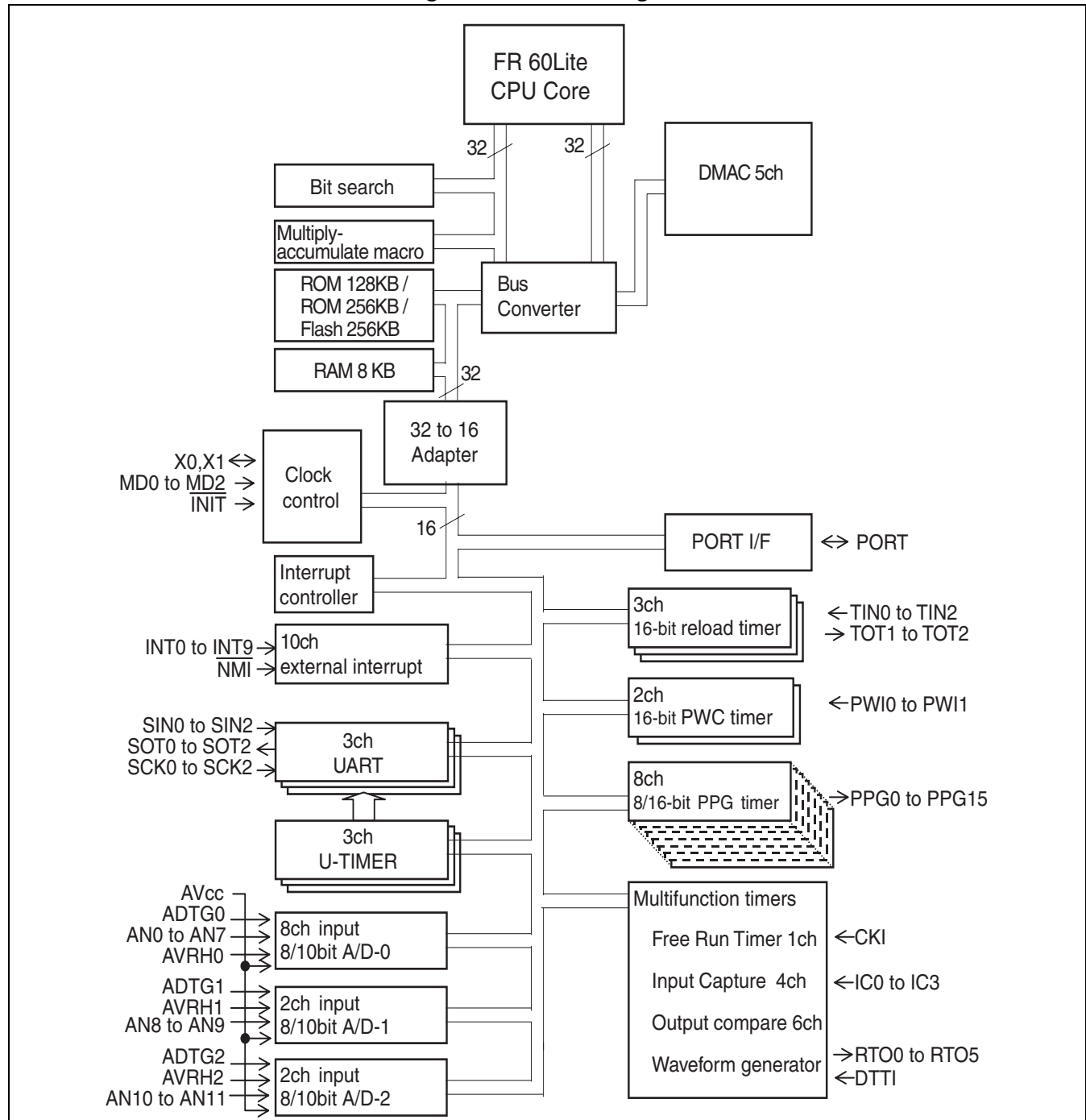
- Package: QFP-100, LQFP-100
- CMOS 0.35 μm technology
- Single power supply [$V_{cc} = 4.0 \text{ V}$ to 5.5 V]

1.2 Block Diagram

This section shows a block diagram of the MB91260B series.

■ Block Diagram

Figure 1.2-1 Block Diagram

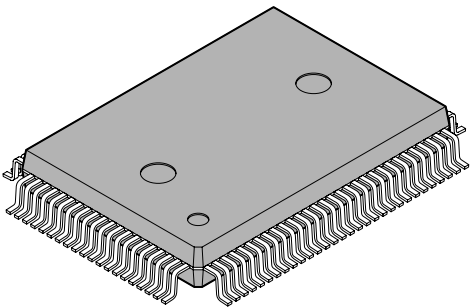


1.3 Package Dimension

This section shows the package dimension of FPT-100P-M06 and FPT-100P-M05.

■ Package Dimension

Figure 1.3-1 FPT-100P-M06 Package Dimensions

<div>100-pin plastic QFP</div>  <div>(FPT-100P-M06)</div>	Lead pitch	0.65 mm
	Package width × package length	14.00 × 20.00 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	3.35 mm MAX
	Code (Reference)	P-QFP100-14×20-0.65

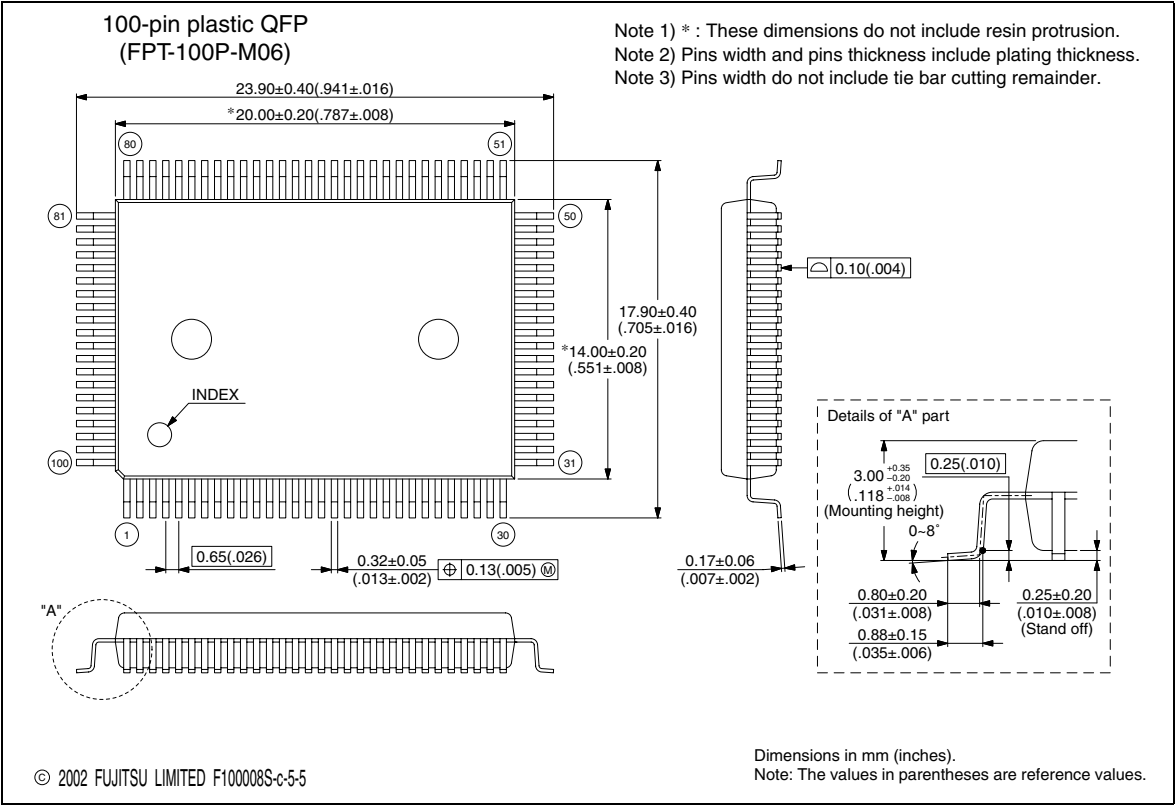
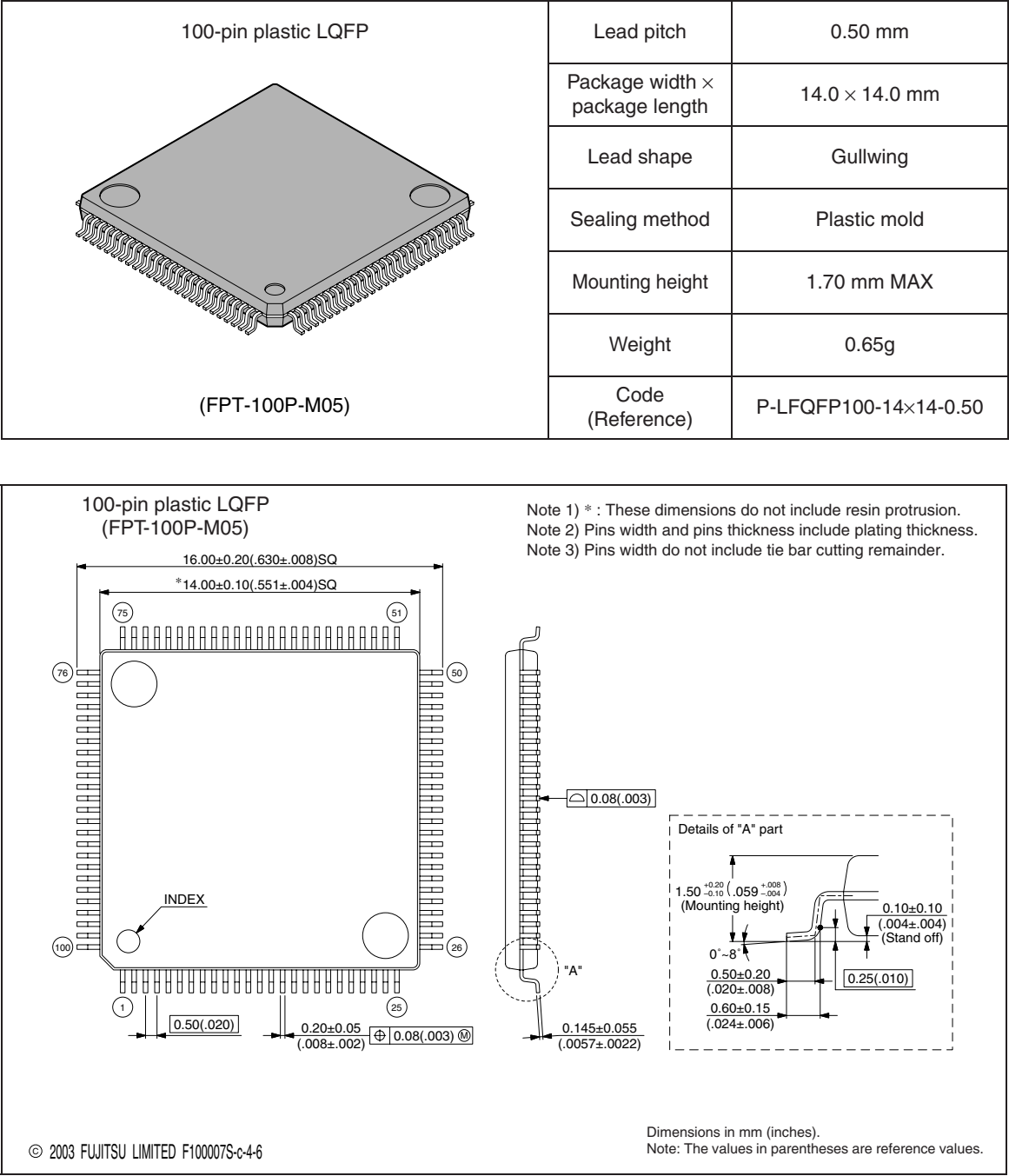


Figure 1.3-2 FPT-100P-M05 Package Dimensions



1.4 Pin Assignment

This section shows the pin assignment of the MB91260B series.

■ Pin Assignment

Figure 1.4-1 QFP-100

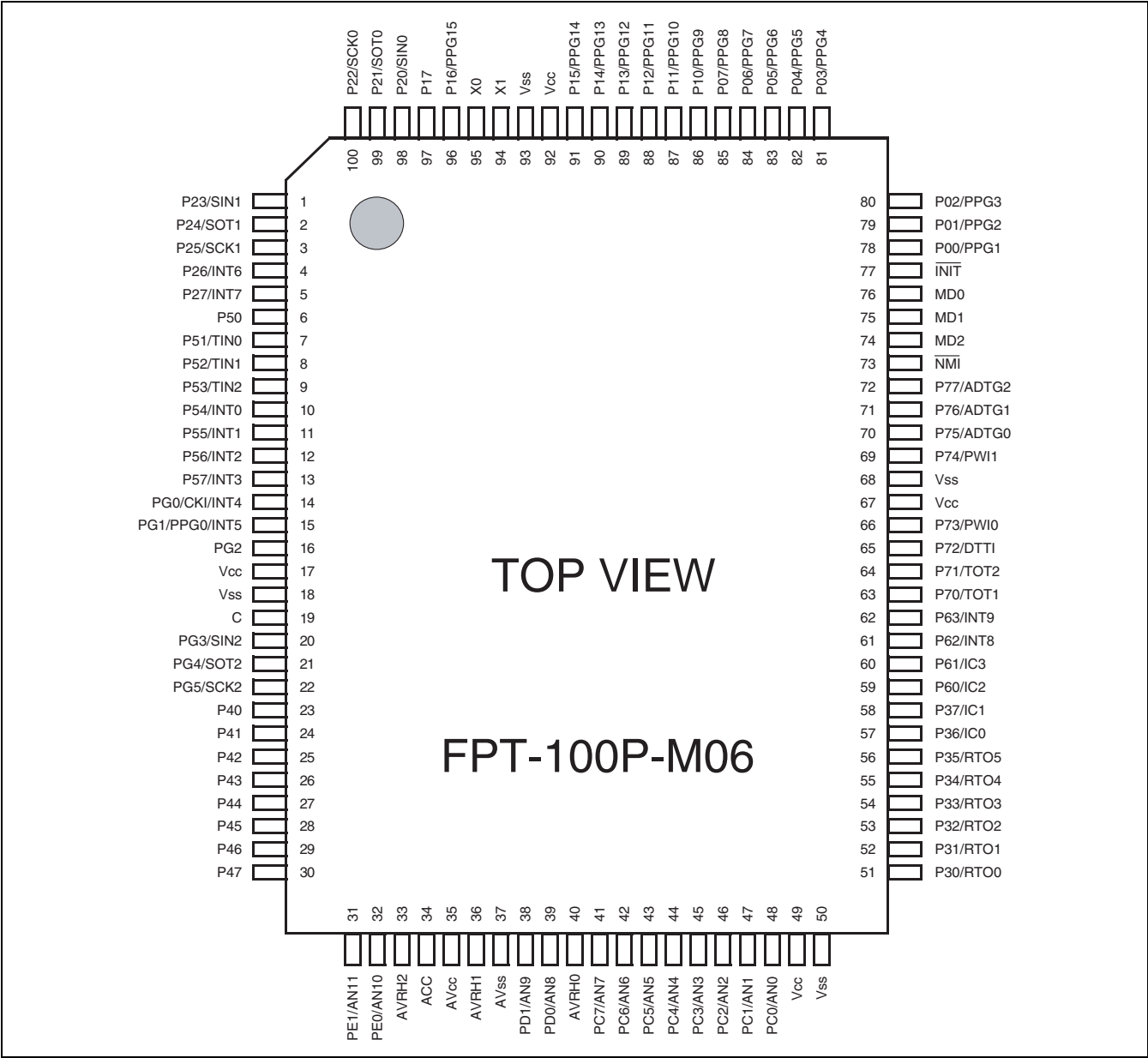
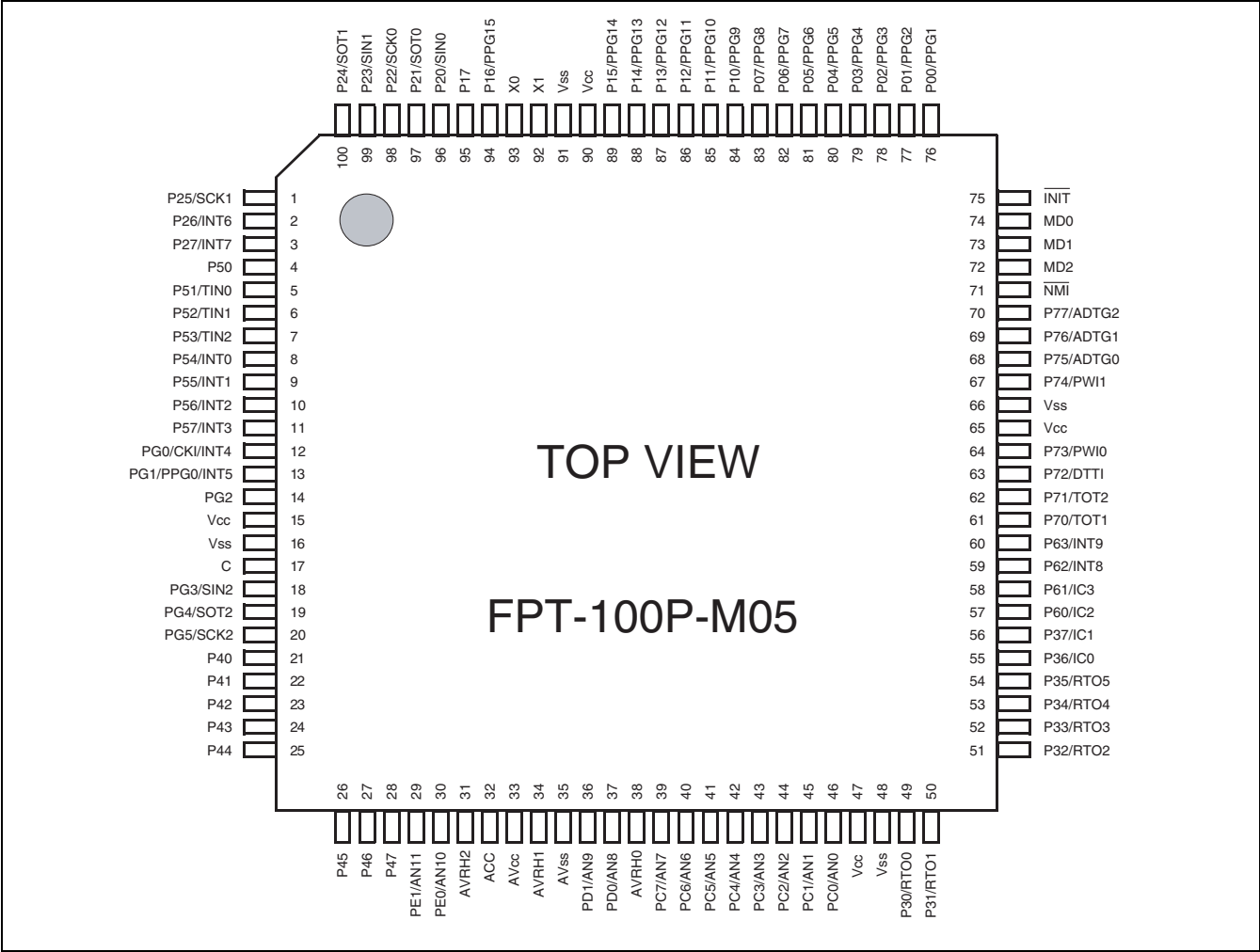


Figure 1.4-2 LQFP-100



1.5 Pin Description

This section shows the pin description of the MB91260B series.

■ Pin Description

Table 1.5-1 Pin Description (1 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
1	99	SIN1	D	UART1 data input. Since this input is used as required when UART1 is performing input operation, the port output must remain off unless used intentionally.
		P23		General-purpose I/O port. This port is enabled when UART1 data input is disabled.
2	100	SOT1	D	UART1 data output. This function is enabled when UART1 data output is enabled.
		P24		General-purpose I/O port. This function is enabled when UART1 data output is disabled.
3	1	SCK1	D	UART1 clock input/output. This function is enabled when UART1 clock output is enabled.
		P25		General-purpose I/O port. This function is enabled when UART1 clock output is disabled.
4	2	INT6	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P26		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
5	3	INT7	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P27		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
6	4	P50	C	General-purpose I/O port. This port is enabled in single-chip mode.
7	5	TIN0	C	Reload timer 0 external trigger input. Since this input is used as required when trigger input is enabled, the port output must remain off unless used intentionally.
		P51		General-purpose I/O port. This function is enabled when reload timer 0 external clock input is disabled.
8	6	TIN1	C	Reload timer 1 external trigger input. Since this input is used as required when trigger input is enabled, the port output must remain off unless used intentionally.
		P52		General-purpose I/O port. This function is enabled when reload timer 1 external clock input is disabled.

Table 1.5-1 Pin Description (2 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
9	7	TIN2	C	Reload timer 2 external trigger input. Since this input is used as required when trigger input is enabled, the port output must remain off unless used intentionally.
		P53		General-purpose I/O port. This function is enabled when reload timer 2 external clock input is disabled.
10	8	INT0	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P54		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
11	9	INT1	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P55		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
12	10	INT2	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P56		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
13	11	INT3	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P57		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
14	12	CKI	E	Free-running timer external clock input pin. Since this input is used as required when selected as the external clock input for the free-running timer, the port output must remain off unless used intentionally.
		INT4		External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		PG0		General-purpose I/O port. This port is enabled when free-running timer external clock input and external interrupt input are disabled.
15	13	PPG0	E	PPG timer 0 output. This function is enabled when PPG timer 0 output is enabled.
		INT5		External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		PG1		General-purpose I/O port. This port is enabled when PPG timer 0 output and external interrupt input are disabled.

Table 1.5-1 Pin Description (3 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
16	14	PG2	C	General-purpose I/O port.
20	18	SIN2	D	UART2 data input. Since this input is used as required when UART2 is performing input operation, the port output must remain off unless used intentionally.
		PG3		General-purpose I/O port. This port is enabled when UART2 data input is disabled.
21	19	SOT2	D	UART2 data output. This function is enabled when UART2 data output is enabled.
		PG4		General-purpose I/O port. This port is enabled when UART2 data output is disabled.
22	20	SCK2	D	UART2 clock input/output. This function is enabled when UART2 clock output is enabled.
		PG5		General-purpose I/O port. This function is enabled when UART2 clock output is disabled.
23	21	P40	C	General-purpose I/O port.
24	22	P41	C	General-purpose I/O port.
25	23	P42	C	General-purpose I/O port.
26	24	P43	C	General-purpose I/O port.
27	25	P44	C	General-purpose I/O port.
28	26	P45	C	General-purpose I/O port.
29	27	P46	C	General-purpose I/O port.
30	28	P47	C	General-purpose I/O port.
31	29	AN11	G	A/D converter analog input. This function is enabled when the AICR2 register specifies analog input.
		PE1		General-purpose I/O port. This function is enabled when analog input is disabled.
32	30	AN10	G	A/D converter analog input. This function is enabled when the AICR2 register specifies analog input.
		PE0		General-purpose I/O port. This function is enabled when analog input is disabled.
38	36	AN9	G	A/D converter analog input. This function is enabled when the AICR1 register specifies analog input.
		PD1		General-purpose I/O port. This function is enabled when analog input is disabled.
39	37	AN8	G	A/D converter analog input. This function is enabled when the AICR1 register specifies analog input.
		PD0		General-purpose I/O port. This function is enabled when analog input is disabled.
41	39	AN7	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC7		General-purpose I/O port. This function is enabled when analog input is disabled.

Table 1.5-1 Pin Description (4 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
42	40	AN6	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC6		General-purpose I/O port. This function is enabled when analog input is disabled.
43	41	AN5	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC5		General-purpose I/O port. This function is enabled when analog input is disabled.
44	42	AN4	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC4		General-purpose I/O port. This function is enabled when analog input is disabled.
45	43	AN3	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC3		General-purpose I/O port. This function is enabled when analog input is disabled.
46	44	AN2	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC2		General-purpose I/O port. This function is enabled when analog input is disabled.
47	45	AN1	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC1		General-purpose I/O port. This function is enabled when analog input is disabled.
48	46	AN0	G	A/D converter analog input. This function is enabled when the AICR0 register specifies analog input.
		PC0		General-purpose I/O port. This function is enabled when analog input is disabled.
51	49	RTO0	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P30		General-purpose I/O port. This function is enabled when waveform generator output is disabled.
52	50	RTO1	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P31		General-purpose I/O port. This function is enabled when waveform generator output is disabled.
53	51	RTO2	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P32		General-purpose I/O port. This function is enabled when waveform generator output is disabled.

Table 1.5-1 Pin Description (5 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
54	52	RTO3	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P33		General-purpose I/O port. This function is enabled when waveform generator output is disabled.
55	53	RTO4	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P34		General-purpose I/O port. This function is enabled when waveform generator output is disabled.
56	54	RTO5	J	Multifunction timer waveform generator output. This pin outputs a specified waveform to the waveform generator. The waveform output is enabled when waveform generator output is enabled.
		P35		General-purpose I/O port. This function is enabled when waveform generator output is disabled.
57	55	IC0	D	Input capture 0 trigger input. The trigger can be input when the input capture trigger input and input port are set. Since this input is used as required when selected as the input capture input, the port output must remain off unless used intentionally.
		P36		General-purpose I/O port. This function is enabled when input capture trigger input is disabled.
58	56	IC1	D	Input capture 1 trigger input. The trigger can be input when the input capture trigger input and input port are set. Since this input is used as required when selected as the input capture input, the port output must remain off unless used intentionally.
		P37		General-purpose I/O port. This function is enabled when input capture trigger input is disabled.
59	57	IC2	D	Input capture 2 trigger input. The trigger can be input when the input capture trigger input and input port are set. Since this input is used as required when selected as the input capture input, the port output must remain off unless used intentionally.
		P60		General-purpose I/O port. This function is enabled when input capture trigger input is disabled.
60	58	IC3	D	Input capture 3 trigger input. The trigger can be input when the input capture trigger input and input port are set. Since this input is used as required when selected as the input capture input, the port output must remain off unless used intentionally.
		P61		General-purpose I/O port. This function is enabled when input capture trigger input is disabled.

Table 1.5-1 Pin Description (6 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
61	59	INT8	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P62		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
62	60	INT9	E	External interrupt input. Since this input is used as required when the corresponding external interrupt is enabled, the port output must remain off unless used intentionally.
		P63		General-purpose I/O port. This function is enabled when external interrupt input is disabled.
63	61	TOT1	C	Reload timer 1 output. This function is enabled when reload timer output is enabled.
		P70		General-purpose I/O port. This function is enabled when reload timer output is disabled.
64	62	TOT2	C	Reload timer 2 output. This function is enabled when reload timer output is enabled.
		P71		General-purpose I/O port. This function is enabled when reload timer output is disabled.
65	63	DTTI	D	Input signal for controlling multifunction timer waveform generator outputs RTO0 to RTO5. This function is enabled when DTTI input is enabled.
		P72		General-purpose I/O port. This function is enabled when DTTI input is disabled.
66	64	PWI0	D	PWC timer 0 pulse width counter input. This function is enabled when PWC timer 0 pulse width counter input is enabled.
		P73		General-purpose I/O port. This function is enabled when PWC timer 0 pulse width counter input is disabled.
69	67	PWI1	D	PWC timer 1 pulse width counter input. This function is enabled when PWC timer 1 pulse width counter input is enabled.
		P74		General-purpose I/O port. This function is enabled when PWC timer 1 pulse width counter input is disabled.
70	68	ADTG0	C	A/D converter 0 external trigger input. Since this input is used as required when selected as the A/D converter trigger source, the port output must remain off unless used intentionally.
		P75		General-purpose I/O port. This function is enabled when A/D converter 0 external trigger input is disabled.
71	69	ADTG1	C	A/D converter 1 external trigger input. Since this input is used as required when selected as the A/D converter trigger source, the port output must remain off unless used intentionally.
		P76		General-purpose I/O port. This function is enabled when A/D converter 1 external trigger input is disabled.

Table 1.5-1 Pin Description (7 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
72	70	ADTG2	C	A/D converter 2 external trigger input. Since this input is used as required when selected as the A/D converter trigger source, the port output must remain off unless used intentionally.
		P77		General-purpose I/O port. This function is enabled when A/D converter 2 external trigger input is disabled.
73	71	$\overline{\text{NMI}}$	H	NMI (Non Maskable Interrupt) input.
74	72	MD2	K	Mode pin 2. The setting of this pin determines the basic operation mode. Connect the pin to Vcc or Vss.
75	73	MD1	K	Mode pin 1. The setting of this pin determines the basic operation mode. Connect the pin to Vcc or Vss.
76	74	MD0	K	Mode pin 0. The setting of this pin determines the basic operation mode. Connect the pin to Vcc or Vss.
77	75	$\overline{\text{INIT}}$	I	External reset input.
78	76	PPG1	C	PPG timer 1 output. This function is enabled when PPG timer 1 output is enabled.
		P00		General-purpose I/O port. This function is enabled when PPG timer 1 output is disabled.
79	77	PPG2	C	PPG timer 2 output. This function is enabled when PPG timer 2 output is enabled.
		P01		General-purpose I/O port. This function is enabled when PPG timer 2 output is disabled.
80	78	PPG3	C	PPG timer 3 output. This function is enabled when PPG timer 3 output is enabled.
		P02		General-purpose I/O port. This function is enabled when PPG timer 3 output is disabled.
81	79	PPG4	C	PPG timer 4 output. This function is enabled when PPG timer 4 output is enabled.
		P03		General-purpose I/O port. This function is enabled when PPG timer 4 output is disabled.
82	80	PPG5	C	PPG timer 5 output. This function is enabled when PPG timer 5 output is enabled.
		P04		General-purpose I/O port. This function is enabled when PPG timer 5 output is disabled.
83	81	PPG6	C	PPG timer 6 output. This function is enabled when PPG timer 6 output is enabled.
		P05		General-purpose I/O port. This function is enabled when PPG timer 6 output is disabled.
84	82	PPG7	C	PPG timer 7 output. This function is enabled when PPG timer 7 output is enabled.
		P06		General-purpose I/O port. This function is enabled when PPG timer 7 output is disabled.

Table 1.5-1 Pin Description (8 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
85	83	PPG8	C	PPG timer 8 output. This function is enabled when PPG timer 8 output is enabled.
		P07		General-purpose I/O port. This function is enabled when PPG timer 8 output is disabled.
86	84	PPG9	C	PPG timer 9 output. This function is enabled when PPG timer 9 output is enabled.
		P10		General-purpose I/O port. This function is enabled when PPG timer 9 output is disabled.
87	85	PPG10	C	PPG timer 10 output. This function is enabled when PPG timer 10 output is enabled.
		P11		General-purpose I/O port. This function is enabled when PPG timer 10 output is disabled.
88	86	PPG11	C	PPG timer 11 output. This function is enabled when PPG timer 11 output is enabled.
		P12		General-purpose I/O port. This function is enabled when PPG timer 11 output is disabled.
89	87	PPG12	C	PPG timer 12 output. This function is enabled when PPG timer 12 output is enabled.
		P13		General-purpose I/O port. This function is enabled when PPG timer 12 output is disabled.
90	88	PPG13	C	PPG timer 13 output. This function is enabled when PPG timer 13 output is enabled.
		P14		General-purpose I/O port. This function is enabled when PPG timer 13 output is disabled.
91	89	PPG14	C	PPG timer 14 output. This function is enabled when PPG timer 14 output is enabled.
		P15		General-purpose I/O port. This function is enabled when PPG timer 14 output is disabled.
94	92	X1	A	Clock (oscillation) output.
95	93	X0	A	Clock (oscillation) input.
96	94	PPG15	C	PPG timer 15 output. This function is enabled when PPG timer 15 output is enabled.
		P16		General-purpose I/O port. This function is enabled when PPG timer 15 output is disabled.
97	95	P17	C	General-purpose I/O port.
98	96	SIN0	D	UART0 data input. Since this input is used as required when UART0 is performing input operation, the port output must remain off unless used intentionally.
		P20		General-purpose I/O port. This port is enabled when UART0 data input is disabled.
99	97	SOT0	D	UART0 data output. This function is enabled when UART0 data output is enabled.
		P21		General-purpose I/O port. This port is enabled when UART0 data output is disabled.

Table 1.5-1 Pin Description (9 / 9)

Pin No.		Pin Name	I/O Circuit Type	Function
QFP	LQFP			
100	98	SCK0	D	UART0 clock input/output. This function is enabled when UART0 clock output is enabled.
		P22		General-purpose I/O port. This function is enabled when UART0 clock output is disabled.

[Power-supply and ground pins]

Table 1.5-2 Pin Description

Pin No.		Pin Name	Function
QFP	LQFP		
18, 50, 68, 93	16, 48, 66, 91	Vss	GND pins. Use all of these pins at equal potential.
17, 49, 67, 92	15, 47, 65, 90	Vcc	Power-supply pins. Use all of these pins at equal potential.
35	33	AVcc	Analog power-supply pin for A/D converter
33	31	AVRH2	Analog reference power-supply pin for A/D converter 2
36	34	AVRH1	Analog reference power-supply pin for A/D converter 1
40	38	AVRH0	Analog reference power-supply pin for A/D converter 0
37	35	AVss	Analog GND pin for A/D converter
19	17	C	Capacitor coupling pin for internal regulator
34	32	ACC	Analog capacitor coupling pin

1.6 I/O Circuit Types

This section shows I/O circuit types of the MB91260B series.

■ Input/Output Circuit Types

Table 1.6-1 Input/Output Circuit Type (1 / 3)

Classification	Circuit Type	Remarks
A		Oscillation feedback resistor for high-speed operation (at main clock oscillation frequency): About 1 MΩ
C		<ul style="list-style-type: none">• CMOS level output• CMOS level input• Standby control• Control with pull-up resistor Pull-up resistance = About 50 kΩ (Typ)• I_{OL} = 4 mA
D		<ul style="list-style-type: none">• CMOS level output• CMOS level hysteresis input• Standby control• Control with pull-up resistor Pull-up resistance = About 50 kΩ (Typ)• I_{OL} = 4 mA

Table 1.6-1 Input/Output Circuit Type (2 / 3)

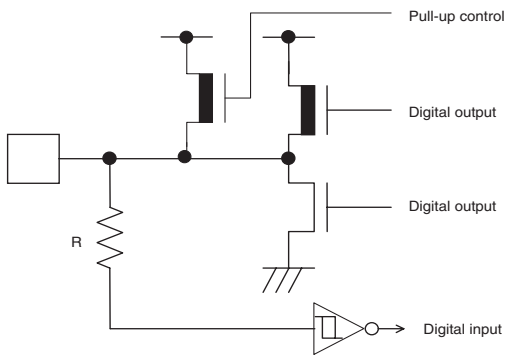
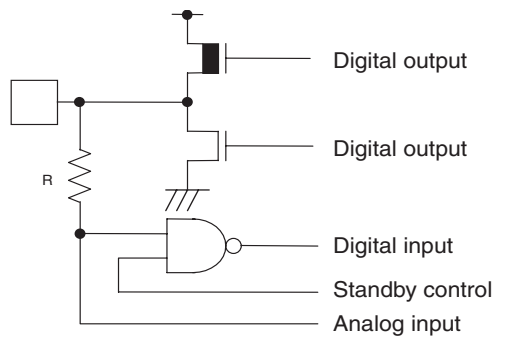
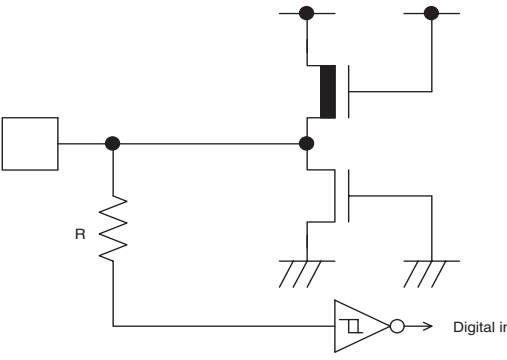
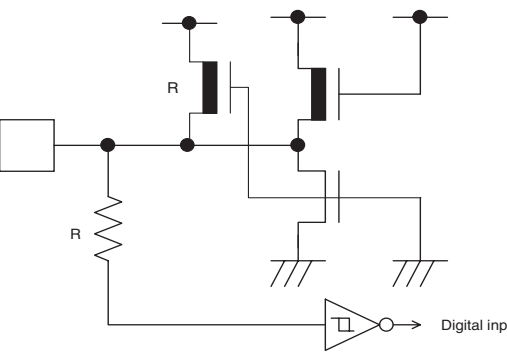
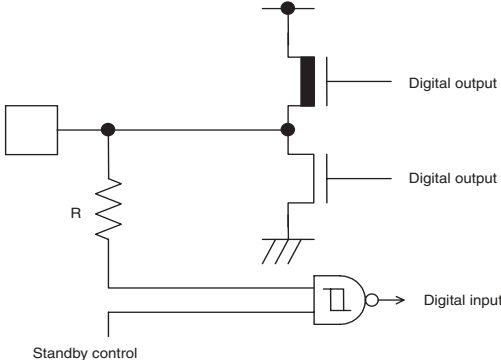
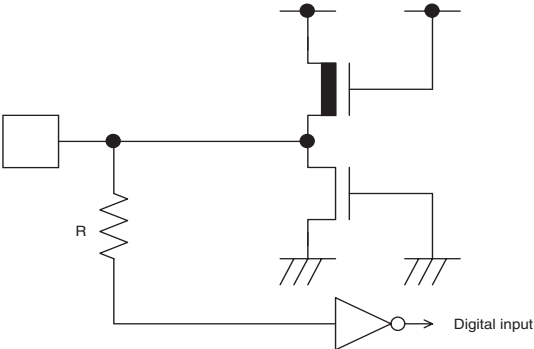
Classification	Circuit Type	Remarks
E		<ul style="list-style-type: none"> • CMOS level output • CMOS level hysteresis input • No standby control • Control with pull-up resistor Pull-up resistance = About 50 kΩ (Typ) • $I_{OL} = 4 \text{ mA}$
G		<ul style="list-style-type: none"> • Analog/CMOS level I/O pins <ul style="list-style-type: none"> - CMOS level output - CMOS level input (With standby control) - Analog input (The analog input is enabled with the AICR's corresponding bit set to "1".) • $I_{OL} = 4 \text{ mA}$
H		<ul style="list-style-type: none"> • CMOS level hysteresis input • No standby control
I		<ul style="list-style-type: none"> • CMOS level hysteresis input • With pull-up resistor Pull-up resistance = About 50 kΩ (Typ) • No standby control

Table 1.6-1 Input/Output Circuit Type (3 / 3)

Classification	Circuit Type	Remarks
J		<ul style="list-style-type: none">• CMOS level output• CMOS level hysteresis input• Standby control• $I_{OL} = 12\text{ mA}$
K		<ul style="list-style-type: none">• CMOS level input• No standby control

CHAPTER 2

HANDLING DEVICES

This chapter provides precautions on caution of using devices the MB91260B series.

2.1 Handling Devices

2.1 Handling Devices

This section explains precautions on CAUTION OF USING DEVICES the MB91260B series including prevention of latch-up, treatment of pins and so on.

■ Handling Devices

● Prevention of Latch-up

Latch-up may occur in a CMOS IC if a voltage greater than V_{cc} or less than V_{ss} is applied to an input or output pin or if a voltage exceeding the rating is applied between V_{cc} and V_{ss} . If latch-up occurs, the supply current increases rapidly, sometimes resulting in thermal breakdown of the device. Therefore, use meticulous care not to let any voltage exceed the maximum rating.

● Treatment of Unused Input Pins

Leaving unused input pins open may cause a malfunction. These pins must therefore be connected to a pull-up or pull-down resistor.

● Treatment of Power-Supply Pins

When provided with multiple V_{cc} pins or V_{ss} pins, the device is designed such that the pins to have equal potential are interconnected internally to prevent malfunctions such as latch-up. All of these pins must however be connected to the equal-potential power supply and ground externally to reduce unwanted radiation, prevent the strobe signal from malfunctioning due to a rise of ground level, and to follow the standards of total output current. In addition, the power pins should be connected to V_{cc} and V_{ss} of this device at the lowest possible impedance from the current supply source.

It is also advisable to connect a ceramic capacitor of approximately 0.1 μF , as a bypass capacitor, between V_{cc} and V_{ss} near this device.

● Crystal Oscillator Circuit

Noise in the vicinity of the X0 and X1 pins can cause this device to malfunction. When designing a printed circuit board that uses the device, therefore, place the X0/X1 pins, crystal (or ceramic) oscillator, and the bypass capacitor leading to the ground as close to one another as possible. It is also strongly recommended to design the artwork for the PC board such that the X0 and X1 pins are surrounded by a ground plane as stable operation can be expected with such a layout.

● Mode Pins (MD0 to MD2)

Connect the mode pins (MD0 to MD2) directly to V_{cc} or V_{ss} . To prevent the device from entering test mode accidentally due to noise, minimize the lengths of the patterns between the individual mode pins and V_{cc} or V_{ss} on the PC board as possible and connect them with as low an impedance as possible.

● Turning the Power On

Immediately after turning the power on, be sure to a setting initialization reset (INIT) using the $\overline{\text{INIT}}$ pin. Immediately after that, also, hold the low-level input to the $\overline{\text{INIT}}$ pin for the stabilization wait time required for the oscillator circuit to take the oscillation stabilization wait time for the oscillator circuit and the stabilization wait time for the regulator. (For INIT via the $\overline{\text{INIT}}$ pin, the oscillation stabilization wait time is initialized to the minimum value.)

● Oscillation Input in Power-on Sequence

When turning the power on, be sure to maintain clock input until the device is released from the oscillation stabilization wait state.

● Note on Operation in PLL Clock Mode

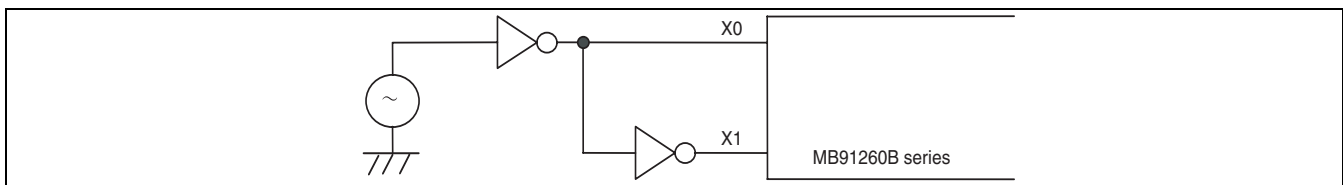
On this microcontroller, if in case the crystal oscillator breaks off or an external reference clock input stops while the PLL clock mode is selected, a self-oscillator circuit contained in the PLL may continue its operation at its self-running frequency. However, Fujitsu will not guarantee results of operation if such failure occurs.

● Note on Using an External Clock

When an external clock is used, in principle, supply the X0 pin with a clock signal and the X1 pin with a clock signal opposite in phase to the signal to the X0 pin simultaneously. To use the STOP mode (oscillation stop mode) along with the external clock, in which the X1 pin is disabled with "H" output, you should insert an external resistor of about $1k\Omega$ to prevent a collision between outputs.

Given below is an example of using an external clock.

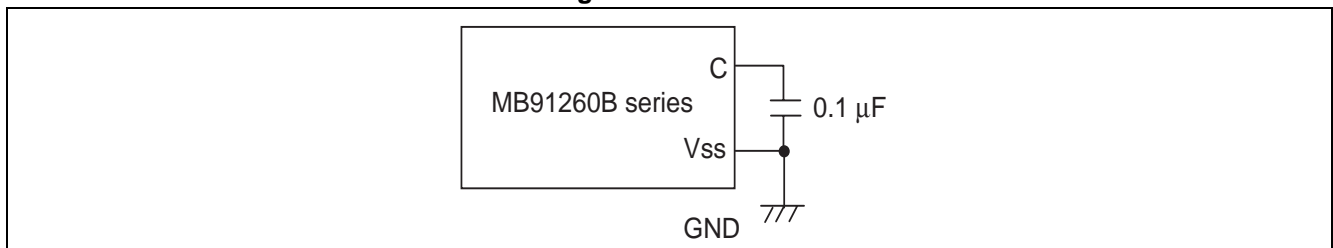
Figure 2.1-1 Example of Using an External Clock



● C Pin

As the MB91260B series contains a regulator, be sure to connect a bus capacitor of about $0.1\ \mu\text{F}$ for the regulator to the C pin.

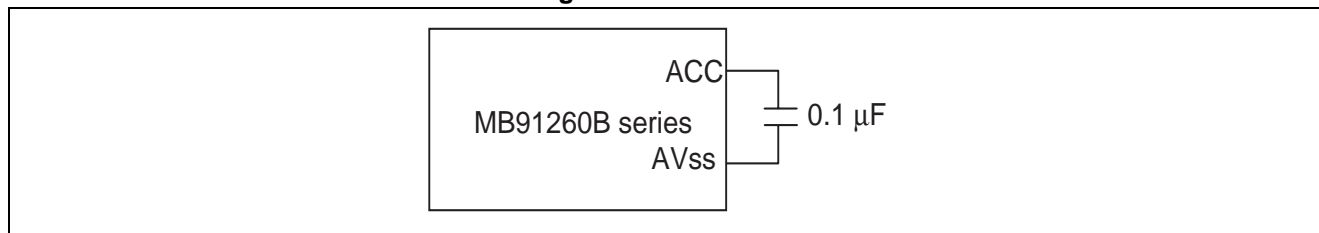
Figure 2.1-2 C Pin



● ACC Pin

As the MB91260B series contains an A/D converter, be sure to insert a capacitor of about 0.1 μF between the ACC and AVss pins.

Figure 2.1-3 ACC Pin



CHAPTER 3

CPU AND CONTROL UNITS

This chapter describes the basics of the architecture, specifications, and instructions of the MB91260B series of CPU cores to introduce their features.

- 3.1 Memory Space
- 3.2 Internal Architecture
- 3.3 Programming Model
- 3.4 Data Structure
- 3.5 Word Alignment
- 3.6 Memory Map
- 3.7 Branch Instructions
- 3.8 EIT (Exception/Interrupt/Trap)
- 3.9 Operation Modes
- 3.10 Reset (Device Initialization)
- 3.11 Clock Generation Control
- 3.12 Device Status Control

3.1 Memory Space

The MB91260B series has 4 Gbytes of logical address space (2^{32} addresses) which can be linearly accessed by the CPU.

■ Direct Addressing Area

Each of the following areas in the address space is used for input/output.
This area is referred to as the direct addressing area which allows an instruction to directly specify the address of a location in that area as the instruction's operand.
The direct addressing area varies as shown below depending on the size of data accessed:

- Byte data access : 000_H to 0FF_H
- Halfword data access : 000_H to 1FF_H
- Word data access : 000_H to 3FF_H

■ Memory Map

Figure 3.1-1 Memory Map

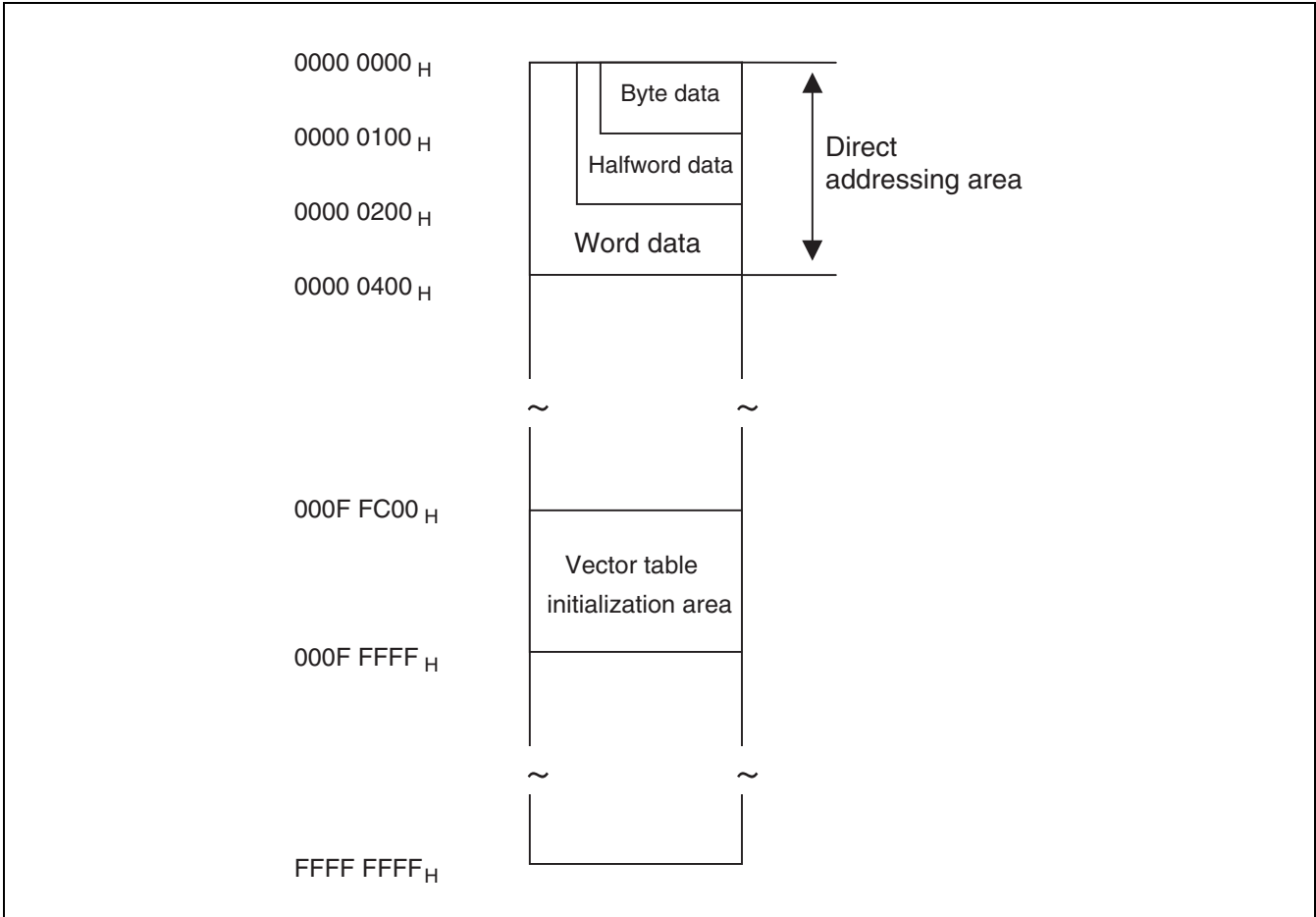
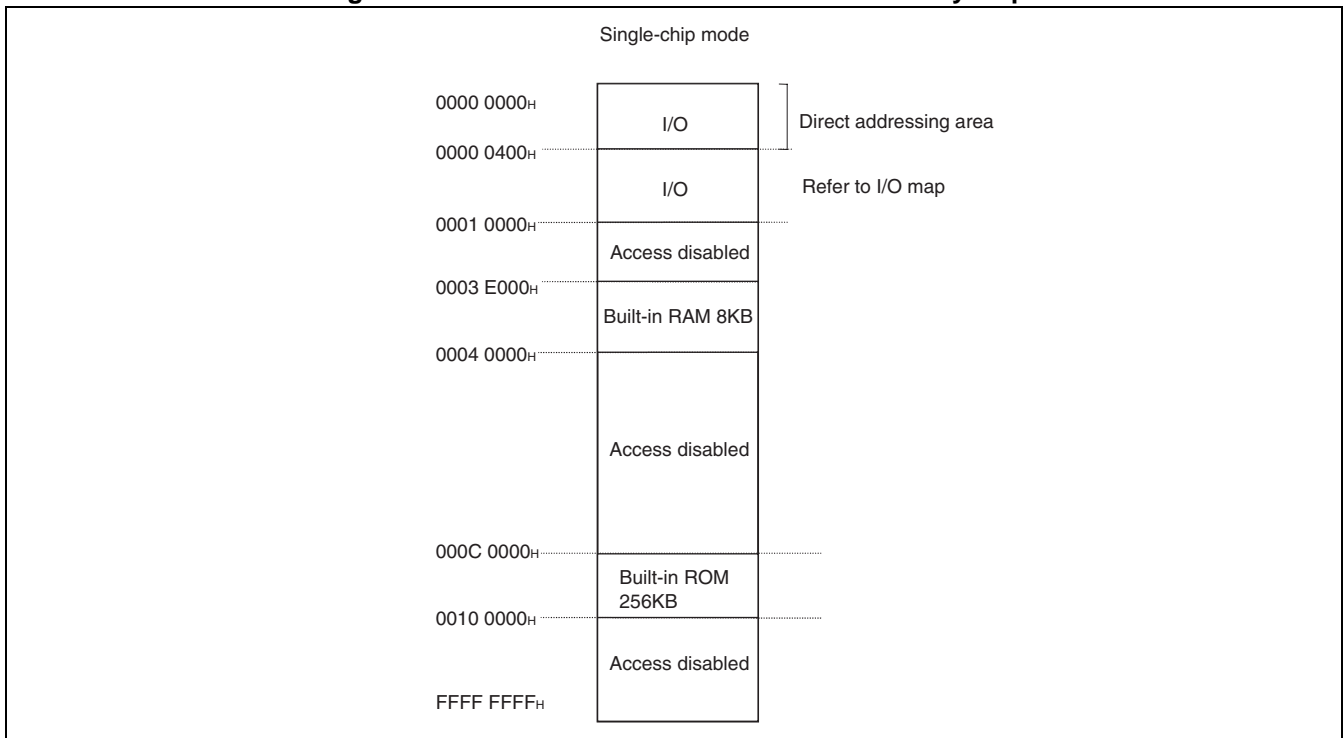
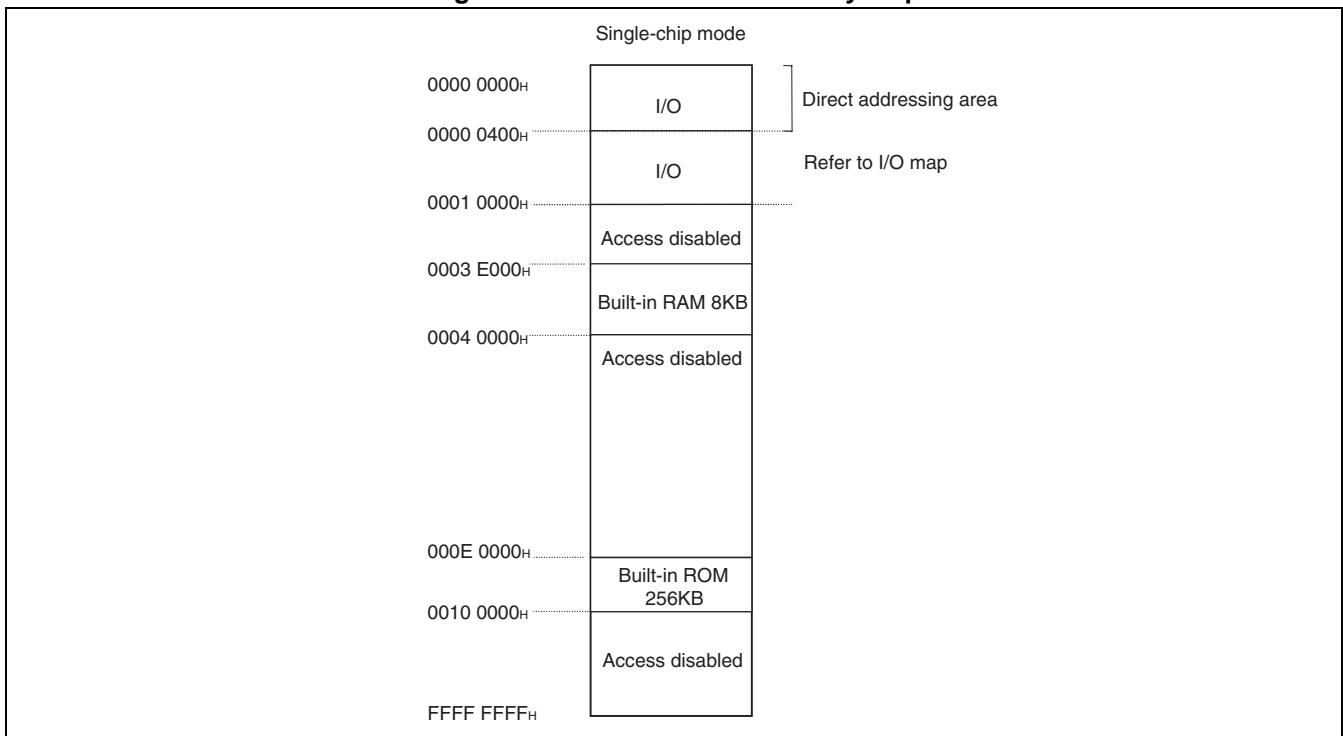


Figure 3.1-2 MB91F264B and MB91264B's Memory Map**Figure 3.1-3 MB91263B's Memory Map**

3.2 Internal Architecture

The FR60Lite CPU is a high performance core based on the RISC architecture while incorporating high-level function instructions for embedded applications.

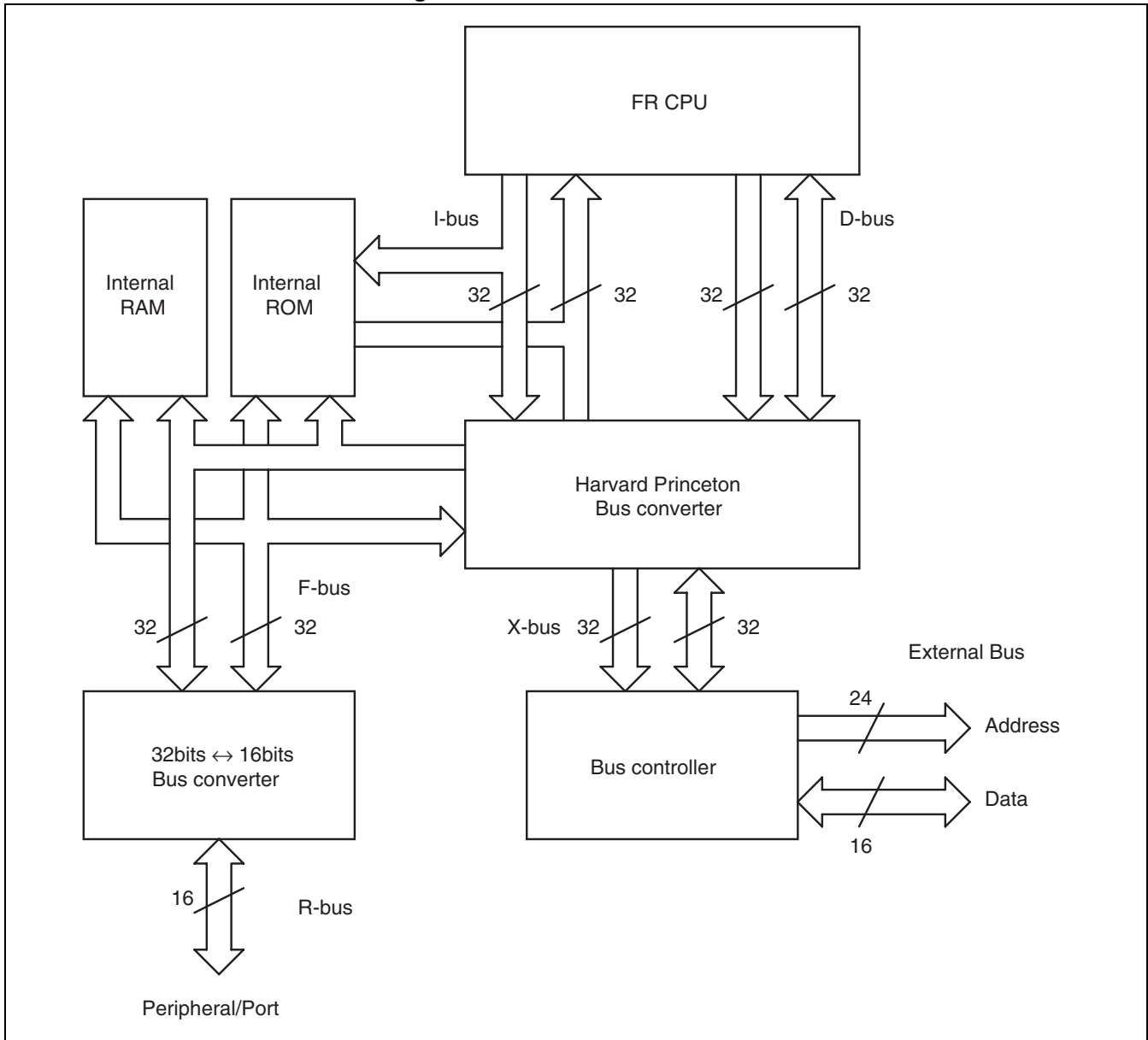
■ Features

- RISC architecture
Basic instructions: 1 instruction per cycle
- 32-bit architecture
32-bit general-purpose register \times 16
- 4 Gbytes of linear memory space
- Built-in multiplier
32-bit \times 32-bit multiplication: 5 cycles
16-bit \times 16-bit multiplication: 3 cycles
- Enhanced interrupt processing function
Quick response speed (6 cycles)
Multiple-interrupt support
Level mask function (16 levels)
- Enhanced instructions for I/O operation
Memory-to-memory transfer instruction
Bit manipulation instruction
- High coding efficiency
Basic instruction word length: 16 bits
- Low-power consumption
Sleep mode and stop mode supported
- Clock frequency divide ratio setting function

■ Internal Architecture

The FR60Lite CPU employs the Harvard architecture in which the instruction and data buses are independent of each other.

The 32-bit \leftrightarrow 16-bit bus converter is connected to a 32-bit bus (F-bus) to provide the interface between the CPU and peripheral resources. The Harvard \leftrightarrow Princeton bus converter is connected to both of the I-bus and D-bus to provide the interface between the CPU and the bus controller.

Figure 3.2-1 Internal Architecture

Note:

No external bus feature is supported.

■ CPU

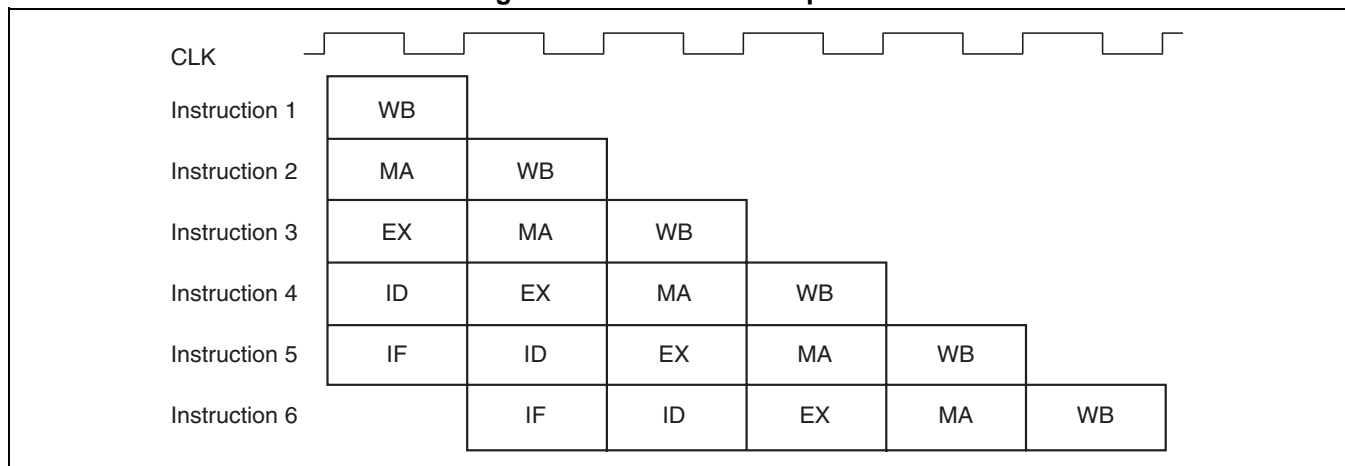
The CPU is a compact implementation of a 32-bit RISC based FR architecture.

The CPU employs a five-stage instruction pipeline to execute one instruction in one cycle.

The pipeline consists of the following stages:

- Instruction fetch (IF) Outputs the instruction address to fetch the instruction.
- Instruction decode (ID) Decodes the fetched instruction.
Also, reads a register.
- Execution (EX) Executes the operation.
- Memory access (MA)..... Access memory for loading or storing.
- Write back (WB) Writes the operation result (or loaded memory data) to the register.

Figure 3.2-2 Instruction Pipeline



Instructions are not executed out of order. That is, if instruction A enters the pipeline ahead of instruction B, instruction A always reaches the writeback stage before instruction B.

In principle, instructions are executed at a rate of one instruction per cycle. Note, however, that a load/store instruction involving a memory wait state, a branch instruction with no delay slot, or a multi-cycle instruction requires more than one cycle for execution. Also, a delay in supplying an instruction slows down the execution speed of the instruction.

■ 32-bit↔16-bit Bus Converter

This bus converter provides the interface between the 32-bit F-bus for fast access and the 16-bit R-bus, enabling data access from the CPU to built-in peripheral resources.

When the CPU attempts to make a 32-bit access, this bus converter converts it to two 16-bit accesses to the R-bus. Some of the built-in peripheral circuits have an access width restriction.

■ Harvard↔Princeton Bus Converter

This bus converter delivers consistency between CPU instruction access and data access to provide a smooth interface to the external bus.

The CPU has the Harvard architecture in which the instruction and data buses are independent of each other. In contrast, the bus controller for controlling the external bus has the Princeton architecture with a single bus. This bus converter assigns priorities to CPU instruction and data accesses to control access to the bus controller. This always optimizes the order of accesses to the external bus.

■ Instruction Overview

The FR family supports a set of general RISC instructions and the logical operation, bit manipulation, and direct addressing instructions optimized for embedded controller applications as well. The entire instruction set is listed in APPENDIX E. Individual instructions are 16 bits long (some instructions are 32 or 48 bits long), allowing memory to be used efficiently.

The instruction set is divided into the following function groups:

- Arithmetic operation
- Load and store
- Branch
- Logical operation and bit manipulation
- Direct addressing
- Other

● Arithmetic operation

This group of instructions includes typical arithmetic operation instructions (add, subtract, compare) and shift instructions (logical shift, arithmetic operation shift). The add and subtract instructions can also serve for the carry producing operation used for a multiword operation and the operation with fixed flag value useful for address calculation.

This group also includes 32×32-bit and 16×16-bit multiply instructions and a 32/32-bit step divide instruction.

Also included are the immediate-value transfer instruction which sets an immediate value to a register and the register-to-register transfer instruction.

All of the arithmetic operation instructions perform operations using the general-purpose and multiplication/division registers in the CPU.

● Load and store

The load and store instructions read from and write to external memory. These are also used to read from and write to on-chip peripheral resources (I/O).

The load and store instructions support three different access lengths: byte, halfword, and word. They also support general register-indirect memory addressing, and some of them support register-indirect memory addressing with displacement and register-indirect memory addressing with register increment/decrement as well.

● Branch

This group of instructions serves for branching, calling, interrupting, and returning. Some of the branch instructions have a delay slot and the others do not, allowing optimization for specific applications. The branch instructions are detailed later in Section 3.6 "Branch Instructions".

● Logical operation and bit manipulation

The logical operation instructions can perform a logical operation of AND, OR, or EOR either between general-purpose registers or between a general-purpose register and memory (or an I/O). The bit manipulation instructions can directly manipulate the content of memory (or an I/O). This group of instructions supports general register-indirect memory addressing.

- Direct addressing

The direct addressing instructions are used for access between an I/O and either a general-purpose register or memory. This group of instructions enables fast and efficient access by directly addressing the address of the I/O in the instruction instead of indirect addressing using a register. Some of these instructions support register-indirect memory addressing with register increment/decrement as well.

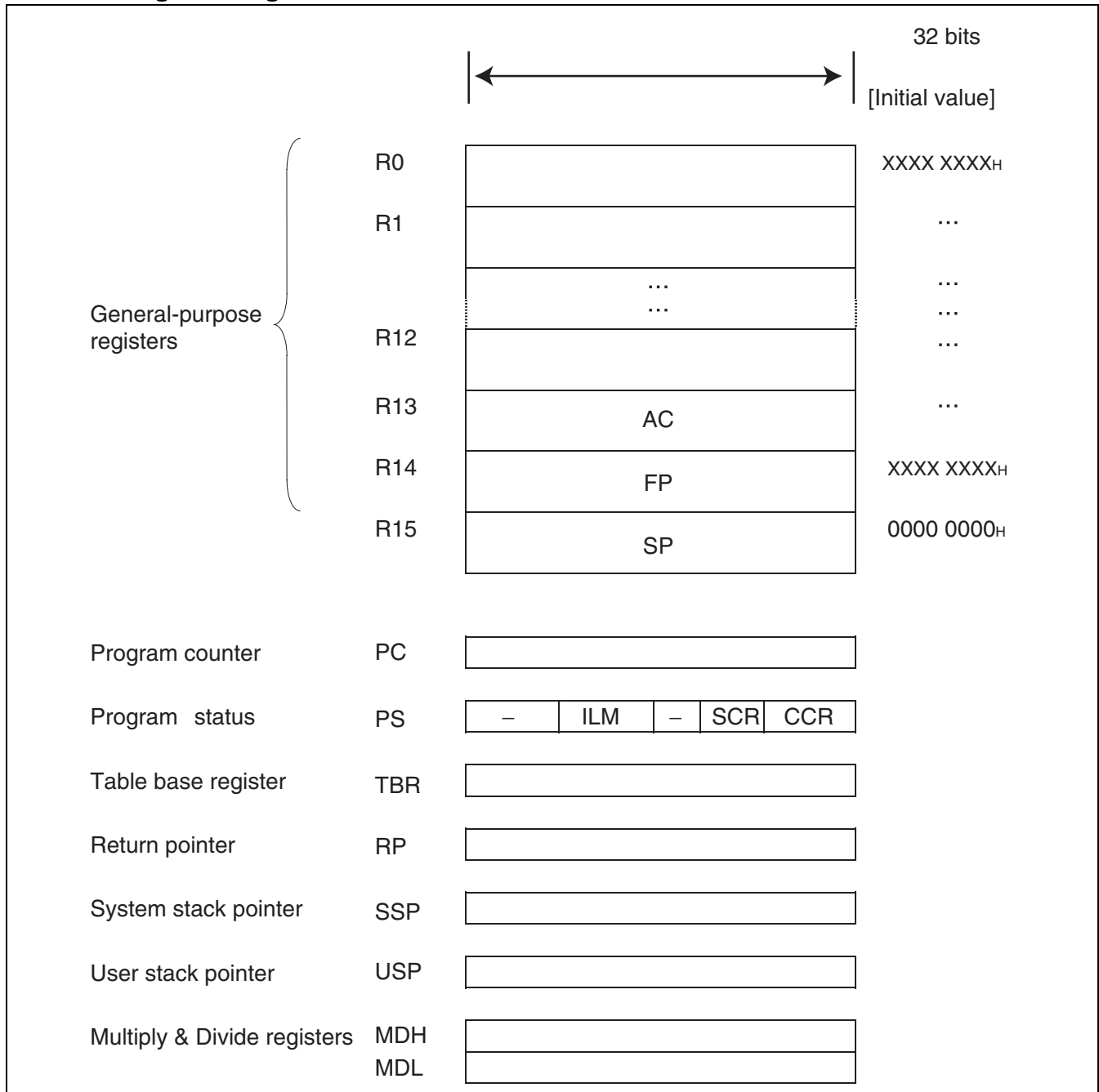
- Other

This group of instructions includes those for PS register flag setting, stack manipulation, sign/zero extension. It also includes the function entrance/exit applicable to high-level languages and register multi-load/store instructions.

3.3 Programming Model

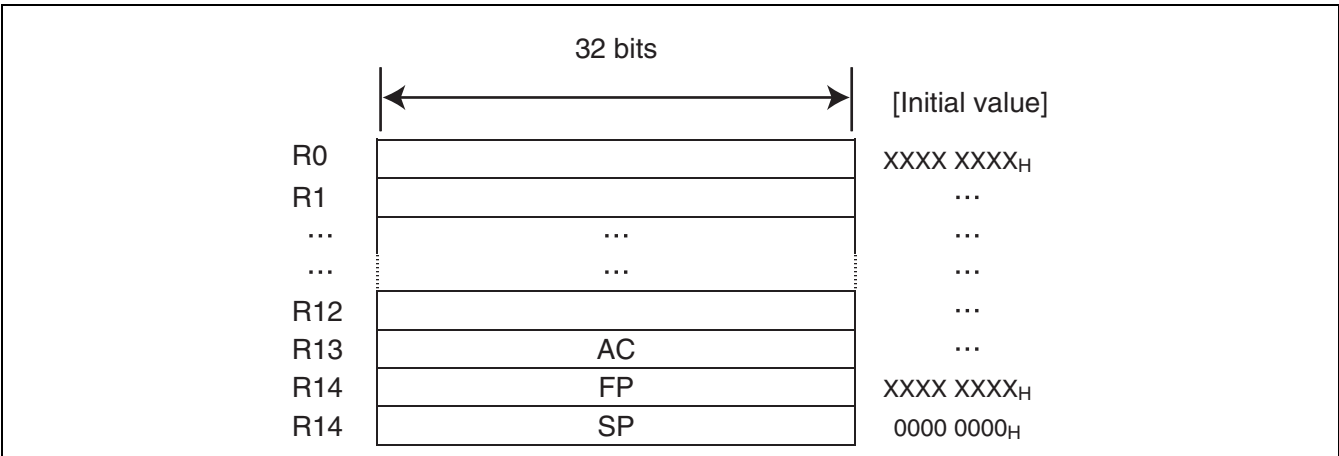
This section explains the programming model of the MB91260B series.

■ Basic Programming Model



■ Registers

● General-purpose registers



Registers R0 to R15 are general-purpose registers. These are used as accumulators for various operations and memory access pointers.

Of these 16 registers, the registers listed below are intended for special applications, for which some instructions are enhanced.

R13: Virtual accumulator

R14: Frame pointer

R15: Stack pointer

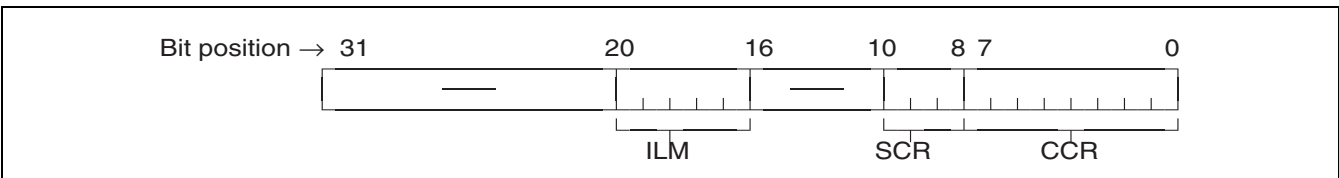
The initial values of R0 to R14 after a reset are indeterminate. The initial value of R15 is "00000000_H" (SSP value).

● PS (Program Status) register

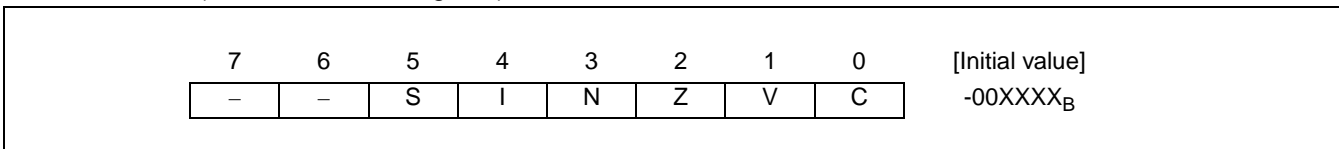
This register holds the program status. It is divided into three parts (registers): ILM, SCR, and CCR.

The undefined bits in the following illustration are all reserved bits. Reading any of these bits always returns "0".

An attempt to write to this register is ignored.



● CCR (Condition Code Register)



[bit5] S: Stack flag

This flag specifies the stack pointer to be used as R15.

Value	Description
0	Selects the SSP to be used as R15. The bit is set to "0" automatically when an EIT occurs. (Note, however, that the value saved to the stack is the one existing before the bit is cleared.)
1	Selects the USP to be used as R15.

This bit is cleared to "0" at a reset.

Set the bit to "0" for execution of the RETI instruction.

[bit4] I: Interrupt enable flag

This flag controls the enabling or disabling of user interrupt requests.

Value	Description
0	Disables user interrupts. The bit is cleared to "0" upon execution of the INT instruction. (Note, however, that the value saved to the stack is the one existing before the bit is cleared.)
1	Enables user interrupts. The masking of user interrupt requests is controlled by the value held in the ILM register.

This bit is cleared to "0" at a reset.

[bit3] N: Negative flag

This flag indicates the sign to be used for the operation result regarded as an integer represented in two's complement.

Value	Description
0	Indicates the operation result as a positive value.
1	Indicates the operation result as a negative value.

The initial state after a reset is indeterminate.

[bit2] Z: Zero flag

This flag indicates whether the operation result is "0".

Value	Description
0	Indicates that the operation result is a value other than "0".
1	Indicates that the operation result is "0".

The initial state after a reset is indeterminate.

[bit1] V: Overflow flag

This flag indicates whether an operand using the operation regarded as an integer represented in two's complement has resulted in an overflow.

Value	Description
0	Indicates that the operation has resulted in no overflow.
1	Indicates that the operation has resulted in an overflow.

The initial state after a reset is indeterminate.

[bit0] C: Carry flag

This flag indicates whether an operation has generated a carry or borrow from the MSB.

Value	Description
0	Indicates that the operation has generated neither a carry nor borrow.
1	Indicates that the operation has generated a carry or borrow.

The initial state after a reset is indeterminate.

● **SCR (System Condition code Register)**

	10	9	8	[Initial value]
	D1	D0	T	XX0 _B

[bit10, bit9] D1, D0: Step division flag

This flag holds intermediate data during execution of step division.

Do not update the content of this flag during execution of the division process.

To execute another process during execution of step division, save and return the values in the PS register so that the step division can be resumed correctly.

The initial state after a reset is indeterminate.

The flag is set by executing the DIV0S instruction to refer the dividend and divisor.

The flag is forced to be cleared by executing the DIV0U instruction.

[bit8] T: Step trace trap flag

This flag enables or disables step trace traps.

Value	Description
0	Disables step trace traps.
1	Enables step trace traps. This disables all of user NMIs and user interrupts.

This bit is initialized to "0" at a reset.

The step trace trap function is used by the emulator. When it is being used by the emulator, it cannot be used in the user program.

● ILM (Interrupt Level Mask) register

20	19	18	17	16	[Initial value]
ILM4	ILM3	ILM2	ILM1	ILM0	01111 _B

The ILM register holds the interrupt level mask value to be used for level masking.

An interrupt request to the CPU is accepted only when its interrupt level is higher than the level held in the ILM register.

The level value ranges from 0 (00000_B) for the highest level to 31 (11111_B) for the lowest.

The level value which can be set from the program is limited.

When the original value is 16 to 31:

The new value which can be set is 16 to 31. After an instruction which sets a value of 0 to 15 is executed, a value of (the specified value + 16) is transferred.

When the original value is 0 to 15:

Any value from 0 to 31 can be set.

This register is initialized to 15 (01111_B) at a reset.

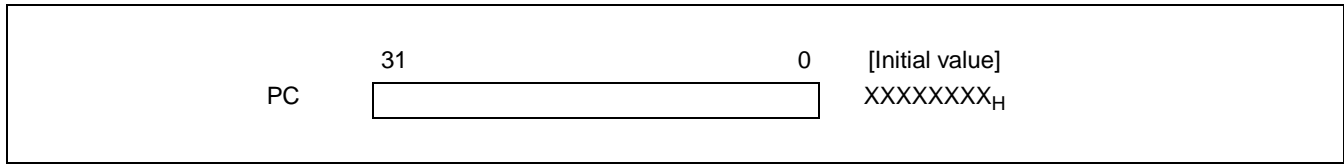
[Notes on the PS register]

Since some instructions manipulate the PS register earlier, the following exceptions may cause the interrupt service routine to break or the PS flag to update its display setting when the debugger is being used.

As the device is designed to carry out reprocessing correctly upon returning from such an EIT event in either case, it performs operations before and after the EIT as specified.

1. The following operations may be performed when the instruction immediately followed by a DIV0U/ DIV0S instruction is (a) accepted by a user interrupt or NMI, (b) single-stepped, or (c) breaks in response to a data event or emulator menu:
 - (1) The D0 and D1 flags are updated earlier.
 - (2) The EIT service routine (user interrupt/NMI or emulator) is executed.
 - (3) Upon returning from the EIT, the DIV0U/DIV0S instruction is executed and the D0 and D1 flags are updated to the same values as those in (1) above.
2. The following operations are performed when the ORCCR/STILM/MOV Ri and PS instructions are executed to enable interrupts when a user interrupt or NMI trigger event has occurred.
 - (1) The PS register is updated earlier.
 - (2) The EIT service routine (user interrupt/NMI) is executed.
 - (3) Upon returning from the EIT, the above instructions are executed and the PS register is updated to the same value as that in (1) above.

● PC (Program Counter)



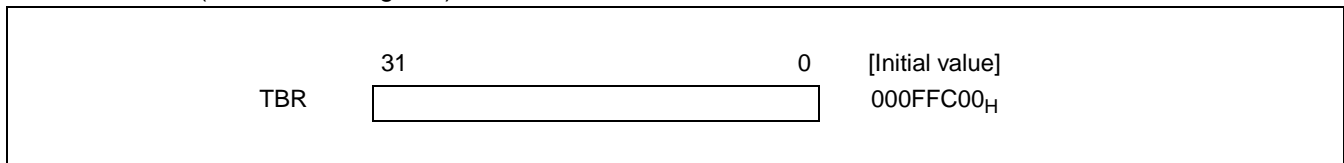
[bit31 to bit0]

The program counter contains the address of the instruction currently being executed.

When the PC is updated as an instruction is executed, bit0 is set to "0". Bit0 may be set to "1" only when an odd-numbered address is specified as the branch destination address. Even in that case, however, bit0 is invalid and the instruction must be placed at an address that is a multiple of the number 2.

The initial value after a reset is indeterminate.

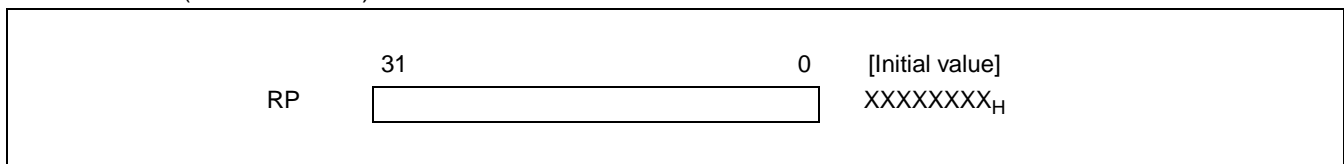
● TBR (Table Base Register)



The table base register contains the start address of the vector table used for servicing EIT events.

The initial value after a reset is 000FFC00_H.

● RP (Return Pointer)



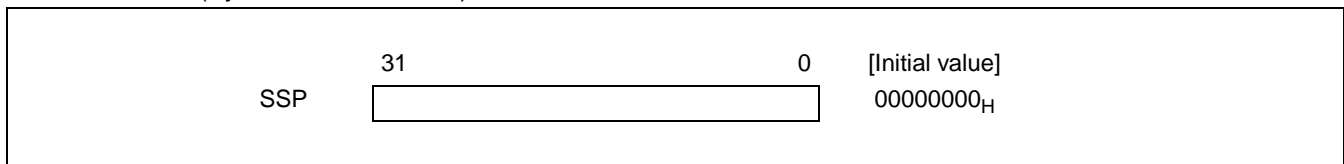
The return pointer contains the return address of a subroutine.

When the CALL instruction is executed, the value in the PC is transferred to the RP.

When the RETI instruction is executed, the value in the RP is transferred to the PC.

The initial value after a reset is indeterminate.

● SSP (System Stack Pointer)



The SSP is a system stack pointer.

It serves as R15 of the general-purpose register when the S-flag contains "0".

The SSP can be explicitly specified. The SSP is also used as the stack pointer that specifies the stack for saving the PS and PC when an EIT event occurs.

The initial value after a reset is "00000000_H".

● USP (User Stack Pointer)



The USP is a user stack pointer.

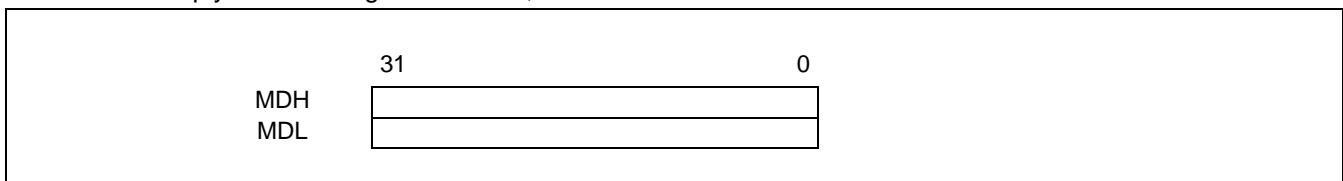
It serves as R15 of the general-purpose register when the S-flag contains "1".

The USP can be explicitly specified.

The initial value after a reset is indeterminate.

This pointer cannot be used by the RETI instruction.

● Multiply & Divide registers : MDH, MDL



These registers hold the results of a multiplication or division. Each of them is 32 bits long.

The initial value after a reset is indeterminate.

When a multiplication is performed:

The operation result of a multiplication of 32 bits \times 32 bits is 64 bits long, which are stored in the multiply & divide result storage registers as follows:

MDH: Upper 32 bits

MDL: Lower 32 bits

The result of a 16×16 -bit multiplication is stored as follows:

MDH: Indeterminate

MDL: Result of 32 bits long

When a division is performed:

The dividend is stored in the MDL at the start of the calculation.

When the division is performed by the DIV0S/DIV0U, DIV1, DIV2, DIV3, or DIV4S instruction, the results are stored in the MDL and MDH as follows:

MDH: Remainder

MDL: Quotient

3.4 Data Structure

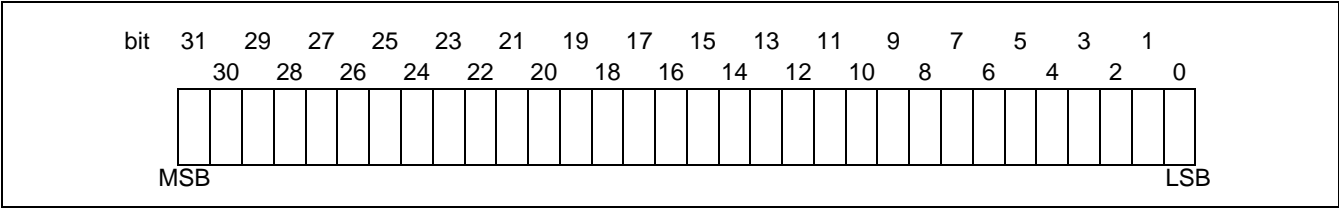
The MB91260B series uses the following two data ordering methods:

- Bit ordering
- Byte ordering

■ Bit Ordering

The FR family uses the little endian method for bit ordering.

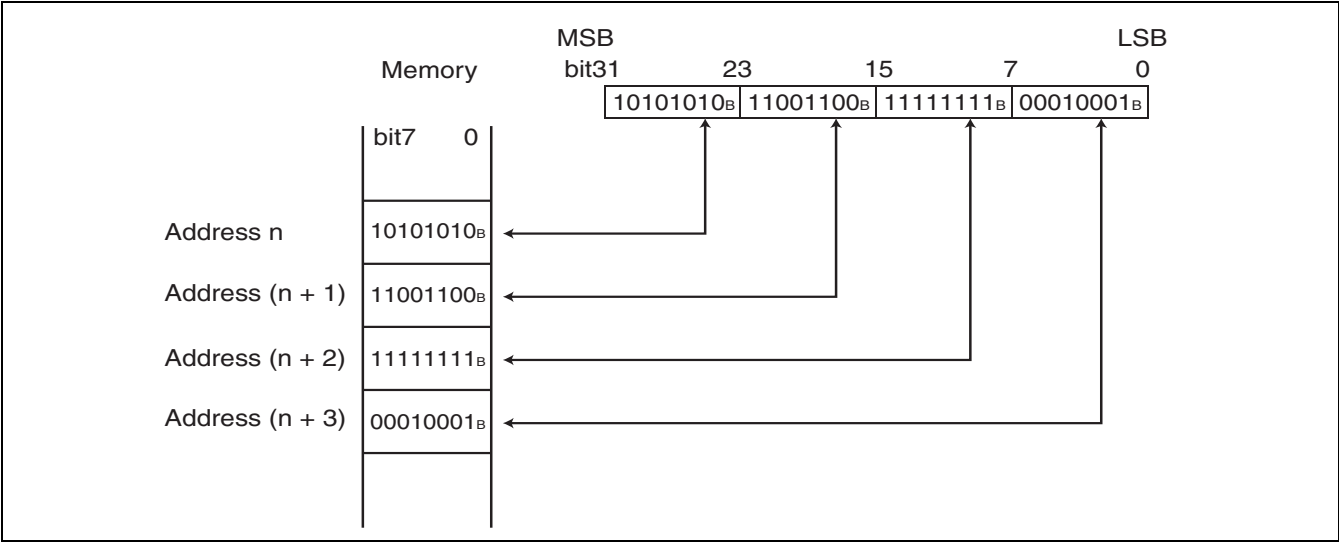
Figure 3.4-1 Bit Ordering



■ Byte Ordering

The FR family uses the big endian method for byte ordering.

Figure 3.4-2 Byte Ordering



3.5 Word Alignment

Since instruction and data are accessed in byte units, the addresses to be placed depend on the instruction length and data width.

■ Program Access

Programs for the MB91260B series must be located at an address that is a multiple of the number 2.

Bit0 in the PC (program counter) is set to "0" when the PC is updated as an instruction is executed. Bit0 may be set to "1" only when an odd-numbered address is specified as the branch destination address. Even in that case, bit0 is invalid and the instruction must be placed at an address that is a multiple of the number 2.

There is no odd-numbered address exception.

■ Data Access

When accessing data, the FR family forces alignment of the address depending on the access width as follows:

Word access : The address is a multiple of the number 4 (with the two LSBs forced to be "00_B").

Half word access : The address is a multiple of the number 2 (with the LSB forced to be "0").

Byte access : –

When word or halfword access is performed, some of the bits in the effective address obtained by calculation are forced to be "0". In the addressing mode with @(R13, Ri), for example, the register before addition (e.g., even with the LSB containing "1") is used for calculation as it is and the lower bits in the result of addition are masked. The register before calculation is not masked.

[Example] : LD@ (R13, R2) , R0

R13	0 0 0 0 2 2 2 2 H
R2	0 0 0 0 0 0 0 3 H
+)	
Result of addition	0 0 0 0 2 2 2 5 H
	↓ Lower two bits are forced to be masked.
Address pin	0 0 0 0 2 2 2 4 H

3.6 Memory Map

This section shows the memory map of the MB91260B series.

■ Memory Map

The address space is a 32-bit, linear space.
Figure 3.6-1 shows the memory map.

Figure 3.6-1 Memory Map

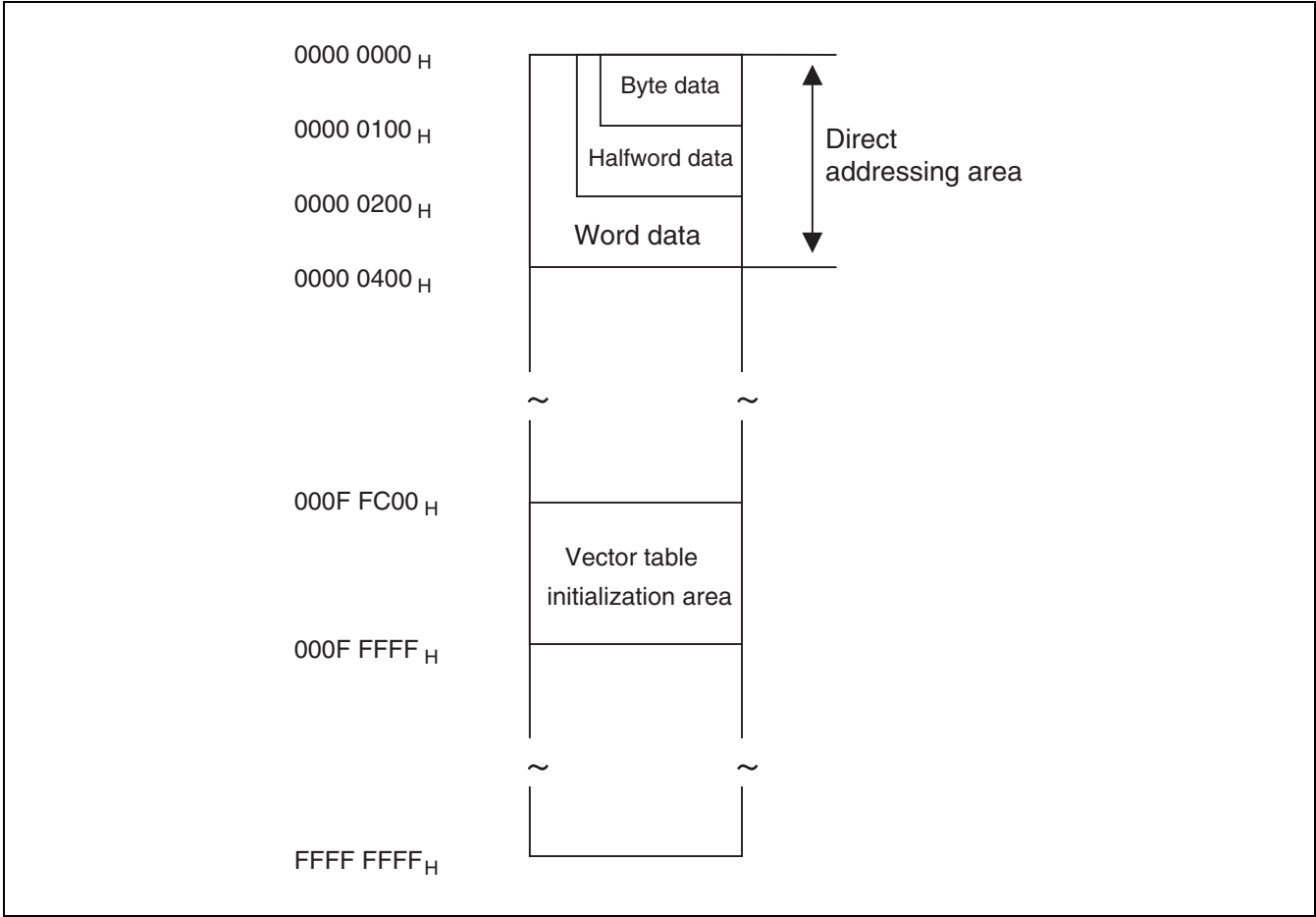
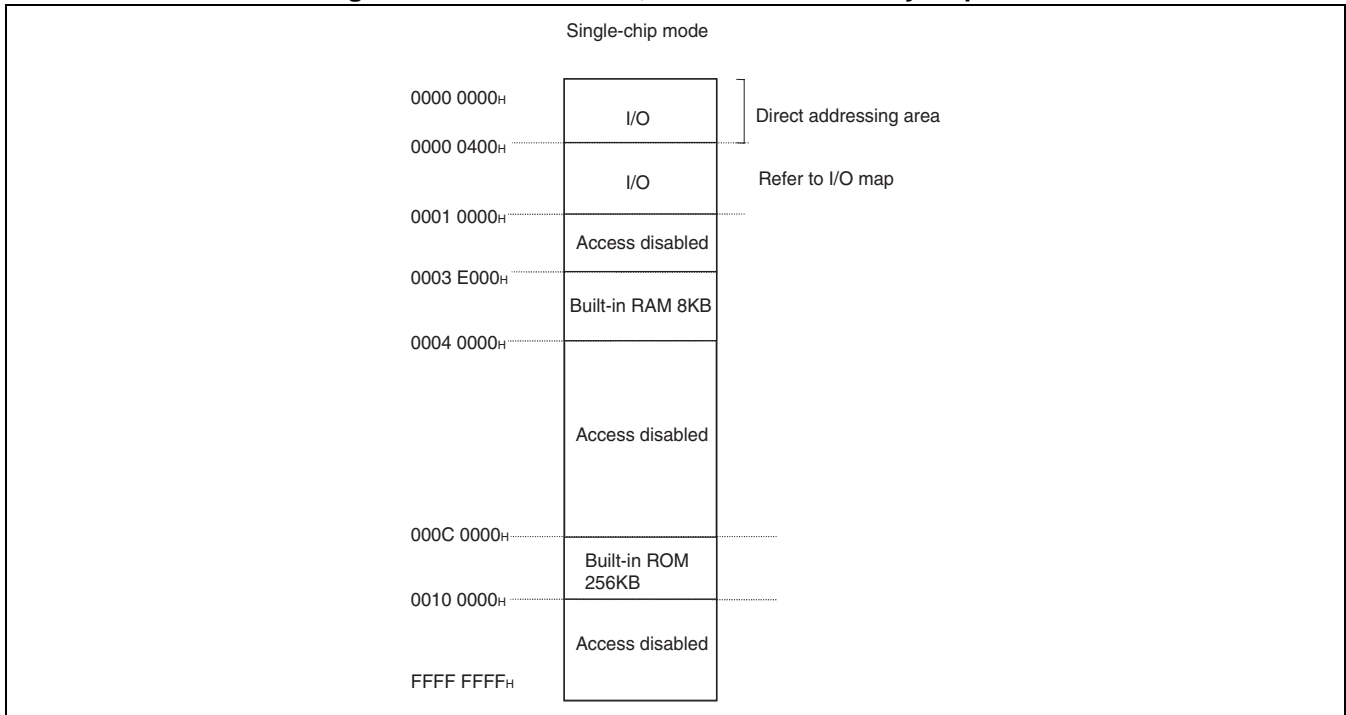
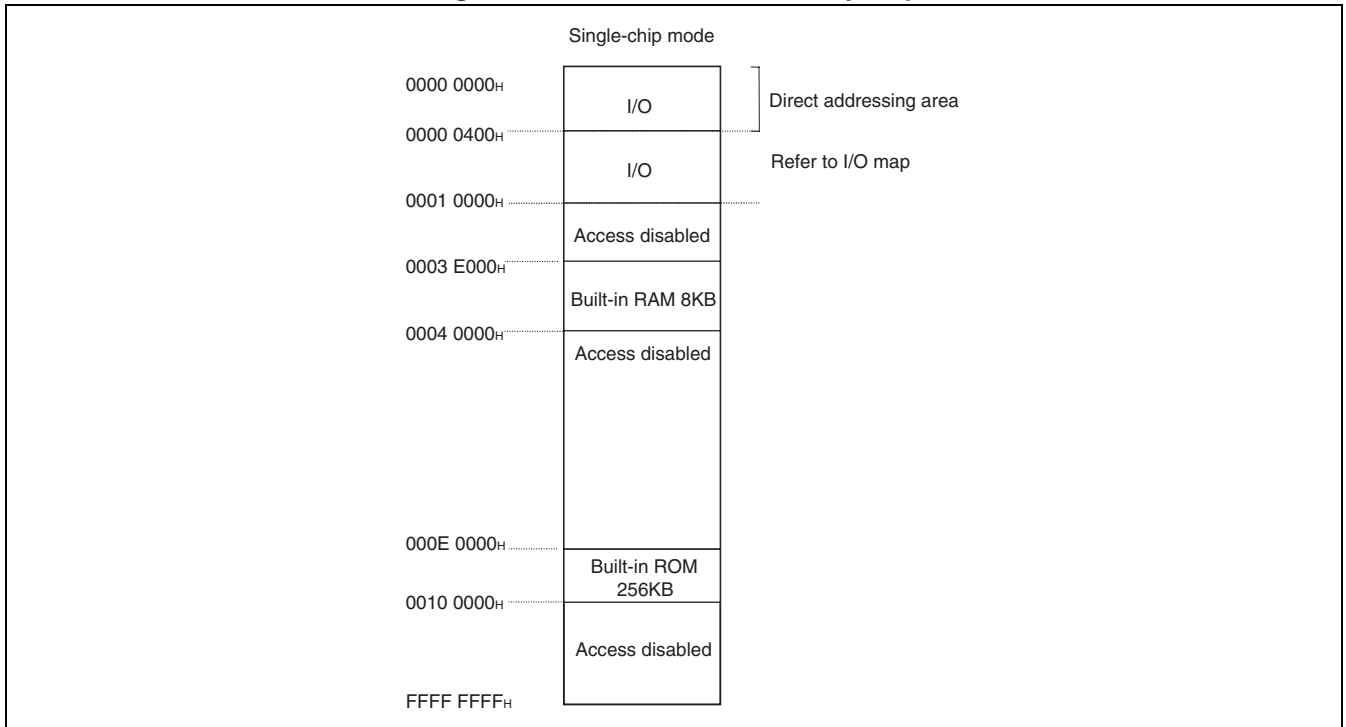


Figure 3.6-2 MB91F264B, MB91264B's Memory Map**Figure 3.6-3 MB91263B's Memory Map**

■ Direct Addressing Area

Each of the following areas in the address space is used for input/output. The area allows direct addressing, where the address of operand can be directly specified in an instruction.

The size of the directly addressable area varies as shown below depending on each data length:

- Byte data (8 bits) : 000_H to 0FF_H
- Halfword data (16 bits) : 000_H to 1FF_H
- Word data (32 bits) : 000_H to 3FF_H

■ Vector Table Initial Area

The area from "000FFC00_H" to "000FFFFFF_H" is an EIT vector table initial area.

The vector table used for servicing EIT events can be located in an arbitrary address by rewriting the TBR. When the device is initialized by a reset, however, the vector table is relocated to this address.

3.7 Branch Instructions

A operation with or without a delay slot can be specified for a branch instruction.

■ Branch Operation with Delay Slot

● Instructions

The instructions listed below perform a branch operation with delay slot:

JMP:D @Ri	CALL:D label12	CALL:D @Ri	RET:D
BRA:D label9	BNO:D label9	BEQ:D label9	BNE:D label9
BC:D label9	BNC:D label9	BN:D label9	BP:D label9
BV:D label9	BNV:D label9	BLT:D label9	BGE:D label9
BLE:D label9	BGT:D label9	BLS:D label9	BHI:D label9

● Explanation of operation

In the operation with delay slot, a branch occurs after the instruction immediately following the branch instruction (called delay slot) is executed, then the instruction at the branch destination is executed. Since the instruction in the delay slot is executed before the delayed branch itself takes place, the execution speed is apparently one cycle. When an effective instruction cannot be put in the delay slot, however, the NOP instruction must be placed instead.

[Example]

```

;      Instruction listing
      ADD    R1, R2    ;
      BRA:D  LABEL    ; Branch instruction
      MOV    R2, R3    ; Delay slot:      Executed before branching
      ...
      LABEL : ST      R3, @R4 ; Branch target

```

When a conditional branch instruction with delay slot is processed, the instruction put in the delay slot is executed even if the branch condition is not satisfied.

When a delayed branch instruction is encountered, some instructions look executed in reverse order. However, the changed order applies only to the update of the PC; the other operations (including register updates and references) are executed exactly in the order in which they are coded.

How the delay slot works is precisely described below.

(1) The Ri referred to by the JMP:D @Ri or CALL:D @Ri instruction is not affected even when the instruction in the delay slot updates the Ri.

[Example]

```

      LDI:32  #Label, R0
      JMP:D   @R0          ; Branch to Label
      LDI:8   #0,         R0      ; No effect on branch destination address
      ...

```

- (2) The RP referred to by the RET:D instruction is not affected even when the instruction in the delay slot updates the RP.

[Example]

```
RET:D                ; Branch to address pointed to by previously set RP
MOV    R8,    RP    ; No effect on return operation
...
```

- (3) The flag referred to by the Bcc:D rel instruction is not affected by the instruction in the delay slot.

[Example]

```
ADD    #1,    R0    ; Update flag
BC:D    Overflow    ; Branch depending on execution result of above instruction
ANDCCR#0                ; This flag update is not referred to by above branch instruction
...
```

- (4) When the instruction in the delay slot of the CALL:D instruction refers the RP, the content of the update made by the CALL:D instruction is read.

[Example]

```
CALL:D Label        ; Update RP and branch
MOV    RP,    R0    ; Transfer RP value resulting from execution of above CALL:D
...
```

● Restrictions

- (1) Instructions that can be executed in a delay slot

The instructions satisfying the following conditions can be executed in the delay slot:

- One-cycle instruction
- Non-branch instruction
- Instruction not affecting the operation even when the execution order is changed

The "one-cycle instruction" means an instruction marked with "1", "a", "b", "c", or "d" in the "CYCLE" column (the number of cycles required) in the instruction list in Appendix E.

- (2) Step trace trap

A step trace trap does not occur between the execution of a branch instruction with delay slot and the delay slot.

- (3) Interrupt or NMI

No interrupt or NMI is accepted between the execution of a branch instruction with delay slot and the delay slot.

- (4) Undefined instruction exception

If an undefined instruction is put in a delay slot, an undefined instruction exception does not occur. In this case, the undefined instruction works as the NOP instruction.

■ Branch Operation without Delay Slot

● Instructions

The instructions listed below perform a branch operation without delay slot:

JMP @Ri	CALL label12	CALL @Ri	RET
BRA label9	BNO label9	BEQ label9	BNE label9
BC label9	BNC label9	BN label9	BP label9
BV label9	BNV label9	BLT label9	BGE label9
BLE label9	BGT label9	BLS label9	BHI label9

● Explanation of operation

When a branch operation without delay slot is encountered, instructions are executed exactly in the order in which they are coded. The instruction that follows is never executed before the branch takes place.

[Example]

```

;      Instruction listing
      ADD    R1, R2    ;
      BRA    LABEL    ; Branch instruction (without delay slot)
      MOV    R2, R3    ; Not executed
      ...
      LABEL ST    R3, @R4 ; Branch target

```

The branch instruction without delay slot requires two execution cycles when it branches control or one execution cycle otherwise.

Compared to the branch instruction with delay slot with NOP specified, the branch instruction without delay slot can improve instruction coding efficiency because there is no effective instruction acceptable to the delay slot.

Select a branch operation with delay slot when an effective instruction can be put in the delay slot. Otherwise, select an operation without delay slot. This improves both of execution speed and coding efficiency.

3.8 EIT (Exception/Interrupt/Trap)

EIT indicates that a program being executed is suspended by an event for the purpose of executing another program. EIT is a generic term for exception, interrupt, and trap.

■ EIT (Exception, Interrupt, and Trap)

An exception is an event that occurs in relation to the current context. The CPU resumes execution of the suspended program from the instruction that caused the exception.

An interrupt is an event that occurs independently of the current context. The event is generated by hardware (event source).

A trap is also an event that occurs in relation to the current context. Some traps are programmed as system calls. The CPU resumes execution of the suspended program from the instruction immediately following the instruction that caused the trap.

■ Features

- Multiple-interrupt support
- Interrupt level masking function (15 levels available to the user)
- Trap instruction (INT)
- Emulator trigger EIT (hardware or software)

■ EIT Sources

EIT sources are as follows:

- Reset
- User interrupt (internal resource or external interrupt)
- NMI (nonmaskable interrupt)
- Delayed interrupt
- Undefined instruction exception
- Trap instruction (INT)
- Trap instruction (INTE)
- Step trace trap
- Coprocessor absence trap
- Coprocessor error trap

■ Return from EIT

RETI instruction is executed to return from EIT.

3.8.1 Interrupt Level

Interrupt levels are 0 to 31; they are managed with five bits.

■ Interrupt Levels

Each interrupt level is assigned as follows.

Table 3.8-1 Interrupt Levels

Level		Interrupt source	Note
Binary	Decimal		
00000 _B	0	(System-reserved)	When the original ILM value is 16 to 31, the ILM register cannot be set to a value in that range by a program.
...	
...	
00011 _B	3	(System-reserved)	
00100 _B	4	{ INTE instruction Step trace trap	
00101 _B	5	(System-reserved)	
...	
...	
01110 _B	14	(System-reserved)	
01111 _B	15	NMI (for user)	
10000 _B	16	Interrupt	No user interrupt is allowed with the ILM register set.
10001 _B	17	Interrupt	
...	
...	
11110 _B	30	Interrupt	
11111 _B	31	—	No interrupt is allowed with the ICR set.

Interrupt levels 16 to 31 can be operated.

Interrupt levels have no effect on the undefined instruction exception, coprocessor absence trap, coprocessor error trap, or INT instruction. Neither they change the ILM value.

■ I-flag

This flag enables or disables interrupts. It is provided as CCR bit 4 in the PS register.

Value	Description
0	Disables interrupts. The flag is cleared to "0" when the INT instruction is executed. (Note that the value saved to the stack is the one existing before the flag is cleared.)
1	Enables interrupts. The masking of interrupt requests is controlled by the level value held in the ILM register.

■ ILM Register

The interrupt level mask (ILM) register is the PS register (bit20 to bit16) that holds an interrupt level mask value.

An interrupt request to the CPU is accepted only when its interrupt level is higher than the level indicated in the ILM register.

The highest level value is 0 (00000_B); the lowest is 31 (11111_B).

The level value which can be set in the program is limited. When the original value is 16 to 31, the new value which can be set is 16 to 31. After an instruction which sets a value of 0 to 15 is executed, a value of (the specified value + 16) is transferred.

When the original value is 0 to 15, any value from 0 to 31 can be set. The STILM instruction is used to set this register.

■ Interrupt/NMI Level Masking

If an NMI or interrupt request occurs, the interrupt level of the interrupt source (see Table 3.8-1) is compared with the level mask value held in the ILM register. The request is masked and rejected when the following condition is satisfied:

Interrupt level of the source \geq Level mask value

3.8.2 ICR (Interrupt Control Register)

The interrupt control register (ICR) is provided in the interrupt controller to set a level for each interrupt request. This register is prepared for each interrupt request input. The register is mapped in the I/O space and accessed from the CPU through a bus.

■ ICR Bit Configuration

ICR00 to ICR47	7	6	5	4	3	2	1	0	Initial value
Address	—	—	—	ICR4	ICR3	ICR2	ICR1	ICR0	---11111 _B
000440 _H to 00046F _H	—	—	—	R	R/W	R/W	R/W	R/W	[R/W, R]

[bit4] ICR4

ICR4 always contains "1".

[bit3 to bit0] ICR3 to ICR0

These are the lower four bits indicating the interrupt level of the corresponding interrupt source.

The ICR, consisting of these bits and bit4, can set a value between 16 and 31.

These bits can be read and written.

■ ICR Mapping

Table 3.8-2 shows the relationship between the interrupt source, interrupt control register, and interrupt vector.

See CHAPTER 5 "INTERRUPT CONTROLLER" for detail of the interrupt.

Table 3.8-2 Interrupt Sources, Interrupt Control Registers, and Interrupt Vectors

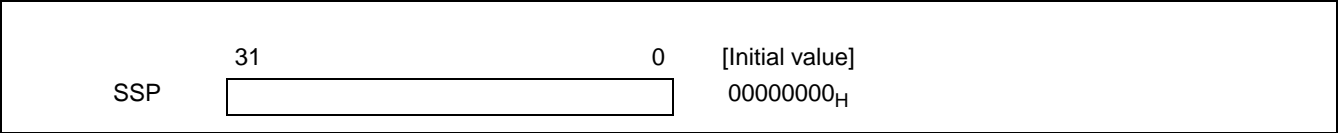
Interrupt source	Interrupt control register		Corresponding interrupt vector		
			Number		Address
			Hex	Dec	
IRQ00	ICR00	00000440 _H	10 _H	16	TBR + 3BC _H
IRQ01	ICR01	00000441 _H	11 _H	17	TBR + 3B8 _H
IRQ02	ICR02	00000442 _H	12 _H	18	TBR + 3B4 _H
...
...
IRQ45	ICR45	0000046D _H	3D _H	61	TBR + 308 _H
IRQ46	ICR46	0000046E _H	3E _H	62	TBR + 304 _H
IRQ47	ICR47	0000046F _H	3F _H	63	TBR + 300 _H

TBR initial value: "000FFC00_H"

3.8.3 SSP (System Stack Pointer)

The system stack pointer (SSP) serves as the pointer that points to the stack for saving data to upon receipt of an EIT or for restoring data from upon recovery from the EIT.

The pointer value is decremented by "8" when an EIT is serviced and incremented by "8" when the RETI instruction is executed to return from the EIT.



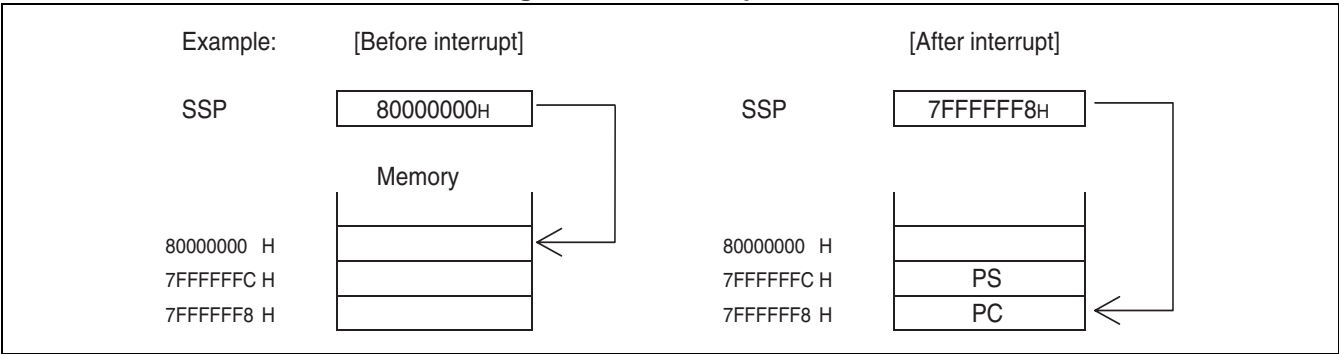
The SSP serves also as the general-purpose register R15 when the S-flag in the CCR (condition code register) contains "0".

The initial value after a reset is "00000000_H".

■ Interrupt Stack

The interrupt stack is the area pointed to by the SSP, to/from which the PC and PS values are saved/restored. After an interrupt occurs, the PC and PS values are stored at the address held in the SSP and at the address of (SSP + 4), respectively.

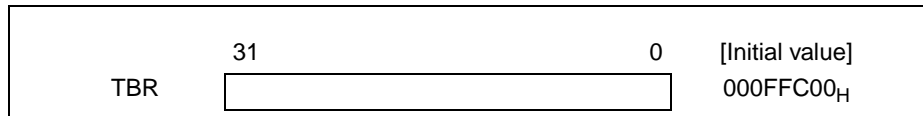
Figure 3.8-1 Interrupt Stack



3.8.4 TBR (Table Base Register)

The table base register (TBR) indicates the start address of the EIT vector table.

■ TBR (Table Base Register)



The vector address is obtained by adding the offset value determined for the EIT source and the TBR value. The initial value after a reset is "000FFC00_H".

■ EIT Vector Table

The EIT vector area is 1 Kbyte long, starting from the address shown in the TBR.

The area size per vector is 4 bytes, and the relationship between vector number and vector address is expressed as follows:

$$\begin{aligned} \text{vctadr} &= \text{TBR} + \text{vctofs} \\ &= \text{TBR} + (3\text{FC}_{\text{H}} - 4 \times \text{vct}) \\ \text{vctadr:} &\quad \text{Vector address} \\ \text{vctofs:} &\quad \text{Vector offset} \\ \text{vct:} &\quad \text{Vector number} \end{aligned}$$

The lower 2 bits of the addition result are always handled as "00_B".

The area from "000FFC00_H" to "000FFFFFF_H" is the initial area for the vector table upon a reset.

Special functions are assigned to some of the vectors.

Table 3.8-3 shows the vector table on the architecture.

Table 3.8-3 Vector Table (1 / 3)

Interrupt source	Interrupt No.		Interrupt level	Offset	TBR default address
	Dec	Hex			
Reset*	0	00	—	3FC _H	000FFFFC _H
Mode vector*	1	01	—	3F8 _H	000FFFF8 _H
System-reserved	2	02	—	3F4 _H	000FFFF4 _H
System-reserved	3	03	—	3F0 _H	000FFFF0 _H
System-reserved	4	04	—	3EC _H	000FFFE4 _H
System-reserved	5	05	—	3E8 _H	000FFFE0 _H
System-reserved	6	06	—	3E4 _H	000FFFE4 _H
Coprocessor absence trap	7	07	—	3E0 _H	000FFFE0 _H
Coprocessor error trap	8	08	—	3DC _H	000FFFD8 _H
INTE instruction	9	09	—	3D8 _H	000FFFD8 _H
System-reserved	10	0A	—	3D4 _H	000FFFD4 _H
System-reserved	11	0B	—	3D0 _H	000FFFD0 _H
Step trace trap	12	0C	—	3CC _H	000FFFC8 _H
NMI request (tool)	13	0D	—	3C8 _H	000FFFC8 _H
Undefined instruction exception	14	0E	—	3C4 _H	000FFFC4 _H

Table 3.8-3 Vector Table (2 / 3)

Interrupt source	Interrupt No.		Interrupt level	Offset	TBR default address
	Dec	Hex			
NMI request	15	0F	Fixed at 15(F _H)	3C0 _H	000FFFC0 _H
External interrupt 0	16	10	ICR00	3BC _H	000FFFBC _H
External interrupt 1	17	11	ICR01	3B8 _H	000FFFB8 _H
External interrupt 2	18	12	ICR02	3B4 _H	000FFFBA _H
External interrupt 3	19	13	ICR03	3B0 _H	000FFFB0 _H
External interrupt 4	20	14	ICR04	3AC _H	000FFFAC _H
External interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H
External interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H
External interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H
Reload timer 0	24	18	ICR08	39C _H	000FFF9C _H
Reload timer 1	25	19	ICR09	398 _H	000FFF98 _H
Reload timer 2	26	1A	ICR10	394 _H	000FFF94 _H
UART0 (Reception)	27	1B	ICR11	390 _H	000FFF90 _H
UART0 (Transmission)	28	1C	ICR12	38C _H	000FFF8C _H
DTTI	29	1D	ICR13	388 _H	000FFF88 _H
DMAC0 (Termination, Error)	30	1E	ICR14	384 _H	000FFF84 _H
DMAC1 (Termination, Error)	31	1F	ICR15	380 _H	000FFF80 _H
DMAC2/3/4 (Termination, Error)	32	20	ICR16	37C _H	000FFF7C _H
UART1 (Reception complete)	33	21	ICR17	378 _H	000FFF78 _H
UART1 (Transmission complete)	34	22	ICR18	374 _H	000FFF74 _H
UART2 (Reception complete)	35	23	ICR19	370 _H	000FFF70 _H
UART2 (Transmission complete)	36	24	ICR20	36C _H	000FFF6C _H
Multiply-accumulate	37	25	ICR21	368 _H	000FFF68 _H
PPG0	38	26	ICR22	364 _H	000FFF64 _H
PPG1	39	27	ICR23	360 _H	000FFF60 _H
PPG2/3	40	28	ICR24	35C _H	000FFF5C _H
PPG4/5/6/7	41	29	ICR25	358 _H	000FFF58 _H
PPG8/9/10/11/12/13/14/15	42	2A	ICR26	354 _H	000FFF54 _H
External interrupt 8/9	43	2B	ICR27	350 _H	000FFF50 _H
Waveform 0 (Underflow)	44	2C	ICR28	34C _H	000FFF4C _H
Waveform 1 (Underflow)	45	2D	ICR29	348 _H	000FFF48 _H
Waveform 2 (Underflow)	46	2E	ICR30	344 _H	000FFF44 _H
Timebase timer overflow	47	2F	ICR31	340 _H	000FFF40 _H
Free-running timer (Compare clear)	48	30	ICR32	33C _H	000FFF3C _H
Free-running timer (Zero detection)	49	31	ICR33	338 _H	000FFF38 _H
A/D0	50	32	ICR34	334 _H	000FFF34 _H
A/D1	51	33	ICR35	330 _H	000FFF30 _H
A/D2	52	34	ICR36	32C _H	000FFF2C _H

Table 3.8-3 Vector Table (3 / 3)

Interrupt source	Interrupt No.		Interrupt level	Offset	TBR default address
	Dec	Hex			
PWC0 (Measurement complete)	53	35	ICR37	328 _H	000FFF28 _H
PWC1 (Measurement complete)	54	36	ICR38	324 _H	000FFF24 _H
PWC0 (Overflow)	55	37	ICR39	320 _H	000FFF20 _H
PWC1 (Overflow)	56	38	ICR40	31C _H	000FFF1C _H
ICU 0 (Capture)	57	39	ICR41	318 _H	000FFF18 _H
ICU 1 (Capture)	58	3A	ICR42	314 _H	000FFF14 _H
ICU 2/3 (Capture)	59	3B	ICR43	310 _H	000FFF10 _H
OCU0/1 (Match)	60	3C	ICR44	30C _H	000FFF0C _H
OCU2/3 (Match)	61	3D	ICR45	308 _H	000FFF08 _H
OCU4/5 (Match)	62	3E	ICR46	304 _H	000FFF04 _H
Delayed interrupt source bit	63	3F	ICR47	300 _H	000FFF00 _H
System-reserved (Used by REALOS)	64	40	–	2FC _H	000FFEFC _H
System-reserved (Used by REALOS)	65	41	–	2F8 _H	000FFE8 _H
System-reserved	66	42	–	2F4 _H	000FFE4 _H
System-reserved	67	43	–	2F0 _H	000FFE0 _H
System-reserved	68	44	–	2EC _H	000FFEEC _H
System-reserved	69	45	–	2E8 _H	000FFEE8 _H
System-reserved	70	46	–	2E4 _H	000FFEE4 _H
System-reserved	71	47	–	2E0 _H	000FFEE0 _H
System-reserved	72	48	–	2DC _H	000FFEDC _H
System-reserved	73	49	–	2D8 _H	000FFED8 _H
System-reserved	74	4A	–	2D4 _H	000FFED4 _H
System-reserved	75	4B	–	2D0 _H	000FFED0 _H
System-reserved	76	4C	–	2CC _H	000FFECC _H
System-reserved	77	4D	–	2C8 _H	000FFEC8 _H
System-reserved	78	4E	–	2C4 _H	000FFEC4 _H
System-reserved	79	4F	–	2C0 _H	000FFEC0 _H
Used by INT instruction	80 to 255	50 to FF	–	2BC _H to 000 _H	000FFEBC _H to 000FFC00 _H

*: Even when the TBR value is changed, the reset vector and mode vector are always fixed at addresses "000FFFFC_H" and "000FFF8_H", respectively.

3.8.5 Multi-EIT Servicing

If two or more EIT sources occur simultaneously, the CPU selects and receives one EIT source, executes the EIT sequence, then detects another EIT source, repeatedly.

If there is no more EIT source detected to be received, the CPU executes the instruction of the handler for the EIT source received last.

When two or more EIT sources occur simultaneously, therefore, the execution order of their respective handlers is determined by the following two elements:

- Reception priority of EIT source
- Masking of other sources after reception

■ Reception Priority of EIT Source

The reception priority of EIT source determines the order in which EIT sources are selected for executing the EIT sequence where the PS and PC values are saved, the PC is updated, and other sources are masked (as required).

The handlers of EIT sources received earlier are not always executed earlier on a first-in first-out basis.

Table 3.8-4 lists the reception priorities of EIT sources along with the masking levels of other sources.

Table 3.8-4 Reception Priorities of EIT Sources and the Masking of Other Sources

Reception priority	Source	Masking of other sources
1	Reset	Discard other sources
2	Undefined instruction exception	Cancel
3	INTE instruction	ILM=4 Discard other sources
4	INT instruction	I-flag = 0
5	Coprocessor absence trap Coprocessor error trap	—
6	User interrupt	ILM = Level of received source
7	NMI (for user)	ILM=15
8	NMI (for emulator)	ILM=4
9	Step trace trap	ILM=4

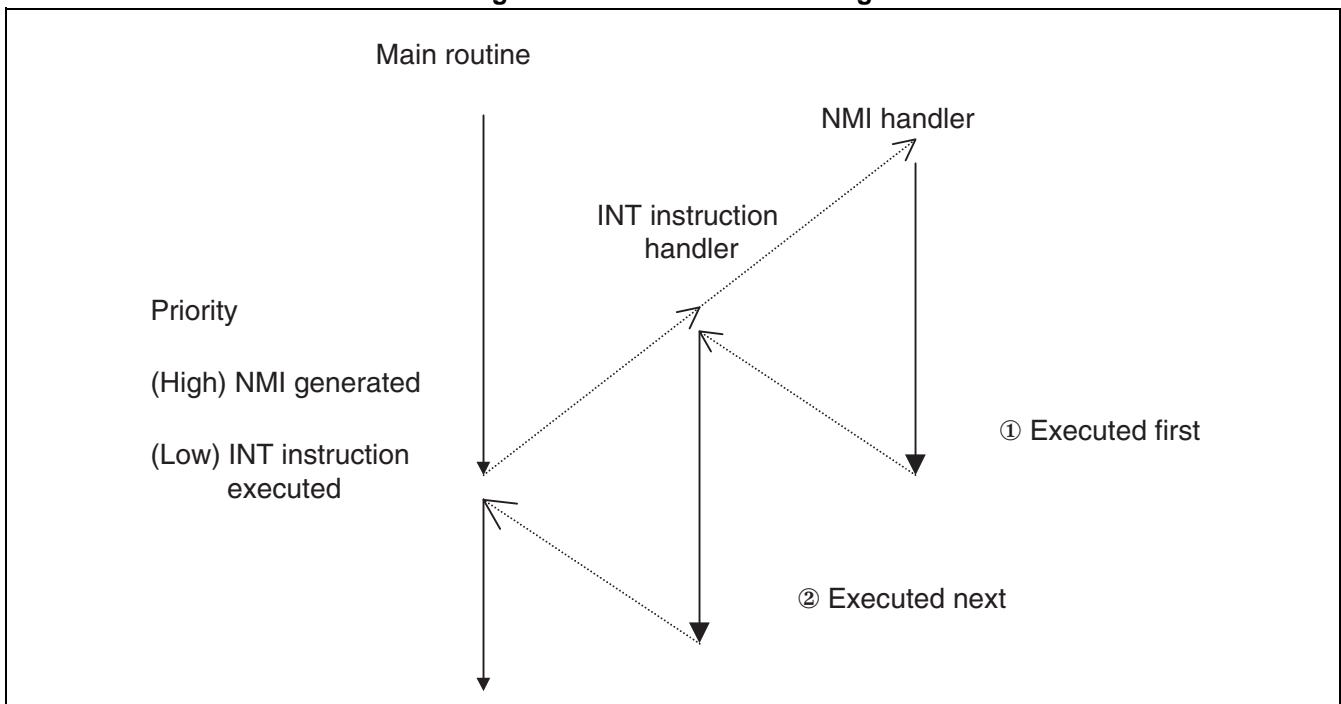
Considering the masking of other sources after an EIT source is received, the handlers for the simultaneously generated EIT sources are executed in the order shown in Table 3.8-5.

Table 3.8-5 EIT Handler Execution Order

Handler execution order	Source
1	Reset*
2	Undefined instruction exception
3	INTE instruction*
4	Step trace trap
5	NMI (for user)
6	INT instruction
7	User interrupt
8	Coprocessor absence trap, Coprocessor error trap

*: Other sources are discarded.

Example:

Figure 3.8-2 Multi-EIT Servicing

3.8.6 Operation

This section explains the operation of EIT.

The "PC" (program counter) as the source of transfer hereafter indicates the address of an instruction where each EIT source has been detected. The "next instruction's address" indicates as follows depending on the EIT-detected instruction:

- LDI: 32 → PC value + 6
- LDI: 20, COPOP, COPLD, COPST, COPSV → PC value + 4
- Else → PC value + 2

Parentheses () in [Processing] below enclose the address specified by the relevant register.

■ Operation of User Interrupt/NMI

When a user interrupt or user NMI interrupt request is issued, the acceptance of the request is determined in the following order:

[Determining the acceptance of interrupt request]

1. The interrupt levels of requests issued simultaneously are compared and the request of the highest level (smallest value) is selected.
The level of a maskable interrupt is compared with the value held in the corresponding ICR and that of an NMI is compared with a predetermined constant.
2. If two or more interrupt requests of the same level are issued, the interrupt request of the smallest interrupt number is selected.
3. The selected interrupt request is masked and rejected if its interrupt level is equal to or greater than the level mask value.
When the interrupt level is smaller than the level mask value, go to step 4. below.
4. If the selected interrupt request is maskable, the interrupt request is masked and rejected when the I-flag is "0". When the I-flag is "1", go to step 5. below.
If the selected interrupt request is an NMI, go to step 5. below regardless of the value in the I-flag.
5. When the above conditions are satisfied, the interrupt request is accepted at a break of instruction processing.

If a user interrupt/NMI request is received upon detection of an EIT request, the CPU performs the following steps using the interrupt number corresponding to the accepted interrupt request.

[Processing]

1. SSP-4 → SSP
2. PS → (SSP)
3. SSP-4 → SSP
4. Next instruction's address → (SSP)
5. Interrupt level of accepted request → ILM
6. "0" → S-flag
7. (TBR + vector offset of accepted interrupt request) → PC

The CPU tries to detect a new EIT before executing the first instruction in the handler upon completion of the interrupt sequence. If an acceptable EIT has occurred, the CPU enters the EIT servicing sequence.

■ Processing of INT Instruction

INT #u8

This instruction causes a branch to the interrupt handler of the vector indicated by u8.

[Processing]

1. $SSP - 4 \rightarrow SSP$
2. $PS \rightarrow (SSP)$
3. $SSP - 4 \rightarrow SSP$
4. $PC + 2 \rightarrow (SSP)$
5. "0" \rightarrow I-flag
6. "0" \rightarrow S-flag
7. $(TBR + 3FC_H - 4 \times u8) \rightarrow PC$

■ Processing of INTE Instruction

INTE

This instruction causes a branch to the interrupt handler of the vector with vector No. #9.

[Processing]

1. $SSP - 4 \rightarrow SSP$
2. $PS \rightarrow (SSP)$
3. $SSP - 4 \rightarrow SSP$
4. $PC + 2 \rightarrow (SSP)$
5. "00100_B" \rightarrow ILM
6. "0" \rightarrow S-flag
7. $(TBR + 3D8_H) \rightarrow PC$

Do not use an INTE instruction in another INTE instruction or in a step trace trap service routine. The INTE instruction does not generate an EIT during single stepping.

■ Processing of Step Trace Trap

If the T-flag in the SCR in the PS register is set to enable the step trace function, a trap occurs at the execution of every instruction, causing a break.

[Step trace trap detection conditions]

- T-flag = 1
- The instruction is not a delayed branch instruction.
- During execution other than the INTE instruction and step trace trap service routine.
- If the above conditions are satisfied, a break is inserted between the executions of instructions.

[Processing]

1. $SSP - 4 \rightarrow SSP$
2. $PS \rightarrow (SSP)$
3. $SSP - 4 \rightarrow SSP$
4. Next instruction's address $\rightarrow (SSP)$

5. "00100_B" → ILM
6. "0" → S-flag
7. (TBR + 3CC_H) → PC

If the T-flag is set to enable the step trace trap function, user NMIs and user interrupts are disabled. In addition, the INTE instruction no longer generates an EIT.

A trap is generated at the execution of the instruction immediately after setting the T-flag.

■ Processing of Undefined Instruction Exception

If an instruction is found undefined during instruction decoding, an undefined instruction exception occurs.

[Undefined instruction exception detection conditions]

- Instruction found undefined during instruction decoding
- Placed outside the delay slot (not immediately after the delayed branch instruction)
- If the above conditions are satisfied, an undefined instruction exception occurs, causing a break.

[Processing]

1. SSP - 4 → SSP
2. PS → (SSP)
3. SSP - 4 → SSP
4. PC → (SSP)
5. "0" → S-flag
6. (TBR + 3C4_H) → PC

The PC saves the address of the instruction where the undefined instruction exception was detected.

■ Coprocessor Absence Trap

A coprocessor absence trap occurs if a coprocessor instruction which attempts to use an absent coprocessor is executed.

[Processing]

1. SSP - 4 → SSP
2. PS → (SSP)
3. SSP - 4 → SSP
4. Next instruction's address → (SSP)
5. "0" → S-flag
6. (TBR + 3E0_H) → PC

■ Coprocessor Error Trap

A coprocessor error trap occurs if an error occurs when a coprocessor is being used and if a coprocessor instruction which attempts to operate the coprocessor is executed then.

[Processing]

1. $SSP - 4$ → SSP
2. PS → (SSP)
3. $SSP - 4$ → SSP
4. Next instruction's address → (SSP)
5. "0" → S-flag
6. $(TBR + 3DC_H)$ → PC

■ Processing of RETI Instruction

The RETI instruction returns control from an EIT service routine.

[Processing]

1. $(R15)$ → PC
2. $R15 + 4$ → $R15$
3. $(R15)$ → PS
4. $R15 + 4$ → $R15$

The RETI instruction must be executed with the S-flag containing "0".

Note:

The delay slot following a branch instruction has a restriction on EITs.
See Section 3.7 "Branch Instructions".

3.9 Operation Modes

Operation modes of the MB91260B series are divided into bus modes and access modes.

However, only single chip mode is supported.

■ Operation Modes

Bus modes

Single chip

● Bus mode

Bus mode controls the operations of internal ROM and the external access functions according to the mode pins (MD2 to MD0).

Note:

The MB91260B series does not support the external bus mode.

■ Bus Modes

See Section 3.6 "Memory Map".

● Single-chip mode

This mode enables access to internal I/O, internal RAM, and internal ROM while disabling access to any other area. External pins serve as peripheral resources or general-purpose ports but not as bus pins.

■ Mode Setting

The FR family uses the mode pins (MD2 to MD0) and the mode data to set the operation mode.

● Mode pins

Three mode pins MD2, MD1, and MD0 are used to specify a mode vector and reset vector fetch.

The mode pins must not be set to any combination of values other than listed in the table below:

Mode pins MD2 MD1 MD0	Mode name	Reset vector access area	Remarks
0 0 0 _B	Internal ROM mode vector	Internal	

Note:

MD2, MD1, MD0: The settings without 000B are prohibited.

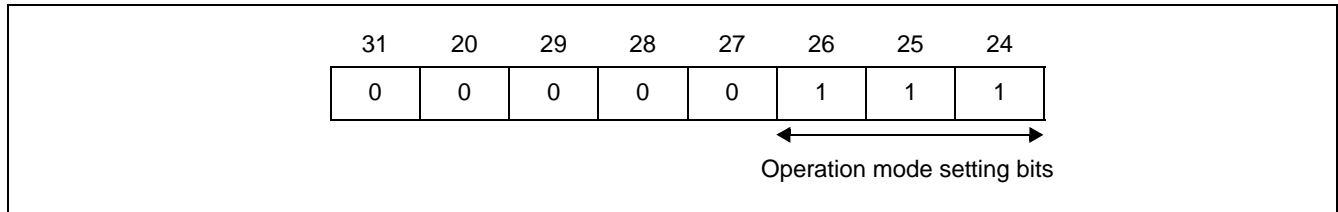
● Mode data

The data written to the internal mode register (MODR) using a mode vector fetch (see "■Reset Sequence" in "3.10 Reset (Device Initialization)") is called mode data.

After the mode register is set, the device operates in the operation mode corresponding to the register setting.

Although the mode register is set by all reset sources, user programs cannot write data to the register.

<Mode data description>



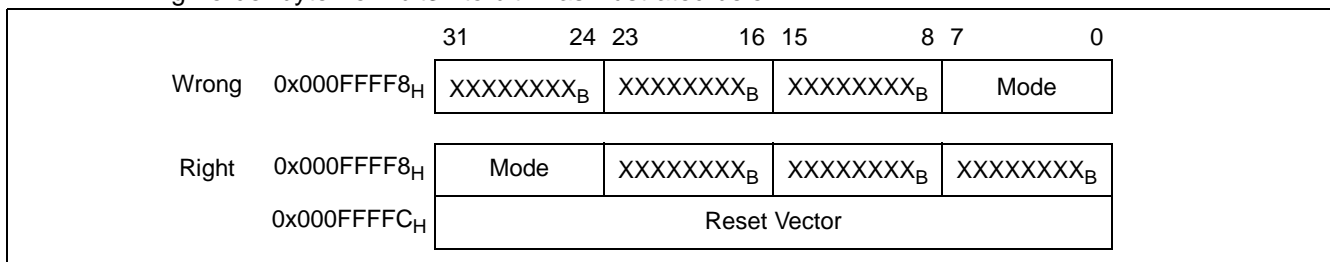
[bit31 to bit24] Reserved bits

Be sure to set these bits to "00000111_B". Setting them to any other value may result in an unpredictable operation.

Note:

Mode data to be set at a mode vector must be placed as byte data at "0x000FFFF8_H".

As the MB91260B series uses the big endian method for byte ordering, place the mode data in the high-order byte from bit31 to bit24 as illustrated below.



3.10 Reset (Device Initialization)

This section describes the reset operation that is initialization of the MB91260B series. When a reset source occurs, the device stops all of the program and hardware operations and initializes the device status. This initialized state is called the reset state.

When the reset source is eliminated, the device restarts program and hardware operations from the initial state. A series of steps taken by the device from the reset state to the beginning of the restart is called the reset sequence.

■ Reset Levels

The MB91260B series resets itself at either of the two levels which are different in the cause of reset and the effect of initialization. Each reset level is described below.

● Setting initialization reset (INIT)

The setting initialization reset (INIT) is the highest level of reset that initializes all settings.

The major settings initialized by the setting initialization reset (INIT) are listed below.

- Device operation mode (bus mode and external bus width settings)
- All of the settings concerning the internal clock (clock source selection, PLL control, and divide ratio settings)
- All of the settings for the external bus CS0 area
- All of the settings concerning other pin states
- All of the sections initialized by the operation initialization reset (RST)

For details on each setting item, see the description of the relevant function.

Note that the setting initialization reset (INIT) must always be performed via the $\overline{\text{INIT}}$ pin immediately after the power is turned on.

● Operation initialization reset (RST)

The operation initialization reset (RST) is the normal level of reset that initializes program operation.

The setting initialization reset (INIT) involves the operation initialization reset (RST).

The major settings initialized by the operation initialization reset (RST) are listed below.

- Program operation
- CPU and internal bus
- Register settings for peripheral circuits
- I/O port settings
- Device operation mode (bus mode and external bus width settings)

For details on each setting item, see the description of the relevant function.

■ Reset Sources

Reset sources which occurred in the past can be identified by reading the RSRR (reset source register).

For information on the registers and flags mentioned below, see "3.11.5 Block Diagram of the Clock Generation Control Unit" and "3.11.6 Registers in the Clock Generation Control Unit".

The following describes each source of a reset and the generated reset level.

● Input to the $\overline{\text{INIT}}$ pin (setting initialization reset pin)

The external $\overline{\text{INIT}}$ pin serves as the setting initialization reset pin.

The setting initialization reset (INIT) request remains issued while the $\overline{\text{INIT}}$ pin maintains low-level input.

High-level input to the $\overline{\text{INIT}}$ pin cancels the INIT request.

When a setting initialization reset (INIT) occurs at the request of this pin, the INIT bit (bit15) in the RSRR (reset source register) is set.

The setting initialization reset (INIT) request via the $\overline{\text{INIT}}$ pin is the highest level of all reset sources, overriding any other input, operation, or state.

Note that the setting initialization reset (INIT) must always be performed via the $\overline{\text{INIT}}$ pin immediately after the power is turned on. Immediately after turning it on, also, keep the "L" level input to the $\overline{\text{INIT}}$ pin for the stabilization wait time required for the oscillator circuit so that the oscillator circuit stabilizes its oscillation within that time. (For INIT via the $\overline{\text{INIT}}$ pin, the oscillation stabilization wait time setting is initialized to the minimum value.)

- Reset source : "L" level input to the external $\overline{\text{INIT}}$ pin
- Cancel source : "H" level input to the external $\overline{\text{INIT}}$ pin
- Reset level : Setting initialization reset (INIT)
- Indication flag : bit15 (INIT bit)

● Writing to the STCR SRST bit (Software reset)

Writing "0" to the SRST bit (bit4) in the STCR (standby control register) generates a software reset request.

The software reset request is an operation initialization reset (RST) request.

When the request is accepted and an operation initialization reset (RST) is generated, the software reset request is canceled.

When an operation initialization reset (RST) is generated by a software reset request, the SRST bit (bit11) in the RSRR (reset source register) is set.

When the SYNCR bit (bit9) in the TBCR (timebase counter control register) has been set (synchronous reset mode), the operation initialization reset (RST) owing to a software reset request occurs only after every bus access is terminated.

Depending on the service states of the buses, therefore, it may take time before the operation initialization reset (RST) can be generated.

- Reset source : Writing "0" to the SRST bit (bit4) in the STCR (standby control register)
- Cancel source : Occurrence of an operation initialization reset (RST)
- Reset level : Operation initialization reset (RST)
- Indication flag : bit11 (SRST bit)

● Watchdog reset

When data is written to the RSRR (watchdog timer control register), the watchdog timer is activated. A watchdog reset request occurs unless A5_H/5A_H writing to the CTBR (timebase counter clear register) is performed within the period set by the WT1 and WT0 bits (bit9 and bit8) in the RSRR.

The watchdog reset request is a setting initialization reset (INIT) request. When the request is accepted and a setting initialization reset (INIT) or operation initialization reset (RST) is generated, the watchdog reset request is canceled.

When a setting initialization reset (INIT) is generated by a watchdog reset request, the WDOG bit (bit13) in the RSRR (reset source register) is set.

Note that the oscillation stabilization wait time setting is not initialized when a setting initialization reset (INIT) is generated by a watchdog reset request. Note also that the oscillation stabilization wait time is not taken unless main oscillation is stopped during a main run or sub-run.

- Reset source : Watchdog timer time-out
- Cancel source : Occurrence of a setting initialization reset (INIT) or operation initialization reset (RST)
- Reset level : Setting initialization reset (INIT)
- Indication flag : bit13 (WDOG bit)

■ Reset Sequence

When a reset source is eliminated, the device starts executing the reset sequence.

The steps performed in the reset sequence vary with the reset level.

The following describes the steps in the reset sequence for each reset level.

● Reset sequence for setting initialization reset (INIT)

When the setting initialization reset (INIT) request is cleared, the device performs the following steps sequentially.

Note, however, that the device does not take the oscillation stabilization wait time in step (2) for a watchdog reset when main oscillation is not stopped during a main run or sub-run.

- (1) Cancels the setting initialization reset (INIT) and enters the oscillation stabilization wait state.
- (2) Maintains the operation initialization reset (RST) state and the internal clock suspended for the oscillation stabilization wait time (set by the OS1 and OS0 bits, or bit3 and bit2 in the STCR).
- (3) Enters the operation initialization reset (RST) state and starts internal clock operation.
- (4) Cancels the operation initialization reset (RST) and enters to the normal operating state.
- (5) Reads a mode vector from address "000FFFF8_H".
- (6) Writes the mode vector to the MODR (mode register).
- (7) Reads a reset vector from address "000FFFFC_H".
- (8) Writes the reset vector to the PC (program counter).
- (9) Starts program operation from the address shown in the PC (program counter).

● Reset sequence for operation initialization reset (RST)

This reset is generated by a software reset.

When the operation initialization reset (RST) request is cleared, the device performs the following steps sequentially.

- (1) Cancels the operation initialization reset (RST) and enters to the normal operating state.
- (2) Reads a mode vector from address "000FFFF8_H".
- (3) Writes the mode vector to the MODR (mode register).
- (4) Reads a reset vector from address "000FFFFC_H".
- (5) Writes the reset vector to the PC (program counter).
- (6) Starts program operation from the address shown in the PC (program counter).

■ Oscillation Stabilization Wait Time

Upon returning from a state in which the device's source oscillation has been (seemingly) suspended, the device enters the oscillation stabilization wait state automatically. This feature prevents the device from using the oscillator output not yet stable after the start of oscillation.

During the oscillation stabilization wait time, the supply of clock signals to the internal and external resources is stopped while only the internal timebase counter operates, waiting until the oscillation stabilization wait time set in the STCR (standby control register) has passed.

The transition to the oscillation stabilization wait state is detailed below.

● Sources of transition to oscillation stabilization wait state

The transition to the oscillation stabilization wait state is caused as follows:

(1) Upon cancellation of a setting initialization reset (INIT)

Immediately after a setting initialization reset (INIT) is canceled by the sources, the device enters the oscillation stabilization wait state.

When the oscillation stabilization wait time has passed, the device then enters the operation initialization reset (RST) state.

Note that, after initialization via the $\overline{\text{INIT}}$ pin, the oscillation stabilization wait time is set to the minimum value and thus the device does not wait for oscillation to become stable. Keep the oscillation stabilization wait time using the input width of the $\overline{\text{INIT}}$ pin.

(2) Upon returning from the stop mode

Immediately after being released from the stop mode, the device enters the oscillation stabilization wait state. If the device is released from it in response to a setting initialization reset (INIT) request, however, the device enters the setting initialization reset (INIT) state instead. The device then enters the oscillation stabilization wait state after the setting initialization reset (INIT) is canceled.

When the oscillation stabilization wait time has passed, the device then enters the state corresponding to the source that canceled the stop mode:

- When returning from stop mode in response to an effective external interrupt request input (including an NMI)
→ The device enters to the normal operating state.
- When returning from stop mode in response to a setting initialization reset (INIT) request
→ The device enters the setting initialization reset (INIT) state.

(3) Upon returning from an abnormal state with the PLL selected

If any PLL control error* occurs when the PLL has been serving as the source clock, the device enters the oscillation stabilization wait state automatically to ensure the PLL lock time.

When the oscillation stabilization wait time has passed, the device then enters to the normal operating state.

*: Examples are an attempt to change the multiplier (multiplication factor) while the PLL is being used and a corruption of the PLL operation enable bit.

● Selecting the oscillation stabilization wait time

The oscillation stabilization wait time is counted by the internal timebase counter.

When an oscillation stabilization wait source occurs and the device enters the oscillation stabilization wait state, the internal timebase counter is initialized and starts measurement of the oscillation stabilization wait time.

The oscillation stabilization wait time setting can be selected from among four options using the OS1 and OS0 bits (bit3 and bit2) in the STCR (standby control register).

Once selected, the oscillation stabilization wait time setting is not initialized unless the setting initialization reset (INIT) via the external $\overline{\text{INIT}}$ pin occurs. Even when a setting initialization reset (INIT) by any other source or an operation initialization reset (RST) occurs, the oscillation stabilization wait time set before the occurrence of such a reset remains in effect.

The following four options are available as the oscillation stabilization wait time settings for their specific cases:

- OS1, OS0 = 00_B : No oscillation stabilization wait time
(The PLL oscillator stops operation with the main oscillation left working in stop mode.)

Note:

For returning from the STOP mode with OSCD1 = 0 when the main PLL clock is being used as the clock source, be sure to set the OS1 and OS0 bits in the STCR to a value other than 00_B to ensure the lock wait time for the main PLL.

- OS1, OS0 = 01_B : PLL lock wait time
(The oscillator does not stop operation with the external clock input or in stop mode.)
- OS1, OS0 = 10_B : Oscillation stabilization wait time (Intermediate)
(A quickly stabilizing type of oscillator is being used, such as a ceramic resonator.)
- OS1, OS0 = 11_B : Oscillation stabilization wait time (Long)
(A general crystal oscillator is being used.)

Note that the setting initialization reset (INIT) via the $\overline{\text{INIT}}$ pin must always be performed immediately after the power is turned on.

In the conditions listed below, also, keep the low-level input to the $\overline{\text{INIT}}$ pin for the oscillation stabilization wait time required for the oscillator circuit so that the oscillator circuit stabilizes its oscillation within that time. (For INIT via the $\overline{\text{INIT}}$ pin, the oscillation stabilization wait time setting is initialized to the minimum value.)

- Immediately after the power is turned on
- In STOP mode with oscillation off
- When main oscillation is off with the sub-clock selected as the clock source

For stable oscillation, therefore, keep the $\overline{\text{INIT}}$ pin input at "L" level for the main clock oscillation stabilization wait time.

■ Reset Operation Modes

The operation initialization reset (RST) has two modes: normal (asynchronous) reset and synchronous reset modes. Either can be selected with the SYNCR bit (bit9) in the TBCR (timebase counter control register).

The reset mode setting is initialized only at a setting initialization reset (INIT).

The setting initialization reset (INIT) is always performed as an asynchronous reset.

The operation in each reset mode is detailed below.

● Normal reset operation

The normal reset operation means the operation of prompt transition to the operation initialization reset (RST) state as soon as an operation initialization reset (RST) request is issued.

On receipt of a reset (RST) request in the normal reset mode, the device enters the reset (RST) state promptly regardless of the operation status of internal bus access.

This mode does not guarantee the result of the bus access being performed at the transition to each state. The mode can however accept the operation initialization reset (RST) request without fail.

The normal reset mode is selected when the SYNCR bit (bit9) in the TBCR (timebase counter control register) contains "0".

The initial value immediately after a setting initialization reset (INIT) occurs selects the normal reset mode.

● Synchronous reset operation

The synchronous reset operation means the operation of transition to the operation initialization reset (RST) state after all bus accesses stop when an operation initialization reset (RST) request is issued.

In the synchronous mode, even though a reset (RST) request is received, the device does not enter the reset (RST) state while any internal bus access is being performed.

When the above request is received, a sleep request is issued to the internal buses. When each internal bus terminates operation to enter the sleep state, the device enters the operation initialization reset (RST) state.

Since all bus accesses stop before the device enters each state in this mode, their results are guaranteed. If any bus access does not stop for some reason, however, the device cannot accept the reset request in that period of time. (Even in this case, the setting initialization reset (INIT) takes effect immediately.)

Bus access won't terminate in the following cases.

- When RDY (ready request) has been input to the external extended bus interface, enabling the bus wait state.

The initial value immediately after a setting initialization reset (INIT) occurs returns the device to the normal reset mode.

Note:

The DMA controller does not delay the transition to each state as it stops transfer on receipt of each request.

The synchronous reset mode is selected when the SYNCR bit (bit9) in the TBCR (timebase counter control register) contains "1".

3.11 Clock Generation Control

The generation and control of each type of clock signal are described below.

The internal operation clock signals for the MB91260B series are generated as follows:

- **Base clock signal:** A base clock signal is generated by frequency-halving the source clock signal or PLL oscillation.
- **Internal clock signals:** Operation clock signals to be supplied to different parts are generated by frequency-dividing the base clock.

For information on the registers and flags mentioned below, see "3.11.5 Block Diagram of the Clock Generation Control Unit" and "3.11.6 Registers in the Clock Generation Control Unit".

■ Selecting the Source Clock Signal

The source clock signal is selected as follows.

The oscillating signal generated by the internal oscillator circuit with an oscillator connected to the external oscillation pins X0 and X1 is used as the source clock signal.

The MB91260B series itself is the source of all clock signals available, including the external bus clock signal.

The main clock signal can be selected arbitrarily during operation between the external oscillation pins and internal oscillator circuit.

- **Main clock signal:** Generated from the X0/X1 pin input to be used as a high-speed clock signal.

The internal base clock signal to be generated is selected from among the following source clock signals:

- Generated by frequency-halving the main clock signal
- Generated by multiplying the main clock signal using a PLL

ϕ is the base clock that is generated from the source clock divided by two or by using the PLL oscillation. Therefore, the system base clock is a clock generated in the above-mentioned internal base clock generation.

The source clock signal is selected by setting the CLKR (clock source control register).

3.11.1 PLL Control

The PLL oscillator circuit for the main clock can be controlled to enable/disable its operation (oscillation) and to set the multiplier (multiplication factor).

The CLKR (clock source control register) is used to control the PLL oscillator circuit for both items.

■ Enabling/disabling PLL Operation

The PLL1EN bit (bit10) in the CLKR (clock source control register) is used to enable or disable the oscillation of the main PLL.

After a setting initialization reset (INIT) occurs, the PLL1EN bit is then initialized to "0", causing the main PLL to stop oscillation. The main PLL output cannot be selected as the source clock signal while the main PLL has its oscillation stopped.

When the device starts program operation, set the multiplier of the main PLL to be used as the clock source, enable the PLL for oscillation, then switch the source clock after the PLL lock wait time has passed. For this PLL lock wait time, it is recommended to use a timebase timer interrupt.

When the main PLL output has been selected as the source clock, the PLL cannot stop operation. An attempt to write to the register is ignored. To stop the PLL, for example, before entering the stop mode, select the frequency-halved main clock signal as the source clock signal before stopping the PLL.

If the OSCD1 bit (bit0) in the STCR (standby control register) has been set to stop oscillation during the stop mode, the PLL stops it automatically when the device enters the stop mode, eliminating the need for setting the stop of oscillation. When the device returns from the stop mode, the PLL restarts oscillation automatically. If the PLL has been set to stop oscillation during the stop mode without stopping main oscillation, the main oscillation does not stop automatically. (For returning from the stop mode then, be sure to keep the PLL lock wait time.) In this case, stop PLL oscillation before entering the stop mode as required.

■ PLL Multiplier

The multiplier (multiplication factor) of the main PLL is set by using the PLL1S2, PLL1S1, and PLL1S0 bits (bit14 to bit12) in the CLKR (clock source control register).

After a setting initialization reset (INIT) occurs, all of these bits are then initialized to "0".

● Setting of PLL multiplier

To change the PLL multiplier setting from the initial value, start program operation and set the new value either before or at the same time as enabling the PLL for oscillation. After changing the multiplier, switch the source clock after the lock wait time has passed. For this PLL lock wait time, it is recommended to use a timebase timer interrupt.

Before changing the PLL multiplier setting during operation, switch the source clock to any resource other than the PLL. After changing the multiplier, switch the source clock after the lock wait time has passed in the same way as above.

The PLL multiplier setting can be changed while the PLL is being used. In this case, however, the device enters the oscillation stabilization wait state automatically after updating the PLL multiplier and stops program operation until the oscillation stabilization wait time has passed. When the clock source is switched to any resource other than the PLL, the device does not stop program operation.

3.11.2 Oscillation Stabilization Wait Time and PLL Lock Wait Time

The oscillation stabilization wait time is required when the clock selected as the source clock is not yet stable in oscillation.

(See "■ Oscillation stabilization wait time" in Section 3.10 "Reset (Device Initialization)".)

The PLL requires the lock wait time until the output stabilizes at the preset frequency after starting oscillation.

■ Wait Times after the Power is Turned on

After the power is turned on, the oscillation stabilization wait time of the oscillator circuit for the main clock is required first.

The oscillation stabilization wait time setting is initialized to be the minimum value by the $\overline{\text{INIT}}$ pin input (setting initialization reset pin). The oscillation stabilization wait time required in this case is kept based on the time for which low-level input to the $\overline{\text{INIT}}$ pin lasts.

Since the PLL is not enabled for oscillation in either state, you do not have to consider the lock wait time in this case.

■ Wait Times after Setting Initialization

When a setting initialization reset (INIT) is canceled, the device enters the oscillation stabilization wait state. The preset oscillation stabilization wait time is generated internally in this state.

In the first oscillation stabilization wait state after $\overline{\text{INIT}}$ pin input, the setting time is initialized to the minimum value. Therefore, the device leaves this state soon and enters the operation initialization reset (RST) state. Note also that the device does not take the oscillation stabilization wait time for a watchdog reset when main oscillation is not stopped during a main run or sub-run.

Since the PLL is not enabled for oscillation in any of these states, you do not have to consider the lock wait time in this case.

■ Wait Times after PLL Operation is Enabled

When the stopped PLL is enabled for oscillation after the start of program operation, the PLL output must not be used until the lock wait time has passed.

If the main PLL has not been selected as the source clock, the device can perform program operation even during the lock wait time. For the PLL lock wait time in this case, it is recommended to use a timebase timer interrupt.

■ Wait Times after the PLL Multiplier is Changed

When the PLL is enabled for oscillation after the start of program operation, the PLL output must not be used until the lock wait time has passed.

If the main PLL has not been selected as the source clock, the device can perform program operation even during the lock wait time. For the PLL lock wait time in this case, it is recommended to use a timebase timer interrupt.

■ Wait Times after Returning from Stop Mode

When returning from the stop mode after the start of program operation, the programmed oscillation stabilization wait time is generated internally.

If the clock oscillator circuit selected as the source clock has been set to stop in the stop mode, the

oscillation stabilization wait time for the oscillator circuit or the used PLL lock wait time, whichever time is longer, is required. Set the longer oscillation stabilization wait time before entering the stop mode.

If the clock oscillator circuit selected as the source clock has been set to operate even in the stop mode, the PLL stops operation. Set the oscillation stabilization wait time to a value other than OS1, OS0 = 0, 0 before entering the stop mode.

3.11.3 Clock Distribution

The operation clock signals for various functions are produced based on the base clock signal generated from the source clock.

The MB91260B series has a total of three types of internal operation clock signals, each of which can be set independently for the frequency divide ratio.

■ CPU Clock Signal (CLKB)

This clock signal is used for the CPU, internal memory, and internal buses.

The circuits using this clock signal include:

- CPU
- Internal RAM and internal ROM
- Bit search module
- I-bus, D-bus, F-bus, and X-bus
- DMA controller
- On-chip Debug Support Unit (DSU)

Do not set a combination of the multiplier and divide ratio which results in a frequency higher than the maximum operating frequency.

■ Peripheral Clock Signal (CLKP)

This clock signal is used for peripheral resources and peripheral buses.

The circuits using this clock signal include:

- Peripheral buses
- Clock control unit (only the bus interface unit)
- Interrupt controller
- I/O ports
- External interrupt inputs, UART, 16-bit timer, and other peripheral resources

Do not set a combination of the multiplier and divide ratio which results in a frequency higher than the maximum operating frequency.

■ External Bus Clock Signal (CLKT)

This clock signal is used for the external extended bus interface.

The circuits using this clock signal include:

- External extended bus interface
- External CLK outputs

Do not set a combination of the multiplier and divide ratio which results in a frequency higher than the maximum operating frequency. However, the MB91260B series does not support the external bus mode.

3.11.4 Clock Frequency Division

For each type of internal operation clock signal, the divide ratio relative to the base clock frequency can be set. This feature allows the optimum operating frequency to be set for each circuit.

The divide ratio is set by the combination of the DIVR0 (base clock frequency division setting register 0) and DIVR1 (base clock frequency division setting register 1). Each of the registers has four setting bits for each type of clock signal. The divide ratio of a clock signal relative to the base clock frequency is expressed as "the register value for that signal + 1". Even though a divide ratio setting is an odd number, the duty ratio is always 50%.

When the divide ratio setting is changed, the new setting takes effect at the next rise of the clock signal.

Even when an operation initialization reset (RST) occurs, the divide ratio setting is not initialized but remains unchanged. It is initialized only at a setting initialization reset (INIT). Before changing the source clock to a faster one from the initial state, be sure to set the divide ratio.

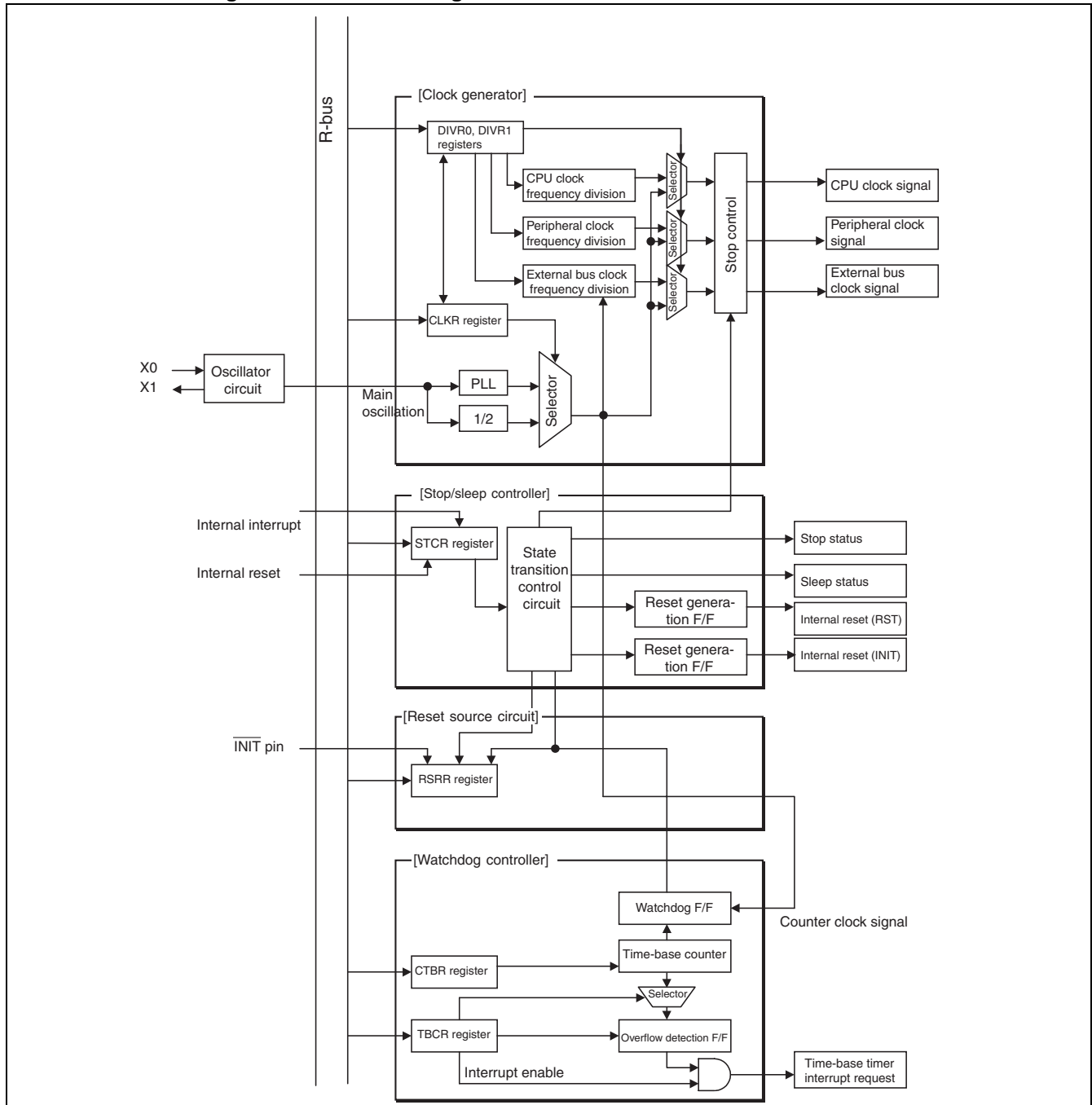
Note:

If you set a combination of the source clock, main PLL multiplier, and divide ratio, which results in a frequency higher than the maximum operating frequency, the operation of the device is not guaranteed. Use meticulous care not to set such a combination of values (in particular, do not select the source clock and change settings in wrong order).

3.11.5 Block Diagram of the Clock Generation Control Unit

This section provides a block diagram of the clock generation control unit. For information on the registers in the diagram, see "3.11.6 Registers in the Clock Generation Control Unit".

Figure 3.11-1 Block Diagram of the Clock Generation Control Unit



The MB91260B series does not support the external bus mode.

3.11.6 Registers in the Clock Generation Control Unit

This section describes the function of the registers to be used in the clock generation control unit.

■ RSRR: Reset Source Register/watchdog Timer Control Register

Bit	15	14	13	12	11	10	9	8
Address: 000480 _H	INIT	—	WDOG	—	SRST	—	WT1	WT0
	R	R	R	R	R	R	R/W	R/W
Initial value ($\overline{\text{INIT}}$ pin)	1	0	0	0	0	0	0	0
Initial value (INIT)	*	*	*	x	x	*	0	0
Initial value (RST)	x	x	x	*	*	x	0	0

*: Variable with the reset source
 x: Not initialized

This register holds the source of the most recently generated reset, sets the time interval for the watchdog timer, and controls its activation.

Read access to this register clears the reset source held there after it is read. If two or more resets occur before it is read, multiple reset source flags are accumulated and set.

Write access to this register activates the watchdog timer. From then on, the watchdog timer keeps on operating until a reset (RST) occurs.

[bit15] INIT (INITialize reset occurred)

This bit indicates whether a reset (INIT) by $\overline{\text{INIT}}$ pin input has occurred.

0	INIT by $\overline{\text{INIT}}$ pin input has not occurred.
1	INIT by $\overline{\text{INIT}}$ pin input has occurred.

- The bit is initialized to "0" immediately after a read.
- A read is possible; a write does not affect the bit value.

[bit14] (reserved bit)

[bit13] WDOG (WatchDOG reset occurred)

This bit indicates whether a reset (INIT) by the watchdog timer has occurred.

0	INIT by watchdog timer has not occurred.
1	INIT by watchdog timer has occurred.

- The bit is initialized to "0" immediately after either a reset (INIT) by $\overline{\text{INIT}}$ pin input or a read.
- A read is possible; a write does not affect the bit value.

[bit12] (reserved bit)

[bit11] SRST (Software ReSeT occurred)

This bit indicates whether a reset (RST) by writing to the SRST bit (software reset) in the STCR register has occurred.

0	RST by software reset has not occurred.
1	RST by software reset has occurred.

- The bit is initialized to "0" immediately after either a reset (INIT) by $\overline{\text{INIT}}$ pin input or a read.
- A read is possible; a write does not affect the bit value.

[bit10] (reserved bit)**[bit9, bit8] WT1, WT0 (Watchdog interval Time select)**

These bits are used to specify the time interval required for the watchdog timer.

The combination of values written to these bits selects the time interval for the watchdog timer from among the four options listed below.

WT1	WT0	Minimum write-to-CTBR interval required for suppressing watchdog reset	Time from last 5AH write to CTBR until watchdog reset
0	0	$\phi \times 2^{16}$ (Initial value)	$\phi \times 2^{16}$ to $\phi \times 2^{17}$
0	1	$\phi \times 2^{18}$	$\phi \times 2^{18}$ to $\phi \times 2^{19}$
1	0	$\phi \times 2^{20}$	$\phi \times 2^{20}$ to $\phi \times 2^{21}$
1	1	$\phi \times 2^{22}$	$\phi \times 2^{22}$ to $\phi \times 2^{23}$

(ϕ represents the internal base clock period.)

- These bits are initialized to "00_B" at a reset (RST)
- A read is possible; a write is valid only once after a reset (RST). Any further write is invalid.

■ STCR: Standby Control Register

Bit	7	6	5	4	3	2	1	0
Address: 000481 _H	STOP	SLEEP	HIZ	SRST	OS1	OS0	–	OSCD1
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value ($\overline{\text{INIT}}$ pin)	0	0	1	1	0	0	1	1
Initial value (INIT)	0	0	1	1	×	×	1	1
Initial value (RST)	0	0	×	1	×	×	×	×

This register controls the operation mode of the device.

The register controls the transition to each of the two standby modes (stop and sleep modes), controls the pin status and oscillation disable mode during the stop mode, sets the oscillation stabilization wait time, and issues a software reset.

Note:

For transition to standby mode, select the synchronous standby mode (using the SYNCS bit (bit8) in the TBCR (timebase counter control register)) and be sure to use the following sequence:

```

/* Writing STCR */
ldi    #_STCR, R0          ; STCR register (0x0481)
ldi    #Val_of_Stby, r1     ; Val_of_Stby is the write data to STCR.
stb     r1, @r0             ; Writing to STCR

/* Writing STBR */
ldi     #_CTBR, r2          ; CTBR register (0x0483)
ldi     #0xA5, r1           ; Clear command (1)
stb     r1, @r2             ; Writing A5 to CTBR
ldi     #0xA5, r1           ; Clear command (2)
stb     r1, @r2             ; Writing A5 to CTBR

/* Clearing the time base counter in here */
ldub    @r0, r1             ; Reading STCR

/* Starting synchronous standby transition */
ldub    @r0, r1             ; Reading dummy STCR
nop                                           ; NOP × 5 for timing adjustment
nop
nop
nop
nop

```

Each bit in the standby control register (STCR) functions as follows:

[bit7] STOP (STOP mode)

This bit selects whether to cause a transition to the stop mode. If "1" is written to both of this bit and the SLEEP bit (bit6), this bit overrides the other, causing a transition to the stop mode.

0	Do not enter stop mode. (Initial value)
1	Enter stop mode.

- The bit is initialized to "0" either at a reset (RST) or when a stop-mode return source is generated.
- A read and a write are possible.

[bit6] SLEEP (SLEEP mode)

This bit selects whether to cause a transition to the sleep mode. If "1" is written to both of this bit and the STOP bit (bit7), the STOP bit (bit7) overrides this bit, causing a transition to the stop mode.

0	Do not enter sleep mode. (Initial value)
1	Enter sleep mode.

- The bit is initialized to "0" either at a reset (RST) or when a sleep-mode return source is generated.
- A read and a write are possible.

[bit5] HIZ (HIZ mode)

This bit controls the pin status during the stop mode.

0	Maintain the pin status existing prior to transition to stop mode.
1	Keep pin output in high impedance state during stop mode. (Initial value)

- The bit is initialized to "0" at a reset (INIT).
- A read and a write are possible.

[bit4] SRST (Software ReSeT)

This bit issues a software reset (RST).

0	Issue software reset.
1	Do not issue software reset. (Initial value)

- The bit is initialized to "1" at a reset (RST).
- A read and a write are possible. Reading the bit always returns "1".

[bit3, bit2] OS1, OS0 (Oscillation Stabilization time select)

These bits are used to specify the oscillation stabilization wait time to be waited for immediately after a reset (INIT) or a return from the stop mode.

The combination of values written to these bits selects the oscillation stabilization wait time from among the four options listed below.

OS1	OS0	Oscillation stabilization wait time	Main oscillation at 4 MHz
0	0	$\phi \times 2^1$ (Initial value)	1.0 [μ s]
0	1	$\phi \times 2^{11}$	1.0 [ms]
1	0	$\phi \times 2^{16}$	32.7 [ms]
1	1	$\phi \times 2^{22}$	2.0 [s]

(ϕ represents the internal base clock period; it is twice the main oscillation.)

- These bits are initialized to "00_B" at a reset (INIT) by $\overline{\text{INIT}}$ pin input.
- A read and a write are possible.

[bit1] (reserved bit)

- This bit is a reserved bit. Always write "1" to this bit.

[bit0] OSCD1 (Oscillation Disable mode for XIN1)

This bit controls the stopping of oscillation at main oscillation inputs (X0, X1) during the stop mode.

0	Do not stop main oscillation but PLL oscillation during stop mode.
1	Stop main oscillation during stop mode. (Initial value)

- This bit is initialized to "1" at a reset (INIT).

- A read and a write are possible.

Note:

For returning from the STOP mode with $OSCD1 = 0$ when the main PLL clock is being used as the clock source, be sure to set the OS1 and OS0 bits in the SPCR to a value other than 00_B to ensure the lock wait time for the main PLL.

■ TBCR: Timebase Counter Control Register

Bit	15	14	13	12	11	10	9	8
Address: 000482 _H	TBIF	TBIE	TBC2	TBC1	TBC0	–	SYNCR	SYNCS
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	×	×	×	×	0	0
Initial value (RST)	0	0	×	×	×	×	×	×

This register controls timebase timer interrupts.

The register enables timebase timer interrupts, selects the interrupt time interval, and sets optional functions for reset operation.

Each bit in the timebase counter control register (TBCR) functions as follows:

[bit15] TBIF (TimeBasetimer Interrupt Flag)

This bit serves as the timebase timer interrupt flag.

The flag indicates that the timebase counter has expired the time interval (set by the TBC2 to TBC0 bits, or bit13 to bit11).

If this bit is set to "1" with the TBIE bit (bit14) enabling interrupts (TBIE = "1"), a timebase timer interrupt request is generated.

Clear source	"0" is written by instruction.
Set source	Specified time interval has passed (falling edge of timebase counter output is detected).

- The bit is initialized to "0" at a reset (RST).
- A read and a write are possible. Note, however, that only "0" can be written. An attempt to write "1" has no effect on the bit value. Note also that reading the bit using a read modify write instruction always returns "1".

[bit14] TBIE (TimeBasetimer Interrupt Enable)

This bit serves as the timebase timer interrupt request output enable bit.

The bit controls interrupt request output when the timebase counter has expired the time interval. When the TBIF bit (bit15) is set to "1" with this bit containing "1", a timebase timer interrupt request occurs.

0	Disable timebase timer interrupt request output. (Initial value)
1	Enable timebase timer interrupt request output.

- The bit is initialized to "0" at a reset (RST).
- A read and a write are possible.

[bit13 to bit11] TBC2 to TBC0 (TimeBasetimer Counting time select)

These bits are used to set the time interval required for the timebase counter used for the timebase timer.

The combination of values written to these bits selects the time interval for the timebase counter from among the eight options listed below.

TBC2	TBC1	TBC0	Timer interval	Oscillation frequency of 4 MHz and PLL multiplier of 8x
0	0	0	$\phi \times 2^{11}$	64 μ s
0	0	1	$\phi \times 2^{12}$	128 μ s
0	1	0	$\phi \times 2^{13}$	256 μ s
0	1	1	$\phi \times 2^{22}$	131 ms
1	0	0	$\phi \times 2^{23}$	262 ms
1	0	1	$\phi \times 2^{24}$	524 ms
1	1	0	$\phi \times 2^{25}$	1048 ms
1	1	1	$\phi \times 2^{26}$	2097 ms

(ϕ represents the internal base clock period, or the main PLL output period.)

- The initial value is indeterminate. Be sure to set a value before enabling interrupts.
- A read and a write are possible.

[bit10] (reserved bit)

This bit is a reserved bit. The value read is indeterminate. A write to this bit has no effect on any function of the register.

[bit9] SYNCR (SYNChronous Reset enable)

This bit serves as the synchronous reset operation enable bit.

The bit selects one of the two types of reset operation to be performed when an operation initialization reset (RST) request or hardware standby request has occurred. One is the normal reset operation for prompt transition to the reset (RST) or hardware standby state as soon as the request is issued. The other is the synchronous reset operation for transition to the operation initialization reset (RST) or hardware standby state after all bus accesses terminate.

0	Normal reset operation (Initial value)
1	Synchronous reset operation

- The bit is initialized to "0" at a reset (INIT).
- A read and a write are possible.

[bit8] SYNCS (SYNChronous Standby enable)

This bit serves as the synchronous standby operation enable bit.

To use the standby mode (sleep mode or stop mode), be sure to set the bit to "1".

0	Normal standby operation (Initial value)
1	Synchronous standby operation

- The bit is initialized to "0" at a reset (INIT).
- A read and a write are possible.

■ CTBR: Timebase Counter Clear Register

Bit	7	6	5	4	3	2	1	0
Address: 000483 _H	D7	D6	D5	D4	D3	D2	D1	D0
	W	W	W	W	W	W	W	W
Initial value (INIT)	×	×	×	×	×	×	×	×
Initial value (RST)	×	×	×	×	×	×	×	×

This register initializes the timebase counter.

When "A5_H" and "5A_H" are written to this register in succession, all the bits in the timebase counter are cleared to "0" immediately after the "5A_H" write. There is no restriction on the interval between "A5_H" write and "5A_H" write. If any data other than "5A_H" is written following the "A5_H" write, however, "A5_H" must be written again before "5A_H" is written in order to clear the timebase counter.

Note, however, that the FF is cleared automatically when the CPU is not operating such as in the stop or sleep mode or during DMA transfer. If such a condition develops, therefore, a watchdog reset is deferred automatically. For details, see the section "3.11.7 Peripheral Circuits in the Clock Control Unit".

The value read from this register is indeterminate.

Note:

Clearing the timebase counter using this register temporarily changes the oscillation stabilization wait time, watchdog timer interval, and timebase timer interval.

■ CLKR: Clock Source Control Register

Bit	15	14	13	12	11	10	9	8
Address: 00000484 _H	—	PLL1S2	PLL1S1	PLL1S0	—	PLL1EN	CLKS1	CLKS0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	0	0	0	0	0	0
Initial value (RST)	×	×	×	×	×	×	×	×

This register selects the clock source as the internal base clock and controls the main PLL.

Use this register to select the clock source, enable or disable main PLL operation, and select the PLL multiplier.

[bit15] (reserved bit)

This bit is a reserved bit. Always write "0" to this bit.

[bit14 to bit12] PLL1S2, PLL1S1, PLL1S0 (PLL1 ratio Select 2 to 0)

These bits are used to select the multiplier of the main PLL.

Select one of the following combinations of values to specify the multiplier of the main PLL.

These bits must not be updated with the main PLL selected as the clock source. Do not set the bits to a multiplier which results in a frequency higher than the maximum operating frequency.

PLL1S2	PLL1S1	PLL1S0	Main PLL multiplier	At main oscillation frequency of 4 MHz
0	0	0	×1 (Equal frequency) *	$\phi = 250$ ns (at 4 MHz)
0	0	1	×2 (Double)	$\phi = 125$ ns (at 8 MHz)
0	1	0	×3 (Treble)	$\phi = 83.33$ ns (at 12 MHz)
0	1	1	×4 (Quadruple)	$\phi = 62.50$ ns (at 16 MHz)
1	0	0	×5 (Quintuple)	$\phi = 50.00$ ns (at 20 MHz)
1	0	1	×6 (Sextuple)	$\phi = 41.67$ ns (at 24 MHz)
1	1	0	×7 (Septuple)	$\phi = 35.71$ ns (at 28 MHz)
1	1	1	×8 (Octuple)	$\phi = 31.25$ ns (at 32 MHz)

*: Please input 8MHz or more at ×1 multiplier.

(ϕ represents the internal base clock period.)

- These bits are initialized to "000_B" at a reset (INIT).
- A read and a write are possible.

[bit11] (reserved bit)

This bit is a reserved bit. Always write "0" to this bit.

[bit10] PLL1EN (PLL1 ENable)

This bit is the main PLL operation enable bit.

It is prohibited to update the bit with the main PLL selected as the clock source. It is also prohibited to select the main PLL as the clock source with this bit containing "0".

(See the settings of the CLKS1 and CLKS0 bits (bit9 and bit8).)

When the OSCD1 bit (bit1) in the STCR contains "1", the main PLL stops operation during the stop mode even with this bit containing "1". The PLL is enabled for operation after the device returns from the stop mode.

0	Stop main PLL. (Initial value)
1	Enable main PLL operation.

- The bit is initialized to "0" at a reset (INIT).
- A read and a write are possible.

[bit9, bit8] CLKS1, CLKS0 (CLOCK source Select)

These bits are used to set the clock source to be used.

The combination of values written to the bits selects one of the following three types of clock source as shown below.

CLKS1	CLKS0	Clock source setting
0	0	Frequency-halved version of oscillation input from X0/X1 (Initial value)
0	1	Setting disabled
1	0	Main PLL
1	1	Setting disabled

- These bits are initialized to "00_B" at a reset (INIT).
- A read and a write are possible.

Notes:

- The value of the CLKS0 bit (bit8) cannot be changed with the CLKS1 bit (bit9) containing "1".

[Combination that can be changed]
"00 _B " → "10 _B "
"10 _B " → "00 _B "

- The bits must not be set to any other combination of values.

■ DIVR0: Base Clock Divide Ratio Setting Register 0

bit	15	14	13	12	11	10	9	8
Address: 000486 _H	B3	B2	B1	B0	P3	P2	P1	P0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	0	0	0	0	1	1
Initial value (RST)	×	×	×	×	×	×	×	×

This register controls the frequency divide ratio for each type of internal clock signal relative to the base clock signal.

The register sets the divide ratios for the CPU/internal bus clock signal (CLKB) and for the peripheral circuit/peripheral bus clock signal (CLKP).

If you set a combination of the source clock, main PLL multiplier, and divide ratio, which results in a frequency higher than the maximum operating frequency, the operation of the device is not guaranteed. Use meticulous care not to set such a combination of values. Also be careful not to select the source clock and change settings in wrong order.

When the divide ratio setting in this register is changed, the new setting takes effect at the next clock rate.

[bit15 to bit12] B3, B2, B1, B0 (clkB divide select 3 to 0)

These bits are used to set the CPU clock (CLKB) frequency divide ratio.

The clock frequency divide ratio set by these bits applies to the clock signal (CLKB) for the CPU, internal

memory, and internal buses.

The combination of values written to these bits selects the divide ratio (clock frequency) for the CPU/ internal bus clock signal relative to the base clock signal, from among the 16 types listed below.

Do not set the bits to a divide ratio which results in a frequency higher than the maximum operating frequency.

B3	B2	B1	B0	Clock divide ratio	Clock frequency: Oscillation frequency of 4 MHz and main PLL multiplier of 8×
0	0	0	0	ϕ	32 MHz (Initial value)
0	0	0	1	$\phi \times 2$ (Divide by 2)	16 MHz
0	0	1	0	$\phi \times 3$ (Divide by 3)	10.7 MHz
0	0	1	1	$\phi \times 4$ (Divide by 4)	8 MHz
0	1	0	0	$\phi \times 5$ (Divide by 5)	6.4 MHz
0	1	0	1	$\phi \times 6$ (Divide by 6)	5.33 MHz
0	1	1	0	$\phi \times 7$ (Divide by 7)	4.57 MHz
0	1	1	1	$\phi \times 8$ (Divide by 8)	4 MHz
...
1	1	1	1	$\phi \times 16$ (Divide by 16)	2 MHz

(ϕ represents the internal base clock period.)

- These bits are initialized to "0000_B" at a reset (INIT).
- A read and a write are possible.

[bit11 to bit8] P3, P2, P1, P0 (clkP divide select 3 to 0)

These bits are used to set the peripheral clock (CLKP) frequency divide ratio.

The clock frequency divide ratio set by these bits applies to the clock signal (CLKP) for peripheral circuits and peripheral buses.

The combination of values written to these bits selects the divide ratio (clock frequency) for the peripheral circuit/peripheral bus clock signal relative to the base clock signal, from among the 16 types listed below.

Do not set the bits to a divide ratio which results in a frequency higher than the maximum operating frequency.

P3	P2	P1	P0	Clock divide ratio	Clock frequency: Oscillation frequency of 4 MHz and main PLL multiplier of 8×
0	0	0	0	ϕ	32 MHz
0	0	0	1	$\phi \times 2$ (Divide by 2)	16 MHz
0	0	1	0	$\phi \times 3$ (Divide by 3)	10.7 MHz
0	0	1	1	$\phi \times 4$ (Divide by 4)	8 MHz (Initial value)
0	1	0	0	$\phi \times 5$ (Divide by 5)	6.4 MHz
0	1	0	1	$\phi \times 6$ (Divide by 6)	5.33 MHz
0	1	1	0	$\phi \times 7$ (Divide by 7)	4.57 MHz
0	1	1	1	$\phi \times 8$ (Divide by 8)	4 MHz
...
1	1	1	1	$\phi \times 16$ (Divide by 16)	2 MHz

(ϕ represents the system base clock period.)

- These bits are initialized to "0011_B" at a reset (INIT).
- A read and a write are possible.

■ DIVR1: Base Clock Divide Ratio Setting Register 1

Bit	7	6	5	4	3	2	1	0
Address: 000487 _H	T3	T2	T1	T0	—	—	—	—
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value (INIT)	0	0	0	0	0	0	0	0
Initial value (RST)	×	×	×	×	×	×	×	×

This register controls the frequency divide ratio for each type of internal clock signal relative to the base clock signal.

The register sets the divide ratio for the external extended bus interface clock signal (CLKT).

If you set a combination of the source clock, main PLL multiplier, and divide ratio, which results in a frequency higher than the maximum operating frequency, the operation of the device is not guaranteed. Use meticulous care not to set such a combination of values. Also be careful not to select the source clock and change settings in wrong order.

When the divide ratio setting in this register is changed, the new setting takes effect at the next clock rate.

[bit7 to bit4] T3, T2, T1, T0 (clkT divide select 3 to 0)

These bits are used to set the external bus clock (CLKT) frequency divide ratio.

The clock frequency divide ratio set by these bits applies to the clock signal (CLKT) for the external bus interface.

The combination of values written to these bits selects the divide ratio (clock frequency) for the external extended bus interface relative to the base clock signal, from among the 16 types listed below.

Do not set the bits to a divide ratio which results in a frequency higher than the maximum operating frequency.

T3	T2	T1	T0	Clock divide ratio	Clock frequency: Oscillation frequency of 4 MHz and main PLL multiplier of 8×
0	0	0	0	ϕ	32 MHz (Initial value)
0	0	0	1	$\phi \times 2$ (Divide by 2)	16 MHz
0	0	1	0	$\phi \times 3$ (Divide by 3)	10.7 MHz
0	0	1	1	$\phi \times 4$ (Divide by 4)	8 MHz
0	1	0	0	$\phi \times 5$ (Divide by 5)	6.4 MHz
0	1	0	1	$\phi \times 6$ (Divide by 6)	5.33 MHz
0	1	1	0	$\phi \times 7$ (Divide by 7)	4.57 MHz
0	1	1	1	$\phi \times 8$ (Divide by 8)	4 MHz
...
1	1	1	1	$\phi \times 16$ (Divide by 16)	2 MHz

(ϕ represents the system base clock period.)

- These bits are initialized to "0000_B" at a reset (INIT).
- A read and a write are possible.

It is advisable to set the bits to "1111_B" (divide by 16) because the external bus interface is not used by the MB91260B series.

Note:

The MB91260B series does not support the external bus mode.

[bit3 to bit0] (reserved bit)

- These bits are initialized to "0000_B" at a reset (INIT).
- Always write "0000_B" to these bits.

3.11.7 Peripheral Circuits in the Clock Control Unit

This section describes peripheral circuits in the clock control unit.

■ Timebase Counter

The clock control unit incorporates a 26-bit timebase counter operating with the internal base clock.

The timebase counter is used for measurement of the oscillation stabilization wait time (detailed in "■ Oscillation stabilization wait time" in Section 3.10 "Reset (Device Initialization)") and for the following applications as well:

- Watchdog timer
The watchdog timer for detecting system runaway performs measurement using the bit output of the timebase counter.
- Timebase timer
The timebase timer generates an interval interrupt using the timebase counter output.

These functions are described below:

● Watchdog timer

The watchdog timer is a runaway detection timer using the timebase counter output. If a program runs out of control, preventing the watchdog reset defer function from being executed within the set interval, for example, the watchdog timer generates a setting initialization reset (INIT) request for a watchdog reset.

[Activating the watchdog timer and setting its time interval]

The watchdog timer is activated upon the first write to the RSRR (reset source register/watchdog timer control register) after a reset (RST). At this time, the WT1 and WT0 bits (bit9 and bit8) are used to set the time interval for the watchdog timer. Only the time interval set at the first write to the RSRR takes effect; the settings made at any further writes are ignored.

[Deferring the generation of a watchdog reset]

Once the watchdog timer is activated, "A5_H" and "5A_H" must be written, in this order, periodically to the CTBR (timebase counter clear register) by a program. This initializes the watchdog reset generation flag.

[Generating a watchdog reset]

The watchdog reset generation flag is set at the falling edge of the timebase counter output during the set interval. If the flag has been set when the second falling edge is detected, a setting initialization reset (INIT) request is generated for a watchdog reset.

[Stopping the watchdog timer]

Once the watchdog timer is activated, it cannot be stopped until an operation initialization reset (RST) occurs.

In the following states in which an operation initialization reset (RST) occurs, the watchdog timer stops and will not function until it is programmed to be reactivated.

- Operation initialization reset (RST) state
- Setting initialization reset (INIT) state
- Oscillation stabilization wait reset (RST) state

[Suspending the watchdog timer (deferring automatic generation)]

The watchdog timer initializes the watchdog reset generation flag to defer the generation of a watchdog reset when the CPU is suspending program operation. Put briefly, the CPU suspends program operation in the following states:

- Sleep state
- Stop state
- Oscillation stabilization wait RUN state
- During DMA transfer to the D-bus (data bus)
- During a break when the emulator debugger is being used

Note that, when the timebase counter is cleared, the watchdog reset generation flag is also initialized at the same time, deferring the generation of a watchdog reset.

● Timebase timer

The timebase timer generates an interval interrupt using the timebase counter output. This timer is suitable for applications which involve time measurement for relatively long time up to {base clock signal $\times 2^{27}$ } cycles, such as the main PLL lock wait time or subclock oscillation stabilization wait time.

The timebase timer generates a timebase timer interrupt request upon detection of that falling edge of the timebase counter output which corresponds to the set interval.

[Activating the timebase timer and setting its time interval]

The time interval for the timebase timer is set by the TBC2, TBC1, and TBC0 bits (bit13 to bit11) in the TBCR (timebase counter control register).

Since that falling edge of the timebase counter output which corresponds to the set interval is always detected, clear the TBIF bit (bit15) first after setting the time interval and then set the TBIE bit (bit14) to "1" to enable the interrupt request output.

Before changing the time interval, set the TBIE bit (bit14) to "0" to disable the interrupt request output.

The timebase counter keeps counting without being affected by these settings. To obtain the accurate interval interrupt time, therefore, clear the timebase counter before enabling interrupts. Otherwise, an interrupt request may occur immediately after interrupts are enabled.

[Clearing the timebase counter using a program]

When "A5_H" and "5A_H" are written, in this order, to the CTBR (timebase counter clear register), all the bits in the timebase counter are cleared to "0" immediately after the "5A_H" write. There is no restriction on the interval between "A5_H" write and "5A_H" write. If any data other than "5A_H" is written following the "A5_H" write, however, "A5_H" must be written again before "5A_H" is written in order to clear the timebase counter.

When the timebase counter is cleared in this way, the watchdog reset generation flag is initialized at the same time, temporarily deferring the generation of a watchdog reset.

[Clearing the timebase counter by device state]

All the bits in the timebase counter are cleared to "0" when the device causes a transition to the following states:

- Stop state
- Setting initialization reset (INIT) state
- Hardware standby state

When the device enters the stop state, in particular, the timebase timer may cause an unintentional interval interrupt as the timebase counter is used for measurement of oscillation stabilization wait time. Before setting the stop mode, therefore, disable timebase timer interrupts and stop using the timebase timer.

When the device enters any other state, an operation initialization reset (RST) occurs and thus timebase timer interrupts are disabled automatically.

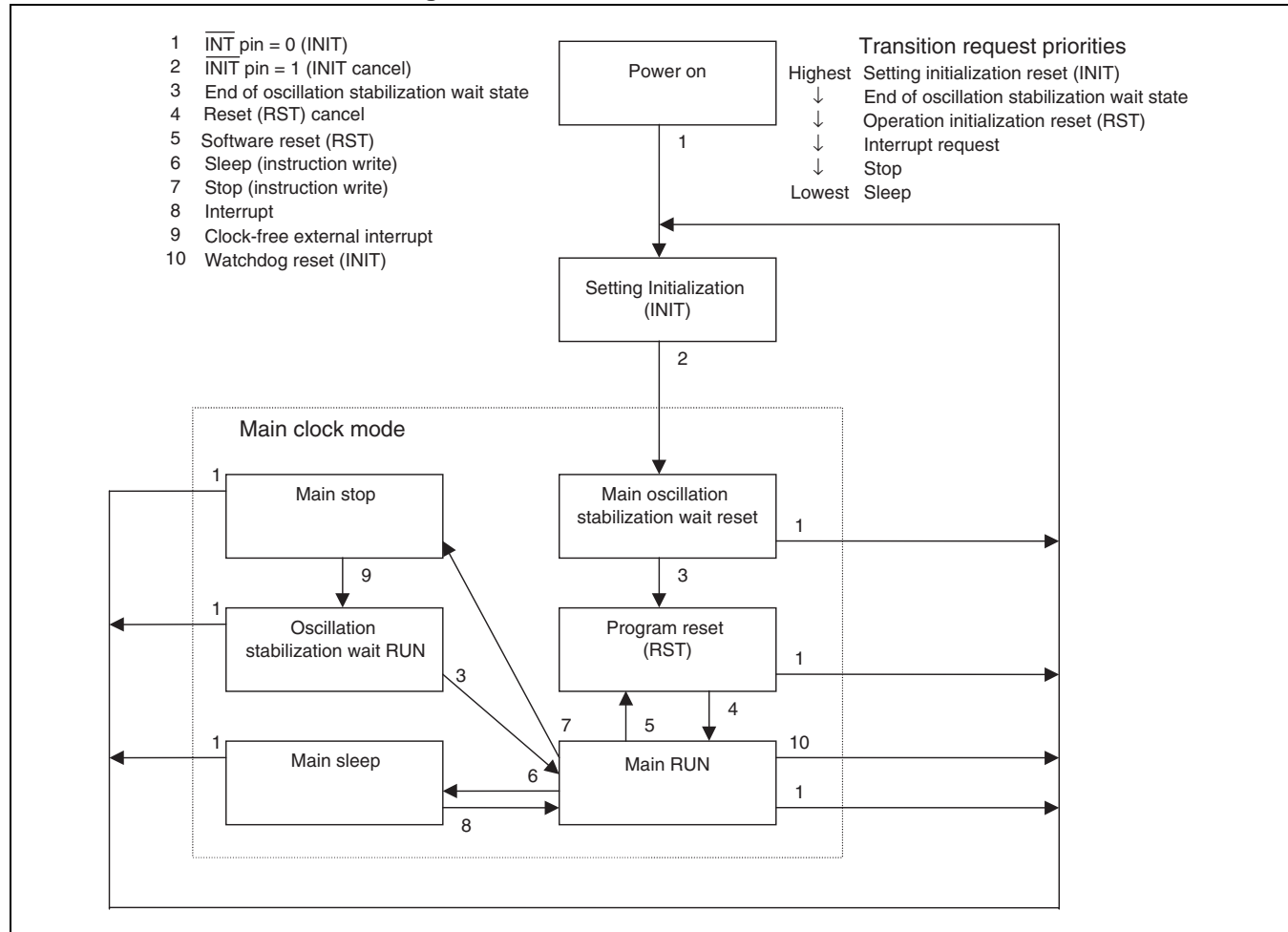
3.12 Device Status Control

This section describes each status of the MB91260B series and its control.

■ Device States and State Transitions

The MB91260B series takes transitions as illustrated below:

Figure 3.12-1 Transition of Device Status



The MB91260B series has the following operating states.

● RUN state (normal operating state)

This state is the program execution state.

In this state, all of the circuits can operate with all the internal clock signals supplied. Note, however, that only the 16-bit peripheral bus has only the bus clock stopped when no access is being made.

Although each state transition request is acceptable, state transitions in response to certain requests in the synchronous reset mode are different from those in the normal reset mode. For details, see "●Synchronous reset operation" in Section 3.10 "Reset (Device Initialization)".

● Sleep state

This state is the program idle state. The device enters the state by program operation.

Only the CPU stops program execution while peripheral circuits can operate. Various built-in memory modules and internal/external buses remain idle until requested by the DMA controller.

When a valid interrupt request occurs, the device is released from this state and enters the RUN state (normal operating state).

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state.

When an operation initialization reset (RST) request occurs, the device enters the operation initialization reset (RST) state.

● Stop state

This state is a device idle state. The device enters the state by program operation.

All of the internal circuits stop operation. All of the internal clock signals are stopped. The oscillator circuit and main PLL can be stopped by making the settings. It is also possible to place external pins (with some exceptions) in a high impedance state by making the settings.

When a specific (clock-free) valid interrupt request occurs, the device enters the oscillation stabilization wait RUN state.

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state.

When an operation initialization reset (RST) request occurs, the device enters the oscillation stabilization wait reset (RST) state.

● Oscillation stabilization wait RUN state

This state is a device idle state. The device enters the state when returning from the stop state.

All of the internal circuits stop operation, excluding the clock generation control unit (timebase counter and device status control unit). Although all of the internal clock signals are stopped, the oscillator circuit and the enabled main PLL are working.

This state cancels high-impedance control applied to external pins in the stop state.

The device enters the RUN state (normal operating state) as the set oscillation stabilization wait time has passed.

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state.

When an operation initialization reset (RST) request occurs, the device enters the oscillation stabilization wait reset (RST) state.

● Oscillation stabilization wait reset (RST) state

This state is a device idle state. The device enters the state when returning from the stop state or setting initialization reset (INIT) state.

All of the internal circuits stop operation, excluding the clock generation control unit (timebase counter and device status control unit). Although all of the internal clock signals are stopped, the oscillator circuit and the enabled main PLL are working.

This state cancels high-impedance control applied to external pins in the stop state.

An operation initialization reset (RST) is output to the internal circuitry.

The device enters the oscillation stabilization wait reset (RST) state as the set oscillation stabilization wait time has passed.

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state.

● Operation initialization reset (RST) state

This state is the program initialized state. The device enters the state either on receipt of an operation initialization reset (RST) request or upon termination of the oscillation stabilization wait reset (RST) state.

The CPU stops program execution and the program counter is initialized. Most of the peripheral circuits are initialized. All of the internal clocks, oscillator circuit, and the enabled main PLL are working.

An operation initialization reset (RST) is output to the internal circuitry.

When the operation initialization reset (RST) request is eliminated, the device enters the RUN state (normal operating state) and executes the operation initialization reset sequence. When returning from the setting initialization reset (INIT) state, the device executes the setting initialization reset sequence instead.

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state.

● Setting initialization reset (INIT) state

This state is the state in which all settings are initialized. The device enters the state upon receipt of a setting initialization reset (INIT) request.

The CPU stops program execution and the program counter is initialized. All of the peripheral circuits are initialized. The oscillator circuit operates while the main PLL stops operation. All of the internal clocks operate except when the external $\overline{\text{INIT}}$ pin maintains "L" level input.

A setting initialization reset (INIT) and an operation initialization reset (RST) are output to the internal circuitry.

When the setting initialization reset (INIT) request is eliminated, the device is released from this state and enters the oscillation stabilization wait reset (RST) state. The device then enters the operation initialization reset (RST) state and executes the setting initialization reset sequence.

● Priorities of state transition requests

State transition requests follow their priorities given as shown below regardless of the current state. Note, however, that some requests are generated only under specific conditions; they are made valid only in such conditions.

[Highest]	Setting initialization reset (INIT) request
↓	End of oscillation stabilization wait time (Occurs only in the oscillation stabilization wait reset or oscillation stabilization wait RUN state.)
↓	Operation initialization reset (RST) request
↓	Valid interrupt request (Occurs only in the RUN, sleep, or stop state.)
↓	Stop mode request (register write) (Occurs only in the RUN state.)
[Lowest]	Sleep mode request (register write) (Occurs only in the RUN state.)

■ Low-power Consumption Modes

This section describes the individual low-power consumption modes of the MB91260B series and explains how to use them.

The MB91260B series has the following low-power consumption modes:

- Sleep mode
The device enters the sleep state in response to a register write.
- Stop mode
The device enters the stop state in response to a register write.

Each of these modes is detailed below.

● Sleep mode

Writing "1" to the SLEEP bit (bit6) in the STCR (standby control register) establishes the sleep mode and causes the device to enter the sleep state. The device remains in the sleep state until a sleep-state return source (an event that triggers the device to return from the sleep state) is generated.

For information about the sleep state, see "● Sleep state" of the section 3.12 "Device Status Control" as well.

[Transition to sleep mode]

For transition to sleep mode, select the synchronous standby mode (using the SYNCs bit (bit8) in the TBCR (timebase counter control register)) and be sure to use the following sequence:

```
/* Writing STCR */
ldi    #_STCR, R0          ; STCR register (0x0481)
ldi    #Val_of_Stby, r1     ; Val_of_Stby is the write data to STCR.
stb     r1, @r0             ; Writing to STCR

/* Writing STBR */
ldi     #_CTBR, r2          ; CTBR register (0x0483)
ldi     #0xA5, r1           ; Clear command (1)
stb     r1, @r2             ; Writing A5 to CTBR
ldi     #0xA5, r1           ; Clear command (2)
stb     r1, @r2             ; Writing A5 to CTBR

/* Clearing the time base counter in here */
ldub    @r0, r1             ; Reading STCR

/* Starting synchronous standby transition */
ldub    @r0, r1             ; Reading dummy STCR
nop                                           ; NOP × 5 for timing adjustment
nop
nop
nop
nop
```

If "1" is written to both of this bit and the STOP bit (bit7) in the STCR (standby control register), the STOP bit (bit7) overrides the other, causing the transition to the stop state.

[Circuits which stop in the sleep state]

CPU during program execution

The following circuits work when DMA transfer occurs:

- Bit search module
- Built-in memory modules
- Internal/external buses

However, the MB91260B series does not support the external bus mode.

[Circuits which do not stop in the sleep state]

- Oscillator circuit
- Operation-enabled main PLL
- Clock generation control unit
- Interrupt controller
- Peripheral circuits
- Watch timer
- Main oscillation stabilization wait timer
- DMA controller
- On chip Debug Support Unit (DSU)

[Sleep-state return sources]

- Occurrence of a valid interrupt request

When a request for an interrupt not disabled (1111_B) in the ICR register occurs, the sleep mode is canceled and the device enters the RUN state (normal operating state). In this case, set the I-flag in the PS register of the CPU to "1" to enable interrupts and execute the interrupt handler after returning from the sleep state.

The sleep mode is not canceled even when a request for an interrupt disabled (1111_B) in the ICR register occurs.

- Occurrence of a setting initialization reset (INIT) request

When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state unconditionally.

- Occurrence of an operation initialization reset (RST) request

When an operation initialization reset (RST) request occurs, the device enters the operation initialization reset (RST) state unconditionally.

For the priority of each type of source, see "● Priorities of state transition requests" of the section 3.12 "Device Status Control".

[Synchronous standby operation]

When the SYNC_S bit (bit8) in the TBCR (timebase counter control register) contains "1", the synchronous standby operation is enabled. In this case, the device does not enter the sleep state only by writing to the SLEEP bit. It enters the sleep state by reading the STCR register after that.

To enter the sleep mode, be sure to use the sequence given in [Transition to sleep mode].

● Stop mode

Writing "1" to the STOP bit (bit7) in the STCR (standby control register) establishes the stop mode, causing the transition to the stop state. The device remains in the stop state until a stop-state return source (an event that triggers the device to return from the stop state) is generated.

For information about the stop state, see "● Stop state" of the section 3.12 "Device Status Control" as well.

[Transition to stop mode]

For transition to stop mode, select the synchronous standby mode (using the SYNCNS bit (bit8) in the TBCR (timebase counter control register)) and be sure to use the following sequence:

```
/* Writing STCR */
ldi    #_STCR, R0            ; STCR register (0x0481)
ldi    #Val_of_Stby, r1      ; Val_of_Stby is the write data to STCR.
stb     r1, @r0              ; Writing to STCR

/* Writing STBR */
ldi    #_CTBR, r2            ; CTBR register (0x0483)
ldi    #0xA5, r1             ; Clear command (1)
stb     r1, @r2              ; Writing A5 to CTBR
ldi    #0xA5, r1             ; Clear command (2)
stb     r1, @r2              ; Writing A5 to CTBR

/* Clearing the time base counter in here */
ldub    @r0, r1              ; Reading STCR

/* Starting synchronous standby transition */
ldub    @r0, r1              ; Reading dummy STCR
nop                                           ; NOP × 5 for timing adjustment
nop
nop
nop
nop
```

If "1" is written to both of this bit and the SLEEP bit (bit6) in the STCR (standby control register), the STOP bit (bit7) overrides the other, causing the transition to the stop state.

[Circuits which stop in the stop state]

All of the internal circuits except the following:

[Circuits which do not stop in the stop state]

- Oscillator circuit not set to stop operation
When the OSCD1 bit (bit0) in the STCR (standby control register) contains "0", the main clock oscillator circuit does not stop operation even in the stop state.
- Main oscillator circuit connected to the operation-enabled oscillator circuit not set to stop operation
When the OSCD1 bit (bit0) in the STCR (standby control register) contains "0" and the PLL1EN bit (bit10) in the CLKR (clock source control register) contains "1", the main oscillator circuit does not stop operation even in the stop state but the PLL oscillator circuit stops operation.

[High-impedance control of pins in the stop state]

When the HIZ bit (bit5) in the STCR (standby control register) contains "1", the outputs of some pins in the stop state remain in a high impedance state. For the pins to be subjected to this control, see Appendix C "Pin Status in Each CPU State".

When the HIZ bit (bit5) in the STCR (standby control register) contains "0", the outputs of some pins in the stop state maintain the values existing prior to the transition to the stop state. For details, see APPENDIX C "Pin Status In Each CPU State".

[Stop-state return sources]

- Occurrence of a specific (clock-free) valid interrupt request
Only some external interrupt and NMI input pins are valid.
When a request for an interrupt not disabled (1111_B) in the ICR register occurs, the stop mode is canceled and the device enters the oscillation stabilization wait RUN state. In this case, set the I-flag in the PS register of the CPU to "1" to enable interrupts and execute the interrupt handler after returning from the stop state.
The stop mode is not canceled even when a request for an interrupt disabled (1111_B) in the ICR register occurs.
- Occurrence of a setting initialization reset (INIT) request
When a setting initialization reset (INIT) request occurs, the device enters the setting initialization reset (INIT) state unconditionally.
- Occurrence of an operation initialization reset (RST) request
When an operation initialization reset (RST) request occurs, the device enters the operation initialization reset (RST) state unconditionally.

For the priority of each type of source, see "● Priorities of state transition requests" of the section 3.12 "Device Status Control".

[Selecting the clock source for the stop mode]

In self-oscillation mode, select the frequency-halved version of the main clock signal as the source clock signal before setting the stop mode. For details, see Section "3.11.1 PLL Control" therein.

Note that the restrictions on the divide ratio setting are the same as those for normal operation.

[Synchronous standby operation]

When the SYNC_{CS} bit (bit8) in the TBCR (timebase counter control register) contains "1", the synchronous standby operation is enabled. In this case, the device does not enter the stop state only by writing to the STOP bit. It enters the stop state by reading the STCR register after that.

To enter the stop mode, be sure to use the sequence given in [Transition to stop mode].

CHAPTER 4

I/O PORTS

This chapter outlines the I/O ports and describes the configuration and functions of their registers.

- 4.1 Overview of I/O Ports
- 4.2 Registers of I/O Port
- 4.3 Analog Input Ports

4.1 Overview of I/O Ports

This LSI can use its pins as I/O ports when they are set not to serve for input to or output from their respective external bus interfaces or peripherals.

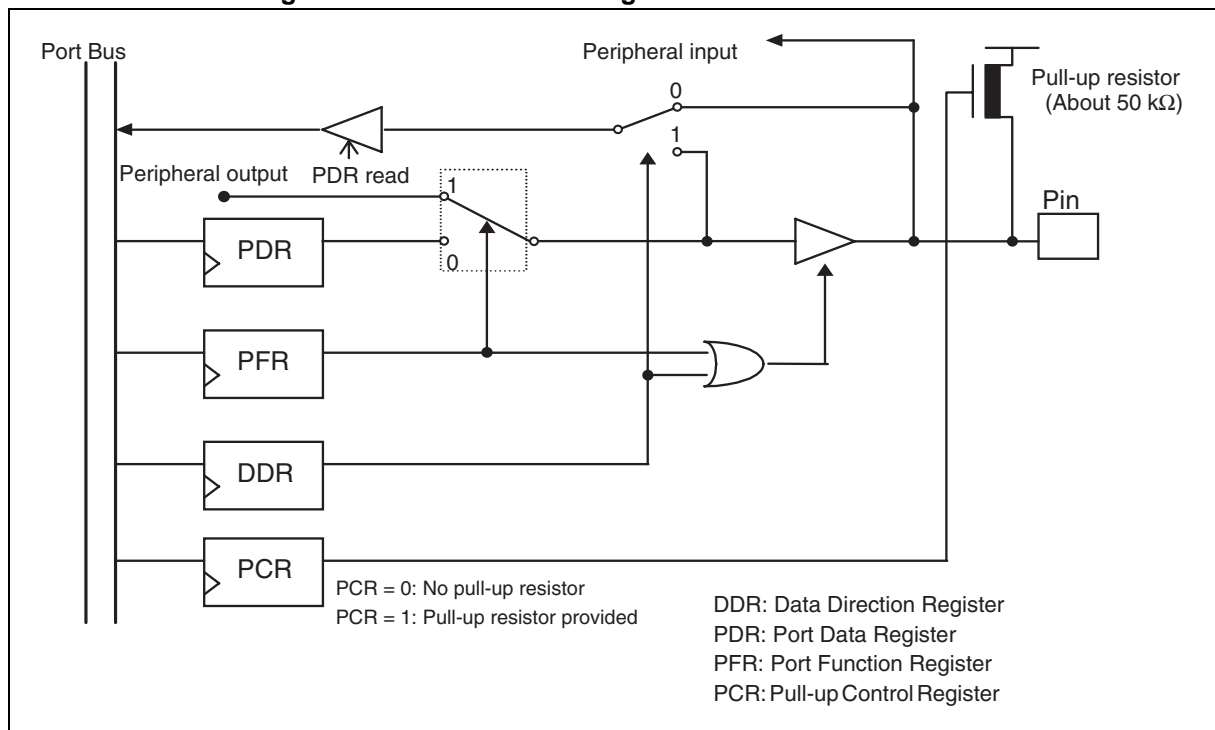
■ Basic Block Diagram of I/O Ports

An I/O port with pull-up resistor comprises the following registers:

- PDR (Port Data Register)
- DDR (Data Direction Register)
- PFR (Port Function Register)
- PCR (Pull-up Control Register)

The following diagram shows a basic configuration of an I/O port.

Figure 4.1-1 Basic Block Diagram of I/O Ports



■ Mode for I/O Ports

- In port input mode (PFR=0 & DDR=0)

PDR read: The level at the corresponding external pin is read.

PDR write: A setting value is written to the PDR.

- In port output mode (PFR=0 & DDR=1)

PDR read: The PDR value is read.

PDR write: The PDR value is output to the corresponding external pin.

● In peripheral output mode (PFR=1 & DDR=x)

PDR read: The output value from the corresponding peripheral is read.

PDR write: A setting value is written to the PDR.

Notes:

- Access each port in bytes.
 - In stop mode (HIZ=0), the setting in the pull-up resistor control register is preferential.
 - In stop mode (HIZ=1), the setting in the pull-up resistor control register is invalid.
-

4.2 Registers of I/O Port

This section explains the configuration and the function of registers which are used in I/O ports.

■ Port Data Registers (PDR:PDR0 to PDR7, PDRC, PDRD, PDRE and PDRG)

PDR0 Address: 00000000 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P07	P06	P05	P04	P03	P02	P01	P00		
PDR1 Address: 00000001 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P17	P16	P15	P14	P13	P12	P11	P10		
PDR2 Address: 00000002 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P27	P26	P25	P24	P23	P22	P21	P20		
PDR3 Address: 00000003 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P37	P36	P35	P34	P33	P32	P31	P30		
PDR4 Address: 00000004 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P47	P46	P45	P44	P43	P42	P41	P40		
PDR5 Address: 00000005 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P57	P56	P55	P54	P53	P52	P51	P50		
PDR6 Address: 00000006 _H	7	6	5	4	3	2	1	0	Initial value ----X X X X _B	Access R/W
	–	–	–	–	P63	P62	P61	P60		
PDR7 Address: 00000007 _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	P77	P76	P75	P74	P73	P72	P71	P70		
PDRC Address: 0000000C _H	7	6	5	4	3	2	1	0	Initial value XXXXXXXX _B	Access R/W
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0		
PDRD Address: 0000000D _H	7	6	5	4	3	2	1	0	Initial value -----X X _B	Access R/W
	–	–	–	–	–	–	PD1	PD0		
PDRE Address: 0000000E _H	7	6	5	4	3	2	1	0	Initial value -----X X _B	Access R/W
	–	–	–	–	–	–	PE1	PE0		
PDRG Address: 00000010 _H	7	6	5	4	3	2	1	0	Initial value --XXXXXXXX _B	Access R/W
	–	–	PG5	PG4	PG3	PG2	PG1	PG0		

PDR0 to PDR7, PDRC, PDRD, PDRE and PDRG are I/O data registers of the I/O port. These are controlled for input/output by their respective DDR0 to DDR7, DDRC, DDRD, DDRE, DDRG, PFR0 to PFR2, PFR7 and PFRG.

■ Data Direction Registers (DDR:DDR0 to DDR7, DDRC, DDRD, DDRE and DDRG)

DDR0	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000400 _H	P07	P06	P05	P04	P03	P02	P01	P00	00000000 _B	R/W
DDR1	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000401 _H	P17	P16	P15	P14	P13	P12	P11	P10	00000000 _B	R/W
DDR2	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000402 _H	P27	P26	P25	P24	P23	P22	P21	P20	00000000 _B	R/W
DDR3	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000403 _H	P37	P36	P35	P34	P33	P32	P31	P30	00000000 _B	R/W
DDR4	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000404 _H	P47	P46	P45	P44	P43	P42	P41	P40	00000000 _B	R/W
DDR5	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000405 _H	P57	P56	P55	P54	P53	P52	P51	P50	00000000 _B	R/W
DDR6	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000406 _H	–	–	–	–	P63	P62	P61	P60	----0000 _B	R/W
DDR7	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000407 _H	P77	P76	P75	P74	P73	P72	P71	P70	00000000 _B	R/W
DDRC	7	6	5	4	3	2	1	0	Initial value	Access
Address: 0000040C _H	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	00000000 _B	R/W
DDRD	7	6	5	4	3	2	1	0	Initial value	Access
Address: 0000040D _H	–	–	–	–	–	–	PD1	PD0	-----00 _B	R/W
DDRE	7	6	5	4	3	2	1	0	Initial value	Access
Address: 0000040E _H	–	–	–	–	–	–	PE1	PE0	-----00 _B	R/W
DDRG	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000410 _H	–	–	PG5	PG4	PG3	PG2	PG1	PG0	--000000 _B	R/W

DDR0 to DDR7, DDRC, DDRD, DDRE and DDRG control the I/O direction of each bit in the corresponding I/O port.

When PFR = 0 DDR = 0: Port input

DDR = 1: Port output

When PFR = 1 DDR = 0: Peripheral input

DDR = 1: Peripheral output

■ Pull-up Control Registers (PCR:PCR0 to PCR7 and PCRG)

PCR0	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000600 _H	P07	P06	P05	P04	P03	P02	P01	P00	00000000 _B	R/W
PCR1	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000601 _H	P17	P16	P15	P14	P13	P12	P11	P10	00000000 _B	R/W
PCR2	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000602 _H	P27	P26	P25	P24	P23	P22	P21	P20	00000000 _B	R/W
PCR3	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000603 _H	P37	P36	—	—	—	—	—	—	00----- _B	R/W
PCR4	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000604 _H	P47	P46	P45	P44	P43	P42	P41	P40	00000000 _B	R/W
PCR5	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000605 _H	P57	P56	P55	P54	P53	P52	P51	P50	00000000 _B	R/W
PCR6	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000606 _H	—	—	—	—	P63	P62	P61	P60	----0000 _B	R/W
PCR7	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000607 _H	P77	P76	P75	P74	P73	P72	P71	P70	00000000 _B	R/W
PCRG	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000610 _H	—	—	PG5	PG4	PG3	PG2	PG1	PG0	--000000 _B	R/W

PCR0 to PCR7 and PCRG control the pull-up resistor for the corresponding I/O port.

PCR = 0: No pull-up resistor

PCR = 1: Pull-up resistor provided

P30 to P35, PC0 to PC7, PD0, PD1, PE0 and PE1 have no pull-up resistor.

■ Port Function Registers (PFR:PFR0 to PFR2, PFR7 and PFRG)

PFR0	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000420 _H	PPG8E	PPG7E	PPG6E	PPG5E	PPG4E	PPG3E	PPG2E	PPG1E	00000000 _B	R/W
PFR1	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000421 _H	–	PPG15E	PPG14E	PPG13E	PPG12E	PPG11E	PPG10E	PPG9E	–0000000 _B	R/W
PFR2	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000422 _H	–	–	SCK1E	SO1E	–	SCK0E	SO0E	–	--00-00- _B	R/W
PFR7	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000427 _H	–	–	–	–	–	–	TO2E	TO1E	-----00 _B	R/W
PFRG	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000430 _H	–	–	SCK2E	SO2E	–	–	PPG0E	–	--00--0- _B	R/W

PFR0 to PFR2, PFR7 and PFRG control each bit in the output of the corresponding peripheral.

Be sure to write "0" to unused bits.

The following table lists individual PFR registers, their initial values and functions:

Register Name	Bit	Bit Name	Setting value	Function
PFR0	0	PPG1E	0	General-purpose port [Initial value]
			1	PPG timer 1 output
	1	PPG2E	0	General-purpose port [Initial value]
			1	PPG timer 2 output
	2	PPG3E	0	General-purpose port [Initial value]
			1	PPG timer 3 output
	3	PPG4E	0	General-purpose port [Initial value]
			1	PPG timer 4 output
	4	PPG5E	0	General-purpose port [Initial value]
			1	PPG timer 5 output
	5	PPG6E	0	General-purpose port [Initial value]
			1	PPG timer 6 output
	6	PPG7E	0	General-purpose port [Initial value]
			1	PPG timer 7 output
	7	PPG8E	0	General-purpose port [Initial value]
			1	PPG timer 8 output
PFR1	0	PPG9E	0	General-purpose port [Initial value]
			1	PPG timer 9 output
	1	PPG10E	0	General-purpose port [Initial value]
			1	PPG timer 10 output
	2	PPG11E	0	General-purpose port [Initial value]
			1	PPG timer 11 output
	3	PPG12E	0	General-purpose port [Initial value]
			1	PPG timer 12 output
	4	PPG13E	0	General-purpose port [Initial value]
			1	PPG timer 13 output
	5	PPG14E	0	General-purpose port [Initial value]
			1	PPG timer 14 output
	6	PPG15E	0	General-purpose port [Initial value]
			1	PPG timer 15 output

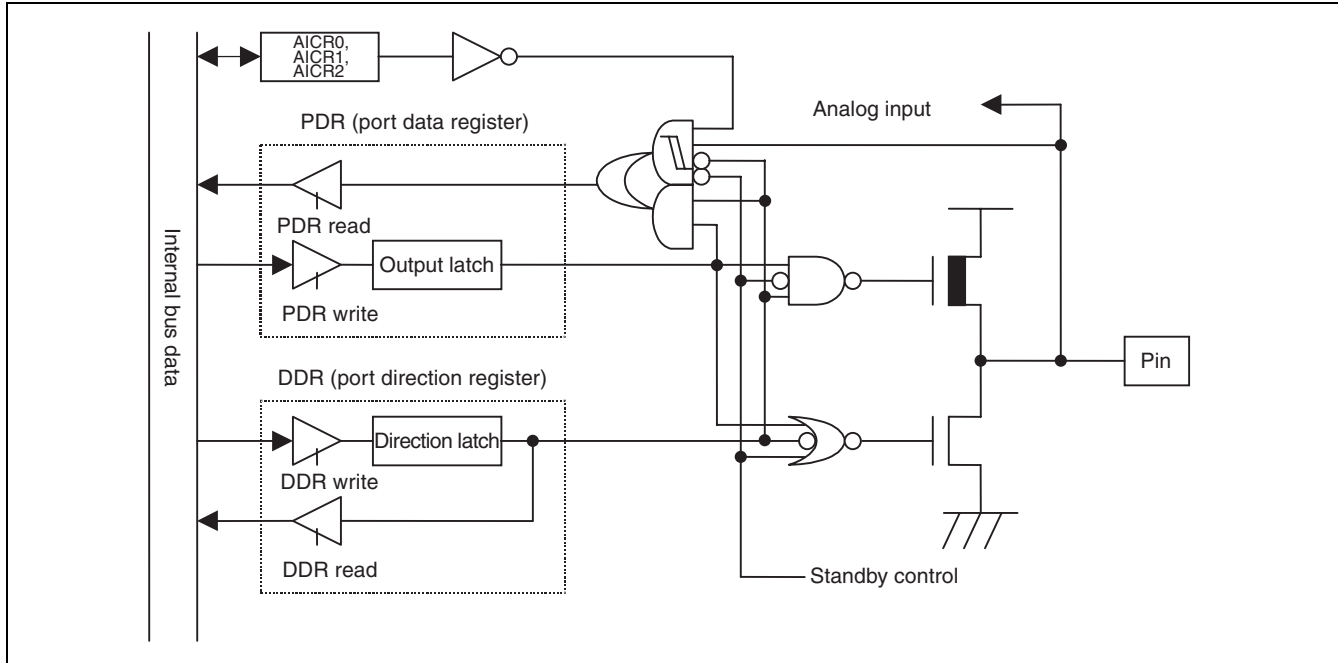
Register Name	Bit	Bit Name	Setting value	Function
PFR2	1	SO0E	0	General-purpose port [Initial value]
			1	UART0 data output
	2	SCK0E	0	General-purpose port [Initial value]
			1	UART0 clock input/output
	4	SO1E	0	General-purpose port [Initial value]
			1	UART1 data output
	5	SCK1E	0	General-purpose port [Initial value]
			1	UART1 clock input/output
PFR7	0	TO1E	0	General-purpose port [Initial value]
			1	Reload timer 1 output
	1	TO2E	0	General-purpose port [Initial value]
			1	Reload timer 2 output
PFRG	1	PPG0E	0	General-purpose port [Initial value]
			1	PPG timer 0 output
	4	SO2E	0	General-purpose port [Initial value]
			1	UART2 data output
	5	SCK2E	0	General-purpose port [Initial value]
			1	UART2 clock input/output

4.3 Analog Input Ports

This section shows a block diagram and register configuration of 8/10-bit A/D converter pin.

■ Block Diagram of 8/10-bit A/D Converter Pin

Figure 4.3-1 AN0 to AN11 Pin Block Diagram



Notes:

- To use a pin as an input port, set its bit in the corresponding DDR register to "0" and add a pull-up resistor to the external pin. Also set the corresponding bit in the AICR register to "0".
- To use a pin as an analog input pin, set the corresponding bit in the AICR register to "1". The value read from the PDR register at this time matches the PDR register value.

■ Analog Input Control Register (AICR:AICR0 to AICR2)

AICR0	7	6	5	4	3	2	1	0	Initial value	Access
Address: 0000007E _H	AN7E	AN6E	AN5E	AN4E	AN3E	AN2E	AN1E	AN0E	00000000 _B	R/W
AICR1	7	6	5	4	3	2	1	0	Initial value	Access
Address: 00000086 _H	—	—	—	—	—	—	AN9E	AN8E	-----00 _B	R/W
AICR2	7	6	5	4	3	2	1	0	Initial value	Access
Address: 0000008E _H	—	—	—	—	—	—	AN11E	AN10E	-----00 _B	R/W

AICRs control the pins of their respective I/O ports as follows:

AICR = 0: Port input mode

AICR = 1: Analog input mode

Cleared to "0" at reset.

CHAPTER 5

INTERRUPT CONTROLLER

This chapter outlines the interrupt control, describes its register configuration/functions and its operations, and gives an example of using the hold request cancel request.

5.1 Overview

5.2 Interrupt Control Registers

5.3 Operation of Interrupt Controller

5.1 Overview

The interrupt controller controls interrupt acceptance and arbitration.

■ Hardware Configuration

This module consists of the following:

- ICR registers
- Interrupt priority evaluation circuit
- Interrupt level and interrupt number (vector) generator
- Hold request cancel request generator

■ Major Functions

This module provides the following functions:

- NMI request/interrupt request detection
- Priority evaluation (by interrupt level and number)
- Transfer of prioritizing interrupt level (to the CPU)
- Transfer of prioritizing interrupt number (to the CPU)
- Request (to the CPU) for returning from stop mode in response to an NMI/interrupt request with interrupt level other than "1111_B"
- Generation of a hold request cancel request to DMAC

■ Register List

Figure 5.1-1 Register List (1/2)

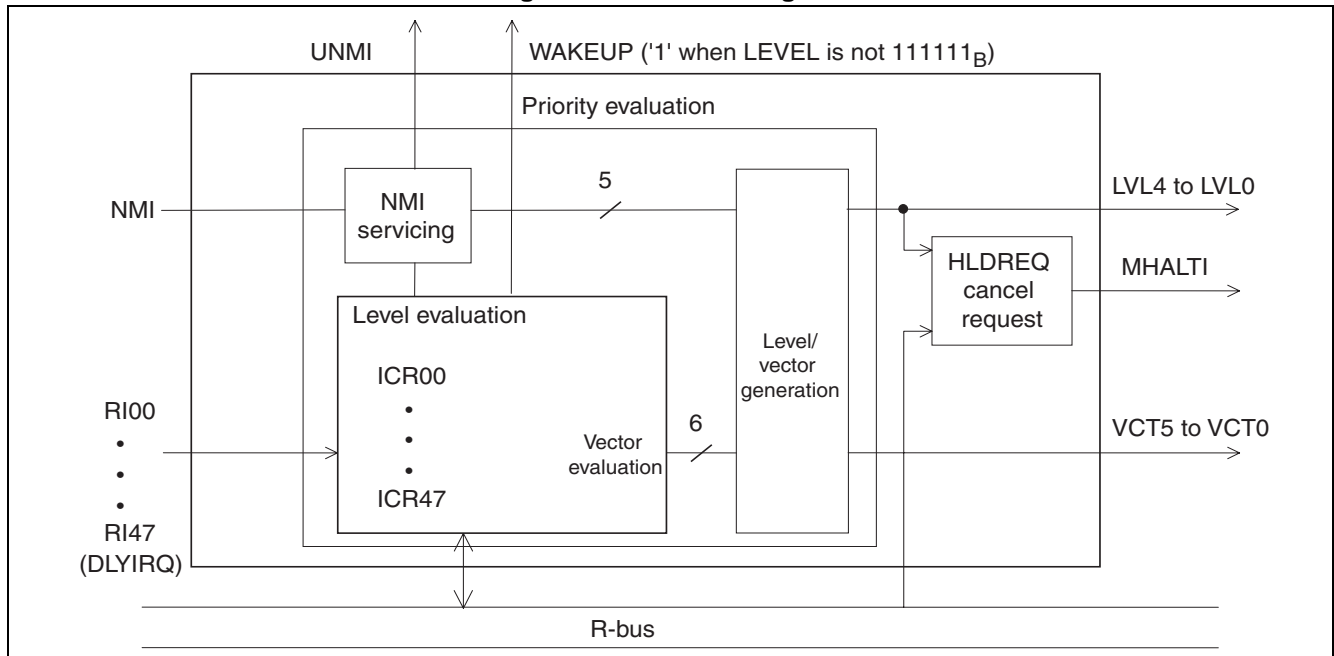
Address	7	6	5	4	3	2	1	0	← Bit No.
00000440 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR00
00000441 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR01
00000442 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR02
00000443 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR03
00000444 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR04
00000445 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR05
00000446 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR06
00000447 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR07
00000448 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR08
00000449 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR09
0000044A _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR10
0000044B _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR11
0000044C _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR12
0000044D _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR13
0000044E _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR14
0000044F _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR15
00000450 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR16
00000451 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR17
00000452 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR18
00000453 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR19
00000454 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR20
00000455 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR21
00000456 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR22
00000457 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR23
00000458 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR24
00000459 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR25
0000045A _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR26
0000045B _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR27
0000045C _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR28
0000045D _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR29
0000045E _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR30
0000045F _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR31
				R	R/W	R/W	R/W	R/W	

Figure 5.1-1 Register List (2/2)

Address	7	6	5	4	3	2	1	0	← Bit No.
00000460 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR32
00000461 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR33
00000462 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR34
00000463 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR35
00000464 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR36
00000465 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR37
00000466 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR38
00000467 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR39
00000468 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR40
00000469 _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR41
0000046A _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR42
0000046B _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR43
0000046C _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR44
0000046D _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR45
0000046E _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR46
0000046F _H	–	–	–	ICR4	ICR3	ICR2	ICR1	ICR0	ICR47
				R	R/W	R/W	R/W	R/W	
00000045 _H	MHALTI	–	–	LVL4	LVL3	LVL2	LVL1	LVL0	HRCL
				R	R/W	R/W	R/W	R/W	

■ Block Diagram

Figure 5.1-2 Block Diagram



5.2 Interrupt Control Registers

This section describes register configuration and functions of interrupt controller.

■ Interrupt Control Register (ICR:ICR00 to ICR47)

ICR00 to ICR47	Bit No. →	7	6	5	4	3	2	1	0	Initial value
Address		—	—	—	ICR4	ICR3	ICR2	ICR1	ICR0	---11111 _B
000440 _H to 00046F _H					R	R/W	R/W	R/W	R/W	

An interrupt control register (ICR) is provided for each interrupt input and is used to set the interrupt level of the corresponding interrupt request.




[bit4 to bit0] ICR4 to ICR0

These bits are interrupt level setting bits to specify the interrupt level of the corresponding interrupt request. If the interrupt level set in this register is higher than the level mask value set in the ILM register in the CPU, an interrupt request is masked on the CPU side.

The register is initialized to "11111_B" at reset.

Table 5.2-1 below lists the available combinations of values of the interrupt level setting bits and their respective interrupt levels.

Table 5.2-1 Interrupt Level Setting Bit Values and Interrupt Levels

ICR4	ICR3	ICR2	ICR1	ICR0	Interrupt Level	
0	0	0	0	0	0	 System-reserved
0	1	1	1	0	14	
0	1	1	1	1	15	NMI
1	0	0	0	0	16	<div>Highest level available</div> <div>(High)</div>   <div>(Low)</div>
1	0	0	0	1	17	
1	0	0	1	0	18	
1	0	0	1	1	19	
1	0	1	0	0	20	
1	0	1	0	1	21	
1	0	1	1	0	22	
1	0	1	1	1	23	
1	1	0	0	0	24	
1	1	0	0	1	25	
1	1	0	1	0	26	
1	1	0	1	1	27	
1	1	1	0	0	28	
1	1	1	0	1	29	
1	1	1	1	0	30	
1	1	1	1	1	31	Interrupt-disabled

The ICR4 bit is fixed at "1"; it cannot be set to "0".

■ Hold Request Cancel Level Register (HRCL)

Address	7	6	5	4	3	2	1	0	← Bit No.
000045 _H	MHALTI	—	—	LVL4	LVL3	LVL2	LVL1	LVL0	HRCL
	R/W			R	R/W	R/W	R/W	R/W	0--11111 _B (Initial value)

This register is an interrupt level setting register for generating a hold request cancel request.

[bit7] MHALTI

MHALTI suppresses DMA transfer by an NMI request. This bit is set to "1" by an NMI request; writing "0" to the bit clears it. Clear the bit at the end of the NMI routine in the same way as with an ordinary interrupt routine.

[bit4 to bit0] LVL4 to LVL0

These bits are used to set the interrupt level for issuing a hold request cancel request to the bus master.

If an interrupt request having a higher level than that set in this register is generated, a hold request cancel request is issued to the bus master.

The LVL4 bit is fixed at "1"; it cannot be set to "0".

5.3 Operation of Interrupt Controller

This section explains operations of interrupt controller including the following.

- **Priority evaluation**
 - **Non Maskable interrupt (NMI)**
 - **Hold request cancel/request**
 - **Returning from standby mode (Stop or sleep mode)**
-

■ Priority Evaluation

This module selects the interrupt source of the highest priority from among the interrupt sources occurring at the same time and outputs the interrupt level and interrupt number of the selected interrupt source to the CPU.

The criteria for evaluating the priorities of interrupt sources are as follows:

1. NMI
2. Interrupt sources satisfying the following conditions:
 - Interrupt sources with an interrupt level value other than 31. ("31" disables interrupts.)
 - Interrupt sources with the smallest interrupt level value
 - Interrupt source with the smallest interrupt number among above

If any interrupt source is not selected according to the above criteria, the interrupt controller outputs an interrupt level of 31 (1111_B), where the interrupt number is indeterminate.

Table 5.3-1 lists interrupt sources, interrupt numbers, and interrupt levels.

Table 5.3-1 Interrupt Sources, Interrupt Numbers, and Interrupt Levels (1 / 3)

Interrupt Source	Interrupt No.		Interrupt Level	Offset	TBR Default Address	RN
	Decimal	Hexa-decimal				
Reset	0	00	–	3FC _H	000FFFC _H	–
Mode vector	1	01	–	3F8 _H	000FFF8 _H	–
System-reserved	2	02	–	3F4 _H	000FFF4 _H	–
System-reserved	3	03	–	3F0 _H	000FFF0 _H	–
System-reserved	4	04	–	3EC _H	000FFFE _C	–
System-reserved	5	05	–	3E8 _H	000FFFE8 _H	–
System-reserved	6	06	–	3E4 _H	000FFFE4 _H	–
Coprocessor absence trap	7	07	–	3E0 _H	000FFFE0 _H	–
Coprocessor error trap	8	08	–	3DC _H	000FFFD _C	–
INTE instruction	9	09	–	3D8 _H	000FFFD8 _H	–
System-reserved	10	0A	–	3D4 _H	000FFFD4 _H	–
System-reserved	11	0B	–	3D0 _H	000FFFD0 _H	–
Step trace trap	12	0C	–	3CC _H	000FFFC _C	–
NMI request (tool)	13	0D	–	3C8 _H	000FFFC8 _H	–
Undefined instruction exception	14	0E	–	3C4 _H	000FFFC4 _H	–
NMI request	15	0F	15(F _H) Fixed	3C0 _H	000FFFC0 _H	–
External interrupt 0	16	10	ICR00	3BC _H	000FFFB _C	6
External interrupt 1	17	11	ICR01	3B8 _H	000FFFB8 _H	7
External interrupt 2	18	12	ICR02	3B4 _H	000FFFB4 _H	–
External interrupt 3	19	13	ICR03	3B0 _H	000FFFB0 _H	–
External interrupt 4	20	14	ICR04	3AC _H	000FFFA _C	–
External interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H	–
External interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H	–
External interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H	–
Reload timer 0	24	18	ICR08	39C _H	000FFF9 _C	8
Reload timer 1	25	19	ICR09	398 _H	000FFF98 _H	9
Reload timer 2	26	1A	ICR10	394 _H	000FFF94 _H	10
UART0 (Reception complete)	27	1B	ICR11	390 _H	000FFF90 _H	0

Table 5.3-1 Interrupt Sources, Interrupt Numbers, and Interrupt Levels (2 / 3)

Interrupt Source	Interrupt No.		Interrupt Level	Offset	TBR Default Address	RN
	Decimal	Hexa-decimal				
UART0 (Transmission complete)	28	1C	ICR12	38C _H	000FFF8C _H	3
DTTI	29	1D	ICR13	388 _H	000FFF88 _H	–
DMAC0 (Termination, Error)	30	1E	ICR14	384 _H	000FFF84 _H	–
DMAC1 (Termination, Error)	31	1F	ICR15	380 _H	000FFF80 _H	–
DMAC2/3/4 (Termination, Error)	32	20	ICR16	37C _H	000FFF7C _H	–
UART1 (Reception complete)	33	21	ICR17	378 _H	000FFF78 _H	1
UART1 (Transmission complete)	34	22	ICR18	374 _H	000FFF74 _H	4
UART2 (Reception complete)	35	23	ICR19	370 _H	000FFF70 _H	2
UART2 (Transmission complete)	36	24	ICR20	36C _H	000FFF6C _H	5
Multiply-accumulate	37	25	ICR21	368 _H	000FFF68 _H	–
PPG0	38	26	ICR22	364 _H	000FFF64 _H	–
PPG1	39	27	ICR23	360 _H	000FFF60 _H	–
PPG2/3	40	28	ICR24	35C _H	000FFF5C _H	–
PPG4/5/6/7	41	29	ICR25	358 _H	000FFF58 _H	–
PPG8/9/10/11/12/13/14/15	42	2A	ICR26	354 _H	000FFF54 _H	–
External interrupt 8/9	43	2B	ICR27	350 _H	000FFF50 _H	–
Waveform 0 (underflow)	44	2C	ICR28	34C _H	000FFF4C _H	–
Waveform 1 (underflow)	45	2D	ICR29	348 _H	000FFF48 _H	–
Waveform 2 (underflow)	46	2E	ICR30	344 _H	000FFF44 _H	–
Time-base timer overflow	47	2F	ICR31	340 _H	000FFF40 _H	–
Free-running timer (Compare clear)	48	30	ICR32	33C _H	000FFF3C _H	–
Free-running timer (zero detection)	49	31	ICR33	338 _H	000FFF38 _H	–
A/D0	50	32	ICR34	334 _H	000FFF34 _H	–
A/D1	51	33	ICR35	330 _H	000FFF30 _H	–
A/D2	52	34	ICR36	32C _H	000FFF2C _H	–
PWC0 (Measurement complete)	53	35	ICR37	328 _H	000FFF28 _H	–
PWC1 (Measurement complete)	54	36	ICR38	324 _H	000FFF24 _H	–
PWC0 (Overflow)	55	37	ICR39	320 _H	000FFF20 _H	–

Table 5.3-1 Interrupt Sources, Interrupt Numbers, and Interrupt Levels (3 / 3)

Interrupt Source	Interrupt No.		Interrupt Level	Offset	TBR Default Address	RN
	Decimal	Hexa-decimal				
PWC1 (Overflow)	56	38	ICR40	31C _H	000FFF1C _H	–
ICU 0 (Capture)	57	39	ICR41	318 _H	000FFF18 _H	–
ICU 1 (Capture)	58	3A	ICR42	314 _H	000FFF14 _H	–
ICU 2/3 (Capture)	59	3B	ICR43	310 _H	000FFF10 _H	–
OCU0/1 (Match)	60	3C	ICR44	30C _H	000FFF0C _H	–
OCU2/3 (Match)	61	3D	ICR45	308 _H	000FFF08 _H	–
OCU4/5 (Match)	62	3E	ICR46	304 _H	000FFF04 _H	–
Delayed interrupt source bit	63	3F	ICR47	300 _H	000FFF00 _H	–
System-reserved (used by REALOS)	64	40	–	2FC _H	000FFEFC _H	–
System-reserved (used by REALOS)	65	41	–	2F8 _H	000FFE8 _H	–
System-reserved	66	42	–	2F4 _H	000FFE4 _H	–
System-reserved	67	43	–	2F0 _H	000FFE0 _H	–
System-reserved	68	44	–	2EC _H	000FEEC _H	–
System-reserved	69	45	–	2E8 _H	000FEE8 _H	–
System-reserved	70	46	–	2E4 _H	000FEE4 _H	–
System-reserved	71	47	–	2E0 _H	000FEE0 _H	–
System-reserved	72	48	–	2DC _H	000FFEDC _H	–
System-reserved	73	49	–	2D8 _H	000FFED8 _H	–
System-reserved	74	4A	–	2D4 _H	000FFED4 _H	–
System-reserved	75	4B	–	2D0 _H	000FFED0 _H	–
System-reserved	76	4C	–	2CC _H	000FFECC _H	–
System-reserved	77	4D	–	2C8 _H	000FFEC8 _H	–
System-reserved	78	4E	–	2C4 _H	000FFEC4 _H	–
System-reserved	79	4F	–	2C0 _H	000FFEC0 _H	–
Used by INT instruction	80 to 255	50 to FF	–	2BC _H to 000 _H	000FFEBC _H to 000FFC00 _H	–

■ NMI (Non Maskable Interrupt)

NMIs have the highest priority among the interrupt sources handled by this module.

An NMI is therefore always selected even whenever it is generated at the same time as another interrupt source.

- When an NMI occurs, the interrupt controller passes the following items of information to the CPU:

Interrupt level : 15 (01111_B)

Interrupt number: 15 (0001111_B)

- NMI detection

NMIs are set and detected by the external interrupt/NMI controller module. This module only generates an interrupt level, interrupt number, and MHALTI in response to an NMI request.

- Suppressing DMA transfer upon NMI request

When an NMI request occurs, the MHALTI bit in the HRCL register is set to "1", suppressing DMA transfer. To permit DMA transfer, clear the MHALTI bit to "0" at the end of the NMI routine.

■ Hold Request Cancel Request

When a high-priority interrupt needs to be serviced when the CPU has been put on hold (during DMA transfer), it is necessary to request the hold request issuer to cancel the hold request. Use the HRCL register to set the interrupt level as the reference level for generating the hold request cancel request.

- Conditions for generating a hold request cancel request

A hold request cancel request is issued to the DMAC when an interrupt source of a higher interrupt level than that set in the HRCL register occurs.

Interrupt level set in the HRCL register > Interrupt level after priority evaluation → Cancel request generated

Interrupt level set in the HRCL register ≤ Interrupt level after priority evaluation → No cancel request

Once issued, the cancel request remains in effect until the interrupt source causing that request is cleared, accordingly leaving DMA transfer prevented from being executed. Therefore, be sure to clear the relevant interrupt source. When an NMI is used, the MHALTI bit in the HRCL register is "1" and thus the cancel request is in effect.

- Interrupt levels available

The HRCL register accepts a value from "10000_B" to "11111_B" like the ICR register.

If the HRCL register is set to "11111_B", a cancel request is generated for every level of interrupt. If it is set to "10000_B", a cancel request is generated for NMIs only.

Table 5.3-2 lists the interrupt levels for which a hold request cancel request is generated.

Table 5.3-2 Settings of Interrupt Levels for Which Hold Request Cancel Request Is Generated

HRCL Register	Interrupt Level for Which Cancel Request Is Generated
16	NMI only
17	NMI or interrupt level 16
18	NMI or interrupt level 16/17
19	NMI or interrupt level 16/17
20	NMI or interrupt level 16/17
21	NMI or interrupt level 16/17
22	NMI or interrupt level 16/17
23	NMI or interrupt level 16/17
24	NMI or interrupt level 16/17
25	NMI or interrupt level 16/17
26	NMI or interrupt level 16/17
27	NMI or interrupt level 16/17
28	NMI or interrupt level 16/17
29	NMI or interrupt level 16/17
30	NMI or interrupt level 16/17
31	NMI or interrupt level 16 to 30 [initial value]

After reset, DMA transfer is suppressed for any level of interrupt. Since DMA transfer is not executed with an interrupt generated, set the HRCL register to an appropriate value.

■ Returning from Standby Mode (Stop or Sleep Mode)

This module provides the function to return from stop mode when an interrupt request occurs.

If any interrupt request (with an interrupt level other than 11111_B), including an NMI, occurs from a peripheral resource, this module issues a request to the clock control unit to return from stop mode.

Since the priority evaluation circuit restarts operation after the clock supply recovers after returning from the stop mode, the CPU continues to execute instructions until a priority evaluation result is obtained.

Even after returning from the sleep state, this module operates in the same way. Note also that the registers in this module are accessible even in sleep mode.

Notes:

- Even an NMI request causes return from stop mode. This assumes however that an input level valid in the stop state is given to the NMI pin.
 - To prevent an interrupt source from causing return from the stop or sleep state, use the relevant control register of the corresponding peripheral resource to set the interrupt level to "11111_B".
-

■ Example of Using the Hold Request Cancel Register (HRCL)

To execute a high-priority process during DMA transfer, the CPU must request the DMA controller to cancel the hold request for releasing itself from the hold status. That is, the HRCL can use an interrupt to make the DMA controller to cancel a hold request, or to give priority to the CPU.

● Control registers

Hold request cancel level (HRCL) register:

If an interrupt request of a higher level than that set in this register is generated, a hold request cancel request is issued to the DMA controller. This register is used to set that reference level.

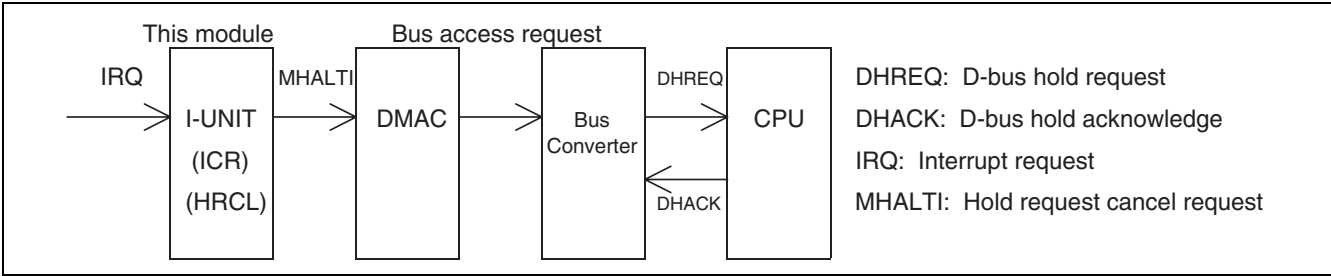
Interrupt control register (ICR):

A higher level than that in the HRCL register is set in the ICR register corresponding to the interrupt source to be used.

● Hardware configuration

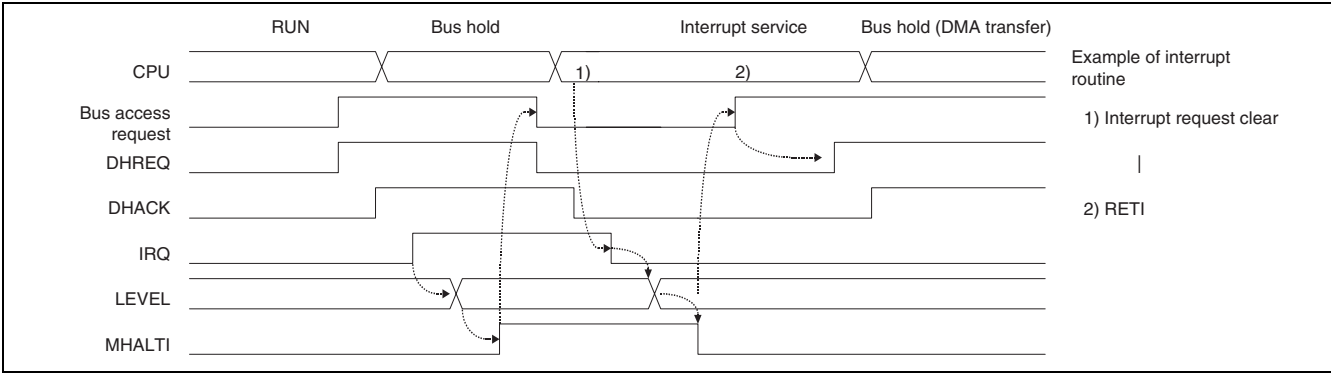
The flow of each signal is illustrated below.

Figure 5.3-1 Flow of Each Signal



● Sequence

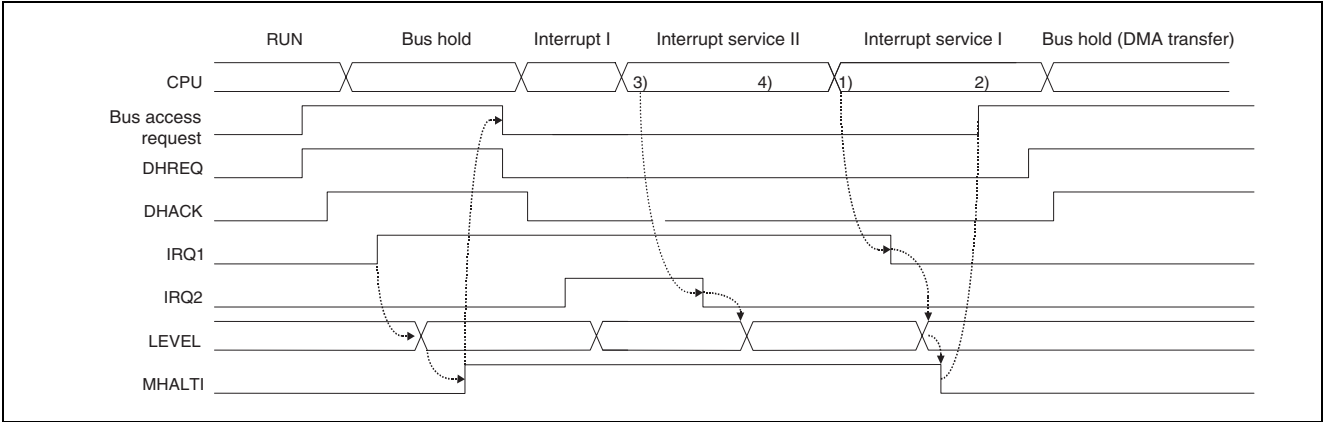
Figure 5.3-2 Interrupt Level: HRCL < ICR (LEVEL)



If an interrupt request is generated and the interrupt level becomes higher than that set in the HRCL register, MHALTI becomes active to the DMA controller. Then the DMA controller cancels the access request, allowing the CPU to return from the hold status for servicing the interrupt.

Given below is an example of handling multiple interrupts.

Figure 5.3-3 Interrupt Level: HRCL < ICR (Interrupt I) < ICR (Interrupt II)



Example of interrupt routine

1), 3) Interrupt source clear

—

2), 4) RETI

In the above example, an interrupt of a higher priority occurs during execution of interrupt routine I.

DHREQ remains low when an interrupt of a higher level than the interrupt level set in the HRCL register has been generated.

Note:

Pay due attention to the relationships between the interrupt levels set in the HRCL and ICR registers.

CHAPTER 6

EXTERNAL INTERRUPT AND NMI CONTROLLER

This chapter describes the external interrupt and NMI controller, the configuration and functions of registers, and operation of the external interrupt and NMI controller.

- 6.1 Overview of External Interrupt/NMI Controller
- 6.2 Registers of External Interrupt/NMI Controller
- 6.3 Operation of External Interrupt/NMI Controller

6.1 Overview of External Interrupt/NMI Controller

The external interrupt controller is a block that controls external interrupt requests input to $\overline{\text{NMI}}$ and INT0 to INT9.

For external interrupt input, "H" level, "L" level, "rising edge", or "falling edge" can be selected as the level of a request to be detected.

■ Register List

Following figure shows the register list of the external interrupt/NMI controller.

External interrupt source register

Bit No. →	7	6	5	4	3	2	1	0	Initial value
EIRR0 Address: 00000040 _H	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	00000000 _B [R/W]
EIRR1 Address: 000000B8 _H	—	—	—	—	—	—	ER9	ER8	-----00 _B [R/W]

Interrupt enable register

Bit No. →	7	6	5	4	3	2	1	0	Initial value
ENIR0 Address: 00000041 _H	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	00000000 _B [R/W]
ENIR1 Address: 000000B9 _H	—	—	—	—	—	—	EN9	EN8	-----00 _B [R/W]

External interrupt request level setting register

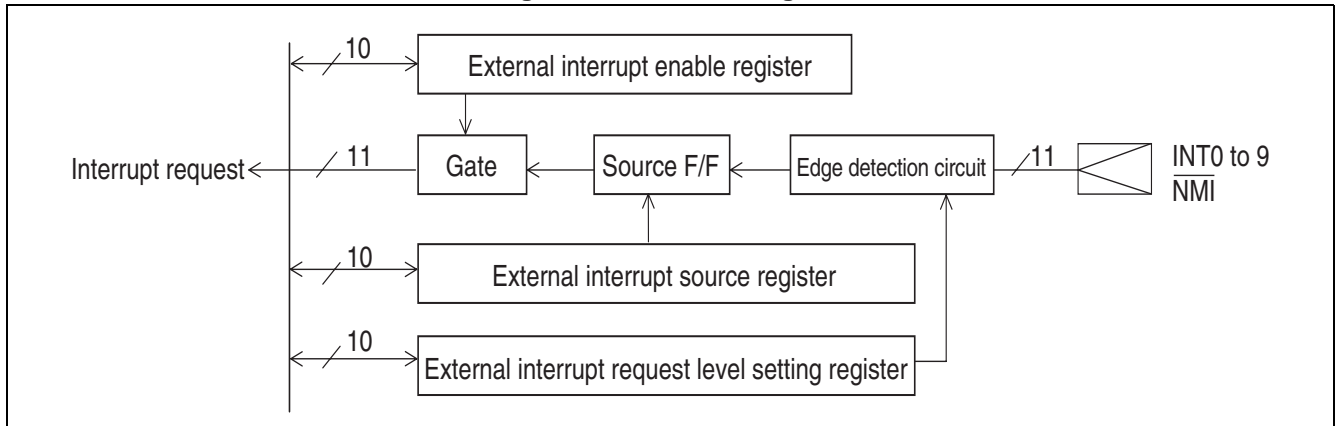
Bit No. →	15	14	13	12	11	10	9	8	Initial value
ELVR0 Address: 00000042 _H	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	00000000 _B [R/W]
ELVR1 Address: 000000BA _H	—	—	—	—	—	—	—	—	----- _B [R/W]

Bit No. →	7	6	5	4	3	2	1	0	Initial value
ELVR0 Address: 00000043 _H	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	00000000 _B [R/W]
ELVR1 Address: 000000BB _H	—	—	—	—	LB9	LA9	LB8	LA8	----0000 _B [R/W]

■ Block Diagram

Figure 6.1-1 shows a block diagram for external interrupt/NMI controller.

Figure 6.1-1 Block Diagram



6.2 Registers of External Interrupt/NMI Controller

This section explains the configuration and functions of registers used by external interrupt/NMI controller.

■ Interrupt Enable Register (ENIR (ENIR0, ENIR1): ENable Interrupt Request Register)

Bit No. →	7	6	5	4	3	2	1	0	Initial value
ENIR0 Address: 00000041 _H	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	00000000 _B [R/W]
ENIR1 Address: 000000B9 _H	—	—	—	—	—	—	EN9	EN8	-----00 _B [R/W]

ENIR register controls mask of the external interrupt request output. Output for an interrupt request is enabled based on the bit in this register to which "1" has been written (INT0 enable is controlled by EN0), and the interrupt request is output to the interrupt controller. The pin corresponding to the bit to which "0" is written holds the interrupt source but does not generate a request to the interrupt controller.

Please make sure to write "0" to bit7 to bit2 of ENIR1 register.

No enable bit exists for NMI.

■ External Interrupt Source Register (EIRR (EIRR0, EIRR1): External Interrupt Request Register)

Bit No. →	7	6	5	4	3	2	1	0	Initial value
EIRR0 Address: 00000040 _H	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	00000000 _B [R/W]
EIRR1 Address: 000000B8 _H	—	—	—	—	—	—	ER9	ER8	-----00 _B [R/W]

EIRR register is a register that shows a corresponding external interrupt request exists when reading, and that clears a content of the flip-flop showing this request when writing.

If the read value of this EIRR register is "1", there is an external interrupt request at the pin corresponding to this bit. Write "0" to this register to clear the request flip-flop of the corresponding bit.

Writing "1" to this register is invalid.

"1" is read in a read operation of the read-modify-write.

The flag for NMI cannot be accessed by a user.

■ External Interrupt Request Level Setting Register (ELVR (ELVR0,ELVR1): External LeVel Register)

Bit No. →	15	14	13	12	11	10	9	8	Initial value
ELVR0 Address: 00000042 _H	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	00000000 _B [R/W]
ELVR1 Address: 000000BA _H	—	—	—	—	—	—	—	—	----- _B [R/W]
Bit No. →	7	6	5	4	3	2	1	0	Initial value
ELVR0 Address: 00000043 _H	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	00000000 _B [R/W]
ELVR1 Address: 000000BB _H	—	—	—	—	LB9	LA9	LB8	LA8	----0000 _B [R/W]

ELVR is a register to select request detections. In ELVR, two bits each are assigned to INT0 to INT9, which results in the settings shown in table below. When each bit of the ELVR is cleared while the level is in the request input level, an appropriate bit is set again as long as the input is at active level.

Set "H" level or "L" level request when returning from the stop state.

Table 6.2-1 Assignment of ELVR

LB9 to LB0, LA9 to LA0	Operation
0 0 _B	"L" level indicates the existence of a request
0 1 _B	"H" level indicates the existence of a request
1 0 _B	A rising edge indicates the existence of a request
1 1 _B	A falling edge indicates the existence of a request

Detection level of NMI is always a falling edge level. Also, when using NMI to return from the stop state, detection level is "L" level.

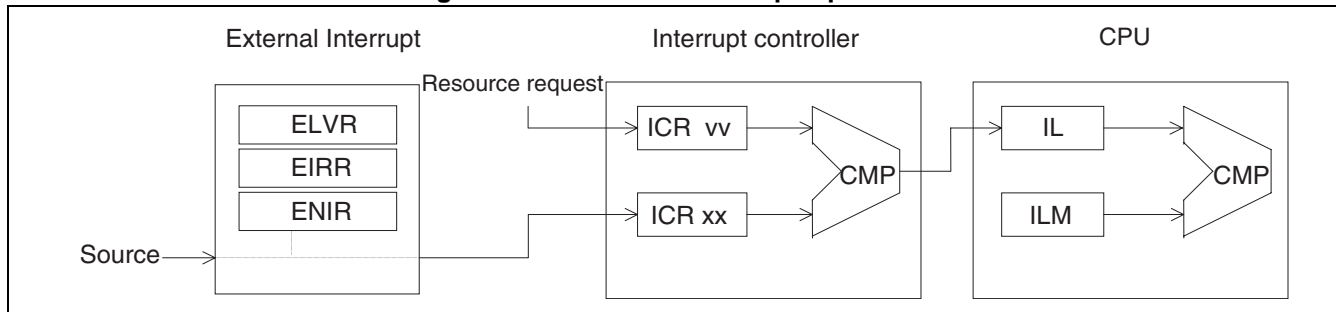
6.3 Operation of External Interrupt/NMI Controller

If, after a request level and an enable register are defined, a request defined in the ELVR register is input to the corresponding pin, this module generates an interrupt request signal to the interrupt controller. For simultaneous interrupt requests from resources, the interrupt controller determines the interrupt request with the highest priority and generates an interrupt for it.

■ Operation of an External Interrupt

Figure 6.3-1 shows the external interrupt operation.

Figure 6.3-1 External Interrupt Operation



■ Return from Standby

To use an external interrupt to return from the standby state in the clock stop mode, set "H" level or "L" level request as the input request.

If you use an edge request, the device does not return from the stop state in clock stop mode.

Please make sure to disable unused channels before entering a standby mode.

■ Operating Procedure for an External Interrupt

Set up a register located inside the external interrupt controller as follows:

1. Set the general-purpose I/O port served dual use as external interrupt input pin as the input port.
2. Disable the target bit in the enable register.
3. Set the target bit in the request level setting register.
4. Clear the target bit in the interrupt source register.
5. Enable the target bit in the enable register.

(Simultaneous writing of 16-bit data is supported for steps 4. and 5.)

Before setting a register in this module, you must disable the enable register. In addition, before enabling the enable register, you must clear the interrupt source register. This procedure is required to prevent an interrupt source from occurring by mistake while a register is being set or an interrupt is enabled.

■ External Interrupt Request Level

1. If the request level is an edge request, a pulse width of at least three machine cycles (peripheral clock machine cycles) is required to detect an edge.
2. If the request input level is a level setting, a request input is entered from outside and is then cancelled, the request to the interrupt controller remains active because a source holding circuit exists internally.

The external interrupt source register must be cleared to cancel a request to the interrupt controller.

Figure 6.3-2 Clearing the External Interrupt Source Register when a Level is Set

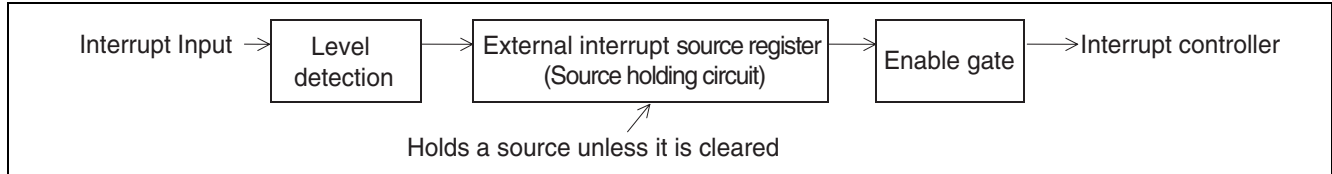
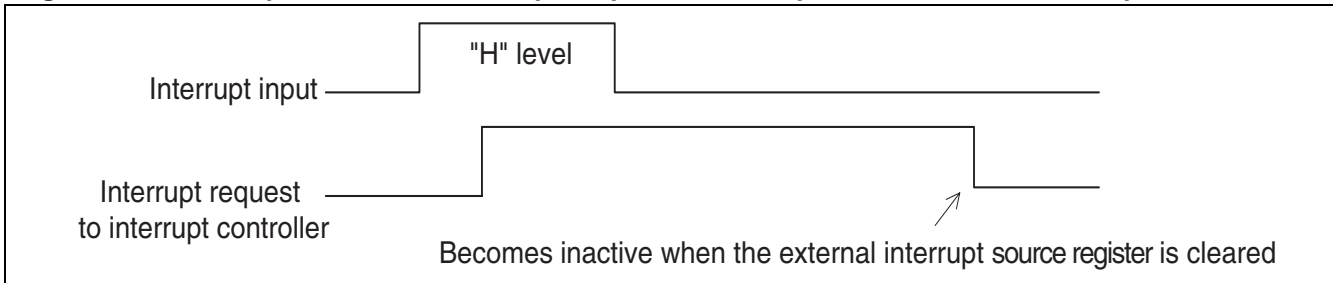


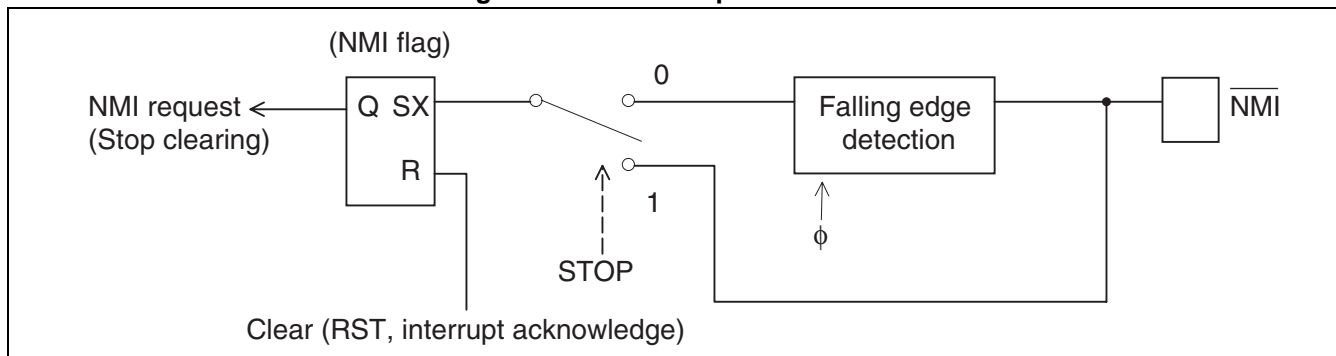
Figure 6.3-3 Interrupt Source and Interrupt Request to Interrupt Controller when Interrupts are Enabled



■ NMI

1. An NMI has the highest level among the user interrupts and cannot be masked.
However, as an exception, when NMI is activated without setting ILM, NMI source is detected but CPU will not accept the NMI request. At this time, the NMI source will be held until ILM is set to be accepted by NMI. For this reason, use NMI after resetting and setting ILM value to 16 or higher. Also, since an internal source flag of NMI cannot be accessed from CPU, keep $\overline{\text{NMI}}$ pin to "H" level after reset.
2. An NMI is accepted under the following conditions:
Normal state :Falling edge
STOP state : "L" level
3. An NMI can be used to clear stop mode. Inputting the "L" level in the stop state clears the stop state and causes the oscillation stabilization wait time to start.
The NMI request detector has an NMI flag that is set for an NMI request and is cleared only if an interrupt for the NMI itself is accepted or reset occurs.
Note that this bit is not readable or writable.

Figure 6.3-4 NMI Request Detector

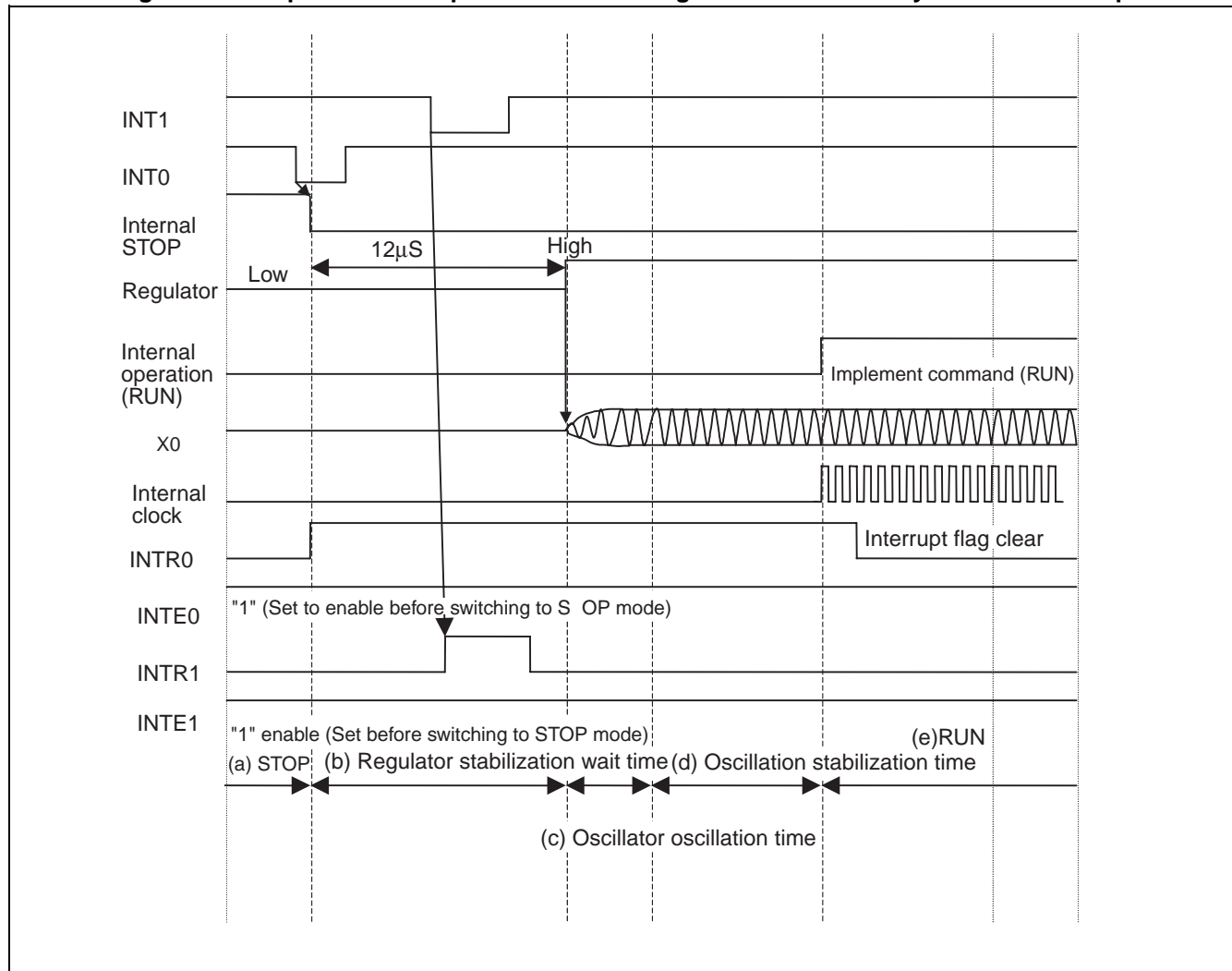


■ Precautions when Returning from STOP State Using External Interrupt

The external interrupt signal that is initially input to the INT pin in a STOP state is input asynchronously, allowing the device to return from the STOP state. Note, however, that there are periods from the release of the STOP state till the end of the oscillation stabilization wait time, in such periods the input of other external interrupt signals cannot be identified (period of $b + c + d$ in Figure 6.3-5). This is because the external input signal after the release of STOP mode is synchronized with the internal clock; consequently, the corresponding interrupt source cannot be retained while the clock is still unstable.

Therefore, input an external interrupt signal after the oscillation stabilization wait time has passed, when inputting an external interrupt after the release of STOP mode.

Figure 6.3-5 Operational Sequence for Returning from STOP State by External Interrupt



■ Return Operation from STOP State

The following operation is performed when using an external interrupt to allow the current circuit to return from the STOP state.

● Processing procedure before the device enters the STOP state

Setting external interrupts

You must enable the interrupt input pass for the STOP state before the device enters the STOP state. This can be done by setting the PFR register (Port Function Register). Nevertheless, the interrupt input pass is already enabled in normal states (those other than STOP state); therefore, you do not have to be conscious to set the register. In the STOP state, on the other hand, the input pass is controlled by the PFR register.

Pin name	Setting for returning from STOP state by external interrupt
P63/INT9	Set bit 3 in PCR6 to "0".
P62/INT8	Set bit 2 in PCR6 to "0".
P26/INT7	Set bit 7 in PCR2 to "0".
P27/INT6	Set bit 6 in PCR2 to "0".
P62/INT5/PPG0	Set bit 1 in PCR6 to "0".
P60/INT4/CKI	Set bit 0 in PCR6 to "0".
P57/INT3	Set bit 7 in PFR5 to "0".
P56/INT2	Set bit 6 in PFR5 to "0".
P55/INT1	Set bit 5 in PFR5 to "0".
P54/INT0	Set bit 4 in PFR5 to "0".

Inputting external interrupts

When allowing the device to return from the STOP state, external interrupt signals are in such a state that they send input signals asynchronously. As soon as this interrupt input is asserted, the internal STOP signal is caused to fall. At the same time, the external interrupt circuit switches its mode to synchronize the interrupt input of other levels.

● Regulator stabilization wait time

When the internal STOP signal falls, the device starts switching from the STOP regulator to the RUN regulator. If internal operation starts before the output of RUN regulator voltage stabilizes, the operation will become unstable. For this reason, a regulator stabilization wait time of approximately 12μs is secured as the stabilization wait time for the internal output voltage. The clock is stopped during this period.

● Oscillation time of oscillator

The clock starts oscillation upon the completion of the regulator stabilization wait time. The oscillation time of the oscillator varies depending on the oscillator used.

- Oscillation stabilization wait time

The oscillation stabilization wait time is spent within the device after the oscillation time of the oscillator. The oscillation stabilization wait time is specified by using the OS1 and OS0 bits in the standby control register. Upon the completion of the oscillation stabilization wait time, the internal clock will be supplied and the interrupt instruction operation will start using an external interrupt. In addition, it will be enabled to accept external interrupt sources other than those for returning from the STOP state.

CHAPTER 7

REALOS-RELATED HARDWARE

REALOS-related hardware is used by the realtime OS. Therefore, when using REALOS, the hardware cannot be used with the user program.

- 7.1 Delayed Interrupt Module
- 7.2 Register of Delayed Interrupt Module
- 7.3 Operation of Delayed Interrupt Module
- 7.4 Bit Search Module
- 7.5 Register of Bit Search Module
- 7.6 Operation of Bit Search Module

7.1 Delayed Interrupt Module

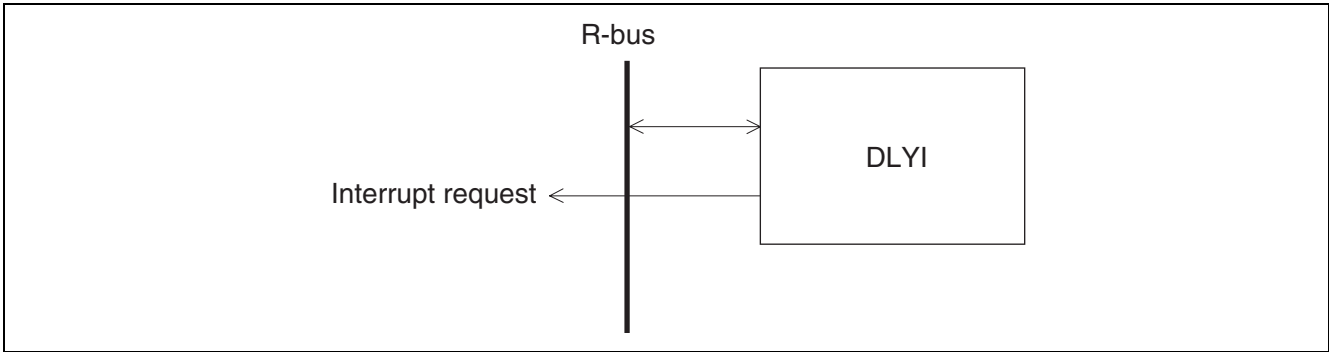
The delayed interrupt module generates an interrupt for switching tasks. Use this module to allow software to generate and clear an interrupt request for the CPU.

■ Register List

Address	7	6	5	4	3	2	1	0	← Bit No.
00000044 _H	—	—	—	—	—	—	—	DLYI	DICR
								[R/W]	

■ Block Diagram

Figure 7.1-1 Block Diagram



7.2 Register of Delayed Interrupt Module

This section describes the configuration and functions of a register used by the delayed interrupt module.

■ DICR (Delayed Interrupt Control Register)

Address	7	6	5	4	3	2	1	0	Initial value
000044 _H	—	—	—	—	—	—	—	DLYI	-----0 _B
								[R/W]	

The delayed interrupt control register (DICR) controls delayed interrupts.

[bit0] DLYI

DLYI	Description
0	A delayed interrupt source is cleared or no request exists. [Initial value]
1	A delayed interrupt source is generated.

This bit controls generation and cancellation of the corresponding interrupt source.

7.3 Operation of Delayed Interrupt Module

A delayed interrupt generates an interrupt for switching tasks. Use this function to allow software to generate and clear an interrupt request for the CPU.

■ Interrupt Number

A delayed interrupt is assigned to the interrupt source corresponding to the largest interrupt number.

On MB91260B series, a delayed interrupt is assigned to interrupt number 63 (3F_H).

■ DLYI Bit of DICR

Write "1" to this bit to generate a delayed interrupt source. Write "0" to it to clear a delayed interrupt source.

This bit is the same as the interrupt source flag for a normal interrupt. Therefore, clear this bit and switch tasks in the interrupt routine.

7.4 Bit Search Module

The bit search module searches for "0", "1", or any points of change for data written to the input register and then returns the detected bit locations.

■ Register List

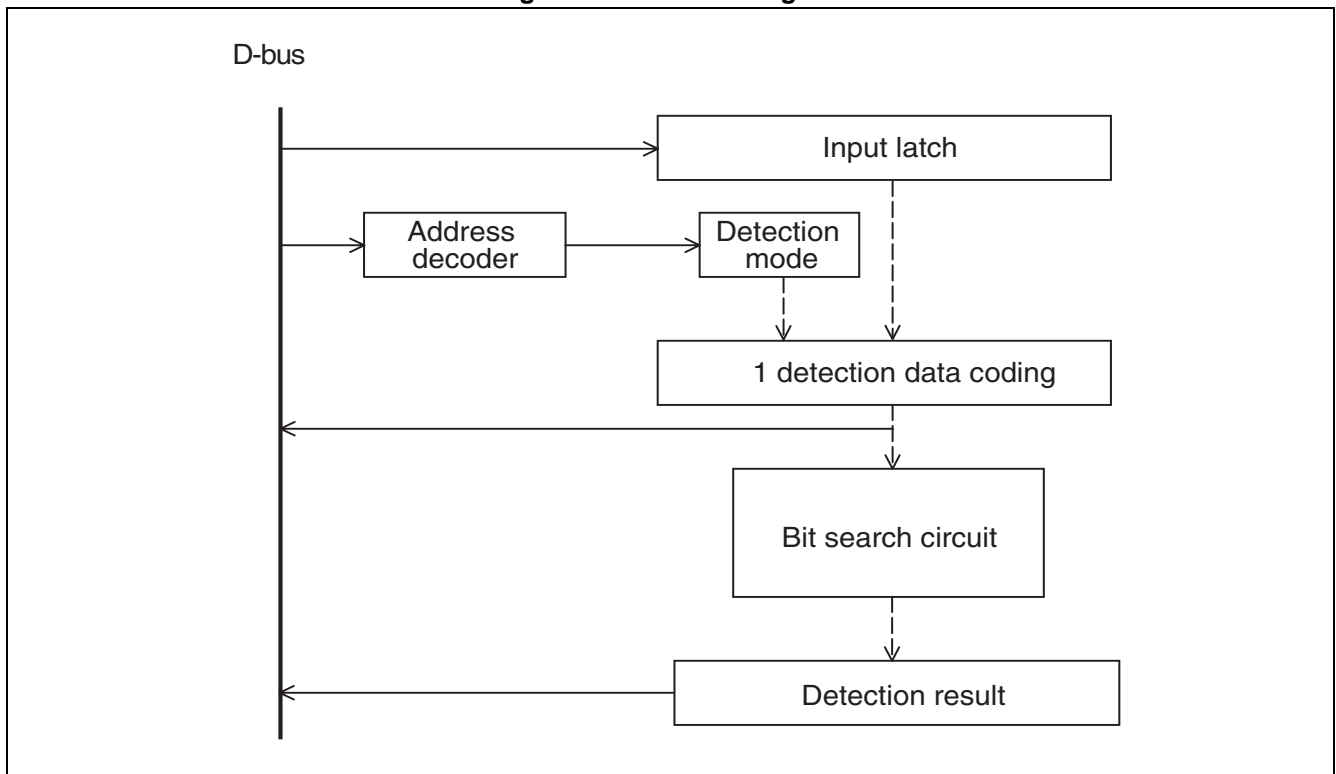
Figure 7.4-1 shows the register list for bit search module.

Figure 7.4-1 Register List

Address	31	0	
000003F0 _H	BSD0		0 detection data register
000003F4 _H	BSD1		1 detection data register
000003F8 _H	BSDC		Change point detection data register
000003FC _H	BSSR		Detection result register

■ Block Diagram

Figure 7.4-2 Block Diagram



7.5 Register of Bit Search Module

This section describes the configuration and functions of registers used by the bit search module.

■ 0 Detection Data Register (BSD0)

Address	31	0
000003F0 _H		
Read/Write →	W	
Initial value →	Undefined XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX _B	

0 detection is performed for the written value.

The initial value after reset is undefined. The read value is undefined.

Use a 32-bit length data transfer instruction for data transfer (Do not use 8-bit or 16-bit length data transfer instruction).

■ 1 Detection Data Register (BSD1)

Address	31	0
000003F4 _H		
Read/Write →	R/W	
Initial value →	Undefined XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX _B	

Use a 32-bit length data transfer instruction for data transfer (Do not use 8-bit or 16-bit length data transfer instruction).

- Writing
Detect "1" for the written value.
- Reading
Save data of the internal state of the bit search module is read. This register is used to save or restore the original state when the bit search module is used by, for example, an interrupt handler.
Even though data is written to the 0 detection or change point detection or data register, data can be saved or restored only by using the 1 detection data register.
The initial value after reset is undefined.

■ Change Point Detection Data Register (BSDC)

Address	31	0
000003F8 _H		
Read/Write →	W	
Initial value →	Undefined	XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX _B

Points of change are detected in the written value.

The initial value after reset is undefined.

The read value is undefined.

Use a 32-bit length data transfer instruction for data transfer (Do not use 8-bit or 16-bit length data transfer instruction).

■ Detection Result Register (BSRR)

Address	31	0
000003FC _H		
Read/Write →	R	
Initial value →	Undefined	XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX _B

The 0, 1, or change point detection result is read from this register.

The detection result to be read is determined by the last written data register.

7.6 Operation of Bit Search Module

This section explains 3 types of the bit search module operation.

- 0 detection
- 1 detection
- Change point detection

■ 0 Detection

The bit search module scans data written to the 0 detection data register from the MSB to LSB and returns the location where the first "0" is detected.

The detection result can be obtained by reading the detection result register.

The relationship between the detected location and the return value is given in Table 7.6-1.

If a "0" is not found (that is, if the value is FFFFFFFF_H), 32 is returned as the detection result.

[Execution example]

Write data	Read value (decimal)
111111111111111111000000000000 _B (FFFFF000 _H)	→ 20
11111000010010011110000010101010 _B (F849E0AA _H)	→ 5
100000000000000101010101010101010 _B (8002AAAA _H)	→ 1
11111111111111111111111111111111 _B (FFFFFFF _H)	→ 32

■ 1 Detection

The bit search module scans data written to the 1 detection data register from the MSB to LSB and returns the location where the first "1" is detected.

The detection result can be obtained by reading the detection result register.

The relationship between the detected location and the return value is given in Table 7.6-1.

If a "1" is not found (that is, if the value is 00000000_H), 32 is returned as the detection result.

[Execution example]

Write data	Read value (decimal)
00100000000000000000000000000000 _B (20000000 _H)	→ 2
00000001001000110100010101100111 _B (01234567 _H)	→ 7
00000000000000111111111111111111 _B (0003FFFF _H)	→ 14
00000000000000000000000000000001 _B (00000001 _H)	→ 31
00000000000000000000000000000000 _B (00000000 _H)	→ 32

■ Change Point Detection

The bit search module scans data written to the change point detection data register from bit30 to the LSB for comparison with the MSB value. The first location where a value that is different from that of the MSB is detected is returned.

The detection result can be obtained by reading the detection result register.

The relationship between the detected location and the return value is given in Table 7.6-1.

If a change point is not detected, 32 is returned.

In change point detection, "0" is never returned as a result.

[Execution example]

Write data	Read value (decimal)
00100000000000000000000000000000 _B (20000000 _H)	→ 2
00000001001000110100010101100111 _B (01234567 _H)	→ 7
00000000000000111111111111111111 _B (0003FFFF _H)	→ 14
00000000000000000000000000000001 _B (00000001 _H)	→ 31
00000000000000000000000000000000 _B (00000000 _H)	→ 32
11111111111111111111000000000000 _B (FFFFFF00 _H)	→ 20
11111000010010011110000010101010 _B (F849E0AA _H)	→ 5
10000000000000101010101010101010 _B (8002AAAA _H)	→ 1
11111111111111111111111111111111 _B (FFFFFFFF _H)	→ 32

Table 7.6-1 Bit Locations and Return Values (decimal)

Detected bit location	Return value	Detected bit location	Return value	Detected bit location	Return value	Detected bit location	Return value
31	0	23	8	15	16	7	24
30	1	22	9	14	17	6	25
29	2	21	10	13	18	5	26
28	3	20	11	12	19	4	27
27	4	19	12	11	20	3	28
26	5	18	13	10	21	2	29
25	6	17	14	9	22	1	30
24	7	16	15	8	23	0	31
						32 is returned when the detected bit is not found	

■ Save/Restore Processing

If it is necessary to save and restore the internal state of the bit search module, such as when the bit search module is used in an interrupt handler, use the following procedure:

1. Read the 1 detection data register and save its contents (save).
2. Use the bit search module.
3. Write the data saved in 1 to the 1 detection data register (restore).

With the above operation, the value obtained when the detection result register is read the next time corresponds to the value written to the bit search module before 1. If the data register written to last is the 0 detection or change point detection register, the value is restored correctly with the above procedure.

CHAPTER 8

16-BIT RELOAD TIMER

This chapter describes the 16-bit reload timer, the configuration and functions of registers, and 16-bit reload timer operation.

- 8.1 Overview
- 8.2 16-bit Reload Timer Block Diagram
- 8.3 16-bit Reload Timer Registers
- 8.4 Operation of 16-bit Reload Timer

8.1 Overview

The 16-bit reload timer consists of a 16-bit down counter, a 16-bit reload register, an internal counter, a prescaler for clock generation, and a control register.

The clock source can be selected from three internal clocks (machine clock divided by 2, 8, 32, 64 and 128) and external triggers.

The MB91260B series have three built-in channels for this reload timer.

No output to a pin of the reload timer 0.

Register List

Control status register (TMCSR:TMCSR0 to TMCSR2)

TMCSR0 Address 00004E _H	15	14	13	12	11	10	9	8	← Bit No.
TMCSR1 Address 000056 _H	—	—	—	CSL2	CSL1	CSL0	MOD2	MOD1
TMCSR2 Address 00005E _H	—	—	—	R/W	R/W	R/W	R/W	R/W
Read/Write →	—	—	—	R/W	R/W	R/W	R/W	R/W	
Initial Value →	(X)	(X)	(X)	(0)	(0)	(0)	(0)	(0)	
	7	6	5	4	3	2	1	0	← Bit No.
	MOD0	—	OUTL	RELD	INTE	UF	CNTE	TRG	
Read/Write →	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

16-bit timer register (TMR:TMR0 to TMR2)

TMR0 Address 00004A _H	15	14	13	12	11	10	9	8	← Bit No.
TMR1 Address 000052 _H								
TMR2 Address 00005A _H								
Read/Write →	R	R	R	R	R	R	R	R	
Initial Value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	
	7	6	5	4	3	2	1	0	← Bit No.
		—							
Read/Write →	R	R	R	R	R	R	R	R	
Initial Value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

16-bit reload register (TMRLR:TMRLR0 to TMRLR2)

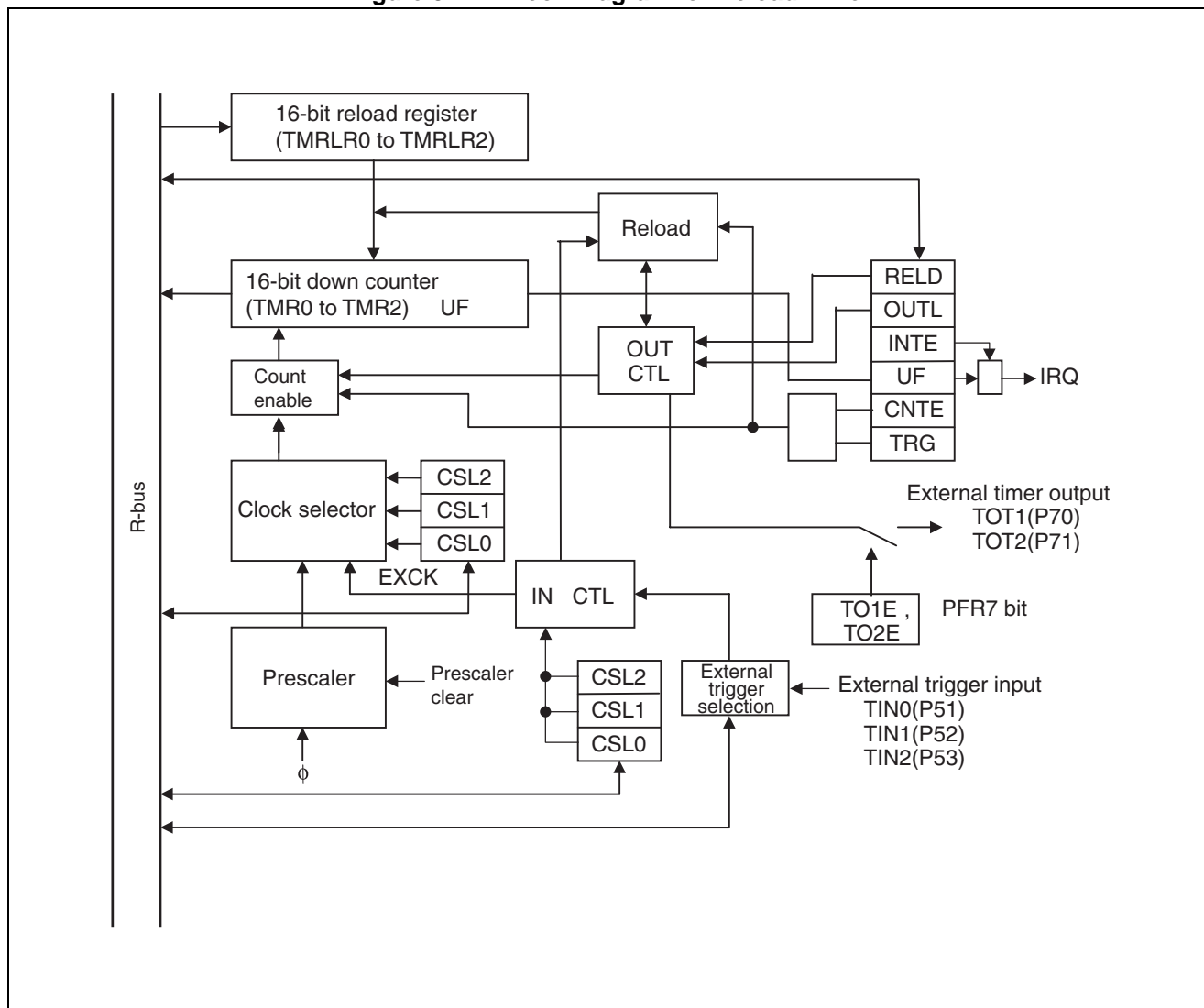
TMRLR0 Address 000048 _H	15	14	13	12	11	10	9	8	← Bit No.
TMRLR1 Address 000050 _H								
TMRLR2 Address 000058 _H								
Read/Write →	W	W	W	W	W	W	W	W	
Initial Value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	
	7	6	5	4	3	2	1	0	← Bit No.
		—							
Read/Write →	W	W	W	W	W	W	W	W	
Initial Value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

8.2 16-bit Reload Timer Block Diagram

A block diagram of the reload timer is shown below.

■ Reload Timer Block Diagram

Figure 8.2-1 Block Diagram for Reload Timer



8.3 16-bit Reload Timer Registers

This section describes the configuration and functions of reload timer registers.

■ Control Status Register (TMCSR:TMCSR0 to TMCSR2)

Control status register (TMCSR)									
Address	15	14	13	12	11	10	9	8	← Bit No.
TMCSR0: 00004E _H	–	–	–	CSL2	CSL1	CSL0	MOD2	MOD1	...
TMCSR1: 000056 _H	–	–	–	R/W	R/W	R/W	R/W	R/W	← Read/Write
TMCSR2: 00005E _H	(X)	(X)	(X)	(0)	(0)	(0)	(0)	(0)	← Initial value
	7	6	5	4	3	2	1	0	← Bit No.
...	MOD0	–	OUTL	RELD	INTE	UF	CNTE	TRG	
...	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

This register controls 16-bit timer operation modes and interrupts.

Rewrite bits other than UF, CNTE, and TRG only when CNTE = 0.

[bit12, bit11, bit10] CSL2, CSL1, CSL0 (Count source select)

These bits are count source select bits. Count sources can be selected from internal clocks and external triggers. Selectable count sources are as follows.

CSL2	CSL1	CSL0	Count source (ϕ : Machine clock)	$\phi=32$ MHz	$\phi=16$ MHz
0	0	0	Internal clock $\phi/2^1$ (Initial value)	62.5 ns	125 ns
0	0	1	Internal clock $\phi/2^3$	250 ns	0.5 μ s
0	1	0	Internal clock $\phi/2^5$	1.0 μ s	2.0 μ s
0	1	1	External trigger	–	–
1	0	0	Setting prohibited	–	–
1	0	1	Internal clock $\phi/2^6$	2.0 μ s	4.0 μ s
1	1	0	Internal clock $\phi/2^7$	4.0 μ s	8.0 μ s
1	1	1	ch.1 timer output (only ch.2 is settable)	ch.1	ch.1

The count effective edges are set using the MOD1 and MOD0 bits when external triggers are specified for count sources.

The minimum pulse width necessary for external triggers is $2 \times T$ (T: Machine clock cycle).

By using ch.1+ch.2 cascade connection, only ch.2 register can be set when CSL2, CSL1, CSL0="111_B". Setting is prohibited in ch.1.

[bit9, bit8, bit7] MOD2, MOD1, MOD0 (Mode)

These bits select the operation mode. The function is changed when the count source is "internal clock" or "external trigger".

- When internal clock mode: reload trigger setting
- When external trigger mode: count effective edge setting

Be sure to set MOD2 bit to "0".

[Reload trigger setting at internal clock selection]

When the internal clock is selected as the count source, it loads the content of reload registers and continues the count operation after the effective edge is inputted by setting MOD2 to MOD0 bits.

MOD2	MOD1	MOD0	Effective edge
0	0	0	Software trigger (initial value)
0	0	1	External trigger (rising edge)
0	1	0	External trigger (falling edge)
0	1	1	External trigger (both edges)
1	X	X	Setting prohibited

[Effective edge setting at external trigger selection]

When the external clock event is selected as the count source, it counts effective edge of the external trigger by setting of MOD2 to MOD0 bits.

MOD2	MOD1	MOD0	Effective edge
X	0	0	—
X	0	1	External trigger (rising edge)
X	1	0	External trigger (falling edge)
X	1	1	External trigger (both edges)
X	X	X	Setting prohibited

When external trigger is selected, the reload is generated by software trigger and underflow.

[bit6] This bit is unused.

The read value is always "0".

[bit5] OUTL

This bit sets external timer output level. The output level is opposite when this bit is "0" and "1".

[bit4] RELD

This bit is the reload enable bit. If it is set to "1", reload mode is entered. As soon as the counter value underflows from "0000_H" to "FFFF_H", the contents of the reload register are loaded into the counter and the count operation is continued.

If this bit is set to "0", one-shot mode is entered and the count operation is stopped when the counter underflows from "0000_H" to "FFFF_H".

TO1E, TO2E	OUTL	RELD	Output waveform
0	X	X	Output prohibited
1	0	0	"H" rectangular wave while counting
1	1	0	"L" rectangular wave while counting
1	0	1	"L" toggle output when counting starts
1	1	1	"H" toggle output when counting starts

TOxE is TO1E and TO2E in the PFR7 (Port Function Register). No output to a pin of the reload timer 0.

[bit3] INTE

This bit is the interrupt request enable bit. If the INTE bit is set to "1", an interrupt request is generated when the UF bit is set to "1". If it is set to "0", no interrupt request is generated.

[bit2] UF

This bit is the timer interrupt request flag. This bit is set to "1" when the counter value underflows from "0000_H" to "FFFF_H". Write "0" to this bit to clear it.

Writing "1" to this bit is meaningless.

When this bit is read by read-modify-write instructions, "1" is always read.

[bit1] CNTE

This bit is the count enable bit of the timer. Write "1" to this bit to enter the start trigger wait state. Write "0" to this bit to stop the count operation.

[bit0] TRG

This bit is the software trigger bit. Write "1" to this bit to generate a software trigger, load the contents of the reload register into the counter, and start the count operation.

Writing "0" to this bit is meaningless. The read value is always "0".

The trigger input to this register is valid only if CNTE=1. No operation occurs if CNTE=0.

■ TMR Register (16-bit Timer Register)

16-bit Timer Register (TMR)									
Address	15	14	13	12	11	10	9	8	← Bit No.
TMR0: 00004A _H									...
TMR1: 000052 _H	R	R	R	R	R	R	R	R	← Read/Write
TMR2: 00005A _H	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	← Initial value
	7	6	5	4	3	2	1	0	← Bit No.
...		–							...
...	R	R	R	R	R	R	R	R	← Read/Write
	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	← Initial value

This register can read the count value of the 16-bit timer. The initial value is undefined. Be sure to read this register using a 16-bit data transfer instruction.

■ TMRLR Register (16-bit Reload Register)

16-bit Reload Register (TMRLR)									
Address	15	14	13	12	11	10	9	8	← Bit No.
TMRLR0: 000048 _H									...
TMRLR1: 000050 _H	W	W	W	W	W	W	W	W	← Read/Write
TMRLR2: 000058 _H	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	← Initial value
	7	6	5	4	3	2	1	0	← Bit No.
...		—							...
	W	W	W	W	W	W	W	W	← Read/Write
	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	← Initial value

This register holds the initial value of a counter. The initial value is undefined. Be sure to read this register using a 16-bit data transfer instruction.

8.4 Operation of 16-bit Reload Timer

This section describes the reload timer operation.

■ Internal Clock Operation

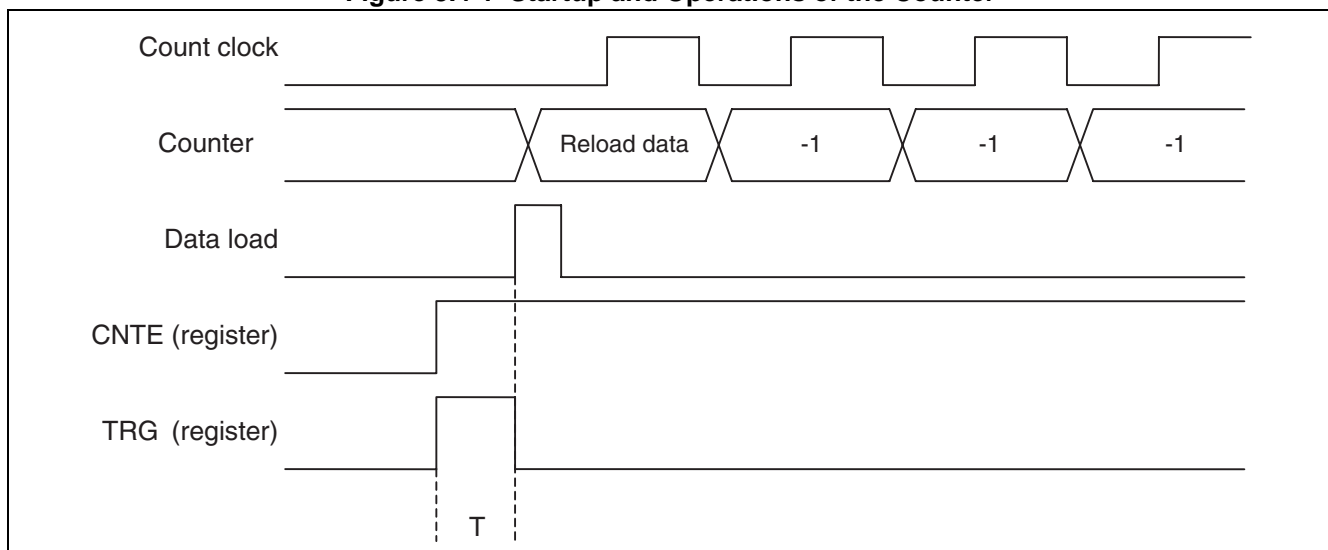
If the timer operates with a divide-by clock of the internal clock, one of the clocks generated by dividing the machine clock by 2, 8, 32, 64, or 128 can be selected as the clock source.

To start the count operation as soon as counting is enabled, write "1" to both CNTE and TRG bits of the control status register. Trigger input occurring due to the TRG bit is always valid regardless of the operating mode while the timer is running (CNTE=1).

Figure 8.4-1 shows the startup and operations of the counter.

After the counter start trigger is input, Time T (T: peripheral clock machine cycle) is required for loading data of the reload register into the counter.

Figure 8.4-1 Startup and Operations of the Counter



■ Underflow Operation

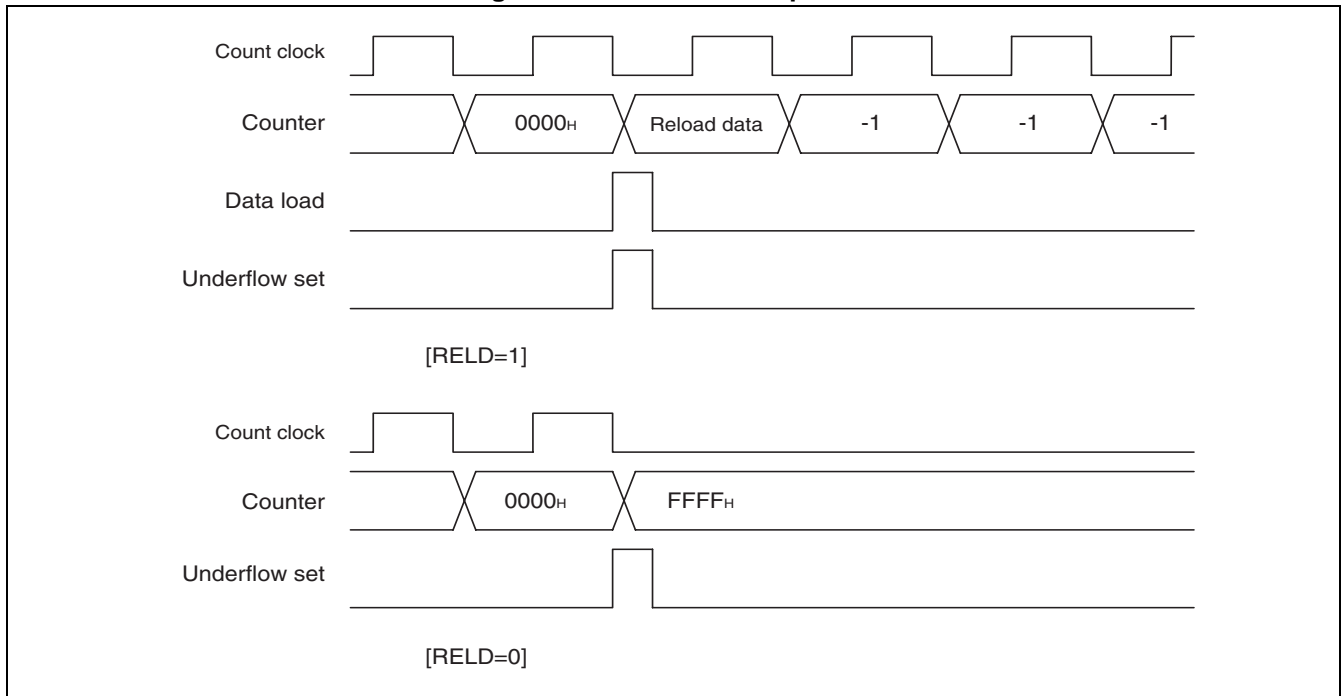
This timer defines an underflow as an event in which the counter value changes from "0000_H" to "FFFF_H". Thus, an underflow occurs at the count of [Reload register setting value + 1].

If the RELD bit of the control status register is set to "1" when an underflow occurs, the contents of the reload register are loaded into the counter and the count operation is continued. If the RELD bit is set to "0", the counter stops at "FFFF_H".

When an underflow occurs, it sets the UF bit of the control status register. If the INTE bit is set to "1" at this time, it generates an interrupt request.

Figure 8.4-2 shows the operation when an underflow occurs.

Figure 8.4-2 Underflow Operation



■ Output Terminal Functions

TOT1 and TOT2 output terminals perform as a toggle output inverting by underflows when reload mode, and as a pulse output indicating the counting process when one-shot mode. Output polarity can be set by OUTL bit of the register. When OUTL=0, an initial value of the toggle output is "0", and one-shot pulse output outputs "1" while counting. When setting OUTL=1, output waveform is inverted.

Figure 8.4-3 Output Terminal Function [RELD=1, OUTL=0]

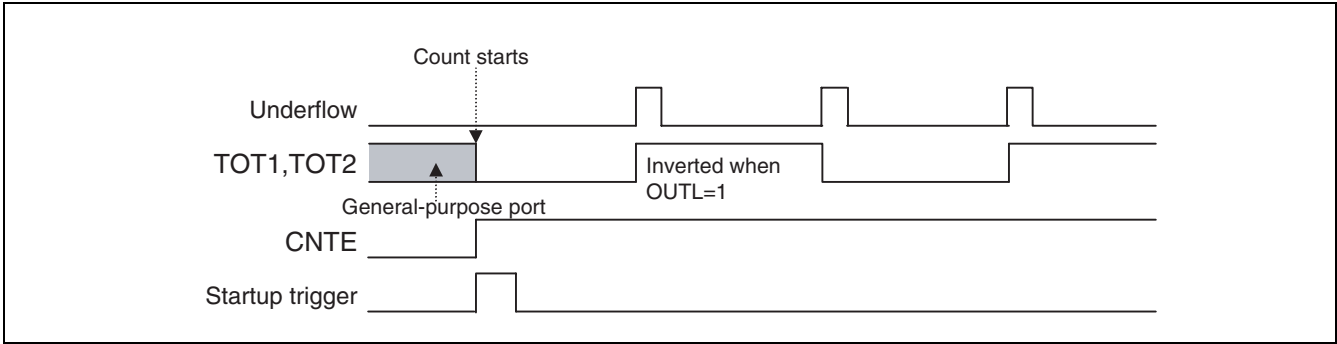
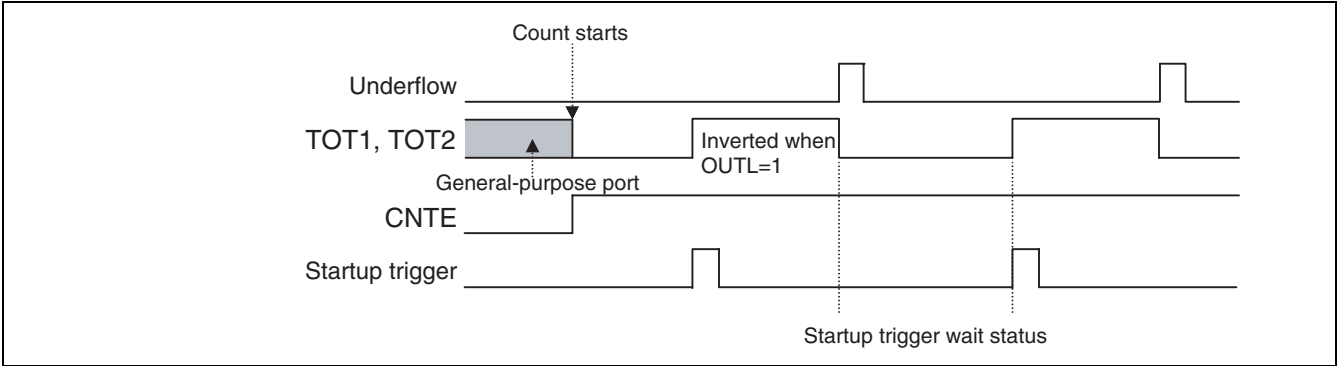


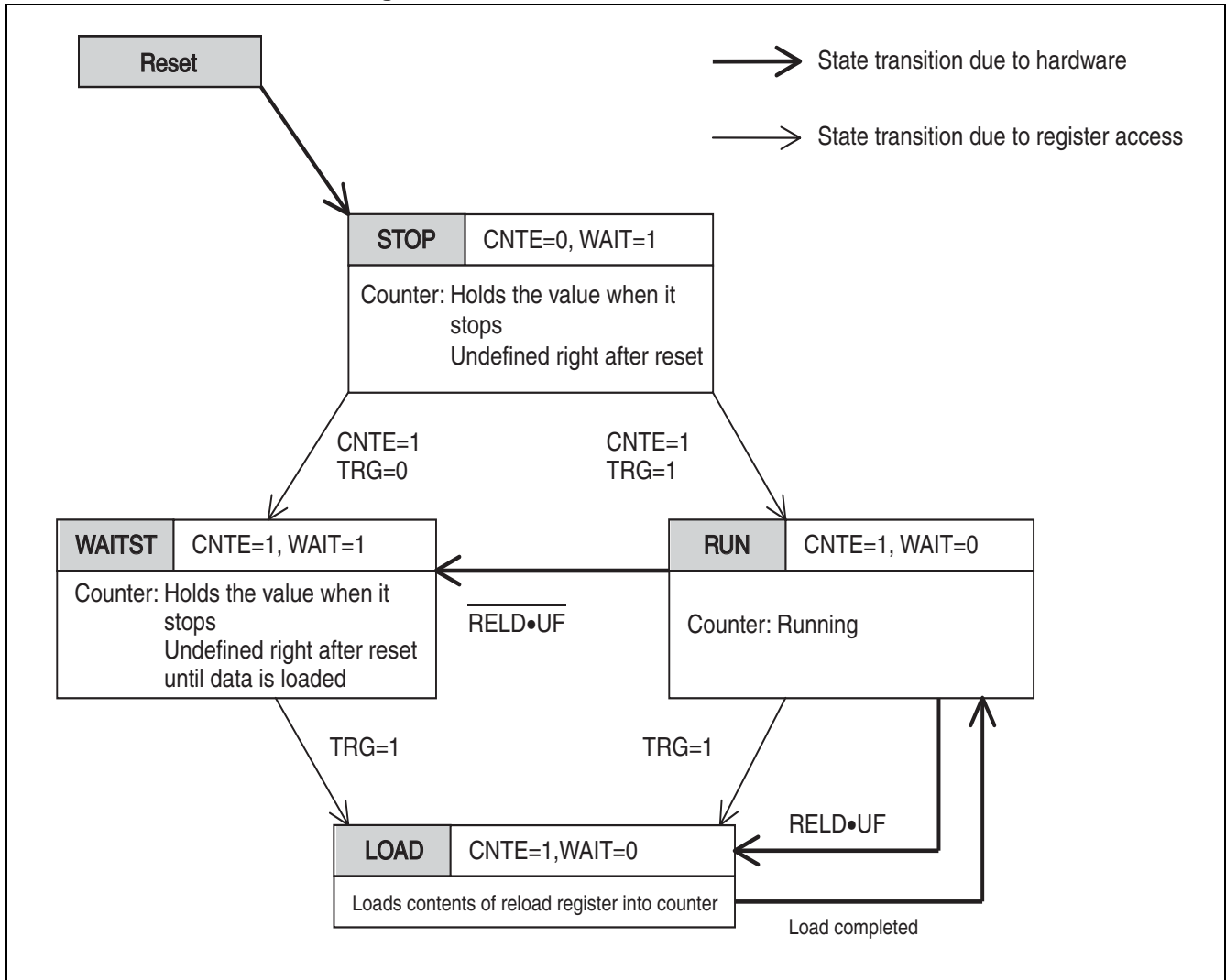
Figure 8.4-4 Output Pin Function [RELD=0, OUTL=0]



■ Operating States of the Counter

The CNTE bit of the control register and the internal signal WAIT determine the counter status. The states that can be set include the stop state, when CNTE=0 and WAIT=1 (STOP state); the startup trigger wait state, when CNTE=1 and WAIT=1 (WAIT status); and the operation state, when CNTE=1 and WAIT=0 (RUN state). Figure 8.4-5 shows the state transitions.

Figure 8.4-5 Status Transitions of Counter



■ Notes

- The internal prescaler is enabled if a trigger (soft trigger or external trigger) is applied when bit1 (timer enable: CNTE) of the control status register is set to "1".
- If the device attempts to set and clear the interrupt request flag at the same time, the flag is set preferentially and the clear operation becomes ineffective.
- If the device attempts to write to the 16-bit reload timer register and reload the data into the 16-bit reload timer register at the same time, old data is loaded into the counter. New data is loaded into the counter at the next reload timing.
- If the device attempts to load and count the 16-bit timer register at the same time, the load (reload) operation takes precedence.

CHAPTER 9

PPG (Programmable Pulse Generator)

This chapter describes the overview of the PPG (Programmable Pulse Generator) timer, the configuration and functions of registers, and the operation of the PPG timer.

- 9.1 Overview
- 9.2 Block Diagram
- 9.3 Register of PPG
- 9.4 Operation Explanation

9.1 Overview

PPG, the 8-bit reload timer module, performs the PPG output by the pulse output control according to timer operation.

The hardware consists of 16 8-bit down counters, 32 8-bit reload registers, control register, 16 external pulse outputs, and 16 interrupt outputs.

The MB91260B series has 16 channel for 8-bit PPG and 8 channel for 16-bit PPG.

■ Function of PPG

- 8-bit PPG output independent operation mode

Independent PPG output operation can be performed.

- 16-bit PPG output operation mode

One channel 16-bit PPG output can be operated.

- 8 + 8-bit PPG output operation mode

With setting the $ch(n + 1)$ output as the $ch(n)$ clock input, the 8-bit PPG output in any cycle can be operated. ($n=0, 2, 4, 6, 8, 10, 12, 14$)

- 16 + 16-bit PPG output operation mode

This mode sets the 16-bit prescaler output, $ch(n + 3) + ch(n + 2)$ as a clock input for the 16-bit PPG, $ch(n + 1) + ch(n)$. ($n=0, 4, 8, 12$)

- PPG output operation

Outputs a pulse waveform with any period and duty ratio.

Can also be used in conjunction with an external circuit to form a D/A converter.

- Output invert function

The PPG output value can be inverted.

■ Register List

- PPG start register (TRG)

Address: 000130H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	←Bit No.
	PEN15	PEN14	PEN13	PEN12	PEN11	PEN10	PEN09	PEN08	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	←Bit No.
	PEN07	PEN06	PEN05	PEN04	PEN03	PEN02	PEN01	PEN00	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

- Output invert register (REVC)

Address: 000134 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	←Bit No.
	REV15	REV14	REV13	REV12	REV11	REV10	REV09	REV08	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	←Bit No.
	REV07	REV06	REV05	REV04	REV03	REV02	REV01	REV00	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

- GATE function control register (GATEC)

Address: 000133 _H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	←Bit No.
	-	-	-	-	-	-	STGR	EDGE	
Read/Write →	-	-	-	-	-	-	R/W	R/W	
Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(0)	(0)	

- PPG 0 to PPG15 operation mode control register (PPGC0 to PPGC15)

Address:	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	PIEn	PUFn	INTMn	PCS1	PCS0	MD1*	MD0*	-	
ch0 :000108 _H									
ch1 :000109 _H									
ch2 :00010A _H									
ch3 :00010B _H									
ch4 :000114 _H									
ch5 :000115 _H									
ch6 :000116 _H									
ch7 :000117 _H									
ch8 :000120 _H									
ch9 :000121 _H									
ch10 :000122 _H									
ch11 :000123 _H									
ch12 :00012C _H									
ch13 :00012D _H									
ch14 :00012E _H									
ch15 :00012F _H									

Read/Write → R/W R/W R/W R/W R/W R/W R/W R/W -

Initial value → (0) (0) (0) (0) (0) (0) (0) (0) (X)

n = 0 to 15

*: MD1 and MD0 exist only in even-numbered channel, but they do not exist in odd-numbered channel. The initial value of odd-numbered channel is undefined. Writing to them is meaningless.

● Reload register: 8-bit PPG mode

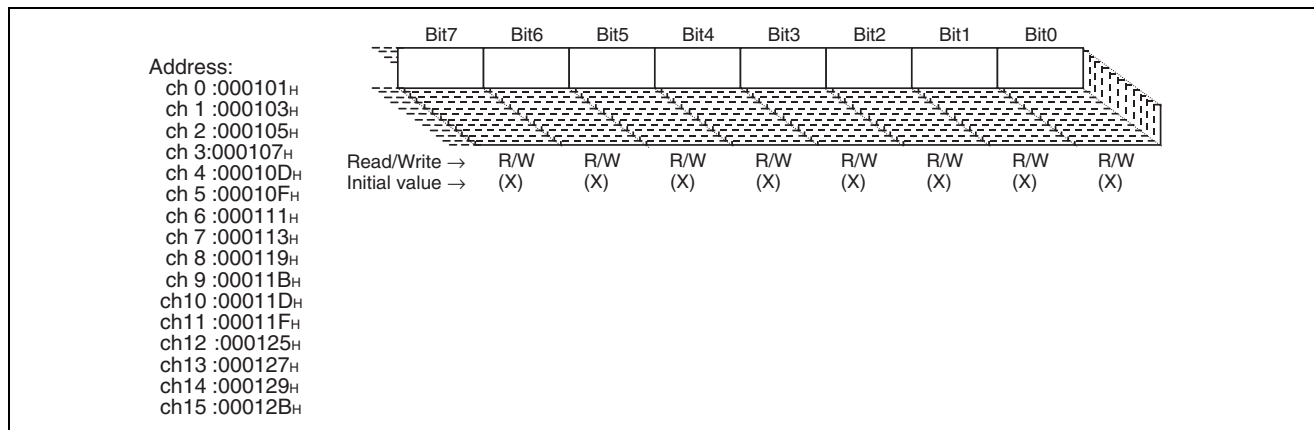
- Reload Registers H (PRLH0 to PRLH15)

Address:	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
ch 0 :000100 _H									
ch 1 :000102 _H									
ch 2 :000104 _H									
ch 3 :000106 _H									
ch 4 :00010C _H									
ch 5 :00010E _H									
ch 6 :000110 _H									
ch 7 :000112 _H									
ch 8 :000118 _H									
ch 9 :00011A _H									
ch10 :00011C _H									
ch11 :00011E _H									
ch12 :000124 _H									
ch13 :000126 _H									
ch14 :000128 _H									
ch15 :00012A _H									

Read/Write → R/W R/W R/W R/W R/W R/W R/W R/W

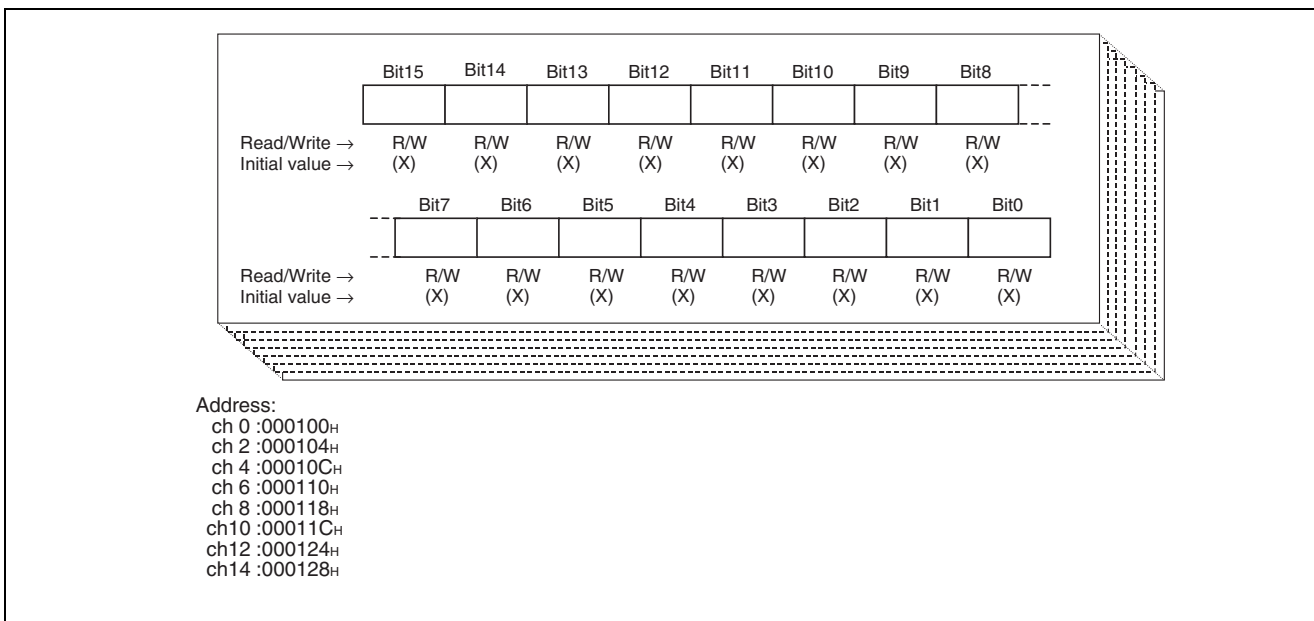
Initial value → (X) (X) (X) (X) (X) (X) (X) (X)

- Reload Registers L (PRLL0 to PRLL15)

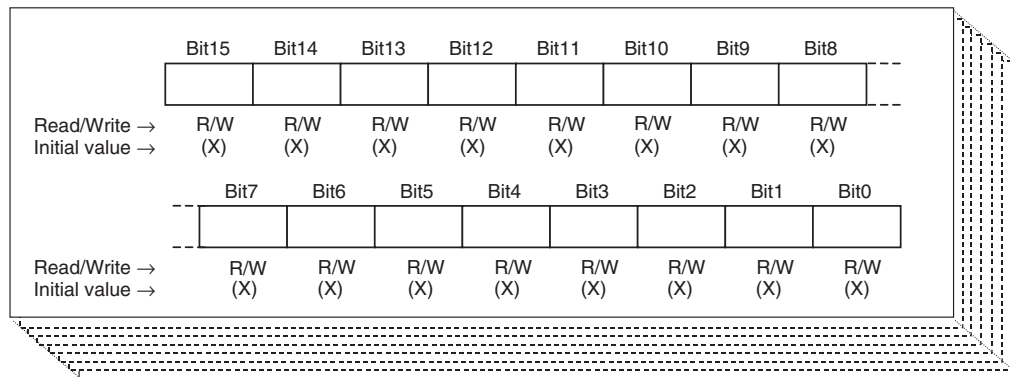


● Reload register: 16-bit PPG mode

- Reload Registers H (PRLH0, PRLH2, PRLH4, PRLH6, PRLH8, PRLH10, PRLH12, PRLH14)



- Reload Registers L (PRLL0, PRLL2, PRLL4, PRLL6, PRLL8, PRLL10, PRLL12, PRLL14)



Address:

ch 0 :000102H
 ch 2 :000106H
 ch 4 :00010EH
 ch 6 :000112H
 ch 8 :00011AH
 ch10 :00011EH
 ch12 :000126H
 ch14 :00012AH

9.2 Block Diagram

This section explains the PPG block diagram.

■ Block Diagram

Figure 9.2-1 Block Diagram of 8-bit PPG (ch0, ch2, ch4, ch6, ch8, ch10, ch12, ch14)

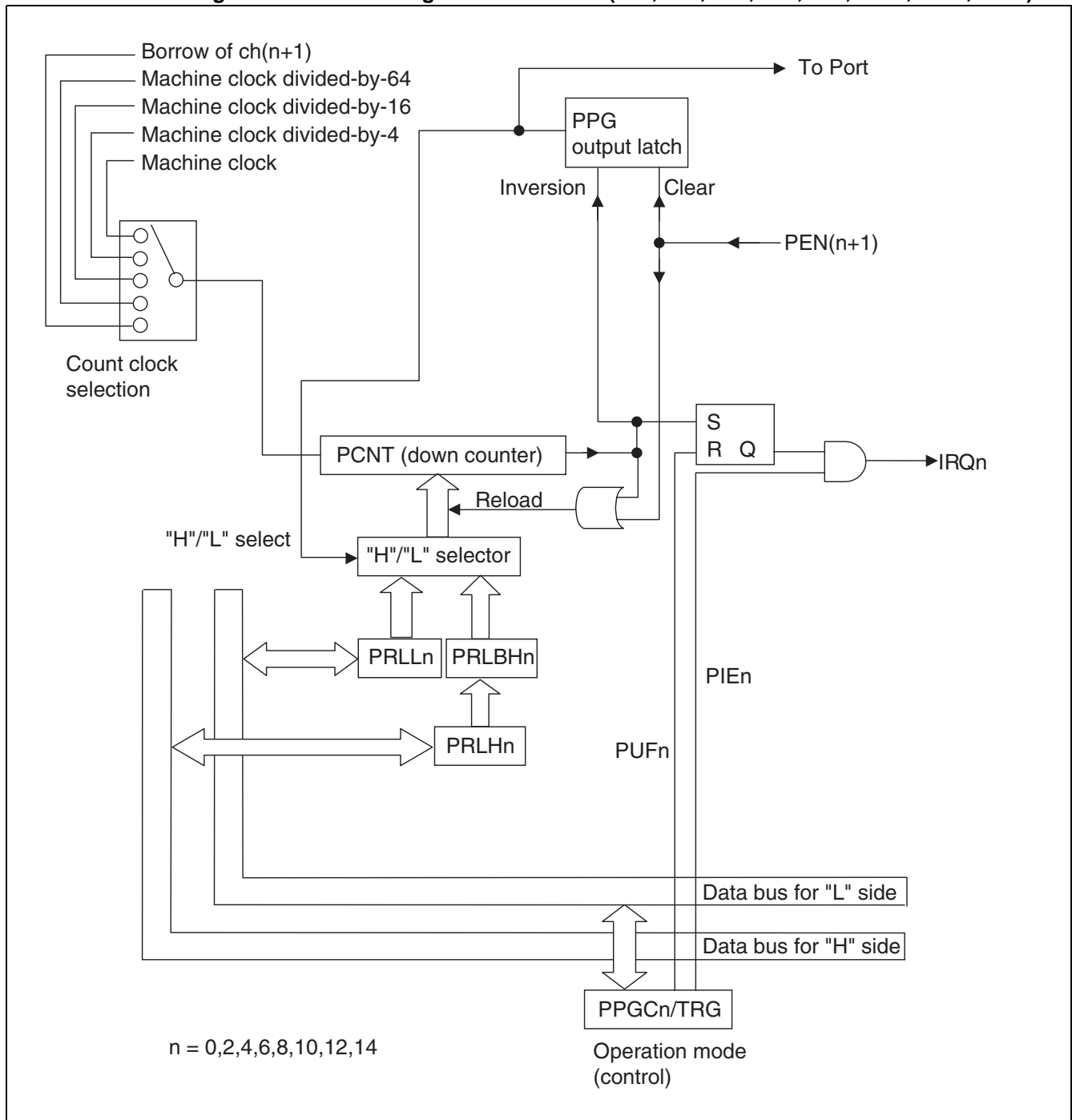


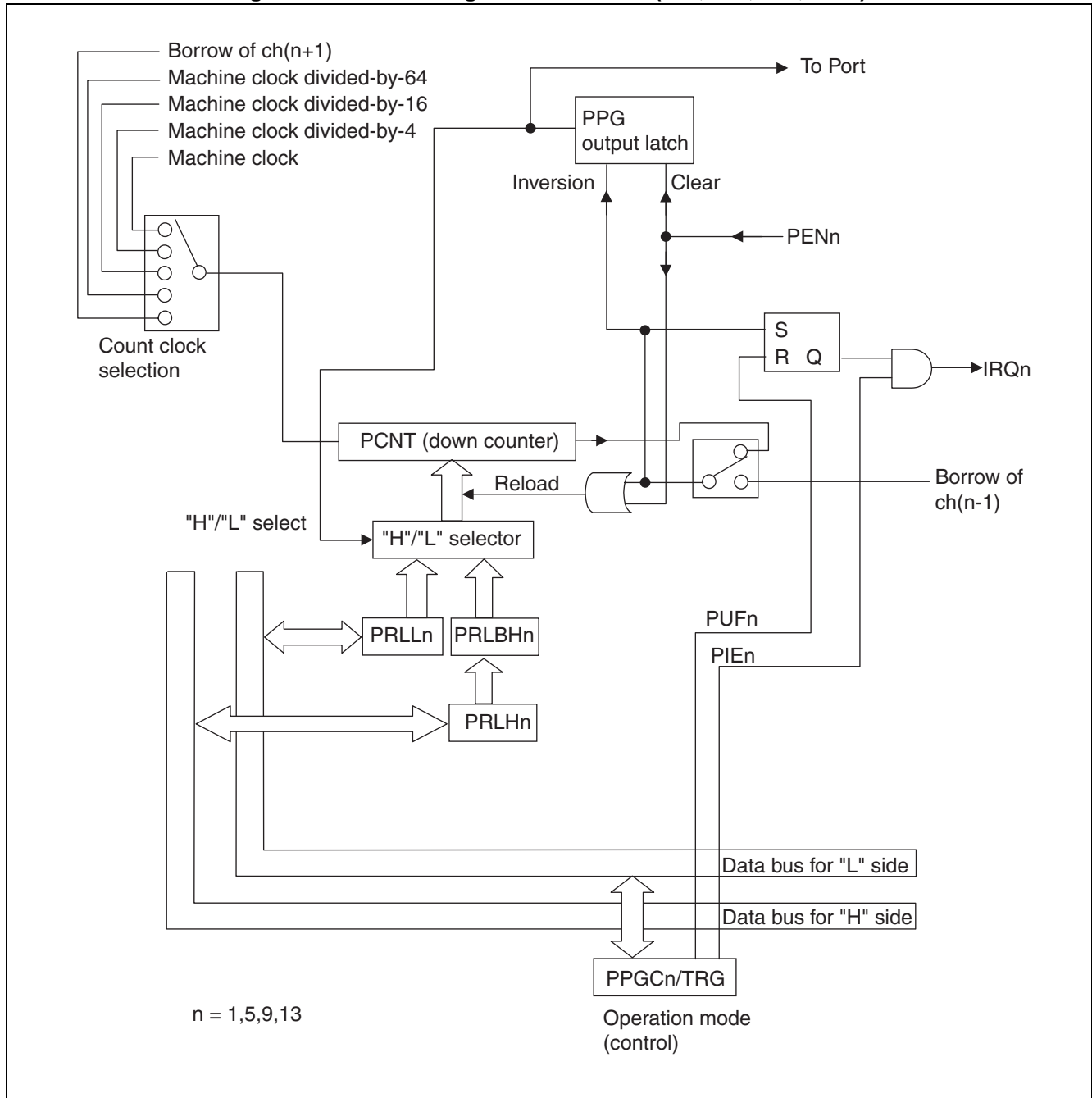
Figure 9.2-2 Block Diagram of 8-bit PPG (ch1, ch5, ch9, ch13)

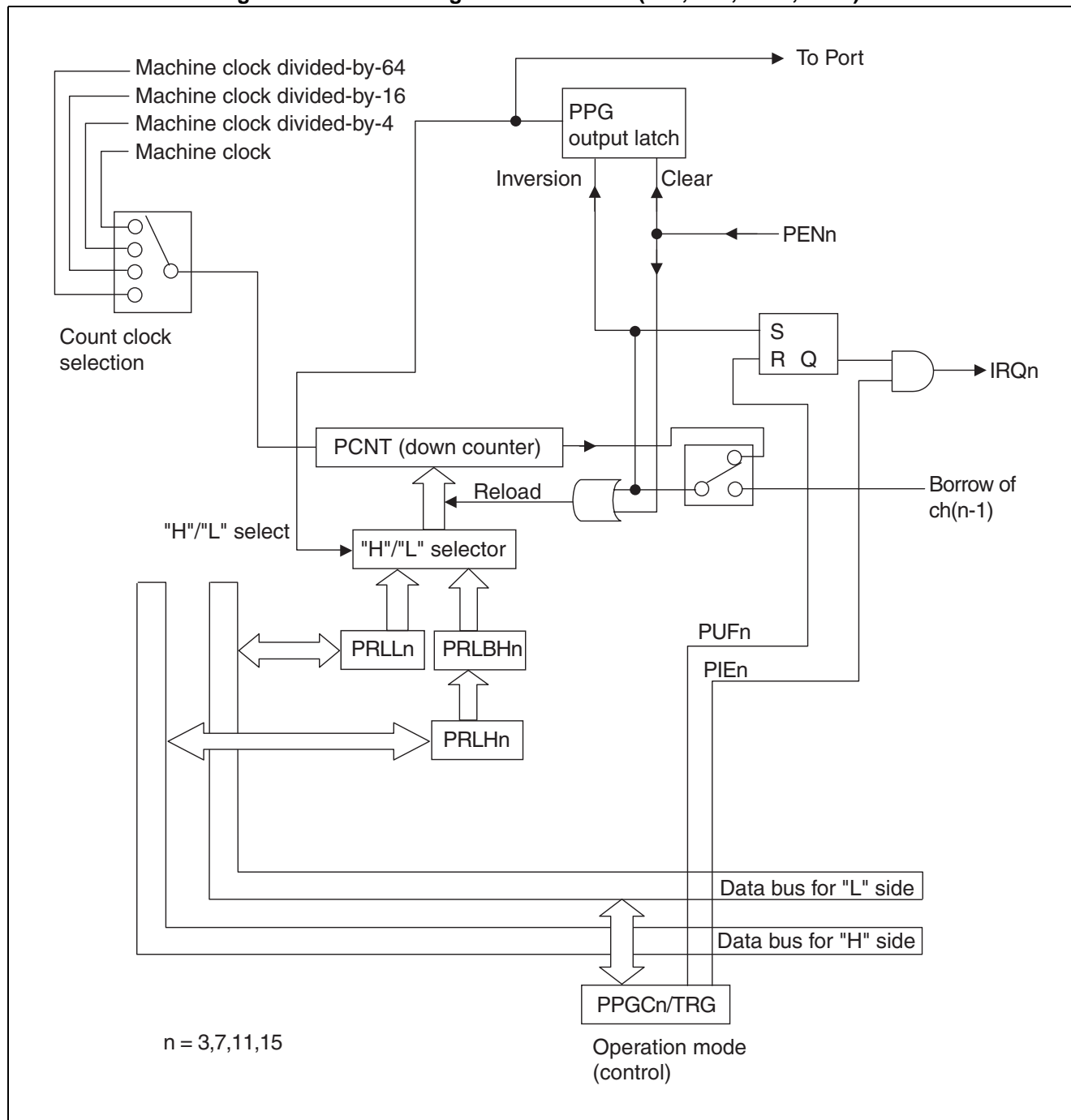
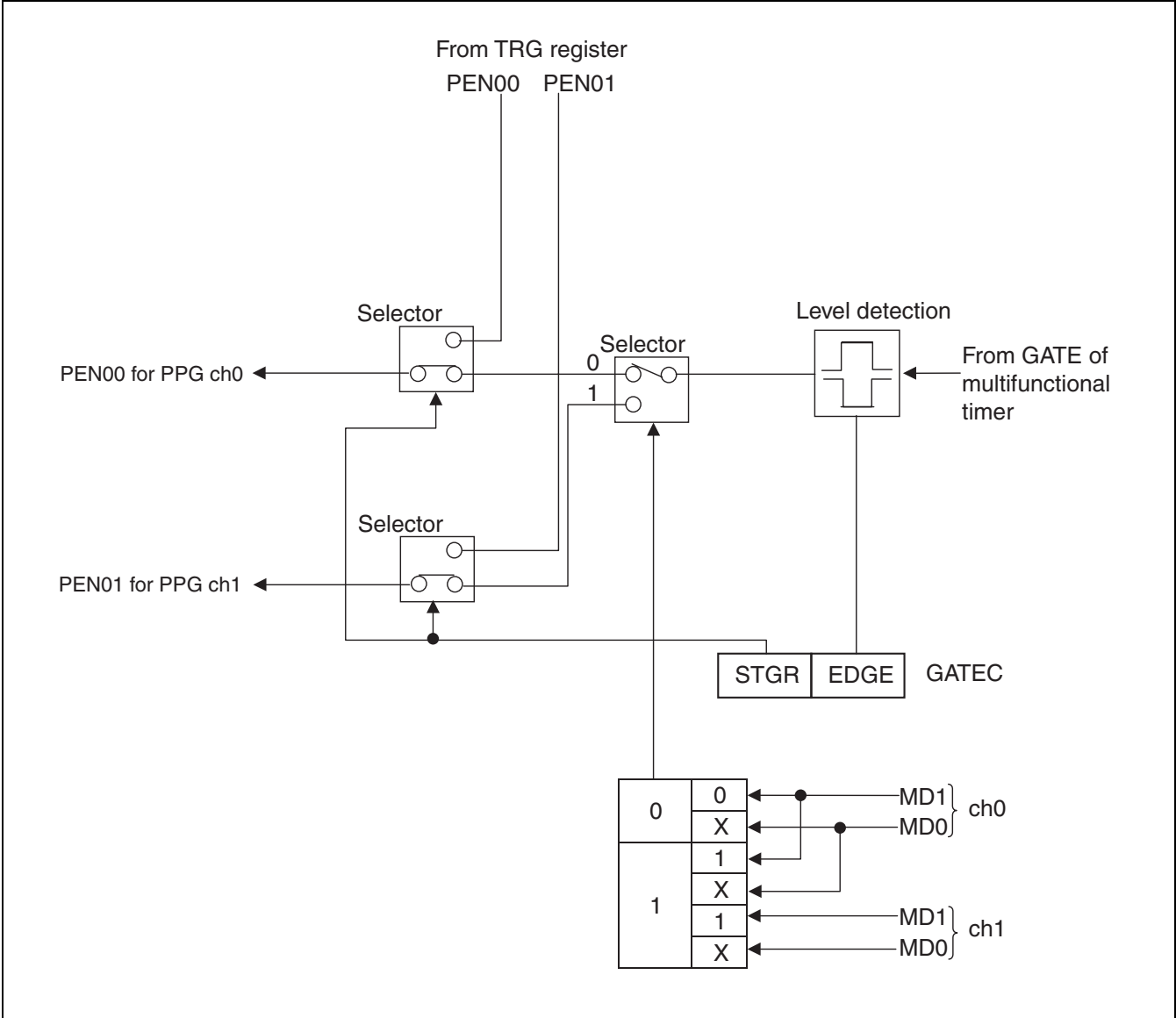
Figure 9.2-3 Block Diagram of 8-bit PPG (ch3, ch7, ch11, ch15)

Figure 9.2-4 Block Diagram of Gate Function



9.3 Register of PPG

This section describes the PPG registers.

■ PPGCn Register (PPGn Operation Mode Control Register)

Figure 9.3-1 PPGCn Register (PPGn Operation Mode Control Register)
n=0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

PPGC0 to PPGC15 operation mode control register (PPGCn) n = 0 to 15									
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Address:		PIEn	PUFn	INTMn	PCS1	PCS0	MD1	MD0	-
ch 0	:000108 _H	Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	-
ch 1	:000109 _H								
ch 2	:00010A _H	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(X)
ch 3	:00010B _H								
ch 4	:000114 _H								
ch 5	:000115 _H								
ch 6	:000116 _H								
ch 7	:000117 _H								
ch 8	:000120 _H								
ch 9	:000121 _H								
ch10	:000122 _H								
ch11	:000123 _H								
ch12	:00012C _H								
ch13	:00012D _H								
ch14	:00012E _H								
ch15	:00012F _H								

[bit7] PIEn (Ppg Interrupt Enable): PPG interrupt enable bit

This bit controls a PPG interrupt enable as described below.

0	Disables the interrupt.
1	Enables the interrupt.

- If this bit is set to "1", an interrupt request is generated when the PUFn is set to "1".
- If this bit is set to "0", no interrupt request is generated.
- Initialized to "0" by reset.
- The read / write is possible.

[bit6] PUFn (Ppg Underflow Flag): PPG counter underflow bit

This bit controls PPG counter underflow bit as described below.

0	PPG counter underflow has not been detected.
1	PPG counter underflow has been detected.

- In the 8-bit PPG 2ch mode and the 8-bit prescaler + 8-bit PPG mode, this bit is set to "1" when the count value of ch0 underflows from "00_H" to "FF_H".
- In the 16-bit PPG 1ch mode, this bit is set to "1" when the count value of ch1/ch0 underflows from "0000_H" to "FFFF_H".
- Writing "0" clears the bit to "0".

- Writing "1" to this bit is meaningless.
- When this bit is read to a read modify write instruction, 1 is always read.
- Initialized to "0" by reset.
- The read /write is possible.

[bit5] INTMn (Interrupt Mode): Interrupt mode bit

This bit can limit the PUFn bit detection at an underflow only from PRLBHn.

0	PUFn is set to 1 at an underflow.
1	PUFn is set to 1 at an underflow only from PRLBHn.

- Initialized to "0" by reset.
- The read / write is possible.
- If this bit is set to "1", an interrupt is enabled at one cycle output of PPG waveform.
- Do not rewrite this bit when the interrupt is allowed.

[bit4, bit3] PCS1/PCS0(PPG Count Select): Count clock select bits

These bits are used to select the down counter operating clock as shown below.

PCS1	PCS0	Operating Mode
0	0	Machine clock (62.5 ns machine clock at 16 MHz)
0	1	Machine clock / 4 (250 ns machine clock at 16 MHz)
1	0	Machine clock / 16 (1 μ s machine clock at 16 MHz)
1	1	Machine clock / 64 (4 μ s machine clock at 16 MHz)

- Initialized to "00_B" by reset.
- The read / write is possible.

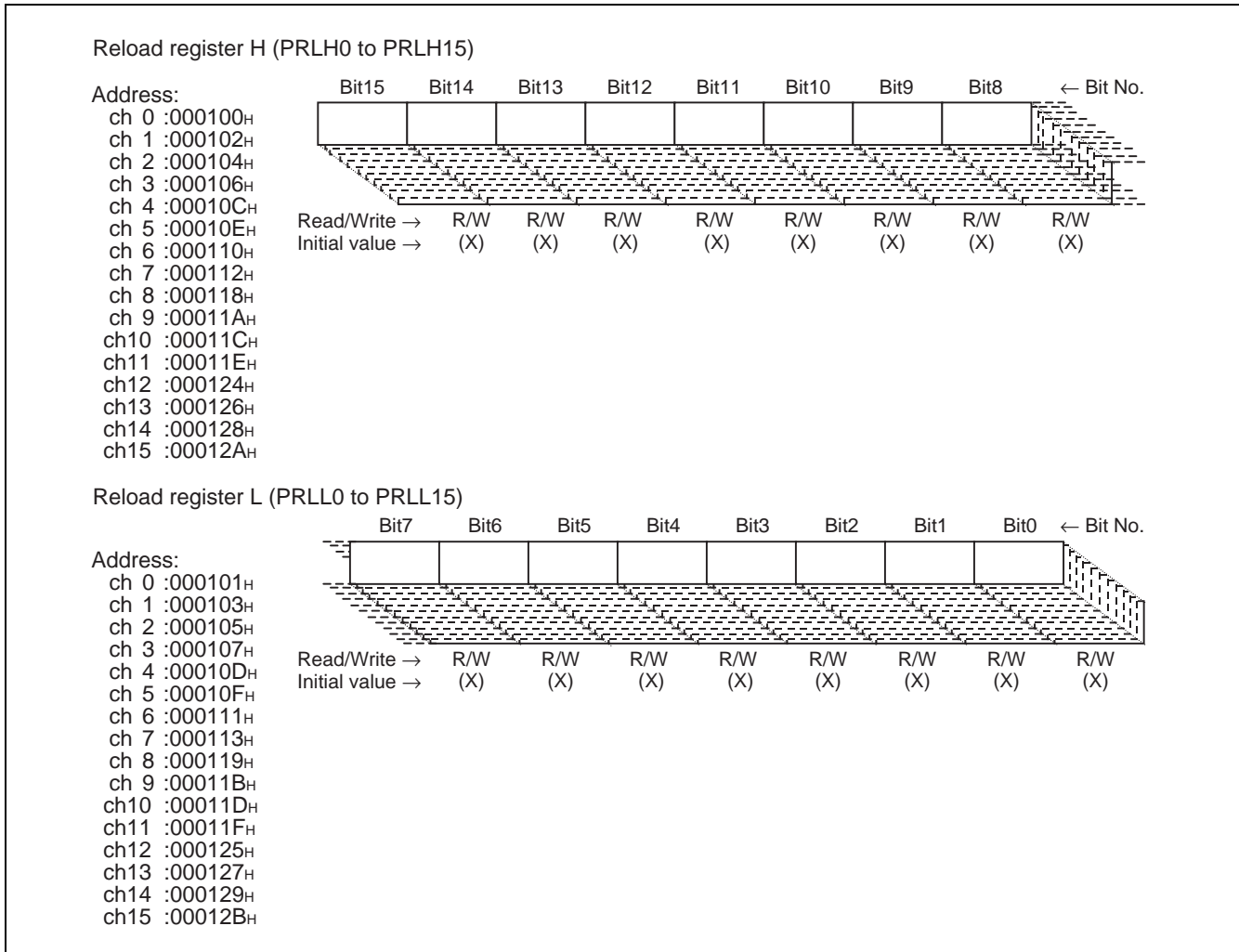
[bit2, bit1] MD1/MD0(ppg count MoDe): Operation mode select bits

These bits are used to select the PPG timer operation mode as shown below.

MD1	MD0	Operating Mode
0	0	8-bit PPG 2ch independent mode
0	1	8-bit prescaler + 8-bit PPG mode
1	0	16-bit PPG mode
1	1	16-bit prescaler + 16-bit PPG mode

- Initialized to "00_B" by reset.
- The read / write is possible.
- These bits exist only in even-numbered channels.

■ PRLH/PRLH Register (Reload Register)



These registers hold reload values for a down counter PCNT. Each register has own role as shown below.

Register Name	Function
PRL	Hold reload value at "L" side
PRLH	Hold reload value at "H" side

Each register can read or write.

Note:

When this register used in the 8-bit prescaler + 8-bit PPG mode and the 16-bit prescaler + 16-bit PPG mode, PPG waveforms might vary from cycle to cycle if the different values are set to the PRL and the PRLH on the prescaler side. Therefore, it is recommended to set the same value to the PRL and the PRLH on the prescaler side.

■ TRG Register (PPG Activation Register)

PPG activation register (TRG)								
Address: 000130 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8 ← Bit No.
	PEN15	PEN14	PEN13	PEN12	PEN11	PEN10	PEN09	PEN08
	Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0 ← Bit No.
	PEN07	PEN06	PEN05	PEN04	PEN03	PEN02	PEN01	PEN00
	Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)

[bit15 to bit0] PEN15 to PEN0 (Ppg ENable): PPG Operation Enable bits

These bits are used to select the PPG operation start and the operation mode as shown below.

PEN15 to PEN0	Operating State
0	Operation stop ("L" level output maintenance)
1	PPG enabling operations

- Initialized to "0" by reset.
- The read / write is possible.

■ REVC Register (Output Invert Register)

Output invert register (REVC)								
Address: 000134 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8 ← Bit No.
	REV15	REV14	REV13	REV12	REV11	REV10	REV09	REV08
	Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0 ← Bit No.
	REV07	REV06	REV05	REV04	REV03	REV02	REV01	REV00
	Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)

[bit15 to bit0] REV15 to REV0: Output inversion bits

The PPG output values including the initial levels are inverted.

REV15 to REV0	Output level
0	Normal
1	Inversion

- Initialized to "0" by reset.
- The read / write is possible.
- Since these bits invert the PPG outputs, they also invert the initial levels. The relationship between the reload register "L" and "H" is reversed as well.

■ GATEC Register (GATE Function Control Register)

GATE function control register (GATEC)								
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0 ← Bit No.
Address: 000133 _H	-	-	-	-	-	-	STGR	EDGE
Read/Write →	-	-	-	-	-	-	R/W	R/W
Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(0)	(0)

[bit1] STGR: Gate function select bit

This bit selects whether to use the start signal from a multifunctional timer or to start by the TRG register.

STGR	Operating mode
0	Start by TRG register
1	Start by a start signal from the multifunctional timer

- Initialized to "0" by reset.
- The read / write is possible.

[bit0] EDGE: Start valid edge select bit

This bit is used to select the start valid edge from the multifunctional timer as shown below.

EDGE	Operating mode
0	Rising start → Falling stop ^{*1}
1	Falling start → Rising stop ^{*2}

- Initialized to "0" by reset.
- The read / write is possible.

*1: Start at "H" period.

*2: Start at "L" period.

9.4 Operation Explanation



PPG has 16 channels for the 8-bit length PPG unit and can operate four modes in total by combining: independent mode, 8-bit prescaler + 8-bit PPG mode, 16-bit PPG 1 channel mode, and 16-bit prescaler + 16-bit PPG mode.

■ Overview of Operation

Each of 8-bit length PPG units has two 8-bit-length reload registers for the "L" and "H" sides (PRL, PRLH). The value written to this register is reloaded into "L"/"H" sides of 8-bit down counter (PCNT) alternately, counted down based on the count clock, and inverted a pin output (PPG) at reloading when a counter borrow occurs. With this operation, the pin output (PPG) becomes a pulse output, which has "L"/"H" width corresponding to the value of reload register.

The operation starts/restarts when the bit of the register is written.

The relationship between the reload operation and the pulse output is shown below.

Reload Operation	Terminal Output Change
PRLH → PCNT	PPGn [0 → 1] 
PRL → PCNT	PPGn [1 → 0] 

n=0 to 15

Also, if the bit7: PIEn of PPGCn register is set to "1", an interrupt request outputs by borrow to the counter value from "00_H" to "FF_H" (borrow from "0000_H" to "FFFF_H" in 16-bit PPG mode).

■ Operating Mode

There are four operation modes: independent mode, 8-bit prescaler + 8-bit PPG mode, 16-bit PPG 1 channel mode, and 16-bit prescaler + 16-bit PPG mode.

- In the independent mode, a channel can operate as 8-bit PPG independently. The PPG output of ch(n) is connected to PPG(n) pin. (n = 0 to 15)
- The 8-bit prescaler + 8-bit PPG mode makes 1 channel operate as an 8-bit prescaler, counts by its borrow output, and then allows the 8-bit PPG waveform in any cycle to be outputted. For example, the prescaler output of ch1 is connected to the PPG1 pin; the PPG output of ch0 is connected to the PPG0 pin.
- In the 16-bit PPG 1 channel mode, two channels are combined, and the combined channel operates as 16-bit PPG. For example, if ch0 and ch1 are combined, 16-bit PPG outputs are connected to both PPG0 pin and PPG1 pin.

■ PPG Output Operation

PPG activates this block and starts counting when the bits of each channel on the TRG register (PPG activation register) are set to "1". After operation starts, the count operation is stopped when each channel bit of TRG register is set to "0". After having stopped, the pulse output holds "L" level.

Do not set the PPG channel as the operating state, with the prescaler channel as the stopped state, in the 8-bit prescaler + 8-bit PPG mode and the 16-bit prescaler + 16-bit PPG mode.

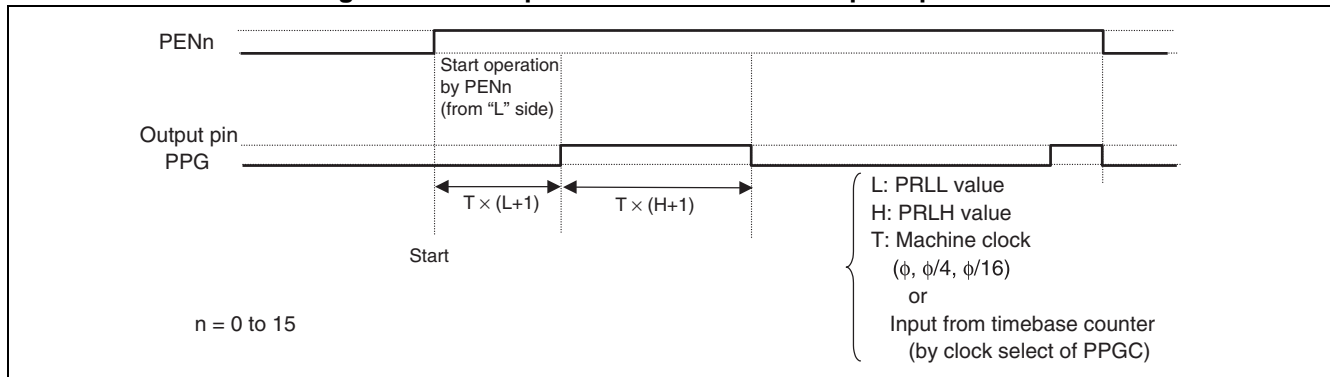
In the 16-bit PPG mode, control the start and stop for PENn of each channel on TRG register simultaneously (n = 0 to 15).

PPG output operation is explained below.

In PPG operation, the pulse wave with any frequency/duty ratio (the ratio between "H" level period and "L" level period in pulse wave) is outputted continuously. If the pulse wave output is started, PPG will not stop it before operation stop is set.

Figure 9.4-1 shows the output waveform of PPG operation.

Figure 9.4-1 Output Waveform of PPG Output Operation



■ Relation between Reload Value and Pulse Width

The pulse width to be outputted is the value that multiplies the cycle of the count clock by the value in the reload register plus 1. Note that the pulse width will be one cycle of the count clock when the reload register value is set to "00_H" at operating the 8-bit PPG and when the reload register value is set to "0000_H" at operating the 16-bit PPG. Furthermore, the pulse width will be 256 cycles of the count clock when the reload register value is set to "FF_H" at operating the 8-bit PPG and will be 65536 cycles of the count clock when the reload register value is set to "FFFF_H" at operating the 16-bit PPG.

The equations for calculating the pulse width are shown below:

$$\begin{aligned} Pl &= T \times (L + 1) \\ Ph &= T \times (H + 1) \end{aligned} \quad \left\{ \begin{array}{l} L: \text{PRL value} \\ H: \text{PRLH value} \\ T: \text{input clock cycle} \\ Ph: \text{high pulse width} \\ Pl: \text{low pulse width} \end{array} \right.$$

Count Clock Selection

The count clock to be used for this block operation uses the input for the peripheral clock and timebase counter and can be selected from one of the following four types of count clock inputs.

The count clock operates as shown below.

PPGC0 to PPGC15 Registers		Count Clock Operation
PCS1	PCS0	
0	0	One count per peripheral clock
0	1	One count per 4 cycles of peripheral clock
1	0	One count per 16 cycles of peripheral clock
1	1	One count per 64 cycles of peripheral clock

Note that the first count cycle may become out of synchronization if the PPG side is started, in the 8-bit prescaler + 8-bit PPG mode and the 16-bit prescaler + 16-bit PPG mode, and when the prescaler side is in the operating state and the PPG side is in the stop state.

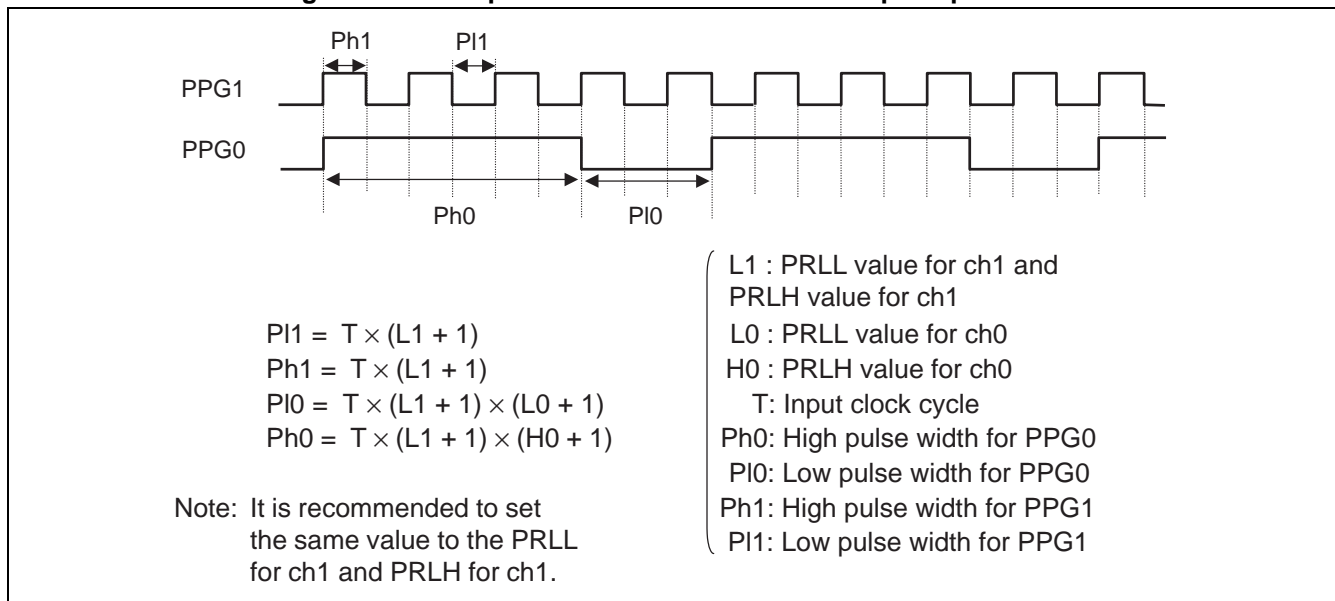
Control of Pulse Pin Output

The pulse output generated by operating this module can be outputted from external pins (PPG0 to PPG15).

In the 16-bit PPG mode, the same output can be obtained from allowing either external pin output because PPG (m) and PPG (m + 1) output the same waveform. (m = 0, 2, 4, 6, 8, 10, 12, 14)

In the 8-bit prescaler + 8-bit PPG mode and the 16-bit prescaler + 16-bit PPG mode, the 8-bit prescaler toggle waveform is outputted on the prescaler side, and the 8-bit PPG waveform is outputted on the PPG side. Figure 9.4-2 shows an example of the output waveform in this mode.

Figure 9.4-2 Output Waveform of 8 + 8 PPG Output Operation



■ Interruption

The interrupt on this module becomes active when a reload value is counted out and a borrow occurs. However, when INTMn bit is set to "1", the interrupt becomes active only at underflow (borrow) from PRLBHn. That is, the interrupt occurs when "H" width pulse ends.

In the 8-bit PPG mode and the 8-bit prescaler + 8-bit PPG mode, each counter borrow performs own interrupt request. In the 16-bit PPG mode and the 16-bit prescaler + 16 bit PPG mode, 16-bit counter borrow sets PUF(m) and PUF (m + 1) simultaneously. For this reason, it is recommended that either PIE (m) or PIE (m + 1) is enabled in order to unify the interrupt causes. Also, it is recommended to simultaneously set PUF (m) and PUF (m + 1) for clearing the interrupt causes. (m = 0, 2, 4, 6, 8, 10, 12, 14)

■ GATE Function

By using the multifunctional timer signal, PPG can be: started-> stopped.

- In the 8-bit PPG mode and the 8-bit prescaler + 8-bit PPG mode, this function can activate the PPG ch0.
- In the 16-bit PPG mode and the 16-bit prescaler + 16-bit PPG mode, this function can activate the PPG ch0, ch1.

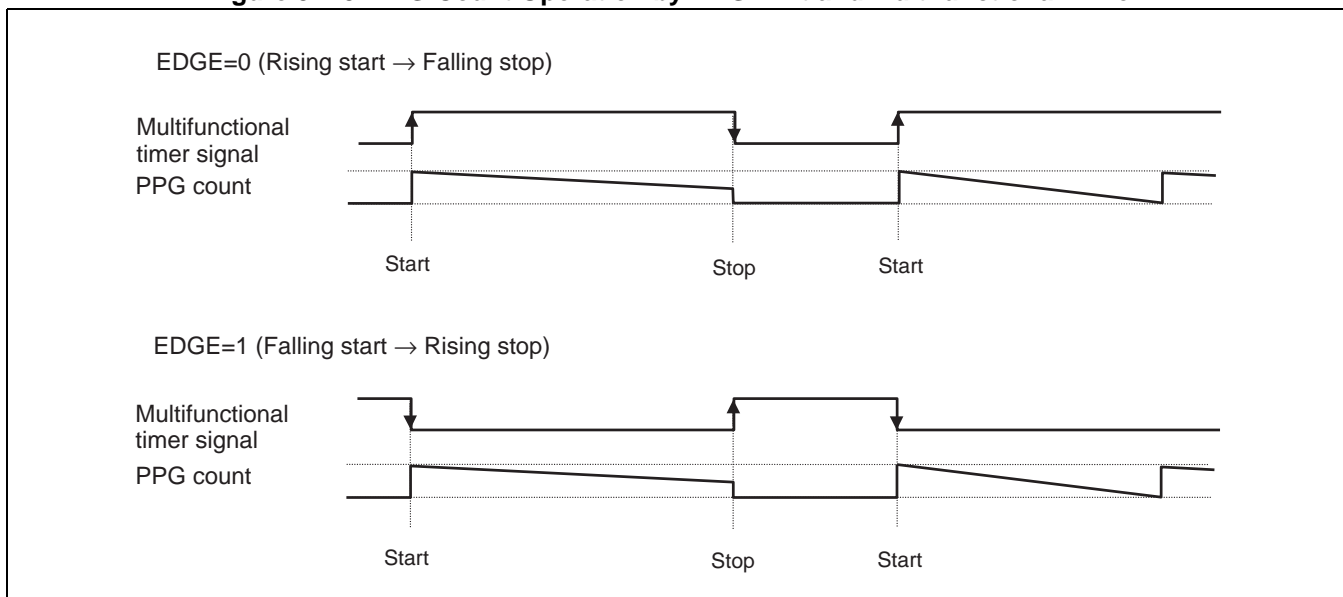
Switching of each mode activation is based on the MD register setting of each PPG.

- PPG ch0: PPG ch0 start (8-bit PPG) at MD1, MD0 = 0, X
- PPG ch0: PPG ch0, ch1 start (16-bit PPG) at MD1, MD0 = 1, X

EDGE bit and the multifunctional timer signal can control the period when PPG activation is valid.

The following shows the PPG count operation by the EDGE bit and multifunctional timer.

Figure 9.4-3 PPG Count Operation by EDGE Bit and Multifunctional Timer



■ Initial Value of Each Hardware

Each hardware of this block is initialized by a reset as shown below.

<Register>	PPGC(n) → 0000000X _B
	TRG → 0000000000000000 _B
	REVC → 0000000000000000 _B
	GATEC → XXXXXX00 _B

<Pulse output> PPG(n) → "L"

<Interrupt request> IRQ(n) → "L" (n = 0 to 15)

Any register other than those above is not initialized.

■ PPG Combinations

Combinations of each PPG operation mode are listed below.

ch0: PPGC		ch2: PPGC		ch0	ch1	ch2	ch3
MD1	MD0	MD1	MD0				
0	0	0	0	8-bit PPG	8-bit PPG	8-bit PPG	8-bit PPG
0	0	0	1	8-bit PPG	8-bit PPG	8-bit PPG ←	8-bit prescaler
0	0	1	0	8-bit PPG	8-bit PPG	16-bit PPG	
0	0	1	1	Setting disabled			
0	1	0	0	8-bit PPG ← 8-bit prescaler		8-bit PPG	8-bit PPG
0	1	0	1	8-bit PPG ← 8-bit prescaler		8-bit PPG ←	8-bit prescaler
0	1	1	0	8-bit PPG ← 8-bit prescaler		16-bit PPG	
0	1	1	1	Setting disabled			
1	0	0	0	16-bit PPG		8-bit PPG	8-bit PPG
1	0	0	1	16-bit PPG		8-bit PPG	8-bit prescaler
1	0	1	0	16-bit PPG		16-bit PPG	
1	0	1	1	Setting disabled			
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1	16-bit PPG ←		16-bit prescaler	

The operations for ch.(4, 5, 6, 7), ch.(8, 9, 10, 11), and ch.(12, 13, 14, 15) can be combined in the same way as for ch.(0, 1, 2, 3).

Replace as shown below.

$$\left\{ \begin{array}{l} \text{ch0}=\text{ch4}/\text{ch8}/\text{ch12} \\ \text{ch1}=\text{ch5}/\text{ch9}/\text{ch13} \\ \text{ch2}=\text{ch6}/\text{ch10}/\text{ch14} \\ \text{ch3}=\text{ch7}/\text{ch11}/\text{ch15} \end{array} \right.$$

CHAPTER 10

PWC (Pulse Width Count: Pulse Width Measurement)

This chapter explains the overview of the pulse width counter (PWC), the register configuration and functions and the counter operation.

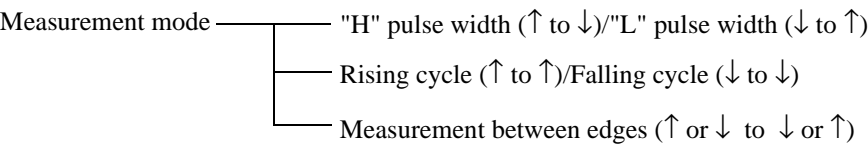
- 10.1 Overview
- 10.2 Block Diagram
- 10.3 Register of PWC
- 10.4 Operation Explanation

10.1 Overview

This section explains the pulse width measurement function for input signal. There are two sets of a channel in total, which consists of one 16-bit up counter, one input pulse divider and divide-by rate control register, one measurement input pin, and one 16-bit control register. They implement the following functions.

■ Pulse Width Measurement Function

Measure the time between arbitrary events of external pulse inputs.
Internal clock to be used as the criterion can be selected from one of the following 3 types.
(4/16/32 division of machine clock)



Enable to measure the cycle after an input pulse divided by 2n (n = 1, 2, 3, 4, 5, 6, 7, 8) with the 8-bit input divider.
Enable to generate an interrupt request when the measurement is completed.
Enable to select either the one-time-only measurement or the continuous measurement.

■ Register List

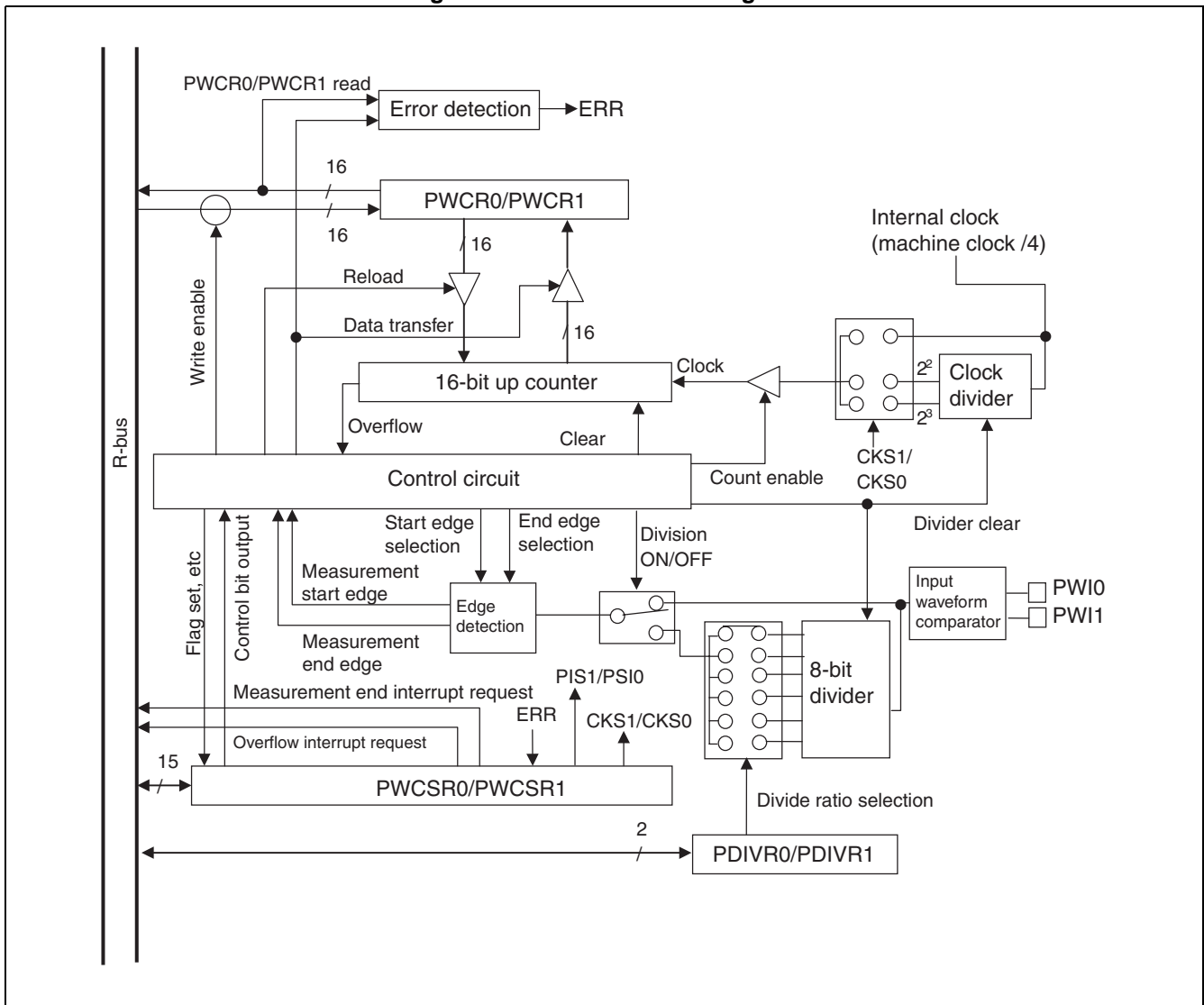
Address		Bit15 to Bit8	Bit7 to Bit0	
0000E9 _H			PDIVR0	Ratio of dividing frequency control register 0
0000E0 _H	0000E1 _H	PWCSR0		Control/status register 0
0000E2 _H	0000E3 _H	PWCR0		Data buffer register 0
	0000EB _H		PDIVR1	Ratio of dividing frequency control register 1
0000E4 _H	0000E5 _H	PWCSR1		Control/status register 1
0000E6 _H	0000E7 _H	PWCR1		Data buffer register 1

10.2 Block Diagram

This section shows the PWC block diagram.

■ PWC Block Diagram

Figure 10.2-1 PWC Block Diagram



10.3 Register of PWC

This section describes the PWC registers.

■ PWCSR Register (PWC Control/Status Register)

PWCSR0/PWCSR1 (Upper)								
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
ch0:0000E0 _H	STRT	STOP	EDIR	EDIE	OVIR	OVIE	ERR	-
ch1:0000E4 _H								
Read/Write →	R/W	R/W	R	R/W	R/W	R/W	R	-
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
PWCSR0/PWCSR1 (Lower)								
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ch0:0000E1 _H	CKS1	CKS0	PIS1	PIS0	SC	MOD2	MOD1	MOD0
ch1:0000E5 _H								
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

[bit15] STRT: Counter start bit

[bit14] STOP: Counter stop bit

The bits start/restart/stop the 16-bit up counter, and indicate the counter operation status at reading. The following shows the bit functions.

1. Function at writing (operation control)

STRT	STOP	Operation Control Functions
0	0	The influence is not in the function / the operation.
0	1	Start/restart a counter (counter enable) *
1	0	Stop a counter operation forcibly (disable count) *
1	1	The influence is not in the function / the operation.

*:The clear bit instruction is usable.

2. Function at reading (operating status indication)

STRT	STOP	Display of Operation Status
0	0	Count stopped (not started or measurement completed) [Initial value]
1	1	Count operating (measuring)

- Initialized to "00_B" when resetting.
- The read / write is possible. However, there are different meanings in writing and in reading as shown above.

- If a read modify write instruction is issued, "11_B" is read regardless of the operation.
- A bit-processing instruction (such as a bit clear) corresponding to each bit can be used for writing to the STRT and STOP bit to start/stop the counter. However, the bit-processing instructions cannot be used for reading an operating status. (Note that the status must be in operation if it is read.)

[bit13] EDIR: Measurement end interrupt request flag

The flag indicates that the measurement is completed in the pulse width count mode. A measurement end interrupt request is generated if this bit is set when the measurement end interrupt request is enabled (bit12: EDIE=1).

Set factor	Set when a pulse width count is completed (the measurement result is stored in PWCR0 or PWCR1).
Clear factor	Cleared when PWCR0 or PWCR1 (measurement result) is read.

- Initialized to "0" at resetting.
- This bit is read-only. Writing does not change the bit value.

[bit12] EDIE: Measurement end interrupt request enable bit

This bit controls the measurement end interrupt request in the pulse width count mode as described below.

0	Disable the measurement end interrupt request output (an interrupt not generated when EDIR is set) [Initial value]
1	Enable measurement end interrupt request output (interrupt is generated if EDIR is set)

- Initialized to "0" at resetting.
- The read / write is possible.

[bit11] OVIR: Counter overflow Interrupt request flag

This flag indicates that the 16-bit up counter has overflowed from "FFFF_H" to "0000_H" in all modes. A counter overflow interrupt request is generated if this bit is set when the counter overflow interrupt request is enabled (bit10: OVIE = 1).

Set factor	Set when a counter overflow occurs ("FFFF _H " to "0000 _H ").
Clear factor	Cleared by writing 0.

- Initialized to "0" at resetting.
- The read / write is possible. However, only "0" can be written on it. Writing "1" does not change the bit value.
- The read value by a read-modify-write instruction is always "1", regardless of the bit value.

[bit10] OVIE: Counter overflow interrupt request enable bit

This bit controls a counter overflow interrupt request as described below.

0	Disable the overflow interrupt request output (an interrupt not generated when OVIR is set) [Initial value]
1	Enable overflow interrupt request output (interrupt is generated when OVIR is set)

- Initialized to "0" when resetting.
- The read / write is possible.

[bit9] ERR: Error flag

The flag indicates that the next measuring is completed before reading the measurement result in PWCR0 or PWCR1 in the continuous measurement mode of the pulse width count mode. At this point, the PWCR0 or PWCR1 is updated to a new measurement result, the previous measurement result will be lost. The measurement is continued regardless of this bit value.

Set factor	Set when the measurement result has not been read is lost because of the next result.
Clear factor	Cleared when PWCR0 or PWCR1 (measurement result) is read.

- Initialized to "0" at resetting.
- This bit is read-only. Writing does not change the bit value.

[bit8] (Reserved)

Reserved bit.

Read value is "0".

Be sure to write "0".

[bit7, bit6] CKS1, CKS0: Clock selection bits

Select an internal counter as listed in the table below.

CKS1	CKS0	Count Clock Selection
0	0	Clock generated by dividing the machine clock by 4 [Initial value]
0	1	Clock generated by dividing the machine clock by 16
1	0	Clock generated by dividing the machine clock by 32
1	1	Setting disabled

- Initialized to "00_B" at resetting.
- The read / write is possible. Do not set "11_B".
- These bits must not be updated after starting. These bits must be written before starting or after stopped.

[bit5, bit4] PIS1, PIS0: Pulse width measurement input pin select bit

These bits select the pulse width measurement input pin.

PIS1	PIS0	Input Clock Selection
0	0	(The pin PWC0 is selected) [Initial value]
0	1	2 input compare-select (rising-edge comparison)
1	0	2 input compare-select (falling-edge comparison)
1	1	Setting disabled

- Initialized to "00_B" at resetting.
- Read and write are enabled. However, do not set "11_B".
- This bit is effective only to PWC ch0. (PWI0/PWI1 is used as an input.) For details, see "■ Details for pulse width measuring operation" in section 10.4 Operation Explanation.
- These bits must not be updated after starting. These bits must be written before starting or after stopped.

[bit3] SC: Measurement mode (single/continuous) select bit

Select a measurement mode as shown below.

SC	Measurement Mode Selection	At Pulse Width Measuring Mode
0	Single measurement mode [Initial value]	Stop after one measurement
1	Continuous measurement mode	Continuous measurement: buffer register enabled

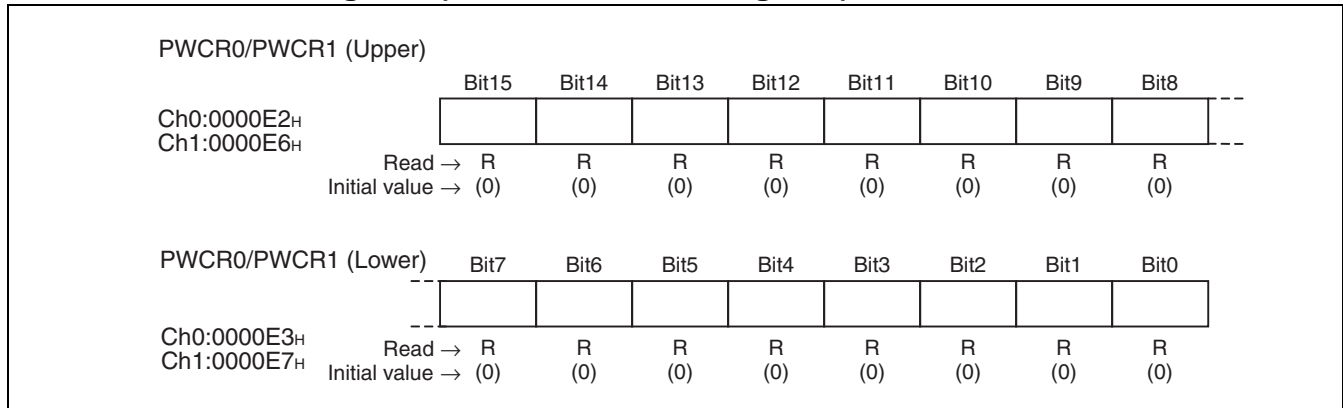
- Initialized to "0" at resetting.
- The read / write is possible.
- This bit must not be updated after starting. This bit must be written before starting or after stopped.

[bit2, bit1, bit0] MOD2, MOD1, MOD0: Operation mode/measurement edge select bit

Select an operation mode and an edge for width measurement as listed in the table below.

MOD2	MOD1	MOD0	Operation Mode/Measurement Edge Selection
0	0	0	Pulse width measurement mode between all edges (\uparrow or \downarrow to \downarrow or \uparrow) [initial value]
0	0	1	Division cycle measurement mode (input divider enabled)
0	1	0	Rising edge-to-edge cycle measurement mode (\uparrow to \uparrow)
0	1	1	"H" pulse width measurement mode (\uparrow to \downarrow)
1	0	0	"L" pulse width measurement mode (\downarrow to \uparrow)
1	0	1	Falling edge-to-edge cycle measurement mode (\downarrow to \downarrow)
1	1	0	Setting disabled
1	1	1	

- Initialized to "000_B" at resetting.
- The read / write is possible.
- These bits must not be updated after starting. These bits must be written before starting or after stopped.

■ PWCR0, PWCR1 Register (PWC Data Buffer Register)**● Pulse width measurement mode**

In the continuous measurement mode (PWCSR0 or PWCSR1 bit3: SC = 1), this register becomes a buffer register which holds the previous measurement result. In this case, only read operation is possible, but write operation does not affect the register value.

In the single measurement mode (PWCSR0 or PWCSR1 bit3: SC = 0), this register becomes an interface to access the up counter directly. Also in this case, only read operation is possible, but write operation does not affect the register value. Read operation is always possible, and can get the count value while counting is in progress. The measurement result is saved after the measurement ends.

Be sure to use a halfword or word transfer instruction to access this register.

- Initialized to "0000_H" at resetting.
- Only read operation is possible.

■ PDIVR (Ratio of Dividing Frequency Control Register)

PDIVR0/1								
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Ch0:0000E9 _H	-	-	-	-	-	DIV2	DIV1	DIV0
Ch1:0000EB _H	-	-	-	-	-	R/W	R/W	R/W
Read/Write →	-	-	-	-	-	(0)	(0)	(0)
Initial value →	(X)	(X)	(X)	(X)	(X)	(0)	(0)	(0)

This register is used in the division cycle measurement mode (PWCSR bit2, bit1, bit0: MOD2, MOD1, MOD0=001_B), and invalid in other modes.

In the division cycle measurement mode, divide the input pulse in the count pin by the divide-by rate set in this register and measure one cycle width after divided. Select the divide-by rate as shown below.

DIV2	DIV1	DIV0	Divide Ratio Selection
0	0	0	$2^1=2$ dividing frequency [Initial value]
0	0	1	$2^2=4$ dividing frequency
0	1	0	$2^3=8$ dividing frequency
0	1	1	$2^4=16$ dividing frequency
1	0	0	$2^5=32$ dividing frequency
1	0	1	$2^6=64$ dividing frequency
1	1	0	$2^7=128$ dividing frequency
1	1	1	$2^8=256$ dividing frequency

- Initialized to "000_B" at resetting.
- The read / write is possible.
- These bits must not be updated after starting. These bits must be written before starting or after stopped.

10.4 Operation Explanation

PWC has the measurement input pin, 8-bit input division, and so on. PWC has also the pulse width count function. Three types of count clocks can be selected. The following describes the basic functions/operations of the pulse width count.

● Pulse width measurement function

Enable to measure the time/cycle between any events of input pulse by the counter.

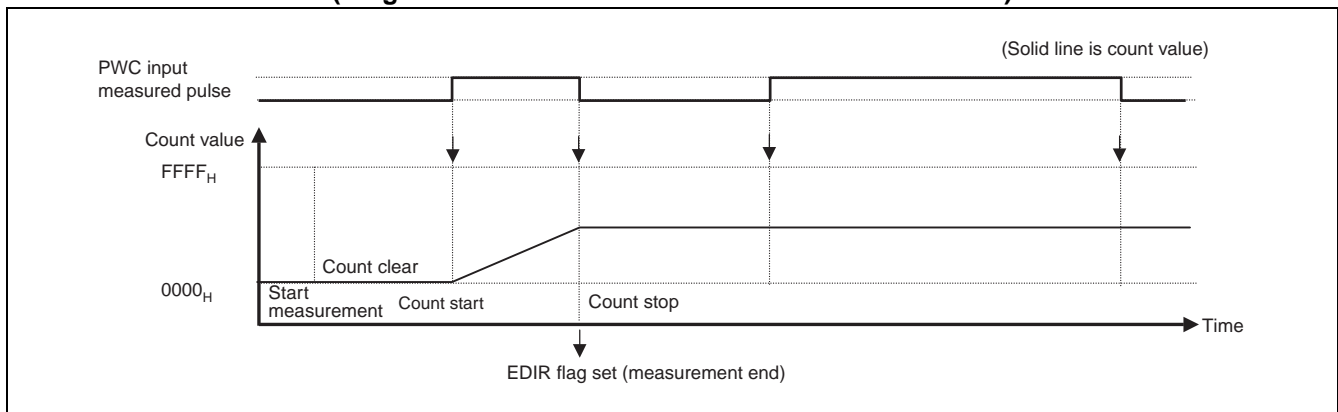
After starting, counting will not be performed until the defined measurement start edge is inputted. Start incremental counting after the counter which detects a start edge is cleared to "0000_H". Stop the counting when a stop edge is detected. The value counted in this period is stored in the register as the pulse width.

Generate an interrupt requests when the measurement ends and an overflow occurs.

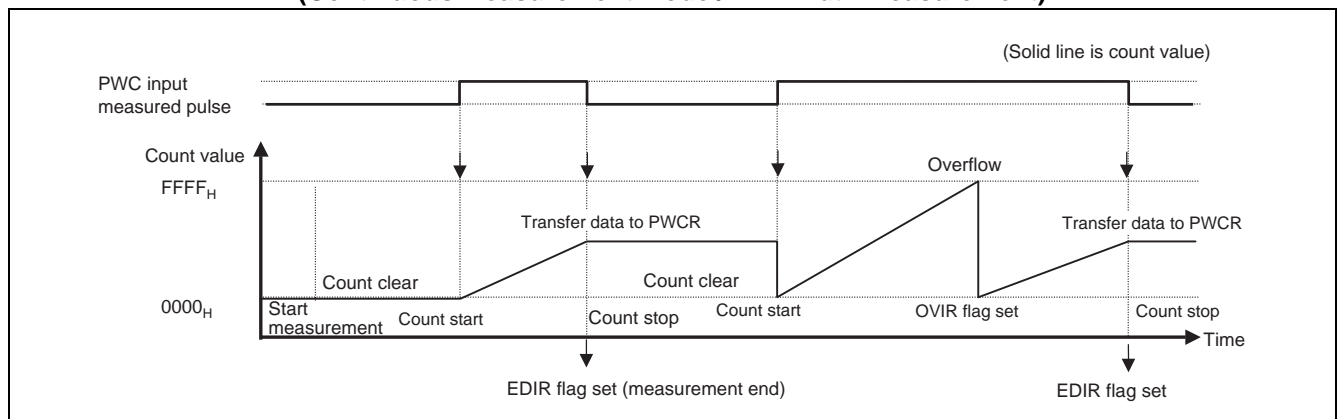
Operates as follows depending on the measurement mode after the measurement end.

- In single measurement mode: Stop operation.
- In continuous measurement mode: After transferred a counter value to the buffer register, stop the counting until the measurement start edge is inputted again.

**Figure 10.4-1 Pulse Width Measurement Operation
(Single Measurement Mode / "H" Width Measurement)**



**Figure 10.4-2 Pulse Width Measurement Operation
(Continuous Measurement Mode / "H" Width Measurement)**



Count Clock Selection

The count clock of counter can select three types of internal clock sources by setting of PWCSR register (bit7, bit6: CKS1, CKS0).

Selectable count clocks are as shown below.

PWCSR	Selecting Internal Count Clock
CKS1, CKS0	
00 _B	Machine clock 4-divided [initial value]
01 _B	Machine clock 16-divided
10 _B	Machine clock 32-divided

The clock generated by dividing the machine clock by 4 is selected for the initial value after a reset.

Be sure to select the count clock before the counter starts.

Selects Operation Mode

The PWCSR setting is used to select each operation/measurement mode.

- Operation mode setting: PWCSR bit2 to bit0: MOD2, MOD1, MOD0
(for example, selecting a pulse width count mode, or determining a measurement edge)
- Measurement mode setting: PWCSR bit3: SC
(selecting a single/continuous measurement)

The selection of operation modes by the combination of the mode selecting bits is listed below.

Operating Mode			SC	MOD2	MOD1	MOD0
Pulse width measurement	↑ or ↓ to ↑ or ↓ Measurement between all edges	Single measurement: Buffer invalidity	0	0	0	0
		Continuous measurement: Buffer effective	1	0	0	0
	Measurement at cycle of dividing frequency (1-frequency division to 256-frequency division)	Single measurement: Buffer invalidity	0	0	0	1
		Continuous measurement: Buffer effective	1	0	0	1
	↑ to ↑ Rising-to-rising cycle measurement	Single measurement: Buffer invalidity	0	0	1	0
		Continuous measurement: Buffer effective	1	0	1	0
	↑ to ↓ "H" pulse width measurement	Single measurement: Buffer invalidity	0	0	1	1
		Continuous measurement: Buffer effective	1	0	1	1
	↓ to ↑ "L" pulse width measurement	Single measurement: Buffer invalidity	0	1	0	0
		Continuous measurement: Buffer effective	1	1	0	0
	↓ to ↓ Falling-to-falling cycle measurement	Single measurement: Buffer invalidity	0	1	0	1
		Continuous measurement: Buffer effective	1	1	0	1
Setting disabled			0	1	1	0
			1	1	1	0
			0	1	1	1
			1	1	1	1

All edge-to-edge measurement-single measurement mode is selected for the initial value after a reset.

Be sure to select the activation mode before the counter starts.

■ Starting and Stopping of Pulse Width Count

The PWCSR (bit15, bit14: STRT, STOP bits) is used to start/restart/forcibly stop operations.

STRT bit starts/restarts a pulse width count, and STOP bit forcibly stops it. They can work when "0" is written to their bit. But they cannot work unless the value written to both bits is exclusive. Be sure to write the following combination when an instruction other than bit manipulation instructions (other than byte) is written.

Function	STRT	STOP
Starting / restarting of the pulse width count	0	1
Forcibly stopping of the pulse width count	1	0

When a bit manipulation instruction (bit clear instruction) is used, these combinations are written by the hardware automatically without having to take them into consideration.

● Operation after startup

The following describes the operation after starting a pulse width count mode.

Pulse width count mode: The count will not be performed until the measurement start edge is inputted. After the measurement start edge is detected, 16-bit up counter is cleared to "0000_H", and the counting is started.

● Reactivation

A restart is to start a pulse width count (write "0" to the STRT bit) during operation after the count has been started. By restarting, the following operation occurs.

There is no influence in the operation, at measurement start edge wait state. If the measurement is in progress, stops the counting, and then waits for the measurement start edge again. At this time, if a restart occurs simultaneously with detecting a measurement end edge, the measurement end flag (EDIR) is set, and the measurement result is transferred to PWCR in continuous measurement mode,

● Stop

In single measurement mode, the counter overflow or the measurement end stops the count operation automatically without having to take this into consideration. In other modes, or to stop operation before stopping automatically, the operation must be stopped forcibly.

At two input selection of comparison

When the selected PWI1 edge is not inputted before the forced stop, the first measurement result after the restart becomes erroneous. Perform the forced stop after the PWI1 edge is inputted.

● Checking the operation status

The above STRT bit and STOP bit work as an operating status indication bit during reading. The value displayed means the following description.

STRT	STOP	Operating State
0	0	Counter stopped (except in waiting for a measurement start edge) It is not active or shows that the measurement ended.
1	1	Counter operating, or waiting for a measurement start edge

The same value is read from either the STRT bit or the STOP bit. If a read modify write instruction (such as bit-processing instruction) is issued, "11_B" is always read from these bits. Do not read by using such instruction.

■ Counter Clear

The 16-bit up counter is cleared to "0000_H" in the following cases.

- At a reset
- Start the count when a measurement start edge is detected in the pulse width count mode.

■ Details for Pulse Width Measuring Operation

● Single measurement and continuous measurement

The pulse width count has the mode to perform the measurement once only and the mode to perform the measurement continuously. PWCSR SC bit is used to select each mode (see "■ Selects the operation mode"). The differences between these modes are as described below.

- Single measurement mode: When the first measurement end edge is inputted, the operation stops counter count, sets the measurement end flag (EDIR) in PWCSR, and performs no further measurement. However, if a restart occurs simultaneously, the operation waits for a measurement start.
- Continuous count mode: When the count end edge is inputted, the operation stops the counter count, sets the count end flag (EDIR) in PWCSR, and stops the count until the count start edge is inputted again. When the measurement start edge is inputted again, the operation clears the counter to "0000_H", and then starts the measurement. When the measurement ends, the measurement result of the counter is transferred to PWCR.

Be sure to select/change the measurement mode while the counter is stopped.

● Measured data

The differences between the single measurement mode and the continuous measurement mode are in handling of a measurement result and a count value, and the PWCR function. The following describes the differences of the measurement results in both modes.

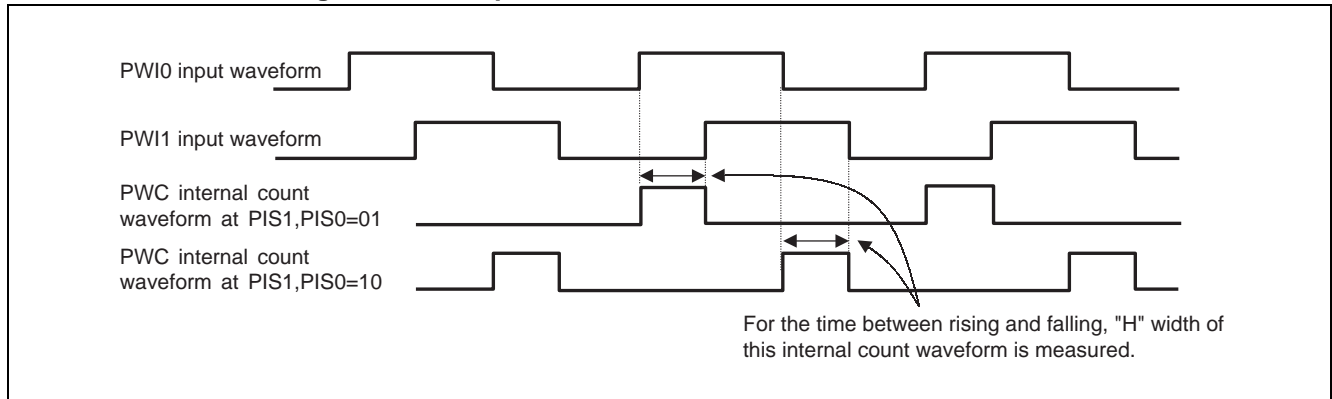
- Single measurement mode: A count value in measuring can be obtained when PWCR is read during operation. A measurement result can be obtained when PWCR is read after the measurement ends.
- Continuous measurement mode: A measurement result in counter is transferred to PWCR when the measurement ends.
When PWCR is read, a previous measurement result can be obtained and held while the measurement is in progress. A count value in measuring cannot be read.

If the next measurement is completed before reading the measurement result in continuous measurement mode, the previous measurement result is cleared by the new measurement result. In this case, the error flag in PWCSR (ERR) is set. An error flag (ERR) is cleared automatically when PWCR is read.

● Input pin selection

There are 2 channels for PWC. Pins used for signal inputs for a pulse width count have 2 channels: PWI0 and PWI1. Therefore, the pin can be used independently for each channel. For input pin PWI0 and PWI1, the time between rising and falling of each input waveform can be measured by combining PIS1 and PIS0 in PWCSR0. Note that PWI0 in the PWCR register is used. At this time, "H" width of PWC internal count waveform is measured. Settings of MOD2, MOD1, MOD0 are ignored.

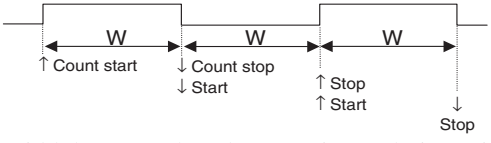
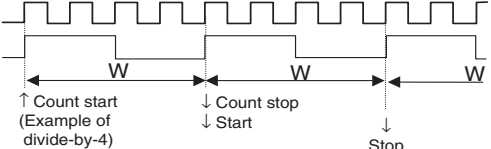
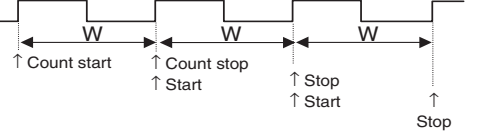
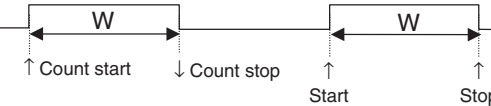
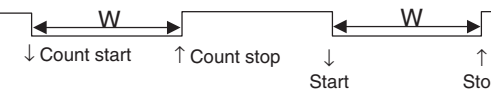
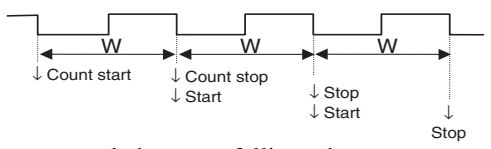
Figure 10.4-3 Input Waveform and Internal Count Waveform



- If you compare two inputs, start counting at PWI0, stop counting at PWI1 even for rising/falling detection.
- If you change the detection mode for rising/falling, be sure to perform after the measurement ends.

● Measurement mode and count operation

The measurement mode can be selected from five types depending on the place where the input pulse is measured. The cycle measurement mode is also prepared to arbitrarily divide the input pulse for high-precision measurement of higher frequency pulse width. They are described below.

Measurement Mode	MOD2	MOD1	MOD0	Measurement Details (W: Pulse Width to Be Measured)
Pulse width measurement between all edges	0	0	0	 <p>The width between the edges continuously input is measured. Count (measurement) beginning: When edge is detected Count (measurement) end: When edge is detected</p>
Measurement at cycle of dividing frequency	0	0	1	 <p>Only the divide ratio of frequency selected by divide ratio frequency setting register PDIVR0 or PDIVR1 divides the input pulse. And, the cycle is measured. Count (measurement) start: At rising edge detection (directly after start) Count (measurement) end: when one cycle ends after it is divided</p>
Rising edge-to-edge cycle measurement	0	1	0	 <p>Measure a cycle between rising edges. Count (measurement) start: At rising edge detection Count (measurement) end: At rising edge detection</p>
"H" pulse width measurement	0	1	1	 <p>The width at H period is measured. Count (measurement) start: At rising edge detection Count (measurement) end: At falling edge detection</p>
"L" pulse width measurement	1	0	0	 <p>The width at L period is measured. Count (measurement) start: At falling edge detection Count (measurement) end: At rising edge detection</p>
Rising edge-to-edge cycle measurement	1	0	1	 <p>Measure a cycle between falling edges. Count (measurement) start: At falling edge detection Count (measurement) end: At falling edge detection</p>

In any mode, after a measurement is started, the counter does not perform a count operation until a measurement start edge is inputted. When the measurement start edge is inputted, the counter is cleared to "0000_H", and then continues the up count per count clock until a measurement end edge is inputted.

When the measurement end edge is inputted, the following operation occurs.

1. Measurement ending flag (EDIR) in PWCSR is set.
2. Stop the counter count operation. (Except in case that a restart occurs simultaneously.)
3. In continuous measurement mode: A counter value (count result) is transferred to PWCR, stops the counting until the next measurement start edge is inputted.
4. In single measurement mode: End the measurement. (Except in case that a restart occurs simultaneously.)

When all edge-to-edge pulse width count or a cycle measurement in continuous mode is performed, the end edge is used as the next measurement start edge.

● Minimum input pulse width

The following restrictions apply to the pulse that can be inputted to the pulse width count input pin (PW11, PW10).

Minimum input width: More than machine cycle $\times 4$
(more than 0.25 μs if the machine clock is 16 MHz)

If the pulse whose width is smaller than the above pulse width is inputted, the operation will be unpredictable.

● Calculation method of pulse width/cycle

After the measurement ends, the pulse width/cycle to be counted can be calculated from the data of the measurement result obtained in PWCR.

	T_W : Pulse width to be measured/cycle (μs)
	n : Measurement result data in PWCR
	t : Count clock cycle (μs)
	D_{IV} : Division ratio selected by the division ratio register PDIVR (Substitute "1" in a mode other than the division cycle measurement mode.)
$T_W = n \times t / D_{IV} [\mu\text{s}]$	

● Pulse width/cycle measurement range

The measurable range of the pulse width/cycle depends on the combination of a count clock and a divide-by rate of the input divider.

The following table lists an example of the measurement range when machine clock (referred to as ϕ) = 16 MHz.

Divide Ratio	DIV2, DIV1, DIV0	At CKS1, CKS0=00 _B ($\phi/4$)	At CKS1, CKS0=01 _B ($\phi/16$)	At CKS1, CKS0=10 _B ($\phi/32$)
No frequency division	-	0.25 μ s to 16.4 ms [250 ns]	0.25 μ s to 65.5 ms [1.0 μ s]	0.25 μ s to 131 ms [2.0 μ s]
2-frequency division	000 _B	0.25 μ s to 8.19 ms [125 ns]	0.25 μ s to 32.8 ms [0.5 μ s]	0.25 μ s to 65.5 ms [1.0 μ s]
4-frequency division	001 _B	0.25 μ s to 4.10 ms [62.5 ns]	0.25 μ s to 16.4 ms [250 ns]	0.25 μ s to 32.8 ms [0.5 μ s]
8-frequency division	010 _B	0.25 μ s to 2.05 ms [31.25 ns]	0.25 μ s to 8.19 ms [125 ns]	0.25 μ s to 16.4 ms [250 ns]
16-frequency division	011 _B	0.25 μ s to 1.02 ms [15.6 ns]	0.25 μ s to 4.10 ms [62.5 ns]	0.25 μ s to 8.19 ms [125 ns]
64-frequency division	100 _B	0.25 μ s to 256 μ s [3.91 ns]	0.25 μ s to 1.024 ms [15.6 ns]	0.25 μ s to 2.05 ms [31.25 ns]
256-frequency division	101 _B	0.25 μ s to 64.0 μ s [0.98 ns]	0.25 μ s to 256 μ s [3.91 ns]	0.25 μ s to 512 μ s [7.81 ns]
-	Others	Setting disabled		

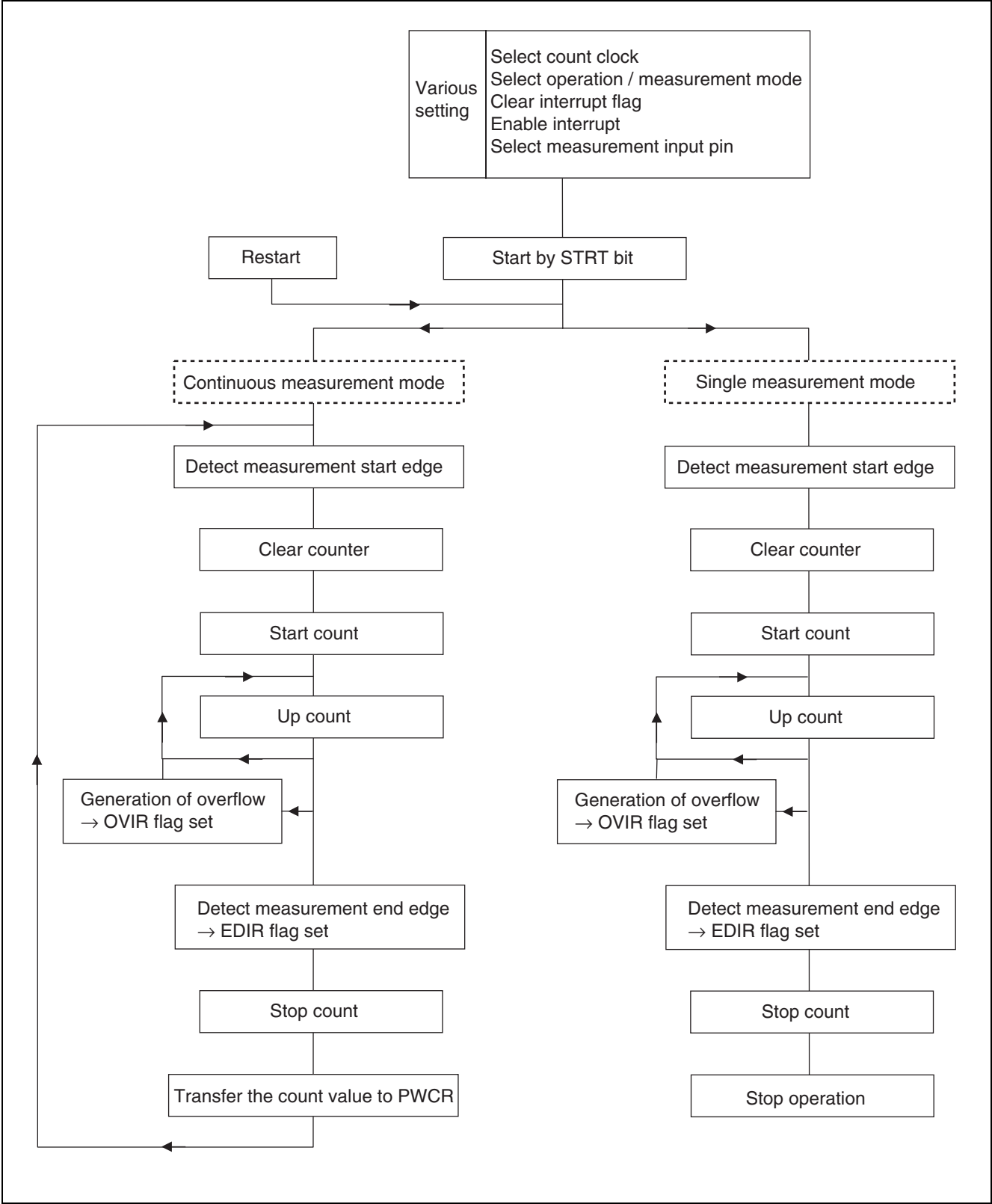
- At machine clock ϕ = 16 MHz
- The value in [] indicates the resolution per bit.

● Interrupt request generation

In the pulse width count mode, the following two interrupt requests can be generated.

- Interrupt request when a counter overflow occurs.
When an incremental count causes an overflow during measuring, an overflow flag is set, and an interrupt request is generated if an overflow interrupt request is enabled.
- Interrupt request by the measurement end
When the measurement end edge is detected, the measurement end flag (EDIR) in the PWCSR is set. If the measurement termination interrupt request is permitted, the interrupt request occurs.
A measurement end flag (EDIR) is cleared automatically when a measurement result PWCR is read.

● Operation flowchart for pulse width count



■ Note

● Note on register rewriting

The following bits of the PWCSR register must not be rewritten during an operation. Be sure to rewrite them before operation is started or after operation is stopped.

[bit7, bit6] CKS1, CKS0: Clock select bit

[bit5, bit4] PIS1, PIS0: Pulse width count input pin select bit

[bit3] SC: Measurement mode (single/continuous) select bit

[bit2, bit1, bit0] MOD2, MOD1, MOD0: Operation mode/measurement edge select bit

The PDIVR register must not be rewritten during an operation. Be sure to rewrite it before operation is started or after operation is stopped.

● STRT bit and STOP bit of the PWCSR register

Note that both bits have different meanings in writing and in reading (See "10.3 Register of PWC"). Also, the read value by a read-modify-write instruction is always "11_B", regardless of the bit value. For this reason, note that the bit-processing instructions cannot be used for reading an operation status (the status must be in operation if it is read).

A bit-processing instruction (such as a bit clear instruction) corresponding to each bit can be used for writing to the STRT and STOP bits to start/stop the counter.

● Counter clear

In the pulse width count mode, because a measurement start edge clears a counter, the data which exists in counter before starting becomes invalidated.

● Minimum input pulse width

The following restrictions apply to the pulse that can be inputted to the pulse width count input pin.

- Minimum input width: Machine cycle x 4 (≥ 250 ns when machine cycle is 62.5 ns)
- Maximum input frequency: Clock generated by dividing the machine clock by 4 (≤ 4 MHz when machine cycle is 16 MHz)

If the pulse which has smaller width/higher frequency than the above is inputted, the operation will be unpredictable. If there may be such noise on the input signal, remove the noise through a filter outside the chip, and then input the pulse.

● Measurement mode at cycle of dividing frequency

Note that the pulse width obtained from the measured result calculation is the average value because the input pulses are divided in the division cycle measurement mode among the pulse width measurement modes.

● Clock select bit

Setting "11_B" to the (bit7, bit6): CKS1, CKS0: clock select bit of the PWCSR register is prohibited.

● Reserved bit

The (bit8) of PWCSR register is a reserved bit. When writing to this bit, be sure to set to "0".

● Reactivation under operation

If a count operation is restarted after it was started, the followings may occur depending on the timing.

- In the pulse width single count mode, if starting a measurement end edge is simultaneously
The operation is restarted and waits for a measurement start edge, but a measurement end flag (EDIR) is set.
- In the pulse width continuous measurement mode, if starting a measurement end edge is simultaneously
The operation is restarted and waits for a measurement start edge, but the measurement end flag (EDIR) is set. The measurement result at that point is transferred to the PWCR.

When restarting an operation in progress, interrupt control or others must be performed with care to the operation of flags, as mentioned above.

CHAPTER 11

MULTIFUNCTIONAL TIMER

This chapter explains the overview of the multifunction timer, the configuration and functions of registers, and operation of the multifunction timer.

- 11.1 Overview
- 11.2 Block Diagram
- 11.3 Pins of Multifunctional Timer
- 11.4 Multifunctional Timer Register
- 11.5 Multifunctional Timer Interrupt
- 11.6 Operation of Multifunctional Timer
- 11.7 Notes on Using Multifunctional Timer
- 11.8 Program Example of Multifunctional Timer

11.1 Overview

Multifunctional timer consists of one 16-bit free-run timer, six 16-bit output compares, four 16-bit input captures, 8 channels of 8/16-bit PPG timer, one waveform generator, and three A/D activation compares. If this waveform generator is used, 12 different waveforms can be outputted from the 16-bit free-run timer, and an input pulse width and an external clock cycle can be measured.

■ Configuration of Multifunctional Timer

● 16-bit free-run timer (×1)

- The 16-bit free-run timer consists of 16-bit up/down counters, control registers, 16-bit compare clear registers (with buffer registers), and prescalers.
- Nine kinds of count operation clocks (ϕ , $\phi/2$, $\phi/4$, $\phi/8$, $\phi/16$, $\phi/32$, $\phi/64$, $\phi/128$, $\phi/256$) can be selected (ϕ : Machine clock).
- A compare clear interrupt is generated when a compare clear register compares and matches 16-bit free-run timer. A zero detection interrupt is generated while the 16-bit free-run timer detects a count value "0".
- A compare clear register has selectable buffer registers. (Data written to this buffer register is transferred to a compare clear register.) When the 16-bit free-run timer is stopped and data is written to the buffer, the transfer is performed immediately. When the timer value "0" is detected during the 16-bit free-run timer operation, data is transferred from the buffer.
- When a compare match with a reset, a software clear, or a compare clear register occurs in the up count mode, the count value is reset to "0000_H".
- This counter output value can be used as a multifunctional timer output compare and an input capture clock count.
- When a zero detection or a compare match occurs, A/D can be activated.

● 16-bit output compare (×6)

- The 16-bit output compare consists of six 16-bit compare register (with selectable buffer registers), compare output latches, and compare control registers. When the 16-bit free-run timer value matches the compare register, an interrupt is generated and an output level is reversed.
- Six compare registers can be operated independently. An output pin and an interrupt flag are assigned to each compare register.
- Two compare registers can be paired to control an output pin. The output pin can be reversed with using two compare registers together.
- The initial value of the output pin can be set.
- An interrupt can be generated when the output compare register matches the 16-bit free-run timer.

● 16 bit input capture (×4)

- The input capture consists of four independent external input pins, and capture registers and capture control registers associated to these pins. When an edge of an input signal is detected on an external pin, the value of the 16-bit free-run timer can be stored to the capture register. Also, an interrupt can be generated at the same time.

- The trigger edge for the external input signal can be selected from the three types: rising edge, falling edge, and both edges. Also, there are registers that indicate whether the trigger edge is a rising edge or a falling edge.
- Four input captures can be used independently.
- An interrupt can be generated when a valid edge of an external input signal is detected.

● 8/16-bit PPG timer (×8)

- The PPG timer 0 is used to provide the PPG signal for the waveform generator. For more information on PPG timer 0, see "CHAPTER 9 PPG (Programmable Pulse Generator)".

● Waveform generator

- The waveform generator consists of three 16-bit dead timers, three timer control registers, and one 16-bit waveform control register.
- The waveform generator can generate real-time output, 16-bit PPG waveform output, non-overlap 3-phase waveform output (for inverter control), and DC chopper waveform output.
- The non-overlap waveform output can be generated on the basis of the dead time of the 16-bit dead timer (dead time timer function).
- The non-overlap waveform output can be generated when the real-time output is activated in 2 channel mode (dead time timer function).
- When a real-time output compare match is detected, GATE signal is generated to started or stopped the PPG timer operation (GATE function).
- When a real-time output compare match is detected, the 16-bit dead timer becomes active. With generating the GATE signal for controlling the PPG operation, the PPG timer 0 can be start or stop easily (GATE function).
- DTTI pins can be used to control stopping forcibly.
- DTTI registers can be used to control stopping forcibly.

● AD activate compare (×3)

- When the 16-bit free-run timer value matches the compare register, A/D can be activated.
- Compare 0 can activate the unit 0 of the A/D converter.
- Compare 1 can activate the unit 1 of the A/D converter.
- Compare 2 can activate the unit 2 of the A/D converter.

11.2 Block Diagram

This section describes the block diagram of the multifunctional timer.

■ Block Diagram of Multifunctional Timer

Figure 11.2-1 Block Diagram of Multifunctional Timer

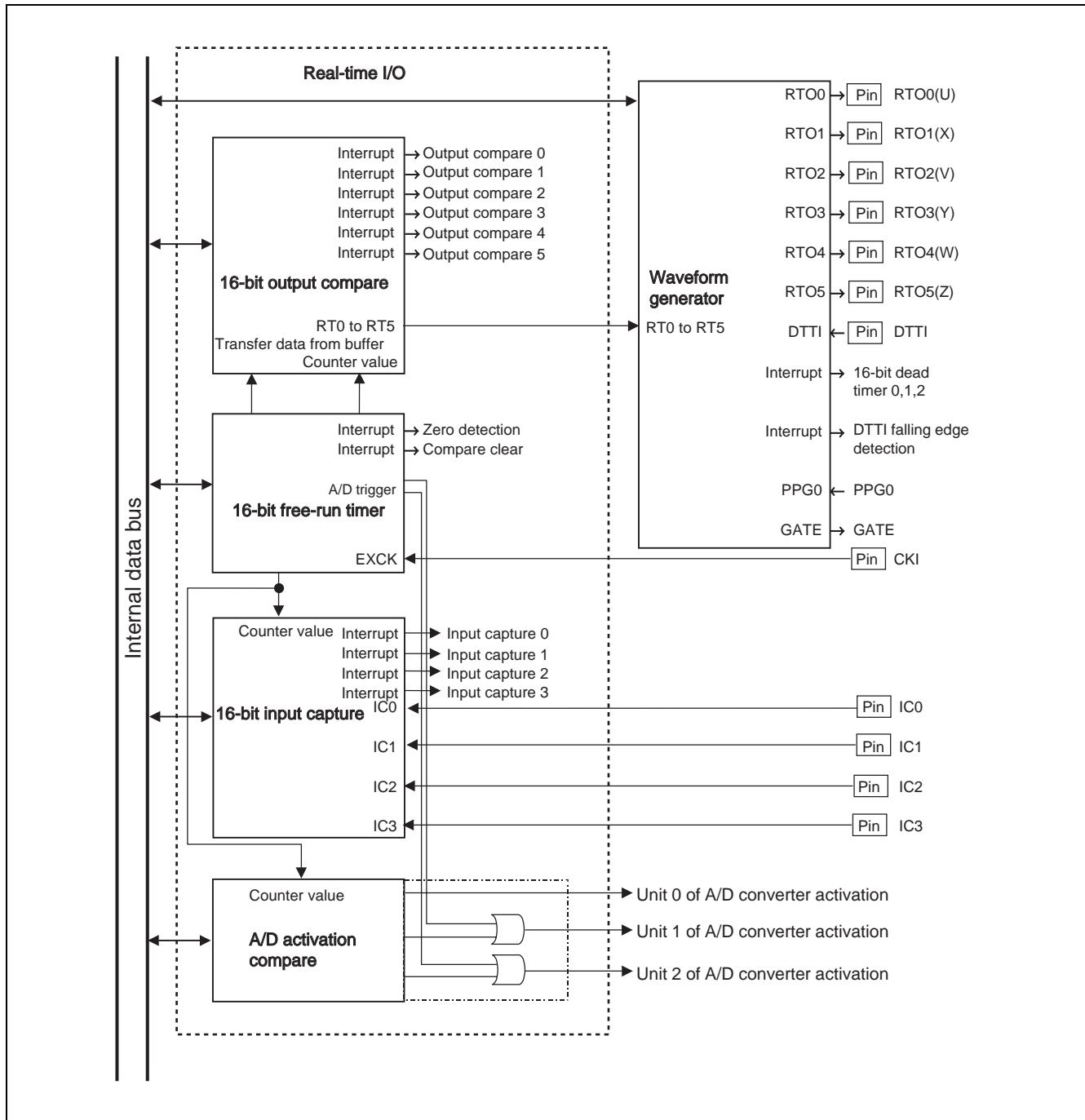


Figure 11.2-2 Block Diagram of 16-bit Free-run Timer

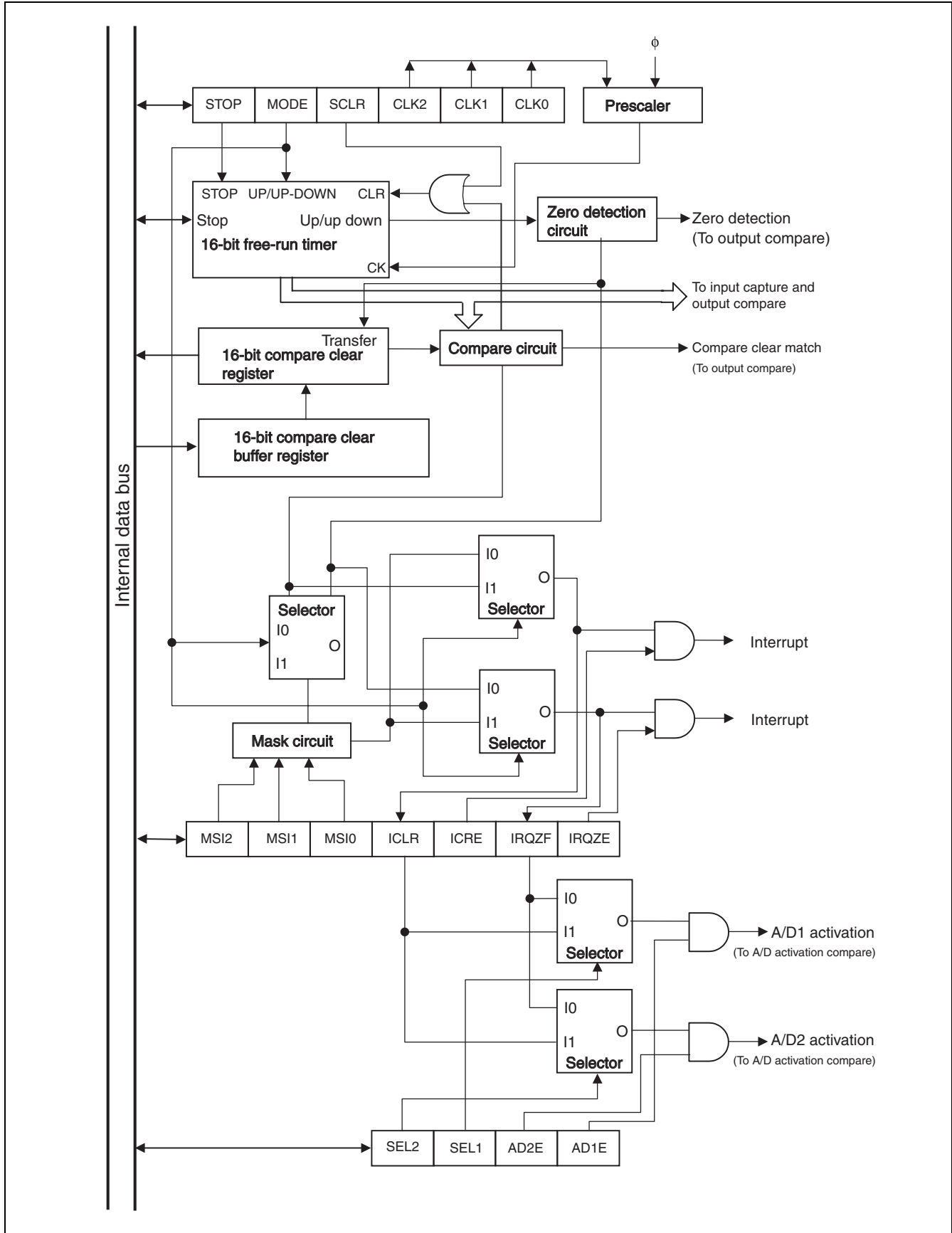


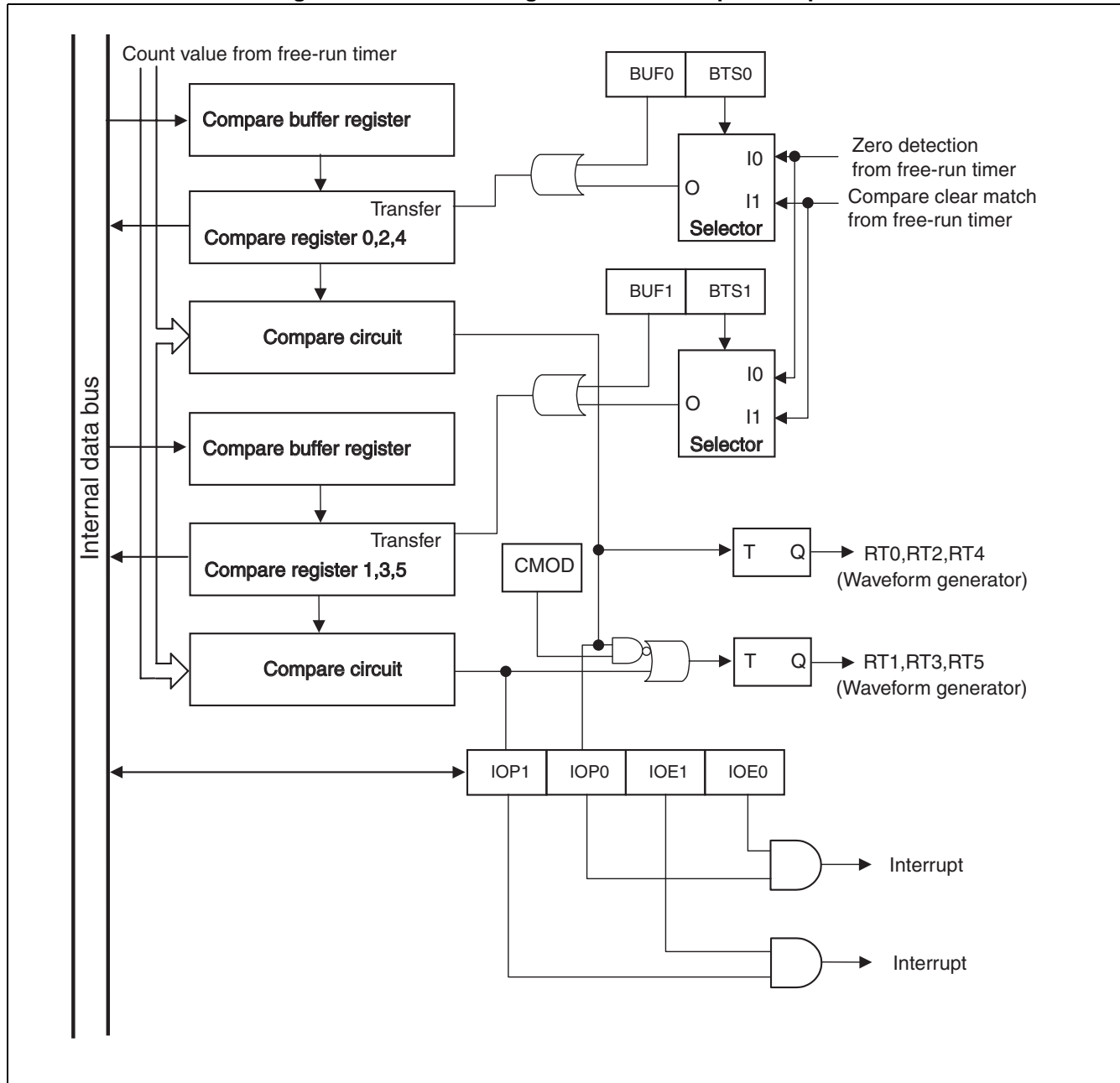
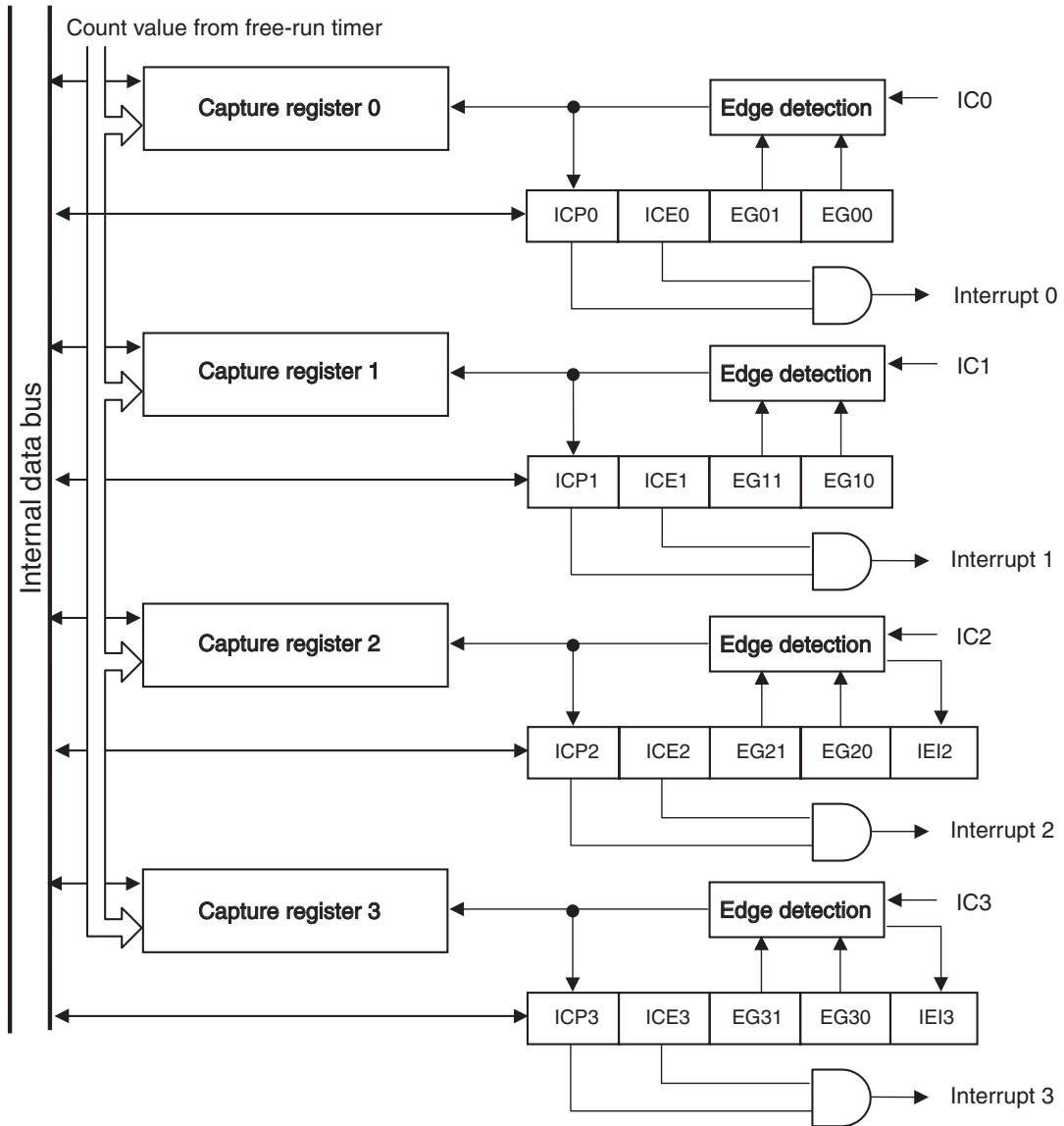
Figure 11.2-3 Block Diagram of 16-bit Output Compare

Figure 11.2-4 Block Diagram of 16-bit Input Capture

Note: Input capture 0 and 1 do not have the detection edge register.

Figure 11.2-5 Block Diagram of Waveform Generator

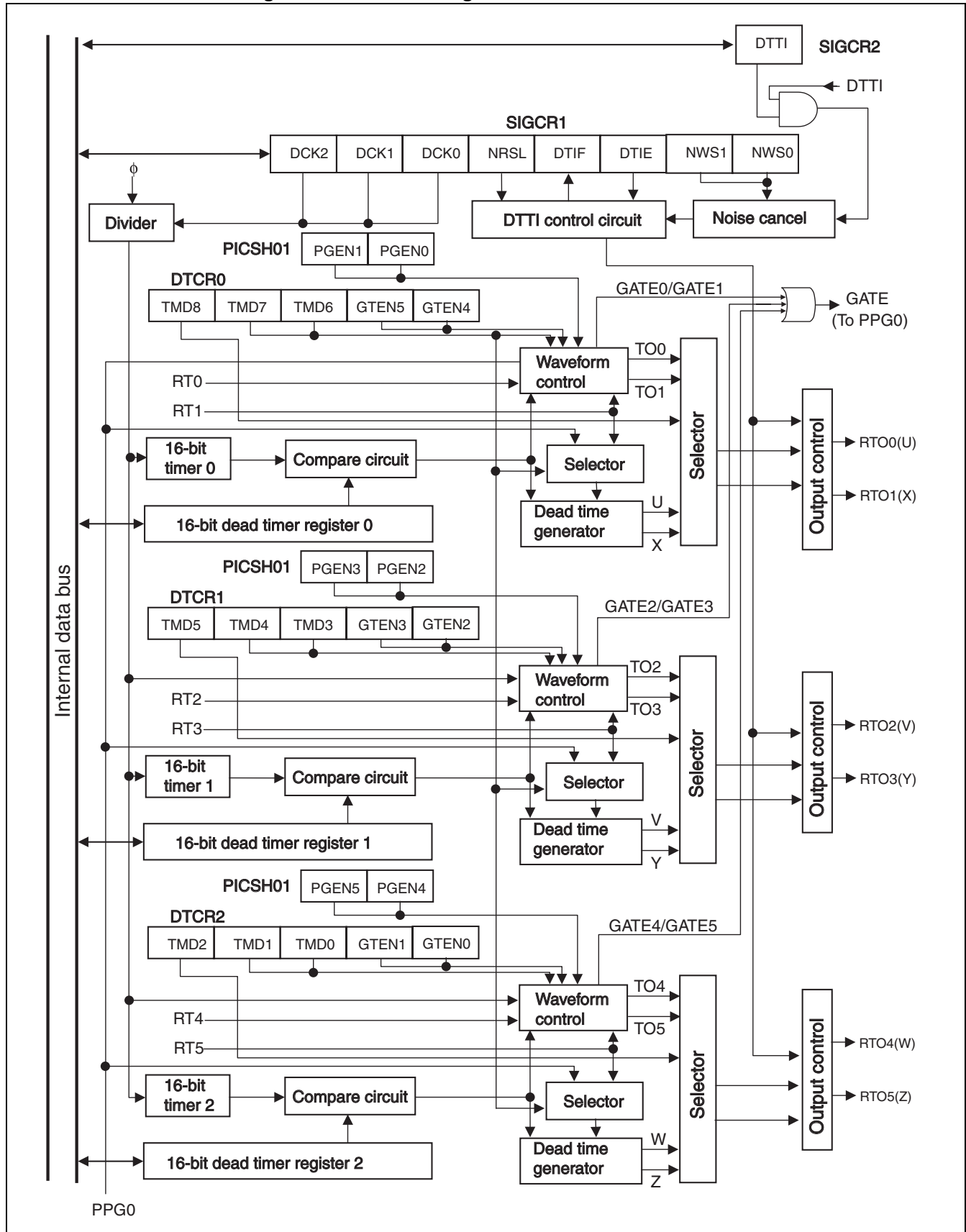
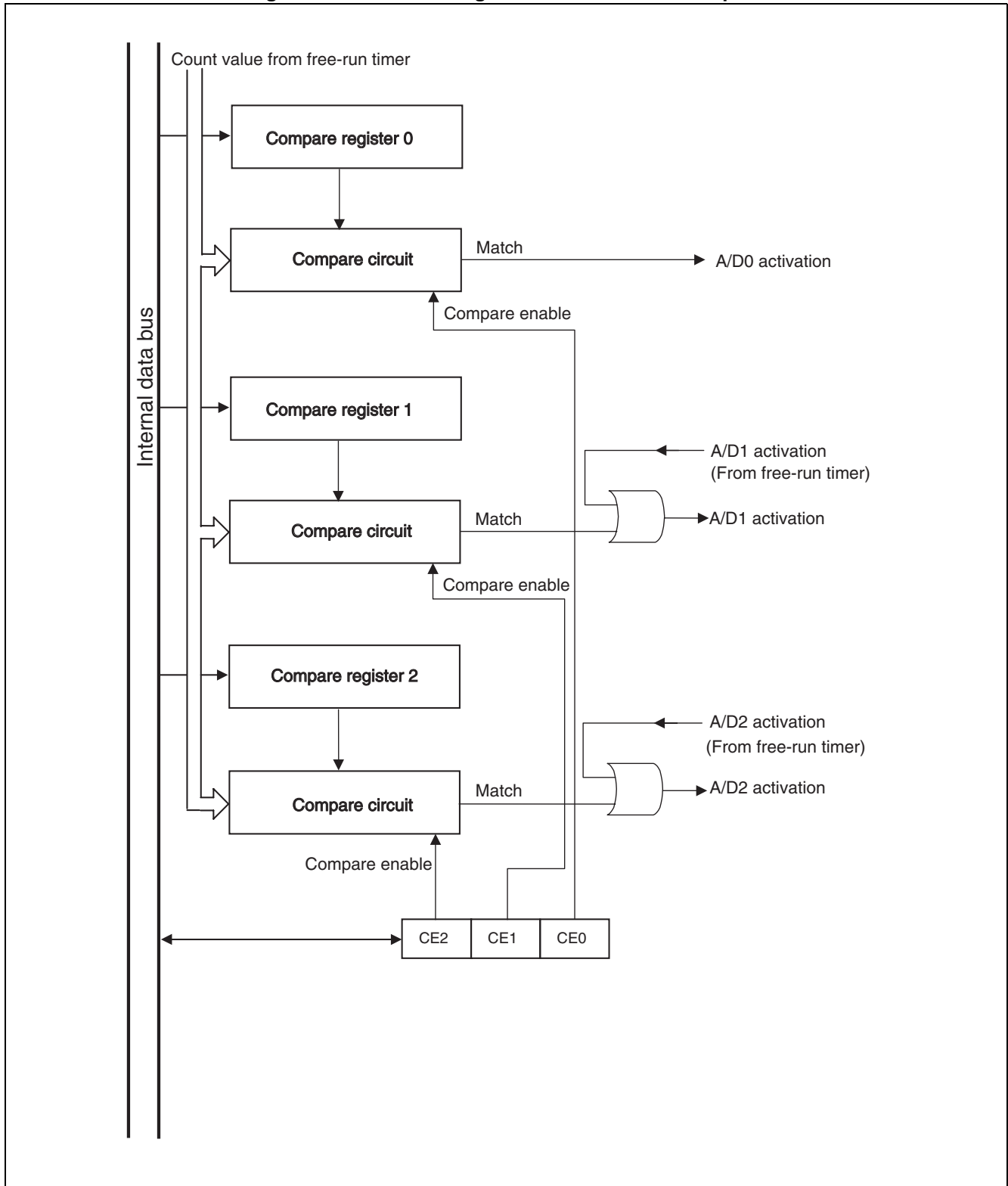


Figure 11.2-6 Block Diagram of A/D Activate Compare

11.3 Pins of Multifunctional Timer

This section describes the pins of the multifunctional timer.

■ Pins of Multifunctional Timer

Table 11.3-1 Pins of Multifunctional Timer

Pin Name	Pin Function	I/O type	Pull-up Option	Stand-by Control	Pin Setting
P72/DTTI	Port7 I/O, DTTI	CMOS output, CMOS hysteresis input	Selectable	Yes	Set a pin as input port (DDR7: bit2 = 0).
PG0/INT4/CKI	Port G I/O, external interrupt input, external clock			None	Set a pin as input port (DDRG: bit0 = 0).
P36/IC0	Port3 I/O, input capture 0			Yes	Set a pin as input port (DDR3: bit6 = 0).
P37/IC1	Port3 I/O, input capture 1				Set a pin as input port (DDR3: bit7 = 0).
P60/IC2	Port6 I/O, input capture 2				Set a pin as input port 6 (DDR6: bit0 = 0).
P61/IC3	Port6 I/O, input capture 3				Set a pin as input port (DDR6: bit1 = 0).
P30/RTO0 (U)	Port3 I/O, RTO0		None		Set RTO0 output. (OCSH1: OTE0=1, DDR3: bit0=1)
P31/RTO1 (X)	Port3 I/O, RTO1				Set RTO1 output. (OCSH1: OTE1=1, DDR3: bit1=1)
P32/RTO2 (V)	Port3 I/O, RTO2			Set RTO2 output. (OCSH3: OTE0=1, DDR3: bit2=1)	
P33/RTO3 (Y)	Port3 I/O, RTO3			Set RTO3 output. (OCSH3: OTE1=1, DDR3: bit3=1)	
P34/RTO4 (W)	Port3 I/O, RTO4			Set RTO4 output. (OCSH5: OTE0=1, DDR3: bit4=1)	
P35/RTO5 (Z)	Port3 I/O, RTO5			Set RTO5 output. (OCSH5: OTE1=1, DDR3: bit5=1)	

DDR_x: Port direction register

OCSH_x: Compare control register

11.4 Multifunctional Timer Register

This section describes the multifunctional timer registers.

■ 16-bit Free-run Timer Register

Compare clear buffer register, Compare clear register (Upper)

CPCLRBH/CPCLRH Address: 0000A4 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	---
	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	
CPCLRBH Write →	W	W	W	W	W	W	W	W	
CPCLRH Read →	R	R	R	R	R	R	R	R	
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

Compare clear buffer register, Compare clear register (Lower)

CPCLRBL/CPCLRL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	---
	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	
CPCLRBL Write →	W	W	W	W	W	W	W	W	
CPCLRL Read →	R	R	R	R	R	R	R	R	
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	

Timer data register (Upper)

TCDTH Address: 0000A6 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	---
	T15	T14	T13	T12	T11	T10	T09	T08	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Timer data register (Lower)

TCDTL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	---
	T07	T06	T05	T04	T03	T02	T01	T00	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Timer state control register (Upper)

TCCSH Address: 0000A8 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	---
	ECKE	IRQZF	IRQZE	MSI2	MSI1	MSI0	ICLR	ICRE	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Timer state control register (Lower)

TCCSL Address: 0000A9 _H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	---
	BFE	STOP	MODE	SCLR	CLK3	CLK2	CLK1	CLK0	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(1)	(0)	(0)	(0)	(0)	(0)	(0)	

A/D trigger control register

ADTRGC Address: 0000AB _H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	---
	-	-	-	-	SEL2	SEL1	AD2E	AD1E	
Read/Write →	-	-	-	-	R/W	R/W	R/W	R/W	
Initial value →	(X)	(X)	(X)	(X)	(0)	(0)	(0)	(0)	

■ 16-bit Output Compare Register

Output compare buffer register, Output compare register (Upper)

OCCPBH0 to OCCPBH5/
OCCPH0 to OCCPH5

Address:

000090_H

000092_H

000094_H

000096_H

000098_H

00009A_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	OP15	OP14	OP13	OP12	OP11	OP10	OP09	OP08
OCCPBH Write →	W	W	W	W	W	W	W	W
OCCPH Read →	R	R	R	R	R	R	R	R
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Output compare buffer register, Output compare register (Lower)

OCCPBL0 to OCCPBL5/
OCCPL0 to OCCPL5

Address:

000090_H

000092_H

000094_H

000096_H

000098_H

00009A_H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	OP07	OP06	OP05	OP04	OP03	OP02	OP01	OP00
OCCPBL Write →	W	W	W	W	W	W	W	W
OCCPL Read →	R	R	R	R	R	R	R	R
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Compare control register 1,3,5 (Upper)

OCSH1,OCSH3,OCSH5

Address:

00009C_H

00009E_H

0000A0_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	-	BTS1	BTS0	CMOD	OTE1	OTE0	OTD1	OTD0
Read/Write →	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(X)	(1)	(1)	(0)	(0)	(0)	(0)	(0)

Compare control register 0,2,4 (Lower)

OCSL0,OCSL2,OCSL4

Address:

00009D_H

00009F_H

0000A1_H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	IOP1	IOP0	IOE1	IOE0	BUF1	BUF0	CST1	CST0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(0)	(0)	(0)	(0)	(1)	(1)	(0)	(0)

Compare mode control register

OCMOD

Address:

0000A2_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	-	-	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10
Read/Write →	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(X)	(X)	(0)	(0)	(0)	(0)	(0)	(0)

■ 16-bit Input Capture Registers

Input capture data register (Upper)

IPCPH0 to IPCPH3

Address:

0000AC_H

0000AE_H

0000B0_H

0000B2_H

Read →

Initial value →

Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08
R	R	R	R	R	R	R	R
(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)

Input capture data register (Lower)

IPCPL0 to IPCPL3

Read →

Initial value →

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00
R	R	R	R	R	R	R	R
(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)

Input capture state control register (ch2,3) (Upper)

ICSH23

Address: 0000B6_H

Read →

Initial value →

Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
-	-	-	-	-	-	IEI3	IEI2
-	-	-	-	-	-	R	R
(X)	(X)	(X)	(X)	(X)	(X)	(0)	(0)

Input capture state control register (ch2,3) (Lower)

ICSL23

Address: 0000B7_H

Read/Write →

Initial value →

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

PPG output control / Input capture state control register (ch0,1) (Upper)

PICSH01

Address: 0000B4_H

Write →

Initial value →

Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
PGEN5	PGEN4	PGEN3	PGEN2	PGEN1	PGEN0	-	-
W	W	W	W	W	W	-	-
(0)	(0)	(0)	(0)	(0)	(0)	(X)	(X)

Input capture state control register (ch0,1) (Lower)

PICSL01

Address: 0000B5_H

Read/Write →

Initial value →

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

■ Waveform Generator Registers

16-bit dead timer register (Upper)

TMRRH0, TMRRH1,
TMRRH2

Address:

0000BC_H

0000BE_H

0000C0_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	TR15	TR14	TR13	TR12	TR11	TR10	TR09	TR08	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

16-bit dead timer register (Lower)

TMRRL0, TMRRL1,
TMRRL2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	TR07	TR06	TR05	TR04	TR03	TR02	TR01	TR00	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

16-bit dead timer control register 0

DTCCR0

Address: 0000C4_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	DMOD0	GTEN1	GTEN0	TMIF0	TMIE0	TMD2	TMD1	TMD0	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

16-bit dead timer control register 1

DTCCR1

Address: 0000C5_H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	DMOD1	GTEN3	GTEN2	TMIF1	TMIE1	TMD5	TMD4	TMD3	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

16-bit dead timer control register 2

DTCCR2

Address: 0000C6_H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	DMOD2	GTEN5	GTEN4	TMIF2	TMIE2	TMD8	TMD7	TMD6	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Waveform control register 1

SIGCR1

Address: 0000C9_H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	DTIE	DTIF	NRSL	DCK2	DCK1	DCK0	NWS1	NWS0	
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Waveform control register 2

SIGCR2

Address: 0000CB_H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	-	-	-	-	-	-	-	DTTI	
Read/Write →	-	-	-	-	-	-	-	R/W	
Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(1)	

■ A/D Activation Compare Registers

Compare register 0,1,2 (Upper)

ADCOMP0/ADCOMP1/
ADCOMP2

Address :

ch0:0000CCH

ch1:0000CEH

ch2:0000D0H

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	CMP15	CMP14	CMP13	CMP12	CMP11	CMP10	CMP09	CMP08	--
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Compare register 0,1,2 (Lower)

ADCOMP0/ADCOMP1/
ADCOMP2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	CMP07	CMP06	CMP05	CMP04	CMP03	CMP02	CMP01	CMP00	--
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

Control register

ADCOMPC

Address: 0000D3H

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	-	-	-	-	-	CE2	CE1	CE0	
Read/Write →	-	-	-	-	-	R/W	R/W	R/W	
Initial value →	(X)	(X)	(X)	(X)	(X)	(0)	(0)	(0)	

11.4.1 Compare Clear Buffer Register (CPCLRBH, CPCLRBL) / Compare Clear Register (CPCLRH, CPCLRL)

The compare clear buffer register (CPCLRBH, CPCLRBL) is a 16-bit buffer register which exists in the compare clear register (CPCLRH, CPCLRL).

Both the CPCLRBH, CPCLRBL register and the CPCLRH, CPCLRL register exist in the same address.

■ Compare-clear Buffer Register (CPCLRBH, CPCLRBL)

Compare clear buffer register (Upper)								
CPCLRBH	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Address: 0000A4H	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08
Write →	W	W	W	W	W	W	W	W
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
Compare clear buffer register (Lower)								
CPCLRBL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00
Write →	W	W	W	W	W	W	W	W
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)

The compare clear buffer register is a buffer register which exists in the same address of the compare clear register (CPCLRH, CPCLRL). When the buffer function is disabled (the lower of the timer state control register (TCCSL), BFE: bit7 = 0), or the free-run timer is stopped, the value of the compare clear buffer register is transferred to the compare clear register immediately. If the buffer function is enabled, the value is transferred to the compare clear register when the count value 0 of the 16-bit free-run timer is detected.

To access this register, use a halfword or word access instruction.

■ Compare Clear Register (CPCLR_H, CPCLR_L)

Compare clear register (Upper)								
CPCLR _H Address: 0000A4 _H	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08
Read →	R	R	R	R	R	R	R	R
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
Compare clear register (Lower)								
CPCLR _L Address: 0000A5 _H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00
Read →	R	R	R	R	R	R	R	R
Initial value →	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)

The compare clear register is used to compare with the count value of the 16-bit free-run timer. In the up count mode, the 16-bit free-run timer is reset to "0000_H" when this register matches with the count value of the 16-bit free-run timer. In up/down count mode, the 16-bit free-run timer changes its mode from the up counting to the down counting when this register matches the count value of the 16-bit free-run timer; or changes from the down counting to up counting when a zero detection occurs.

To access this register, use a halfword or word access instruction.

11.4.2 Timer Data Register (TCDTH, TCDTL)

The timer data register (TCDTH, TCDTL) is used to read the count value of the 16-bit free-run timer.

■ Timer Data Register (TCDTH, TCDTL)

Timer data register (Upper)								
TCDTH	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Address: 0000A6 _H	T15	T14	T13	T12	T11	T10	T09	T08
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
Timer data register (Lower)								
TCDTL	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Address: 0000A7 _H	T07	T06	T05	T04	T03	T02	T01	T00
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

The timer data register is used to read the count value of the 16-bit free-run timer. The count value is cleared to "0000_H" immediately when a reset occurs. Writing the value to this register can be set the timer value. However, write a value while the timer is stopped (the lower of the timer state control register (TCCSL), STOP: bit6 = 1). To access the timer data register, use a halfword or word access instruction.

16-bit free-run timer is initialized immediately when the following factors occur.

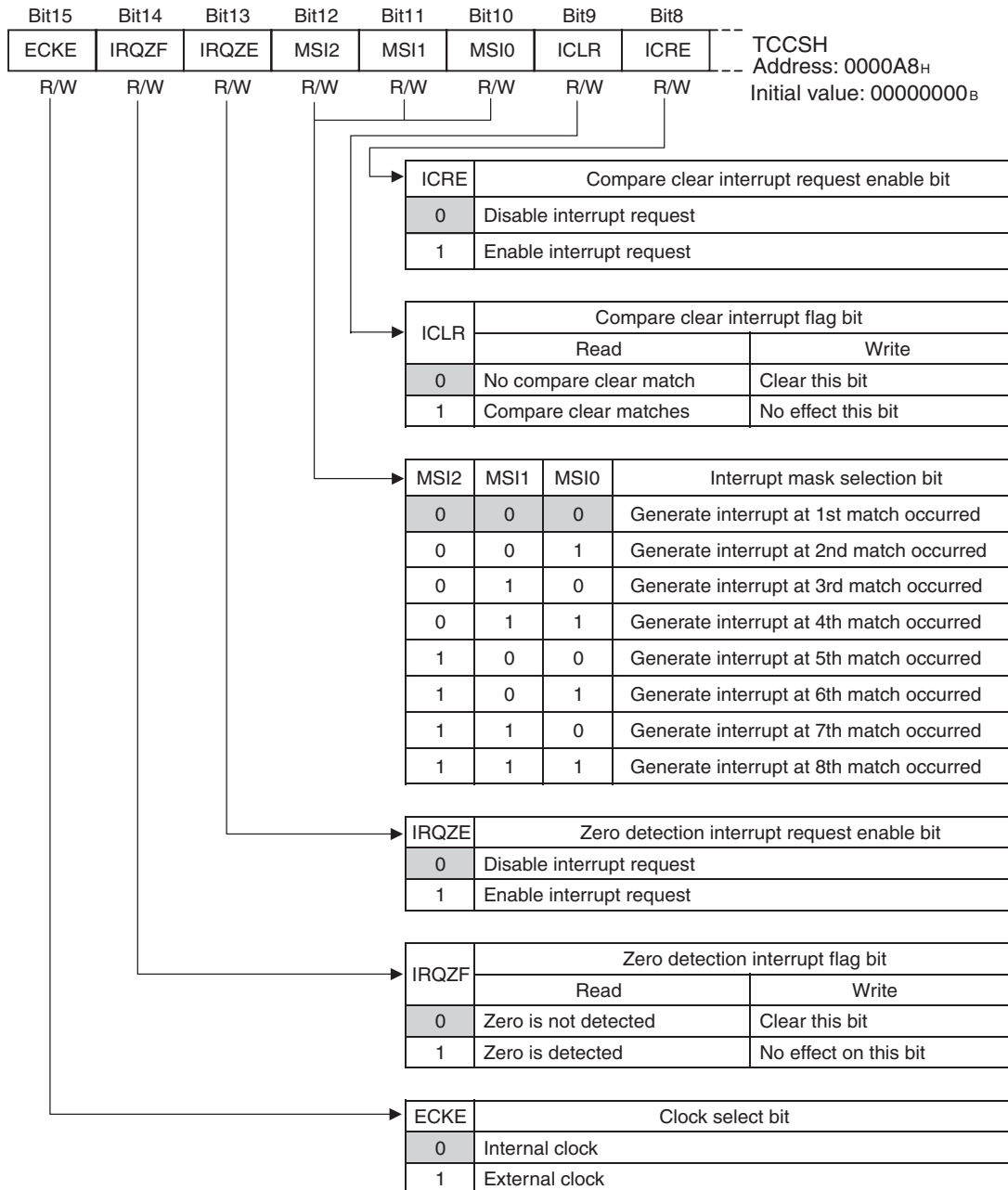
- Reset
- Clear bit of the timer state control register (TCCSL) (SCLR: bit4) = 1
- Match of the compare clear register and the timer count value in the up count mode (the lower of the timer state control register (TCCSL), MODE: bit5 = 0)

11.4.3 Timer State Control Register (TCCSH, TCCSL)

The timer state control register (TCCSH, TCCSL) is a 16-bit register which is used to control the operation of the 16-bit free-run timer.

■ Timer State Control Register, Upper Byte (TCCSH)

Timer state control register (Upper)



R/W: Readable/Writable

■ : Initial value

Table 11.4-1 Timer State Control Register, Upper Byte (TCCSH) (1 / 2)

Bit Name		Function
bit15	ECKE: Clock select bit	<ul style="list-style-type: none"> This bit is used to select the internal clock or the external clock as a count clock of the 16-bit free-run timer. Setting this bit to "0" selects the internal clock. To select a count clock frequency, you also need to select a clock frequency select bit of the TCCSL register (CLK 3 to CLK 0: bit3 to bit0). Setting this bit to "1" selects the external clock. The external clock is inputted from CKI pins. Therefore, write "0" to the bit7 of the port direction register (DDR1) to enable the external clock input. <p>Note: The count clock is changed immediately when this bit is set. Therefore, this bit must be changed when the output compare and the input capture are stopped.</p>
bit14	IRQZF: Zero detection interrupt flag bit	<ul style="list-style-type: none"> When the count value of the 16-bit free-run timer is "0000_H", this bit is set to "1". When this bit is set to "0": Clears the bit. Setting this bit to "1" has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read. <p>Note: In the up count mode (the lower of the timer state control register (TCCSL), MODE: bit5 = 0), this bit is set to "1" when the interrupt defined in the interrupt mask select bit (the upper of the timer state control register (TCCSH), MSI 2 to MSI 0: bit12 to bit10 is the value other than "000_B") occurs. When no interrupt occurs, this bit is not set to "1". In the up count mode (MODE: bit5 = 1), this bit is set every time the zero detection occurs regardless of the value for the MSI 2 to MSI 0: bit12 to bit10. When the timer count clock is the machine cycle (ϕ) The software clear (TCCSL: bit4 SCLR = 1) cannot be set this bit. When the timer count clock is the machine cycle (ϕ) division The software clear (TCCSL: bit4 SCLR = 1) can set this bit.</p>
bit13	IRQZE: Zero detection interrupt request enable bit	When this bit and the interrupt flag bit (IRQZF: bit14) are set to "1", an interrupt request to the CPU can be generated.
bit12 bit11 bit10	MSI2 to MSI0: Interrupt mask selection bit	<ul style="list-style-type: none"> In the up count mode (MODE = 0), these bits are used to set the number of masking of the compare clear interrupt. In up/down count mode (MODE = 1), they are used to set the number of masking for the zero detection interrupt. Setting these bits to "0" do not mask the interrupt cause. <p>Note: When the interrupt cause has been masked twice, and then the third interrupt is processed, these bits must be set to "010_B". The read value is masked count value.</p>

Table 11.4-1 Timer State Control Register, Upper Byte (TCCSH) (2 / 2)

Bit Name		Function
bit9	ICLR: Compare clear interrupt flag bit	<ul style="list-style-type: none"> • This bit is set to "1" when the value of the compare clear matches the value of the 16-bit free-run timer. • When this bit is set to "0": Clears the bit. • Setting this bit to "1" has no effect on this bit. • When this bit is read to a read modify write instruction, "1" is always read. <p>Note:</p> <p>In the up count mode (the lower of the timer state control register (TCCSL), MODE: bit5 = 0), this bit is set to "1" when the interrupt defined in the interrupt mask select bit (the upper of the timer state control register (TCCSH), MSI 2 to MSI 0: bit12 to bit10 is the value other than "000_B") occurs. When no interrupt occurs, this bit is not set to "1".</p> <p>In the up/down count mode (MODE = 1), this bit is set every time a compare clear occurs regardless of the value for the MSI 2 to MSI 0.</p>
bit8	ICRE: compare clear interrupt request enable bit	An interrupt request to the CPU can be generated when this bit and the compare clear interrupt flag bit (ICLR: bit9) are set to "1".

■ Timer State Control Register, Lower Byte (TCCSL)

Timer state control register (Lower)

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
BFE	STOP	MODE	SCLR	CLK3	CLK2	CLK1	CLK0	TCCSL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Address: 0000A9 _H Initial value: 01000000 _B

CLK3	CLK2	CLK1	CLK0	Clock frequency select bit					
				Count clock	ϕ =32MHz	ϕ =16MHz	ϕ =8MHz	ϕ =4MHz	ϕ =1MHz
0	0	0	0	ϕ	31.25 ns	62.5 ns	125 ns	0.25 μ s	1 μ s
0	0	0	1	ϕ /2	62.5 ns	125 ns	0.25 μ s	0.5 μ s	2 μ s
0	0	1	0	ϕ /4	125 ns	0.25 μ s	0.5 μ s	1 μ s	4 μ s
0	0	1	1	ϕ /8	0.25 μ s	0.5 μ s	1 μ s	2 μ s	8 μ s
0	1	0	0	ϕ /16	0.5 μ s	1 μ s	2 μ s	4 μ s	16 μ s
0	1	0	1	ϕ /32	1 μ s	2 μ s	4 μ s	8 μ s	32 μ s
0	1	1	0	ϕ /64	2 μ s	4 μ s	8 μ s	16 μ s	64 μ s
0	1	1	1	ϕ /128	4 μ s	8 μ s	16 μ s	32 μ s	128 μ s
1	0	0	0	ϕ /256	8 μ s	16 μ s	32 μ s	64 μ s	256 μ s
Disable other setting				-	-	-	-	-	-

 ϕ : Machine cycle

SCLR	Timer clear bit	
	Read	Write
0	Always read "0"	No effect on this bit
1		Initialized counter to "0000 _H "

MODE	Timer count mode bit
0	Up count mode
1	Up/down count mode

STOP	Timer enable bit
0	Enable count (start count)
1	Disable count (stop count)

BFE	Compare clear buffer enabled bit
0	Disable compare clear buffer
1	Enable compare clear buffer

R/W: Readable/Writable

 : Initial value

Table 11.4-2 Timer State Control Register, Lower Byte (TCCSL)

Bit Name		Function
bit7	BFE: Compare clear buffer enabled bit	<ul style="list-style-type: none"> This bit is used to enable a compare clear buffer. When this bit is set to "0": The compare clear buffer is disabled. Thus, the compare clear register (CPCLR_H, CPCLR_L) can be written directly. When this bit is set to "1": The compare clear buffer is enabled. The data written and stored into the compare clear buffer is transferred to the compare clear register when the count value "0" of the 16-bit free-run timer is detected.
bit6	STOP: Timer enable bit	<ul style="list-style-type: none"> This bit is used to start/stop the 16-bit free-run timer counting. When this bit is set to "0": the 16-bit free-run timer counting is started. When this bit is set to "1": the 16-bit free-run timer counting is stopped. <p>Note: When the 16-bit free-run timer stops, the operation of the output compare also stops.</p>
bit5	MODE: Timer count mode bit	<ul style="list-style-type: none"> This bit is used to select a count mode of the 16-bit free-run timer. When this bit is set to "0": The up count mode is selected. The timer continues to perform incremental counting until the count value matches a compare clear register and is reset "0000_H". Then, the timer restarts to perform incremental counting. When this bit is set to "1": The up/down count mode is selected. The timer continues to perform incremental counting until the count value matches a compare clear register. Then, the mode changes to the down count. After that, the mode changes to the up count again when the count value reaches to "0000_H". This bit can be written even if the timer is operating or stopped. When timer is operating, the value written to this bit is saved in a buffer. Then, the buffer value changes the count mode when the timer value becomes "0000_H".
bit4	SCLR: Timer clear bit	<ul style="list-style-type: none"> This bit is used to initialize the 16-bit free-run timer to "0000_H". When this bit is set to "0": there is no meaning. When this bit is set to "1": the 16-bit free-run timer is initialized to "0000_H". The read value is always "0". <p>Notes:</p> <ul style="list-style-type: none"> When the timer count clock is the machine cycle (ϕ) Writing "1" to this bit does not set the zero detection interrupt flag. The zero detection interrupt does not occur. When the timer count clock is the machine cycle (ϕ) division Writing "1" to this bit sets the zero detection interrupt flag. If the interrupt is enabled, the zero detection interrupt occurs.
bit3 bit2 bit1 bit0	CLK3 to CLK0: Clock frequency select bit	<ul style="list-style-type: none"> These bits are used to select a count clock frequency of the 16-bit free-run timer. The count clock is changed immediately when these bits are set. Therefore, these bits must be changed when the output compare and the input capture are stopped.

11.4.4 A/D Trigger Control Register (ADTRGC)

This register controls the A/D trigger signal output when a free-run timer compare match or a zero detection occurs.

■ A/D Trigger Control Register (ADTRGC)

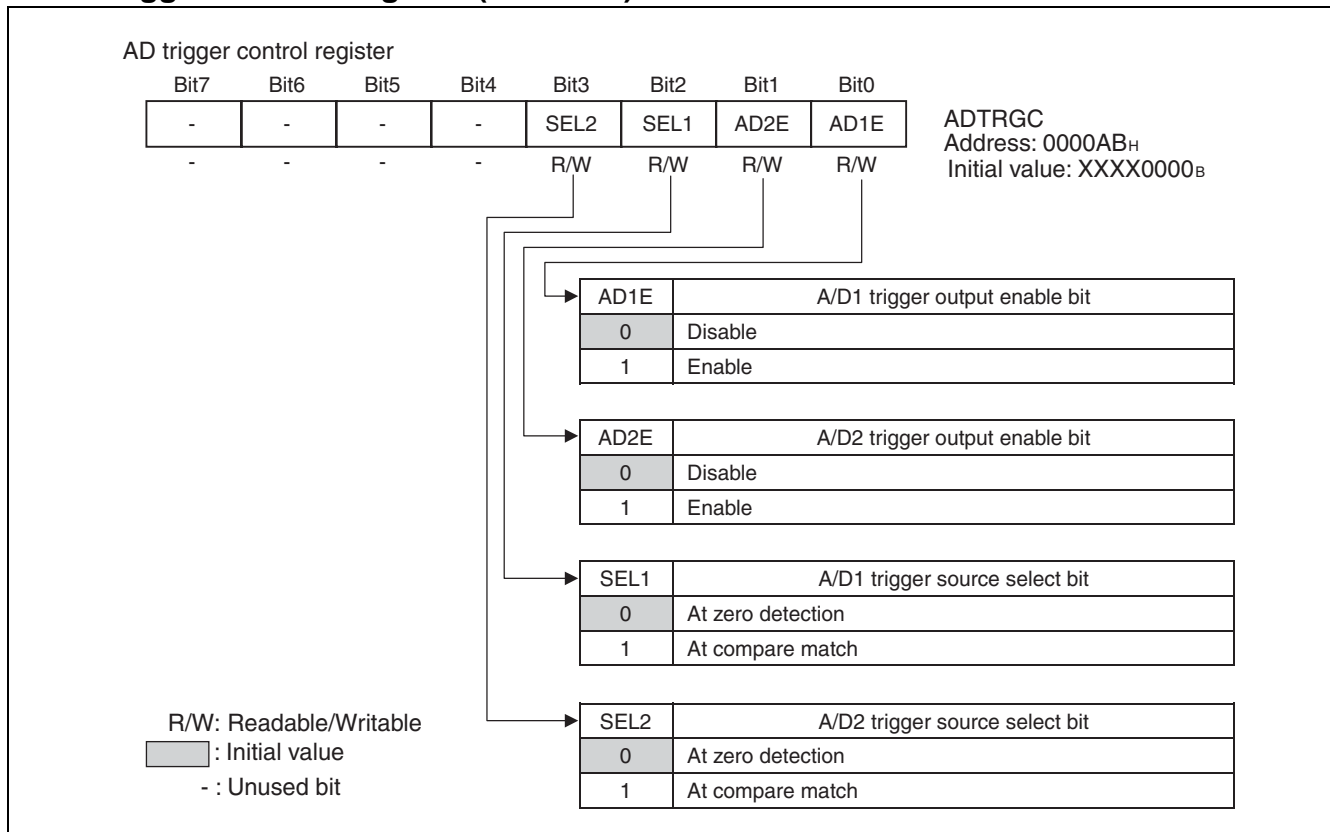


Table 11.4-3 A/D Trigger Control Register (ADTRGC)

Bit Name		Function
bit7 bit6 bit5 bit4	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect on operation.
bit3	SEL2: A/D2 trigger source select bit	This bit is the select bit if A/D2 trigger is outputted when a zero detection of the free-run timer or a compare match occurs.
bit2	SEL1: A/D1 trigger source select bit	This bit is the select bit if A/D1 trigger is outputted when a zero detection of the free-run timer or a compare match occurs.
bit1	AD2E: A/D2 trigger enable bit	<ul style="list-style-type: none"> When this bit is set to "0", the A/D2 trigger signal is not outputted. When this bit is set to "1", the A/D2 trigger signal can be outputted.
bit0	AD1E: A/D1 trigger enable bit	<ul style="list-style-type: none"> When this bit is set to "0", the A/D1 trigger signal is not outputted. When this bit is set to "1", the A/D1 trigger signal can be outputted.

11.4.5 Output Compare Buffer Register (OCCPBH0 to OCCPBH5, OCCPBL0 to OCCPBL5) / Output Compare Register (OCCPH0 to OCCPH5, OCCPL0 to OCCPL5)

The output compare buffer register (OCCPBH, OCCPBL) is a 16-bit buffer register for the output compare register (OCCPH, OCCPL).

Both the OCCPBH, OCCPBL register and the OCCPH, OCCPL register exist in the same address.

■ Output Compare Buffer Register (OCCPBH: OCCPBH0 to OCCPBH5, OCCPBL: OCCPBL0 to OCCPBL5)

Output compare buffer register (Upper)									
OCCPBH0 to OCCPBH5	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	---
Address :	OP15	OP14	OP13	OP12	OP11	OP10	OP09	OP08	---
000090H									
000092H									
000094H	Write →	W	W	W	W	W	W	W	
000096H	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	
000098H									
00009AH									
Output compare buffer register (Lower)									
OCCPBL0 to OCCPBL5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	OP07	OP06	OP05	OP04	OP03	OP02	OP01	OP00	
	Write →	W	W	W	W	W	W	W	
	Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	

The output compare buffer register is a buffer register for the output compare register (OCCPH, OCCPL). When the buffer function is disabled (the lower of the compare control register (OCSL0, OCSL2, OCSL4), BUF1, BUF0: bit3, bit2 = "11_B"), or the free-run timer is stopped, the value of the output compare buffer register is transferred to the output compare register immediately. When the buffer function is enabled (the lower of the compare control register (OCSL0, OCSL2, OCSL4), BUF1, BUF0: bit3, bit2 = "00_B"), the value is transferred according to the transfer select bit (BTS1, BTS0: bit14, bit13) of the upper of the compare control register (OCSH1, OCSH3, OCSH5) when a compare clear match or a zero detection occurs.

To access this register, use a halfword or word access instruction.

■ Output Compare Register (OCCPH: OCCPH0 to OCCPH5, OCCPL: OCCPL0 to OCCPL5)

Output compare register (Upper)								
OCCPH0 to OCCPH5	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Address:	OP15	OP14	OP13	OP12	OP11	OP10	OP09	OP08
000090H								
000092H	Read → R	R	R	R	R	R	R	R
000094H	Initial value → (0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
000096H								
000098H								
00009AH								
Output compare register (Lower)								
OCCPL0 to OCCPL5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	OP07	OP06	OP05	OP04	OP03	OP02	OP01	OP00
Read	R	R	R	R	R	R	R	R
Initial value →	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

The output compare register is a 16-bit register used to compare with the count value of the 16-bit free-run timer. Set the value of the output compare buffer register (OCCPBH, OCCPBL) before the timer operation is enabled.

When the value of the output compare register matches the count value of the 16-bit free-run timer, a compare signal is generated and the output compare interrupt flag bit (the lower of the compare control register (OCSL0, OCSL2 OCSL4), IOP1, IOP0: bit7, bit6) is set. When the output level is set (the upper of the compare control register (OCSH1, OCSH3, OCSH5), OTD1, OTD0: bit9, bit8), an output level waveform generator (RTO0 to RTO5) corresponding to the output compare register (OCCPH0 to OCCPH5, OCCPL0 to OCCPL5) can be reversed.

To access this register, use a halfword or word access instruction.

11.4.6 Compare Control Register (OCSH0 to OCSH5, OCSL0 to OCSL5)

The compare control register is used to control the output level, output enable, output level reverse mode, compare operation enable, compare match interrupt enable, and compare match interrupt flag of RT0 to RT5.

■ Compare Control Register, Upper Byte (OCSH1, OCSH3, OCSH5)

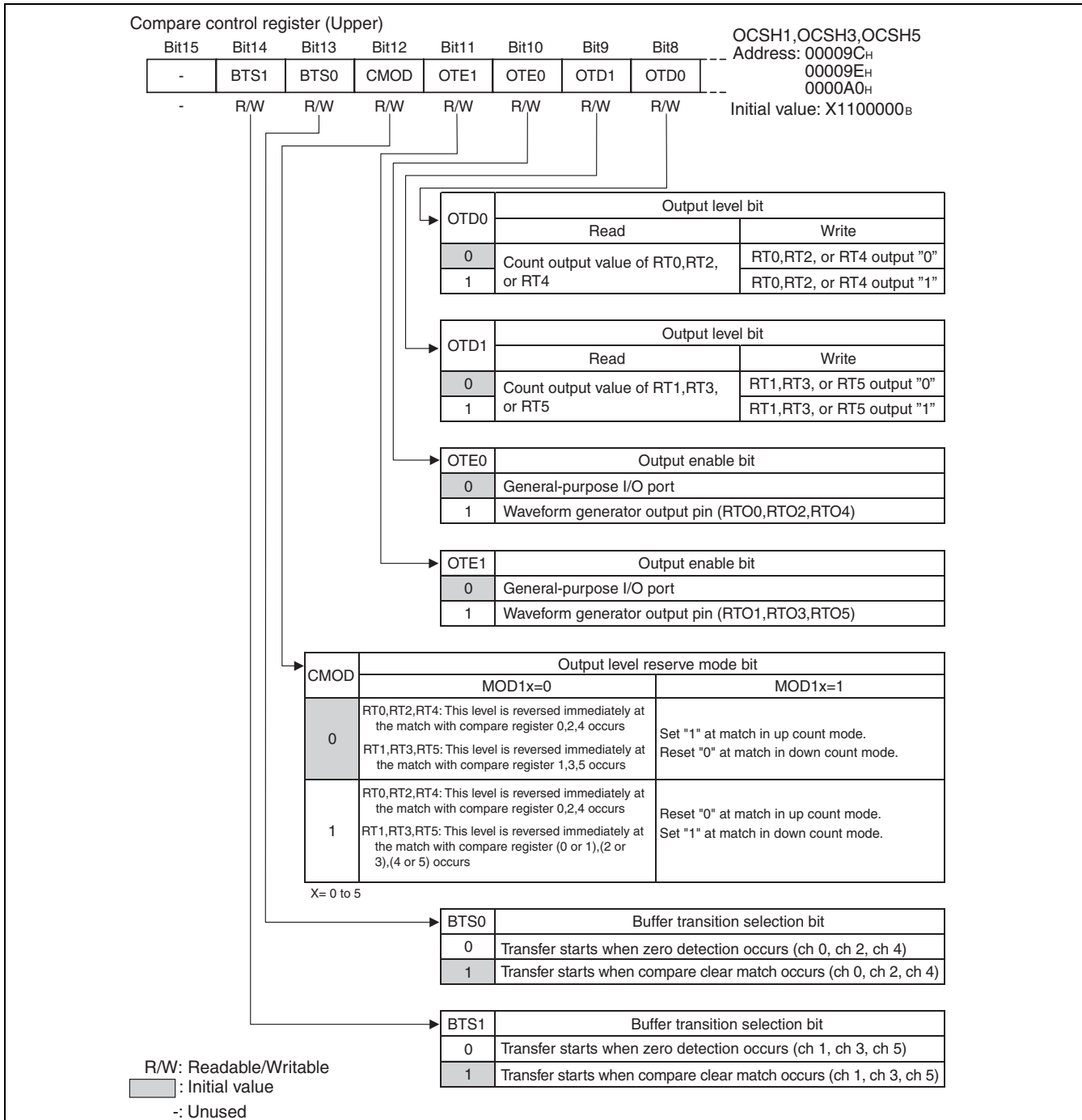


Table 11.4-4 Compare Control Register, Upper Byte (OCSH1, OCSH3, OCSH5) (1 / 2)

Bit Name		Function
bit15	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to this bit has no effect on operation.
bit14	BTS1: Buffer transition selection bit	<ul style="list-style-type: none"> This bit is used to select the time when the data is transferred from the output compare buffer registers (OCCPBH0, OCCPBH2, OCCPBH4, OCCPBL0, OCCPBL2, OCCPBL4) to the output compare registers (OCCPH1, OCCPH3, OCCPH5, OCCPL1, OCCPL3, OCCPL5). Setting this bit to "0" starts the data transfer when the count value 0 of the 16-bit free-run timer is detected. Setting this bit to "1" starts the data transfer when a compare clear match of the 16-bit free-run timer occurs.
bit13	BTS0: Buffer transition selection bit	<ul style="list-style-type: none"> This bit is used to select the time when the data is transferred from the output compare buffer registers (OCCPBH0, OCCPBH2, OCCPBH4, OCCPBL0, OCCPBL2, OCCPBL4) to the output compare registers (OCCPH1, OCCPH3, OCCPH5, OCCPL1, OCCPL3, OCCPL5). Setting this bit to "0" starts the data transfer when the count value 0 of the 16-bit free-run timer is detected. Setting this bit to "1" starts the data transfer when a compare clear match of the 16-bit free-run timer occurs.
bit12	CMOD: Output level reverse mode bit	<ul style="list-style-type: none"> This bit is used to switch the pin output level reverse mode immediately when the match occurs while the pin output is enabled (OTE1 = 1 or OTE0 = 1). When this bit is set to "0": The compare mode control register (OCMOD): MOD1x = 0 <ul style="list-style-type: none"> RT0, RT2, RT4: The level is reversed immediately when the compare registers 0, 2, 4 match the 16-bit free-run timer. RT1, RT3, RT5: The level is reversed immediately when the compare registers 1, 3, 5 match the 16-bit free-run timer. The compare mode control register (OCMOD): MOD1x = 1 <ul style="list-style-type: none"> Set to "1" when the match occurs in the up count mode. Reset to "0" when the match occurs in the down count mode. When this bit is set to "1": The compare mode control register (OCMOD): MOD1x = 0 <ul style="list-style-type: none"> RT0, RT2, RT4: The level is reversed immediately when the compare registers 0, 2, 4 match the 16-bit free-run timer. RT1, RT3, RT5: The level is reversed immediately when the compare registers (0 or 1) (2 or 3) (4 or 5) match the 16-bit free-run timer. When the value of the compare register 0, 2, 4 and 1, 3, 5 is the same, the operation is the same operation as only one compare register is used. The compare mode control register (OCMOD): MOD1x = 1 <ul style="list-style-type: none"> Reset to "0" when the match occurs in the up count mode. Set to "1" when the match occurs in the down count mode.

Table 11.4-4 Compare Control Register, Upper Byte (OCSH1, OCSH3, OCSH5) (2 / 2)

Bit Name		Function
bit11	OTE1: Output enable bit	<ul style="list-style-type: none"> This bit is used to enable the waveform generator output (RTO1, RTO3, RTO5) to ports. The initial value of this bit is "0". <p>Note: When the waveform generator is disabled (the lower of the 16-bit dead timer control register (DTCR0, DTCR1, DTCR2), TMD2 to TMD0, TMD5 to TMD3, TMD8 to TMD6: bit2 to bit0 = 000_B), RTO1, RTO3, RTO5 outputs the same value as the output compare.</p>
bit10	OTE0: Output enable bit	<ul style="list-style-type: none"> This bit is used to enable the waveform generator output (RTO0, RTO2, RTO4) to ports. The initial value of this bit is "0". <p>Note: When the waveform generator is disabled (the lower of the 16-bit dead timer control register (DTCR0, DTCR1, DTCR2), TMD2 to TMD0, TMD5 to TMD3, TMD8 to TMD6: bit2 to bit0 = 000_B), RTO0, RTO2, RTO4 outputs the same value as the output compare.</p>
bit9	OTD1: Output level bit	<ul style="list-style-type: none"> This bit is used to change a pin output level of the output compare 1, 3, 5 (RT1, RT3, RT5). The initial value of the compare pin output is "0". Before data is written, be sure to stop the compare operation. The read value of this bit indicates the output compare value of RT1, RT3, RT5.
bit8	OTD0: Output level bit	<ul style="list-style-type: none"> This bit is used to change a pin output level of the output compare 0, 2, 4 (RT0, RT2, RT4). The initial value of the compare pin output is "0". Before data is written, be sure to stop the compare operation. The read value of this bit indicates the output compare value of RT0, RT2, RT4.

■ Compare Control Register, Lower Byte (OCSL0, OCSL2, OCSL4)

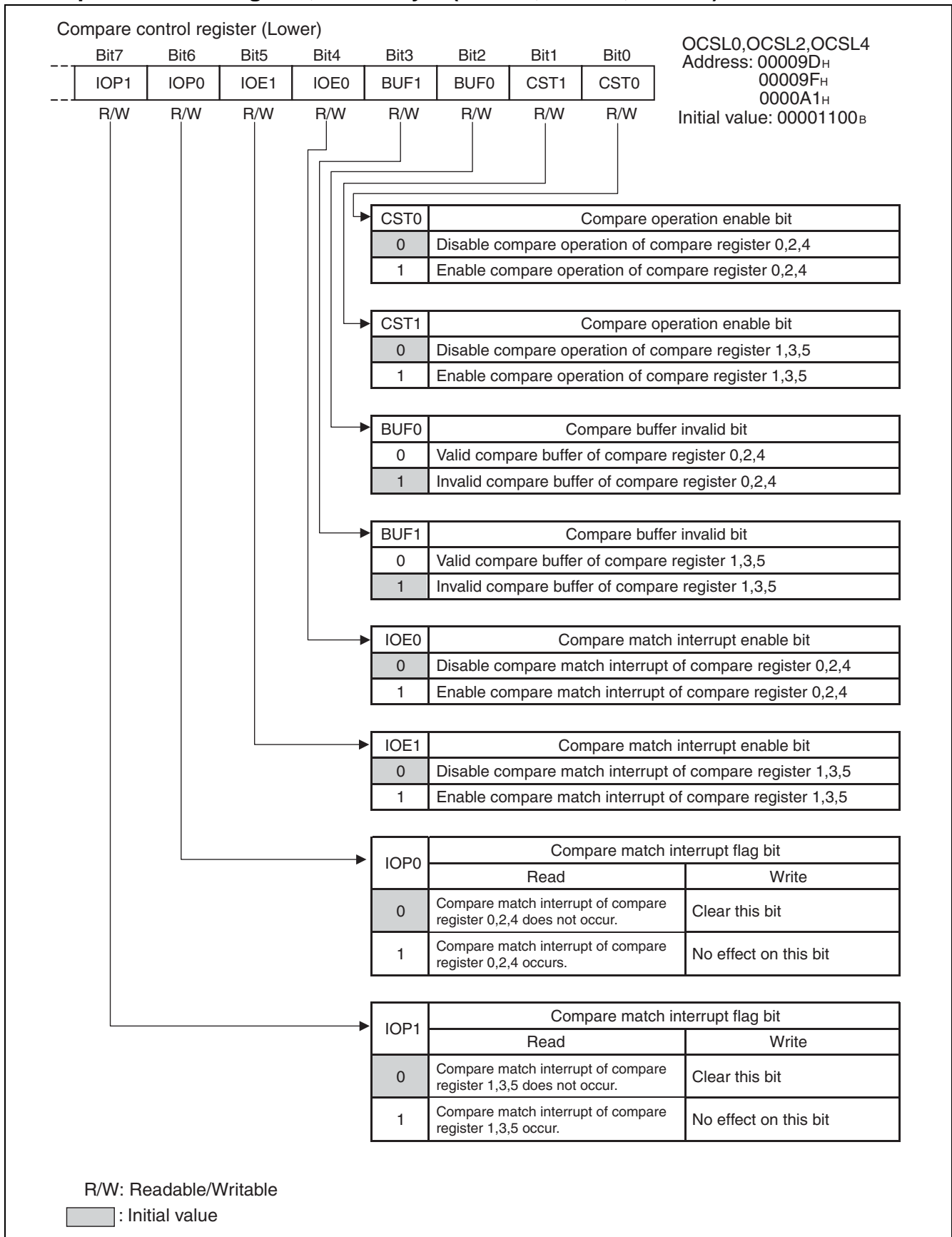


Table 11.4-5 Compare Control Register, Lower Byte (OCSL0, OCSL2, OCSL4)

Bit Name		Function
bit7	IOP1: Compare match interrupt flag bit	<ul style="list-style-type: none"> This bit is an interrupt flag which indicates that the compare register 1, 3, 5 matches the value of the 16-bit free-run timer. This bit is set to "1" when the value of the compare register matches the value of the 16-bit free-run timer. When this bit is set while the compare match interrupt enable bit (IOE1: bit5) is enabled, the output compare interrupt occurs. When this bit is set to "0": Clears the bit. Setting this bit to "1" has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read.
bit6	IOP0: Compare match interrupt flag bit	<ul style="list-style-type: none"> This bit is an interrupt flag which indicates that the compare register 0, 2, 4 matches the value of the 16-bit free-run timer. This bit is set to "1" when the value of the compare register matches the value of the 16-bit free-run timer. When this bit is set while the compare match interrupt enable bit (IOE0: bit4) is enabled, the output compare interrupt occurs. When this bit is set to "0": Clears the bit. Setting this bit to "1" has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read.
bit5	IOE1: Compare match interrupt enable bit	<ul style="list-style-type: none"> This bit is used to enable the output compare interrupt of the compare register 1, 3, 5. When the compare match interrupt flag bit (IOP1: bit7) is set while "1" is written to this bit, the output compare interrupt occurs.
bit4	IOE0: Compare match interrupt enable bit	<ul style="list-style-type: none"> This bit is used to enable the output compare interrupt of the compare register 0, 2, 4. When the compare match interrupt flag bit (IOP0: bit6) is set while "1" is written to this bit, the output compare interrupt occurs.
bit3	BUF1: Compare buffer disable bit	<ul style="list-style-type: none"> This bit is used to disable the buffer function of the output compare register 1, 3, 5. Setting this bit to "0" enables the buffer function.
bit2	BUF0: Compare buffer disable bit	<ul style="list-style-type: none"> This bit is used to disable the buffer function of the output compare register 0, 2, 4. Setting this bit to "0" enables the buffer function.
bit1	CST1: Compare operation enable bit	<ul style="list-style-type: none"> This bit is used to enable the compare operation between the 16-bit free-run timer and compare register 1, 3, 5. Before the compare operation is enabled, be sure to write the value to the compare register 1, 3, 5 and the timer data register (TCDTH, TCDTL). <p>Note: The output compare is synchronized with the 16-bit free-run timer clock. Therefore, when the 16-bit free-run timer is stopped, the compare operation is also stopped.</p>
bit0	CST0: Compare operation enable bit	<ul style="list-style-type: none"> This bit is used to enable the compare operation between the 16-bit free-run timer and compare register 0, 2, 4. Before the compare operation is enabled, be sure to write the value to the compare register 0, 2, 4 and the timer data register (TCDTH, TCDTL). <p>Note: The output compare is synchronized with the 16-bit free-run timer clock. Therefore, when the 16-bit free-run timer is stopped, the zero detection and the compare operation are also stopped.</p>

11.4.7 Compare Mode Control Register (OCMOD)

The compare mode control register controls to reverse or set/reset the output level when the compare match occurs.

■ Compare Mode Control Register (OCMOD)

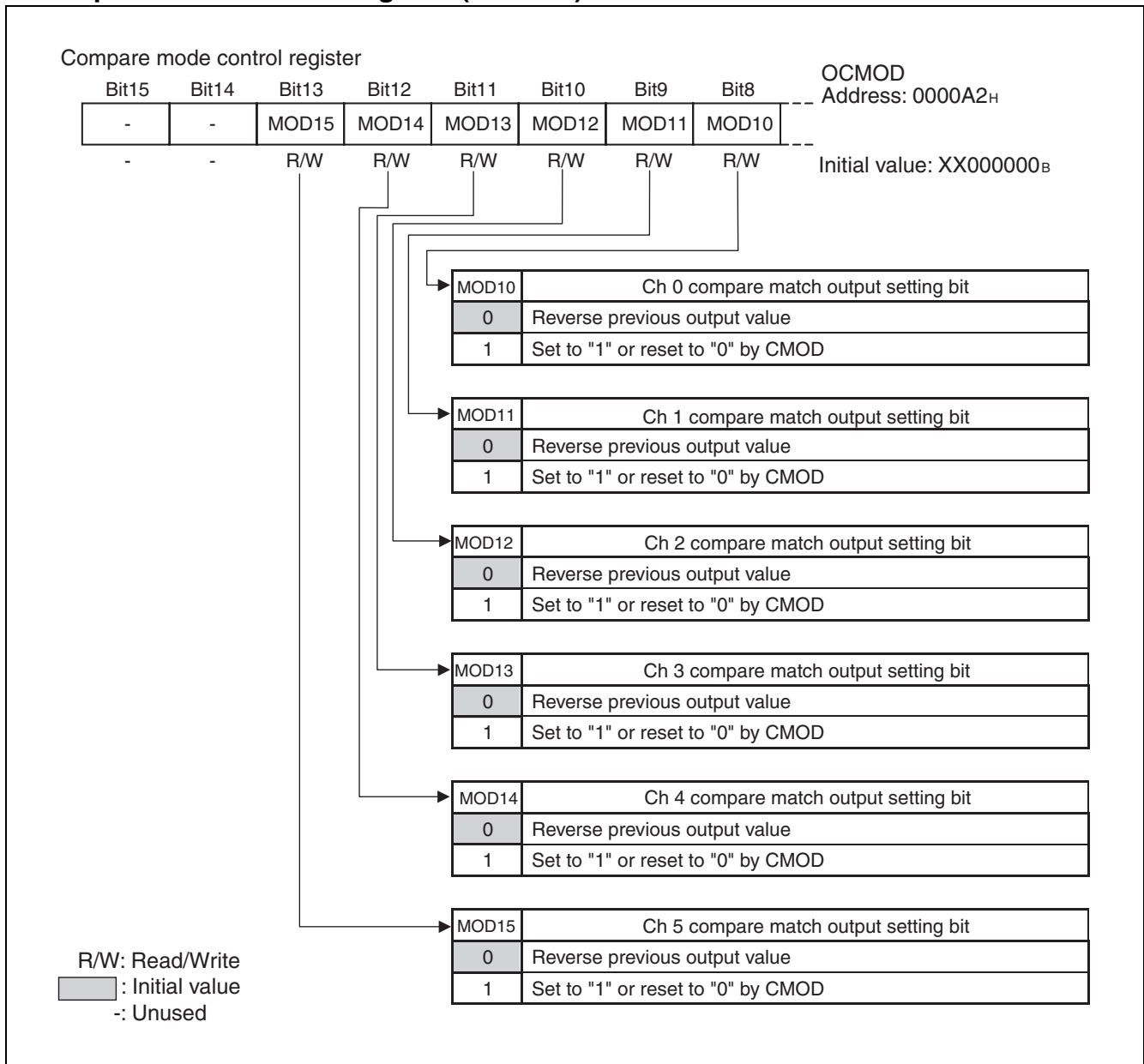


Table 11.4-6 Compare Mode Control Register (OCMOD)

Bit Name		Function
bit15, bit14	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect on the operation.
bit13	MOD15: ch5 compare match output setting bit	<ul style="list-style-type: none"> This bit specifies the operation when the compare match of the output compare output occurs. The initial value is "0" When the bit is set to "0", reverse the output value temporarily when the match occurs. When the bit is set to "1", set the output value to "1" or reset the output value to "0" when the match occurs. CMOD bit of the compare control register (OCSH) sets the set/reset switch. Before data is written, be sure to stop the compare operation. CMOD is set for ch0 and ch1, ch2 and ch3, ch4 and ch5. <ul style="list-style-type: none"> Reset/Set is not available to ch0 and ch1 separately. Reset/Set is not available to ch2 and ch3 separately. Reset/Set is not available to ch4 and ch5 separately.
bit12	MOD14: ch4 compare match output setting bit	
bit11	MOD13: ch3 compare match output setting bit	
bit10	MOD12: ch2 compare match output setting bit	
bit9	MOD11: ch1 compare match output setting bit	
bit8	MOD10: ch0 compare match output setting bit	

11.4.8 Input Capture Data Registers (IPCPH0 to IPCPH3, IPCPL0 to IPCPL3)

The input capture data register is used to store the count value of the free-run timer on detection of a valid edge for the input waveform.

■ Input Capture Data Register (IPCPH: IPCPH0 to IPCPH3, IPCPL: IPCPL0 to IPCPL3)

Input capture data register (Upper)									
IPCPH0 to IPCPH3		Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Address:		CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08
0000AC _H									
0000AE _H		Read →	R	R	R	R	R	R	R
0000B0 _H		Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)
0000B2 _H									
Input capture data register (Lower)									
IPCPL0 to IPCPL3		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00
		Read →	R	R	R	R	R	R	R
		Initial value →	(X)	(X)	(X)	(X)	(X)	(X)	(X)

This register is used to store the value of the free-run timer on detection of a valid edge for the corresponding external pin input waveform. (To access this register, use a halfword or word access instruction. Data cannot be written to this register.)

11.4.9 Input Capture State Control/PPG Output Control Register (ICSH23, ICSL23, PICSH01, PICSL01)

The input capture state control/PPG output control register (ICSH23, ICSL23, PICSH01, PICSL01) is used to control the edge selection, the interrupt request enable, the interrupt request flag, and the PPG output. This register is also used to indicate a valid edge which was detected on the input capture 2, 3.

■ Input Capture State Control Register (ch.2, ch.3), Upper Byte (ICSH23)

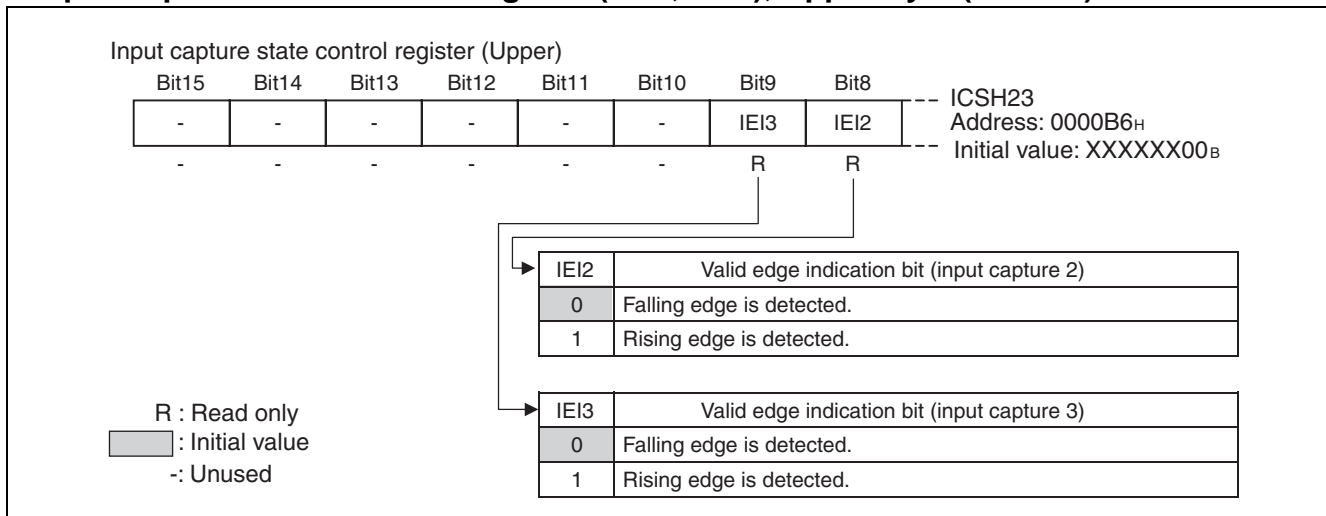


Table 11.4-7 Input Capture State Control Register (ch.2, ch.3), Upper Byte (ICSH23)

Bit Name		Function
bit15 bit14 bit13 bit12 bit11 bit10	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect on operation.
bit9	IEI3: Valid edge indication bit (input capture3)	<ul style="list-style-type: none"> This bit is a valid edge indication bit of the capture register 3, and indicates that the rising or falling edge was detected. When the falling edge is detected, "0" is written to this bit. When the rising edge is detected, "1" is written to this bit. This bit is a read-only bit. <p>Note: When the lower of the input capture state control register (ICSL23), EG31, EG30: bit3, bit 2 = 00_B, the read value has no meaning.</p>
bit8	IEI2: Valid edge indication bit (Input capture 2)	<ul style="list-style-type: none"> This bit is a valid edge indication bit of the capture register 2, and indicates that the rising or falling edge was detected. When the falling edge is detected, "0" is written to this bit. When the rising edge is detected, "1" is written to this bit. This bit is a read-only bit. <p>Note: When the lower of the input capture state control register (ICSL23), EG21, EG20: bit1, bit0 = 00_B, the read value has no meaning.</p>

■ Input Capture State Control Register (ch.2, ch.3), Lower Byte (ICSL23)

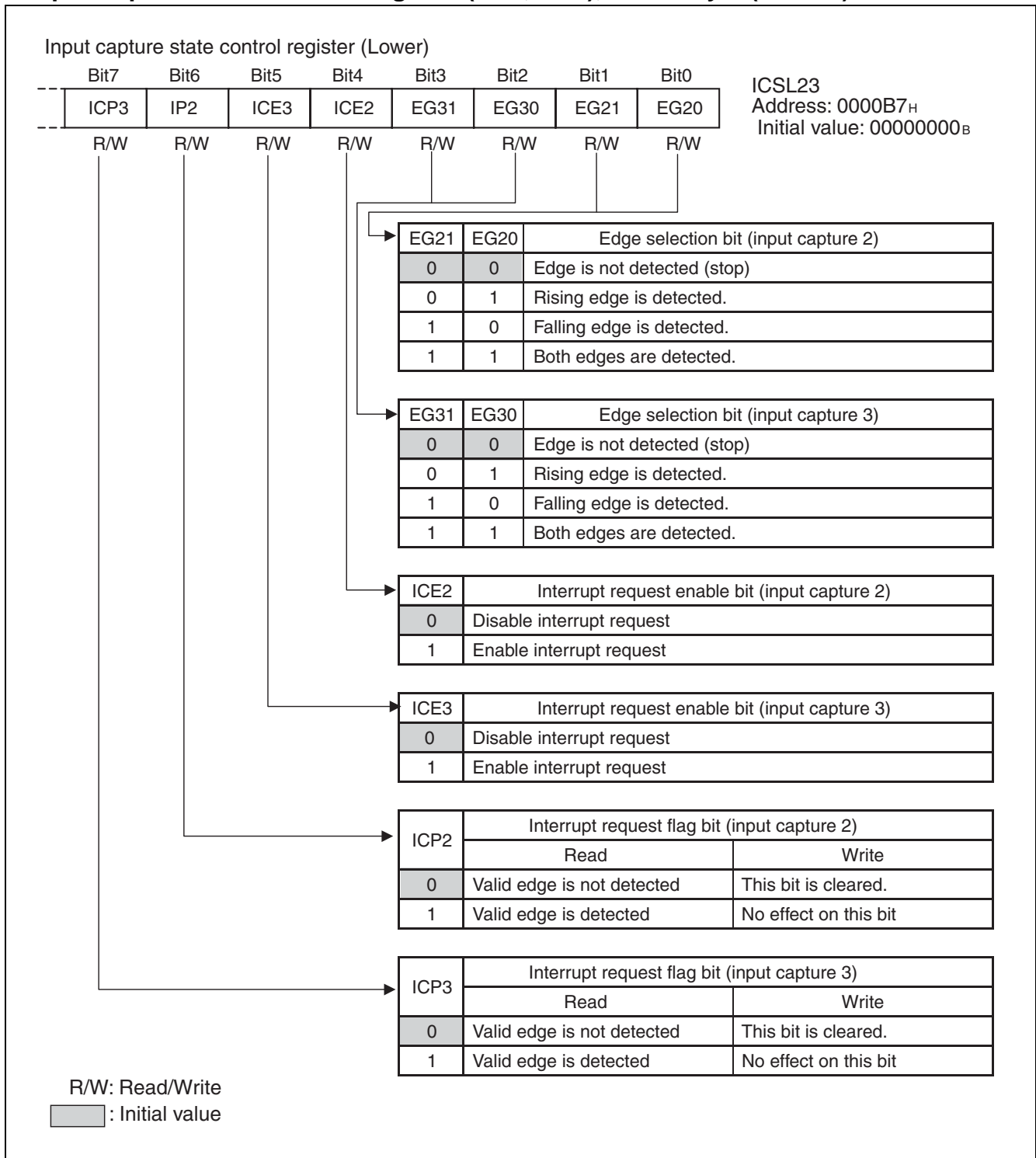


Table 11.4-8 Input Capture State Control Register (ch.2, ch.3), Lower Byte (ICSL23)

Bit Name		Function
bit7	ICP3: Interrupt request flag bit (Input capture 3)	<ul style="list-style-type: none"> • This bit is used as an interrupt request flag of the input capture 3. • This bit is set to "1" immediately when a valid edge of an external input pin is detected. • When a valid edge is detected while the interrupt enable bit (ICE3: bit5) is set, the interrupt can be generated immediately. • When this bit is set to "0": Clears the bit. • Setting this bit to "1" has no effect on this bit. • When this bit is read to a read modify write instruction, "1" is always read.
bit6	ICP2: Interrupt request flag bit (Input capture 2)	<ul style="list-style-type: none"> • This bit is used as an interrupt request flag of the input capture 2. • This bit is set to "1" immediately when a valid edge of an external input pin is detected. • When a valid edge is detected while the interrupt enable bit (ICE2: bit4) is set, the interrupt can be generated immediately. • When this bit is set to "0": Clears the bit. • Setting this bit to "1" has no effect on this bit. • When this bit is read to a read modify write instruction, "1" is always read.
bit5	ICE3: Interrupt request enable bit (Input capture 3)	<ul style="list-style-type: none"> • This bit is used to enable an input capture interrupt request of the input capture 3. • When the interrupt flag bit (ICP3: bit7) is set while this bit is set to "1", the input capture 3 interrupt is generated.
bit4	ICE2: Interrupt request enable bit (Input capture 2)	<ul style="list-style-type: none"> • This bit is used to enable an input capture interrupt request of the input capture 2. • When the interrupt flag bit (ICP2: bit6) is set while this bit is set to "1", the input capture 2 interrupt is generated.
bit3 bit2	EG31, EG30: Edge selection bit (Input capture 3)	<ul style="list-style-type: none"> • These bits are used to specify the valid edge polarity of the external input for the input capture 3. • These bits are used also to enable an operation of the input capture 3.
bit1 bit0	EG21, EG20: Edge selection bit (Input capture 2)	<ul style="list-style-type: none"> • These bits are used to specify the valid edge polarity of the external input for the input capture 2. • These bits are used also to enable an operation of the input capture 2.

■ PPG Output Control Register, Upper Byte (PICSH01)

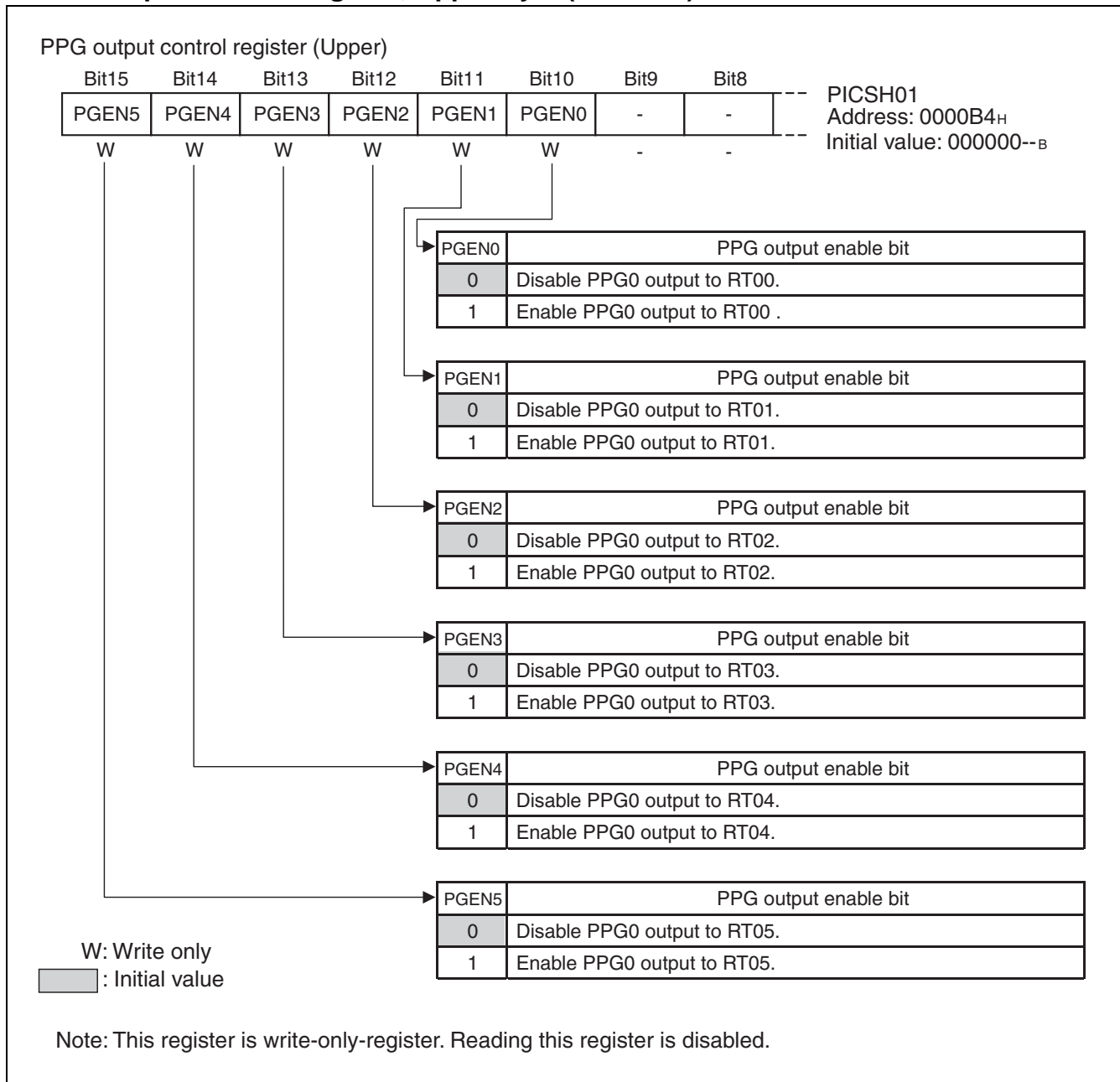


Table 11.4-9 PPG Output Control, Upper Byte (PICSH01)

Bit Name		Function
bit15 bit14 bit13 bit12 bit11 bit10	PGEN5 to PGEN0: PPG output enable bit	<ul style="list-style-type: none"> These bits are used to select the PPG0 output to RTO0 to RTO5. Only write can be performed. Reading these bits are disabled.

■ PPG Control Register, Lower Byte (PICSL01)

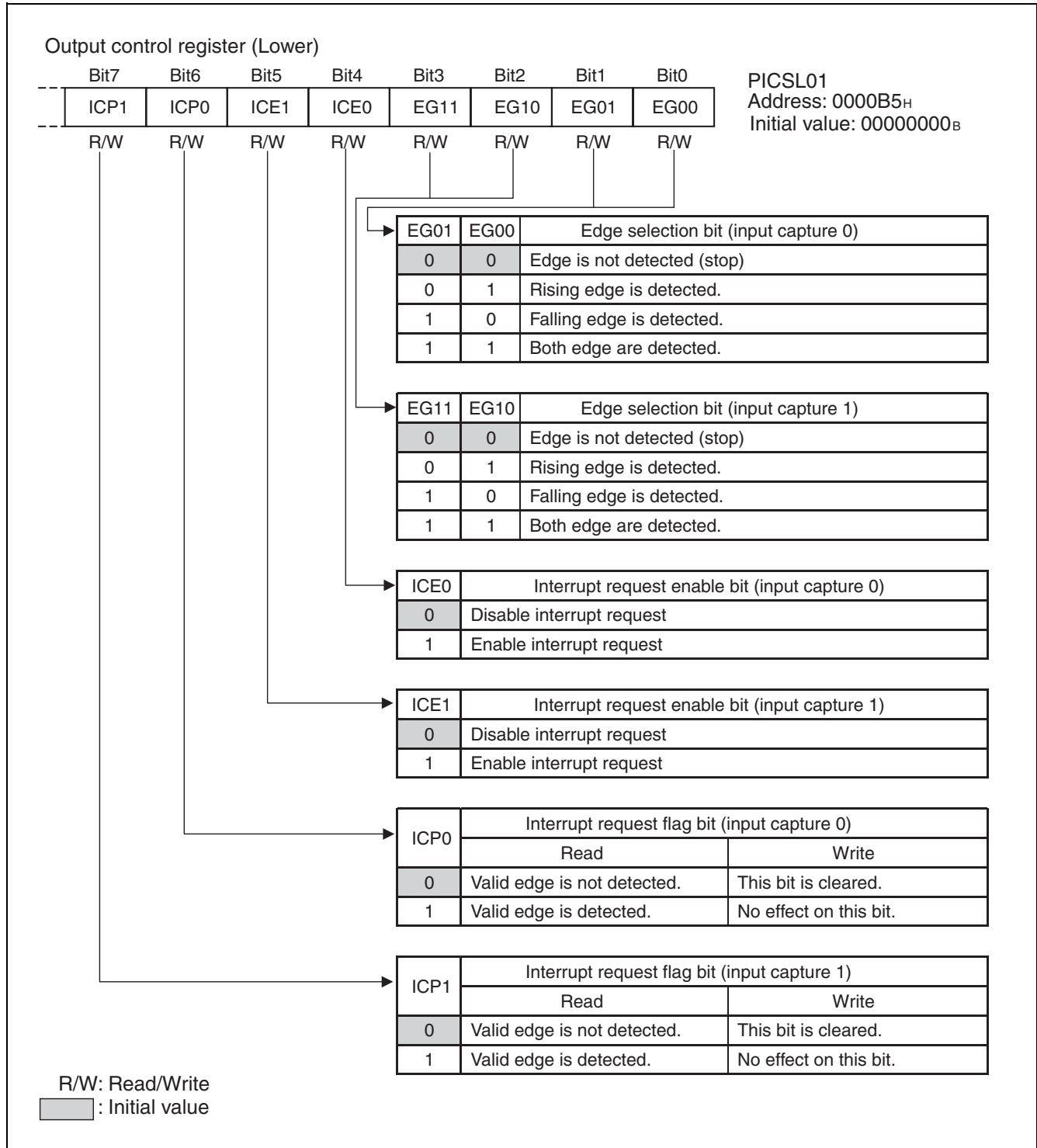


Table 11.4-10 PPG Output Control Register, Lower Byte (PICSL01)

Bit Name		Function
bit7	ICP1: Interrupt request flag bit (Input capture 1)	<ul style="list-style-type: none"> • This bit is used as an interrupt request flag of the input capture 1. • This bit is set to "1" immediately when a valid edge of an external input pin is detected. • When a valid edge is detected while the interrupt enable bit (ICE1: bit5) is set, the interrupt is generated immediately. • When this bit is set to "0": Clears the bit. • Setting this bit to "1" has no effect on this bit. • When this bit is read to a read modify write instruction, "1" is always read.
bit6	ICP0: Interrupt request flag bit (Input capture 0)	<ul style="list-style-type: none"> • This bit is used as an interrupt request flag of the input capture 0. • This bit is set to "1" immediately when a valid edge of an external input pin is detected. • When a valid edge is detected while the interrupt enable bit (ICE0: bit4) is set, the interrupt is generated immediately. • When this bit is set to "0": Clears the bit. • Setting this bit to "1" has no effect on this bit. • When this bit is read to a read modify write instruction, "1" is always read.
bit5	ICE1: Interrupt request enable bit (Input capture 1)	<ul style="list-style-type: none"> • This bit is used to enable an input capture interrupt request of the input capture 1. • When the interrupt flag bit (ICP1: bit7) is set while this bit is set to "1", the input capture 1 interrupt is generated.
bit4	ICE0: Interrupt request enable bit (Input capture 0)	<ul style="list-style-type: none"> • This bit is used to enable an input capture interrupt request of the input capture 0. • When the interrupt flag bit (ICP0: bit6) is set while this bit is set to "1", the input capture 0 interrupt is generated.
bit3 bit2	EG11, EG10: Edge selection bit (Input capture 1)	<ul style="list-style-type: none"> • These bits are used to specify the valid edge polarity of the external input for the input capture 1. • These bits are used also to enable an operation of the input capture 1.
bit1 bit0	EG01, EG00: Edge selection bit (Input capture 0)	<ul style="list-style-type: none"> • These bits are used to specify the valid edge polarity of the external input for the input capture 0. • These bits are used also to enable an operation of the input capture 0.

11.4.10 16-bit Dead Timer Register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2)

The 16-bit dead timer register stores the compare value of the 16-bit dead timer.

■ 16-bit Dead Timer Register (TMRRH: TMRRH0 to TMRRH2, TMRRL: TMRRL0 to TMRRL2)

16-bit dead timer register (Upper)										
TMRRH0 to TMRRH2		Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	--
Address:		TR15	TR14	TR13	TR12	TR11	TR10	TR09	TR08	--
0000BC _H										
0000BE _H										
0000C0 _H										
Read/Write →		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →		(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	
16-bit dead timer register (Lower)										
TMRRL0 to TMRRL2		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
		TR07	TR06	TR05	TR04	TR03	TR02	TR01	TR00	
Read/Write →		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →		(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

These registers are used to store the compare value of the 16-bit dead timer.

These register values are reloaded when the 16-bit dead timer starts the operation.

If the values are rewritten to these registers during the timer operation, these new values are enabled in the next timer start/operation.

To access these registers, use a halfword or word access instruction.

In the dead time timer mode, these registers are used to set the non-overlap time.

Non-overlap time = (Setting value) × Selected clock

In the timer mode, these registers are used to set the GATE time of the PPG0 timer operation.

GATE time = (Setting value) × Selected clock

Note:

In the dead time timer mode and the time timer mode, "0000_H" cannot be set to these registers.

11.4.11 16-bit Dead Timer Control Register (DTCR0 to DTCR2)

The 16-bit dead timer control register (DTCR0 to DTCR2) is used to control the operation mode of the waveform generator, the interrupt request enable, the interrupt request flag, the GATE signal enable, and the output level polarity.

■ 16-bit Dead Timer Control Register (DTCR0)

16-bit dead timer control register

Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
DMOD0	GTEN1	GTEN0	TMIF0	TMIE0	TMD2	TMD1	TMD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DTCR0
Address: 0000C4_H
Initial value: 00000000_B

TMD2	TMD1	TMD0	Operation mode bit
0	0	0	Waveform generator stops.
0	0	1	PPG0 timer outputs pulse while RT signal is at "H".
0	1	0	Rising edge of each RT signal becomes trigger, and 16-bit dead timer starts. The PPG timer outputs pulse until the 16-bit dead timer stops. (timer mode)
1	0	0	Non-overlap signal is generated by the RT signal. (dead time timer mode)
1	1	1	Non-overlap signal is generated by PPG0 timers. (dead time timer mode)
Other			Disable

TMIE0	Interrupt request enable bit, software trigger bit
0	Interrupt is not generated even though underflow is generated in 16-bit dead timer.
1	Interrupt is generated when underflow is generated in 16-bit dead timer.

TMIF0	Interrupt request flag bit	
	Read	Write
0	Underflow of counter is not detected.	This bit is cleared.
1	Underflow of counter is detected.	No effect on this bit.

GTEN0	GATE signal control bit 0
0	The GATE signal is not controlled by RT0. (asynchronous mode)
1	The GATE signal is controlled by RT0. (synchronous mode)

GTEN1	GATE signal control bit 1
0	The GATE signal is not controlled by RT1. (asynchronous mode)
1	The GATE signal is controlled by RT1. (synchronous mode)

DMOD0	Output polarity control bit
0	Normal polarity output
1	Reverse polarity output

R/W: Readable/Writable

Initial value

Table 11.4-11 16-bit Dead Timer Control Register (DTCR0)

Bit Name		Function
bit15	DMOD0: Output polarity control bit	<ul style="list-style-type: none"> This bit is used to set the U/V/W output in the dead time timer mode. Setting this bit reverses the U/V/W output polarity. <p>Note: When the dead time timer mode is not selected, (TMD2: bit10 = 0), this bit has no meaning.</p>
bit14	GTEN1: GATE signal control bit 1	This bit is used to control the GATE signal output of the PPG0 timer with RT1.
bit13	GTEN0: GATE signal control bit 0	This bit is used to control the GATE signal output of the PPG0 timer with RT0.
bit12	TMIF0: Interrupt request flag bit	<ul style="list-style-type: none"> This bit is used as an interrupt request flag of the 16-bit dead timer. This bit is set to "1" when the underflow of the 16-bit dead timer occurs. Writing "0" to this bit clears this bit. Writing "1" to this bit has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read. <p>Note: This bit works only when the register values (TMD2 to TMD0: bit10 to bit8) are 000_B or 001_B, and becomes always "0" when they are other values. When a software clear (writing "0") and a hardware set (the underflow of the 16-bit dead timer 0 occurs) occur at the same time, the software clear takes precedence over the hardware set, and this bit is cleared.</p>
bit11	TMIE0: Interrupt request enable bit, software trigger bit	<ul style="list-style-type: none"> This bit is used as a software trigger bit and an interrupt enable bit of the 16-bit dead timer. TMD2 to TMD0: bit10 to bit8 = 000_B or 001_B: This bit is used as a software trigger of the 16-bit dead timer. When this bit changes from "0" to "1", it becomes a trigger of the 16-bit dead timer, reloads the value, and starts the down count. When this bit is "1" and the interrupt request flag bit (TMIF0: bit12) is "1", an interrupt request is sent to the CPU. <p>Note: If the 16-bit dead timer is set as the trigger again, be sure to write "0" to this bit first and then write "1".</p>
bit10 bit9 bit8	TMD2 to TMD0: Operating mode bit	<ul style="list-style-type: none"> These bits are used to select the operation mode of the waveform generator. TMD2 to TMD0: bit10 to bit8 = 000_B: The RT0 and RT1 signals of the output compare are respectively outputted from the RTO0 and RTO1. The 16-bit dead timer is also used as a reload timer. TMD2 to TMD0: When bit10 to bit8 = 001_B, the RT0 and RT1 signals of the output compare are outputted respectively from the RTO0 and RTO1 when the PPG0 output is disabled (the upper of the PPG output control/input capture state control register (PICSH01), PGEN0: bit10 = 0, PGEN1: bit11 = 0). The 16-bit dead timer is also used as a reload timer. <p>Note: To operate the waveform generator in the dead time timer mode, be sure to select the 2 channel mode (the upper of the compare control register (OCSH1), CMOD: bit12 = 1) for RT1. TMD2 to TMD0: bit10 to bit8 = "111_B": The RTO0 and RTO1 outputs are independent of the register settings (the upper of the PPG output control/input capture state control register (PICSH01), PGEN0: bit10 = 0, PGEN1: bit11 = 0).</p>

■ 16-bit Dead Timer Control Register (DTCR1)

16-bit dead timer control register

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DMOD1	GTEN3	GTEN2	TMIF1	TMIE1	TMD5	TMD4	TMD3
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DTCR1

Address: 0000C5_HInitial value: 00000000_B

TMD5	TMD4	TMD3	Operation mode bit
0	0	0	Waveform generator stops.
0	0	1	PPG0 timer outputs pulse while RT signal is at "H".
0	1	0	Rising edge of each RT signal becomes trigger, and 16-bit dead timer starts. The PPG timer outputs pulse until the 16-bit dead timer stops. (timer mode)
1	0	0	Non-overlap signal is generated by the RT signal. (dead time timer mode)
1	1	1	Non-overlap signal is generated by PPG0 timers. (dead time timer mode)
Other			Disable

TMIE1	Interrupt request enable bit, software trigger bit
0	Interrupt is not generated even though underflow is generated in 16-bit dead timer.
1	Interrupt is generated when underflow is generated in 16-bit dead timer.

TMIF1	Interrupt request flag bit	
	Read	Write
0	Underflow of counter is not detected.	This bit is cleared.
1	Underflow of counter is detected.	No effect on this bit.

GTEN2	GATE signal control bit 2
0	The GATE signal is not controlled by RT2. (asynchronous mode)
1	The GATE signal is controlled by RT2. (synchronous mode)

GTEN3	GATE signal control bit 3
0	The GATE signal is not controlled by RT3. (asynchronous mode)
1	The GATE signal is controlled by RT3. (synchronous mode)

DMOD1	Output polarity control bit
0	Normal polarity output
1	Reverse polarity output

R/W: Read/Write

 : Initial value

Table 11.4-12 16-bit Dead Timer Control Register (DTCR1)

Bit Name		Function
bit7	DMOD1: Output polarity control bit	<ul style="list-style-type: none"> This bit is used to set the U/V/W output in the dead time timer mode. Setting this bit reverses the U/V/W output polarity. <p>Note: When the dead time timer mode is not selected, (TMD5: bit2 = 0), this bit has no meaning.</p>
bit6	GTEN3: GATE signal control bit 3	This bit is used to control the GATE signal output of the PPG0 timer with RT3.
bit5	GTEN2: GATE signal control bit 2	This bit is used to control the GATE signal output of the PPG0 timer with RT2.
bit4	TMIF1: Interrupt request flag bit	<ul style="list-style-type: none"> This bit is used as an interrupt request flag of the 16-bit dead timer. This bit is set to "1" when the underflow of the 16-bit dead timer occurs. Writing "0" to this bit clears this bit. Writing "1" to this bit has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read. <p>Note: This bit works only when the register values (TMD5 to TMD3: bit2 to bit0) are 000_B or 001_B, and becomes always "0" when they are other values. When a software clear (writing "0") and a hardware set (the underflow of the 16-bit dead timer occurs) occur at the same time, the software clear takes precedence over the hardware set, and this bit is cleared.</p>
bit3	TMIE1: Interrupt request enable bit, software trigger bit	<ul style="list-style-type: none"> This bit is used as a software trigger bit and an interrupt enable bit of the 16-bit dead timer. TMD5 to TMD3: bit2 to bit0 = 000_B or 001_B: This bit is used as a software trigger of the 16-bit dead timer. When this bit changes from "0" to "1", it becomes a trigger of the 16-bit dead timer, reloads the value, and starts the down count. When this bit is "1" and the interrupt request flag bit (TMIF1: bit4) is "1", an interrupt request is sent to the CPU. <p>Note: If the 16-bit dead timer is set as the trigger again, be sure to write "0" to this bit first and then write "1".</p>
bit2 bit1 bit0	TMD5 to TMD3: Operating mode bit	<ul style="list-style-type: none"> These bits are used to select the operation mode of the waveform generator. TMD5 to TMD3: bit2 to bit0 = 000_B: The RT2 and RT3 signals of the output compare are respectively outputted from the RTO2 and RTO3. The 16-bit dead timer is also used as a reload timer. TMD5 to TMD3: When bit2 to bit0 = 001_B, the RT2 and RT3 signals of the output compare are outputted respectively from the RTO2 and RTO3 when the PPG0 output is disabled (the upper of the PPG output control/input capture state control register (PICSH01), PGEN2: bit12 = 0, PGEN3: bit13 = 0). The 16-bit dead timer is also used as a reload timer. <p>Note: To operate the waveform generator in the dead time timer mode, be sure to select the 2 channel mode (the upper of the compare control register (OCSH3), CMOD: bit12 = 1) for RT3. TMD5 to TMD3: bit2 to bit0 = 111_B: The RTO2 and RTO3 outputs are independent of the register settings (the upper of the PPG output control/input capture state control register (PICSH01), PGEN2: bit12 = 0, PGEN3: bit13 = 0).</p>

■ 16-bit Dead Timer Control Register (DTCR2)

16-bit dead timer control register

Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
DMOD2	GTEN5	GTEN4	TMIF2	TMIE2	TMD8	TMD7	TMD6
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DTCR2

Address: 0000C6_HInitial value: 00000000_B

TMD8	TMD7	TMD6	Operation mode bit
0	0	0	Waveform generator stops.
0	0	1	PPG0 timer outputs pulse while RT signal is at "H".
0	1	0	Rising edge of each RT signal becomes trigger, and 16-bit dead timer starts. The PPG timer outputs pulse until the 16-bit dead timer stops. (timer mode)
1	0	0	Non-overlap signal is generated by the RT signal. (dead time timer mode)
1	1	1	Non-overlap signal is generated by PPG0 timers. (dead time timer mode)
Other			Disable

TMIE2	Interrupt request enable bit, software trigger bit
0	Interrupt is not generated even though underflow is generated in 16-bit dead timer.
1	Interrupt is generated when underflow is generated in 16-bit dead timer.

TMIF2	Interrupt request flag bit	
	Read	Write
0	Underflow of counter is not detected.	This bit is cleared.
1	Underflow of counter is detected.	No effect on this bit.

GTEN4	GATE signal control bit 4
0	The GATE signal is not controlled by RT4. (asynchronous mode)
1	The GATE signal is controlled by RT4. (synchronous mode)

GTEN5	GATE signal control bit 5
0	The GATE signal is not controlled by RT5. (asynchronous mode)
1	The GATE signal is controlled by RT5. (synchronous mode)

DMOD2	Output polarity control bit
0	Normal polarity output
1	Reverse polarity output

R/W: Readable/Writable

 : Initial value

Table 11.4-13 16-bit Dead Timer Control Register (DTCR2)

Bit Name		Function
bit15	DMOD2: Output polarity control bit	<ul style="list-style-type: none"> This bit is used to set the U/V/W output in the dead time timer mode. Setting this bit reverses the U/V/W output polarity. <p>Note: When the dead time timer mode is not selected, (TMD8: bit10 = 0), this bit has no meaning.</p>
bit14	GTEN5: GATE signal control bit 5	This bit is used to control the GATE signal output of the PPG0 timer with RT5.
bit13	GTEN4: GATE signal control bit 4	This bit is used to control the GATE signal output of the PPG0 timer with RT4.
bit12	TMIF2: Interrupt request flag bit	<ul style="list-style-type: none"> This bit is used as an interrupt request flag of the 16-bit dead timer. This bit is set to "1" when the underflow of the 16-bit dead timer occurs. Writing "0" to this bit clears this bit. Writing "1" to this bit has no effect on this bit. When this bit is read to a read modify write instruction, "1" is always read. <p>Note: This bit works only when the register values (TMD8 to TMD6: bit10 to bit8) are 000_B or 001_B, and becomes always "0" when they are other values. When a software clear (writing "0") and a hardware set (the underflow of the 16-bit dead timer 2 occurs) occur at the same time, the software clear takes precedence over the hardware set, and this bit is cleared.</p>
bit11	TMIE2: Interrupt request enable bit, software trigger bit	<ul style="list-style-type: none"> This bit is used as a software trigger bit and an interrupt enable bit of the 16-bit dead timer. TMD8 to TMD6: bit10 to bit8 = 000_B or 001_B: This bit is used as a software trigger of the 16-bit dead timer. When this bit changes from "0" to "1", it becomes a trigger of the 16-bit dead timer, reloads the value, and starts the down count. When this bit is "1" and the interrupt request flag bit (TMIF2: bit12) is "1", an interrupt request is sent to the CPU. <p>Note: If the 16-bit dead timer is set as the trigger again, be sure to write "0" to this bit first and then write "1".</p>
bit10 bit9 bit8	TMD8 to TMD6: Operating mode bit	<ul style="list-style-type: none"> These bits are used to select the operation mode of the waveform generator. TMD8 to TMD6: bit10 to bit8 = 000_B: The RT4 and RT5 signals of the output compare are respectively outputted from the RTO4 and RTO5. The 16-bit dead timer is also used as a reload timer. TMD8 to TMD6: When the bit10 to bit8 = 001_B, the RT4 and RT5 signals of the output compare are outputted respectively from the RTO4 and RTO5 when the PPG0 output is disabled (the upper of the PPG output control/input capture state control register (PICSH01), PGEN4: bit14 = 0, PGEN5: bit15 = 0). The 16-bit dead timer is also used as a reload timer. <p>Note: To operate the waveform generator in the dead time timer mode, be sure to select the 2 channel mode (the upper of the compare control register (OCSH5), CMOD: bit12 = 1) for RT5. TMD8 to TMD6: bit10 to bit8 = 111_B: The RTO4 and RTO5 outputs are independent of the register settings (the upper of the PPG output control/input capture state control register (PICSH01), PGEN4: bit14 = 0, PGEN5: bit15 = 0).</p>

11.4.12 Waveform Control Register (SIGCR1, SIGCR2)

The waveform control register is used to control the operating clock frequency, enable the noise cancel feature, enable DTTI input, and control DTTI interrupts.

■ Waveform Control Register 1 (SIGCR1)

Waveform control register 1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DTIE	DTIF	NRSL	DCK2	DCK1	DCK0	NWS1	NWS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SIGCR1
Address: 0000C9_H
Initial value: 00000000_B

NWS1	NWS0	DTTI noise width selection bit
0	0	Cancel noise of 4cycles
0	1	Cancel noise of 8cycles
1	0	Cancel noise of 16cycles
1	1	Cancel noise of 32cycles

DCK2	DCK1	DCK0	Operation clock selection bit
0	0	0	ϕ (62.5ns, ϕ =16MHz)
0	0	1	$\phi/2$ (125ns, ϕ =16MHz)
0	1	0	$\phi/4$ (250ns, ϕ =16MHz)
0	1	1	$\phi/8$ (500ns, ϕ =16MHz)
1	0	0	$\phi/16$ (1ms, ϕ =16MHz)
1	0	1	$\phi/32$ (2ms, ϕ =16MHz)
1	1	0	$\phi/64$ (4ms, ϕ =16MHz)
1	1	1	Disable

ϕ : Machine cycle

NRSL	Noise cancel function valid bit
0	Noise cancel circuit of DTTI input is invalid.
1	Noise cancel circuit of DTTI input is valid.

DTIF	DTTI interrupt flag bit	
	Read	Write
0	No interrupt request	This bit is cleared.
1	Interrupt request	No effect on this bit.

DTIE	DTTI input valid bit
0	Invalid DTTI input
1	Valid DTTI input

R/W: Readable/Writable

: Initial value

Table 11.4-14 Waveform Control Register 1 (SIGCR1)

Bit Name		Function
bit7	DTIE: DTTI input enable bit	This bit is used to enable the output level control DTTI signal of RTO0 to RTO5 pins.
bit6	DTIF: DTTI interrupt flag bit	<ul style="list-style-type: none"> This bit is the DTTI interrupt flag. When DTTI input is enabled (DTIE: bit7 = 1) and DTTI "L" level is detected, this bit is set, and an interrupt request is sent to the CPU. When this bit is set to "0": Clears the bit. Setting this bit to "1" has no effect on this bit. "1" is always read during read-modify-write. <p>Note: When the noise cancel feature is enabled (NRSL: bit5 = 1), and noise pulse width is passed, this bit is set to "1". If a software clear (write "0") and hardware set (DTTI Low level detected) occur simultaneously, the software clear is given precedence over the hardware set, and this bit is cleared.</p>
bit5	NRSL: Noise cancel function enable bit	<ul style="list-style-type: none"> This bit is used to enable the noise cancel feature. If a "L" level is maintained until a counter overflow occurs, the noise cancel circuit receives a DTTI input signal. The counter is a n-bit counter operated by "L" level input. n is set by NWS1, NWS0 bit: 1, 0; Based on the settings of bit1 and bit0, the value of n is 2, 3, 4, or 5. <p>Note: Approximately 2ⁿ machine cycles are required to cancel the noise pulse width. When a noise cancel circuit is selected, input is disabled when in a mode that stops the internal clock (e.g. stop mode).</p>
bit4 bit3 bit2	DCK2 to DCK0: Operation clock selection bits	These bits are used to select the operating clock of the 16-bit dead timer.
bit1 bit0	NWS1, NWS0: DTTI noise width selection bits	These bits are used to select the DTTI pin noise pulse width to cancel.

■ Waveform Control Register 2 (SIGCR2)

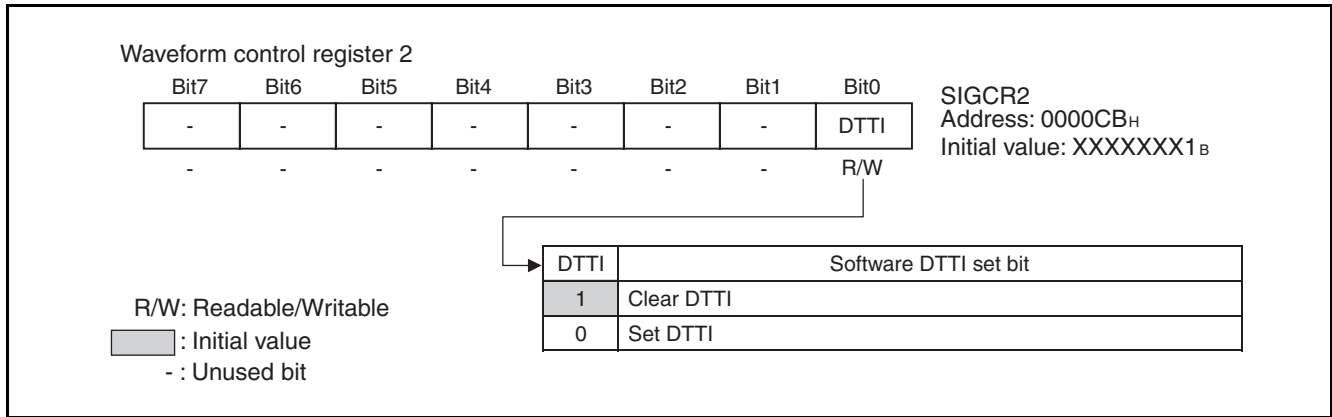


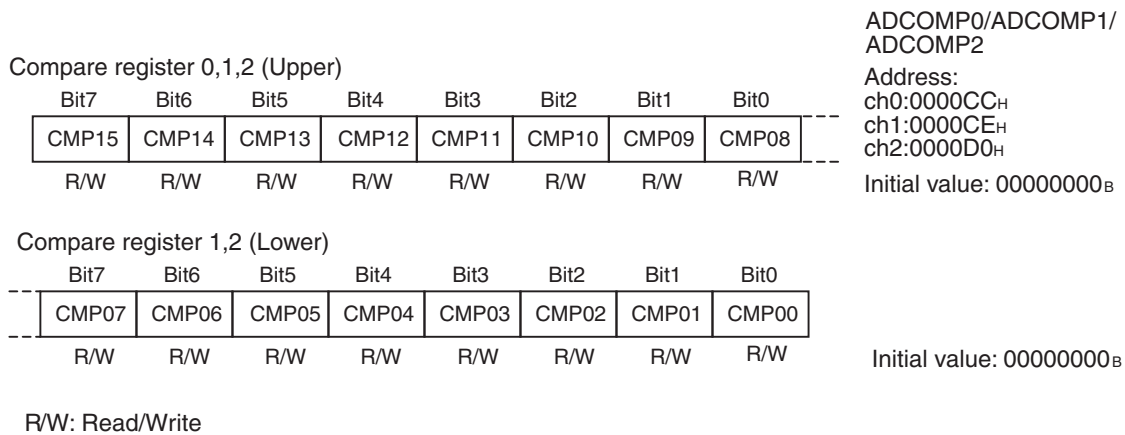
Table 11.4-15 Waveform Control Register2 (SIGCR2)

Bit Name		Function
bit7 to bit1	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect on operation.
bit0	DTTI: Software DTTI setting bit	<ul style="list-style-type: none"> Write "0" to set DTTI. Write "1" to clear bit. <p>Note: As this is OR with external input DTTI, however, DTTI depends on the external input level.</p>

11.4.13 A/D Activation Compare Register (ADCOMP0, ADCOMP1, ADCOMP2, ADCOMPC)

Compare registers 0, 1, and 2 activate A/D converters 0, 1, and 2 when their values match that of the free-run timer. The compare register is used to write compare values. The control register enables or disables the activation request to the A/D converter when a compare match occurs.

■ Compare Register 0, 1, 2 (ADCOMP0, ADCOMP1, ADCOMP2)



The compare register is used to write data for comparison with the 16-bit free-run timer count value. It is possible to activate A/D when the free-run timer and compare values match.

The value written to the compare register is used for a comparison immediately.

Write to the compare register in word or half-word units.

■ Control Register (ADCOMPC)

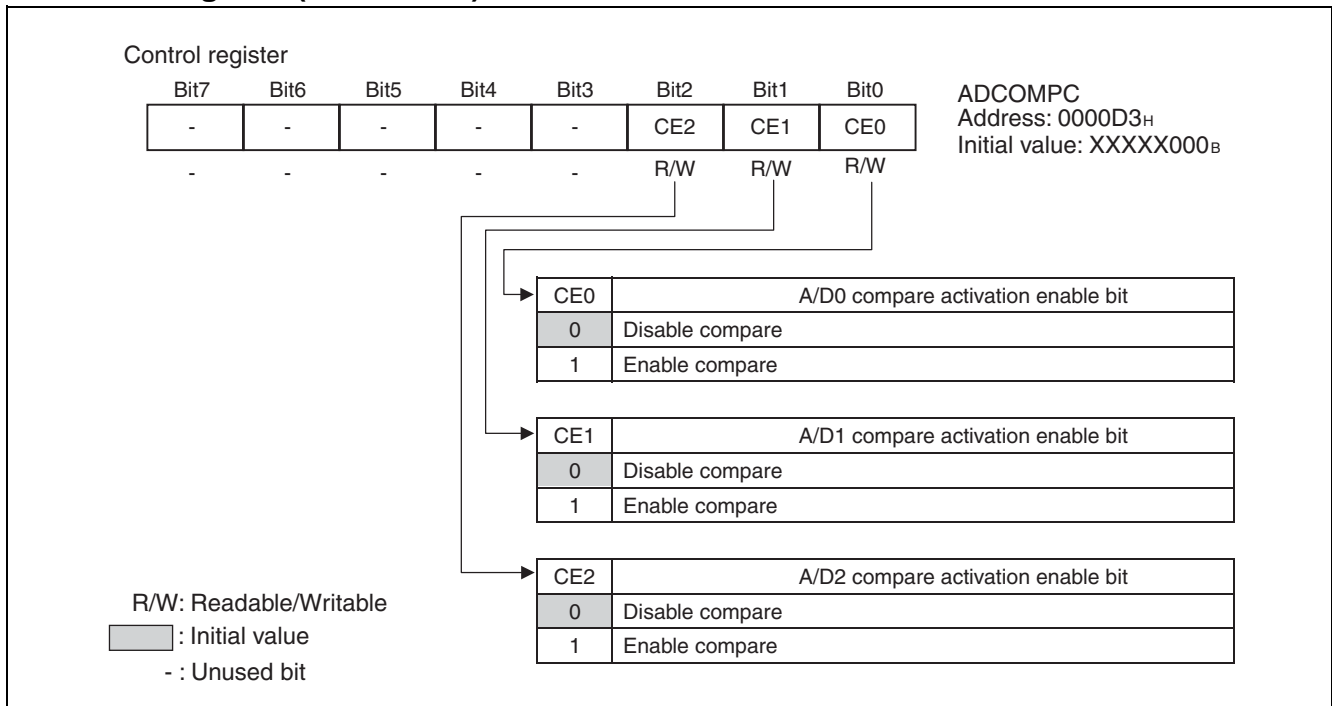


Table 11.4-16 Control Register (ADCOMPC)

Bit Name		Function
bit7 to bit3	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect on operation.
bit2	CE2: A/D2 compare activation enable bit	<ul style="list-style-type: none"> Write "1" to this bit to output a activation request for A/D unit 2 when there is a compare match. Write "0" to this bit to disable compare operation.
bit1	CE1: A/D1 compare activation enable bit	<ul style="list-style-type: none"> Write "1" to this bit to output a activation request for A/D unit 1 when there is a compare match. Write "0" to this bit to disable compare operation.
bit0	CE0: A/D0 compare activation enable bit	<ul style="list-style-type: none"> Write "1" to this bit to output a activation request for A/D unit 0 when there is a compare match. Write "0" to this bit to disable compare operation.

11.5 Multifunctional Timer Interrupt

The multifunctional timer can generate 16-bit free-run timer interrupts, 16-bit output compare interrupts, 16-bit input capture interrupts, and waveform generator interrupts.

■ 16-bit Free-run Timer Interrupt

See Table 11.5-1 for 16-bit free-run timer interrupt control bits and interrupt causes.

Table 11.5-1 16-bit Free-run Timer Interrupt Control Bits and Interrupt Causes

	16-bit Free-run Timer	
	Compare Clear	Zero Detection
Interrupt request flag bit	ICLR of upper timer status register (TCCSH): bit9	IRQZF of upper timer status register (TCCSH): bit14
Interrupt request enable bit	ICRE of upper timer status register (TCCSH): bit8	IRQZE of upper timer status register (TCCSH): bit13
Interrupt cause	The 16-bit free-run timer value and compare-clear register (CPCLRH, CPCLRL) match.	16-bit free-run timer value is set to "0".

When the 16-bit free-run timer value and compare-clear register (CPCLRH/CPCLRL) match, the timer status register (TCCSH) ICLR: bit9 is set to "1". When interrupt requests are enabled in this state (TCCSH register ICRE: bit8 = 1), interrupt requests are outputted to the interrupt controller.

When the timer value is "0000_H", the timer status control register (TCCSH) IRQZF: bit14 is set to "1".

When interrupt requests are enabled in this state (TCCSH register IRQZE: bit13 = 1), interrupt requests are outputted to the interrupt controller.

■ 16-bit Output Compare Interrupt

See Table 11.5-2 for 16-bit output compare interrupt control bits and interrupt causes.

Table 11.5-2 16-bit Output Compare 0 to 5 Interrupt Control Bits and Interrupt Causes

	16-bit Output Compare 0, 1	16-bit Output Compare 2, 3	16-bit Output Compare 4, 5
Interrupt request flag bit	IOP1, IOP0 (bit7, bit6) of lower compare control register (OCSL0)	IOP1, IOP0 (bit7, bit6) of lower compare control register (OCSL2)	IOP1, IOP0 (bit7, bit6) of lower compare control register (OCSL4)
Interrupt request enable bit	IOE1, IOE0 (bit5, bit4) of lower compare control register (OCSL0)	IOE1, IOE0 (bit5, bit4) of lower compare control register (OCSL2)	IOE1, IOE0 (bit5, bit4) of lower compare control register (OCSL4)
Interrupt cause	The 16-bit free-run timer value and output compare register (OCCPH0, OCCPH1, OCCPL0, OCCPL1) match.	The 16-bit free-run timer value and output compare register (OCCPH2, OCCPH3, OCCPL2, OCCPL3) match.	The 16-bit free-run timer value and output compare register (OCCPH4, OCCPH5, OCCPL4, OCCPL5) match.

When the 16-bit free-run timer value and output compare register (OCCPH0 to OCCPH5, OCCPL0 to OCCPL5) match, the compare control register low-order (OCSL0, OCSL2, and OCSL4) IOP 1 and 0: bit7 and bit6 are set to "1". When interrupt requests are enabled in this state (OCSL0, OCSL2, and OCSL4 register IOE1 and IOE0: bit5, bit4 = 11_B), interrupt requests are outputted to the interrupt controller.

■ 16-bit Input Capture Interrupt

See Table 11.5-3 for 16-bit input capture interrupt control bits and interrupt causes.

Table 11.5-3 16-bit Input Capture 0 to 3 Interrupt Control Bits and Interrupt Causes

	16 Bit Input Capture0, 1	16 Bit Input Capture2, 3
Interrupt request flag bit	Input capture status control register, lower (PICSL01): ICP1, ICP0 (bit7, bit6)	Input capture status control register, lower (ICSL23): ICP3, ICP2 (bit7, bit6)
Interrupt request enable bit	Input capture status control register, lower (PICSL01): ICE1, ICE0 (bit5, bit4)	Input capture status control register, lower (ICSL23): ICE3, ICE2 (bit5, bit4)
Interrupt cause	Valid edges are detected by IC0, IC1 pins.	Valid edges are detected by IC2, IC3 pins.

With 16-bit input capture, when a valid edge is detected by IC0, IC1, IC2, IC3 pins, the input capture-status control registers (PICSL01 and ICSL23) ICP3, ICP2, ICP1, and ICP0: bit7 and bit6 are both set to 11_B. When interrupt requests are enabled in this state (PICSL01 and ICSL23 registers ICE3, ICE2, ICE1, ICE0: bit5 and bit4 are both 11_B), interrupt requests are outputted to the interrupt controller.

■ Waveform Generator Interrupts

See Table 11.5-4 for waveform generator interrupt control bits and interrupt causes.

Table 11.5-4 Waveform Generator Interrupt Control Bits and Interrupt Causes

	Waveform Generator	
	16-bit Dead Timer 0, 1, 2	DTTI0
Interrupt request flag bit	TMIF0 to TMIF2 (upper order is bit12, lower order is bit4) of the 16-bit dead timer control register higher order and lower order (DTCR0, DTCR1, DTCR2)	DTIF (bit6) of waveform control register 1 (SIGCR1)
Interrupt request enable bit	TMIE0 to TMIE2 (upper order is bit11, lower order is bit3) of the 16-bit dead timer control register higher order and lower order (DTCR0, DTCR1, DTCR2)	—
Interrupt cause	16-bit dead timer 0, 1, and 2 underflow	"L" level detected in DTTI.

In the waveform generator, when a 16-bit dead timer underflow occurs, and TMD8 to TMD0 of the DTCR 0, DTCR1, and DTCR2 register (higher-order bits are 10 to 8, and lower-order bits are 2 to 0) are "000_B" or "001_B", TMIF0 to TMIF2 (higher-order bit is 12, and lower-order bit is 4) of the 16-bit dead timer control register (DTCR0, DTCR1, and DTCR2) are set to "1". When interrupt requests are enabled in this state (DTCR0, DTCR1, and DTCR2 register TMIE0 to TMIE2 (upper-order bit is 11, lower-order bit is 3) = 1), interrupt requests are outputted to the interrupt controller.

11.6 Operation of Multifunctional Timer

The operation of the multifunctional timer is described below.

■ Operation of Multifunctional Timer

- 16-bit free-run timer

When the 16-bit free-run timer enables count operation, the counter begins counting up from the value set in the timer data register (TCDTH, TCDTL). The count value is used as the standard time of the 16-bit output compare and 16-bit input capture.

- 16-bit output compare

16-bit output compare is used to compare the value set in the output compare register with the 16-bit free-run timer value. If a match is detected, the interrupt flag is set, and the output level is reversed.

- 16-bit input capture

16-bit input capture is used to detect specified valid edges.

When a valid edge is detected, the interrupt flag is set, and the value of the 16-bit free-run timer is retrieved and stored in the input capture data register.

- Waveform generator

The waveform generator generates a variety of waveforms (including dead times) using the real-time output (RTO 0 to RTO5), the 16-bit PPG timer 0, and the 16-bit dead timer.

- A/D activation compare

When the 16-bit free-run timer reaches the specified value, A/D is activated.

11.6.1 Operation of 16-bit Free-run Timer

After 16-bit free-run timer reset is completed, the counter begins counting up from the value set in the timer data register (TCDTH/TCDTL). The count value is used as the standard time of the 16-bit output compare and 16-bit input capture.

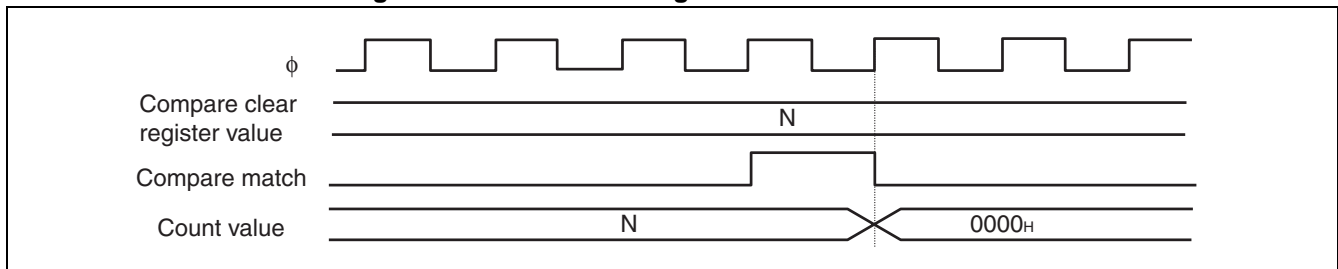
■ Timer Clear

The count value of the 16-bit free-run timer is cleared when one of the following:

- When a match with the compare-clear register is detected via up count mode (TCCSL register MODE: bit5 = 0)
- When "1" is written to the TCCSL register's SCLR: bit4 during operation
- When "0000_H" is written to the TCDTH/TCDTL register while stopped
- Resetting

After a reset, the counter is immediately cleared. When there is a match with the compare-clear register, the counter is cleared in synchronization with the count timing.

Figure 11.6-1 Clear Timing of 16-bit Free-run Timer



■ Timer Mode

The following modes can be selected for the 16-bit free-run timer.

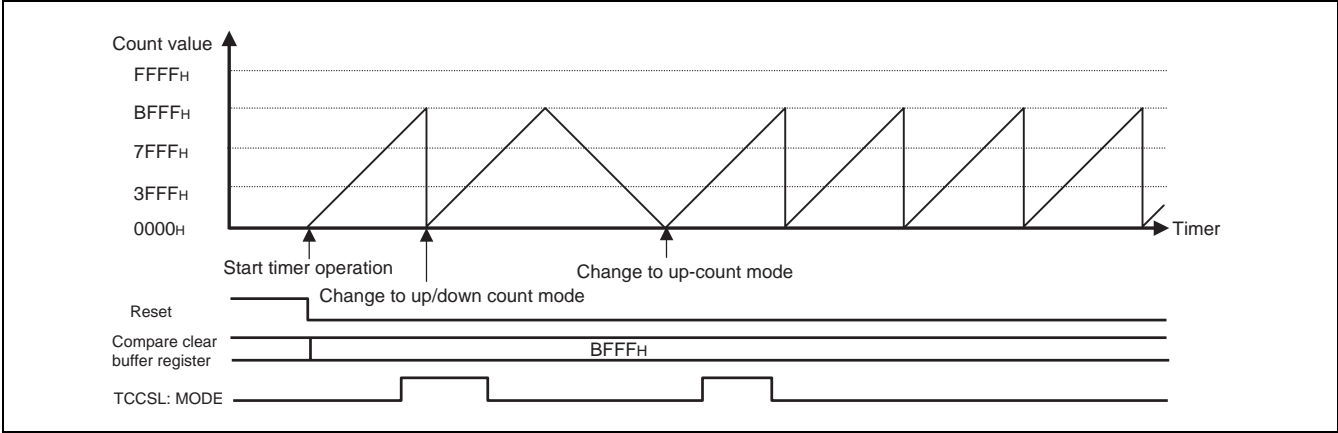
- Up count mode (TCCSL register MODE: bit5 = 0)
- Up/down count mode (TCCSL register MODE: bit5 = 1)

When in up count mode, the counter starts counting from a value previously set in the timer data register (TCDTH/TCDTL), and counts up until the count value matches the value of the compare-clear register (CPCLR_H/CPCLRL). The counter is then cleared to "0000_H", and starts counting up again.

When in up/down count mode, the counter starts counting from a value previously set in the timer data register (TCDTH/TCDTL), and counts up until the count value matches the value of the compare-clear register (CPCLR_H/CPCLRL). The counter then starts counting from up to down, until it reaches the value "0000_H", after which it again starts counting up.

It is possible to write a value to the mode bit (TCCSL register's MODE: bit5) at any time, whether the timer is operating or stopped. Values written to this bit while the timer is operating go into the buffer, and the count mode changes when the timer value reaches "0000_H".

Figure 11.6-2 Change Timer Mode during Timer Operation



■ Compare Clear Buffer

The compare-clear register (CPCLR_H/CPCLR_L) has a buffer feature that can be enabled or disabled. If the buffer feature is enabled (TCCSL register's BFE: bit7 = 1), data written to the compare-clear buffer register (CPCLR_{BH}/CPCLR_{BL}) is sent to the CPCLR_H/CPCLR_L register when a value of 0 is detected in the 16-bit free-run timer. When the buffer feature is disabled (TCCSL bits' BFE: bit7 = 0), the CPCLR_{BH}/CPCLR_{BL} register becomes permeable, and data can be written directly to the CPCLR_H/CPCLR_L register.

Figure 11.6-3 Operation in Up Count Mode when Compare Clear Buffer Is Disabled (TCCSL Register's BFE: Bit7 = 0)

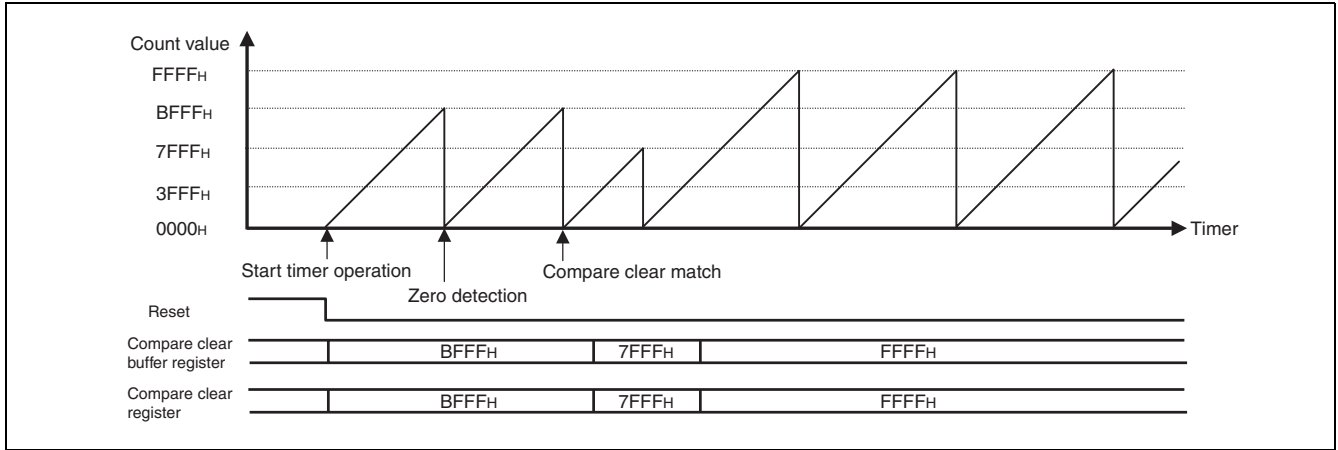


Figure 11.6-4 Operation in Up Count Mode when Compare Clear Buffer Is Enabled (TCCSL Register's BFE: Bit7 = 1)

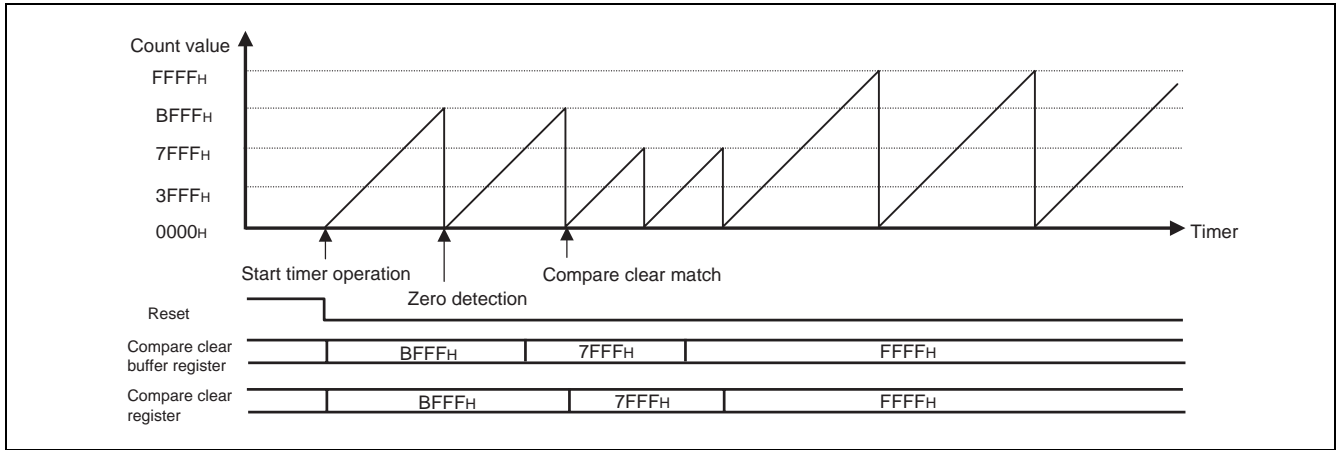
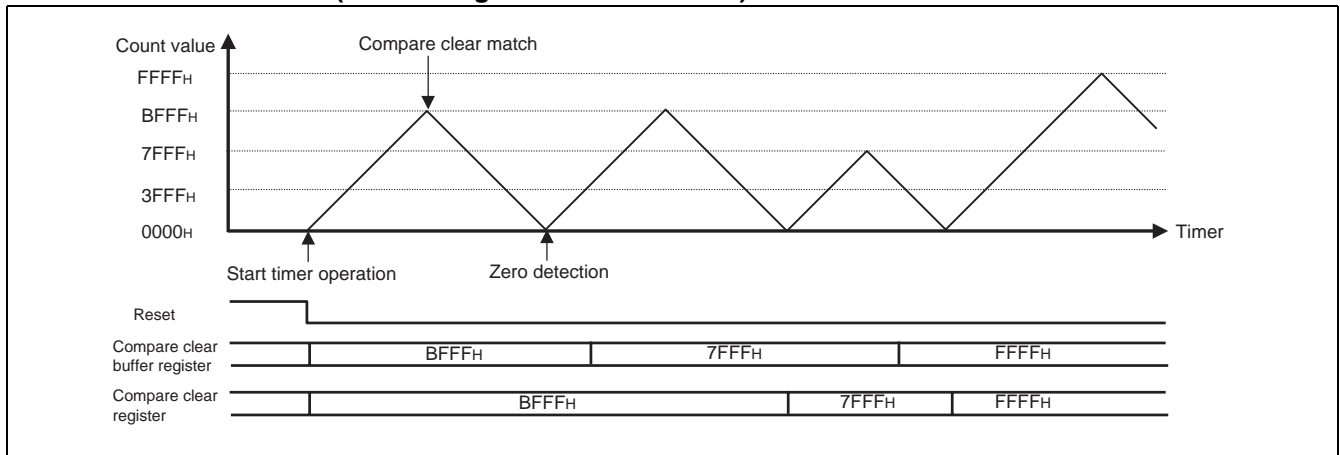


Figure 11.6-5 Operation in Up/Down Count Mode when Compare Clear Buffer Is Enabled (TCCSL Register's BFE: Bit7 = 1)



■ **Timer Interrupt**

The 16-bit free-run timer can generate the following two interrupts.

- Compare clear interrupt
- Zero detection interrupt

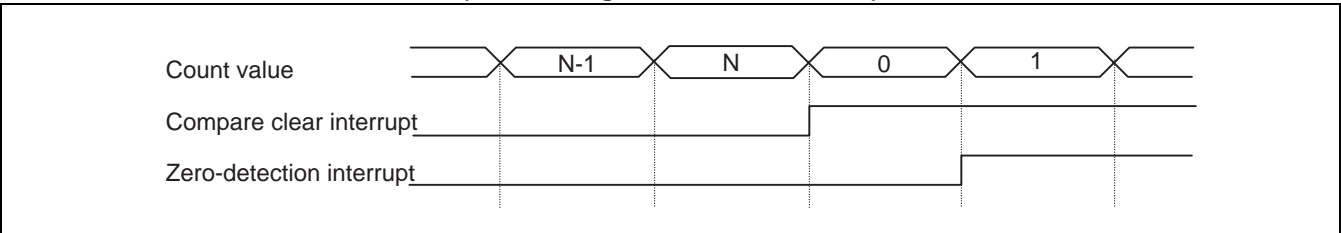
Compare-clear interrupts are generated when the timer value matches the value of the compare-clear register (CPCLRH, CPCLRL).

Zero-detection interrupts are generated when the timer value reaches "0000_H".

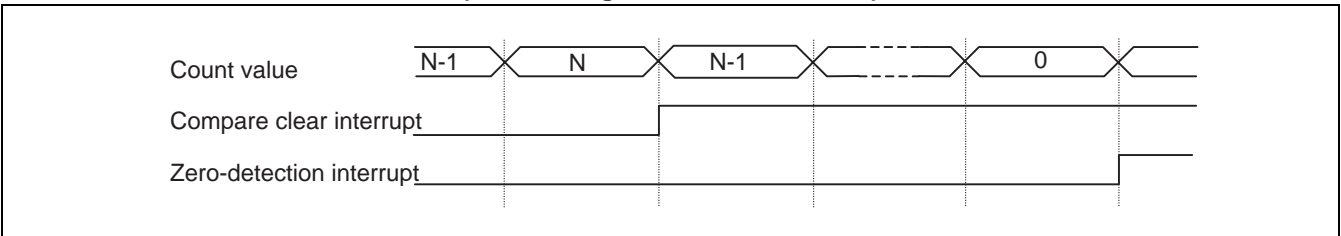
Notes:

- When the timer count clock is the machine cycle (ϕ)
Even if a software clear (TCCSL: bit4 SCLR = 1) is performed, the zero-detection interrupt flag is not set, and a zero-detection interrupt is not generated.
- When the timer count clock is the machine cycle (ϕ) division
When a software clear (TCCSL: bit4 SCLR = 1) is performed, and the zero-detection interrupt flag is set and interrupts are enabled, then a zero-detection interrupt is generated.

**Figure 11.6-6 Interrupt Generated in Up Count Mode
(TCCSL Register MODE: Bit5 = 0)**



**Figure 11.6-7 Interrupt Generated in Up/Down Count Mode
(TCCSL Register MODE: Bit5 = 1)**



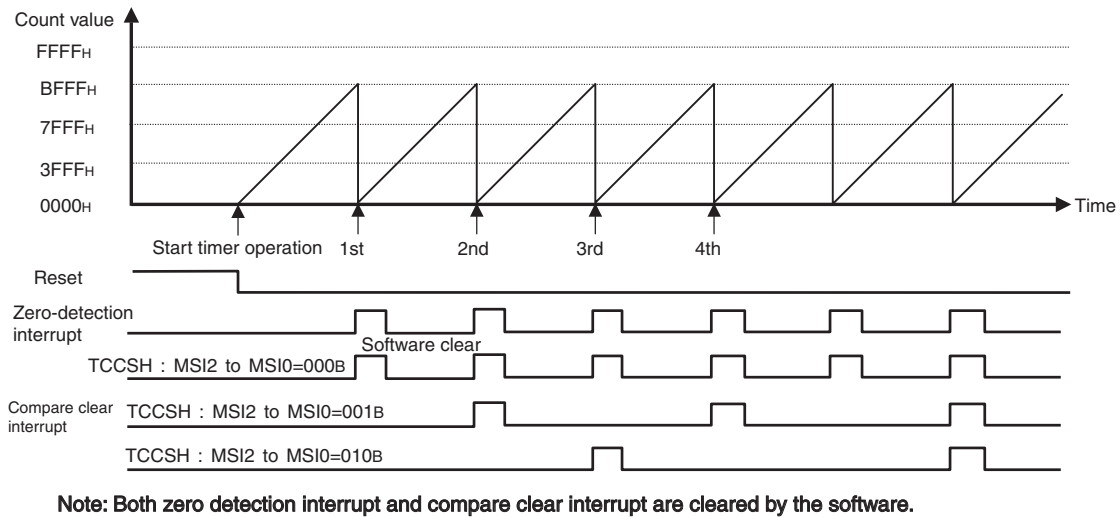
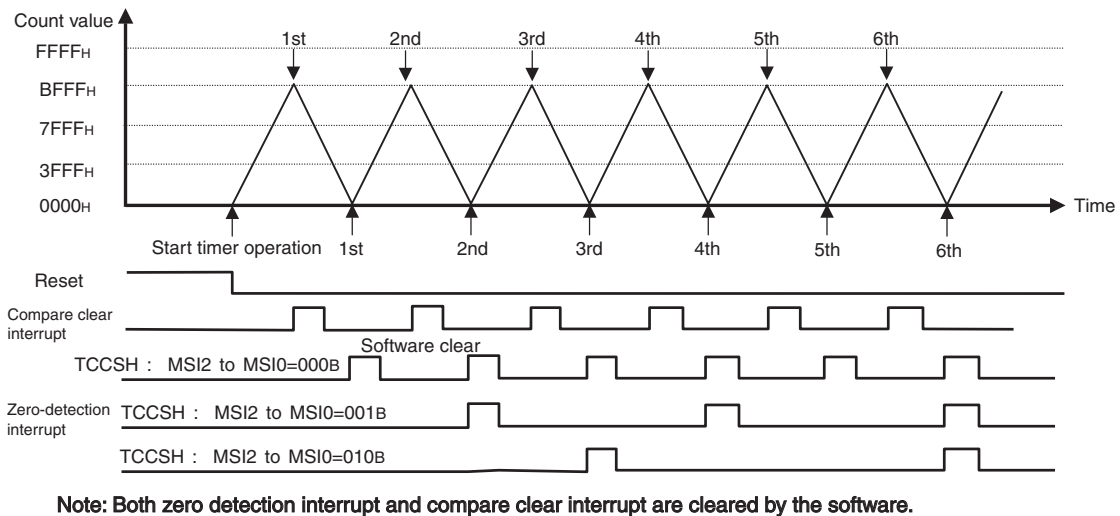
■ **Interrupt Mask Function**

It is possible to mask interrupt requests by setting the TCCSH register MSI2 to MSI0: bit12 to bit10. MSI2 to MSI0 bits are a 3-bit reload down counter that reload when the count value reaches 000_B. The counter value can be reloaded by writing directly to MSI2 to MSI0 bits. The mask count is the value set in MSI2 to MSI0 bits. When MSI2 to MSI0 bits are 000_B, interrupt causes are not masked.

The interrupt cause depends on the count mode (TCCSL register MODE: bit5). In up count mode, it is only possible to mask compare-clear interrupts, and zero-detection interrupts are generated each time "0" is detected. In up/down count mode, it is only possible to mask zero-detection interrupts, and compare-clear interrupts are generated each time a compare clear is detected.

Notes:

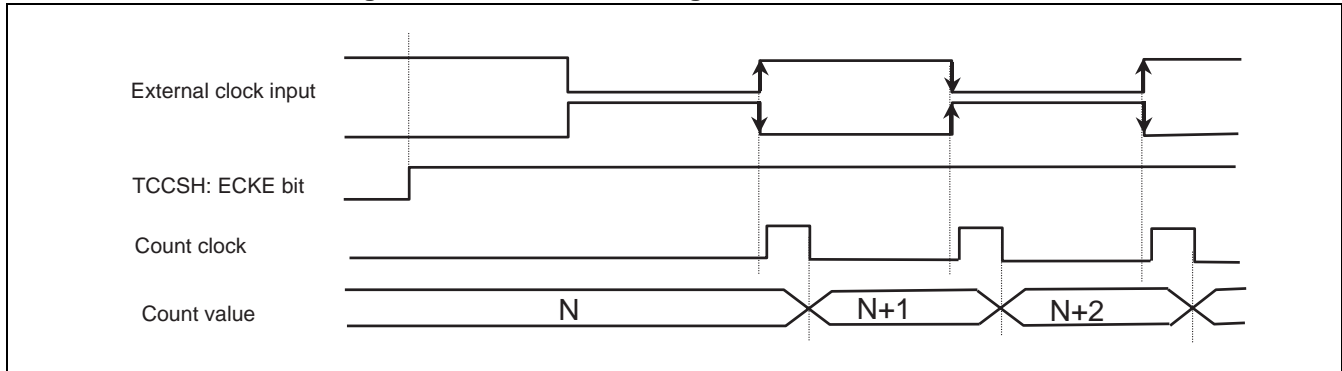
- When the timer count clock is the machine cycle (ϕ)
Even if a software clear (TCCSL: bit4 SCLR = 1) is performed, the zero-detection interrupt flag is not set, and a zero-detection interrupt is not generated.
- When the timer count clock is the machine cycle (ϕ) division
When a software clear (TCCSL: bit4 SCLR = 1) is performed, and the zero-detection interrupt flag is set and interrupts are enabled, then a zero-detection interrupt is generated.

Figure 11.6-8 Compare Clear Interrupt Masked in Up Count Mode**Figure 11.6-9 Zero Detection Interrupt Masked in Up/Down Count Mode**

■ Selected External Count Clock

The 16-bit free-run timer is incremented based on the input clock (internal clock or external clock). When an external clock is selected, then after external clock mode is selected (TCCSH register's ECKE: bit15=1), the 16-bit free-run timer starts counting up on a rising edge if the initial value of external input is "1". Subsequently, it counts up on both edges. If the initial value of external input is "0", it starts counting up on a falling edge. Subsequently, it counts up on both edges.

Figure 11.6-10 Count Timing of 16-bit Free-run Timer



■ A/D Activation Via Free-run Timer

It is possible to activate A/D1 and A/D2 upon a compare match or 0 detection of the 16-bit free-run timer. The activation trigger can be selected by means of the A/D trigger cause selection bit (SEL1 and SEL2: bit2 and bit3) of the A/D trigger control register (ADTRGC).

It is possible to halt A/D activation signals, even upon compare match or 0 detection, via the A/D trigger output enable/disable bits (AD1E and AD2E: bit0 and bit1) of the A/D trigger control register (ADTRGC).

Note:

When A/D activation signal output is disabled, if A/D activation signal output is enabled, then when an activation trigger compare match or 0-detection is outputted, the A/D activation signal will be outputted at the same time that the authorization is made.

11.6.2 Operation of 16-bit Output Compare

Output compare is used to compare the value set in the compare clear register with the 16-bit free-run timer value. If a match is detected, the interrupt flag is set, and the output level is reversed.

Match signals are ignored by the free-run timer when compare register value matches with the count peak value of the free-run timer.

■ Operation of 16-bit Output Compare (Inverted Mode, MOD1x=0)

- Compare operation can be performed on each channel (compare control register higher-order (OCSH1, OCSH3, and OCSH5) CMOD: bit12 = 0).

Figure 11.6-11 Example of Output Waveform if Compare Registers 0 and 1 are Used Separately when the Initial Value of Output Is "0" (Free-run Timer Is Up Count Mode)

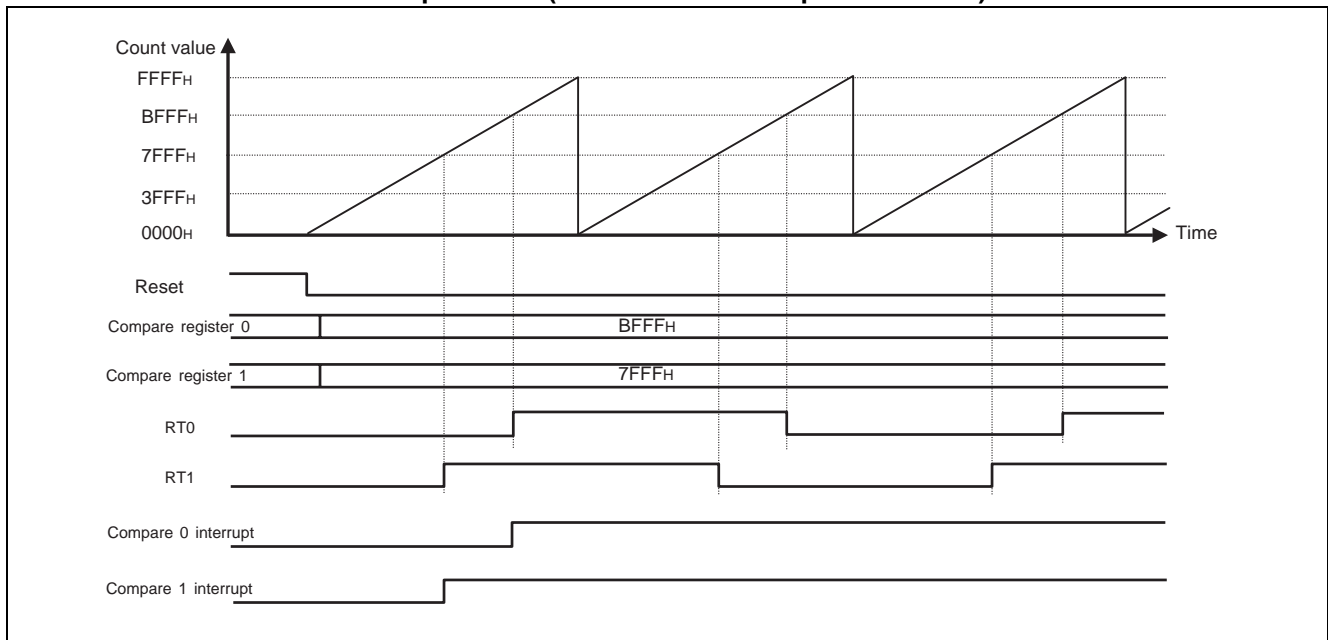
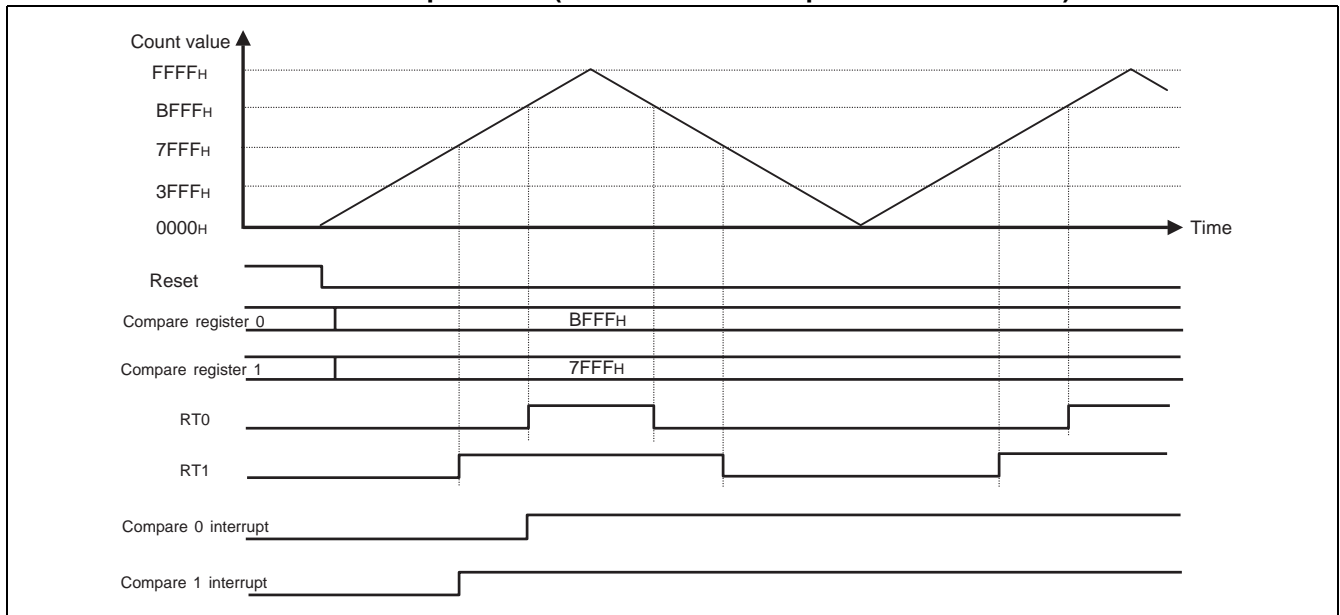


Figure 11.6-12 Example of Output Waveform if Compare Registers 0 and 1 are Used Separately when the Initial Value of Output Is "0" (Free-run Timer Is Up/Down Count Mode)



- The output level can use a single compare register (compare control register higher-order (OCSH 1, OCSH3, and OCSH5) CMOD: bit12 = 1).

Figure 11.6-13 Example of Output Waveform if Compare Registers 0 and 1 are Used in Pairs when the Initial Value of Output Is "0" (Free-run Timer Is Up Count Mode)

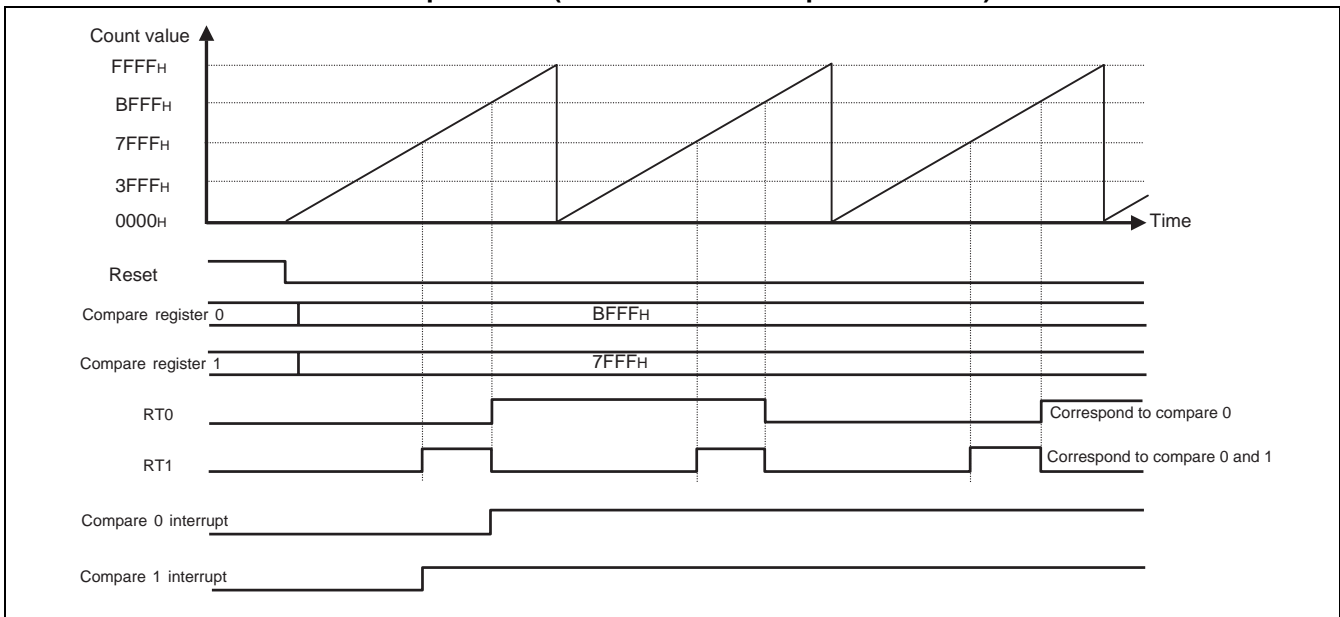
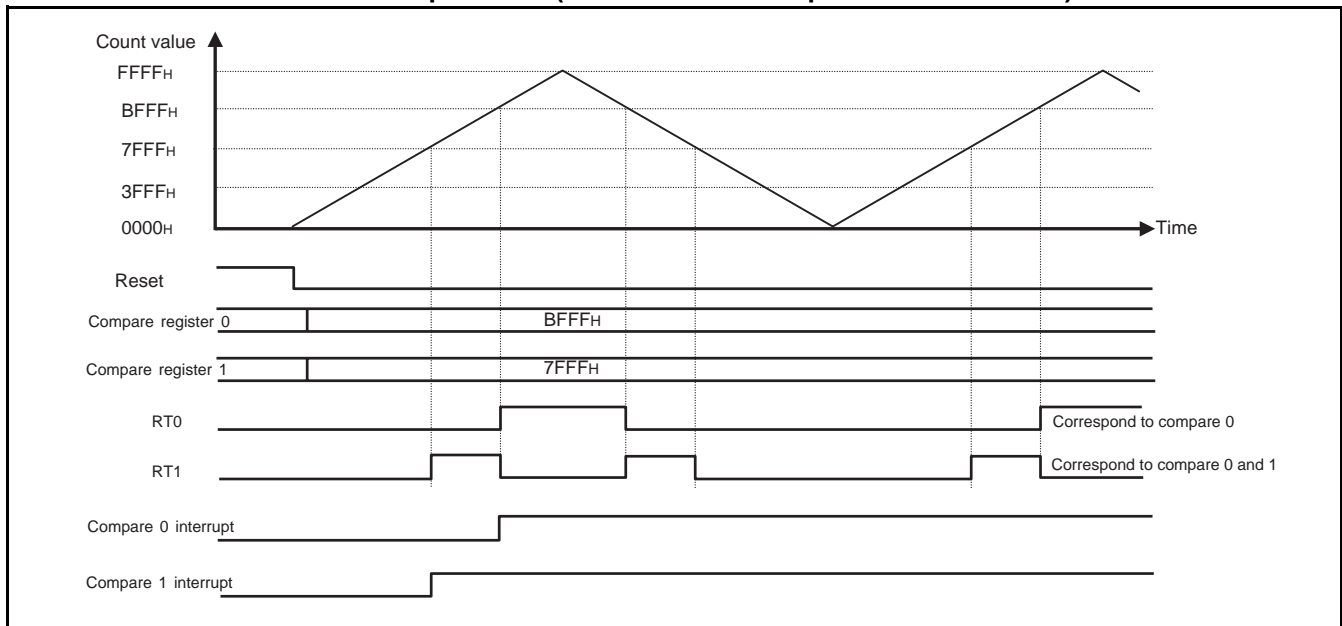
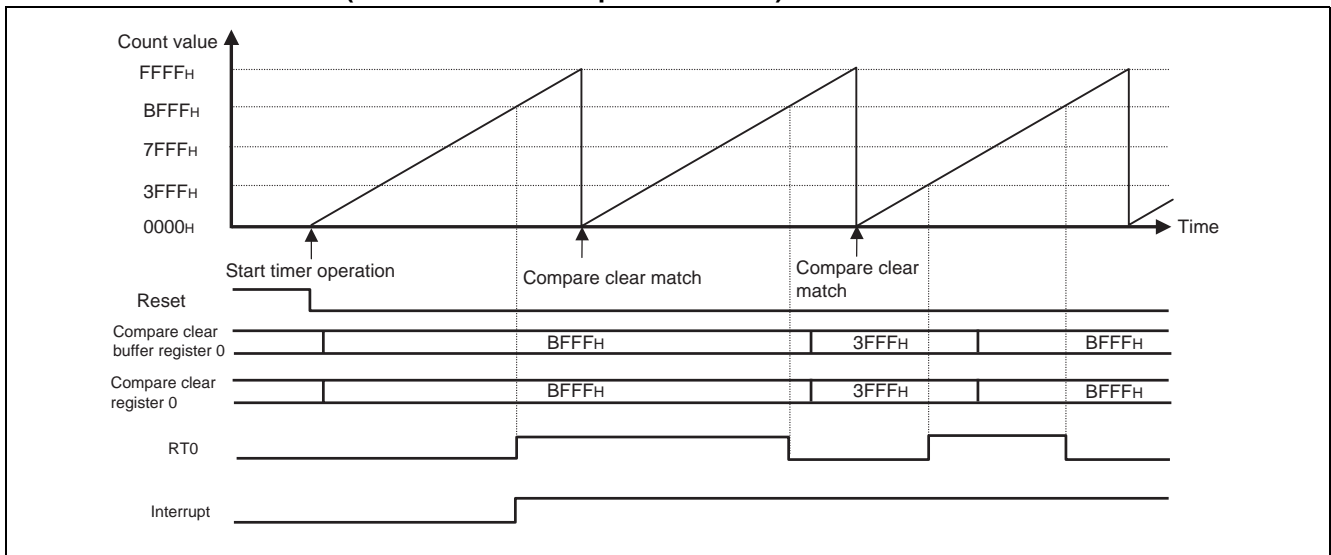


Figure 11.6-14 Example of Output Waveform if Compare Registers 0 and 1 are Used Together when the Initial Value of Output Is "0" (Free-run Timer Is Up/Down Count Mode)



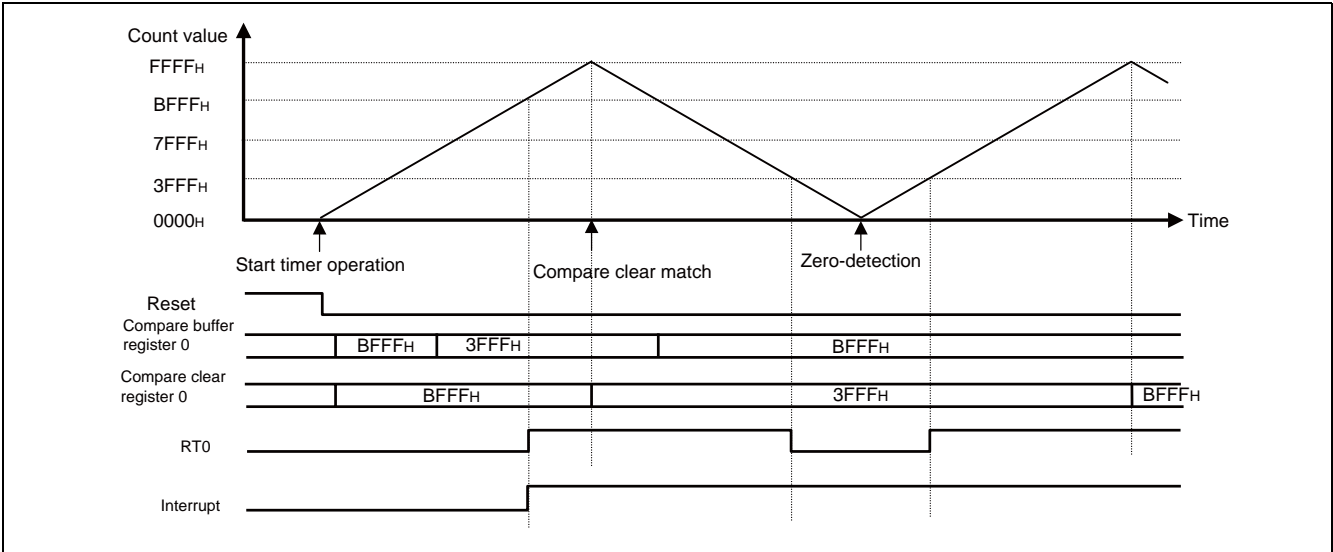
- Output level when compare buffer is disabled

Figure 11.6-15 Example of Output Waveform when Compare Buffer Is Disabled (Free-run Timer is Up Count Mode)

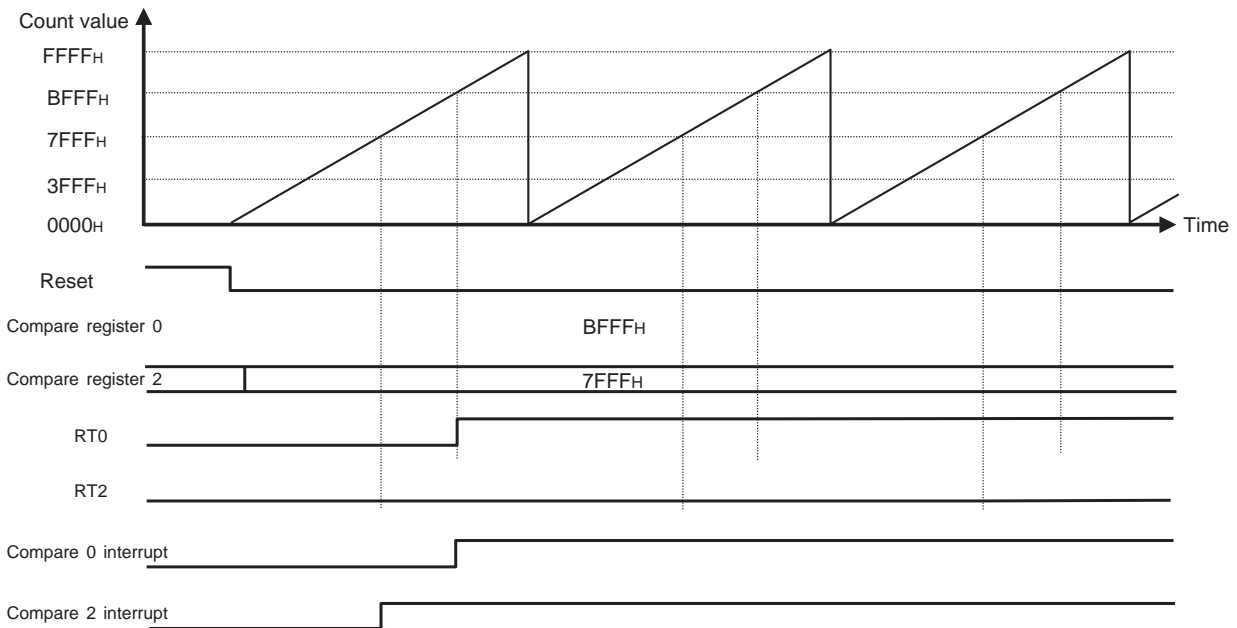


- Output level when compare buffer is selected, and a compare clear match occurs:

Figure 11.6-16 Example of Output Waveform when Compare Buffer Is Enabled (Free-run Timer is Up/Down Count Mode)



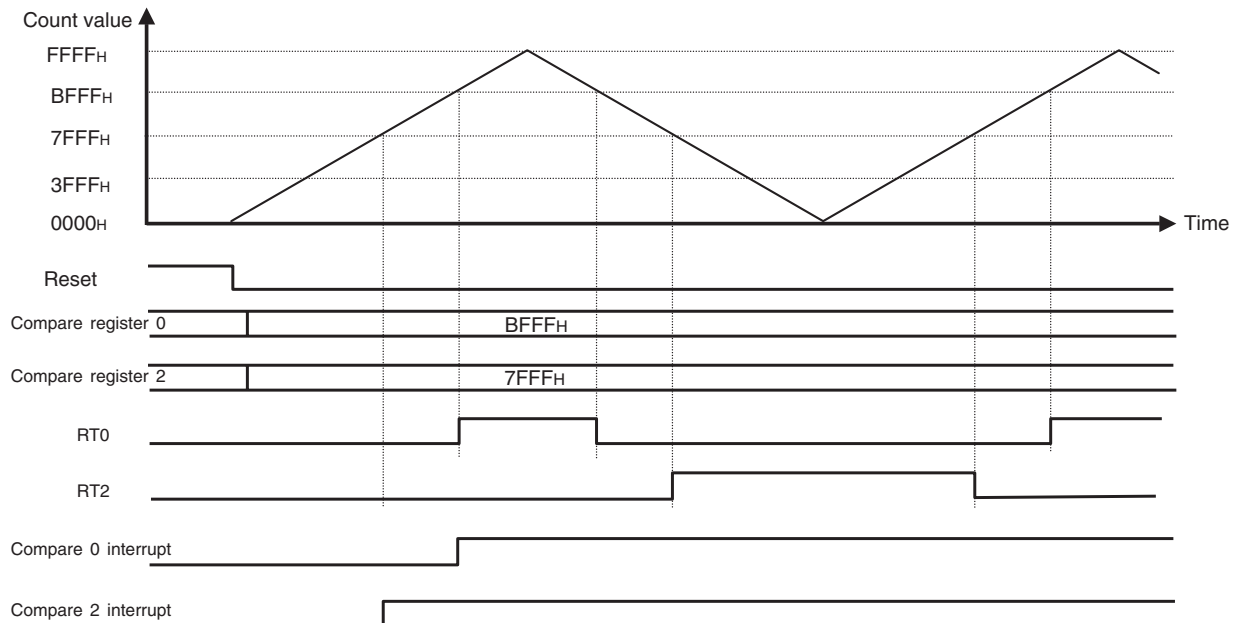
■ Operation of 16-bit Output Compare (Set/Reset Mode, MOD15 to MOD10=1)



ch0: Up count: set, Down count: reset

ch2: Up count: reset, Down count: set

Note: ch0 keep "1" if ch0 match occur. ch2 is always "0".



ch0: Up count: set, Down count: reset

ch2: Up count: reset, Down count: set

■ 16-bit Output Compare Timing

When the free-run timer matches the compare register value, output compare generates a compare match signal and reverses output, then generates an interrupt. When a compare match occurs, output is reversed in synchronization with the count timing of the counter.

Figure 11.6-17 Compare Register Interrupt Timing

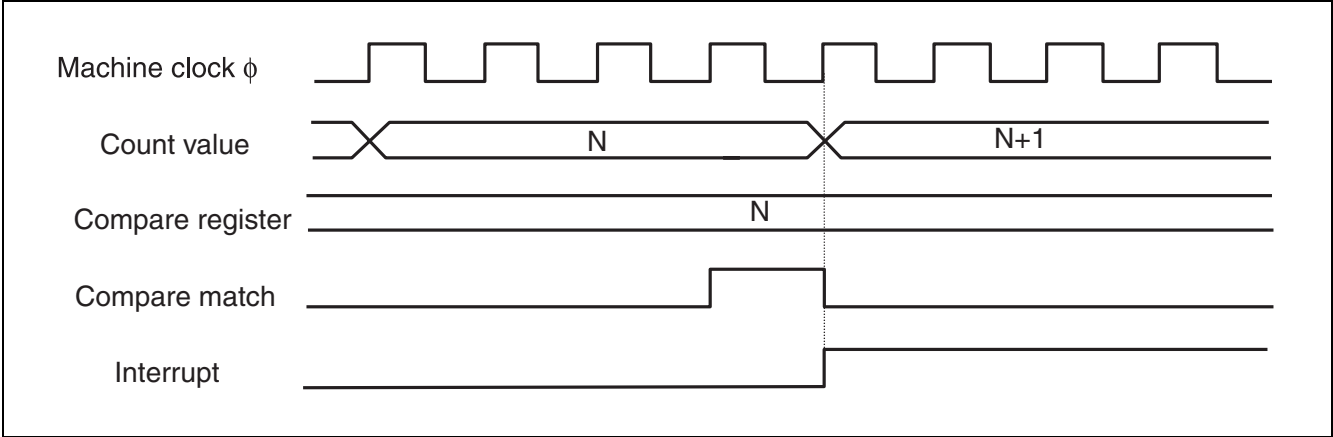
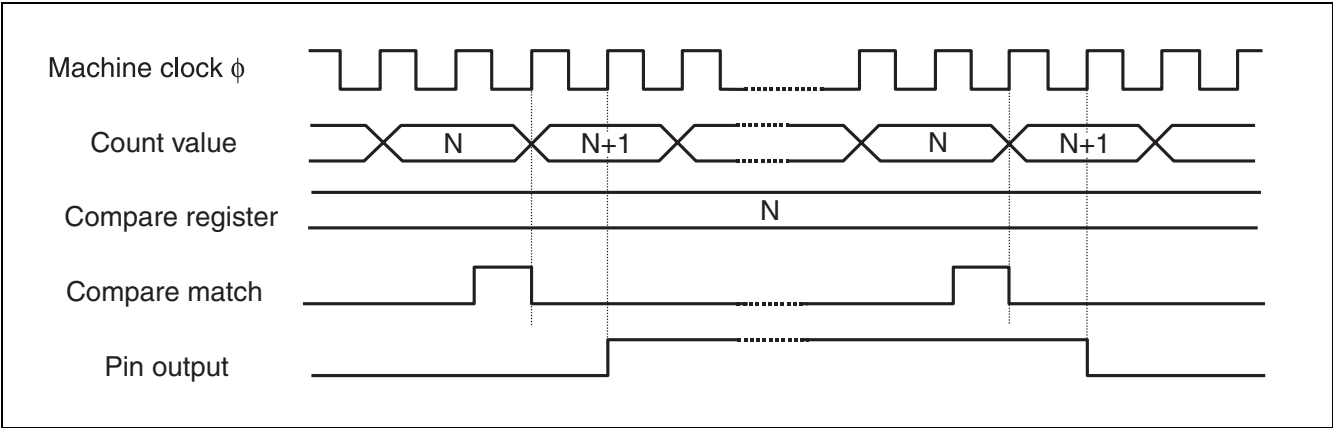
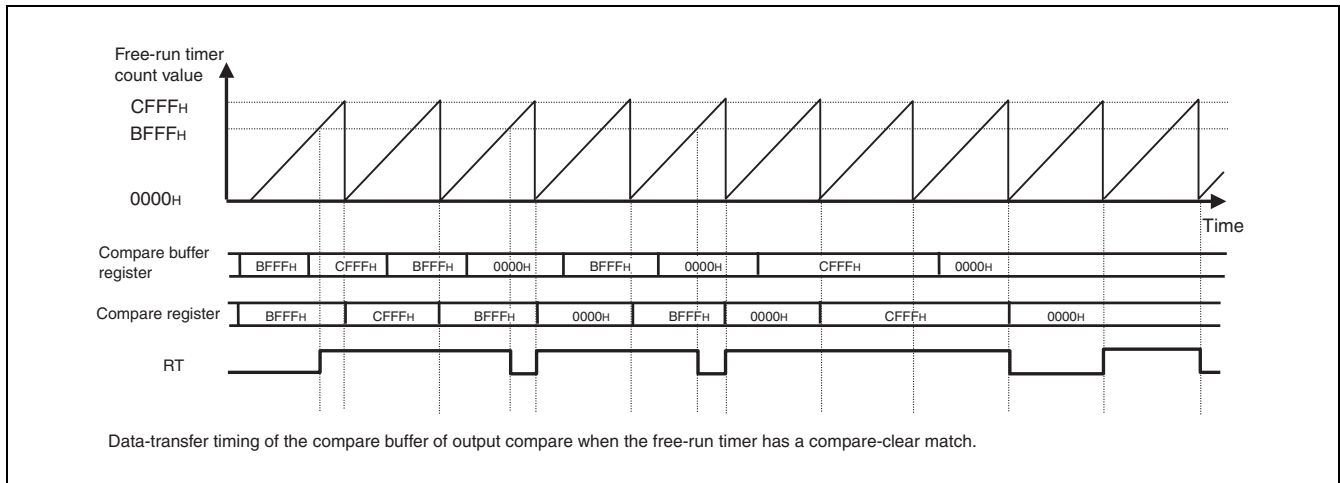


Figure 11.6-18 Compare Timing of Pin Output

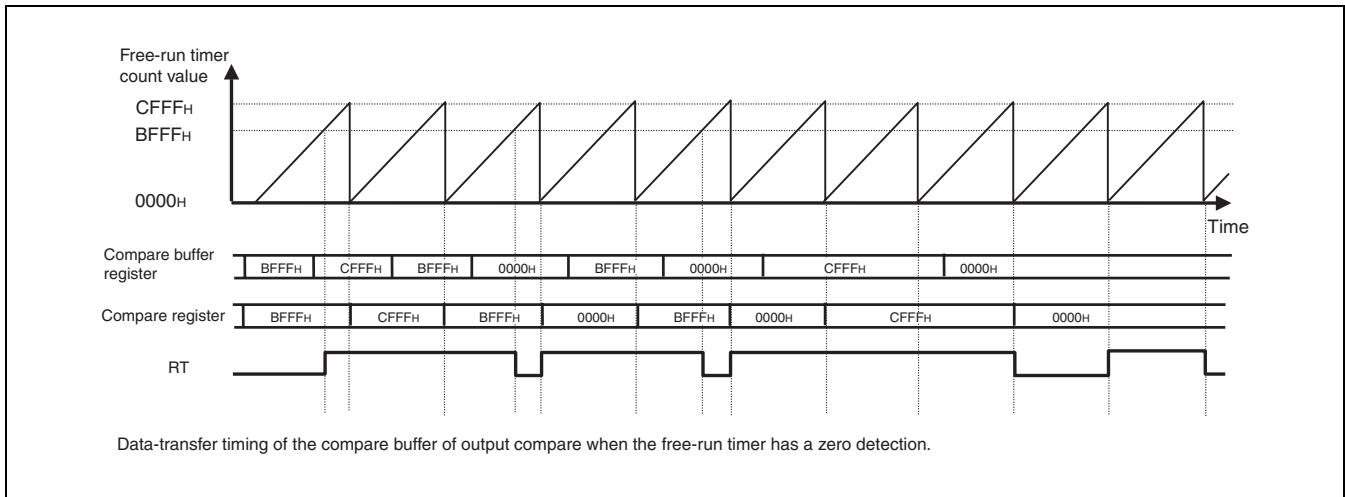


■ 16-bit Output Compare and Free-run Timer Operation

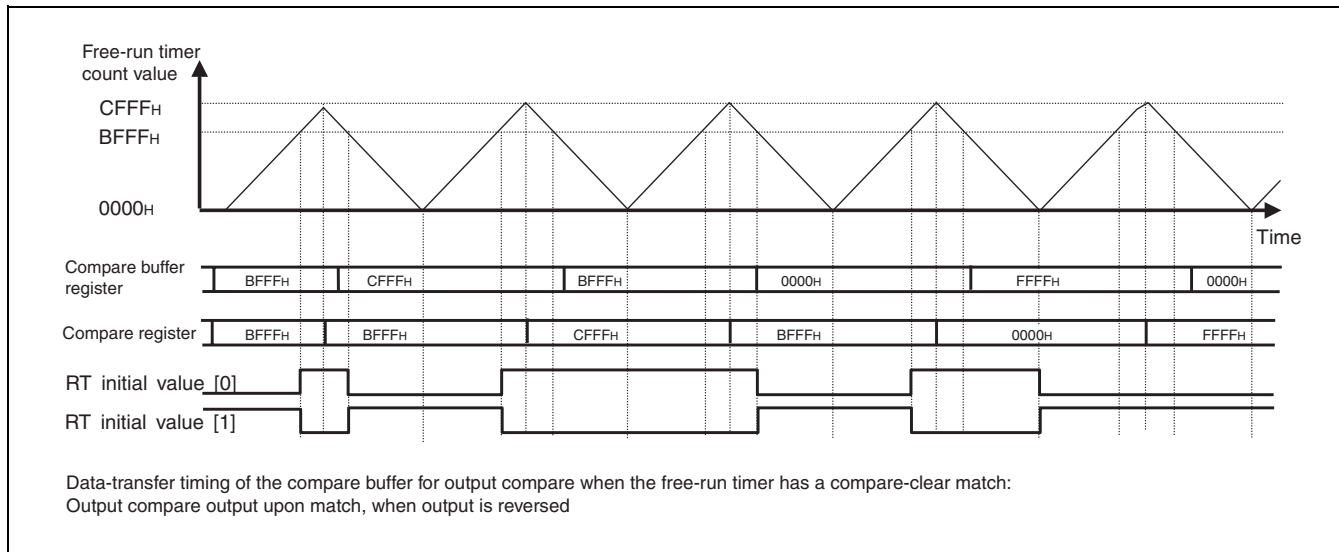
- Free-run timer is in up count mode:



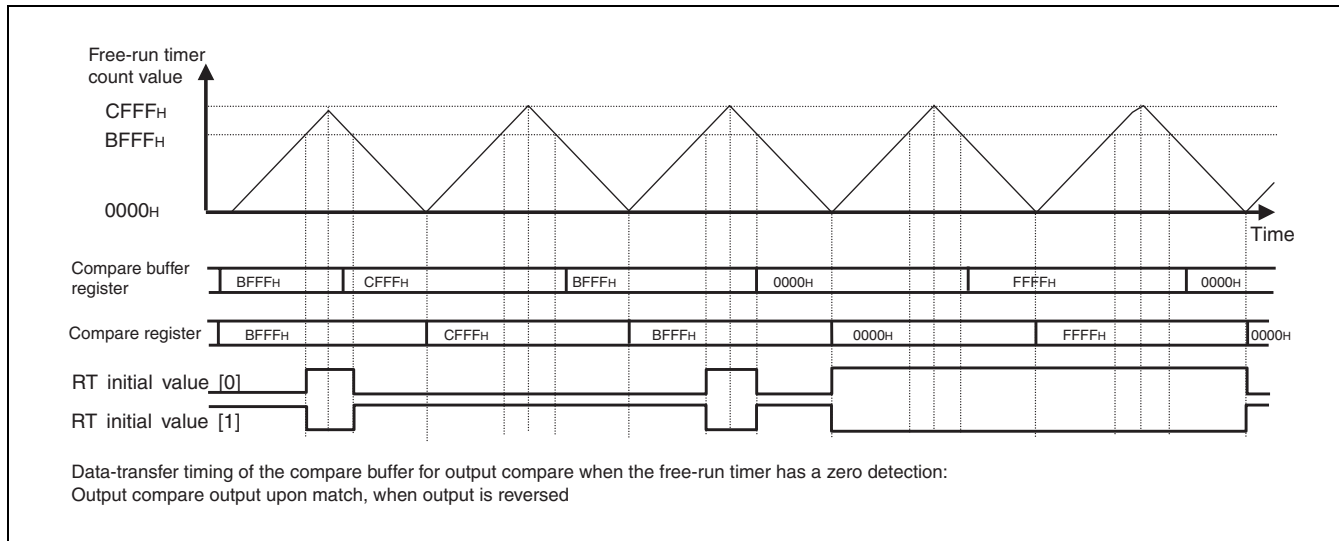
- Free-run timer is in up count mode:



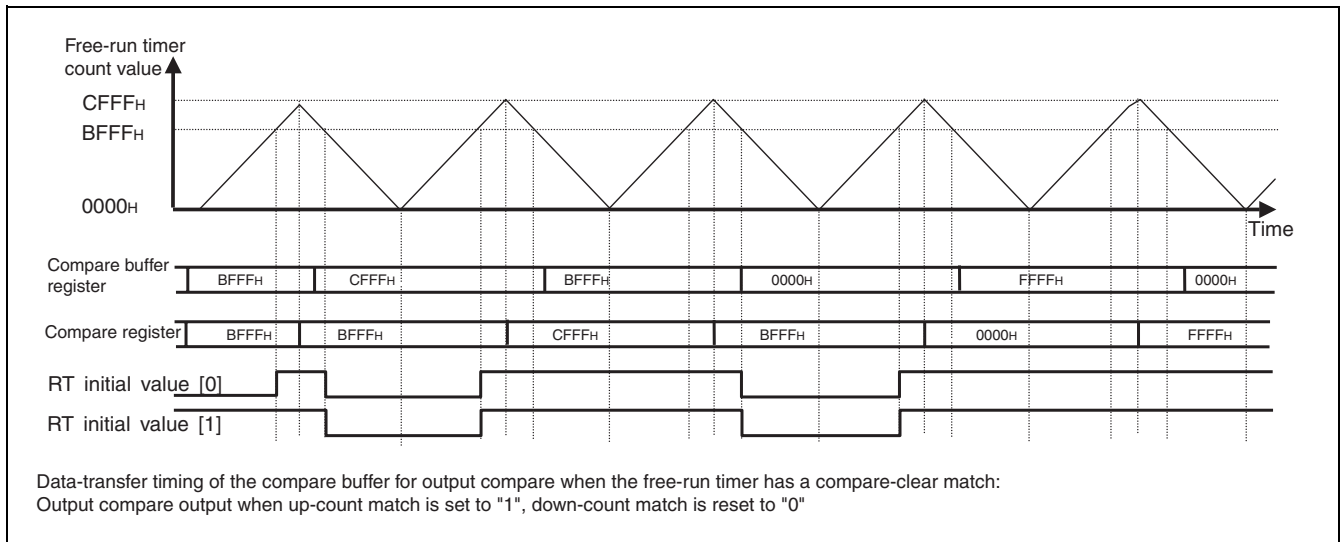
- Free-run timer is in up/down count mode:



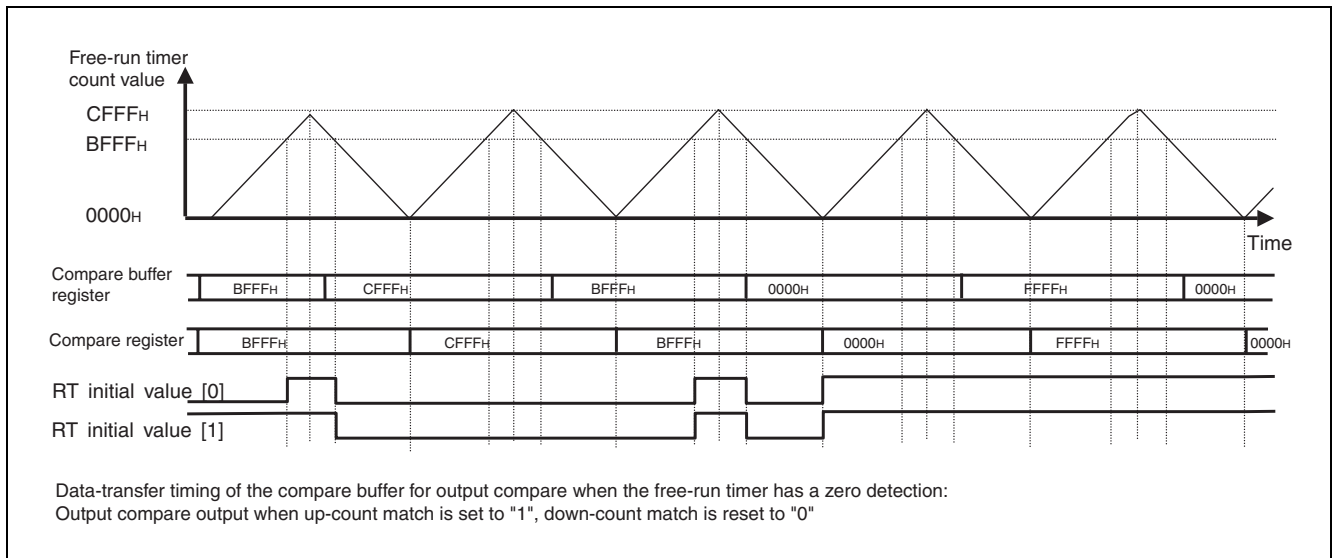
- Free-run timer is in up/down count mode:



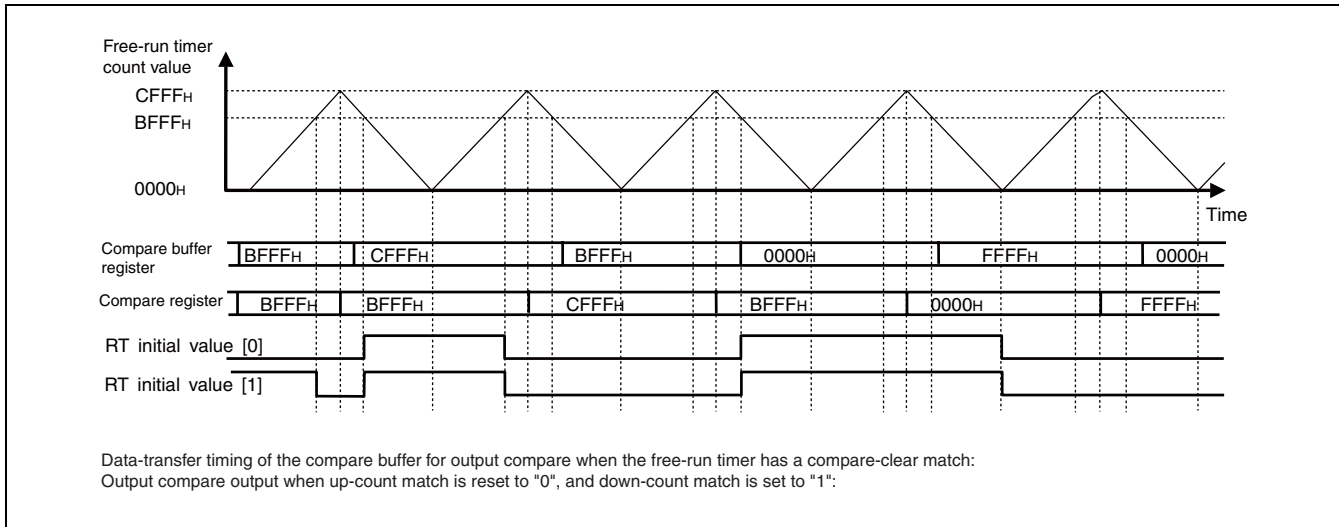
- Free-run timer is in up/down count mode:



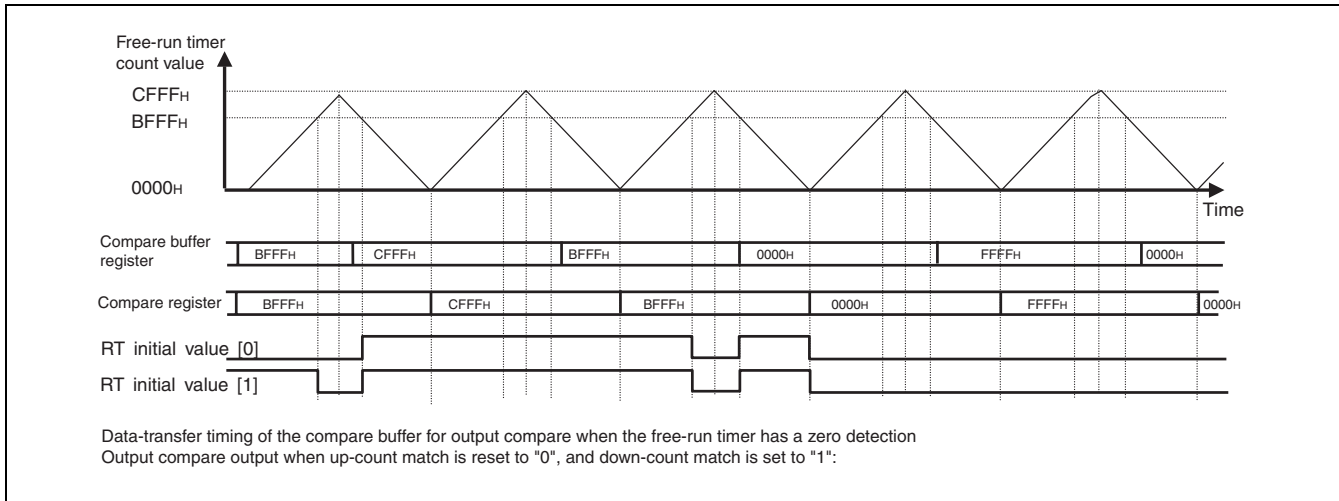
- Free-run timer is in up/down count mode:



- Free-run timer is in up/down count mode:



- Free-run timer is in up/down count mode:

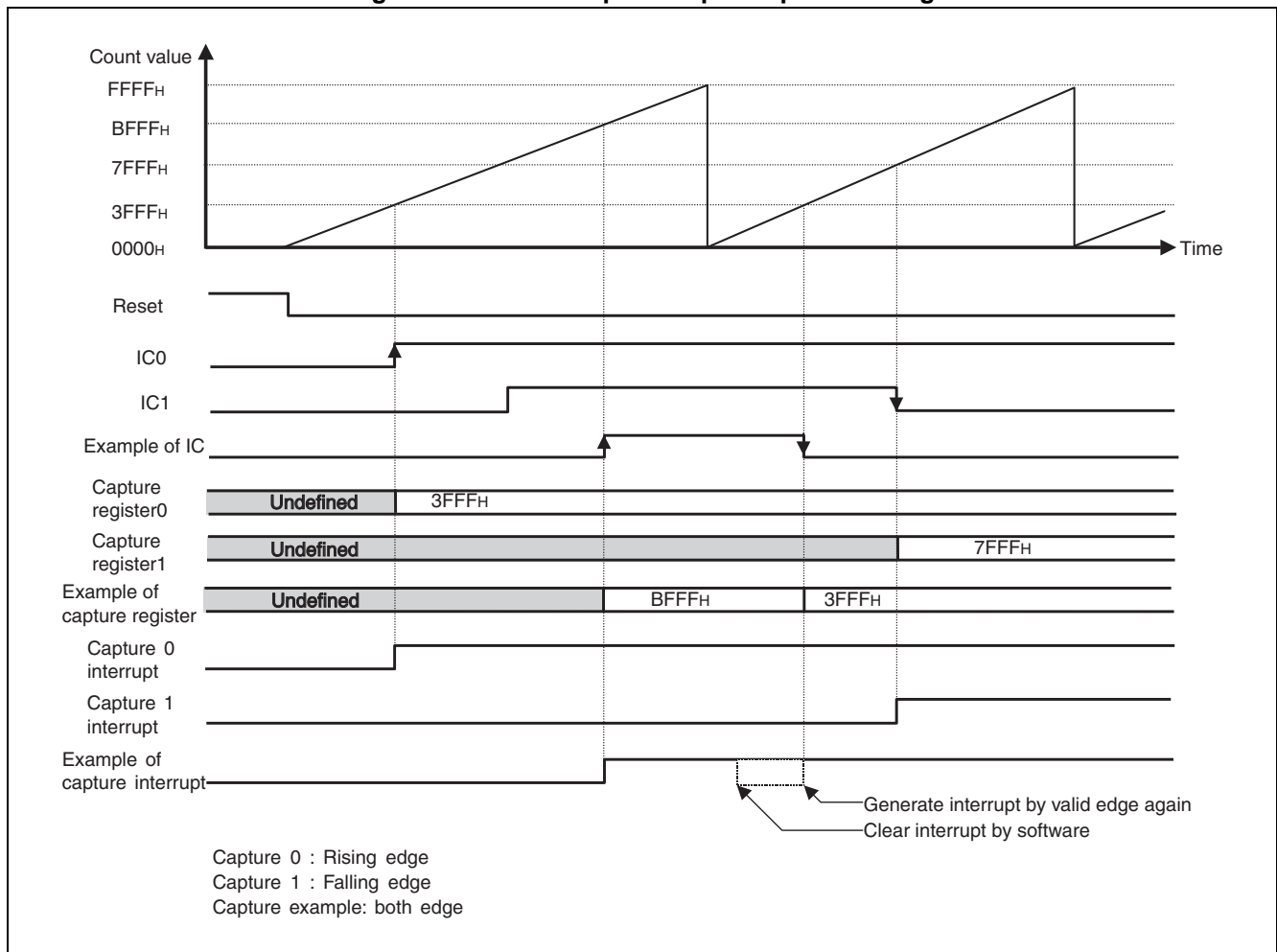


11.6.3 Operation of 16-bit Input Capture

Input capture is used detect specified valid edges. When a valid edge is detected, the interrupt flag is set, and the value of the 16-bit free-run timer is loaded into the capture register.

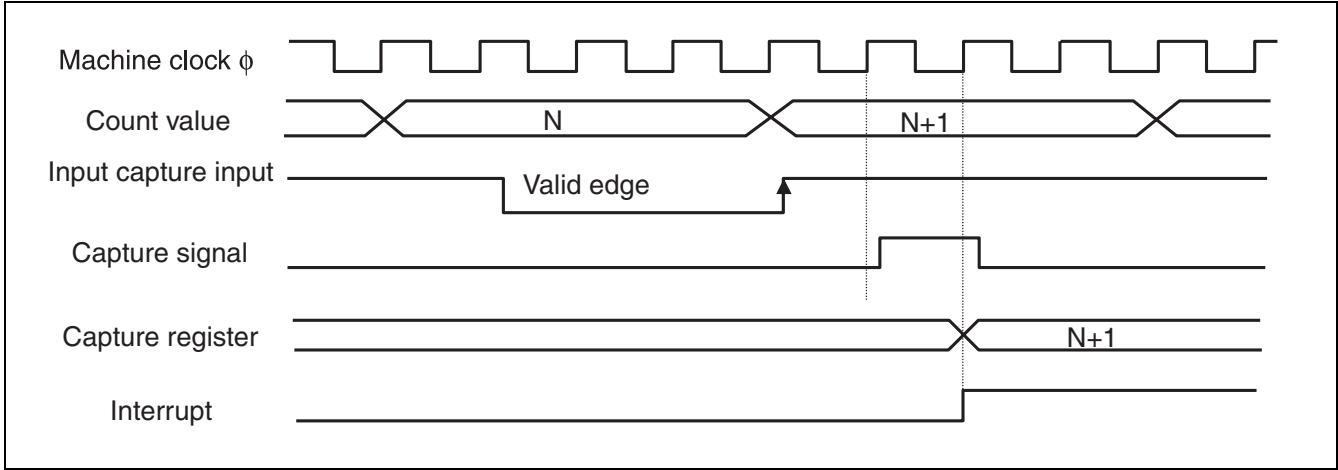
■ Operation of 16-bit Input Capture

Figure 11.6-19 Example of Input Capture Timing



■ Input Timing of 16-bit Input Capture

Figure 11.6-20 Example of 16-bit Input Capture Timing for Input Signal



11.6.4 Waveform Generator Operation

The waveform generator can generate a variety of waveforms (including dead times) using real-time output (RTO0 to RTO5), the 16-bit PPG timer 0, and 16-bit dead timers 0, 1, and 2.

■ Output Status of RTO0 to RTO5 and GATE

Table 11.6-1 RTO0 to RTO5/GATE Output Status and Bit Settings

TMD2	TMD1	TMD0	GTEN5 to GTEN0	PGEN5 to PGEN0	RTO0 to RTO5	GATE
0	0	0	X	X	Real-time output RTx (16-bit output compare output)	Always "0"
0	0	1	X	0	Real-time output RTx (16-bit output compare output)	(RTx & GTENx) *3
0	0	1	0	1	RTx outputs a PPG0 pulse for duration "H". *1	Always "0"
0	0	1	1	1	RTx outputs the PPG0 pulse activated by the GATE signal for duration "H".	(RT0/RT1/ RT2/RT3/ RT4/RT5)
0	1	0	X	0	The RT0, RT1 rising edge activates the 16-bit dead timer 0, and the 16-bit dead timer 0 outputs "H" until it underflows.	Output "H" during the timer operation *4
			X		The RT2, RT3 rising edge activates the 16-bit dead timer 1, and the 16-bit dead timer 1 outputs "H" until it underflows.	
			X		The RT4, RT5 rising edge activates the 16-bit dead timer 2, and the 16-bit dead timer 2 outputs "H" until it underflows.	
0	1	0	0	1	The RT0, RT1 rising edge activates the 16-bit dead timer 0, and the 16-bit dead timer 0 outputs the PPG 0 pulse until it underflows. *1	Always "0"
			0		The RT2, RT3 rising edge activates the 16-bit dead timer 1, and the 16-bit dead timer 1 outputs the PPG 0 pulse until it underflows. *1	
			0		The RT4, RT5 rising edge activates the 16-bit dead timer 2, and the 16-bit dead timer 2 outputs the PPG 0 pulse until it underflows. *1	
0	1	0	1	1	The RT0, RT1 rising edge activates the 16-bit dead timer 0, and the 16-bit dead timer 0 outputs the PPG 0 pulse started by the GATE signal until it underflows.	Output "H" during the timer operation *4
			1		The RT2, RT3 rising edge activates the 16-bit dead timer 1, and the 16-bit dead timer 1 outputs the PPG 0 pulse started by the GATE signal until it underflows.	
			1		The RT4, RT5 rising edge activates the 16-bit dead timer 2, and the 16-bit dead timer 2 outputs the PPG 0 pulse started by the GATE signal until it underflows.	
1	0	0	X	X	A non-overlapping signal is generated by means of RT1. *2	Always "0"
			X		A non-overlapping signal is generated by means of RT3. *2	
			X		A non-overlapping signal is generated by means of RT5. *2	
1	1	1	0	X	A non-overlapping signal is generated by means of PPG 0.	Always "0"
1	1	1	1	X	A non-overlapping signal is generated by means of the PPG 0 activated by the GATE signal.	(RT0/RT1/ RT2/RT3/ RT4/RT5)
Others					Always "0"	Always "0"

- *1: It is necessary to activate PPG 0 beforehand.
- *2: In order to generate a non-overlapping signal, first select 2-channel mode (compare control register higher-order (OCSH1, OCSH3, and OCSH5) CMOD: bit12 = 1) for RT1, RT3, and RT5.
- *3: The GATE signal is generated from the RTx whose GTENx bit is set to "1".
- *4: The GATE signal is generated while the timer activated by the RTx whose GTENx bit is set to "1" is operating. If more than one GATEx bit is set to "1", the GATE signal is the OR of the signals for each of the operating timers.

Note:

RTO0 and RTO1 are controlled by the 16-bit dead timer control register's higher order (DTCR0) TMD2 to TMD0: bit10 to bit8; RTO2 and RTO3 are controlled by lower-order register's (DTCR1) TMD5 to TMD3: bit2 to bit0; and RTO4 and RTO5 are controlled by the higher-order register's (DTCR2) TMD8 to TMD6: bit10 to bit8.

■ PPG0 Output Control

PPG0 output to RTO0 to RTO5 pins can be enabled by means of the PPG output control/input capture-status control register higher-order (PICSH01) PGEN 5 to PGEN 0: bit15 to bit10.

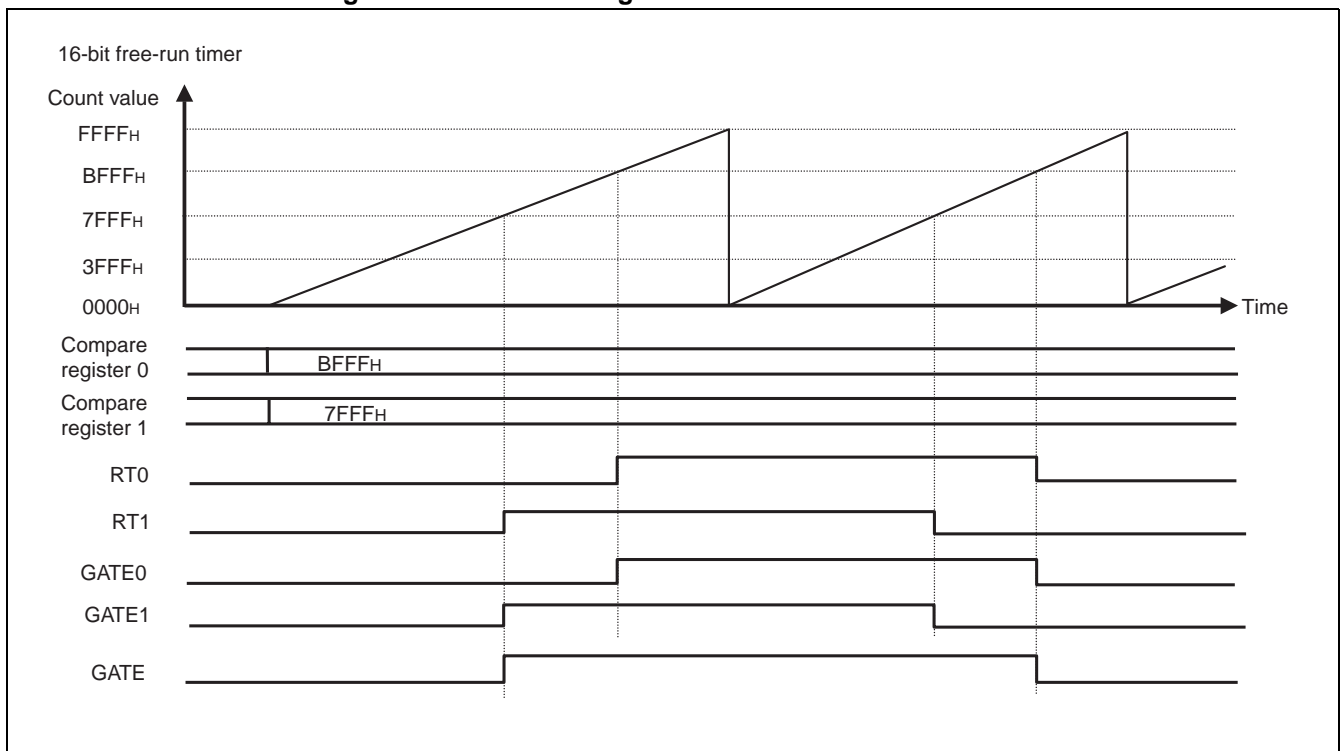
■ PPG0 Output Via Gate Trigger

The waveform generator can generate a GATE signal via real-time output RTO0 to RTO5, and the 16-bit dead timer 0, 1, and 2 can operate the PPG0 count as a trigger. Two real-time outputs (RTO0/RTO2/RTO4 and RTO1/RTO3/RTO5) are controlled by one 16-bit dead timer 0, 1, and 2, generating six separate gate signals. The GATE signal is generated from the logical sum of these gate signals, and becomes trigger the PPG0 count. Also, if PGEN0 to PGEN5 signals are used, it is possible to output 6 different waveforms to RTO0 to RTO5 pins using PPG0 alone.

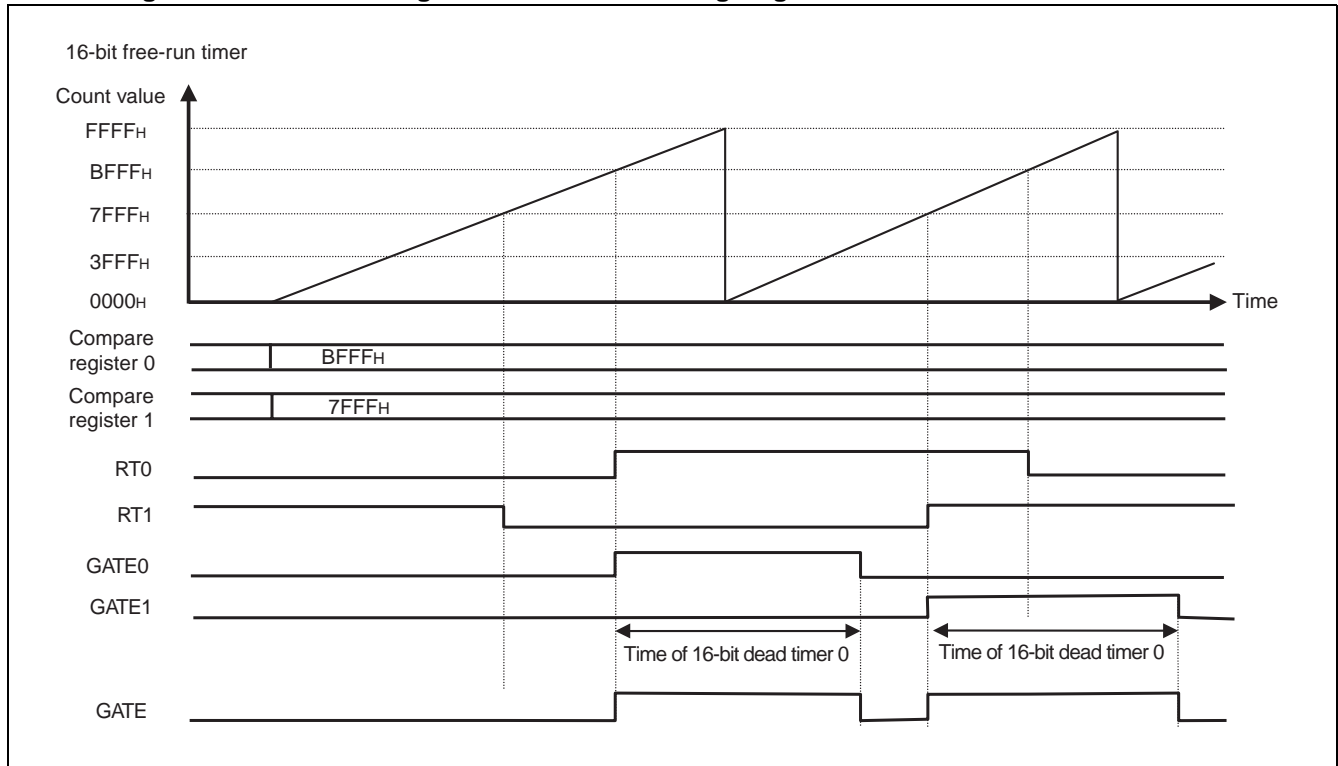
● GATE signal generation at following state;

- when PGENx, GTENx is active (PGENx, GTENx=1) and TMD8 to TMD0 of 16-bit dead timer control register (DTCR0, DTCR1, DTCR2) = 000_B
- when GTENx is active and the TMD8 to TMD0 = 111_B

Figure 11.6-21 GATE Signal Generated when RTx Is "H"



- GATE signal generated when GTENx is active (DTCR 0, DTCR1, and DTCR2 register's TMD8 to TMD0 = 010_B), until the 16-bit dead timer 0, 1, and 2 underflow from the RTx rising edge.

Figure 11.6-22 GATE Signal between RTx Rising Edge and 16-bit Dead Timer Underflow**Note:**

Each 16-bit dead timer is used for two RT. In other words, 16-bit dead timer 0 is used for RT0 and RT1; 16-bit dead timer 1 is used for RT 2 and RT 3; and 16-bit dead timer 3 is used for RT 4 and RT 5. Consequently, you must not try to use a RT to activate a timer that is already operating. If such an attempt is made, the GATE signal output will be extended, which could result in a malfunction.

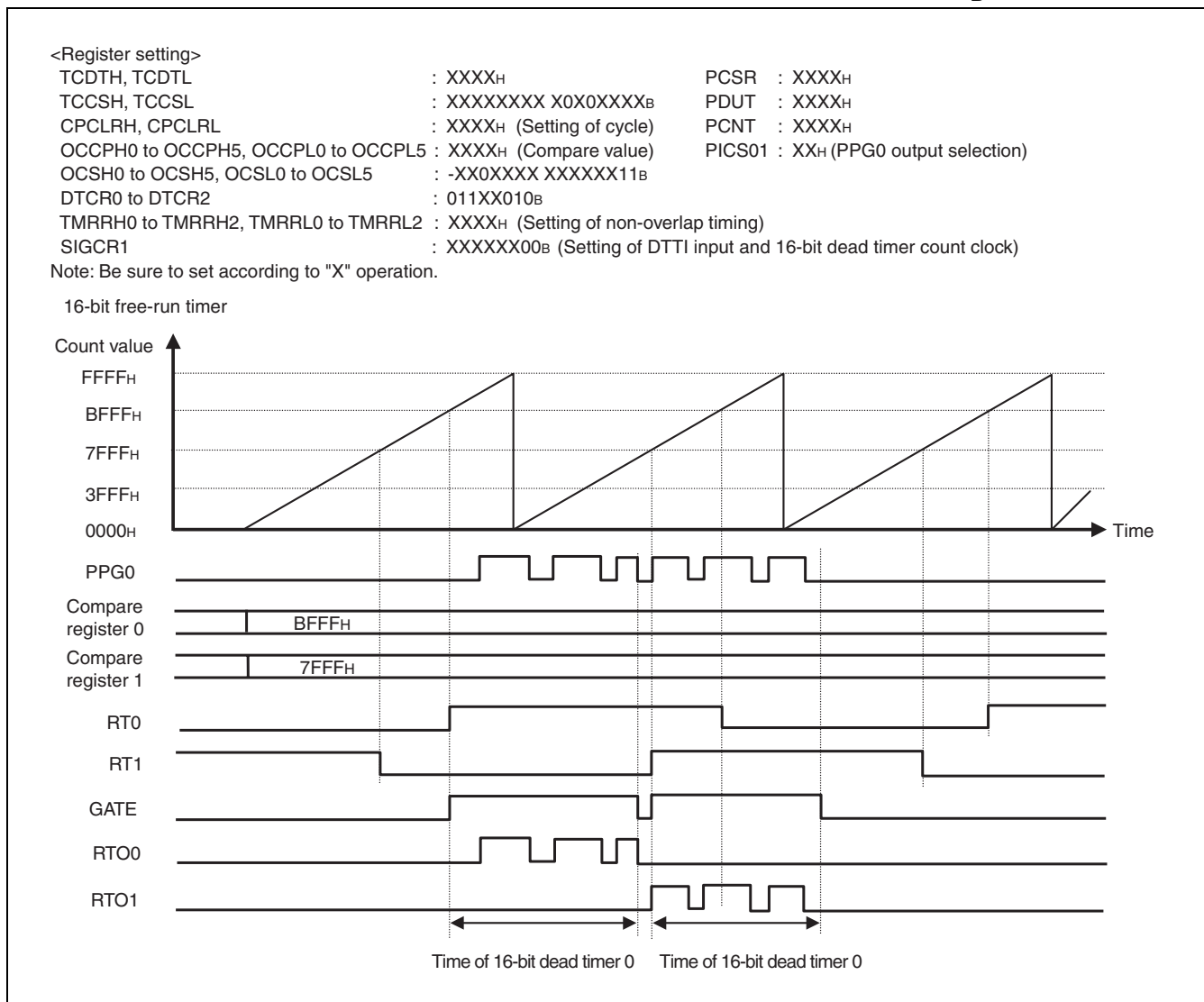
11.6.4.1 Operation of Timer Mode

When an RTO0 to RTO5 pin's rising edge is detected, the value is reloaded into the 16-bit dead timer, and the 16-bit dead timer starts counting down. PPG timer 0 continues to output to RTO0 to RTO5 pins until the 16-bit dead timer generates an underflow.

■ Operation of Timer Mode

- The PPG0 output pulse is generated from the RT rising edge until the 16-bit dead timer underflows (DTCR0, DTCR1, and DTCR2 register's TMD8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) = 010_B).

**Figure 11.6-23 Waveform Generated when TMD2 to TMD0
(Upper-order Bits are 10 to 8; Lower-Order Bits are 2 to 0) are "010_B"**



Note:

Each 16-bit dead timer is used for two RT. In other words, 16-bit dead timer 0 is used for RT 0 and RT 1; 16-bit dead timer 1 is used for RT2 and RT3; and 16-bit dead timer 2 is used for RT4 and RT5. Consequently, you must not try to use a RT to activate a PPG0 that is already operating. If such an attempt is made, the GATE signal output will be extended, which could result in a malfunction.

11.6.4.2 Operation During Dead Time Timer Mode

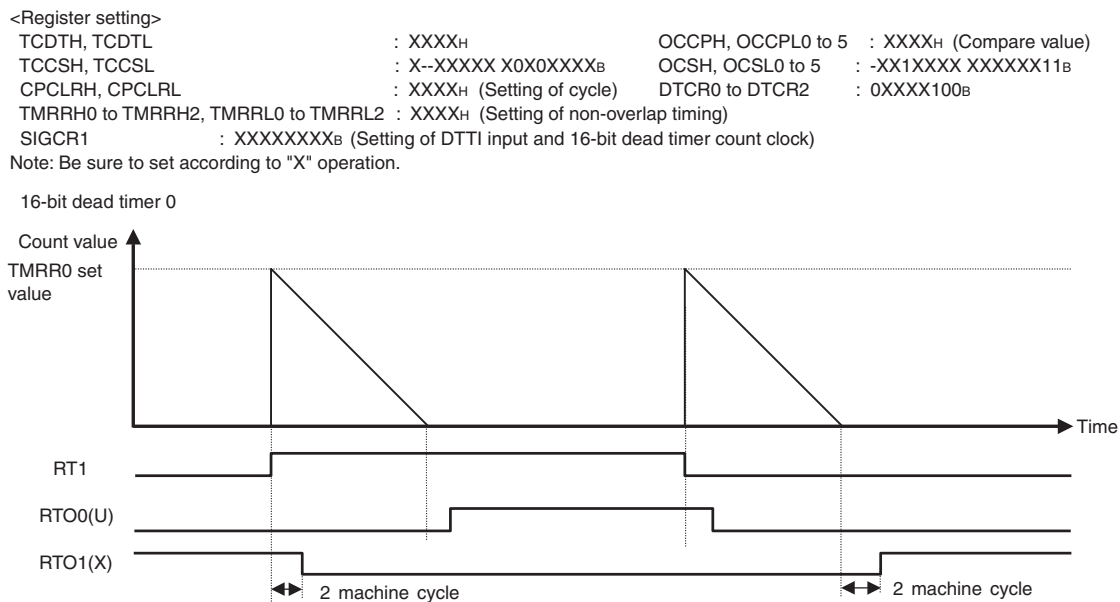
The dead-time generator inputs real-time output (RT1, RT3, and RT5) or PPG0 timer pulse output and outputs a non-overlapping signal (reverse signal) to the external pins (RTO0 to RTO5).

■ Operation During Dead Time Timer Mode

- This non-overlapping signal is generated via normal-polarity RT1, RT3, and RT5 (16-bit dead timer control register (DTCR0, DTCR1, and DTCR2) TMD8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) = 100_B).

When the DTCR0, DTCR1, and DTCR2 register DMOD2 to DMOD0 selects the non-overlapping signal with a value of 0 (normal polarity), a delay equivalent to the non-overlap time set in the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) is applied. This delay is applied to the rising edge or falling edge of the RT1, RT3, and RT5 pins. If the RT1, RT3, and RT5 pulse width is less than the specified non-overlap time, the 16-bit dead timer starts counting down again from the next RT edge TMRRH0 to TMRRH2, TMRRL0 to TMRRL2 register value.

Figure 11.6-24 Non-overlap Signal Generated by RT1, RT3, and RT5 of Normal Polarity



Pin name	Output signal
RTO0 (U)	Delayed signal is applied at rising edge of RT1.
RTO2 (V)	Delayed signal is applied at rising edge of RT3.
RTO4 (W)	Delayed signal is applied at rising edge of RT5.
RTO1 (X)	Delayed inverted signal is applied at falling edge of RT1.
RTO3 (Y)	Delayed inverted signal is applied at falling edge of RT3.
RTO5 (Z)	Delayed inverted signal is applied at falling edge of RT5.

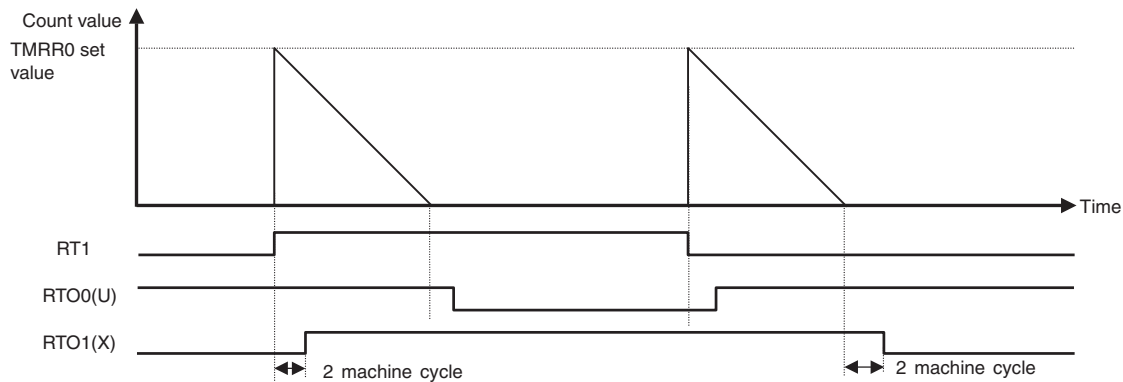
- This non-overlapping signal is generated via reverse-polarity RT1, RT3, and RT5 (16-bit dead timer control register (DTCR0, DTCR1, and DTCR2) TMD8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) = 100_B).

When the DTCR0, DTCR1, and DTCR2 register's DMOD2 to DMOD0 (higher-order bit is 15; lower-order bit is 7) selects the non-overlapping signal with a value of 1 (reverse polarity), a delay equivalent to the non-overlap time set in the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) is applied. This delay is applied to the rising edge or falling edge of the RT1, RT3, and RT5. If the RT1, RT3, and RT5 pulse width is less than the specified non-overlap time, the 16-bit dead timer starts counting down again from the next RT edge TMRRH0 to TMRRH2 and TMRRL0 to TMRRL2 value.

Figure 11.6-25 Non-overlap Signal Generated by RT1, RT3, and RT5 of Inverted Polarity

<Register setting>
 TCDTH, TCDTL : XXXXH OCCPH0 to OCCPH5, OCCPL0 to OCCPL5 : XXXXH (Compare value)
 TCCSH, TCCSL : XXXXXXXX X0X0XXXXB OCSH0 to OCSH5, OCSL0 to OCSL5 : -XX1XXXX XXXXXX11B
 CPCLRH, CPCLRL : XXXXH (Setting of cycle) DTCR0 to DTCR2 : 1XXXX100B
 TMRRH0 to TMRRH2, TMRRL0 to TMRRL2 : XXXXH (Setting of non-overlap timing)
 SIGCR1 : XXXXXXXXB (Setting of DTTI input and 16-bit dead timer count clock)
 Note: Be sure to set according to "X" operation.

16-bit dead timer 0



Pin name	Output signal
RTO0 (U)	Delayed inverted signal is applied at rising edge of RT1.
RTO2 (V)	Delayed inverted signal is applied at rising edge of RT3.
RTO4 (W)	Delayed inverted signal is applied at rising edge of RT5.
RTO1 (X)	Delayed signal is applied at falling edge of RT1.
RTO3 (Y)	Delayed signal is applied at falling edge of RT3.
RTO5 (Z)	Delayed signal is applied at falling edge of RT5.

- This non-overlapping signal is generated via normal-polarity PPG (16-bit dead timer control register (DTCR0, DTCR1, and DTCR2) TMD 8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) = 111_B).

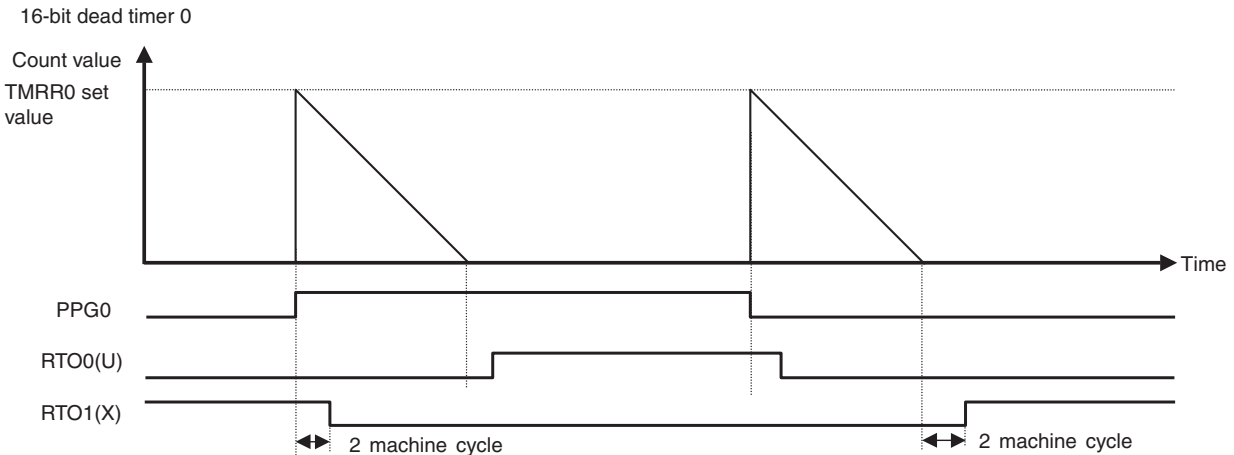
When the DTCR0, DTCR1, and DTCR2 register's DMOD2 to DMOD0 (higher-order bit is 15; lower-order bit is 7) selects the non-overlapping signal with a value of 0 (normal polarity), a delay equivalent to the non-overlap time set in the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) is applied. This delay is applied to the PPG0 timer pulse signal or reverse signal's rising edge. If the PPG0 timer pulse width is less than the specified non-overlap time, the 16-bit dead timer starts counting down again from the TMRRH0 to TMRRH2, TMRRL0 to TMRRL2 value of the edge following that PPG0 pulse.

Figure 11.6-26 Non-overlap Signal Generated by PPG0 Timer of Normal Polarity

<Register setting>

TCDTH, TCDTL	: XXXXH	PCSR : XXXXH
TCCSH, TCCSL	: XXXXXXXX X0X0XXXX _B	PDUT : XXXXH
CPCLRH, CPCLRL	: XXXXH (Setting of cycle)	PCNT : XXXXH
OCCPH0 to OCCPH5, OCCPL0 to OCCPL5	: XXXXH (Compare value)	
OCSH0 to OCSH5, OCSL0 to OCSL5	: -XX1XXXX XXXXXX11 _B	
DTCR0 to DTCR2	: 0XXXX111 _B	
TMRRH0 to TMRRH2, TMRRL0 to TMRRL2	: XXXXH (Setting of non-overlap timing)	
SIGCR1	: XXXXXXX _B (Setting of DTTI input and 16-bit dead timer count clock)	

Note: Be sure to set according to "X" operation.

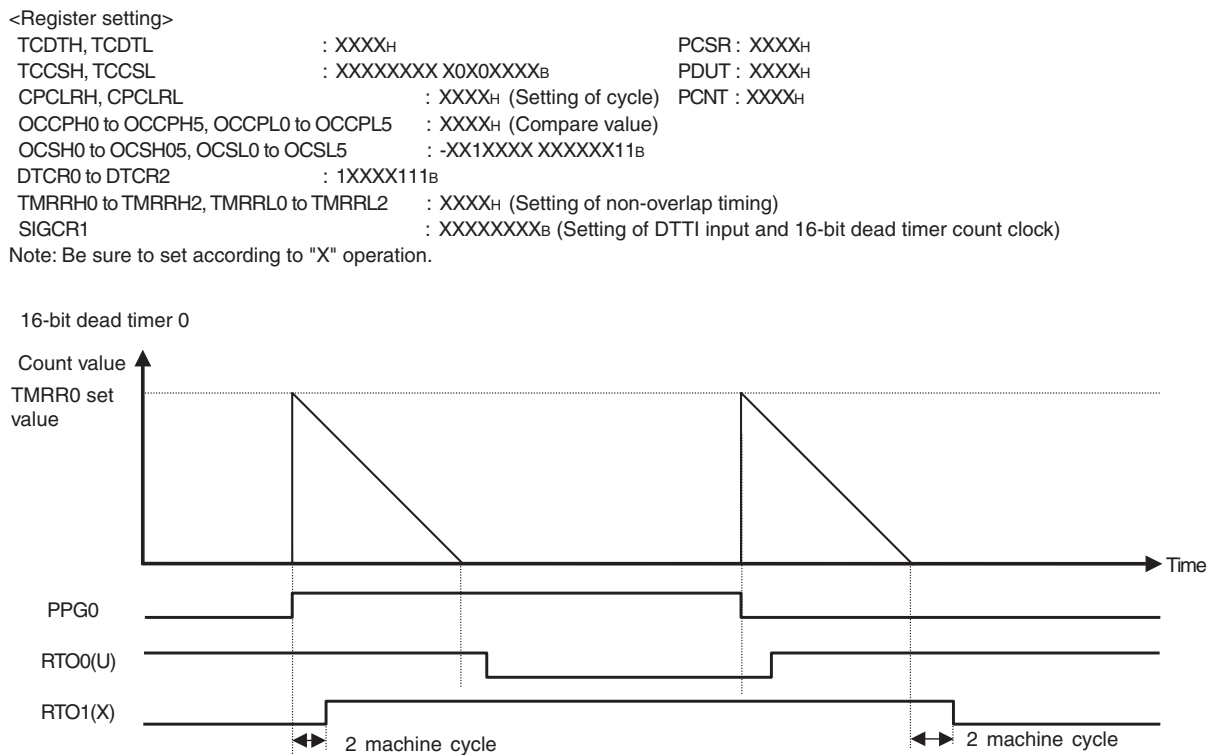


Pin name	Output signal
RTO0 (U)	Delayed signal is applied at rising edge of PPG0.
RTO2 (V)	Delayed signal is applied at rising edge of PPG0.
RTO4 (W)	Delayed signal is applied at rising edge of PPG0.
RTO1 (X)	Delayed inverted signal is applied at falling edge of PPG0.
RTO3 (Y)	Delayed inverted signal is applied at falling edge of PPG0.
RTO5 (Z)	Delayed inverted signal is applied at falling edge of PPG0.

- This non-overlapping signal is generated via reverse-polarity PPG (16-bit dead timer control register (DTCR0, DTCR1, and DTCR2) TMD8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) = 111_B).

When the DTCR0, DTCR1, and DTCR2 register's DMOD2 to DMOD0 (higher-order bit is 15; lower-order bit is 7) selects the non-overlapping signal with a value of 1 (reverse polarity), a delay equivalent to the non-overlap time set in the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) is applied. This delay is applied to the PPG0 timer pulse signal or reverse signal's rising edge. If the PPG0 timer pulse width is less than the specified non-overlap time, the 16-bit dead timer starts counting down again from the TMRRH0 to TMRRH2 and TMRRL0 to TMRRL2 value of the edge following that PPG0 pulse.

Figure 11.6-27 Non-overlap Signal Generated by PPG0 Timer of Inverted Polarity



Pin name	Output signal
RTO0 (U)	Delayed inverted signal is applied at rising edge of PPG0.
RTO2 (V)	Delayed inverted signal is applied at rising edge of PPG0.
RTO4 (W)	Delayed inverted signal is applied at rising edge of PPG0.
RTO1 (X)	Delayed signal is applied at falling edge of PPG0.
RTO3 (Y)	Delayed signal is applied at falling edge of PPG0.
RTO5 (Z)	Delayed signal is applied at falling edge of PPG0.

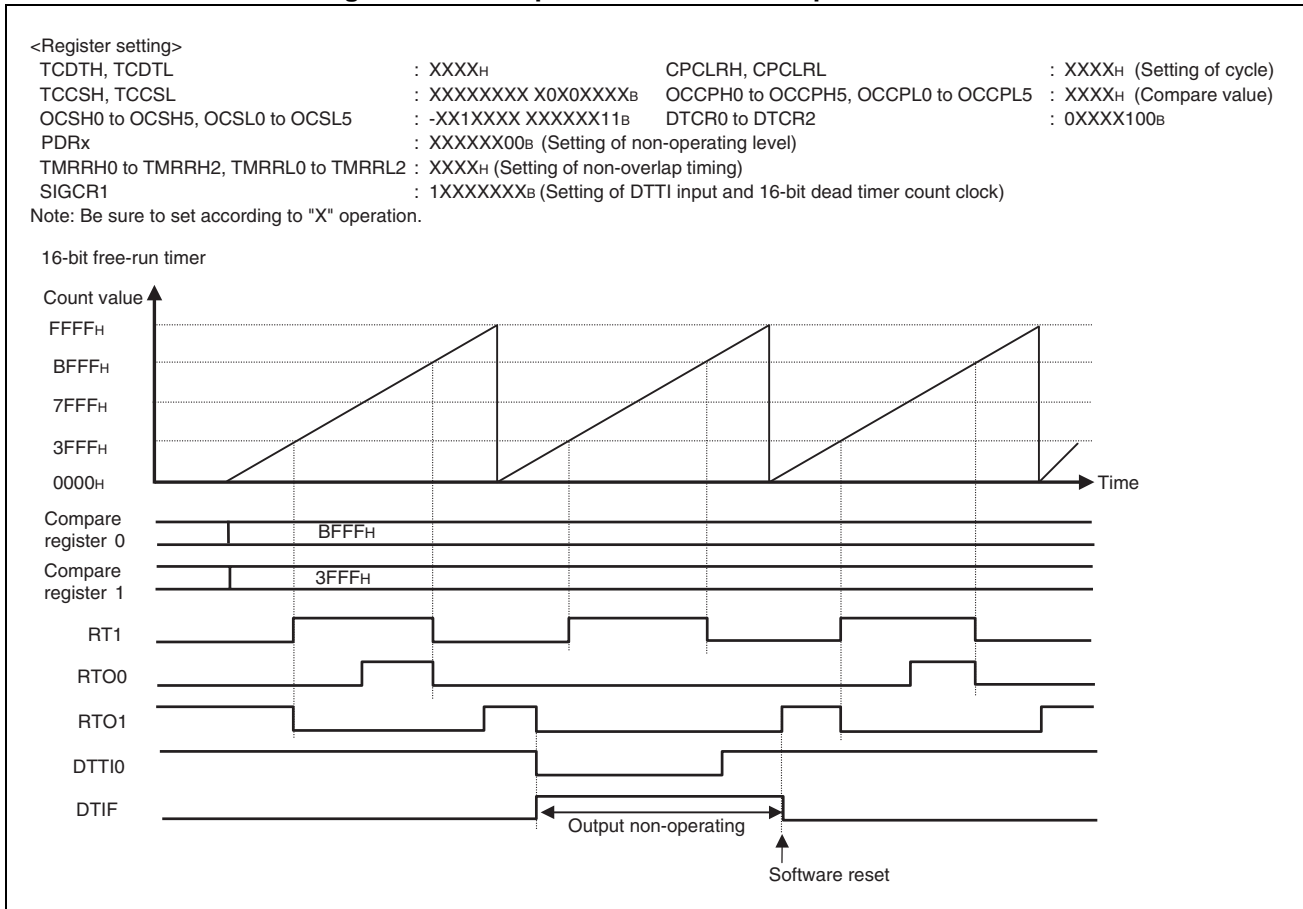
11.6.4.3 DTTI Pin Control Operation

You can control RTO0 to RTO5 output by means of the DTTI pins by setting "1" in the waveform control register 1 (SIGCR1) DTIE: bit7. When a DTTI pin "L" level is detected, RTO0 to RTO5 output is locked at non-operating level, until the interrupt flag (SIGCR register DTIF: bit6) is cleared. When RTO0 to RTO5 is at non-operating level, these pins can be set via software using the port data registers (PDR), which share them. Additionally, if they are used as input ports using the data direction register (DDR), Hi-Z is outputted.

■ DTTI Pin Input Operation

Even when "L" is detected in DTTI pin input, although the timer continues to operate while the waveform generator is operational, waveforms are not outputted to the external RTO0 to RTO5 pins.

Figure 11.6-28 Operation when DTTI Input Is Enabled



■ DTTI Operation of Waveform Control Register 2 (SIGCR2)

The output of waveform control register 2 DTTI: bit0 becomes to DTTI input using the DTTI pin input and OR. Consequently, when "0" is set in this register, control is permanently in DTTI input status, and the input from the DTTI pins has no meaning.

When this register is cleared by writing 1, the DTTI pin input value is used.

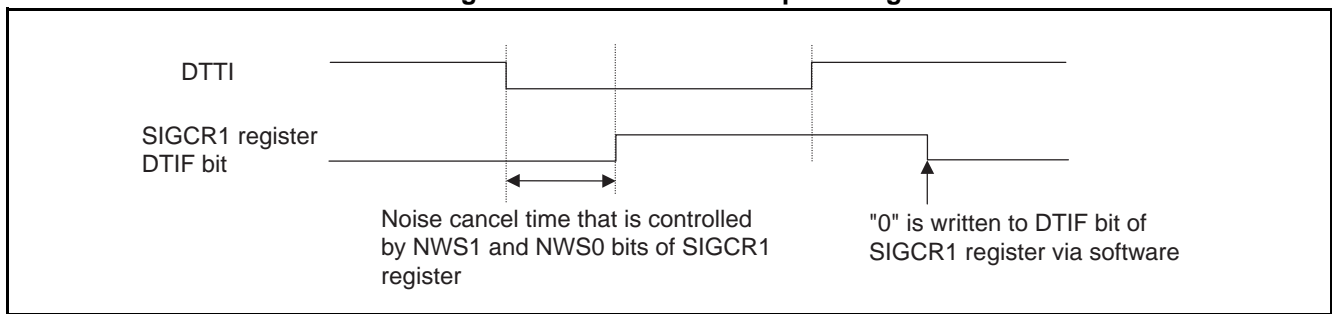
■ DTTI Pin Noise Cancel Feature

The DTTI pin input noise cancel feature is enabled when "1" is set in the waveform control register 1 (SIGCR1) NRSL: bit5. When the noise cancel feature is enabled, there is a delay of 4, 8, 16, or 32 machine cycles (selected via the SIGCR register NWS1 and NWS0: bit1 and bit0), for the amount of time necessary to lock the output pins (RTO0 to RTO5) to non-operating level. Since the noise cancel circuit uses resources, in modes where oscillation is stopped (e.g. stopped mode), input is disabled, even when DTTI input is enabled.

■ DTTI Interrupts

When DTTI "L" level is detected, after the noise cancel time has elapsed, the DTTI interrupt flag (SIGCR1 register DTIF: bit6) is set to "1", and an interrupt request is sent to the interrupt controller.

Figure 11.6-29 DTTI Interrupt Timing



Notes:

- If the SIGCR1 register NWS1 and NWS0 bit's values change within the noise cancel time, a larger (NWS1 and NWS0) noise cancel value is enabled.
- The SIGCR1 register DTIF: bit6 can only be cleared via software.

11.6.5 A/D Activation Compare Operation

When the 16-bit free-run timer reaches the specified value, A/D can be activated.

■ A/D Startup

Three A/D converter units can be activated.

- A/D activation compare 0 → Start A/D unit 0.
- A/D activation compare 1 → Start A/D unit 1.
- A/D activation compare 2 → Start A/D unit 2.

■ A/D Startup Enable

If the compare register value is set, and "1" is set in the control register (ADCOMPC) CE0, CE1, and CE2: bit0, bit1, and bit2, when the free-run timer and compare register value match, an A/D activation signal is generated.

When "0" is set in CE0, CE1, and CE2, even if the free-run timer and compare register value match, an A/D activation signal is not generated.

If the compare register is written to when activation is enabled, the written value is compared for a match.

11.7 Notes on Using Multifunctional Timer

Heed the following cautions when using the multifunctional timer.

■ Notes on Using 16-bit Free-Run Timer

● Cautions for setting via the program

- When a reset is executed, although the timer value becomes "0000_H", the zero-detection interrupt flag is not set.
- Since the timer-mode bit (TCCSL register MODE: bit5) has a buffer, and so timer modes changed after zero detection are enabled.
- A software clear (TCCSL register SCLR: bit4 = 1) initializes the timer. At this time, if the timer-count clock is the machine cycle (ϕ), zero-detection interrupts are not generated. If the timer-count clock is a division of the machine cycle (ϕ), zero-detection interrupts are generated.
- When the compare value and count value match, if the count starts, the compare-clear flag is not set.

● Note on interruption

- If "1" is set in the timer status control register higher-order (TCCSH) IRQZF: bit14, then interrupt requests are enabled (TCCSH register IRQZE: bit13 = 1), control cannot return from interrupt processing. IRQZF: Always clear bit14.
- If "1" is set in the timer status control register higher-order (TCCSH) ICLR: bit9, then interrupt requests are enabled (TCCSH register ICRE: bit8 = 1), control cannot return from interrupt processing. ICLR: Always clear bit9.

■ Notes on Using 16-bit Output Compare

● Note on interruption

If "11_B" is set in the compare control register lower-order (OCSL0, OCSL2, and OCSL4) IOP1, IOP0: bit7 and bit6, then interrupt requests are enabled (OCSL register IOE1, IOE0: bit6 and bit5 = 11_B), control cannot return from interrupt processing. Always clear the IOP0, IOP1 bits.

■ Cautions for Use of 16-bit Input Capture

● Note on interruption

- If "1" is set in the input capture-status control register's lower-order (PICSL01 and ICSL23) ICP3, ICP2, ICP1, and ICP0 (bit7 and bit6 of both), then interrupt requests are enabled (PICSL01 and ICSL23 registers' ICE3, ICE2, ICE1, and ICE0 (bit5 and bit4 of both) = 11_B), control cannot return from interrupt processing. Be sure to clear ICP3, ICP2, ICP1, and 0 (both bit7 and bit6).
- If the level of the input capture pin (IC) changes between the time that ICP bit3, bit2, bit1, and bit0 are set, and the interrupt routine is processed, then the ICP3 and ICP2 valid edge specification bits (ICSH 23 register's IEI3, IEI2: bit9 and bit8) will indicate the last edge detected.

Note:

There are no ICP1, ICP0 valid edge specification bits.

■ Notes on Using Waveform Generator

● Cautions for setting via the program

- Before changing the 16-bit dead timer control register's (DTCR0, DTCR1, DTCR2) TMD8, TMD5, and TMD2 (higher-order bit is 10; lower-order bit is 2), TMD7, TMD4, and TMD1 (higher-order bit is 9; lower-order bit is 1), or TMD6, TMD3, and TMD0 (higher-order bit is 8; lower-order bit is 0) during waveform generator operation (DTCR0, DTCR1, DTCR2 register's TMD2 to TMD0, TMD5 to TMD3, TMD8 to TMD6 is "001_B", "010_B", or "111_B"), be sure that the trigger source and 16-bit dead timer are not counting. If this operation is not performed, the output previously scheduled by the trigger causes an unexpected waveform to be outputted from the RTO pins. However, normal RTO output will resume after the timer underflows, or another trigger is generated by a new trigger source.
- The trigger source is "H" level of RT when the DTCR0, DTCR1, and DTCR2 register's TMD8 to TMD0 (higher-order bits are 10 to 8; lower-order bits are 2 to 0) is "001_B"; RT rising edge when TMD8 to TMD0 bits are "010_B"; RT rising or falling edge when TMD8 to TMD0 are "100_B"; and PPG0 rising edge or falling edge when TMD8 to TMD0 bits are "111_B".

For example, if TMD8 to TMD0 bits change from "100_B" to "111_B", the following steps can be executed.

1. Set the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) to an extremely small value like "0001_H".
 2. RTO1, RTO3, and RTO5 output is awaited until "L" or "H" is set, and timer 0, 1, and 2 underflow.
 3. Change the mode bits (TMD8 to TMD0) and corresponding settings.
 4. A corrected output waveform appears at the RTO pins after 1 machine cycle.
- If a value is written to the 16-bit dead timer register (TMRRH0 to TMRRH2, TMRRL0 to TMRRL2) during timer count, this new value is enabled during the next timer trigger. When accessing the timer registers, be sure to use half-word or word transfer commands.
 - Only change the waveform control register 1 (SIGCR1) DCK2 to DCK0: bit4 to bit2 when the timers are not counting.
 - Only change the waveform control register 1 (SIGCR1) NWS1 and NWS0: bit1 and bit0 when the noise cancel feature is disabled.

● Note on interruption

- If 1 is set in the 16-bit dead timer control register's (DTCR0, DTCR1, DTCR2) TMIF2 to TMIF0 (higher-order bit is 12; lower-order bit is 4), and then interrupt requests are enabled (DTCR0, DTCR1, DTCR2 register's TMIE2 to TMIE0 (higher-order bit is 11; lower-order bit is 3) = 1), control cannot return from interrupt processing. Always clear the TMIF bit.
- Control cannot return from interrupt processing after setting "1" in the waveform control register 1 (SIGCR1) DTIF: bit7. Always clear the DTIF bit.

11.8 Program Example of Multifunctional Timer

Below is a sample multifunctional timer program.

■ Program Example of 16-bit Free-run Timer

● Processing contents

- When the 16-bit free-run timer is 4 ms, generate a compare-clear interrupt.
- This timer is used to re-generate a compare-clear timer during up count mode.
- 16 MHz is for the machine clock, and 62.5 ns is for the count clock.

● Coding example

```

ICR32 .EQU    000460H           ; 16-bit free-run timer compare-clear interrupt control register
TCCSH .EQU    0000A8H           ; Timer control status register
CPCLRBH.EQU    0000A4H           ; Compare-clear buffer register
; ----- Main Program -----
                ORG      C0000H
START:
;               :               ; Assumes that the stack pointer (SP) has already been
;               :               ; initialized.
                ANDCCR   #0EFH           ; Disables the interrupt.
                LDI      #ICR32,r0
                LDI      #00H,r1
                STB       r1,@r0           ; Interrupt levels 16 (strength)
                LDI      #CPCLRBH,r0       ; Set a value in the compare-clear buffer register to
                LDI      #0FA00H,r1        ; generate a compare-clear interrupt at 4 ms when the
                STH       r1,@r0           ; 16-bit free-run timer is in up count mode.
                LDI      #TCCSH,r3        ; Up count down mode,
                LDI      #0110H,r1        ; 62.5 ns count clock setting,
                STH       r1,@r3          ; Compare-clear interrupt enable,
                                           ; Clear the interrupt flag bit for compare clearing.
                                           ; Disable interrupt masking,
                                           ; Clear timer and enable operation.
                STILM     #14H             ; ILM in PS is set to level 20
                ORCCR     #10H             ; Interruption permission
LOOP   LDI      #00H,r0                 ; Infinite loop
                LDI      #01H,r1
                BRA       LOOP
; ----- Interrupt Program -----
WARI   LDI      #0100H,r1
                ANDH      r1,@r3          ; Clear interrupt request flag.
;               :
;               : User processing
;               :

```

RETI : Returns from interrupt.

```
; ----- Vector Settings -----
VECT      .ORG      FFFF8H
          .DATA.W    WARI      ; Set interrupt routine.
          .ORG      FFFF8H
          .DATA.W    0x07000000 ; Set single-chip mode.
          .DATA.W    START     ; Set reset vectors.
          .END
```

■ Program Example of 16-bit Output Compare

● Processing contents

- When the 16-bit free-run timer count value matches the output compare value, an output compare match is generated.
- Use when the 16-bit free-run timer is in up/down count mode.
- 16 MHz is for the machine clock, and 62.5 ms is for the 16-bit free-run timer 0 count clock.

● Coding example

```
ICR44 EQU 00046CH ; Output compare 0/1 interrupt register
TCCSH EQU 0000A8H ; Timer control status register
CPCLRBHEQU 0000A4H ; Compare-clear buffer register
OCCPBH0EQU 000090H ; Output compare buffer register 0
OCCPBH1EQU 000092H ; Output compare buffer register 1
OCSH1 EQU 00009CH ; Compare control register
; ----- Main Program -----
START:
;      : ; Assumes that the stack pointer (SP) has already been
;      : ; initialized.
ANDCCR #0EFH ; Disables the interrupt.
LDI #ICR44,r0
LDI #00H,r1
STB r1,@r0 ; Interrupt levels 16 (strength)
LDI #CPCLRBH,r0 ; Set to 16-bit free-run timer's compare-clear
LDI #0FFFFH,r1 ; buffer register
STH r1,@r0

LDI #OCCPBH0,r0 ; Set the output compare register 0.
LDI #0BFFFH,r1
STH r1,@r0
LDI #OCCPBH1,r0 ; Set the output compare register 1.
LDI #07FFFH,r1
STH r1,@r0
LDI #OCSH1,r3 ; Enable output compare output.
LDI #6C33H,r2 ; Enable compare match interrupts 0/1.
STH r2,@r3 ; Clear the interrupt flag bit.

LDI #TCCSH,r0 ; Up count down mode,
```

```

        LDI    #0010H,r1          ; Clear timer and enable operation.
        STH    r1,@r0
        STILM  #14H              ; ILM in PS is set to level 20
        ORCCR  #10H              ; Interruption permission

LOOP    LDI    #00H,r0            ; Infinite loop
        LDI    #01H,r1
        BRA    LOOP              ;
; ----- Interrupt Program -----
WARI:
        ANDH   r2,@r3            ; Clear interrupt register flag.
;
;                               :
;                               User processing
;                               :
        RETI                     : Returns from interrupt.
; ----- Vector Settings -----
VECT    .ORG      FFFF8H
        .DATA.W   WARI          ; Set interrupt routine.
        .ORG      FFFF8H
        .DATA.W   0x07000000 ; Set single-chip mode.
        .DATA.W   START        ; Set reset vectors.
        .END

```

CHAPTER 12

U-TIMER (16-bit Timer for UART Baud Rate Generation)

This chapter describes the U-TIMER, the configuration and functions of registers, and U-TIMER operation.

12.1 Overview

12.2 Description of Registers

12.3 Description of Operation

12.1 Overview

The U-TIMER is a 16-bit timer used to generate the baud rate for the UART. Use a combination of a chip operating frequency and a reload value of the U-TIMER to specify a baud rate. Also, since it generates an interrupt by count underflow, it can be used as an interval timer.

The MB91260B series have three built-in channels for this timer. When used as an interval timer, two pairs of U-TIMERS can be cascaded to count a maximum interval of $2^{32} \times \phi$.

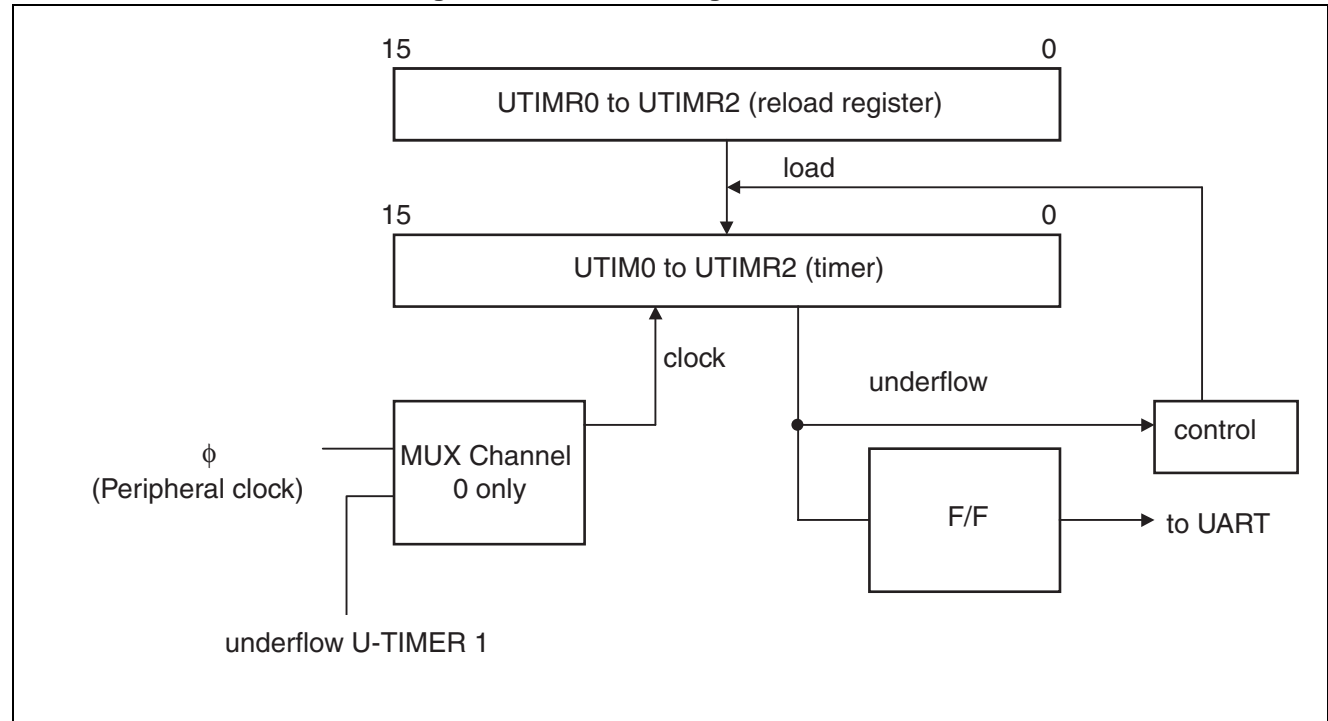
Only a combination of ch.0 and ch.1, and a combination of ch.0 and ch.2 can be connected in cascade mode.

■ Register List

15	8 7	0	
UTIM0 to UTIM2			(R)
UTIMR0 to UTIMR2			(W)
21UTIMC0 to UTIMC2			(R/W)

■ Block Diagram

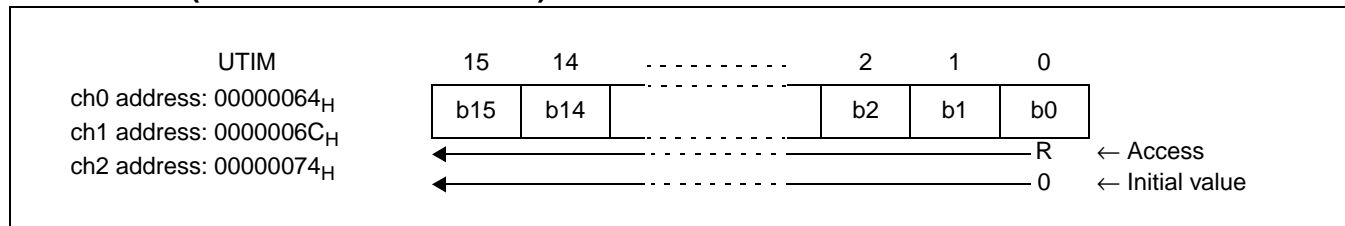
Figure 12.1-1 Block Diagram for U-TIMER



12.2 Description of Registers

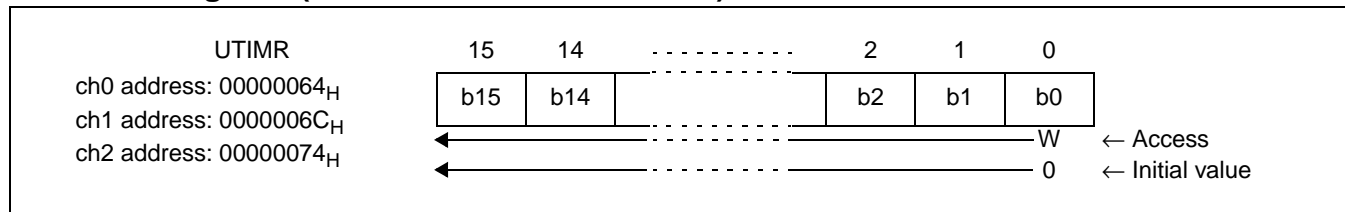
This section describes the configuration and functions of registers used by the U-TIMER.

■ U-TIMER (UTIM: UTIM0 to UTIM2)



UTIM is a register that indicates the timer value. Use a 16-bit transfer instruction to access this register.

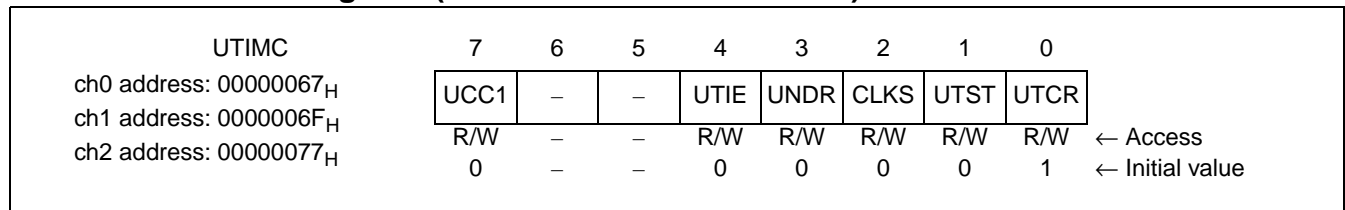
■ Reload Register (UTIMR: UTIMR0 to UTIMR2)



UTIMR is a register that stores the value to be reloaded into UTIM if UTIM underflows.

Be sure to use a 16-bit transfer instruction to access this register.

■ U-Timer Control Register (UTIMC: UTIMC0 to UTIMC2)



UTIMC controls the operation of the U-TIMER.

Be sure to use a byte transfer instruction to access this register.

[bit7] UCC1 (U-timer Count Control 1)

This bit controls the U-TIMER counting method.

UCC1	Operation
0	Normal operation, $\alpha = 2n+2$ [Initial value]
1	+1 mode $\alpha = 2n+3$

n: UTIMR set value

α : Cycle of the output clock for UART

The U-TIMER can set a normal $2(n+1)$ cycle clock, as well as an odd-numbered division for the UART.

Set UCC1 to "1" to generate a cycle of $2n+3$.

Examples:

- UTIMR=5, UCC1=0 \longrightarrow Generation cycle= $2n+2=12$ cycles
- UTIMR=25, UCC1=1 \longrightarrow Generation cycle= $2n+3=53$ cycles
- UTIMR=60, UCC1=0 \longrightarrow Generation cycle= $2n+2=122$ cycles

Set UCC1 to "0" to use the U-TIMER as the interval timer.

[bit6, bit5] (reserved)

[bit4] UTIE (U-Timer Interrupt Enable)

This bit is the interrupt enable bit for a U-TIMER underflow.

0: Interrupt disabled [Initial value]

1: Interrupt enabled

[bit3] UNDR (UNDeR flow flag)

This bit is a flag indicating that an underflow has occurred. If UTIE is "1", an underflow is generated when UNDR is set. The UNDR bit is cleared at reset or when "0" is written to it.

When reading by a read-modify-write instruction, "1" is always read. Writing "1" to the UNDR has no effect.

[bit2] CLKS (CLock Select)

This bit is the cascade specification bit for Channels 0 and 1 of the U-TIMER.

0: Uses a peripheral clock as the clock source (ϕ) [Initial value]

1: Uses an underflow signal of ch.0 as the U-TIMER source clock timing (f.f. shown in the block diagram)

CLKS is valid only for Channels 1 and 2. This bit must always be set to "0" for ch.0.

Note:

ϕ (Peripheral clock=CLKP) has a different cycle depending on the gear setting.

[bit1] UTST (U-Timer STart)

This bit is the U-TIMER operation enable bit.

0: Stop. Writing "0" during operation stops running of the U-TIMER. [Initial value]

1: Write. Writing "1" during operation keeps the operation of U-TIMER.

[bit0] UTCR (U-Timer Clear)

Writing "0" to UTCR clears the U-TIMER to "0000_H" (also clears the f.f. to "0").

The read value is always "1".

Note:

In the stop state, data is automatically reloaded when asserting (starting) the start bit UTST.

1. In the stop state, when asserting both the clear bit UTCR and the start bit UTST at the same time, the counter is cleared to "0" and an underflow is generated in the count-down immediately after the counter is cleared.
 2. During operation, the counter is cleared to "0" by asserting the clear bit UTCR. As a result, a short, whisker-like pulse may be output in the output waveform, and can cause the U-TIMER, located in higher side of UART and cascade mode, to malfunction. While the output clock is being used, do not clear it using the clear bit.
 3. In cascade mode, setting the lower side UTIMR (reload register) to "0" or "1" causes the count to be performed incorrectly.
 4. In the timer stop state, when asserting both bit1 (U-TIMER start bit: UTST) and bit0 (U-TIMER clear bit: UTCR) of the U-TIMER control register at the same time, bit3 (underflow flag: UNDR) of this register is set when the counter is loaded after it has been cleared. At this timing, the internal baud rate clock is set to "H" level.
 5. If the device attempts to set and clear an interrupt request flag at the same time, the flag is set by priority and the clear operation becomes ineffective.
 6. If you select not to use ch0 in cascade mode or use this module only as the timer function, always write "0" to bit2 (Reference clock selection bit: CLKS) of the U-TIMER control register. Additionally, change the setting of the "CLKS" bit when this module is in the stop state.
 7. If the device attempts to write to and reload the data into the U-TIMER reload register at the same time, old data is loaded into the counter. New data is loaded into the counter in the next reload timing.
 8. If the device attempts to timer clear and timer count/reload at the same time, the timer clear operation takes precedence.
 9. Set UTIMR reload value to "3" or higher when using CLK synchronous mode.
-

12.3 Description of Operation

This section explains calculation of U-TIMER baud rate and cascade mode.

■ Calculation of Baud Rate

The UART uses the underflow flip-flop (f.f. in the block diagram) of the corresponding U-TIMER (U-TIMER0 -> UART0, U-TIMER1 -> UART1, U-TIMER2 -> UART2) as the clock source for baud rates.

• Asynchronous (start-stop) mode

The UART uses the U-TIMER output divided by 16.

UCC1=0

$$\text{bps} = \frac{\phi}{(2n+2) \times 16}$$

UCC1=1

$$\text{bps} = \frac{\phi}{(2n+3) \times 16}$$

Maximum bps 33 MHz 515625 bps

n --- UTIMR (reload value)

ϕ --- Peripheral machine clock frequency
(varies depending on the gear)

• CLK synchronous mode

UCC1=0

$$\text{bps} = \frac{\phi}{(2n+2)}$$

UCC1=1

$$\text{bps} = \frac{\phi}{(2n+3)}$$

Maximum bps 33 MHz 4125000 bps

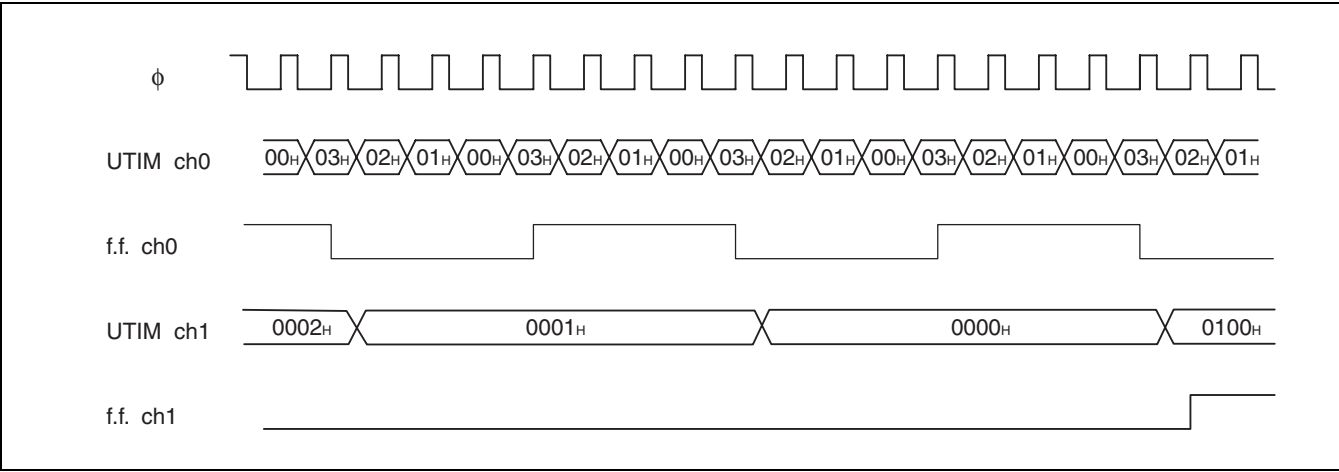
n --- UTIMR (reload value)
(set n=3 or higher number)

ϕ --- Peripheral machine clock frequency
(varies depending on the gear)

■ Cascade Mode

Channels 0 and 1 of the U-TIMER can be used in cascade mode.

Example: When UTIMR ch0 is set to "0003_H" and UTIMR ch1 is set to "0100_H"



CHAPTER 13

UART

This chapter describes the overview of the UART, the configuration and functions of registers, and UART operation.

13.1 Overview

13.2 Detail Description of Registers

13.3 Operation of UART

13.4 Example of Using the UART

13.5 Example of Setting Baud Rates and U-TIMER Reload Values

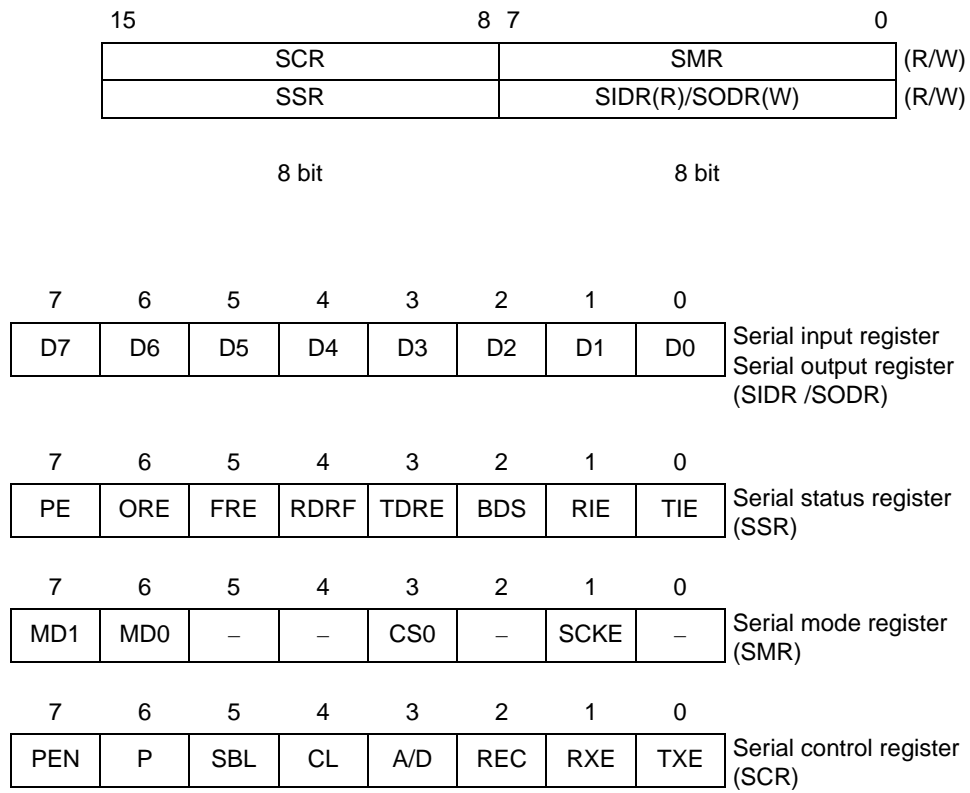
13.1 Overview

The UART is a serial I/O port used to perform asynchronous (start-stop synchronization) communication or CLK synchronous communication. The MB91260B series has three UART channels.

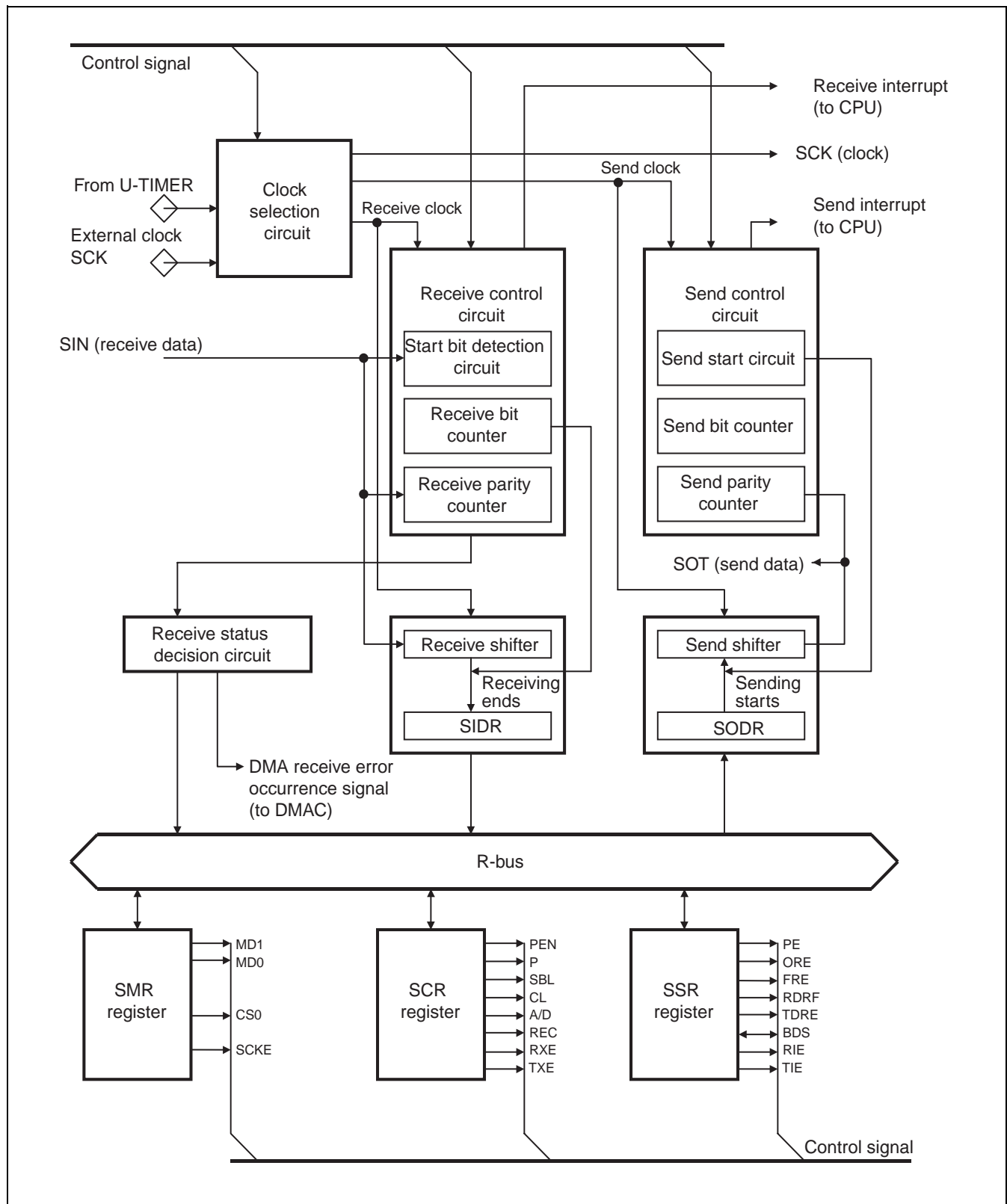
■ Features of UART

- Full-duplex double buffer
- Either asynchronous (start-stop synchronization) or CLK synchronous communication can be selected.
- Multiprocessor mode is supported.
- Fully programmable baud rate.
- An arbitrary baud rate can be set using a built-in timer (See "CHAPTER 12 U-TIMER (16-bit Timer for UART Baud Rate Generation)").
- An external clock can be used to set a baud rate.
- Error detection functions (parity, framing, overrun)
- The transfer signal is an NRZ code.
- DMA transfer can be started by an interruption.

■ Register List



■ Block Diagram



13.2 Detail Description of Registers

This section describes the configuration and functions of registers used by the UART.

■ SMR (Serial Mode Register)

SMR									
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ch0 000063 _H	MD1	MD0	–	–	CS0	–	SCKE	–	00--0-0 _B
ch1 00006B _H	R/W	R/W			W		R/W		
ch2 000073 _H									

The SMR specifies the UART operating mode. Set the operating mode while operation is stopped. Do not write to this register during operation.

[bit7, bit6] MD1, MD0 (MoDe select):

These bits select a UART operating mode.

Table 13.2-1 Settings for UART Operating Modes

Mode	MD1	MD0	Operating mode
0	0	0	Asynchronous (start-stop synchronization): normal mode [Initial value]
1	0	1	Asynchronous (start-stop synchronization): multiprocessor mode
2	1	0	CLK synchronous mode
–	1	1	Setting disabled

Note:

Mode 1, which is CLK asynchronous mode (multiprocessor), is used with more than one slave CPU connecting to one host CPU. This resource cannot identify the data format of received data, therefore, only the master in multiprocessor mode is supported.

Additionally, since the parity check function cannot be used, set PEN of the SCR register to "0".

[bit5, bit4] (reserved)

Always write "1" to these bits.

[bit3] CS0 (Clock Select)

This bit selects the UART operating clock.

0: Built-in timer (U-TIMER) [Initial value]

1: External clock

[bit2] (reserved)

Always write "0" to this bit.

[bit1] SCKE (SCLK Enable):

This bit specifies whether the SCK pin is used as a clock input pin or a clock output pin when communication is performed in CLK synchronous mode (Mode 2).

Set this bit to "0" in CLK asynchronous mode or external clock mode.

- 0: Serves as clock input pin. [Initial value]
- 1: Serves as clock output pin.

Note:

When using the SCK pin as a clock input pin, set the CS0 bit to "1" to select external clock mode.

[bit0] (Reserved):

Unused bit.

■ SCR (Serial Control Register)

SCR									
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ch0 000062 _H	PEN	P	SBL	CL	A/D	REC	RXE	TXE	00000100 _B
ch1 00006A _H	R/W	R/W	R/W	R/W	R/W	W	R/W	R/W	
ch2 000072 _H									

The SCR controls the transfer protocol that is used for serial communication.

[bit7] PEN (Parity Enable):

This bit specifies whether to add parity in serial communication when data communication is performed.

- 0: No parity [Initial value]
- 1: Parity

Note:

Parity can be added only in normal mode (Mode 0) of asynchronous (start-stop synchronization) communication mode. No parity can be added in multiprocessor mode (Mode 1) or CLK synchronous communication mode (Mode 2).

[bit6] P (Parity):

This bit specifies that even or odd parity is used to perform data communication.

- 0: Even parity [Initial value]
- 1: Odd parity

[bit5] SBL (Stop Bit Length):

This bit specifies the length of stop bit, which marks the end of a frame in asynchronous (start-stop synchronization) communication.

- 0: 1 stop bit [Initial value]
- 1: 2 stop bits

[bit4] CL (Character Length):

This bit specifies the data length of one frame that is sent or received.

0: 7-bit data [Initial value]

1: 8-bit data

Note:

7-bit data can be handled only in normal mode (Mode 0) of asynchronous (start-stop synchronization) communication mode. Use 8-bit data in multiprocessor mode (Mode 1) or CLK synchronous communication mode (Mode 2).

[bit3] A/D (Address/Data):

This bit specifies the data format of a frame that is sent or received in multiprocessor mode (Mode 1) of asynchronous (start-stop synchronization) communication mode.

0: Data frame [Initial value]

1: Address frame

[bit2] REC (Receiver Error Clear):

Write "0" to this bit to clear the error flags (PE, ORE, and FRE) in the SSR register.

Writing "1" to this bit has no effect. "1" is always read from this bit.

[bit1] RXE (Receiver Enable):

This bit controls the UART receive operation.

0: Disables receive operation. [Initial value]

1: Enables receive operation.

Note:

If a receive operation is disabled while it is in progress (while data is being input to the receive shift register), reception of the frame is completed and the receive operation is stopped when the received data is stored in the receive data buffer register (SIDR).

[bit0] TXE (Transmitter Enable):

This bit controls the UART send operation.

0: Disables send operation. [Initial value]

1: Enables send operation.

Note:

If a send operation is disabled while it is in progress (while data is being output from the transmission register), sending is stopped when no more send data is stored in the send data buffer register (SODR).

■ SDR: SDR0 to SDR2 (Serial Input Data Register)

SODR: SODR0 to SODR2 (Serial Output Data Register)

SIDR									
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ch0 000061 _H	D7	D6	D5	D4	D3	D2	D1	D0	Undefined
ch1 000069 _H	R	R	R	R	R	R	R	R	
ch2 000071 _H									

SODR									
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ch0 000061 _H	D7	D6	D5	D4	D3	D2	D1	D0	Undefined
ch1 000069 _H	W	W	W	W	W	W	W	W	
ch2 000071 _H									

These registers are data buffer registers for sending and receiving.

If the data length is seven bits, bit7 (D7) of SIDR and SODR contains invalid data. Accessing SIDR and SODR when BDS=1 switches the high-order and low-order data on the bus. As a result, it appears that bit0 (D0) is ignored.

Write to the SODR register only while the TDRE bit of the SSR register is "1".

Note:

Writing to the register with this address means writing to the SODR register. Reading from the register with this address means reading from the SIDR register.

■ SSR: SSR0 to SSR2 (Serial Status Register)

SSR									
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ch0 000060 _H	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	00001000 _B
ch1 000068 _H	R	R	R	R	R	R/W	R/W	R/W	
ch2 000070 _H									

The SSR is configured from flags that indicate the operating status of the UART.

[bit7] PE (Parity Error):

This bit, which is an interrupt request flag, is set when a parity error occurs during reception.

To clear the flag when it has been set, write "0" to the REC bit (bit10) of the SCR register.

If the PE bit is set, the SIDR data becomes invalid.

0: No parity error has occurred. [Initial value]

1: A parity error has occurred.

[bit6] ORE (Over Run Error):

This bit, which is an interrupt request flag, is set when an overrun error occurs during reception.

To clear the flag when it has been set, write "0" to the REC bit of the SCR register.

If the ORE bit is set, the SIDR data becomes invalid.

0: No overrun error has occurred. [Initial value]

1: An overrun error has occurred.

[bit5] FRE (FRaming Error):

This bit, which is an interrupt request flag, is set when a framing error occurs during reception.

To clear the flag when it has been set, write "0" to the REC bit of the SCR register.

If the FRE bit is set, the SDR data becomes invalid.

0: No framing error has occurred. [Initial value]

1: A framing error has occurred.

Notes:

- Switch the internal and external baud rate clocks using bit3 of the serial mode register only while the UART is stopped, since the switching takes effect immediately after writing.
- Bit3 of the serial mode register is write-only.

[bit4] RDRF (Receiver Data Register Full):

This bit, which is an interrupt request flag, indicates that the SDR register has receive data.

This bit is set when receive data is loaded into the SDR register. It is automatically cleared when the data is read from the SDR register.

0: No receive data exists. [Initial value]

1: Receive data exists.

[bit3] TDRE (Transmitter Data Register Empty):

This bit, which is an interrupt request flag, indicates whether send data can be written to SDR register.

This bit is cleared when send data is written to the SDR register. It is set again when the written data is loaded into the send shifter and begins to be transferred, indicating that the next send data can be written.

0: Disables writing of send data.

1: Enables writing of send data. [Initial value]

[bit2] BDS (Bit Direction Select)

This bit selects the transfer direction.

0: Sends starting from the least significant bit (LSB). [Initial value]

1: Sends starting from the most significant bit (MSB).

Note:

Because the high-order and low-order data are switched when the serial data register is written or read, the data will become invalid if the bit is rewritten after data is written to the SDR register.

If the SDR register and BDS are rewritten at the same time using halfwords (16 bits), data will be written to the SDR register based on the BDS value before rewriting.

[bit1] RIE (Receiver Interrupt Enable):

This bit controls a receive interrupt.

0: Disables receive interrupt. [Initial value]

1: Enables receive interrupt.

Note:

Receive interrupt sources include errors due to PE, ORE, and FRE as well as normal receive due to RDRF.

[bit0] TIE (Transmitter Interrupt Enable):

This bit controls a send interrupt.

0: Disables send interrupt. [Initial value]

1: Enables send interrupt.

Note:

Send interrupt sources include send requests due to TDRE.

13.3 Operation of UART

The UART has two operating modes: asynchronous (start-stop synchronization) mode and clock synchronous mode.

Asynchronous mode consists of normal and multiprocessor mode.

This section describes the operation of these operating modes.

■ Operating Modes

The UART has the operating modes shown in Table 13.3-1 and can switch the mode by setting a value in the SMR and SCR registers.

Table 13.3-1 UART Operating Modes

Mode	Parity	Data length	Operating mode	Stop bit length
0	Yes/No	7	Asynchronous (start-stop synchronization): normal mode	1 bit or 2 bits
	Yes/No	8		
1	No	8+1	Asynchronous (start-stop synchronization): multiprocessor mode	
2	No	8	CLK synchronous mode	No

The stop bit length in asynchronous (start-stop synchronization) mode can be specified only for a send operation. The stop bit length is always one bit for a receive operation. Since operation is possible only in the above modes, do not make any other setting.

■ Selecting a Clock for the UART

● Internal Timer

If you select the U-TIMER by setting CS0 to "0", the baud rate is determined according to the reload value set for the U-TIMER. At this time, you can calculate the baud rate as follows:

Asynchronous (start-stop synchronization) $\phi / (16 \times \beta)$

CLK synchronous ϕ / β

ϕ : Peripheral machine clock frequency

β : Cycle defined for the U-TIMER ($2n+2$ or $2n+3$, n is the reload value [$n \geq 3$])

In asynchronous (start-stop synchronization) mode, data can be transferred in the range from -1% to +1% of the specified baud rate.

● External Clock

If you select an external clock by setting CS0 to "1", the baud rate is as follows (the frequency of the external clock is assumed to be ϕ):

Asynchronous (start-stop synchronization): $\phi / 16$

CLK synchronous : ϕ

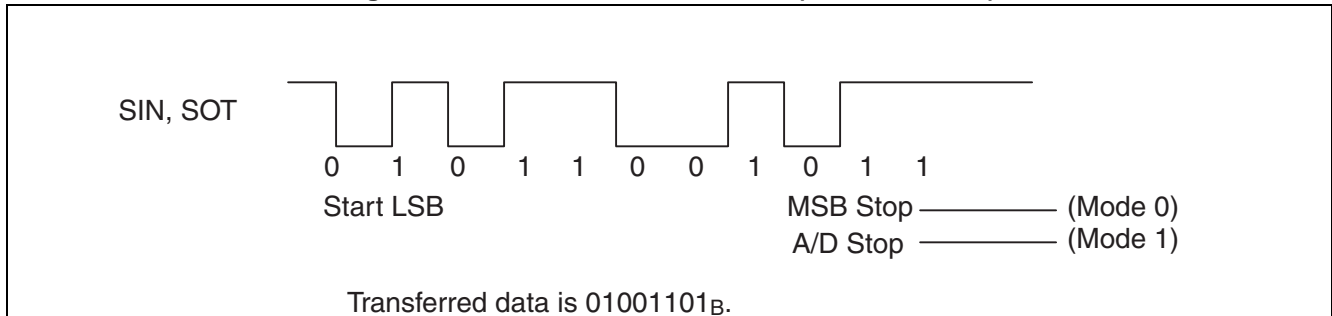
However, the maximum value for ϕ is 3.125 MHz.

■ Asynchronous (Start-stop Synchronization) Mode

● Transfer Data Format

UART handles only data in the NRZ (Non Return to Zero) format. Figure 13.3-1 shows the data format.

Figure 13.3-1 Transfer Data Format (Modes 0 and 1)



As shown in Figure 13.3-1, the transfer data always starts with the start bit ("L" level data), continues to the data bit length specified in LSB First, and ends with a stop bit ("H" level data). If an external clock is selected, be sure to input a clock at all times.

The data length can be set to 7 or 8 bits in normal mode (Mode 0), but must be set to 8 bits in multiprocessor mode (Mode 1). In multiprocessor mode, parity cannot be added. Instead, the A/D bit is always added.

● Receive Operation

If the RXE bit (bit1) of the SCR register is set to "1", a receive operation is always in progress.

If a start bit appears on the receive line, one-frame data is received according to the data format specified in the SCR register. After reception of one frame is completed, if an error occurs, the error flag is set and then the RDRF flag (bit4 of the SSR register) is set. If, at this time, the RIE bit (bit1) of the same SSR register is set to "1", a receive interrupt is generated to the CPU. Check the flags of the SSR register and if normal reception has occurred, read the S IDR register, or if an error has occurred, perform the necessary processing.

The RDRF flag is cleared when the S IDR register is read.

● Send Operation

If the TDRE flag (bit3) of the SSR register is set to "1", send data is written to the SODR register. If, at this time, the TXE bit (bit0) of the SCR register is set to "1", transmission occurs.

The TDRE flag is set again when the data set in the SODR register is loaded into the send shift register and the transfer starts, indicating that the next send data can be set. If, at this time, the TIE bit (bit0) of the same SSR register is set to "1", a send interrupt is generated to the CPU to request to set send data in the SODR register.

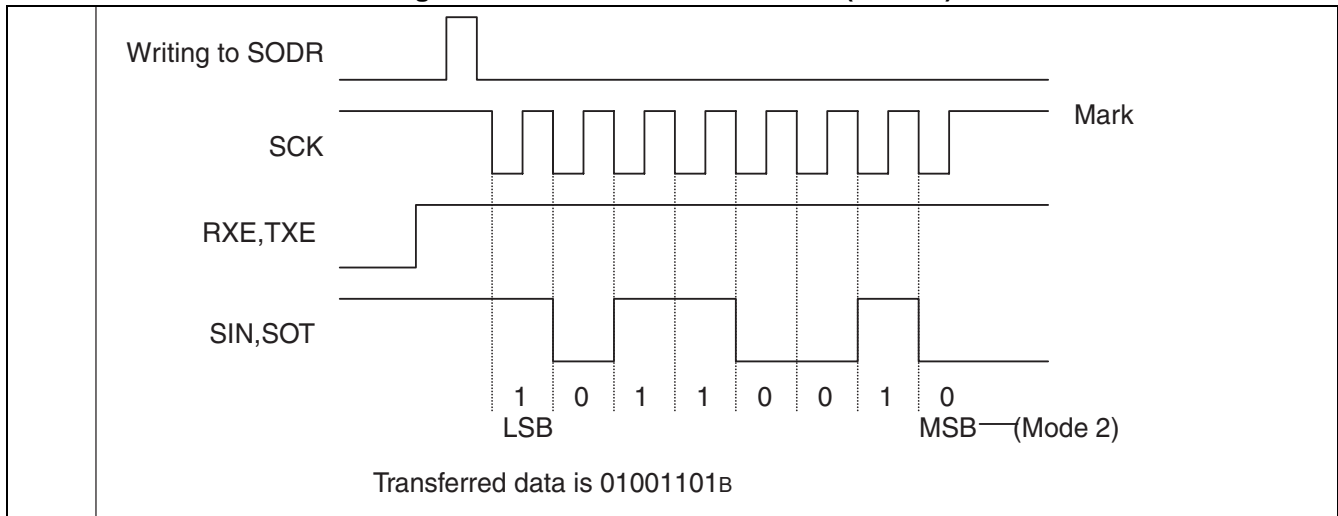
The TDRE flag is cleared if data is set in the SODR register.

■ CLK Synchronous Mode

● Transfer Data Format

The UART handles only data in the NRZ (Non Return to Zero) format. Figure 13.3-2 shows the relationship between send and receive clocks and data.

Figure 13.3-2 Transfer Data Format (Mode 2)



When the internal clock (U-TIMER) has been selected, a data receive synchronous clock is automatically generated as soon as data is received. While an external clock has been selected, after checking the existence of data in the send data buffer SODR register of the send side UART (TDRE flag is "0"), an accurate supply of the clock for one byte is necessary. Before sending starts and after it ends, be sure to set the mark level.

The data length is 8 bits only, and no parity can be added. Only overrun errors are detected because there is no start or stop bit.

● Initialization

The following shows the setting values of the control registers required to use CLK synchronous mode.

(1) SMR register

MD1, MD0 : "10_B"

CS : Specifies the clock input

SCKE : Set to "1" for an internal timer and to "0" for an external clock

(2) SCR register

PEN : "0"

P, SBL, A/D : These bits are meaningless

CL : "1"

REC : "0" (to initialize the register)

RXE, TXE : At least one of the bits must be set to "1"

(3) SSR register

RIE : Set to 1 to enable interrupts and to "0" to disables interrupts

TIE : "0"

● Start of Communication

Write to the SODR register to start communication. If only reception is performed, dummy send data must be written to the SODR register.

● End of Communication

Check for the end of communication by making sure that the RDRF flag of the SSR register has changed to "1". Use the ORE bit of the SSR register to check that communication has been performed correctly.

■ Occurrence of Interrupts and Timing for Setting Flags

The UART has five flags and two interrupt sources.

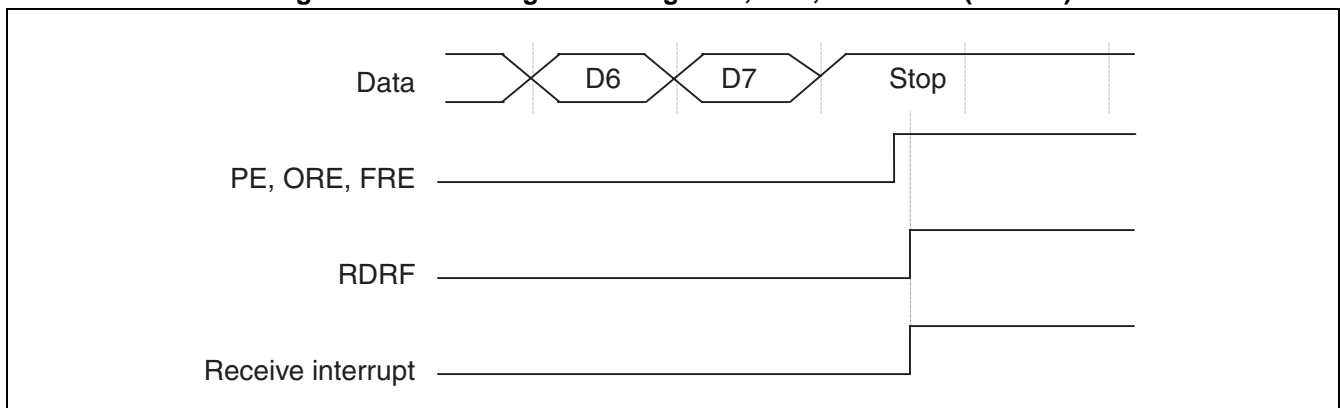
The five flags are PE, ORE, FRE, RDRF, and TDRE. PE means parity error, ORE means overrun error, and FRE means framing error. These flags are set when an error occurs during reception and are cleared when "0" is written to REC of the SCR register. RDRF is set when receive data is loaded into the SIDR register and is cleared when data is read from the SIDR register. However, Mode 1 does not provide a parity detection function and Mode 2 does not provide a parity detection function and a framing error detection function. TDRE is set when the SODR register is empty and writing to it is enabled, and is cleared when data is written to the SODR register.

There are two interrupt sources, one for receiving and the other for sending. During receiving, an interrupt is requested by PE, ORE, FRE, or RDRF. During sending, an interrupt is requested by TDRE. The following shows the timing for setting the interrupt flags in each of these modes.

● Receive Operation in Mode 0

The PE, ORE, FRE, and RDRF flags are set when the last stop bit is detected after a receive transfer is completed, and an interrupt request is generated to the CPU. The SIDR data is invalid while PE, ORE, and FRE are active.

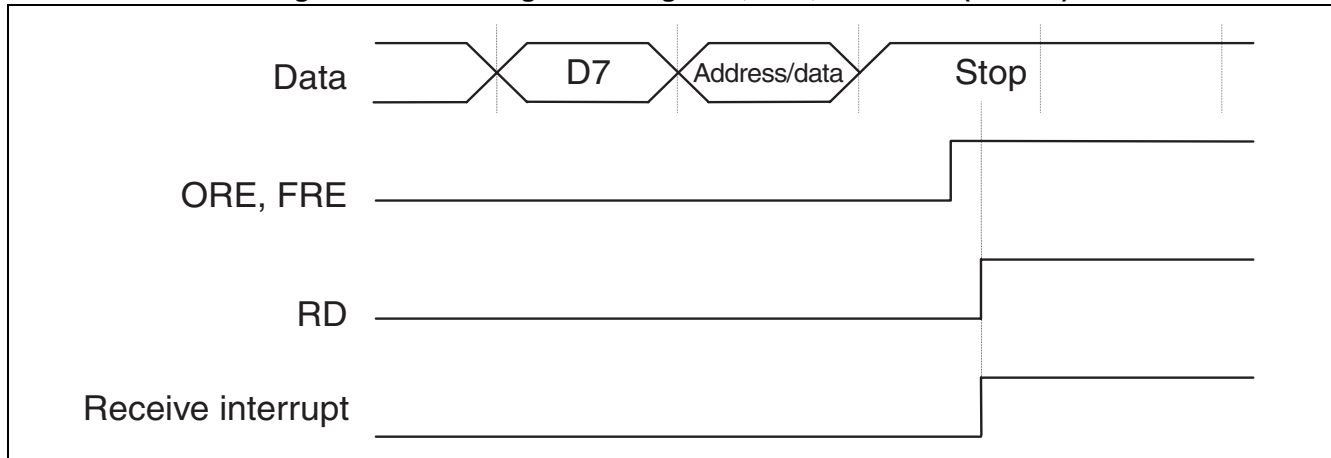
Figure 13.3-3 Timing for Setting ORE, FRE, and RDRF (Mode 0)



● Receive Operation in Mode 1

The ORE, FRE, and RDRF flags are set when the last stop bit is detected after a receive transfer is completed, and an interrupt request is generated to the CPU. The data indicating an address or data in the 9th bit becomes invalid because the receivable data length is 8 bits. The SIDR data is invalid while ORE and FRE are active.

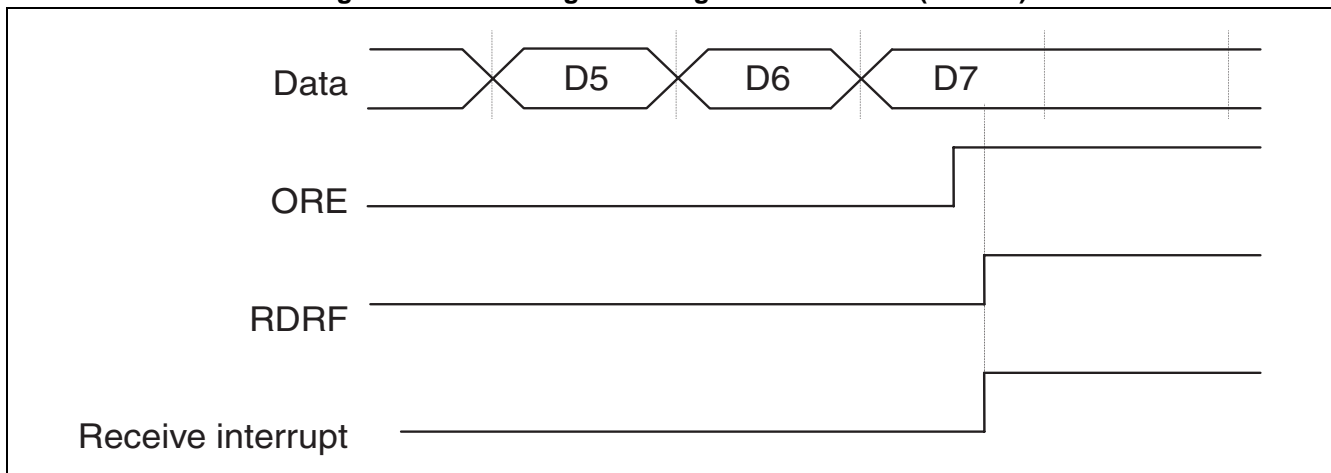
Figure 13.3-4 Timing for Setting ORE, FRE, and RDRF (Mode 1)



● Receive Operation in Mode 2

The ORE and RDRF flags are set when the last data (D7) is detected after the receive transfer is completed, and an interrupt request is generated to the CPU. The SIDR data is invalid while ORE is active.

Figure 13.3-5 Timing of Setting ORE and RDRF (Mode 2)



● Send operation in Modes 0, 1, and 2

TDRE is cleared when data is written to the SODR register. This bit is set when data is transferred to the internal shift register and the next data is ready to be written, and an interrupt request is generated to the CPU. If "0" is written to TXE of the SCR register (as well as RXE in Mode 2) during a send operation, TDRE of the SSR register is set to "1", disabling the UART send operation after the transmission shifter stops. The device sends data written to the SODR register before transmission stops after "0" is written to the TXE of the SCR register (as well as RXE in Mode 2) during the send operation.

Figure 13.3-6 Timing for Setting TDRE (Modes 0 and 1)

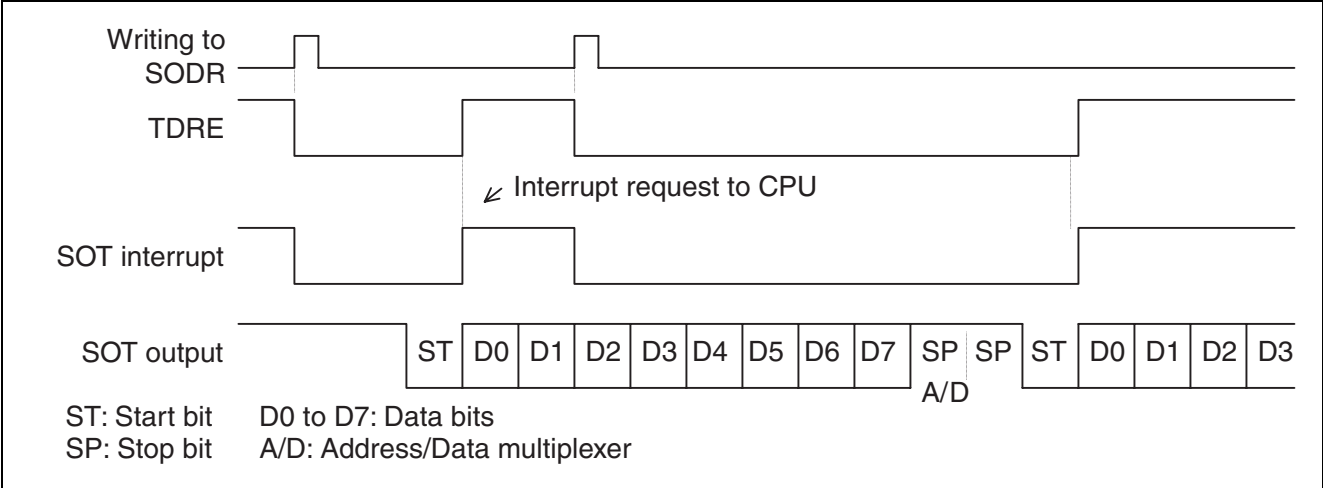
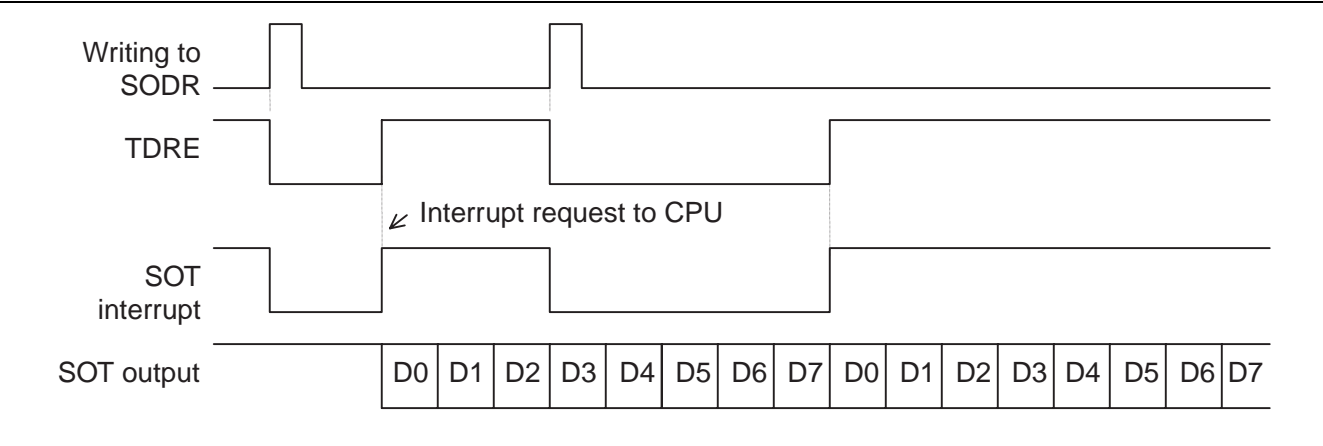


Figure 13.3-7 Timing for Setting TDRE (Mode 2)



■ Precautions on Usage

Writing to the SODR register starts communication. Even for receive only, dummy send data must be written to the SODR register.

Set the operating mode while operation is stopped. Data sent and received while the operating mode is being set is not ensured.

13.4 Example of Using the UART

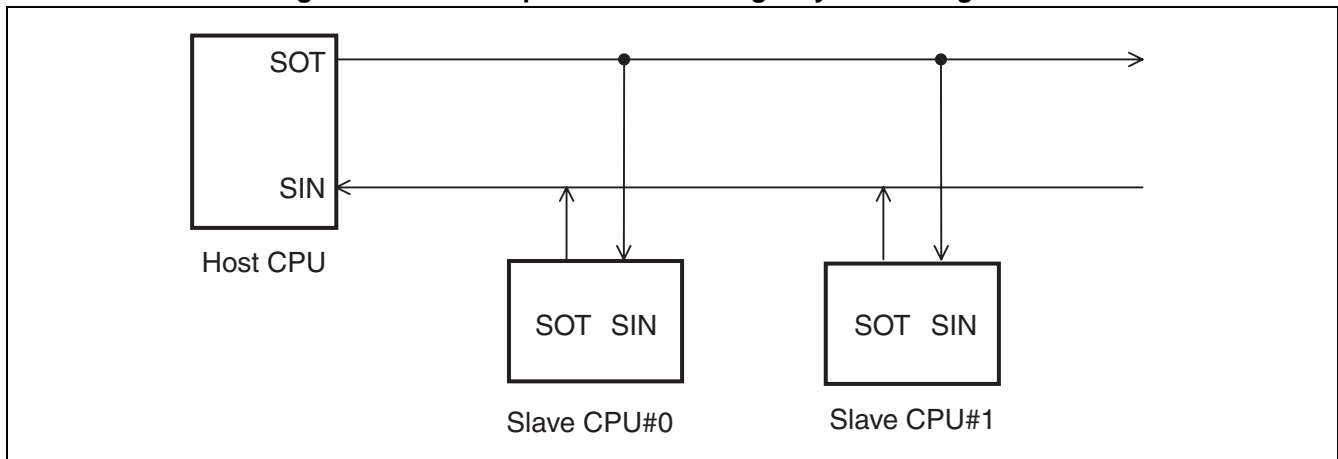
This section provides an example of using the UART. Mode 1 is used if more than one slave CPU is connected to one host CPU.

■ Example of Using the UART

Figure 13.4-1 shows an example of constructing a system using mode 1.

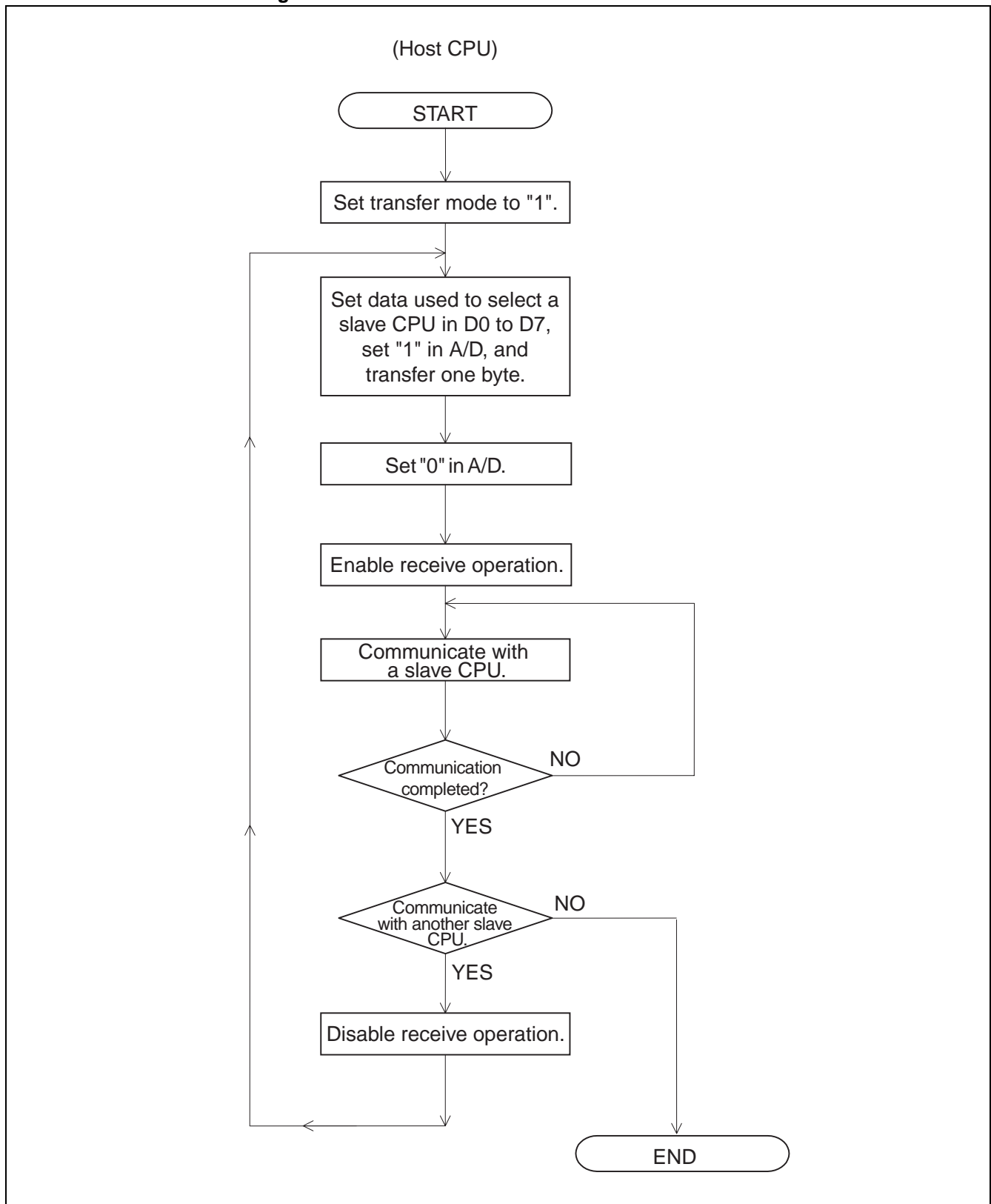
This resource supports only a communications interface of the host side.

Figure 13.4-1 Example of Constructing a System Using Mode 1



Communication starts when the host CPU transfers address data. Address data is data used when A/D of the SCR register is set to "1". This data is used to select a destination slave CPU, enabling communication with the host CPU. Normal data is data used when A/D of the SCR register is set to "0". Figure 13.4-2 shows the flowchart.

In this mode, set the PEN bit of the SCR register to "0" since the parity check function cannot be used.

Figure 13.4-2 Communication Flowchart in Mode 1

13.5 Example of Setting Baud Rates and U-TIMER Reload Values

This section provides an example of setting baud rates and U-TIMER reload values.

A frequency in the tables represents a peripheral machine clock frequency. UCC1 is a value to be set in the UCC1 bit of the UTIMC register of the U-TIMER.

"—" in the tables indicates the setting value cannot be used due to the error exceeding $\pm 1\%$.

Table 13.5-1 Setting Values in Asynchronous (Start-Stop Synchronization) Mode

Baud Rate	ms	33 MHz	20 MHz	16.5 MHz	10 MHz
1200	833.33	858 (UCC1=0)	520 (UCC1=0)	428 (UCC1=1)	259 (UCC1=1)
2400	416.67	428 (UCC1=1)	259 (UCC1=1)	214 (UCC1=0)	129 (UCC1=0)
4800	208.33	214 (UCC1=0)	129 (UCC1=0)	106 (UCC1=0)	64 (UCC1=0)
9600	104.17	106 (UCC1=1)	64 (UCC1=0)	52 (UCC1=1)	31 (UCC1=1)
19200	52.08	52 (UCC1=1)	31 (UCC1=1)	26 (UCC1=0)	—
38400	26.04	26 (UCC1=0)	—	12 (UCC1=1)	—
57600	17.36	17 (UCC1=0)	—	8 (UCC1=0)	—
10400	96.15	98 (UCC1=0)	59 (UCC1=0)	48 (UCC1=1)	29 (UCC1=0)
31250	32.00	32 (UCC1=0)	19 (UCC1=0)	15 (UCC1=1)	9 (UCC1=0)
62500	16.00	15 (UCC1=1)	9 (UCC1=0)	—	4 (UCC1=0)

Table 13.5-2 Setting Values in CLK Synchronous Mode

Baud Rate	ms	33 MHz	20 MHz	16.5 MHz	10 MHz
250K	4.00	65 (UCC1=0)	39 (UCC1=0)	32 (UCC1=0)	19 (UCC1=0)
500K	2.00	32 (UCC1=0)	19 (UCC1=0)	15 (UCC1=1)	9 (UCC1=0)
1M	1.00	15 (UCC1=1)	9 (UCC1=0)	7 (UCC1=0) *	4 (UCC1=0)

*: Having error exceeding $\pm 1\%$

CHAPTER 14

8/10-BIT A/D CONVERTER

This chapter describes the overview of the 8/10-bit A/D converter, the configuration and functions of registers, and the operation of the 8/10-bit A/D converter.

- 14.1 Overview
- 14.2 Configuration
- 14.3 Pin
- 14.4 Registers
- 14.5 Interrupt
- 14.6 Operation Explanation
- 14.7 A/D Conversion Data Protection Function
- 14.8 Precautions on Using

14.1 Overview

The 8/10-bit A/D converter has a feature to convert analog input voltage to a 10 or 8-bit digital value, using the RC successive comparison/conversion method. The input signal can be selected from 8 channels of analog input pin, and three types of conversions can be activated: software, internal clock, and external pin trigger.

■ Function of 8/10-bit A/D Converter

There is an A/D conversion feature to convert analog voltage (input voltage) input to the analog input pins into digital values. This feature has the following benefits.

- The conversion time is a minimum of 1.2 μ s (including the sampling time using the 33 MHz machine clock).
- The conversion method used is the RC successive comparison conversion method with sample hold circuit.
- A resolution of 10- or 8-bit can be selected.
- The analog input pin can be selected from the 8 channels using the program.
- After A/D conversion is completed, an interrupt request can be generated.
- No data are lost in the interrupt enable status since the conversion data protection function works even in the continuous conversion.
- One of the following conversion activation causes can be selected: software, 16-bit reload timer 1 or multifunctional timer (rising edge), or external pin triggers (falling edge).

There are three conversion modes.

Table 14.1-1 Conversion Modes of 8/10-bit A/D Converter

Conversion Mode	Single Conversion Operation	Scanning Conversion Operation
Single conversion mode	The specified channel (one channel only) is converted for one time and terminated.	Continuous plural channels (maximum 8 channels can be specified) are converted for one time and terminated.
Continuous conversion mode	Repeatedly convert specified channel (1 channel only).	Repeatedly convert multiple channels (up to 8 can be specified) in succession.
Pause-conversion mode	The specified channel (one channel only) is converted for one time and suspended until the next start.	Continuous plural channels (maximum 8 channels can be specified) are converted. However, one channel is converted and suspended until the next start.

- This model has three units: Unit 0, which has 8 analog-input channels; and units 1 and 2, which have two analog-input channels.
- The unit-0 activation trigger is the OR of the reload timer 1 and multifunctional timer. Operate reload timer 1 when activation via the multifunctional timer is enabled to activate A/D unit 0 via both activation triggers.

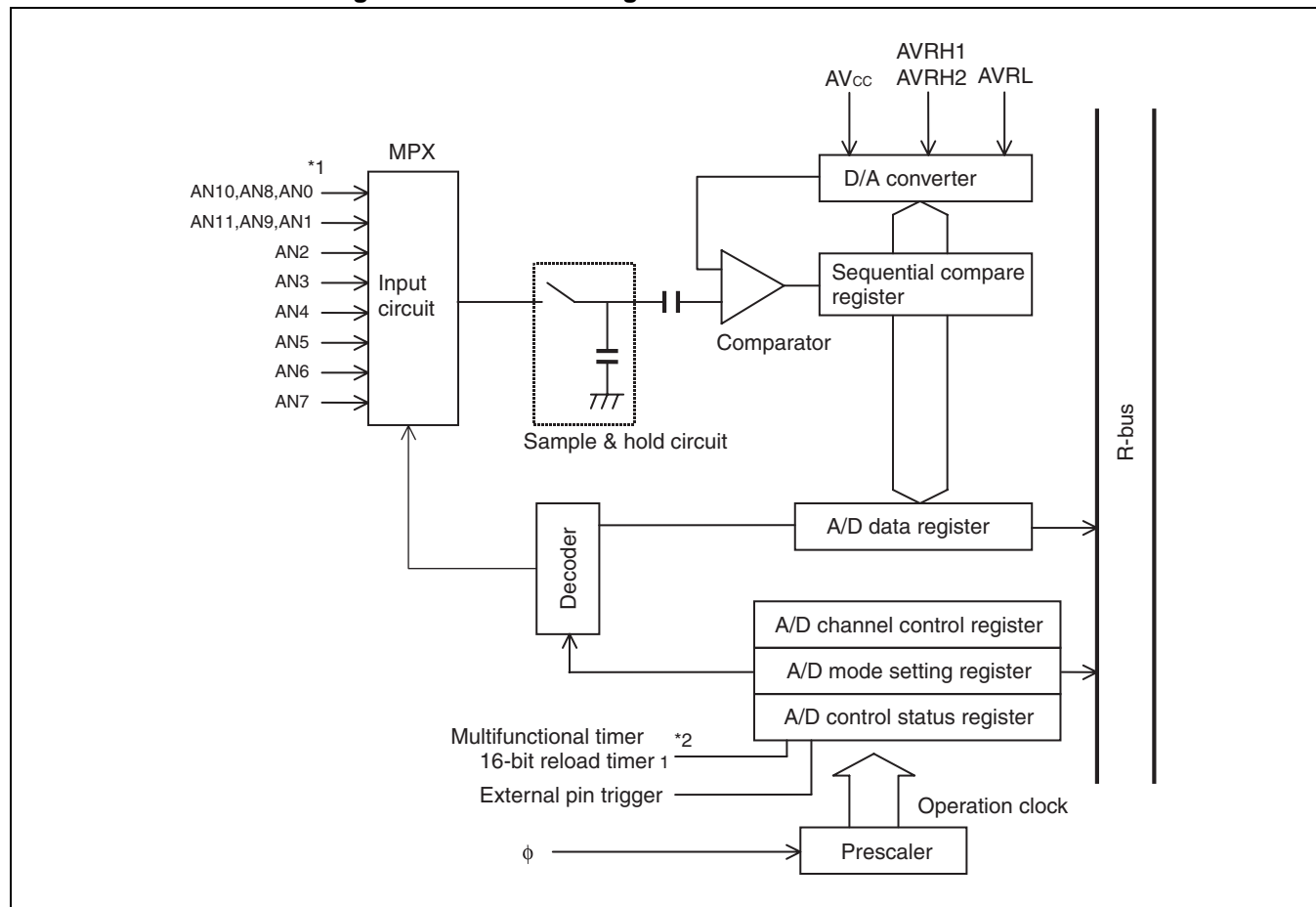
14.2 Configuration

The 8/10-bit A/D converter is made up of the following 11 blocks.

- A/D control status registers (ADCS)
- A/D channel control register (ADCH)
- A/D mode setting register (ADMD)
- A/D data register (ADCD)
- Clock selector (input clock selector for activation of A/D conversion)
- Decoder
- Analog channel selector
- Sample hold circuit
- D/A converter
- Comparator
- Control circuit

■ Block Diagram of 8/10-bit A/D Converter

Figure 14.2-1 Block Diagram of 8/10-bit A/D Converter



- *1: Unit 0: 8ch of AN0 to AN7
Unit 1: 2ch of AN8 and AN9
Unit 2: 2ch of AN10 and AN11
- *2: Unit 0: 16-bit reload timer 1 or multifunctional timer
(Activated via either the reload timer 1 or multifunctional timer.)
Unit 1: multifunctional timer
Unit 2: multifunctional timer

● A/D control status registers (ADCS)

Features are available to suspend and confirm conversion, enable/disable interrupt requests, confirm the status of interrupt requests, and select the A/D conversion resolution.

● A/D channel control register (ADCH)

There is a feature to select the A/D channel.

● A/D mode setting register (ADMD)

There is a feature to select a conversion mode and to set the A/D conversion compare time and sampling time.

● A/D data register (ADCD)

This register stores A/D conversion results.

● Clock selector

This is an A/D conversion activation clock selector. 16-bit reload timer channel 1 output, multifunctional timer, and external pin trigger can be selected as the activation clock.

- Unit 0 is 16-bit reload timer channel 1 output.
- Units 1 and 2 can be activated via multifunctional timer.

● Decoder

The A/D channel control register (ADCH) ANE0 to ANE2 and ANS0 to ANS2 bit settings are a circuit to select the analog input pin to use.

● Analog channel selector

This circuit selects the pin to be used from the 8 analog input pins.

- Unit 0 has 8 analog inputs.
- Unit 1 and 2 have 2 analog inputs.

● Sample hold circuit

This circuit holds the input voltage selected by the analog channel selector. The input voltage can be converted without affected by the input voltage fluctuation in the A/D conversion (in the comparison) by holding the sample of input voltage immediately after starting the A/D conversion.

- D/A converter

The reference voltage is generated to compare the held sample of input voltage.

- Comparator

This compares the input voltage for which sample hold is performed, with the output voltage of the D/A converter to determine which is the greater of the two.

- Control circuit

The signal from the comparator (higher or lower) determines the A/D conversion value. When the A/D conversion is terminated, the conversion result is stored in the A/D data register (ADCD) and the interrupt request is generated.

14.3 Pin

Pins of 8/10-bit A/D converter and block diagram of pin are shown.

■ Pins of 8/10-bit A/D Converter

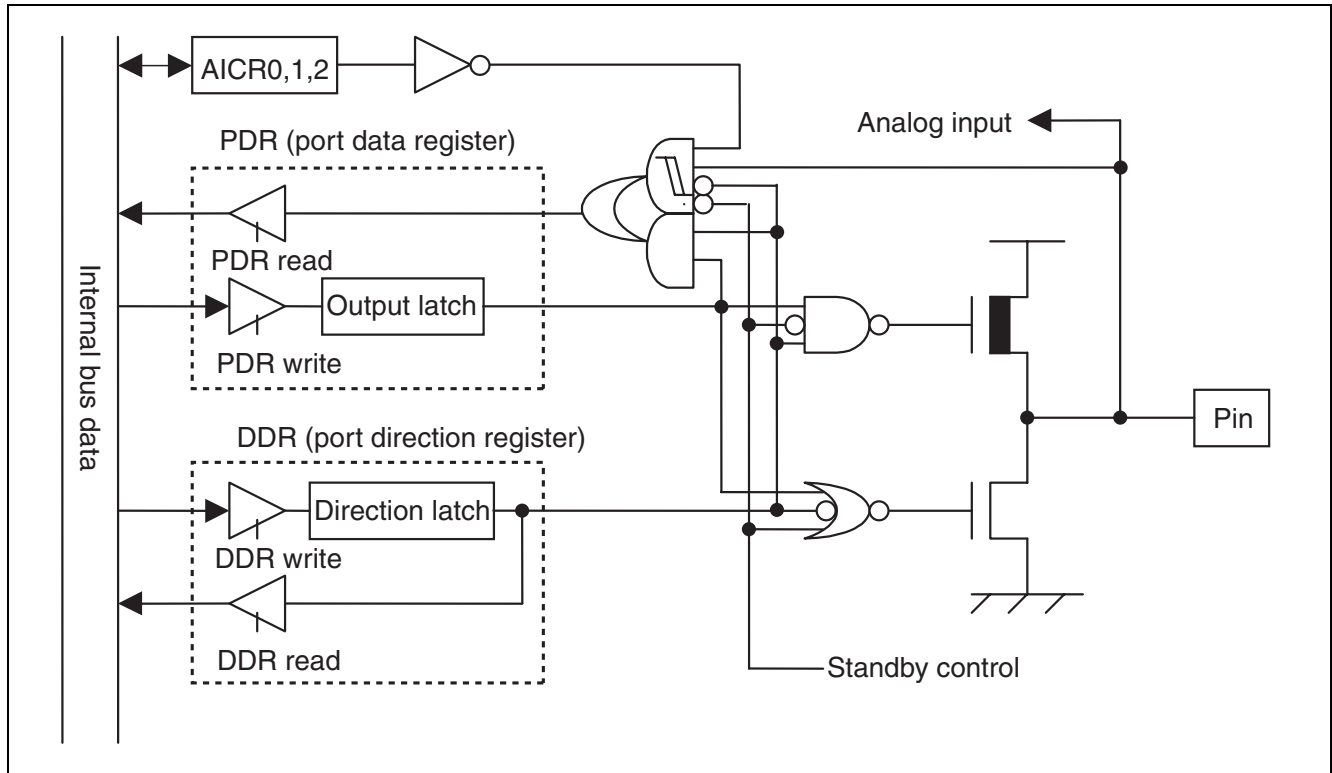
The A/D converter pin serves dual use as a general-purpose port. Table 14.3-1 shows pin functions, I/O type, and settings when using the 8/10-bit A/D converter.

Table 14.3-1 Pins of 8/10-bit A/D Converter

Function	Pin Name	Pin Function	I/O Type	Pull-up Setting	Stand-by Control	I/O Port Setting for Using Pin						
Channel 0	PC0/AN0	Port C I/O/ analog input	CMOS output/ CMOS input or analog input	None	Yes	Input setting of port C (DDRC: bit0 to bit7=0) Set to analog input (AICR0: bit0 to bit7=1)						
Channel 1	PC1/AN1											
Channel 2	PC2/AN2											
Channel 3	PC3/AN3											
Channel 4	PC4/AN4											
Channel 5	PC5/AN5											
Channel 6	PC6/AN6											
Channel 7	PC7/AN7											
Channel 8	PD0/AN8	Port D I/O/ analog input							Input setting of port D (DDRD: bit0, bit1=0) Set to analog input (AICR1: bit0, bit1=1)			
Channel 9	PD1/AN9											
Channel 10	PE0/AN10	Port E I/O/ analog input										Input setting of port E (DDRE: bit0, bit1=0) Set to analog input (AICR2: bit0, bit1=1)
Channel 11	PE1/AN11											

■ Block Diagram of 8/10-bit A/D Converter Pin

Figure 14.3-1 Block Diagram of AN0 to AN11 Pins



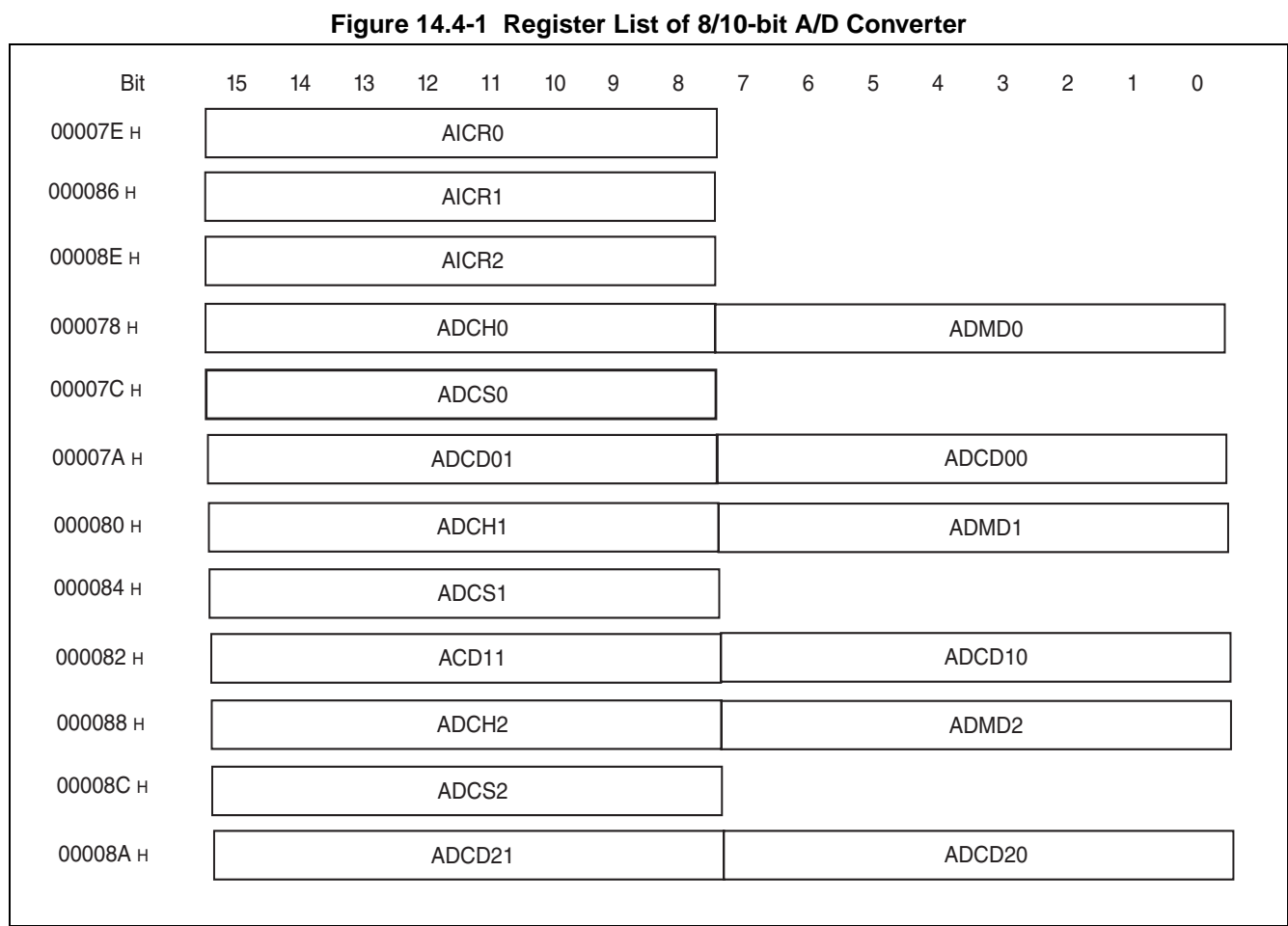
Notes:

- Set the DDR register bits corresponding to the pins to be used as input ports to "0" and apply pull-up resistance to the external pins. Also, set the bits corresponding to the AICR register to "0".
- Set the AICR register bit corresponding to the pin to be used as the analog input pin to "1". When this is done, the value "0" will be read from the PDR register.

14.4 Registers

Register list of 8/10-bit A/D converter is shown.

■ Register List of 8/10-bit A/D Converter



14.4.1 A/D Channel Control Register (ADCH)

The A/D channel control register has a feature to select the A/D conversion channel.

■ A/D Channel Control Register (ADCH: ADCH0 to ADCH2)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
Ch0 000078H	-	-	ANS2	ANS1	ANS0	ANE2	ANE1	ANE0
Ch1 000080H	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Ch2 000088H	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Read/write →	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial value →	(X)	(X)	(0)	(0)	(0)	(0)	(0)	(0)

ANE2	ANE1	ANE0	A/D conversion end channel selection bits
0	0	0	ch.0
0	0	1	ch.1
0	1	0	ch.2
0	1	1	ch.3
1	0	0	ch.4
1	0	1	ch.5
1	1	0	ch.6
1	1	1	ch.7

ANS2	ANS1	ANS0	A/D conversion start channel selection bits
0	0	0	ch.0
0	0	1	ch.1
0	1	0	ch.2
0	1	1	ch.3
1	0	0	ch.4
1	0	1	ch.5
1	1	0	ch.6
1	1	1	ch.7

R/W: Readable/Writable
 : Initial value

Notes:

- A/D unit 0 is ch.0 to ch.7 = AN0 to AN7
- A/D unit 1 is ch.0 and ch.1 = AN8 and AN9 (ch.2 to ch.7 are vacancy.)
- A/D unit 2 is ch.0 and ch.1 = AN10 and AN11 (ch.2 to ch.7 are vacancy.)
- Write 0 to ANE1, ANE2, ANS2, and ANS1 of A/D units 1 and 2.
- Be sure to set A/D units 1 and 2 so that ANS0 is less than or equal to ANE0.

Table 14.4-1 Functions of Each Bits in A/D Channel Control Register (ADCH)

Bit Name		Function
bit15 bit14	Unused bits	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect to operation.
bit13 to bit11	ANS2 to ANS0: A/D conversion start channel selection bits	<ul style="list-style-type: none"> These bits set the start channel of the A/D conversion and indicate the channel numbers under A/D conversion during conversion operation. When A/D conversion is activated, A/D conversion starts from the channels written to these bits. Channel numbers under the conversion can be read during the A/D conversion. The channel number converted immediately before can be read during the suspension in the pause conversion mode.
bit10 to bit8	ANE2 to ANE0: A/D conversion end channel selection bits	<ul style="list-style-type: none"> These bits set the end channel of the A/D conversion. A/D conversion is performed up to the specified channel. When the same channels with ANS2 to ANS0 are set, only those channels are converted. If continuous conversion mode or stop conversion mode is set, when the conversion up to the channels specified in these bits is completed, conversion returns to the start channel set in ANS2 to ANS0. If the setting of the start channel is greater than the end channel, conversion is performed from the start channel to AN7, then from AN0 to the end channel, after which the first conversion operation is completed. <p>Note:</p> <p>After setting the start channel to the A/D conversion start channel selection bit (ANS2 to ANS0), please set neither the A/D conversion mode selection bit (MD1, MD0) nor the A/D conversion end channel selection bit (ANE2 to ANE0) by the read-modify-write type instruction.</p> <p>The last conversion channel is read from the ANS2 to ANS0 bits until the A/D conversion operating starts. Therefore, when the MD1, MD0 bits and ANE2 to ANE0 bits are set by the read-modify-write type instruction after setting the start channel to the ANS2 to ANS0 bits, the value of ANE2 to ANE0 bits may be re-written.</p>

14.4.2 A/D Mode Setting Register (ADMD)

The A/D mode setting register has a feature to select a conversion mode and to set the A/D conversion compare time and sampling time.

■ A/D Mode Setting Register (ADMD: ADMD0 to ADMD2)

Address:

ch0 000079H
ch1 000081H
ch2 000089H

Read/Write →

Initial value →

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
MD1	MD0	STS1	STS0	CT1	CT0	ST1	ST0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
(0)	(0)	(0)	(0)	(1)	(1)	(1)	(1)

ST1	ST0	Sampling time setting bits
0	0	9 machine cycles (450 ns @ 20 MHz) *
0	1	15 machine cycles (450 ns @ 33 MHz) *
1	0	17 machine cycles (510 ns @ 33 MHz) *
1	1	28 machine cycles (840 ns @ 33 MHz) *

*: Set the machine cycle to at least 450ns.

CT1	CT0	Compare time setting bits
0	0	12 machine cycles (750 ns @ 16 MHz) *
0	1	18 machine cycles (720 ns @ 25 MHz) *
1	0	24 machine cycles (720 ns @ 33 MHz) *
1	1	48 machine cycles (1440 ns @ 33 MHz) *

*: Set the machine cycle to at least 720ns.

STS1	STS0	A/D start factor selection bits
0	0	Software start
0	1	External pin trigger (falling edge) or Software start
1	0	Multifunction timer * start (rising edge) or Software start
1	1	External pin trigger (falling edge) or Multifunction timer * start (rising edge) or Software start

*: In the case of unit 0, the timer is 16-bit reload timer 1 or multifunction timer.

MD1	MD0	A/D conversion mode selection bits
0	0	Single conversion mode 1 (restarting is enabled during operation)
0	1	Single conversion mode 2 (restarting is disabled during operation)
1	0	Continuous conversion mode (restarting is disabled during operation)
1	1	Stop conversion mode (restarting is disabled during operation)

R/W: Readable/Writable

Initial value

Table 14.4-2 Functions of Each Bit in A/D Mode Setting Register (ADMD) (1 / 2)

Bit Name	Function
bit7 bit6 MD1, MD0: A/D conversion mode selection bits	<ul style="list-style-type: none"> These bits are used to select the conversion mode during the A/D conversion operation. Two bit values of MD1 and MD0 allow the selection of either the single conversion mode 1, the single conversion mode 2, the continuous conversion mode, or the stop conversion mode. The meaning of each mode is as follows. <p>Single conversion mode 1: The continuous A/D conversion from the setting channels of ANS2 to ANS0 to the setting channels of ANE2 to ANE0 is performed only once. It is possible to reactivate while operating.</p> <p>Single conversion mode 2: The continuous A/D conversion from the setting channels of ANS2 to ANS0 to the setting channels of ANE2 to ANE0 is performed only once. It is not possible to reactivate while operating.</p> <p>Continuous conversion mode: Sequentially perform A/D conversion from the channel set in ANS2 to ANS0 to the channel set in ANE2 to ANE0, and repeat until forcibly stopped by means of the BUSY bit. It is not possible to reactivate while operating.</p> <p>Stop conversion mode: Perform A/D conversion from the channel set in ANS2 to ANS0 to the channel set in ANE2 to ANE0 one channel at a time, pausing between each, and repeat until forcibly stopped by means of the BUSY bit. It is not possible to reactivate while operating. The suspended conversion is restarted by the start factor occurrence selected by the STS1 and 0 bits.</p> <p>Notes:</p> <ul style="list-style-type: none"> Single, continuous, and stop conversion modes that cannot be re-started can be used to activate all timers, external triggers, and software. Only rewrite these bits before conversion begins, with the A/D operation stopped. When A/D conversion mode selection bit (MD1,MD0) is set to "00_B", restart can be operating during A/D conversion. Only software start (STS1,STS0="00_B") can be set in this mode. Please restart according to the following procedure. <ol style="list-style-type: none"> Clear the INT bit to 0. Write STRT bit to 1 and INT bit to 0 at the same time. <p>When A/D conversion mode selection bit (MD1,MD0) is set to "01_B", restart can not be operating during the A/D conversion.</p> <p>When restarting and termination of the A/D conversion occur at the same time, the A/D conversion is terminated without restarting. And, value of 300_H is stored in data register (ADCR1/ADCR0). Therefore, please use restart so that neither the A/D conversion restarting nor the terminating may occur at the same time.</p>
bit5 bit4 STS1, STS0: A/D start factor selection bits	<ul style="list-style-type: none"> The start factor of A/D conversion is selected. When the start factor is in the common use, the first start factor generation starts the operation. <p>Note:</p> <p>The activation trigger changes as soon as the bits are rewritten, so if you wish to rewrite them while A/D conversion is ongoing, switch to a state where your target activation trigger does not exist.</p>

Table 14.4-2 Functions of Each Bit in A/D Mode Setting Register (ADMD) (2 / 2)

Bit Name		Function
bit3 bit2	CT1, CT0: Compare time setting bits	<p>These bits are used to select the comparison time at the A/D conversion. After analog input is loaded (after the sampling time has elapsed), then after the time specified in these bits has passed, the conversion results are checked, and stored in the A/D control status register (ADCD).</p> <p>Notes:</p> <ul style="list-style-type: none"> When CT1 and CT0 = 00_B, 10_B, or 11_B, the compare time must be set to at least 720 ns. When CT1 and CT0 = 01_B, the compare time must be set to at least 900 ns. If the time is not set to these values or greater, it may not be possible to obtain correct analog conversion value. Only rewrite these bits before conversion begins, with the A/D operation stopped.
bit1 bit0	ST1, ST0: Sampling time setting bits	<p>These bits are used to select the sampling time at the A/D conversion.</p> <p>When A/D is activated, analog input is retrieved for the time set in these bits.</p> <p>Notes:</p> <ul style="list-style-type: none"> If the sampling time is not set to 450 ns or more, it may not be possible to obtain correct analog conversion values. Only rewrite these bits before conversion begins, with the A/D operation stopped.

14.4.3 A/D Control Status Register (ADCS)

The A/D control status register has features to suspend and confirm conversion, enable/disable interrupt requests, confirm the status of interrupt requests, and select the A/D conversion resolution.

■ A/D Control Status Register (ADCS: ADCS0 to ADCS2)

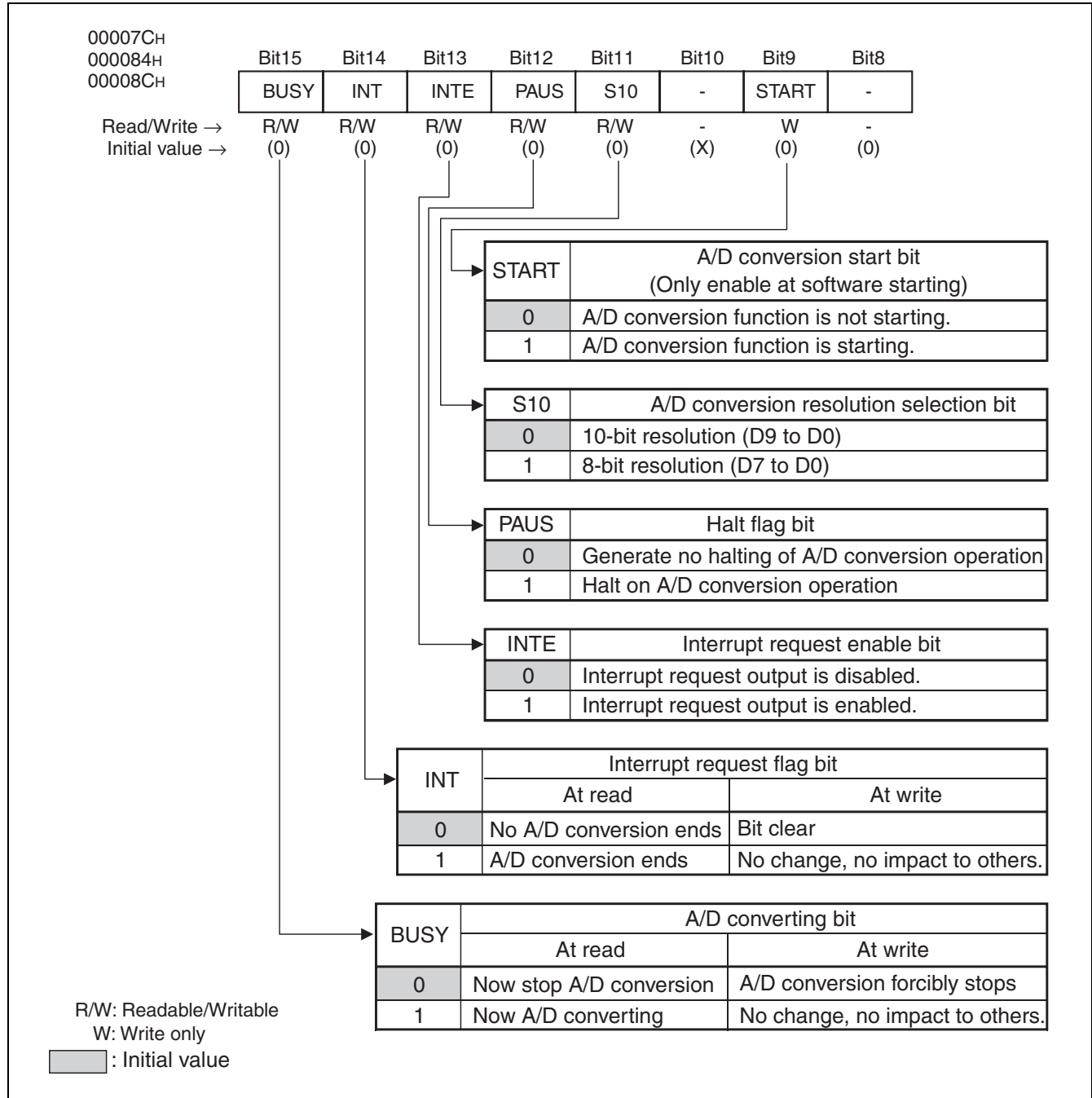


Table 14.4-3 Function of Each Bit in A/D Control Status Register (ADCS) (1 / 2)

Bit Name		Function
bit15	BUSY: A/D converting bit	<ul style="list-style-type: none"> Operational display bit of A/D converter When reading, if this bit is "0", it indicates that A/D conversion is stopped. If it is "1", it indicates that A/D conversion is ongoing. When writing, writing "0" to this bit forcibly stops A/D conversion. When "1" is written, the conversion is not changed and no others are affected. "1" is always read during read-modify-write. <p>Note:</p> <p>Do not perform a forced stop and software activation (BUSY = 0, START = 1) at the same time.</p>
bit14	INT: Interrupt request flag bit	<ul style="list-style-type: none"> This bit is set to "1" if data is set in the A/D data register through A/D conversion. If this bit and the interrupt request enable bit (ADCS: INTE) are "1", an interrupt request is generated. When writing, a "0" clears this bit, and with "1" no change is made, and there are no other effects. "1" is always read during read-modify-write. <p>Note:</p> <p>Clear this bit by writing "0" to it when A/D is stopped.</p>
bit13	INTE: Interrupt request enable bit	<ul style="list-style-type: none"> This bit is used to enable and disable the interrupt output to CPU. If this bit and the interrupt request flag bit (ADCS: INT) are "1", an interrupt request is generated.
bit12	PAUS: Halt flag bit	<ul style="list-style-type: none"> It is set to "1" when A/D conversion is suspended. This A/D converter has only one A/D data register. For this reason, when in continuous conversion mode, if the old conversion results are not completely read via the CPU, they will be overwritten by the next conversion results and lost. Consequently, when using continuous conversion mode, as a rule you should perform settings so that conversion results are sent to memory each time conversion is completed. However, such case can be assumed that the conversion data transfer is not completed before the next conversion in the multiple interrupts, etc. The function of this bit was created in consideration of this case. After conversion is completed, set this bit to "1" while sending the contents of the data register. During this time, A/D conversion will halt and will not store the next conversion data. "1" is always read during read-modify-write.
bit11	S10: Analog to digital conversion resolution selection bit	<ul style="list-style-type: none"> This bit is used to select the A/D conversion resolution. Write "0" to this bit to select 10-bit resolution. Write "1" to this bit to select 8-bit resolution. <p>Notes:</p> <ul style="list-style-type: none"> The data bit used are different depending on the resolution. Only rewrite this bit before conversion begins, with the A/D operation stopped.
bit10	Unused bit	<ul style="list-style-type: none"> The read value is indeterminate. Writing to this bit has no effect to operation.

Table 14.4-3 Function of Each Bit in A/D Control Status Register (ADCS) (2 / 2)

Bit Name		Function
bit9	START: A/D conversion start bit	<ul style="list-style-type: none"> • This bit is used to start the A/D conversion operation by the software. • Write "1" to this bit to activate A/D conversion. • In stop conversion mode, this bit will not function to re-start conversion. <p>Note: Do not perform a forced stop and software activation (BUSY = 0, START = 1) at the same time.</p>
bit8	Reserved bit	Be sure to write "0".

14.4.4 A/D Data Register (ADCD)

The A/D data register stores A/D conversion results.

■ A/D Data Register (ADCD: ADCD00, ADCD01, ADCD10, ADCD11, ADCD20, ADCD21)

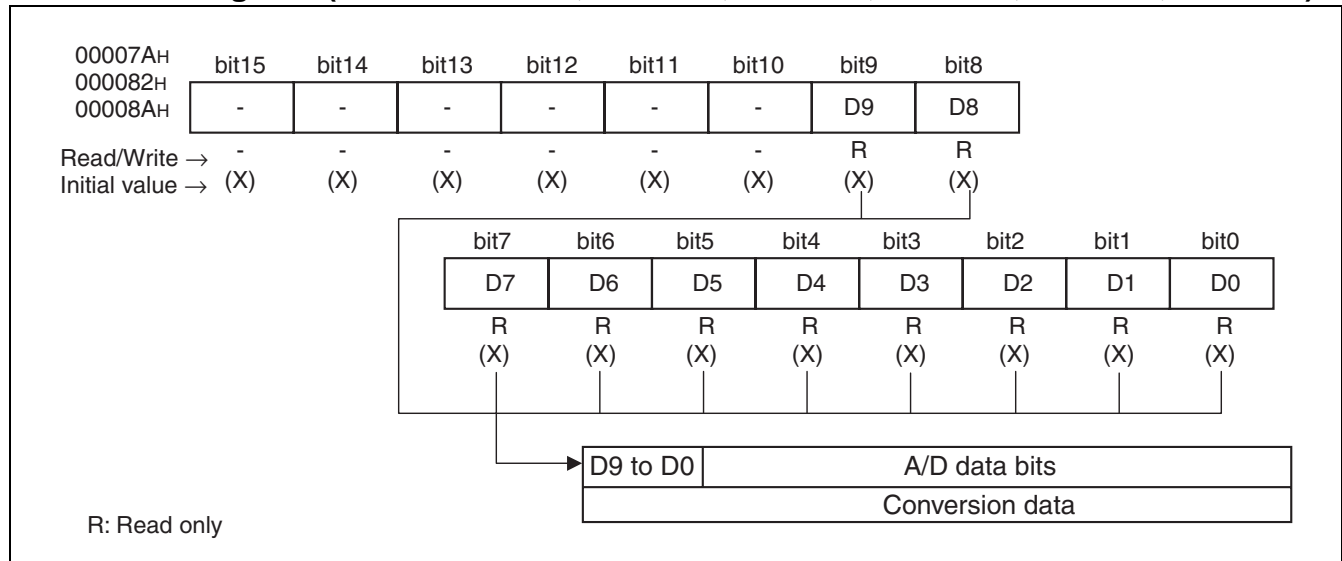


Table 14.4-4 Function of Each Bit in A/D Data Register (ADCD)

Bit Name		Function
bit15 to bit10	Unused bits	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect to operation.
bit9 to bit0	D9 to D0: Data bits	<ul style="list-style-type: none"> This register stores the results of A/D conversion, and is rewritten each time conversion is completed. The final conversion value is stored usually. The initial value of this register is indeterminate. <p>Notes:</p> <ul style="list-style-type: none"> The conversion data protection function is provided. Do not write data to this bit while A/D conversion is ongoing. When 8-bit resolution is selected from D9 and D8, "0" is read.

14.4.5 Analog Input Control Register (AICR)

The analog input control register controls analog input.

■ Analog Input Control Register (AICR: AICR0 to AICR2)

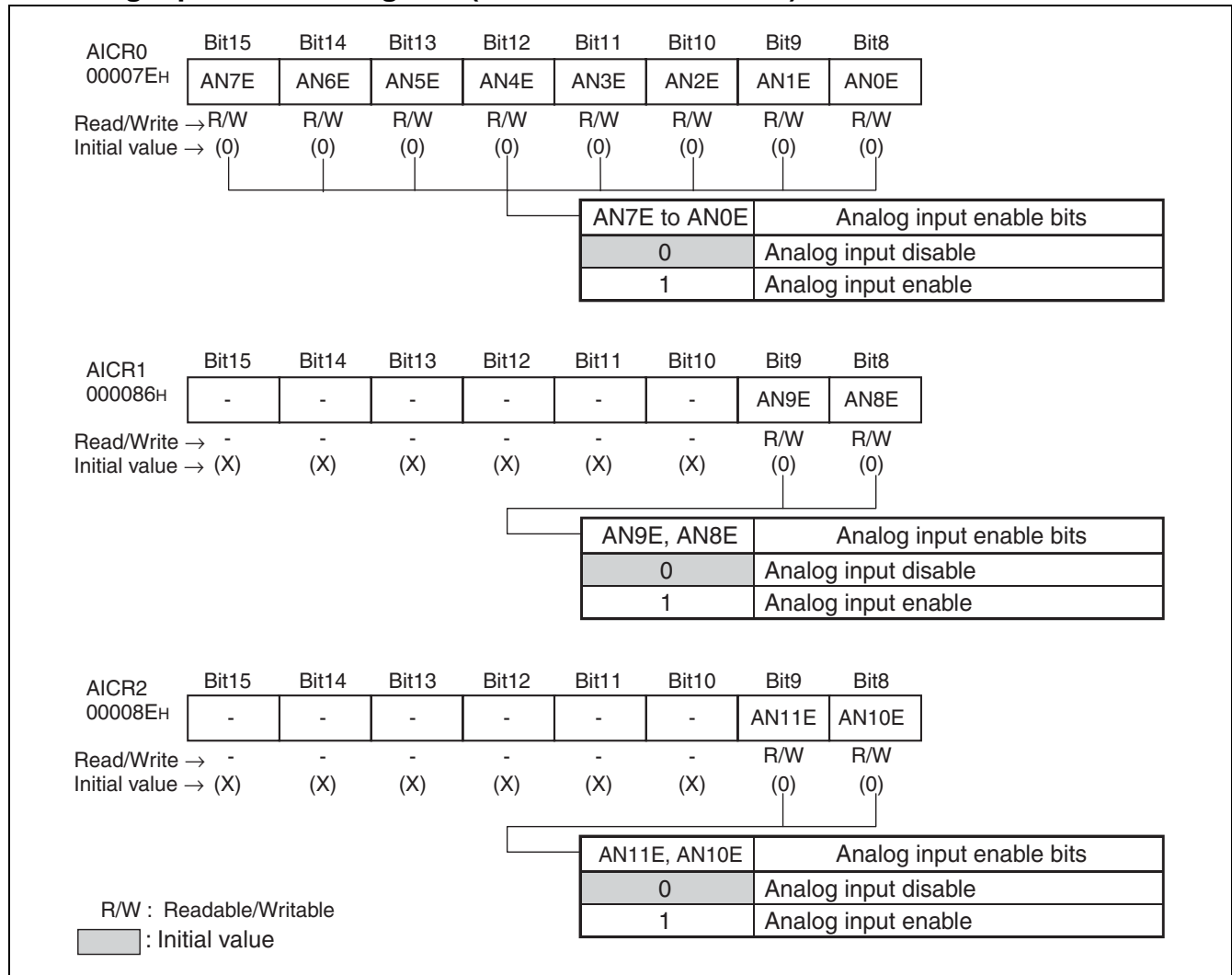


Table 14.4-5 Functions of Each Bit in Analog Input Control Register (AICR)

Bit Name		Function
(AICR0) bit15 to bit8 (AICR1, AICR2) bit9, bit8	AN7E to AN0E, AN9E, AN8E, AN11E, AN10E: Analog input enable bits	<ul style="list-style-type: none"> When this bit is "0", analog input is disabled. When this bit is "1", analog input is enabled. Set the AICR register bit corresponding to the pin to be used as the analog input pin to "1". When this is done, the value "0" will be read from the PDR register.
(AICR1, AICR2) bit15 to bit10	Unused bits	<ul style="list-style-type: none"> The read value is indeterminate. Writing to these bits have no effect to operation.

14.5 Interrupt

The 8/10-bit A/D converter can generate interrupt requests during A/D conversion by setting data in the A/D data register.

■ Interrupt of 8/10-bit A/D Converter

See Table 14.5-1 for the interrupt control bits and interrupt cause of the 8/10-bit A/D converter.

Table 14.5-1 Interrupt Control Bits and Interrupt Cause of 8/10-bit A/D Converter

	8/10-bit A/D converter
Interrupt request flag bit	ADCS: INT
Interrupt request enable bit	ADCS: INTE
Interrupt cause	Writing of A/D conversion result to A/D data register

When A/D conversion activates, and A/D conversion results are set in the A/D data register (ADCD), the INT bit of the A/D control status register (ADCS) is set to "1". At this time, if interrupt requests are enabled (ADCS: INTE = 1), an interrupt request is outputted to the interrupt controller.

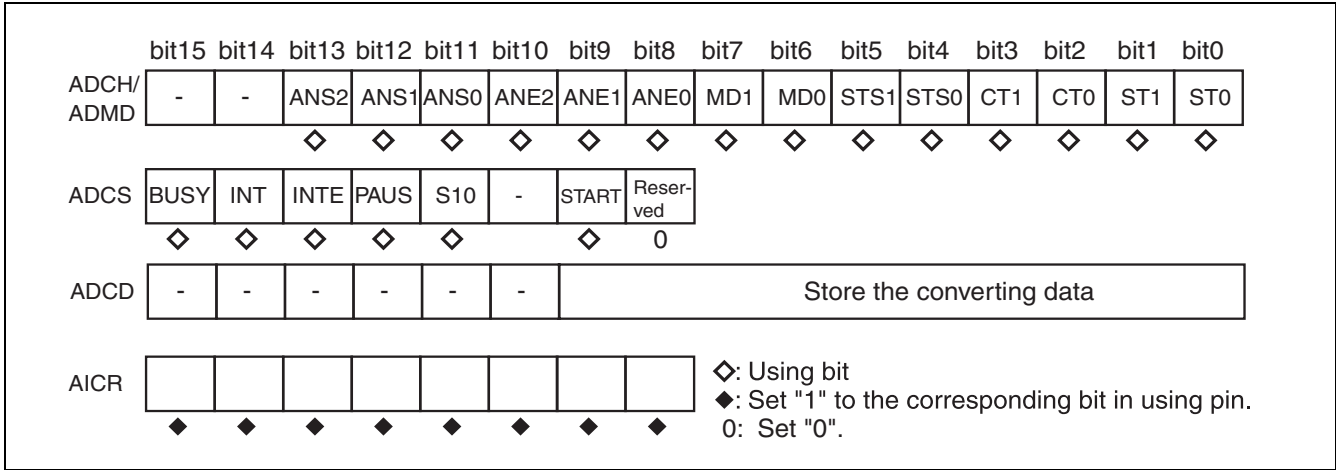
14.6 Operation Explanation

Three mode types, the single conversion, continuous conversion, and stop conversion modes are available for the 8/10-bit A/D converter. The operation explanation in each mode is done.

■ Operation of Single Conversion Mode

In single conversion mode, it sequentially converts the analog input on the channels which have been set by the ANS (ANS0 to ANS2) bit and ANE (ANE0 to ANE2) bit, and when it reaches to the end channel set in ANE (ANE0 to ANE2) bit, it stops the A/D conversion. If the start channel and end channel is the same (ANS=ANE), only one channel specified in the ANS (ANS0 to ANS2) bit will be converted. The settings in Figure 14.6-1 are required in order to operate in single-conversion mode.

Figure 14.6-1 Setting for Single Conversion Mode



Reference:

The example of conversion order in single conversion mode is shown in following.

When ANS=000_B, ANE=011_B: AN0 → AN1 → AN2 → AN3 → end

When ANS=110_B, ANE=010_B: AN6 → AN7 → AN0 → AN1 → AN2 → end

When ANS=011_B, ANE=011_B: AN3 → end

Notes:

- A/D unit 1 uses the 2 channels AN8 and AN9, and A/D unit 2 uses the 2 channels AN10 and AN11.
- Always write "0" to ANE1, ANE2, ANS2, and ANS1 of A/D units 1 and 2.
- Be sure to set A/D units 1 and 2 so that ANS0 is less than or equal to ANE0.
- When A/D conversion mode selection bit (MD1,MD0) is set to "00_B", restart can be operating during A/D conversion. Only software start (STS1,STS0="00_B") can be set in this mode. Please restart according to the following procedure.
 1. Clear the INT bit to "0".
 2. Write STRT bit to "1" and INT bit to "0" at the same time.

When A/D conversion mode selection bit (MD1,MD0) is set to "01_B", restart can not be operating during the A/D conversion.

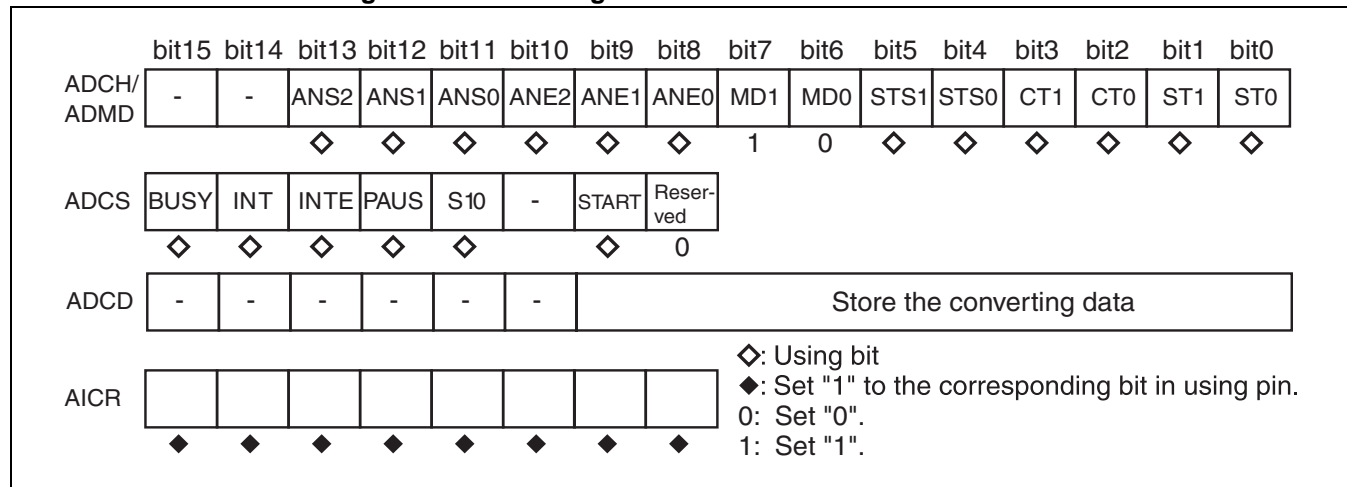
When restarting and termination of the A/D conversion occur at the same time, the A/D conversion is terminated without restarting. And, value of "300_H" is stored in data register (ADCR1/ADCR0).

Therefore, please use restart so that neither the A/D conversion restarting nor the terminating may occur at the same time.

■ Operation of Continuous Conversion Mode

In the continuous conversion mode, the analog inputs set by the ANS and ANE bits are sequentially converted, the analog input set by the ANS bit is resumed at the end of conversion of the end channel set by the ANE bit, and the A/D conversion operation is continued. If the start channel and end channel are identical (ANS=ANE), conversion loops on the channel specified by ANS only. The settings shown in Figure 14.6-2 are required in order to operate in continuous conversion mode.

Figure 14.6-2 Setting at Continuous Conversion Mode



References:

The example of conversion order in continuous conversion mode is shown in following.

- When $ANS=000_B$ and $ANE=011_B$, conversion iterates over AN0, AN1, AN2, AN3, and AN0, in that order.
 - When $ANS=110_B$ and $ANE=010_B$, conversion iterates over AN6, AN7, AN0, AN1, AN2 and AN6, in that order.
 - When $ANS=011_B$ and $ANE=011_B$, conversion iterates over AN3 and AN3, in that order.
-

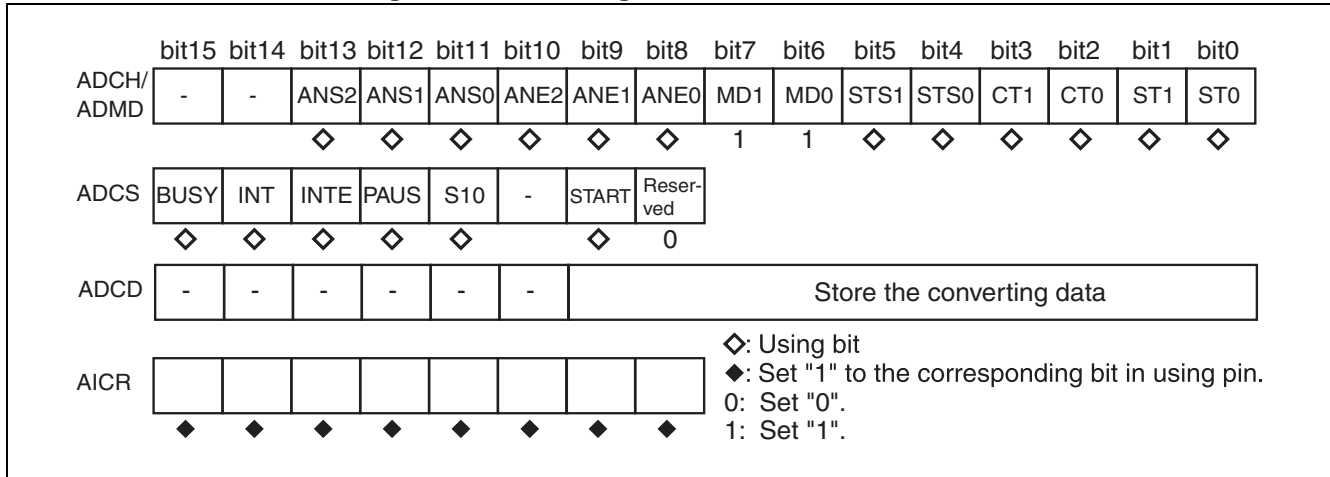
Notes:

- A/D unit 1 uses the 2 channels AN8 and AN9, and A/D unit 2 uses the 2 channels AN10 and AN11.
 - Always write "0" to ANE1, ANE2, ANS2, and ANS1 of A/D units 1 and 2.
 - Be sure to set A/D units 1 and 2 so that ANS0 is less than or equal to ANE0.
-

■ Operation of Pause-conversion Mode

In the stop conversion mode, the analog input set by the ANS and ANE bits is converted by being suspended for every channel, the analog input set by the ANS bit is resumed at the end of conversion of the end channel set by the ANE bit, and the operation of A/D conversion and suspension is continued. If the start channel and end channel are identical (ANS=ANE), conversion loops on the channel specified by the ANS bits only. When the conversion is restarted during the suspension, the start factor specified by the STS1 and 0 bits is generated. The settings in Figure 14.6-3 are required in order to operate in stop conversion mode.

Figure 14.6-3 Setting at Pause-conversion Mode



References:

The example of conversion order in stop conversion mode is shown in following.

- When ANS=000_B and ANE=011_B:
AN0 → Pause → AN1 → Pause → AN2 → Pause → AN0 → Repeat
- When ANS=110_B and ANE=001_B:
AN6 → Pause → AN7 → Pause → AN0 → Pause → AN1 → Pause → AN6 → Repeat
- When ANS=011_B, ANE=011_B:
AN3 → Pause → AN3 → Pause → Repeat

Notes:

- A/D unit 1 uses the 2 channels AN8 and AN9, and A/D unit 2 uses the 2 channels AN10 and AN11.
- Always write 0 to ANE1, ANE2, ANS2, and ANS1 of A/D units 1 and 2.
- Be sure to set A/D units 1 and 2 so that ANS0 is less than or equal to ANE0.

14.7 A/D Conversion Data Protection Function

When the A/D conversion is executed in the interrupt enable status, the conversion data protection function works.

■ A/D Conversion Data Protection Function

The A/D converter only has one data register for storing conversion data. For this reason, when performing A/D conversion, after conversion is completed, the data stored in the data register is rewritten. Therefore, when the converted data transfer to the memory is delayed, a part of previous data may be lost. To get around this, when interrupts are enabled (INTE = 1), the data-protection feature works as described below.

When conversion data is stored in the A/D data register (ADCD), the INT bit of the A/D control status register (ADCS) is set to "1". While the INT bit is "1", conversion data will not be stored to ADCD after the next conversion ends. The PAUS bit is set, and A/D conversion becomes suspended. While suspended, the value immediately prior is retained. In order to cancel the suspend, clear the INT bit. After the suspended status is cleared, the conversion data that had been maintained is stored in ADCD, and the next operation is performed.

Notes:

- The converted data protection function operates only in the interrupt enabled (ADCS: INTE=1) status.
 - When the conversion is restarted during the suspension, the waiting data are destroyed.
-

14.8 Precautions on Using

This section describes precautions on using 8/10-bit A/D converter.

■ Precautions on Using 8/10-bit A/D Converter

● Analog input pin

The A/D input pin does double duty as a port I/O pin. The port-direction register (DDR) and analog input enable register (AICR) are switched and used. For the pins used as analog input, set the bits corresponding to DDR to 0 and port to input, then set AICR register analog input mode (AICR x = 1). Lock the input gate on the port side. When the intermediate level signal is inputted in the port input mode (AICRx = 0), the input leak current flows through the gate.

● Cautions for use of internal timers

To activate the A/D converter by an internal timer, set the STS1 and STS0 bits of the A/D control status register (ADMD). When doing so, set the internal timer input value to the inactive side (for internal timer, this is "L"). If you set it to the active side, the timer may start to operate as soon as you write to the ADMD register.

● Turning-on sequence of power supply to A/D converter and analog inputs

Make sure to apply to the A/D converter power source (AV_{CC} , AVRH 0 to AVRH 2) and apply analog input (AN0 to AN11) after or at the same time as applying digital power source (V_{CC}). When cutting off the power, cut off the digital power source (V_{CC}) after or at the same time as cutting off the A/D converter power source and analog input.

● Power voltage of A/D converter

In order to prevent latch-ups, make sure that the A/D converter power source (AV_{CC}) does not exceed the voltage of the digital power source (V_{CC}).

● A/D unit 1 and 2 settings

A/D unit 1 uses the 2 channels AN8 and AN9, and A/D unit 2 uses the 2 channels AN10 and AN11. For this reason, when using A/D units 1 and 2, always write "0" to ANE1, ANE2, ANS2, and ANS1 of the A/D channel control register (ADCH). Also be sure to set ANS0 to less than or equal to ANE0.

● Restart of A/D converter

When A/D conversion mode selection bit (MD1,MD0) is set to "00_B", restart can be operating during A/D conversion. Only software start (STS1,STS0="00_B") can be set in this mode. Please restart according to the following procedure.

1. Clear the INT bit to 0.
2. Write STRT bit to 1 and INT bit to 0 at the same time.

When A/D conversion mode selection bit (MD1,MD0) is set to "01_B", restart can not be operating during the A/D conversion.

When restarting and termination of the A/D conversion occur at the same time, the A/D conversion is terminated without restarting. And, value of "300_H" is stored in data register (ADCR1/ADCR0). Therefore, please use restart so that neither the A/D conversion restarting nor the terminating may occur at the same time.

CHAPTER 15

MULTIPLICATION AND ADDITION CALCULATOR

This chapter explains the overview of the multiply and accumulate circuit, the configuration and functions of registers, and the macro (the definition and each instruction) of the multiply and accumulate circuit.

- 15.1 Overview
- 15.2 Register Description
- 15.3 Operation Explanation
- 15.4 Instruction Detail Explanation

15.1 Overview

This section shows the features, the register list, the block diagram of multiplication and addition calculator.

■ Features

- High-speed multiplication and addition calculation (1 system clock cycle)
- Data format : 16-bit fixed-point decimal ($16 \times 16 + 40$ -bit)
- Instruction area : 256×16 -bit
- Data area : 64×16 -bit $\times 2$ sets
- Rounding processing available
- Saturation processing available
- Additional item number : 64 items (Max)
- Instruction : MAC instruction, STR instruction, JMP instruction
- Delay processing : Possible to transfer in 64×16 -bit freely
- Fixed-point decimal format : Selectable from Q8 to Q15
- Program execution control : Selectable from eight commands
- Variable monitor : Monitor calculation results of up to 8×16 bits without stopping program operation.

■ Register List

	15	8	7	0	
Address: 39E _H	Reserved area				Access disabled
Address: 3A0 _H	DSP-PC (Program counter)			DSP-CSR (Control/Status)	Read/Write
Address: 3A2 _H	DSP-LY (Delayed register) Upper			DSP-LY (Delayed register) Lower	Read/Write
Address: 3A4 _H	DSP-OT0 (Output queue 0) Upper			DSP-OT0 (Output queue 0) Lower	Read
Address: 3A6 _H	DSP-OT1 (Output queue 1) Upper			DSP-OT1 (Output queue 1) Lower	Read
Address: 3A8 _H	DSP-OT2 (Output queue 2) Upper			DSP-OT2 (Output queue 2) Lower	Read
Address: 3AA _H	DSP-OT3 (Output queue 3) Upper			DSP-OT3 (Output queue 3) Lower	Read
Address: 3AC _H	Reserved area			Reserved area	Access disabled
Address: 3AE _H	Reserved area			Reserved area	Access disabled
Address: 3B0 _H	DSP-OT4 (Output queue 4) Upper			DSP-OT4 (Output queue 4) Lower	Read
Address: 3B2 _H	DSP-OT5 (Output queue 5) Upper			DSP-OT5 (Output queue 5) Lower	Read
Address: 3B4 _H	DSP-OT6 (Output queue 6) Upper			DSP-OT6 (Output queue 6) Lower	Read
Address: 3B6 _H	DSP-OT7 (Output queue 7) Upper			DSP-OT7 (Output queue 7) Lower	Read
Address:				Multiplication and addition macro	Address
C000 _H	X-RAM (coefficient RAM) ... 64 × 16 bit			00 _H	Read/Write
:				:	
C07E _H				3F _H	
Address:				Multiplication and addition macro	Address
C080 _H	Y-RAM (coefficient RAM) ... 64 × 16 bit			00 _H	Read/Write
:				:	
C0FE _H				3F _H	
Address:				Multiplication and addition macro	Address
C100 _H	I-RAM (coefficient RAM) ... 256 × 16 bit			00 _H	Read/Write
:				:	
C2FE _H				FF _H	
Notes:	·Be sure to write to the above register/RAM area from the CPU using the halfword (or word) transfer instruction to even-numbered address. ·Byte write and read to RAM area by the transfer command of CPU can be used. However, when RAM area is used as a multiplication and addition calculator, please use the write and read by the halfword (or half) transfer command to the even address number.				

■ Block Diagram

Figure 15.1-1 Block Diagram of Multiplication and Addition Calculator

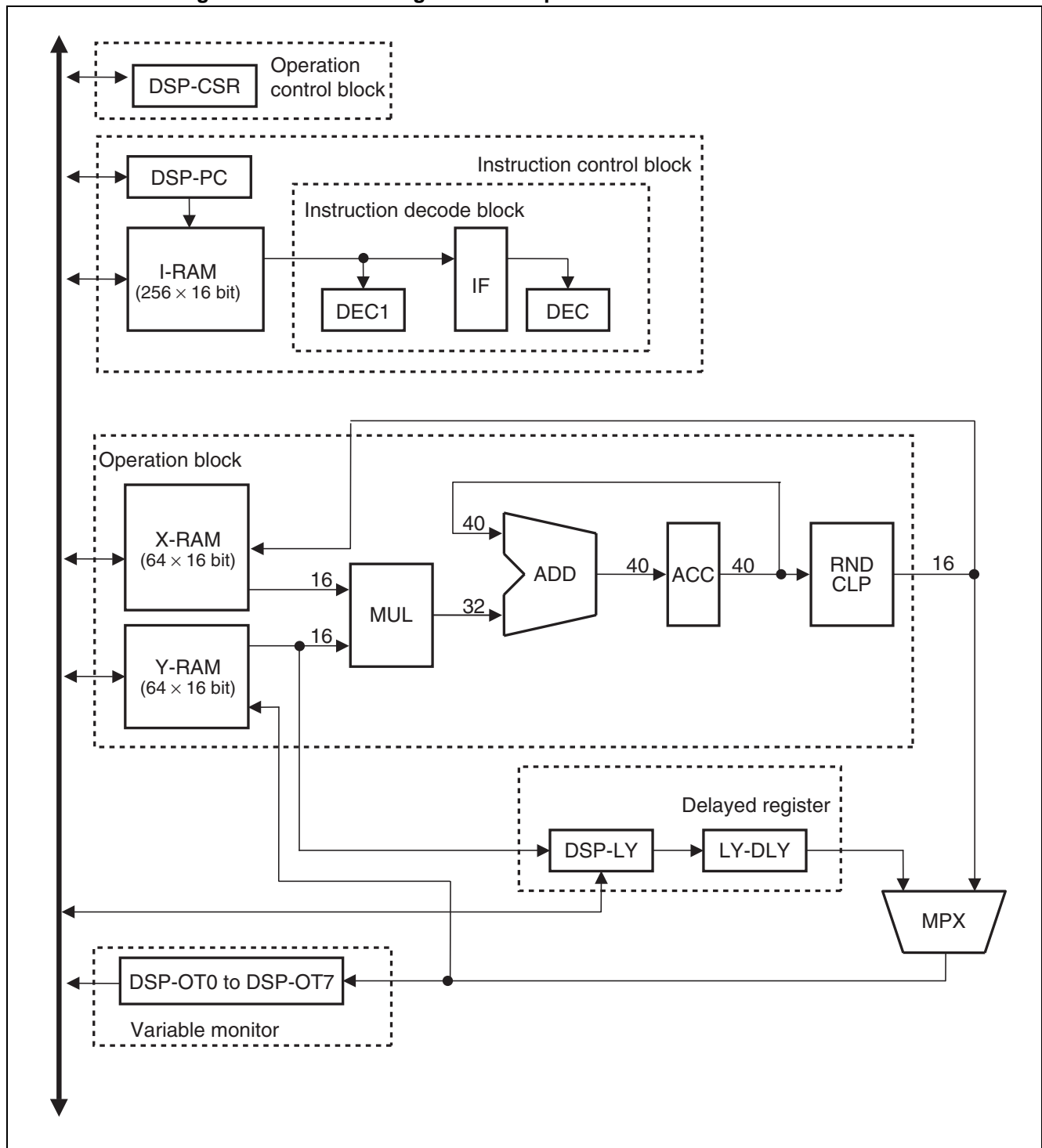


Table 15.1-1 Block Diagram Outline Explanation

Block	Register	Function
Operation control	DSP-CSR	Operation control register of multiplication and addition calculation macro The following operations are controlled from the CPU and servo block. <ul style="list-style-type: none"> • Calculation start/end instructions • Interrupt control • Program flow control (used for conditional branching commands of multiplication and addition macro)
Instruction control	DSP-PC	Program counter Program execution begins from the first address specified by the CPU.
	I-RAM	Instruction RAM of 256×16 -bit The CPU can perform R/W while the multiplication and addition macro calculation is halted. Load the command code from the CPU before starting calculations.
	IF	Instruction fetch register
	DEC1* DEC*	Instruction decoder
Arithmetic unit	X-RAM	Data RAM of 64×16 -bit The CPU can perform R/W while the multiplication and addition macro calculation is halted. Load the coefficient from the CPU before starting calculations.
	Y-RAM	Data RAM of 64×16 -bit The CPU can perform R/W while the multiplication and addition macro calculation is halted. Load the variable from the CPU before starting calculations.
	MUL*	$16 \times 16 \rightarrow 32$ -bit multiplier
	ADD*	$32 + 40 \rightarrow 40$ -bit adder
	ACC*	40-bit accumulator
	CLP* RND* SLQ*	When sending from 40 to 16 bits, if there is out-of-range 16-bit data, saturated to the maximum value. When sending from 40 to 16 bits, the lower-order bits are rounded. When sending from 40 to 16 bits, select the bits to send.
Delayed register	DSP-LY LY-DLY*	Delayed register During multiplication and addition, you can save the variable value, and write it back to Y-RAM.
Variable monitor output	DSP-OT0 to DSP-OT7	Variable monitor output register Stores the same value as addresses 0 to 7 of Y-RAM. During calculation (when Y-RAM access is disabled), you can monitor addresses 0 to 7 of Y-RAM.

*: Access from CPU disabled.

■ Instruction Definitions

The multiplication and addition macro has three main types of command: MAC/STR/JMP. This specification uses commands other than these three in notation. The command hierarchy is as follows.

MAC instruction

- _____ Multiplication and addition instruction (CLAC bit = 0)
- _____ Multiplication instruction (CLAC bit = 1)

STR instruction

- _____ HLT instruction (HLT bit = 1)
- _____ INT instruction (SIRQ bit = 1)

JMP instruction

- _____ Unconditional branch instruction (COND bit = 0)
- _____ Conditional branch instruction (COND bit = 1)
- _____ HLT instruction (HLT bit = 1)
- _____ INT instruction (SIRQ bit = 1)

15.2 Register Description

This section explains the register configuration and function of using the multiplication and addition calculator.

■ DSP-CSR (Control/Status Registers)

The control/status register is an 8-bit register. It consists of flags to change the multiplication and addition macro state, control interrupts to the CPU, and indicate the status of the multiplication and addition macro. This register is also used to set the conditions for conditional branching commands in the multiplication and addition macro.

The 8-bit register always allows external R/W.

● Control function

- Multiplication and addition macro state (calculation start/end) transition (GoDSP and HltDSP)
- Interrupt mask for CPU (IeDSP)
- Sets conditions for conditional branching command of multiplication and addition macro (USR2, USR1, and USR0).
- Multiplication and addition macro state (calculation start/end) transition (GoDSP and HltDSP)
- Interrupt mask for CPU (IeDSP)
- Sets conditions for conditional branching command of multiplication and addition macro (USR2, USR1, and USR0).

● Status function

- Multiplication and addition macro current state acquisition flag (RunDSP)
- Interrupt request flag (IrqDSP)
- Saturation flag (SatDSP)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
DSP-CSR Address: 3A1H	SatDSP	USR2 USR2	USR1 USR1	USR0 USR0	IrqDSP IrqDSP	IeDSP IeDSP	HltDSP	GoDSP RunDSP	Write Read
	(R) 0	(R/W) 0	(R/W) 0	(R/W) 0	(R/W) 0	(R/W) 0	(W) 0	(R/W) 0	Read/Write Initial value

[bit7] SatDSP (Saturation flag): Read only

- This status flag stores whether saturation was performed during calculation.
- This bit is set if saturation is indicated in the STR command (CLP = 1), and saturation was actually performed. If this bit is set during calculation, the setting is maintained until the next calculation starts.
- This bit is cleared when calculation begins.

Set factor: Set if the STR command performed saturation processing during calculation.

Clear factor: Cleared by start of calculation [initial value].

- At reset: Be initialized to "0". (No saturation)
- This bit is read-only. Writing does not change the bit value.

[bit6, bit5, bit4] USR2, USR1, USR0 (Set a jump condition.): Read/Write

- This bit is referred by the multiplication and addition macro conditional branching command (when COND bit =1). A jump is performed if the value of this bit matches the conditional branching command UBP flag (condition attained). In other words, it is possible to switch between 8 different calculation routines from the CPU by combining this conditional branching command with the calculation commands.
- At reset: Be initialized to "000_B".
- The read / write is possible.

[bit3] IrqDSP (interrupt request flag): Read/Write

- This flag indicates that the multiplication and addition macro has generated a software interrupt request. When interrupt requests are enabled (IeDSP = 1), setting this bit creates an interrupt request for the CPU.
- Multiplication and addition macro interrupt requests are generated by the software, by setting the STR or JMP command SIRQ bit to "1".
Set factor: Set via the generation of a multiplication and addition macro software interrupt (STR/JMP command).
Clear factor : Clear by writing "0" [initial value]
- At reset: Be initialized to "0". (no interrupt request)
- The read / write is possible. Note, however, that only "0" can be written. Writing "1" will not change the bit value.
- For read-modify-write commands, the value of "1" is always read, regardless of the value of the bit.

[bit2] IeDSP (interrupt request enable bit): Read/Write

Interrupt requests to the CPU (IrqDSP = 1) are controlled as follows.

0: Interrupt request output disabled (setting IrqDSP will not generate an interrupt) [initial value]

1: Interrupt request output enabled (setting IrqDSP will generate an interrupt)

- At reset: Be initialized to "0". (interrupt request output is disabled.)
- The read / write is possible.

[bit1] HltDSP (calculation stop): Write only

- This bit forcibly stops calculation.
- When "1" is written to this bit, if calculation is being performed (RunDSP = 1), calculation will halt after the currently executing command terminates (if it is a two-cycle command, after two cycles), and the RunDSP flag is cleared.
- If calculation is stopping, this has not effect.
- If a stop is forced via this bit, DSP-PC points to the command address following the stopped command. It is thus possible to continue with command execution.
 - Forcibly stop by writing "1".
 - Writing "0" is invalid. "0" is always read.
- At reset: Be initialized to "0".

[bit0] GoDSP (calculation start): Write only**RunDSP (Calculation running flag): Read only**

- Calculation start is indicated by writing "1" to GoDSP bit. When calculation is stopped (RunDSP = 0), calculation is activated, and the RunDSP flag is set. If calculation is already ongoing (RunDSP = 1), this action has no effect.
- The RunDSP flag indicates that calculation is being executed. This is set by the start of calculation, and cleared by writing "1" to the HltDSP bit or executing the multiplication and addition macro's HLT command.
- When a calculation is being executed (RunDSP = 1), DSP-PC, DSP-LY, X-RAM, Y-RAM, and I-RAM cannot be accessed from the CPU. Only DSP-CSR and DSP-OT 0 to 7 can be monitored.
- To start calculation, you must store the start address of the calculation routine in DSP-PC before or at the same time as activation.

(1) Write-time function (GoDSP: start calculation)

0: No function/No effect on operation

1: When calculation is halted → start calculation

If calculation is ongoing → has no effect

(2) Read-time function (RunDSP: calculation running flag)

0: Calculation is stopped [initial value]

Clear factor → Writing "1" to HltDSP, and executing an HLT command.

1: Calculation is executing.

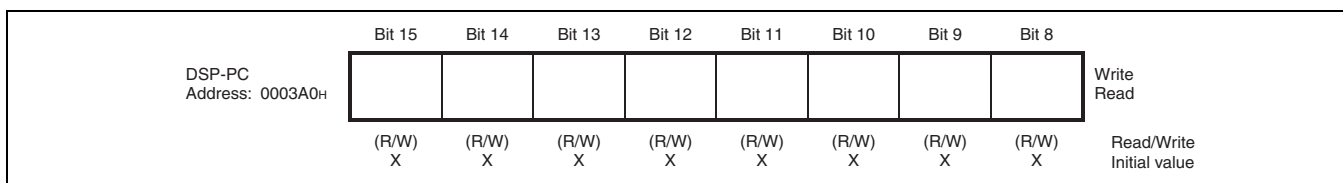
Set factor → start calculation

- At reset: Be initialized to "0". (Calculation is stopping.)
- The read / write is possible. Note, however, that as described above, the significance differs depending on whether reading or writing is being performed.
- For read-modify-write commands, the value of "0" is always read, regardless of the value of the bit.

■ DSP-PC (Program Counter)

The program counter is 8 bits long. It indicates the memory address (I-RAM) where the command code to be executed by the multiplication and addition macro is stored. Although the program counter is automatically updated when a command is executed, it can be rewritten via a multiplication and addition macro JMP. Additionally, access (R/W) from the CPU is only possible when calculation is stopped. You must store the start address of the calculation routine in DSP-PC before or at the same time as calculations starts.

After executing the HLT command, or writing "1" to the DSP-CSR HltDSP, DSP-PC points to the address of the command after the stopped command. Program execution can be continued by again setting GoDSP.

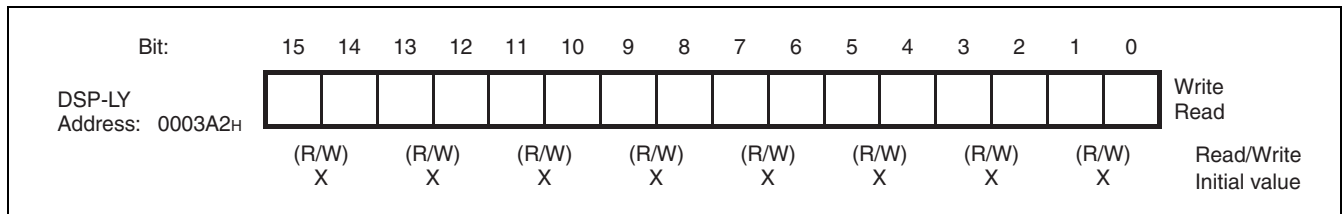


- At reset: Indeterminate.
- Although read/write is enabled, access is only possible when multiplication and addition macro calculation is halted (DSP-CSR: RunDSP = 0).
- When a calculation is running (DSP-CSR: RunDSP = 1), access from the CPU is not possible because it is separated from the bus.

■ DSP-LY (Delayed Register)

DSP-LY is a 16-bit register. It is used when the MAC command delay write bit (LDLY) of the multiplication and addition macro is set to "1". Access is not possible during calculation (DSP-CSR: RunDSP = 1).

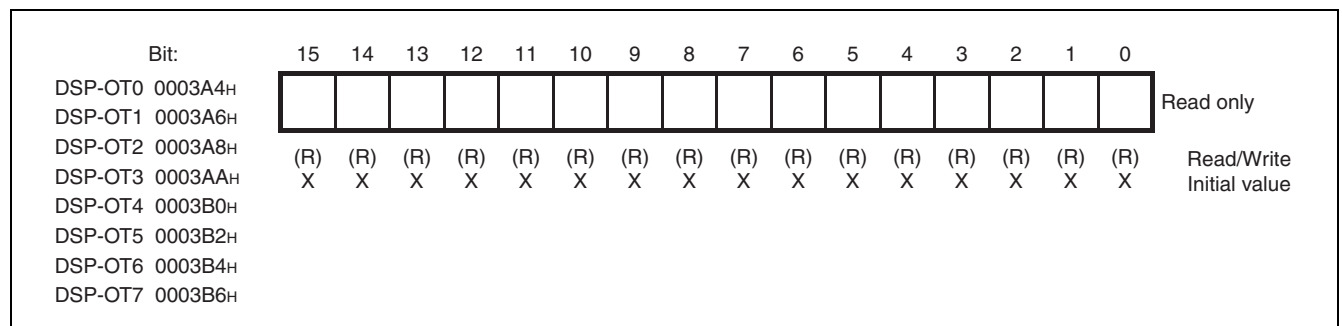
- When the MAC command LDLY bit is set to "1", the following two operations are performed.
 - The contents of the DSP-LY register are sent to the LY-DLY register.
 - The Y-RAM read data selected by means of the MAC command is stored in the DSP-LY register.
- When the MAC command's STLY bit is "1", after the MAC command executes, the value of the LY-DLY register is written to the Y-RAM address selected by the MAC command. At this time, the execution time is two cycles.



- At reset: Indeterminate.
- Although read/write is enabled, access is only possible when DSP-LY calculation is halted (DSP-CSR: RunDSP = 0). When a calculation is running (DSP-CSR: RunDSP = 1), access from the CPU is not possible because it is separated from the bus.

■ DSP-OT0 to DSP-OT7 (Variable Monitor Register)

There are eight 16-bit registers for use as variable monitor registers. With the exception of the initial status when the power is turned on, the same value as addresses 0 to 7 of Y-RAM is held. Only read is always available from the CPU. Even if calculation is ongoing, the contents of addresses 0 to 7 in Y-RAM can be monitored.



- At reset: Indeterminate.
- Only read can always be performed. Read is also possible if multiplication and addition macro program execution is ongoing.

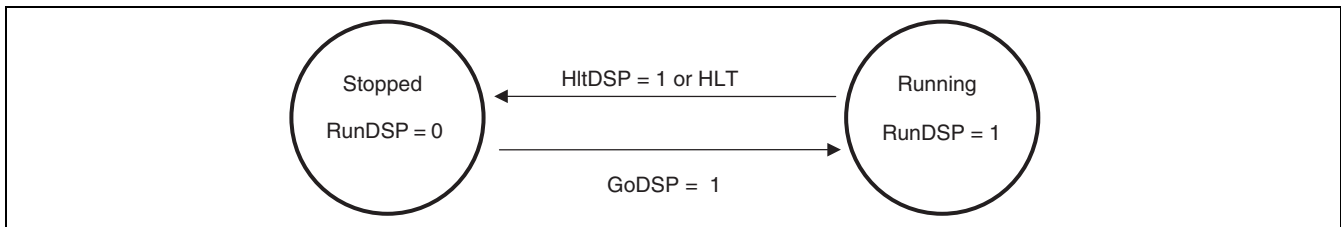
15.3 Operation Explanation

This section explains the operating mode and the instruction operation the multiplication and addition macro.

■ Operating Mode

The action of the multiplication and addition macro is controlled by manipulating the DSP-CSR register.

The multiplication and addition macro can be one of the two states described below. When stopped, the multiplication and addition macro starts program execution when "1" is written to GoDSP, or a GODSPSV signal is inputted from the Servo Block. The registers and memory that can be accessed from CPU depend on whether calculation is being executed or stopped.



Each of these states is described below.

- **Stopped:** The multiplication and addition macro is stopped. The command RAM (I-RAM), data RAM (X-RAM and Y-RAM), and all registers of the multiplication and addition macro can be accessed from the CPU. Transit to this state by writing "1" to HltDSP or executing the HLT command. The system is also initialized to this state when it is reset.
- **Running:** The multiplication and addition macro is running.
When "1" is written to the GoDSP bit while in stopped status, control transitions to this state, and program execution starts from the current DSP-PC (program counter). Transit to the stopped state, and halt program execution, by writing "1" to the HltDSP bit or executing the HLT command. Only the DSP-CSR and the DSP-OT 0 to 7 register are accessible from the CPU. (Access to other registers and RAM is disabled *)

*: Although access is disabled, R/W operations have the following effect:

Write: No effect (Nothing is written.)

Read: Indeterminate

■ Instruction Operation

When "1" is written to the GoDSP bit of the DSP-CSR register, the multiplication and addition macro begins command execution from the current DSP-PC (program counter). (Operation is in parallel with CPU operation)

Before execution, set the I-RAM and DSP-PC values. (DSP-CSR and DSP-PC can be set simultaneously.)

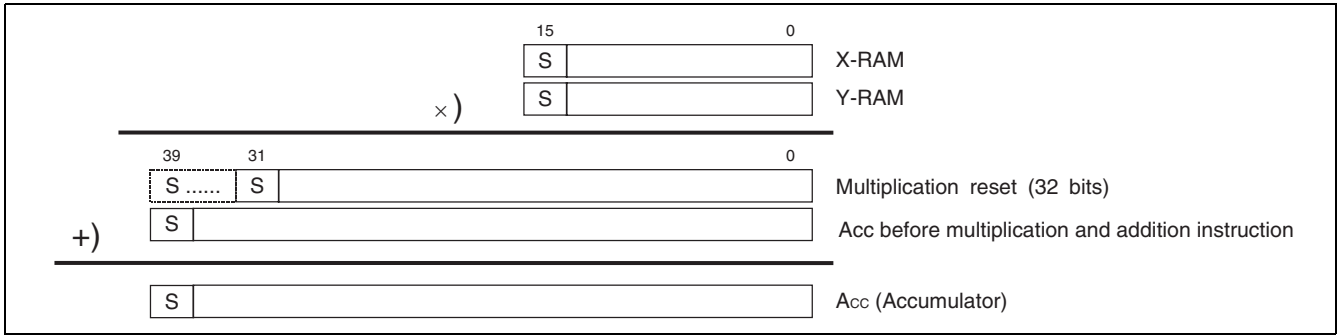
When multiplication and addition macro command execution starts, the following operation control is performed.

- When the multiplication and addition macro executes an HLT command*, transits to stopped state after the command is executed. At this time, DSP-PC stops pointing to the address following the HLT command.
- When a JMP command or STR command is being executed, interrupt requests can be sent to the CPU. (interrupt masking available)
- Use a conditional branching command referring to bits USR0 to USR2 of the DSP-CSR, to switch program flow.

*: An HLT command is a JMP or STR command with the HLT bit set to "1".

■ Calculating Function

The multiplication and addition macro has two 16-bit data RAMs (X-RAM and Y-RAM). When a multiplication and addition (or multiplication) command is executed, data is read from each RAM area, and signed multiplication and addition (or multiplication) calculation is performed and stored in 40-bit accumulator. The data format is as follows.



Notes:

- In the case of multiplication commands, the multiplication results are stored in the accumulator, extended to a signed 40-bit value. (The accumulator contents are zero-cleared immediately beforehand)
- "S" indicates the signed bit.

Results are not guaranteed if a large number of multiplication and addition commands is performed, and an accumulator overflow occurs.

Do not perform more than 512 multiplication and addition commands in a row.

■ Delayed Write Feature

When executing a multiplication and addition (or multiply) command, the following transfer operation can be performed at the same time. Performing this transfer and the calculation in tandem makes it easy to perform delayed data processing with a digital filter.

- The value read from Y-RAM is stored in the DSP-LY register.
- Performs a delayed write of the value in the DSP-LY register prior to executing the instruction to the Y-RAM read address, via the LY-DLY register.

■ Transfer of Operation Results

Although operation results held in the accumulator are transferred to X or Y-RAM as 16-bit values, the following scaling is performed.

- Output bit select
The following bit ranges from the 40-bit accumulator can be selected.
 - bit27 to bit12 (Q12 format)
 - bit28 to bit13 (Q13 format)
 - bit29 to bit14 (Q14 format)
 - bit30 to bit15 (Q15 format)
 - bit23 to bit8 (Q8 format)

bit24 to bit9 (Q9 format)

bit25 to bit10 (Q10 format)

bit26 to bit11 (Q11 format)

- Rounding processing

The LSB is set by rounding up the bit below the LSB of the selected output bits (round up if "1", ignore if "0").

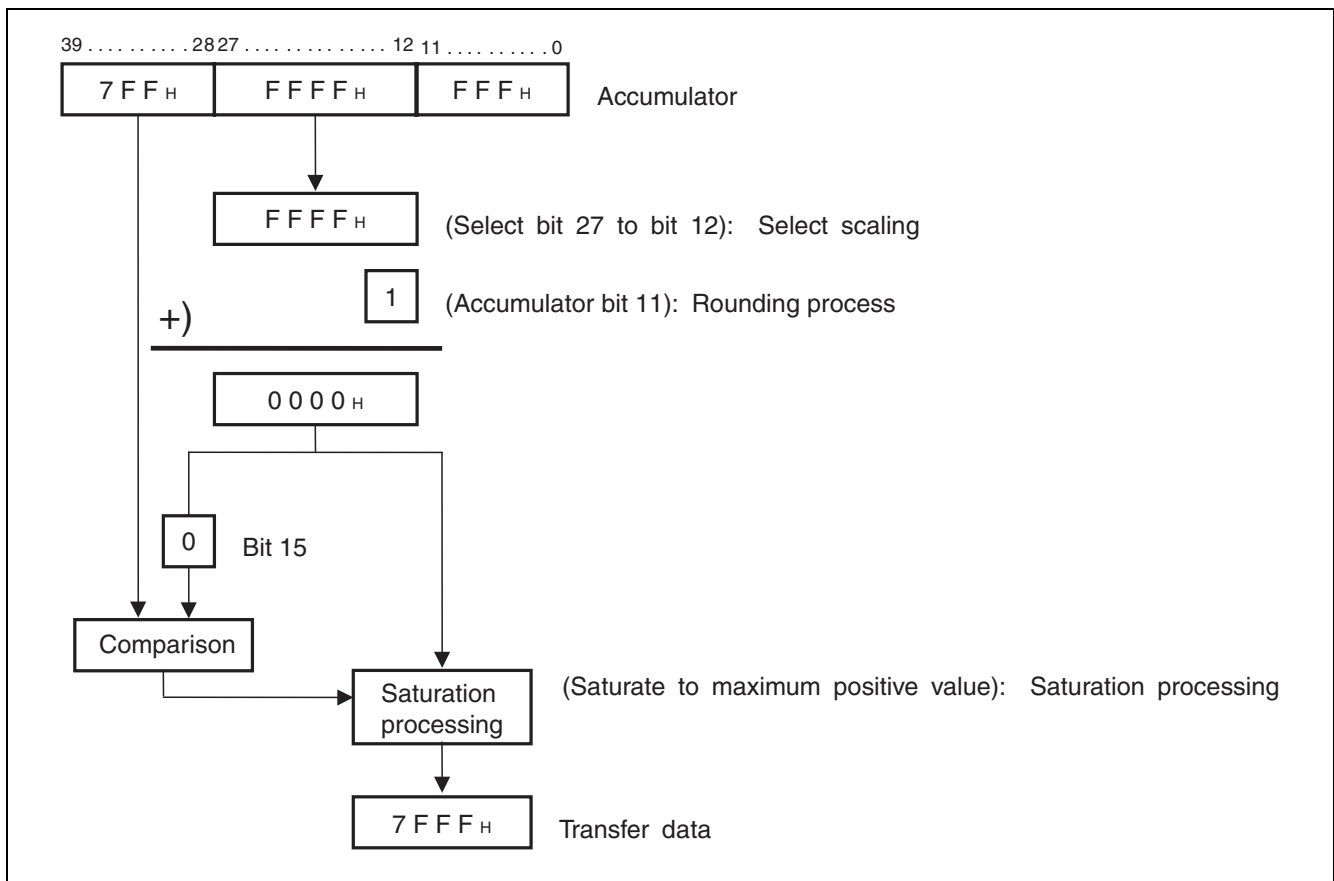
- Saturation processing

This compares the sign bit (MSB) of the rounded 16-bit data with the upper bits of the accumulator. If a bit with a different value is present, the data value is saturated (set to maximum or minimum). The result of saturation depends on the sign bit (MSB) of the accumulator as follows.

If the sign of the accumulator is "0" -> Saturate to maximum positive value "7FFF_H".

If the sign of the accumulator is "1" -> Saturate to maximum negative value "8000_H".

The following shows an example.



■ Variable Monitor Output

The multiplication and addition macro includes registers (DSP-OT0 to DSP-OT7) that always contain the values of addresses 0 to 7 in Y-RAM. Whenever data is written to Y-RAM addresses 0 to 7 (write from CPU, write by STR instruction, or delayed write), the same data is written to the DSP-OT0 to DSP-OT7 registers.

Although CPU access to Y-RAM is prohibited during calculation, by using the STR instruction to store any results required by the CPU in Y-RAM addresses 0 to 7, the CPU can read calculation results at any time.

Notes:

When using the DMA transfer in the multiplication and addition macro,

- Configure the CPU clock equivalent to or faster than the peripheral clocks.
 - If the CPU clock is slower than the peripheral clocks, the DMA transfer does not work properly.
-

15.4 Instruction Detail Explanation

This section explains the instruction detail for using on the multiplication and addition calculator.

■ MAC Instruction

Operation: $ACC \leftarrow ACC + X \text{ data} \times Y \text{ data}$

$LY-DLY \leftarrow DSP-LY$

$DSP-LY \leftarrow Y \text{ data} (LDLY = 1)$

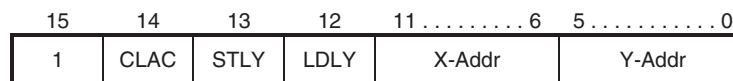
$Y-RAM \leftarrow LY-DLY (STLY = 1)$

Explanation: Add the result of multiplying X data from X-RAM and Y data from Y-RAM to the accumulator. Also transfer the contents of the DSP-LY register to the LY-DLY register at the same time.

Word count : 1 word (16-bit width)

Cycle count : 1 system clock cycle (2 cycles if STLY = 1)

Operation code:



[bit14] CLAC (Clear Acc)

- Setting this bit causes the instruction to act as a multiplication instruction.
 0: $ACC \leftarrow ACC + X \text{ data} \times Y \text{ data}$ [multiplication and addition instruction]
 1: $ACC \leftarrow 0 + X \text{ data} \times Y \text{ data}$ [multiplication instruction]

[bit13] STLY (Store LY)

- The following processing is performed when this bit is "1". If "0", the operation only is performed.
- After performing the operation, save the contents of the LY-DLY register at the Y-Addr address in Y-RAM.
- The execution time only increases to two cycles when this bit is set.

[bit12] LDLY (Load LY)

- The following processing is performed when this bit is "1". If "0", the operation only is performed.
- During the operation, the contents of the Y-Addr address in Y-RAM are stored in the DSP-LY register.

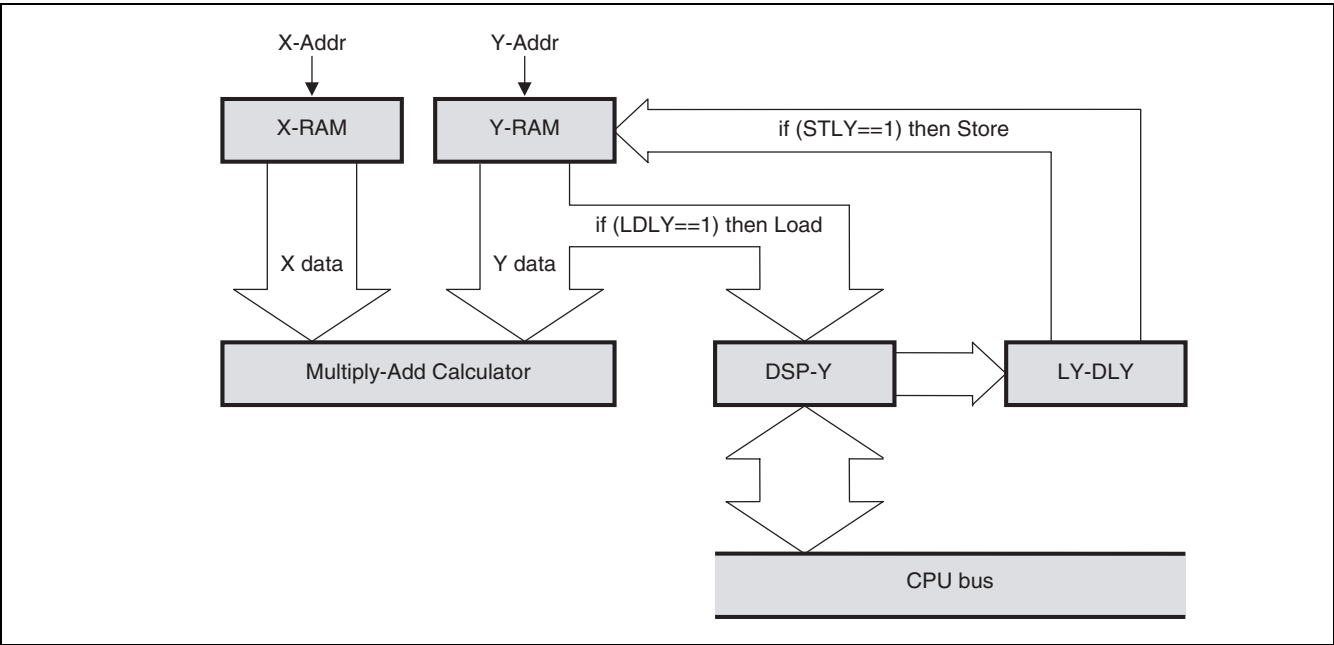
[bit11 to bit6] X-Addr (X-RAM Address)

Address bits for specifying the location of the X data in X-RAM.

[bit5 to bit0] Y-Addr (Y-RAM Address)

- Address bits for specifying the location of the Y data in Y-RAM.

Figure 15.4-1



■ STR Instruction (Transition Instruction)

Operation: Data RAM \leftarrow Accumulator

Explanation: Convert the 40-bit accumulator value to a 16-bit value in accordance with the RND, CLP, and SLQ flags. Store the result in the data RAM specified by the SLY flag and X/Y-Addr.

Word count: 1 word (16-bit width)

Cycle count: 1 system clock cycle

Operation code:

15	14	13	12	11	10	9.....7	6	5.....0
0	1	HLT	SIRQ	RND	CLP	SLQ	SLY	X- and Y-Addr

[bit13] HLT (HLT instruction indicating flag)

Setting this bit causes the multiplication and addition macro to halt program execution after instruction execution completes.

Clears the RunDSP flag in the DSP-CSR register.

[bit12] SIRQ (INT instruction indicating flag)

Setting this bit causes an interrupt request to be sent to the CPU after instruction execution completes, then sets the IrqDSP flag in the DSP-CSR register.

[bit11] RND (Rounding)

This bit specifies whether to perform rounding for 16-bit data specified by the SLQ bits.

Rounding rounds the 16-bit data based on the bit immediately below the LSB (round up if lower bit is "1", ignore if "0").

[bit10] CLP (Clipping)

This bit specifies whether to perform saturation processing on the 16-bit data value specified by the SLQ bits if the calculation result in the accumulator represents an overflow in the 16-bit value.

In practice, saturation processing is performed if the MSB of the accumulator (bit39) is different to the MSB of the 16-bit value (determined by the SLQ bits). If rounding is enabled, the comparison is performed using the result after rounding.

The value transferred to data RAM is the maximum positive value ("7FFF_H") if the value of the accumulator was positive before rounding, or the maximum negative value ("8000_H") if negative.

Rounding and saturation processing does not change the sign of the accumulator.

[bit9 to bit7] SLQ

Specifies which bits of the accumulator to transfer to data RAM.

SLQ Bit	Overflow Judging Bit	Transfer 16-bit Data	Rounding Bit	Fixed-point Decimal Format
0 0 0	bit39 to bit27	bit27 to bit12	bit11	Q12
0 0 1	bit39 to bit28	bit28 to bit13	bit12	Q13
0 1 0	bit39 to bit29	bit29 to bit14	bit13	Q14
0 1 1	bit39 to bit30	bit30 to bit15	bit14	Q15
1 0 0	bit39 to bit23	bit23 to bit8	bit7	Q8
1 0 1	bit39 to bit24	bit24 to bit9	bit8	Q9
1 1 0	bit39 to bit25	bit25 to bit10	bit9	Q10
1 1 1	bit39 to bit26	bit26 to bit11	bit10	Q11

[bit6] SLY

Specifies the transfer destination.

0: X-RAM

1: Y-RAM

[bit5 to bit0] X/Y Addr (RAM Address)

Specifying direct addressing for data RAM

■ JMP Instruction (Branching Command)

Operation: [when condition is satisfied] $\text{DSP-PC} \leftarrow \text{J-Addr8}$
 [when condition is not satisfied] $\text{DSP-PC} \leftarrow \text{DSP-PC} + 1$

Explanation: Branch if condition is satisfied, do not perform any operation if condition is not satisfied.

Word count: 1 word (16-bit width)

Cycle count: 1 system clock cycle

Operation code:

15	14	13	12	11	10 8	7 0
0	0	HLT	SIRQ	COND	UBP	J-Addr8

[bit13] HLT (HLT instruction specifying flag)

Setting this bit causes the multiplication and addition macro to halt program execution after instruction execution completes.

Clears the RunDSP flag in the DSP-CSR register.

[bit12] SIRQ (INT instruction specifying flag)

Setting this bit generates an interrupt request to the CPU after instruction execution completes.

Sets the IrqDSP flag in the DSP-CSR register.

[bit11] COND (CONDition)

0: Unconditional branch

1: Conditional branch

[bit10 to bit8] UBP2 to UBP0 (condition specification)

Sets the condition to use for the conditional branch. The condition is established if these bits match the value of the USR2 to USR0 bits in the DSP-CSR register.

These bits must be set to "000_B" if an unconditional branch is specified.

[bit7 to bit0] J-Addr8 (Jump Address)

Address specification for branch destination

CHAPTER 16

DMAC

(DMA Controller)

This chapter explains the overview of the DMA controller (DMAC), the configuration and functions of registers, and DMAC operation.

- 16.1 Overview
- 16.2 Register Details Explanation
- 16.3 Operation Explanation
- 16.4 Setting Up Transfer Request
- 16.5 Transfer Sequence
- 16.6 Overview of DMA Transfer
- 16.7 Operation Flow
- 16.8 Data Path

16.1 Overview

This module is used to implement DMA (Direct Memory Access) transfer in FR family devices.

This module can be used to increase system performance by using DMA transfer to perform various types of data transfer at high speed without going via the CPU.

■ Hardware Configuration

The module consists of the following main components.

- Independent DMA channel × 5 channels
- 5 channels independent access control circuit
- 20-bit address registers (Reload specifying permitted: ch0 to ch3)
- 24-bit address registers (Reload specifying permitted: ch4)
- 16-bit transfer count registers (Reload specifying permitted: one per channel)
- 4-bit block count registers (one per channel)
- Two-cycle transfer

■ Main Function

The main data transfer functions supported by this module are as follows.

Independent data transfer can be performed for multiple channels (5 channels)

(1) Priority order (ch.0>ch.1>ch.2>ch.3>ch.4)

(2) Alternating transfer is supported between ch0 and ch1.









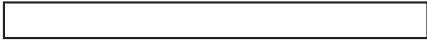
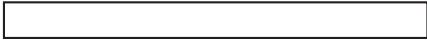
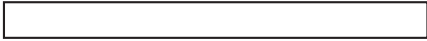
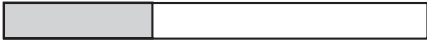
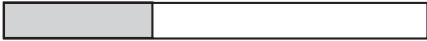



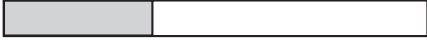
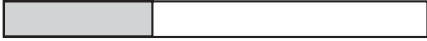



(3) DMAC startup factor

- Request from internal peripheral (uses interrupt requests, including the external interrupts)
- Software request (register write)

(4) Transfer mode

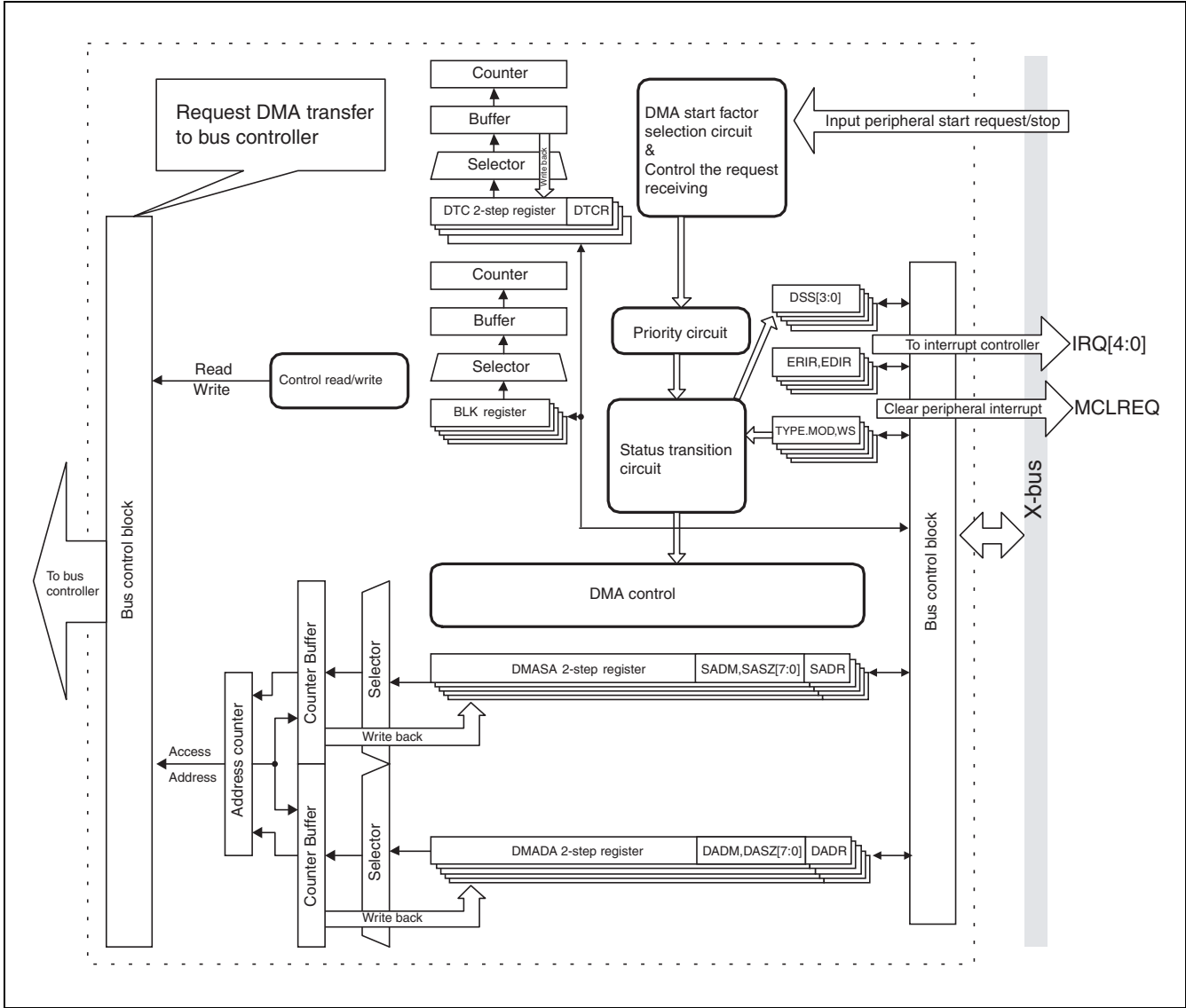
- Burst transfer/step transfer/block transfer
- Addressing mode with a 20-bit (24-bit) address setting (increment, decrement, fixed)
(Increment/decrement width of address can be ± 1 , 2, or 4)
- Data types: byte, half-word, or word
- Selectable single or reload

■ Register Description

			(bit)	31	24	23	16	15	08	07	00	
ch0 control/status register A	DMACA0	00000200 _H										
ch0 control/status register B	DMACB0	00000204 _H										
ch1 control/status register A	DMACA1	00000208 _H										
ch1 control/status register B	DMACB1	0000020C _H										
ch2 control/status register A	DMACA2	00000210 _H										
ch2 control/status register B	DMACB2	00000214 _H										
ch3 control/status register A	DMACA3	00000218 _H										
ch3 control/status register B	DMACB3	0000021C _H										
ch4 control/status register A	DMACA4	00000220 _H										
ch4 control/status register B	DMACB4	00000224 _H										
Total control register	DMACR	00000240 _H										
			(bit)	31	20	19						00
ch0 transferring source address register	DMASA0	00001000 _H										
ch0 transferring destination address register	DMADA0	00001004 _H										
ch1 transferring source address register	DMASA1	00001008 _H										
ch1 transferring destination address register	DMADA1	0000100C _H										
ch2 transferring source address register	DMASA2	00001010 _H										
ch2 transferring destination address register	DMADA2	00001014 _H										
ch3 transferring source address register	DMASA3	00001018 _H										
ch3 transferring destination address register	DMADA3	0000101C _H										
			(bit)	31	24	23						00
ch4 transferring source address register	DMASA4	00001020 _H										
ch4 transferring destination address register	DMADA4	00001024 _H										

■ Block Diagram

Figure 16.1-1 Block Diagram of DMAC 5ch



16.2 Register Details Explanation

This section explains the register configuration and function for using DMAC.

■ Notes on Setting Register

Some bits in the DMAC may only be set when the DMA is halted. If set during operation (during transfer), correct operation cannot be guaranteed.

An asterisk "*" indicates bits that will affect operation if set during DMAC transfer. Only modify this bit when the DMAC transfer is halted (set to the disabled or paused state). Values set while DMA transfer is disabled (DMACR: DMAE=0 or DMACA: DENB=0) become active when DMA is re-enabled.

Values set while DMA transfer is paused (DMACR: DMAH(3:0)≠ 0000_B or DMACA: PAUS=1) become active when DMA is restarted.

■ DMAC-ch.0, ch.1, ch.2, ch.3, ch.4 Control/Status Registers A

[DMACA0 to DMACA4]

These registers control the operation of each DMAC channel. A separate register is provided for each channel.

The function of each bit is as follows.

bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
	DENB	PAUS	STRG	IS[4:0]					-					BLK[3:0]				
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	DTC[15:0]																	
(Initial value: 00000000 00000000 00000000 00000000 _B)																		

[bit31] DENB (Dma ENaBle): DMA Operation Enable Bit

Enables or disables DMA transfer for each transfer channel.

Once a channel is enabled, DMA transfer starts when a transfer request is received.

All transfer requests for disabled channels are ignored.

This bit goes to "0" when the specified number of transfers has completed for an enabled channel and the transfer halts.

Although writing "0" to this bit forcibly halts DMA, always use the PUAS bits (DMACA: bit30) to pause DMA before forcibly halting (writing "0"). If DMA transfer is forcibly halted without pausing, the DMA transfer halts but the validity of the transferred data is not guaranteed. Use the DSS(2:0) bits (DMACB: bit18 to bit16) to check whether transfer has finished.

DENB	Function
0	Corresponding channel DMA is disabled to operate [initial value].
1	Corresponding channel DMA is enabled to operate.

- After a reset or when a halt request is received: Initialized to "0".
- The read / write is possible.

- Writing "1" to this bit is ignored and DMA remains halted if operation has been disabled for all channels by the DMAE bit (bit15 of the overall DMAC control register DMACR). Also, if operation is disabled by the aforementioned bit while still enabled by this bit, this bit is cleared to "0" and transfer is aborted (forcibly halted).

[bit30] PAUS (PAUSE): Pause indication

Pauses DMA transfer for the corresponding channel. No DMA transfer is performed from the time this bit is set until it is cleared again. (The DSS bits go to "1xx_B" when DMA is halted.)

If this bit is set before DMA is enabled, DMA remains paused.

New transfer requests are accepted while this bit is set but the transfers do not start until the bit is cleared. (See "■ Transfer Request Acceptance and Transfer".)

PAUS	Function
0	Corresponding channel DMA is enabled to operate (initial value).
1	Corresponding channel DMA is suspended.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit29] STRG (Software TRiGger): Transfer request

Generates a DMA transfer request for the corresponding channel. Writing "1" to this bit generates a transfer request as soon as the register write completes and starts the transfer on the corresponding channel. However, if the corresponding channel is not enabled, writing to this bit is ignored.

Note:

If a transfer request is set via this bit at the same time as start is enabled by the DMAE bit, the transfer request is valid and transfer starts. If this bit is written to at the same time as writing "1" to the PAUS bit, the transfer request is valid, but DMA transfer does not start until the PAUS bit is cleared to "0".

STRG	Function
0	Not provided
1	DMA activating request

- Initialized to "0" when resetting.
- Reading always returns "0".
- Only writing "1" is meaningful. Writing "0" has no effect on the operation.

[bit28 to bit24] IS4 to IS0 (Input Select)*: Transfer factor selection

The transfer request triggers are selected as follows. However, the software transfer request triggered by the STRG bit remains available regardless of this setting.

IS	Function	Transfer Halt Request
00000 _B	Software transfer request only	Not provided
00001 _B ↓ 01111 _B	Setting disabled ↓ Setting disabled	
10000 _B	UART0 (reception completed)	
10001 _B	UART1 (reception completed)	Provided
10010 _B	UART2 (reception completed)	
10011 _B	UART0 (transfer completed)	
10100 _B	UART1 (transfer completed)	Not provided
10101 _B	UART2 (transfer completed)	
10110 _B	External interrupt 0	
10111 _B	External interrupt 1	
11000 _B	Reload timer 0	
11001 _B	Reload timer 1	
11010 _B	Reload timer 2	
11011 _B	Multiplication and addition macro	
11100 _B	PPG0	
11101 _B	PPG1	
11110 _B	PPG2	
11111 _B	PPG4	

- Initialized to "00000_B" when resetting.
- The read / write is possible.

Notes:

- When an interrupt from a peripheral function is set as a DMA activation (IS = 1XXXX_B), disable the interrupt for the selected function in the ICR register.
- Note also that, if the software transfer request is used to start DMA transfer while triggering DMA transfer from a peripheral function interrupt is enabled, an factor clear is sent to the peripheral function when the transfer completes. As this may clear an existing transfer request, do not use the software transfer request function to start DMA while triggering DMA transfer from a peripheral function interrupt is enabled.

[bit23 to bit20] (Reserved): Unused bit

Reading value is fixed to "0000_B". Writing has no effect.

[bit19 to bit16] BLK3 to BLK0 (BLoCK size): Block size setting

Specifies the size for block transfer on the corresponding channel. The value set in these bits specifies the number of words to transfer for each transfer operation (or, more exactly, the number of times transfer of the specified word size is repeated). Always set "01_H" (size 1) when not performing block transfer.

BLK	Function
XXXX	Specifying of block size for corresponding channel

- Initialized to "0000_B" when resetting.
- The read / write is possible.
- If all-zeros are specified, the block size is set to 16 words.
- Reading always returns the block size (reload value).

[bit15 to bit00] DTC15 to DTC0 (Dma Terminal Count register) *: Transfer count register

This register stores the number of transfers performed. Each register consists of 16 bits.

Each register has its own reload register. On channels that allow the transfer count register to be reloaded, the initial value is automatically returned to the register when the transfer completes.

DTC	Function
XXXX	Specifying the number of transfers for corresponding channel

When DMA transfer starts, the data in this register is stored to the counter buffer in the dedicated DMA transfer counter and the value decremented by one after each transfer. When DMA transfer completes, the value of the counter buffer is written back to this register and the DMA operation ends. Accordingly, you cannot use this register to read the specified number of transfers during a DMA operation.

- Initialized to "00000000_00000000_B" when resetting.
- The read / write is possible. Always use halfword or word access to access the DTC register.
- Reading the register returns the counter value. You cannot read the reload value.

■ DMAC-ch.0, ch.1, ch.2, ch.3, ch.4 Control/Status Registers B

[DMACB0 to DMACB4]

These registers control the operation of each DMAC channel. A separate register is provided for each channel.

The function of each bit is as follows.

bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TYPE [1:0]		MOD [1:0]		WS [1:0]		SADM	DADM	DTCR	SADR	DADR	ERIE	EDIE	DSS[2:0]		
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SASZ[7:0]								DASZ[7:0]							

(Initial value: 00000000_00000000_00000000_00000000_B)

[bit31, bit30] TYPE (TYPE) *: Transfer type setting

Sets the type of operation for the corresponding channel as follows.

Two-cycle transfer mode: This transfer mode sets the transfer source address (DMASA) and transfer destination address (DMADA), then performs the specified number of read and write operations.

TYPE	Function
00 _B	Two-cycle transfer (initial value)
01 _B	Setting disabled
10 _B	Setting disabled
11 _B	Setting disabled

- Initialized to "00_B" when resetting.
- The read / write is possible.
- Always set this bit to "00_B".

[bit29, bit28] MOD (MODE)* : Transfer mode setting

Sets the operation mode for the corresponding channel as follows.

MOD	Function
00 _B	Block/step transfer mode (initial value)
01 _B	Burst transfer mode
10 _B	Setting disabled
11 _B	Setting disabled

- Initialized to "00_B" when resetting.
- The read / write is possible.

[bit27, bit26] WS (Word Size): Transfer data size selection

Sets the transfer data size for the corresponding channel as follows. DMA performs the specified number of transfers using the data size specified in this register.

WS	Function
00 _B	Transfer by BYTE unit (initial value)
01 _B	Transfer by HALF-WORD unit
10 _B	Transfer by WORD width unit
11 _B	Setting disabled

- Initialized to "00_B" when resetting.
- The read / write is possible.

[bit25] SADM (Source-ADdr. count-Mode select) *: Transfer source address count mode setting

Specifies what to do to the transfer source address for the corresponding channel after each transfer.

Address incrementing and decrementing is performed after each transfer in accordance with the "count size for the transfer source address" (SASZ). When the transfer completes, the next access address is written to the corresponding address register (DMASA).

Accordingly, the transfer source address register is not updated until the DMA transfer completes.

To use a fixed address, set this bit to "0" or "1", and set the address count sizes (SASZ and DASZ) to "0".

SADM	Function
0	Transfer source addresses are increased (initial value).
1	Transfer source addresses are decreased.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit24] DADM (Destination-ADdr. Count-Mode select) *: Transfer destination address count mode setting

Specifies what to do to the transfer destination address for the corresponding channel after each transfer.

Address incrementing and decrementing is performed after each transfer in accordance with the "count size for the transfer destination address" (DASZ). When the transfer completes, the next access address is written to the corresponding address register (DMADA).

Accordingly, the transfer destination address register is not updated until the DMA transfer completes.

To use a fixed address, set this bit to "0" or "1", and set the address count sizes (SASZ and DASZ) to "0".

DADM	Function
0	Transfer destination address is increased (initial value).
1	Transfer destination address is decreased.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit23] DTCR (DTC-reg. Reload)*: Reload setting for the transfer count register

Controls the reload function for the transfer count register in the corresponding channel.

If this bit enables reloading, the value of the count register is reset to its initial value after transfer completes. DMA halts and waits for the next transfer request (invoked by the STRG bit or by the trigger request set by the IS bits). (The DENB bit is not cleared when this bit is "1".)

Transfer is forcibly halted when DENB=0 or DMAE=0 is set.

Disabling reloading of the count counter results in a single transfer. That is, DMA halts when the transfer is completed even if reloading is specified in the address register. In this case, the DENB bit is cleared.

DTCR	Function
0	Transfer count register reload is disabled (initial value).
1	Transfer count register reload is enabled.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit22] SADR (Source-ADdr.-reg. Reload) *: Reload setting for transfer source address register

Controls the reload function for the transfer source address register in the corresponding channel.

When reloading is enabled by this bit, the transfer source address register is returned to its initial value when transfer completes.

Disabling reloading of the count counter results in a single transfer. That is, DMA halts when the transfer is complete even if reloading is specified in the address register. In this case, the address register retains its initial value that is reloaded when DMA halts.

When this bit disables reloading, the value of the address register when transfer completes is the next access address after the final address (that is, if incrementing is enabled, it is the incremented address).

SADR	Function
0	Transfer source address register is disabled to reload (initial value).
1	Transfer source address register is enabled to reload.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit21] DADR (Dest.-ADdr.-reg. Reload)*: Reload setting for transfer destination address register

Controls the reload function for the transfer destination address register in the corresponding channel.

When reloading is enabled by this bit, the transfer destination address register is returned to its initial value when transfer completes.

Other details and functions are the same as described for bit22:SADR.

DADR	Function
0	Transfer destination address register is disabled to reload (initial value).
1	Transfer destination address register is enabled to reload.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit20] ERIE (ERror Interrupt Enable)* : Error interrupt output enabled

This bit controls whether to generate an interrupt when transfer ends if an error occurred. The nature of the error is indicated by bits DSS2 to DSS0. Note that this interrupt is not generated by all DMA termination cause. It is generated for specific cause only (see the explanation for bits DSS2 to DSS0).

ERIE	Function
0	Error interrupt request output is disabled (initial value).
1	Error interrupt request output is enabled.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit19] EDIE (EnD Interrupt Enable) *: Enable end interrupt output

Controls whether to output an interrupt when transfer ends normally.

EDIE	Function
0	End interrupt request output is disabled (initial value).
1	End interrupt request output is enabled.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit18 to bit16] DSS2 to DSS0 (Dma Stop Status)*: Transfer halt cause indication

This 3-bit code (termination code) indicates the reason why DMA transfer stopped or halted on the corresponding channel. The termination code meanings are as follows.

DSS2	Function	Interrupt generation
0	Initial value	None
1	DMA is suspended (DMAH, PAUS bit, interrupt, etc.).	None

DSS1, DSS0	Function	Interrupt generation
00	Initial value	None
01	-	None
10	Transfer halt request	Error
11	Normal completion	End

The transfer stop request is only be set when a request from a peripheral circuit is used.

Note:

The "Interrupt generation" column indicates the type of interrupt request that could be generated.

- Initialized to "000B" when resetting.
- Writing "000B" clears the bits.
- Although both reading and writing are permitted, only "000B" is meaningful when writing.

[bit15 to bit8] SASZ7 to SASZ0 (Source Addr count SiZe) *: Count size specification for the transfer source address

Specifies how much to increment or decrement the transfer source address (DMASA) for the corresponding channel after each transfer. The value specified by these bits determines by how much the address is incremented or decremented for each transfer. Whether to increment or decrement the address is specified by the transfer source address count mode (SADM).

SASZ	Function
00 _H	Address fixed
01 _H	Transfer by byte unit
02 _H	Transfer by half-word unit
04 _H	Transfer by word unit
Other than above	Setting disabled

- Initialized to "00000000_B" when resetting.
- The read / write is possible.
- If setting other than a fixed address, ensure that the setting matches the transfer data size (WS).

[bit7 to bit0] DASZ7 to DASZ0 (Des Addr count SiZe) *: Count size specification for the transfer destination address

Specifies how much to increment or decrement the transfer destination address (DMADA) for the corresponding channel after each transfer. The value specified by these bits determines by how much the address is incremented or decremented for each transfer. Whether to increment or decrement the address is specified by the transfer destination address count mode (DADM).

DASZ	Function
00 _H	Address fixed
01 _H	Transfer by byte unit
02 _H	Transfer by half-word unit
04 _H	Transfer by word unit
Other than above	Setting disabled

- Initialized to "00000000_B" when resetting.
- The read / write is possible.
- If setting other than a fixed address, ensure that the setting matches the transfer data size (WS).

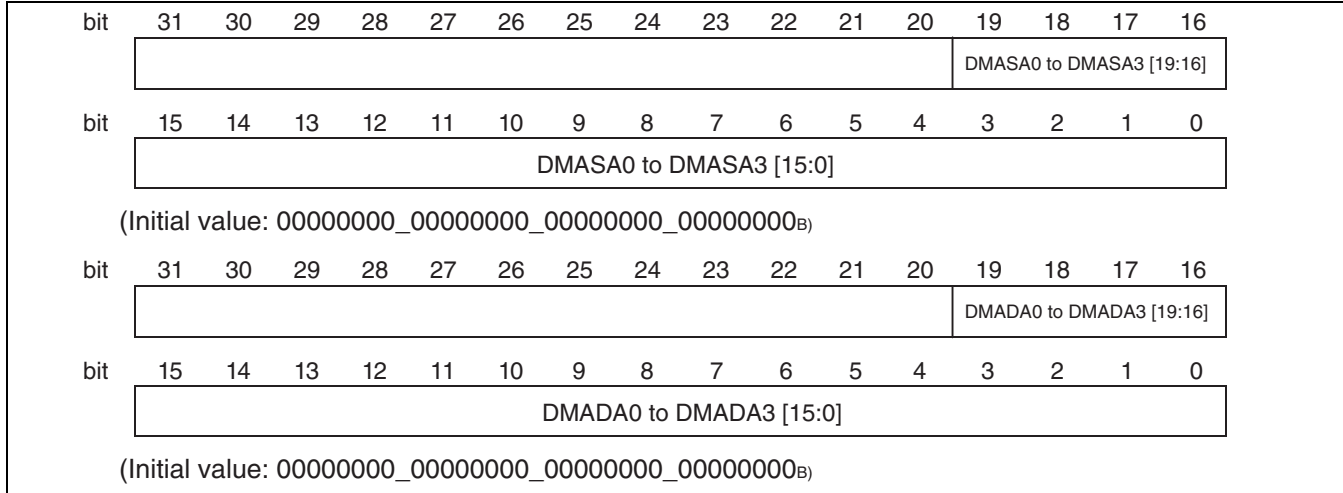
■ DMAC-ch.0, ch.1, ch.2, ch.3, ch.4 Transfer Source/Transfer Destination Address Setting Registers

[DMASA0 to DMASA4/DMADA0 to DMADA4]

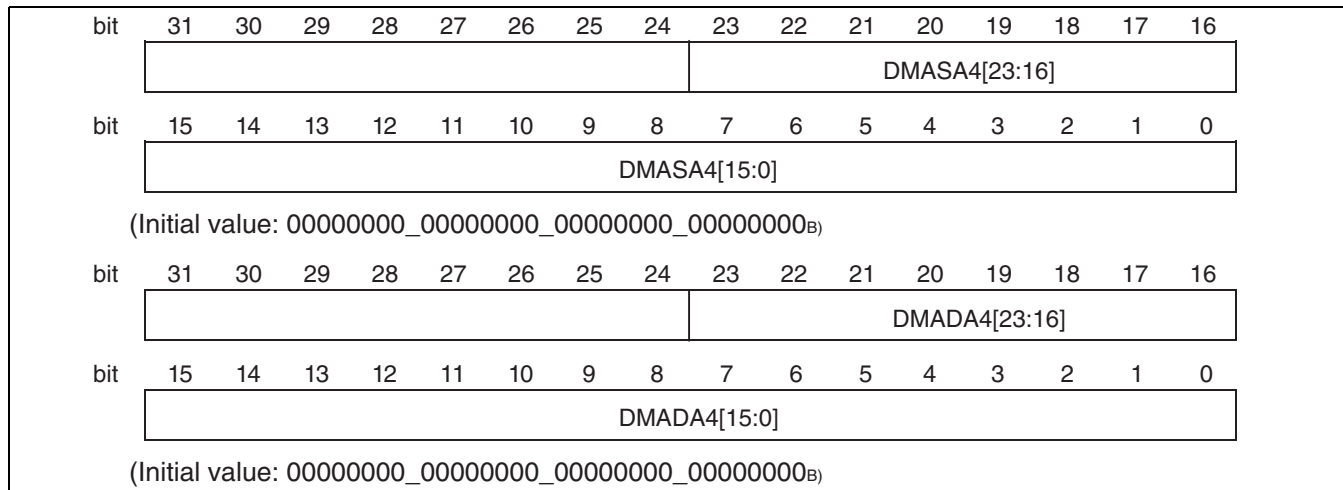
These registers control the operation of each DMAC channel. A separate register is provided for each channel.

The function of each bit is as follows.

- ch.0 to ch.3



- ch.4



These registers store the source and destination addresses for DMA transfer. The registers for ch.0 to ch.3 are 20-bit and the registers for ch4 are 24-bit.

[bit31 to bit0] DMASA (DMA Source Addr) *: Transfer source address setting

Sets the transfer source address.

[bit31 to bit0] DMADA (DMA Destination Addr) *: Transfer destination address setting

Sets the transfer destination address.

When DMA transfer starts, the contents of these registers are copied to the count buffer for the dedicated DMA address counter and the address count is updated after each transfer in accordance with the settings. When DMA transfer completes, the value of the counter buffer is written back to these registers and the DMA operation ends. Accordingly, you cannot read the value of the address counter during DMA operation.

Each register has its own reload register. When used on channels for which reloading the transfer source and destination address registers is enabled, the registers are automatically restored to their initial values when transfer completes. However, this has no effect on other address registers.

- Initialized to "00000000_00000000_00000000_00000000_B" when resetting.
- The read / write is possible. Always use 32-bit access to read or write to these registers.
- During transfer, reading returns the address before transfer started. After transfer completes, reading returns the next access address. You cannot read the reload value. Accordingly, you cannot read the transfer address in real-time.
- Set "0" to the non-existent upper bits.

Note:

Do not set the address of DMAC registers in these registers. Performing DMA transfer to registers in the DMAC is not permitted.

■ DMAC-ch.0, ch.1, ch.2, ch.3, ch.4 Overall DMAC Control Register

[DMACR]

This register controls the overall operation of all five DMAC channels. Always use byte access to read or write to this register.

The function of each bit is as follows.

bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DMAE	-	-	PM01	DMAH[3:0]				-	-	-	-	-	-	-	-
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

(Initial value: 0XX00000XXXXXXXXXXXXXXXXXXXXXXXXXXXX_B)

[bit31] DMAE (DMA Enable): DMA enabling operations

Controls operation for all DMA channels.

When DMA operation is disabled by this bit, transfer is disabled for all channels regardless of operation state or any start/stop settings set for individual channels. Any requests on channels for which transfer is in progress are cancelled and transfer halts at the block boundary. When disabled, any operation to start transfer on any channel is ignored.

When DMA operation is enabled by this bit, starting and stopping each channel is available. Using this bit to enable DMA operation does not actually start transfer for any channel.

Although writing "0" to this bit forcibly halts DMA, always use the DMAH(3:0) bits (DMACR: bit27 to bit24) to pause DMA before forcibly halting (writing "0"). If DMA transfer is forcibly halted without pausing, the DMA transfer halts, but the validity of the transferred data is not guaranteed. Use the DSS(2:0) bits (DMACB: bit18 to bit16) to check whether transfer has finished.

DMAE	Function
0	Disables operation for all DMA channels (initial value).
1	Enables operation for all DMA channels.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit28] PM01 (Priority Mode ch0, 1 robin): Alternate channel priority

This setting causes the priorities for ch0 and ch1 to be alternated after each transfer.

PM01	Function
0	Priority level fixed (ch0 > ch1) (initial value)
1	Priority level alternated (ch1 > ch0)

- Initialized to "0" when resetting.
- The read / write is possible.

[bit27 to bit24] DMAH (DMA Halt): DMA suspended

Pauses DMA for all DMA channels. Setting these bits pauses DMA transfer on all channels until the bits are cleared again.

If these bits are set before enabling DMA, all channels remain paused.

Any transfer requests that occur for channels with DMA transfer enabled (DENB=1) while these bits are set are valid, but transfer does not start until the bits are cleared.

DMAH	Function
0000 _B	Enables operation for all DMA channels (initial value).
Other than 0000 _B	Suspends operation for all DMA channels.

- Initialized to "0" when resetting.
- The read / write is possible.

[bit30, bit29, bit23 to bit0] (Reserved): Unused bit.

The read value is undefined.

16.3 Operation Explanation

DMAC is a multifunctional DMA controller for controlling the high-speed data transfer without using CPU instruction.

■ Overview of Operation

This block is a multi-function DMA controller able to transfer data at high speed without using CPU instructions.

● Main operation

- The operation of each function can be set independently for each channel.
- Once enabled, a channel does not actually start transfer until the specified transfer request is detected.
- On detecting a transfer request, the DMAC outputs a DMA transfer request to the bus controller and starts transfer on receiving bus access rights from the bus controller.
- The transfer sequence is determined by the mode which can be set independently for each channel.

● Transfer mode

Each DMA channel operates transfer in accordance with the transfer mode set by the MOD1 and MOD0 bits in its DMACB register.

Block/step transfer

Only transfers one block of data for each transfer request. Once the transfer completes, DMA removes its transfer request from the bus controller until the next transfer request is received.

Size of transfer block: Block size specified in DMACA:BLK3 to BLK0

Burst transfer

On receiving a transfer request, transfer continues for the specified number of transfers.

Specified number of transfers: Block size \times transfer count

(DMACA:BLK3 to BLK0 \times DMACA:DTC15 to DTC0)

● Transfer type

Two-cycle transfer (Normal transfer)

The DMA controller treats a read and write operation as a single unit.

The DMAC reads the value from the address set in the transfer source register and then writes it to the address in the transfer destination register.

● Transfer address

The following types of addressing are supported. This can be specified independently for each channel transfer source and destination.

Address setting for two-cycle transfer

Accesses the addresses read from the registers in which the address values have previously been set (DMASA and DMADA).

On receiving a transfer request, DMA stores the addresses from these registers in temporary buffers before starting the transfer.

After each transfer (access), the address counter is used to generate the next access address (based on whether incrementing, decrementing, or constant-address is specified) and this new value is restored in the temporary buffer. The contents of the temporary buffers are written back to the registers (DMASA and DMADA) after transfer of each block completes.

Accordingly, the values in the address registers (DMASA and DMADA) are only updated after each block transfer and cannot be used to determine the current transfer address in real-time.

● Number of transfers and ending transfer

Number of transfers

The transfer count register is decremented (-1) after transfer of each block completes. When the transfer count register reaches "0" indicating that the specified number of transfers have been performed, the DMAC indicates the termination code and halts or restarts DMA.

Like the address registers, the transfer count register is only updated after each block is transferred.

If reloading the transfer count register is disabled, transfer ends. If enabled, the register is reloaded with its initial value and the DMAC waits for transfer to restart. (DMACB:DTCR)

The end of transfer

Transfer can be ended by the following cause. When transfer ends, cause is indicated as termination code. (DMACB:DSS2 to DSS0)

- Specified number of transfers completed (DMACA:BLK3 to BLK0 × DMACA:DTC15 to DTC0) => Normal end
- Transfer halt request received from peripheral circuit => Error
- Reset occurred => Reset

A transfer halt cause code (DSS) is set for each end cause and a transfer complete interrupt or transfer error interrupt can be generated.

16.4 Setting Up Transfer Request

The following two types of transfer requests can be used to start DMA transfer. The software request can be used at any time regardless of other request settings.

■ Internal Peripheral Request

The transfer request is generated by an interrupt from an internal peripheral circuit.

Which peripheral interrupt requests generates a transfer request can be set independently for each channel (DMACA: IS4 to IS0 = 1xxxx_B).

Note:

As interrupt request used to generate transfer requests are also passed to the CPU as interrupt requests, always disable these interrupts in the interrupt controller. (ICR register)

■ Software Request

Writing to the trigger bit in the register generates the transfer request. (DMACA:STRG)

This is independent of the above transfer requests and is always available.

If a software request is set at the same time as transfer is enabled, the DMA transfer request to the bus controller is outputted immediately and transfer starts.

Note:

If a software request is issued to a channel that is also set the internal peripheral request, a factor clear signal is sent to the peripheral when the transfer completes. As this may clear an existing transfer request, do not use the software transfer request function in this case.

16.5 Transfer Sequence

The transfer type and transfer mode which determine the operation sequence and similar after DMA transfer started can be set independently for each channel (setting of DMACB:TYPE [1:0], MOD [1:0]).

■ Transfer Sequence Selection

The following sequences can be selected by register settings.

- Burst two-cycle transfer
- Block/step transfer two-cycle transfer

■ Burst Two-cycle Transfer

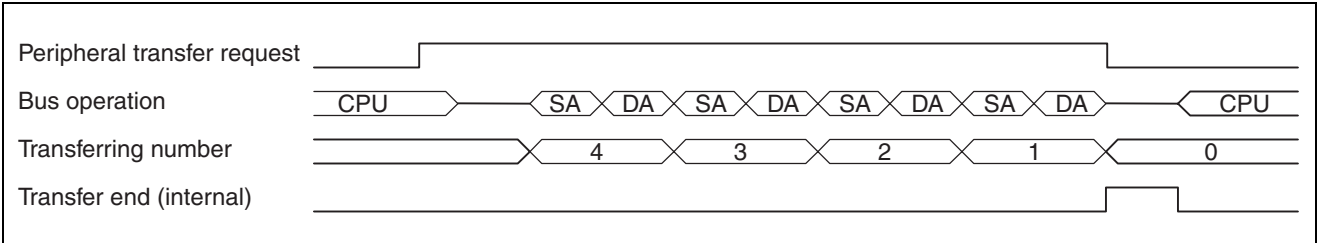
The specified number of transfers is performed each time transfer is invoked. In the case of two-cycle transfer, the transfer source and destination addresses can be specified as 20-bit addresses for ch0 to ch3, and as 24-bit addresses for ch4.

Either a transfer request from a peripheral function or a software can be used to invoke transfer.

[Burst transfer characteristics]

- Each time a transfer request is received, transfer continues until the transfer count register reaches "0".
- The number of transfers is the block size multiplied × the number of blocks to be transferred. (DMACA:BLK3 to BLK0 × DMACA:DTC15 to DTC0)
- If another transfer request is received during a transfer, the request is ignored.
- When the reload function is enabled for the transfer count register, a subsequent transfer request is accepted only after transfer completes.
- If a transfer request for a channel with a higher priority is received during a transfer, the DMAC changes channel at the next block boundary and the transfer does not restart until the request on the higher priority channel is cleared.

Figure 16.5-1 Burst Transferring Example at Peripheral Transfer Request, Block Number = 1, Transferring Number = 4



■ Step/block Transfer Two-cycle Transfer

For step or block transfer (only transfer the specified number of blocks for each transfer request), the transfer source and destination addresses can be specified as 20-bit addresses for ch0 to ch3, and as 24-bit addresses for ch4.

● Step transfer

The step transfer sequence is used if the block size is set to "1".

● Step transfer characteristics

- Each time a transfer request is received, one transfer is performed, and then the transfer request is cleared and transfer halts. (The DMA transfer request is removed from the bus controller.)
- If another transfer request is received during a transfer, the request is ignored.
- If a transfer request for a channel with a higher priority is received during a transfer, the DMAC changes channel and starts the next transfer after the previous transfer completes. For step transfer, priority is only meaningful for the case when transfer requests are generated simultaneously.

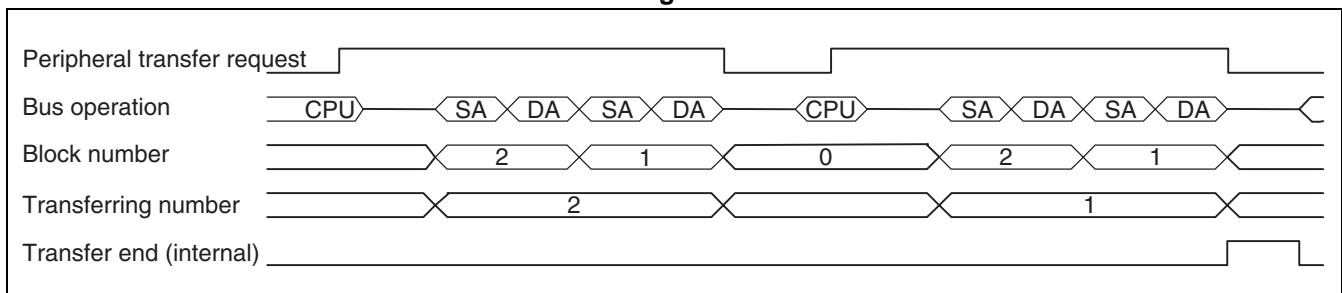
● Block transfer

The block transfer sequence is used if the block size is set to a value other than "1".

● Block transfer characteristics

Except for the fact that each transfer consists of multiple transfer cycles (specified by the number of blocks), the operation is the same as for step transfer.

Figure 16.5-2 Block Transferring Example at Peripheral Transfer Request, Block Number = 2, Transferring Number = 2



16.6 Overview of DMA Transfer

This section explains the overview of the DMA transfer.

■ Block Size

- The value set in the block size setting register specifies the volume of data for each transfer operation (× data width).
- The size of the data transferred in each transfer cycle is fixed the value specified by data width, and each transfer consists of the number of transfer cycles specified by the block size.
- During a transfer, if a transfer request for a channel with a higher priority is received or if a transfer pause request is issued, block transfer does not halt unless set to the boundary of a transfer unit. Although this prevents undesirable splitting or pausing within a data block, it causes the response to be slower if the block size is large.
- In the case of a reset, transfer halts immediately and therefore the validity of the transferred data cannot be guaranteed.

■ Reload Operation

The following three reload functions can be configured for each channel in this module.

● Transfer count register reload function

When the specified number of transfers completes, the transfer count register is reset with its initial value and the DMAC waits for the next trigger.

Use this setting to perform any of the transfer sequences repeatedly.

If reloading is not enabled, the count register remains at zero after the specified number of transfers complete and no further transfers are performed.

● Reload function for transfer source address register

The transfer source address register is reset with its initial value after the specified number of transfers complete.

Use this setting if repeatedly performing a transfer from a fixed region in the transfer source address range.

If reloading is not enabled, the value of the next transfer address remains set in the transfer source address register after the specified number of transfers complete. Use this setting if the address range is not fixed.

● Reload function for transfer destination address register

The transfer destination address register is reset with its initial value after the specified number of transfers complete.

Use this setting if repeatedly performing a transfer to a fixed region in the transfer destination address range.

Only enabling the reload functions for transfer source and destination registers does not restart after the specified number of transfers complete. It only causes the address registers to be reset.

Notes:

Special example of operation modes and reload operation

- If you want to halt transfer after it completes and to restart after another input is detected, do not use the reload function.
 - When using burst, block, or step transfer modes, transfer halts after the reload is performed at the end of the transfer operation, and no further transfer is performed until a new transfer request input is detected.
-

■ Addressing Mode

Specify the transfer destination and transfer source addresses independently for each transfer channel.

The following procedure can be used to set the registers. Set up the registers in accordance with the transfer sequence.

● Address registers

In two-cycle transfer mode, set the transfer source address in the transfer source address setting register (DMASA) and set the transfer destination address in the transfer destination address setting register (DMADA).

Features of address register

These registers are 20-bit for ch0 to ch3 and 24-bit for ch4.

Function of address register

- The registers are read at each time when an access is performed and outputted on the address bus.
 - At the same time, the address counter is used to calculate the address for the next access and the result of this calculation is set in the address register.
 - The address calculation is selected by increment or decrement independently for each channel, transfer destination and source. The increment or decrement width of address depends on the value of address count size specification register. (DMACB:SASZ, DASZ)
 - When the reload function is not enabled for an address register, the result of the final address calculation remains in the register after the transfer ends.
 - If the reload function is enabled, the initial value of the address is reloaded.
-

Notes:

- Transfer continues even if an overflow or underflow occurs for the full 20-bit or 24-bit address calculation. When setting up each channel, take care to ensure that overflow or underflow does not occur.
 - Do not set the addresses of registers in the DMAC in the address registers.
-

■ Data Type

The data length (data width) transferred in each transfer cycle can be selected from the following options.

- byte
- halfword
- word

As word boundaries still apply during DMA transfer, setting a transfer source or destination address setting that conflicts with the data length causes the lower bits to be ignored.

- word: Four bytes starting from an actual access addresses with the lower 2-bit = 00_B.
- halfword: Two bytes starting from an actual access addresses with the least significant bit = 0.
- byte: The specified address is used as the actual access address.

If the lower bits of the transfer source and destination addresses do conflict, the specified address is outputted on the internal address bus without modification, but the actual bus access occurs at the address corrected in accordance with the above rules.

■ Control of Number of Transfers

The number of transfers can be set to any 16-bit value (1 to 65536 transfers). Set the number of transfers in the transfer count register (DMACA:DTC).

The register value is copied to a temporary buffer when transfer starts and is decremented by the transfer count counter. When the count value reaches zero indicating that the specified number of transfers has completed, the channel either halts transfer or waits for the next trigger (if reload is enabled).

Features of the transfer count register

- Each register consists of 16 bits.
- Each register has its own reload register.
- Setting the register to "0" results in transfer being performed 65536 times.

Reload operation

- Only used for registers with a reload function and for which the reload function is enabled.
- The initial value of the count register is saved in the reload register when transfer starts.
- Once the operation of the transfer count counter causes the count to reach "0", a signal indicating transfer completion is outputted, and then the initial value is read from the reload register and written to the count register.

■ CPU Control

When a DMA transfer request is received, DMA issues a transfer request to the bus controller.

The bus controller assigns usage rights for the internal bus to DMA at the next appropriate timing in bus operation, and then DMA transfer starts.

● DMA transfer and interrupts

- During DMA transfer, interrupts are generally not received until the transfer ends.
- If a DMA transfer request occurs during interrupt processing, the transfer request is received and interrupt processing halts until the transfer completes.
- Exceptions to this are the NMI request and interrupt requests with a level higher than the hold override level set in the interrupt controller. In these cases, the DMAC temporarily removes its transfer request from the bus controller at the next transfer boundary (one of block) and pauses the transfer until the interrupt request is cleared. The transfer request is held internally during this time. Once the interrupt request is cleared, the DMAC issues another transfer request to the bus controller to obtain bus usage rights and restarts DMA transfer.

● Overriding DMA

- On the FR family, if an interrupt with a higher priority occurs during DMA transfer, DMA transfer halts and control branches to the interrupt routine. This mechanism remains active while the interrupt request is present. The DMA halt is not operated as soon as the interrupt is cleared, and so the DMA transfer restarts from inside the interrupt handler routine. Accordingly, in handler routines for interrupts with a level that causes DMA transfer to be halted, use the DMA halt function if you wish to prevent DMA from restarting as soon as the interrupt is cleared. The DMA halt function is invoked by writing a non-zero value to the DMAH3 to DMAH0 bits in the DMA overall control register. The override is cleared by setting the bits back to zero.
- This function is primarily used in interrupt handler routines. Increment the DMA halt register by one in the interrupt handler routine before clearing the interrupt. This prevents any subsequent DMA transfer. When interrupt processing is completed, decrement the DMAH3 to DMAH0 bits by one before exiting the routine. If multiple interrupts have occurred, DMA transfer continues to be suppressed since the DMAH3 to DMAH0 bits are not "0" yet. If a single interrupt has occurred, the DMAH3 to DMAH0 bits become "0". DMA requests are then enabled immediately.

Notes:

- Since the register has only four bits, this function cannot be used for multiple interrupts exceeding 15 levels.
 - Be sure to assign the priority of the DMA tasks at a level that is at least 15 levels higher than other interrupt levels.
-

■ Operation Start

Starting of DMA transfer is controlled independently for each channel, but before transfer starts, the operation of all channels needs to be enabled.

● Operation enable of all channels

Before activating each DMAC channel, operation for all channels needs to be enabled in advance with the DMA operation enable bit (DMAE of DMACR). All start settings and transfer requests that occurred before operation is enabled are invalid.

● Transfer starting

The transfer operation can be started by the operation enable bit of the control register for each channel. If a transfer request to an activated channel is accepted, the DMA transfer operation is started in the specified mode.

● Starting from a temporary stop

If a temporary stop occurs before starting with channel-by-channel or all-channel control, the temporary stopped state is maintained even though the transfer operation is started. If transfer requests occur in the meantime, they are accepted and retained. When temporary stopping is released, transfer is started.

■ Transfer Request Acceptance and Transfer

- Sampling for transfer requests set for each channel starts after starting.
- If peripheral interrupt start is selected, DMAC continues the transfer until all transfer requests are cleared. When they are cleared, DMAC stops the transfer after one transfer unit (peripheral interrupt starting).
Since peripheral interrupts are handled as level detection, use interrupt clear by DMA to handle the interrupts.
- Transfer requests are always accepted while other channel requests are being accepted and transfer performed. The channel that will be used for transfer is determined for each transfer unit after priority has been checked.

■ Peripheral Interrupt Clear by DMA

- This DMA has a function that clears peripheral interrupts. This function works when peripheral interrupt is selected as the DMA start source (when IS4 to IS0 = 1xxxx_B).
- Peripheral interrupts are cleared only for the set start sources. That is, only the peripheral functions set by IS4 to IS0 are cleared.

● Generating timing of interrupt clear

The timing for clearing an interrupt depends on the transfer mode. (Refer to "16.7 Operation Flow".)

Block/step transfer

If block transfer is selected, a clear signal is generated after one block (step) transfer.

Burst transfer

If burst transfer is selected, a clear signal is generated after transfer is performed the specified number of times.

■ Temporary Stop

The temporary stopping of DMA transfer occurs in the following cases.

- Setting of temporary stopping by writing to the control register (set independently for each channel or all channels simultaneously)
If temporary stopping is set using the temporary stop bit, transfer on the corresponding channel is stopped until release of temporary stopping is set again. You can check the DSS bits for temporary stopping.
Transfer is restarted when temporary stopping is canceled.
- NMI/hold suppress level interrupt processing
If an NMI request or an interrupt request with a higher level than the hold suppress level occurs, all channels on which transfer is in progress are stopped at the boundary of the transfer unit and the bus right is opened to give priority to NMI/interrupt processing. Transfer requests accepted during NMI/interrupt processing are retained, initiating a wait for completion of NMI processing.
Channels for which requests are retained restart transfer after NMI/interrupt processing is completed.

■ Operation End/Stopping

The end of DMA transfer is controlled independently for each channel. It is also possible to disable operation for all channels at once.

● The end of transfer

If reloading is disabled, transfer is stopped, "Normal end" is displayed as the end code, and all transfer requests are disabled after the transfer count register becomes "0". (Clear the DENB bit of DMACA.)

If reloading is enabled, the initial value is reloaded, "Normal end" is displayed as the end code, and a wait for transfer requests starts again after the transfer count register becomes "0". (Do not clear the DENB bit of DMACA)

● All channels operating disabled

If the operation of all channels is disabled with the DMA operation enable bit DMAE, all DMAC operations, including operations on active channels, are stopped. Then, even if the operation of all channels is enabled again, no transfer is performed unless a channel is restarted. In this case, no interrupt whatever occurs.

■ Stopping due to Error

In addition to cause other than normal end after transfer for the number of times specified, stopping as the result of various types of errors and the forced stopping are provided.

● Transfer stop requests from peripheral circuits

Depending on the peripheral circuit that outputs a transfer request, a transfer stop request is issued when an error is detected (Example: Error when data is received at or sent from a communications system peripheral).

The DMAC, when it receives such a transfer stop request, displays "Transfer stop request" as the end code and stops the transfer on the corresponding channel.

Notes:

- For the availability of a transfer stop request from peripheral circuits, see the description of specifications for transfer factor selection bit, which is bit28 to bit24 (IS4 to IS0) of DMACA register.
 - For details of the conditions under which a transfer stop request is generated, see the specifications for each peripheral circuit.
-

■ DMAC Interrupt Control

Independent of peripheral interrupts that become transfer requests, interrupts can also be outputted for each DMAC channel.

- Transfer end interrupt: Occurs only when operation ends normally.
- Error interrupt: Transfer stop request due to a peripheral circuit (error due to a peripheral). All of these interrupts are outputted according to the meaning of the end code.

An interrupt request can be cleared by writing "000_B" to DSS2 to DSS0 (end code) of DMACS.

Be sure to clear the end code by writing "000_B" before restarting.

If reloading is enabled, the transfer is automatically restarted. At this point, however, the end code is not cleared and is retained until a new end code is written when the next transfer ends.

Since only one end source can be displayed in an end code, the result after considering the order of priority is displayed when multiple sources occur simultaneously. The interrupt that occurs at this point conforms to the displayed end code.

The following shows the priority for displaying end codes (in order of decreasing priority):

- Reset
- Clear by writing "000_B".
- Peripheral stop request
- Successful completion
- Channel selection and control

■ DMA Transfer in Sleep Mode

The DMAC can also operate in sleep mode.

If you anticipate operations during sleep mode, note the following:

- Since the CPU is stopped, DMAC registers cannot be rewritten. Make settings before sleep mode is entered.
- The sleep mode is released by an interrupt. Thus, if a peripheral interrupt is selected as the DMAC start source, interrupts must be disabled by the interrupt controller.

Similarly, if you do not want to release sleep mode with a DMAC end interrupt, disable DMAC end interrupts.

■ Channel Selection and Control

Up to 5 channels can be simultaneously set as transfer channels. In general, an independent function can be set for each channel.

● Priority level for channels

Since DMA transfer is possible only on 1 channel at a time, priority must be set for the channels.

Two modes, fixed and rotation, are provided as the priority settings and can be selected for each channel group (described later).

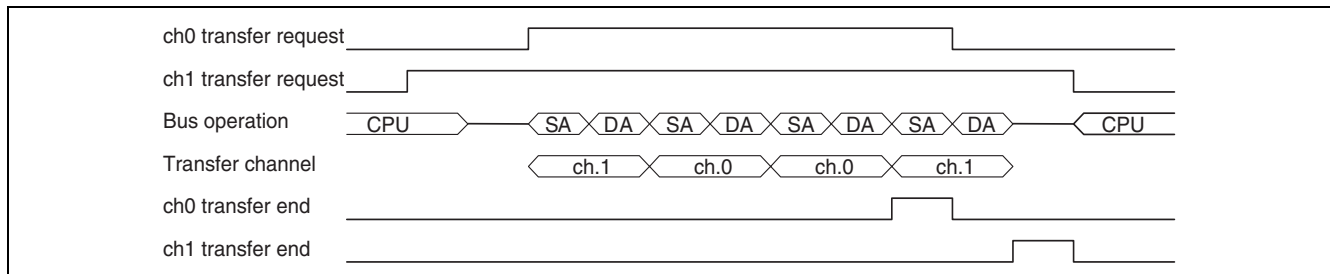
(1) Fixed mode

The priority is fixed by channel number in ascending order.

(ch.0 > ch.1 > ch.2 > ch.3 > ch.4)

If a transfer request with a higher priority is received during a transfer, the transfer channel becomes the channel with the higher priority when the transfer for the transfer unit (number set in the block size specification register x data width) ends.

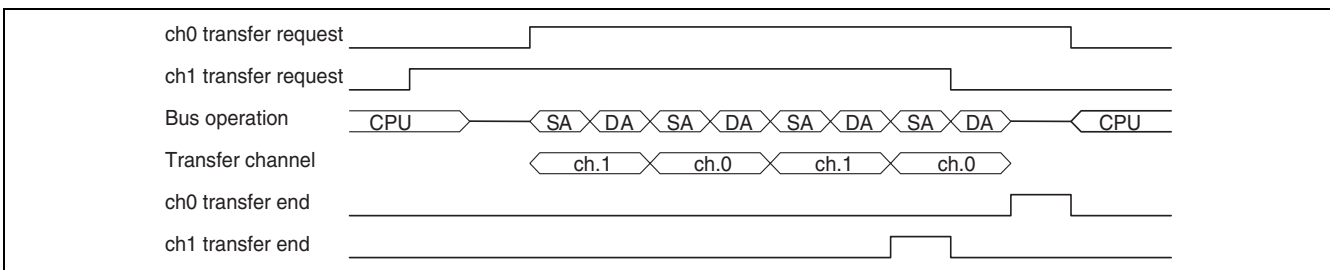
When higher priority transfer is completed, transfer is restarted on the previous channel.



(2) Rotation mode (between ch.0 and ch.1 only)

When operation is enabled, the initial states have the same order that they would have in fixed mode, but at the end of each transfer operation, the priority of the channels is reversed. Thus, if more than one transfer request is outputted at the same time, the channel is switched after each transfer unit.

This mode is effective when continuous or burst transfer is set.



● Channel group

Set the selection priority as explained in the table below.

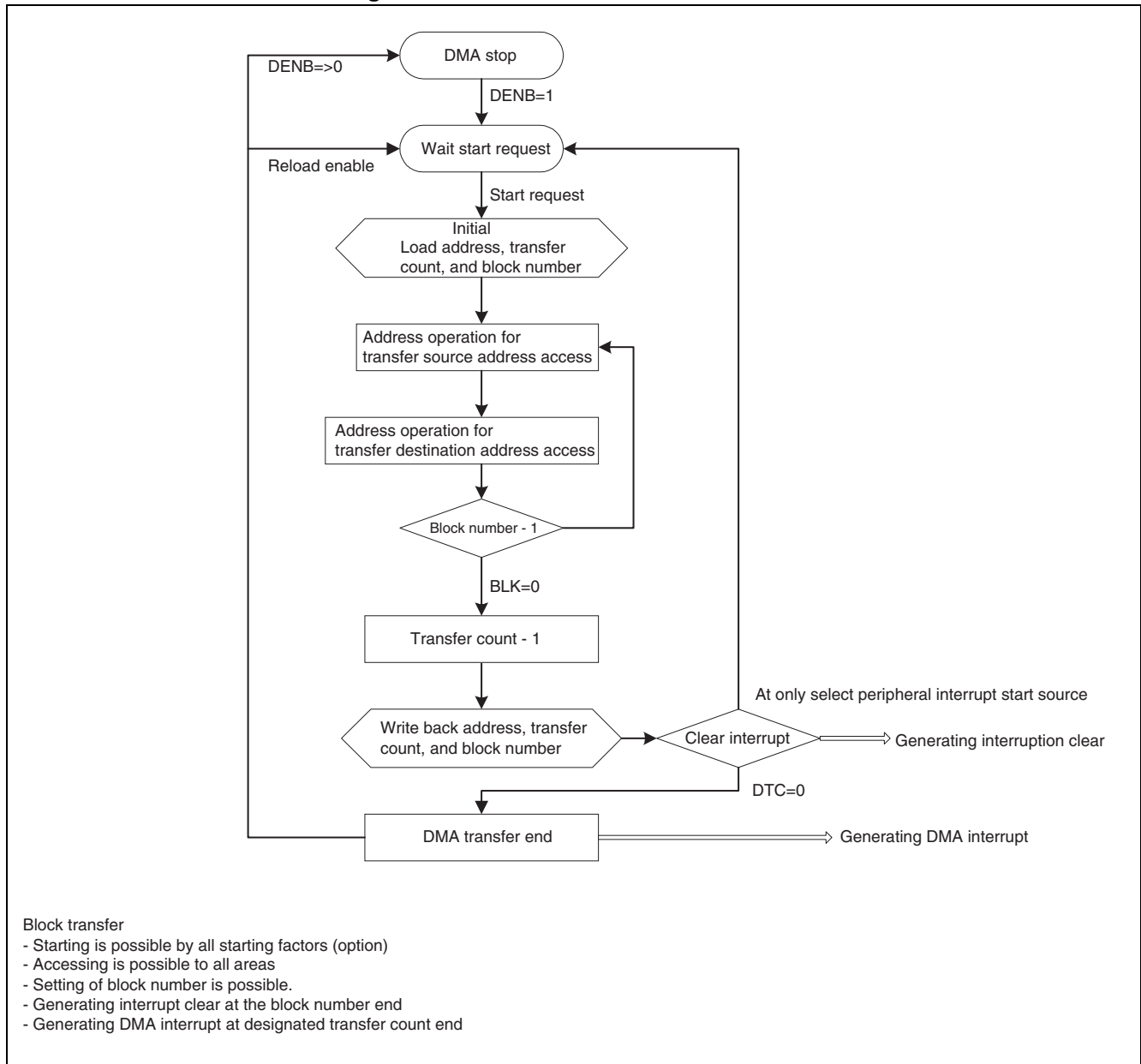
Mode	Priority	Remark
Fixed	ch.0>ch.1	
Rotation	ch.0>ch.1 ↑ ↓ ch.0<ch.1	The initial states have the upper order. When the upper part is transferred, it is reversed.

16.7 Operation Flow

This section shows the flowchart of the DMAC block transfer and burst transfer.

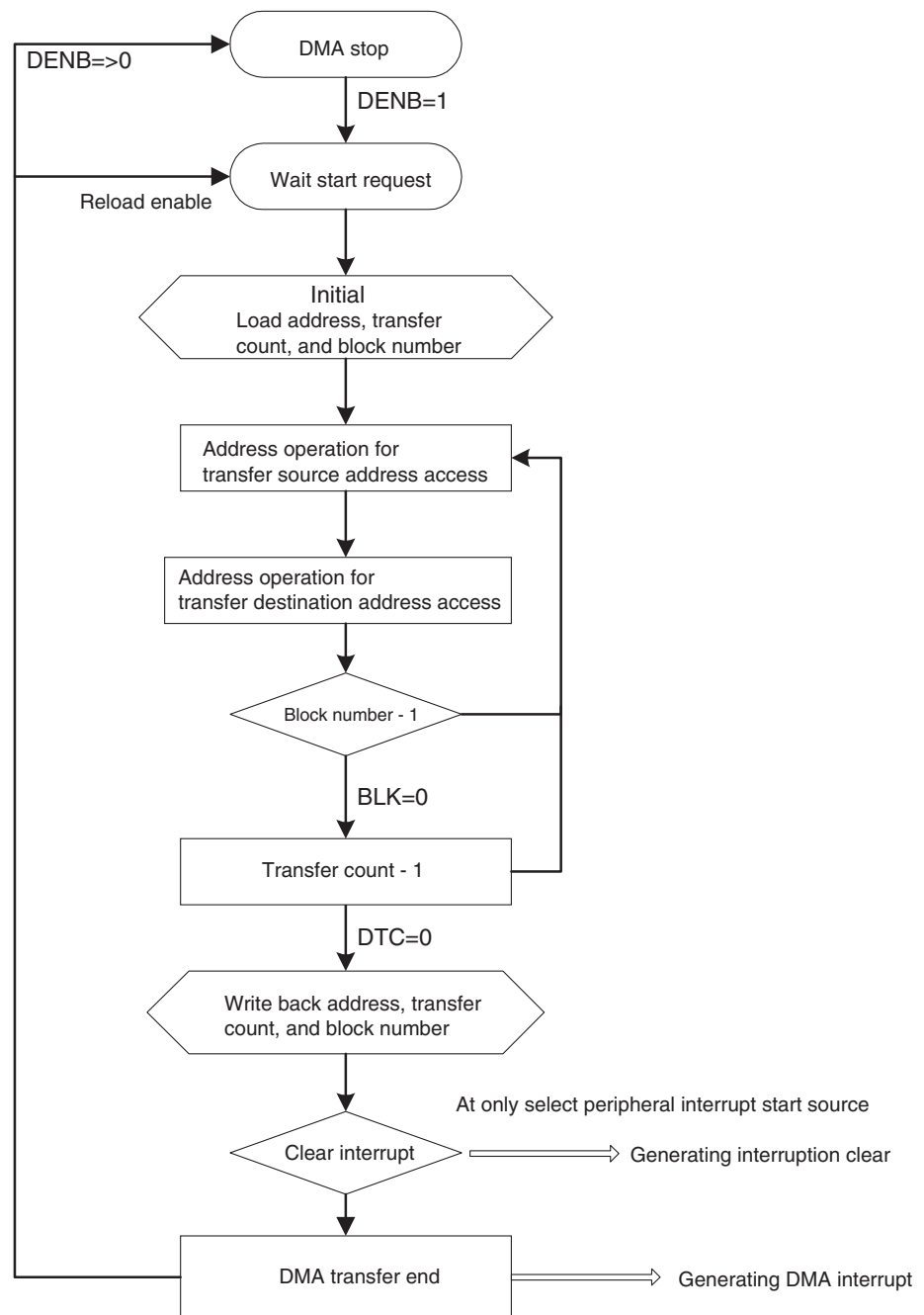
■ Block Transfer

Figure 16.7-1 Flowchart of Block Transfer



■ Burst Transfer

Figure 16.7-2 Flowchart of Burst Transfer



Burst transfer

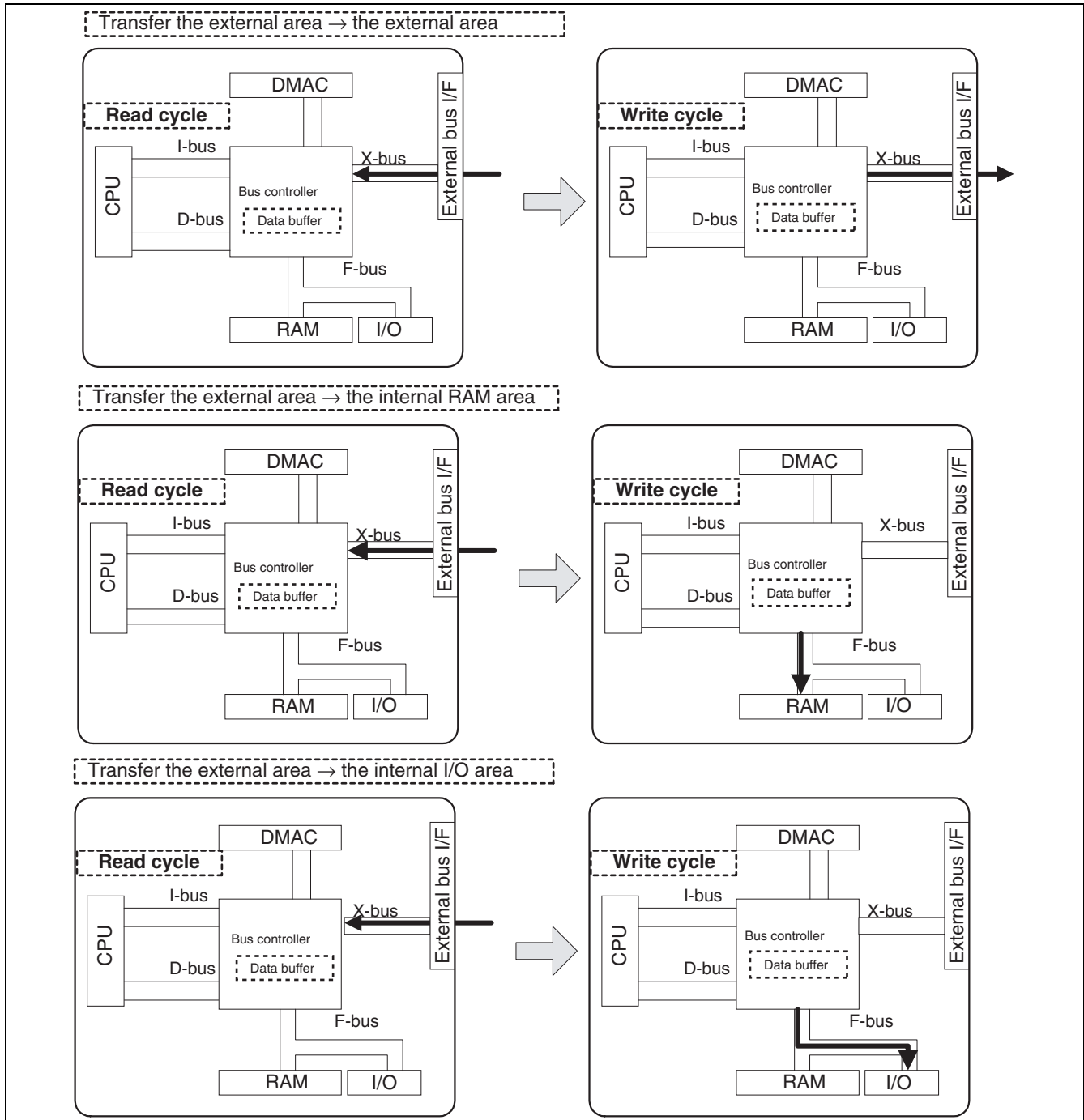
- Starting is possible by all starting factors (option)
- Accessing is possible to all areas
- Setting of block number is possible.
- Generating interrupt clear at designated transfer number end and generating DMA interrupt.

16.8 Data Path

This section shows the data transfer example of using DMAC (other sets are omitted).

■ Data Operation at Two-cycle Transfer

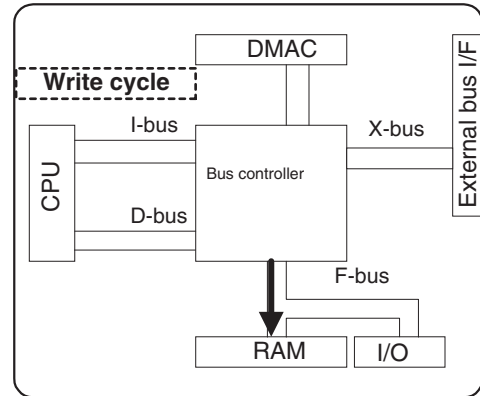
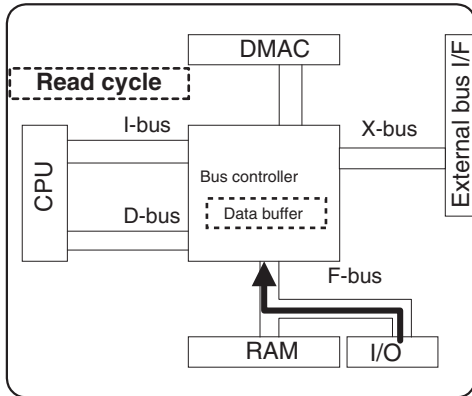
- Six types of transferring example are shown in the following;
(Other combinations are omitted.)



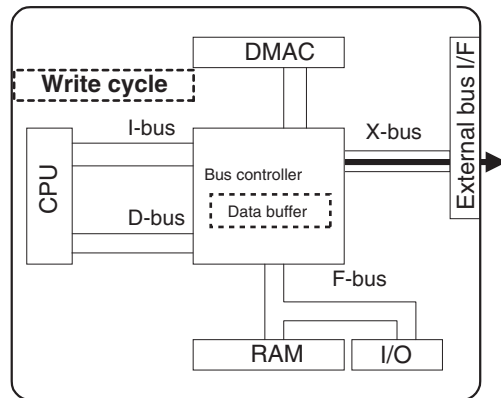
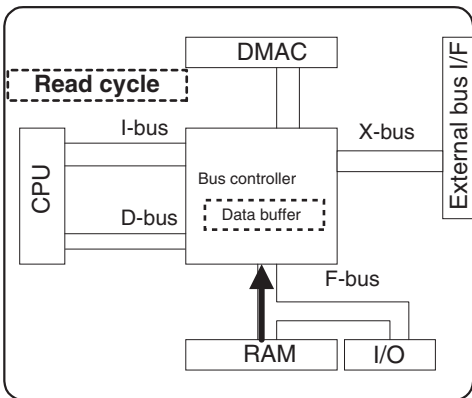
(Continued)

(Continued)

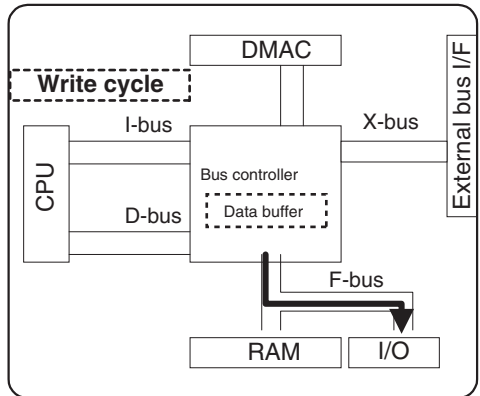
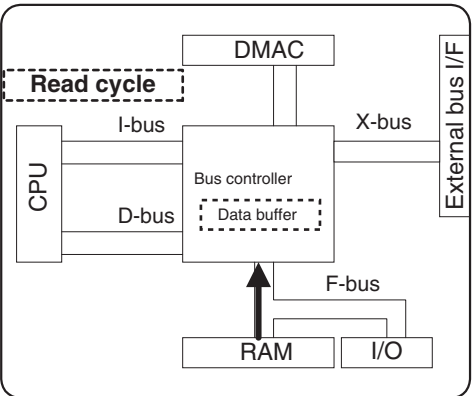
Transfer the internal I/O area → the internal RAM area



Transfer the internal RAM area → the external area



Transfer the internal RAM area → the internal I/O area



CHAPTER 17

FLASH MEMORY

This chapter describes an overview of flash memory and the configuration and functions of registers, access modes, automatic algorithm and sector protect operations.

- 17.1 Overview of Flash Memory
- 17.2 Flash Memory Registers
- 17.3 Access Modes of Flash Memory
- 17.4 Starting the Flash Memory Automatic Algorithm
- 17.5 Automatic Algorithm Execution Status
- 17.6 Sector Protect Operation

17.1 Overview of Flash Memory

The MB91F264B contains 256 KB (2 M bits) flash memory. This internal flash memory can be erased in all sectors together or in sector units by a single power supply voltage of 3.3 V, and can be written in halfword (16 bits) units via the FR-CPU.

■ Overview of Flash Memory

This flash memory is an internal 256 KB flash memory that operates on 3.3 V. This is the same (except for the capacity and some sector structures) as the Fujitsu MBM29LV400TC 4 M bits (512K bytes × 8 / 256K bytes × 16) flash memory, and supports writing with a device-external ROM writer. In addition to the features equivalent to the MBM29LV400TC, instructions and data can be read in word units (32 bits) when the flash memory is used as FR-CPU internal ROM, enabling high speed device operation.

Along with this manual, refer to the MBM29LV400TC Data Sheet.

This flash memory supports the following features by combining the flash memory macro and the FR-CPU interface circuits:

- Features for use as CPU memory, for storing programs and data
 - Accessibility through 32-bit bus width when used as ROM
 - Allowing read, write, and erase (automatic program algorithm^{*}) by the CPU
 - Features of a single flash memory product equivalent to MBM29LV400TC
 - Allowing read, write, and erase (automatic program algorithm^{*}) by a ROM writer
- ^{*}: Automatic program algorithm= Embedded Algorithm™

This section explains use of the flash memory accessed from the FR-CPU.

For information on using the flash memory accessed from a ROM writer, see the instruction manual provided with the ROM writer.

■ Automatic Algorithm Execution Status

When the automatic algorithm is started in CPU programming mode, the operation status of the automatic algorithm can be checked using the internal ready signal (RDY). The level of this signal can be read as the RDY bit in the flash memory status register.

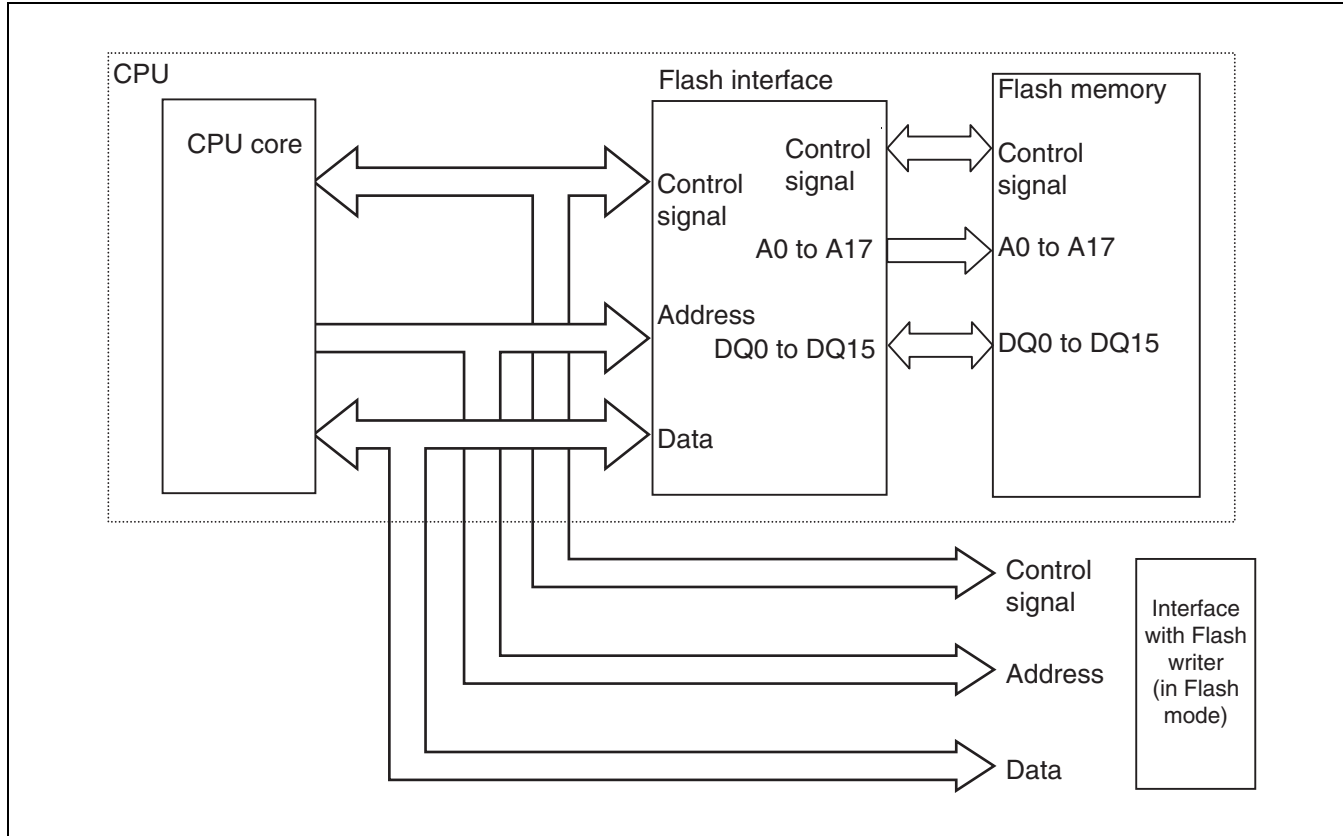
When the RDY bit is set to "0", data is being written or erased with the automatic algorithm, and no write or erase command can be accepted. Moreover, data cannot be read from any address in flash memory. Data read while the RDY bit set to "0" is a hardware sequence flag to indicate flash memory status (see "17.4 Starting the Flash Memory Automatic Algorithm").

■ Writing by a ROM Writer

This flash memory can be written from a device external by using a ROM writer. In this case, a pin feature of the single flash memory product equivalent to MBM29LV400TC is assigned to the device external pin and the FR-CPU stops its operation. The address line connections are changed from those of the CPU mode, and the mapping of the memory area is changed. For details, refer to "Supporting ROM Writer Specifications".

■ Block Diagram of Flash Memory

Figure 17.1-1 Block Diagram of Flash Memory



■ Sector Configuration of Flash Memory

Flash memory employs different address mapping depending on whether it is accessed from the FR-CPU or from the ROM writer. Figure 17.1-2 and Table 17.1-1 show the mapping for access from the FR-CPU. Figure 17.1-3 shows the mapping for access from the ROM writer.

● Mapping for Access from the FR-CPU

Figure 17.1-2 Mapping for Access from the FR-CPU

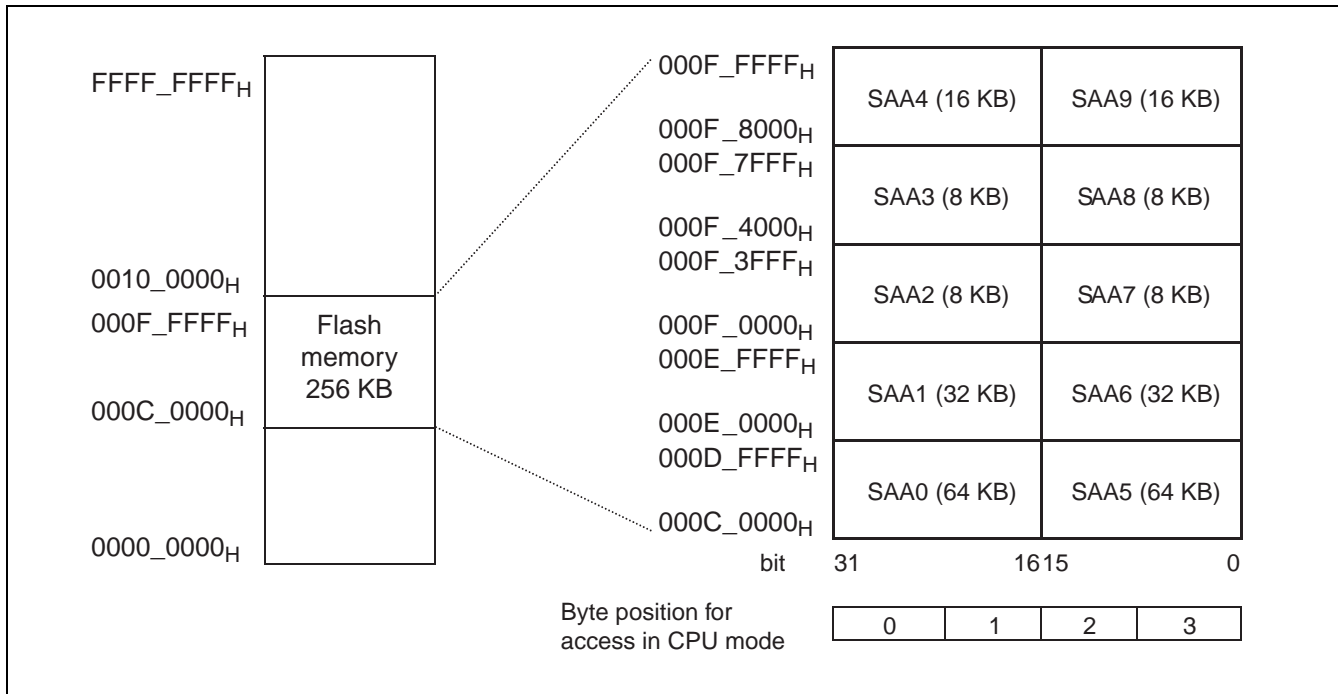


Table 17.1-1 Sector Address List (FR-CPU Access)

Sector Address	Address Range	Bit Position	Sector Capacity
SAA9	F_8002, 3 _H to F_FFFE, F _H	bit15 to bit0	16 KB
SAA8	F_4002, 3 _H to F_7FFE, F _H	bit15 to bit0	8 KB
SAA7	F_0002, 3 _H to F_3FFE, F _H	bit15 to bit0	8 KB
SAA6	E_0002, 3 _H to E_FFFE, F _H	bit15 to bit0	32 KB
SAA5	C_0002, 3 _H to D_FFFE, F _H	bit15 to bit0	64 KB
SAA4	F_8000, 1 _H to F_FFFC, D _H	bit31 to bit15	16 KB
SAA3	F_4000, 1 _H to F_7FFC, D _H	bit31 to bit15	8 KB
SAA2	F_0000, 1 _H to F_3FFC, D _H	bit31 to bit15	8 KB
SAA1	E_0000, 1 _H to E_FFFC, D _H	bit31 to bit15	32 KB
SAA0	C_0000, 1 _H to D_FFFC, D _H	bit31 to bit15	64 KB

● Mapping for Access from the ROM writer

Figure 17.1-3 Address Mapping for Access from the ROM Writer

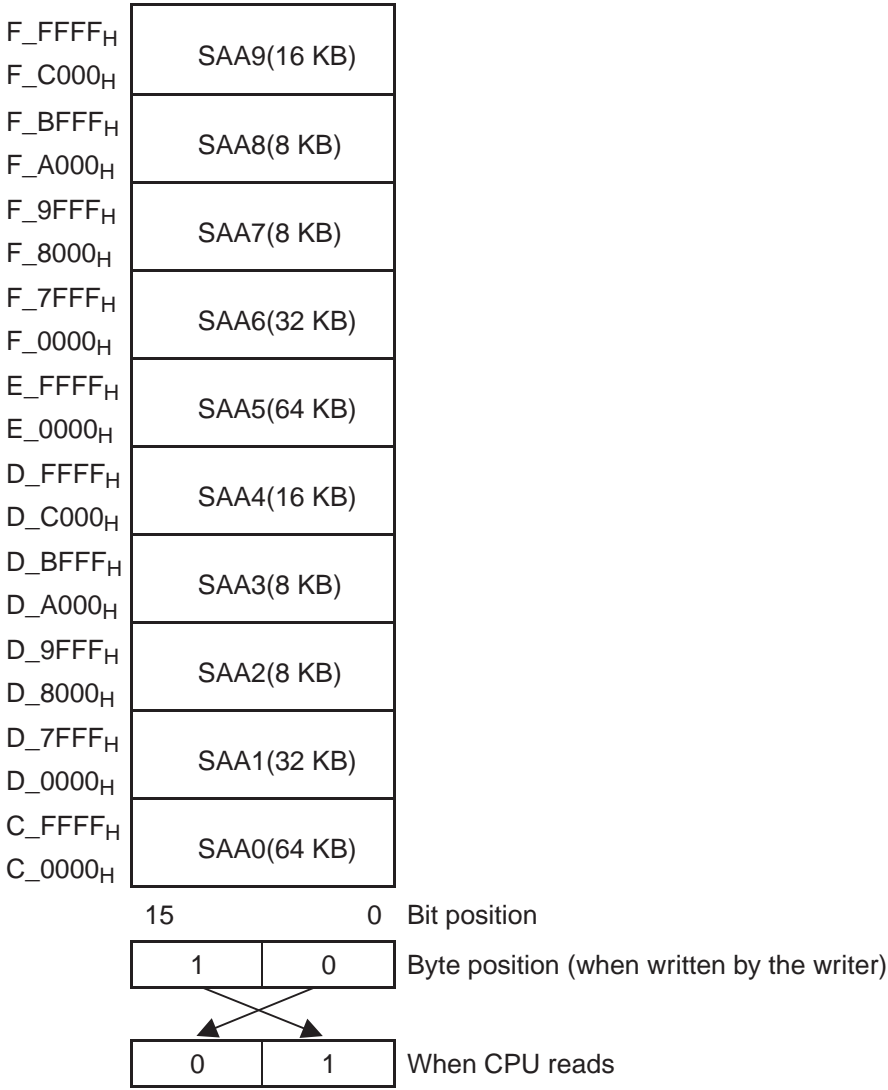


Table 17.1-2 Sector Address List (ROM Writer Access)

Sector Address	Address Range	Bit Position	Sector Capacity
SAA9	F_C000 _H to F_FFFF _H	bit15 to bit0	16 KB
SAA8	F_A000 _H to F_BFFF _H	bit15 to bit0	8 KB
SAA7	F_8000 _H to F_9FFF _H	bit15 to bit0	8 KB
SAA6	F_0000 _H to F_7FFF _H	bit15 to bit0	32 KB
SAA5	E_0000 _H to E_FFFF _H	bit15 to bit0	64 KB
SAA4	D_C000 _H to D_FFFF _H	bit15 to bit0	16 KB
SAA3	D_A000 _H to D_BFFF _H	bit15 to bit0	8 KB
SAA2	D_8000 _H to D_9FFF _H	bit15 to bit0	8 KB
SAA1	D_0000 _H to D_7FFF _H	bit15 to bit0	32 KB
SAA0	C_0000 _H to C_FFFF _H	bit15 to bit0	64 KB

17.2 Flash Memory Registers

The flash memory has two types of registers - Flash Memory Status Register (FLCR) and Flash Memory Wait Register (FLWC).

■ Register List of Flash Memory

Figure 17.2-1 shows a register list of flash memory.

Figure 17.2-1 Register List of Flash Memory

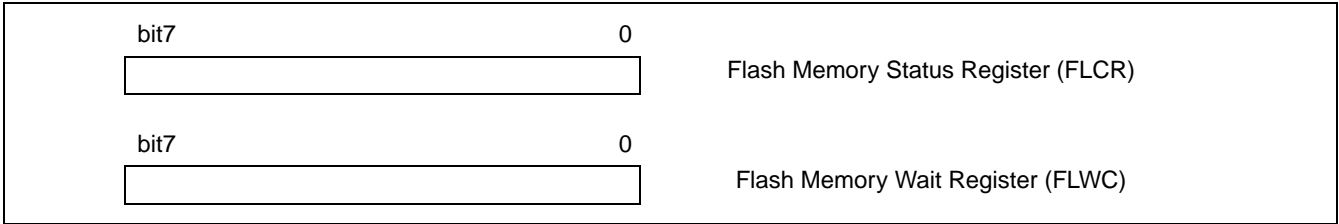


Table 17.2-1 Address Map

Address	Offset				Block
	+0	+1	+2	+3	
007000 _H	FLCR[R/W] B 0110X000	—	—	—	FLASH I/F
007004 _H	FLWC[R/W] B 00000011	—	—	—	

17.2.1 Flash Memory Status Register (FLCR)

The flash memory status register (FLCR) indicates the operation status of the flash memory.

■ Configuration of Flash Memory Status Register (FLCR)

This register controls interrupt to the CUP and writing to flash memory. The FLCR can be accessed only by CPU and cannot be accessed when a writer is installed.

Do not use read-modify-write instructions to access this register.

Figure 17.2-2 shows the bit configuration of FLCR.

Figure 17.2-2 Bit Configuration of Flash Memory Status Register (FLCR)

Address	7	6	5	4	3	2	1	0	← Bit No.
007000 _h	—	—	—	—	RDY	—	WE	—	
	R/W	R/W	R/W	R	R	R/W	R/W	R/W	← Read/Write
	0	1	1	0	X	0	0	0	← Initial value

Functions of each bit in the flash memory status register (FLCR) are described below.

[bit7] (reserved)

Reserved bit. Always set this bit to "0".

[bit6] (reserved)

Reserved bit. Always set this bit to "1".

[bit5] (reserved)

Reserved bit. Always set this bit to "1".

[bit4] (reserved)

Reserved bit. This bit is read only. Write operation does not affect this bit.

[bit3] RDY: This bit indicates the operation status of the automatic algorithm.

RDY	Function
0	Writing or reading is in process, flash memory is not ready to accept a new write or erase command.
1	Flash memory is ready to accept a new data read, write, or erase command.

[bit2] (reserved)

Reserved bit. Always set this bit to "0".

[bit1] WE: Controls the writing of data and commands to flash memory in CPU mode.

When this bit is "0", data and commands cannot be written to flash memory. In addition, data can be read from flash memory at 32-bit access.

When this bit is "1", data and commands can be written to flash memory and the automatic algorithm can be activated. However, data is read from and written to flash memory at 16-bit access. To access to the flash memory, use 16-bit access only. Do not use 32-bit and 8-bit accesses.

If this bit is rewritten, confirm that the automatic algorithm has stopped by checking RDY bit.

When the RDY bit is set to "0", the value of this bit cannot be changed.

WE	Function
0	Writing to flash memory is disabled and data is read from flash memory in 32-bit access mode.
1	Writing to flash memory is enabled and data is read from flash memory in 16-bit access mode.

[bit0] (reserved)

Reserved bit. Always set this bit to "0".

● **Restrictions**

If the WE bit of the FLCR register is rewritten, be sure to execute the instruction sequence below on the FBUS RAM (RAM installed in the CPU). When rewriting the register, do not execute DMA, interrupt, or standby operation.

Instruction Sequence:

```

1:  NOP
2:  NOP
3:  NOP
4:  NOP
5:  NOP
6:  MUL  R2,R3      //32bit dummy multiplication
7:  STB  R11,@R12  //Rewrite register
8:  MUL  R2,R3      //32bit dummy multiplication
9:  NOP

```

17.2.2 Flash Wait Register (FLWC)

The flash wait register (FLWC) controls the wait status of flash memory in CPU mode.

■ Configuration of the Flash Wait Register (FLWC)

Figure 17.2-3 shows the bit configuration of FLWC.

Figure 17.2-3 Bit Configuration of Flash Wait Register (FLWC)

Address	7	6	5	4	3	2	1	0	← Bit No.
007004 _H	–	–	FAC1	FAC0	–	WTC2	WTC1	WTC0	
	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	0	0	0	0	0	0	1	1	← Initial value

Functions of each bit in the flash wait register (FLWC) are described below.

[bit7, bit6] (reserved)

Reserved bits. Always set these bits to "0".

[bit5, bit4] FAC1, FAC0: These bits control the pulse width of the internal write signal.

FAC1	FAC0	ATDIN	EQIN	(Initial value)
0	0	0.5 clock	1 clock	
0	1	1 clock	1.5 clock	
1	0	1.5 clock	2 clock	
1	1	2 clock	2.5 clock	

Note:

ATDIN and EQIN are internal write signals. Use default setting for normal use.
For MASK product, always set "00_B".

[bit3] (reserved)

Reserved bit. Always set this bit to "0".

[bit2 to bit0] WTC2 to WTC0: Wait cycle control bits

These bits control the wait cycle number when accessing flash memory.

WTC2	WTC1	WTC0	Wait Cycle	Reading	Writing	
0	0	0	-	Setting disabled	Setting disabled	
0	0	1	1	Enabled up to 33 MHz	Setting disabled	
0	1	0	2	Enabled up to 33 MHz	Setting disabled	
0	1	1	3	Enabled up to 33 MHz	Enabled up to 33 MHz	(Initial value)
1	0	0	4	Setting disabled	Setting disabled	
1	0	1	5	Setting disabled	Setting disabled	
1	1	0	6	Setting disabled	Setting disabled	
1	1	1	7	Setting disabled	Setting disabled	

Notes:

- Set bits to have cycle number higher than that of set by FAC1 and FAC0.
- Initial value is set for writing. If reading only (when FLCR WE=0), set the wait cycle 1 (WTC2 to WTC0 = 001_B) for reading at the highest speed.
- Initial value for the MASK product, wait cycle 3 (WTC2 to WTC0 = 011_B) is set for reading. Set the wait cycle 1 (WTC2 to WTC0 = 001_B) for reading at the highest speed.

17.3 Access Modes of Flash Memory

FR-The following two types of access modes are available for the FR-CPU access:

- ROM mode word(32-bit) length data can be read at a time, but cannot write data.
 - Programming mode word(32-bit) length access is prohibited but halfword (16-bit) length data can be written.
-

■ FR-CPU ROM Mode (32/16/8-bit, Read Only)

In this mode, the flash memory serves as FR-CPU internal ROM. This mode enables to read 1 word (32 bits) length data at a time but does not enable to write to flash memory or to start the automatic algorithm.

● Mode Specification

- When specifying this mode, set the WE bit of the flash status register to "0".
- This mode is always set after a reset occurs at CPU run time.
- This mode can be set only when the CPU is running.

● Detail of Operation

- In this mode, 1 word (32 bits) length data can be read from the flash memory area at one time. 2 cycles/ 1 word (1 wait) is needed for reading. This will enable instructions to be supplied to FR-CPU without wait.

● Restrictions

- Address assignment and endians in this mode differ from those for writing with the ROM writer. In this mode, both commands and data cannot be written to flash memory.

■ FR-CPU Programming Mode (16-bit, Read/write)

This mode enables data to be written and erased. As 1 word (32 bits) length data cannot be accessed at one time, program execution in flash memory is disabled in this mode.

● Mode Specification

- When specifying this mode, set the WE bit of the flash status register (FLCR) to "1".
- After a reset occurs at CPU run time, the WE bit indicates "0". To set into this mode, set the WE bit to "1". If the WE bit is set again to "0" through a writing operation or because of a reset, the device returns into ROM mode.
- When the RDY bit of the flash memory status register (FLCR) is "0", the WE bit cannot be rewritten. When rewriting the WE bit, ensure that the RDY bit is set to "1".

● Detail of Operation

- One half-word (16 bits) length data can be read from the flash memory area at one time.
- 4 cycles/halfword (3 waits) is needed for reading.
- The automatic algorithm can be started by writing a command to flash memory. When the automatic algorithm starts, data can be written to or erased from flash memory. For details on the automatic algorithm, see "17.4 Starting the Flash Memory Automatic Algorithm" and "17.5 Automatic Algorithm Execution Status".

● Restrictions

- Address assignment and endians in this mode differ from those for writing with the ROM writer. This mode inhibits reading data in words (32 bits).
- This mode inhibits reading data in words (32 bits).
- When switching in the programming access mode, rewrite WE bit by following the restrictions of "17.2.1 Flash Memory Status Register (FLCR)".

17.4 Starting the Flash Memory Automatic Algorithm

Flash memory has an automatic algorithm. Writing to and erasing from flash memory are executed by starting the automatic algorithm.

● Command Operation

To start the automatic algorithm, write halfword (16-bit) data into the flash memory once to six times consecutively. It will be referred to as command. The flash memory is reset to read mode when illegal address are data are written to it, or address and data are written in the wrong order.

Table 17.4-1 shows the command list for flash memory write/erase.

Table 17.4-1 Command List

Command sequence	Number of accesses	First write cycle		Second write cycle		Third write cycle		Fourth write/read cycle		Fifth write cycle		Sixth write cycle	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Read/Reset	1	XXXX _H	F0 _H	-	-	-	-	-	-	-	-	-	-
Read/Reset	4	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	F0 _H	RA	RD	-	-	-	-
Program	4	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	A0 _H	PA	PD	-	-	-	-
Chip erase	6	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	80 _H	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	10 _H
Sector erase	6	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	80 _H	D555 _H	AA _H	CAAAB _H	55 _H	SA	30 _H
Temporary sector erase stop		When Address=XXXX _H and Data=B0 _H are inputted, erasing of the sector currently being erased is temporarily stopped.											
Sector erase restart		When Address=XXXX _H and Data=30 _H are inputted, erasing is restarted after erasing of the sector has been temporarily stopped.											
Auto Select	3	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	90 _H	-	-	-	-	-	-
Continuous mode	3	D555 _H	AA _H	CAAAB _H	55 _H	D555 _H	20 _H	-	-	-	-	-	-
Continuous write	2	XXXX _H	A0 _H	PA	PD	-	-	-	-	-	-	-	-
Continuous mode reset	2	XXXX _H	90 _H	XXXX _H	F0 _H or 00 _H	-	-	-	-	-	-	-	-

The commands are the same for word mode and byte mode. The data of bits that are not listed in the table is arbitrary.

RA: Read address

PA: Write address

SA: Sector address (Specify any address in a sector.)

RD: Read data

PD: Write data

● Read/Reset Command

To return to read mode after the time limit is exceeded, a read/reset command sequence will be issued. Data is read from flash memory in the read cycle. The flash memory remains in reading state until another command is entered.

When the power is turned on, flash memory is automatically set to the read/reset state. In this case, data can be read without a command of the automatic algorithm.

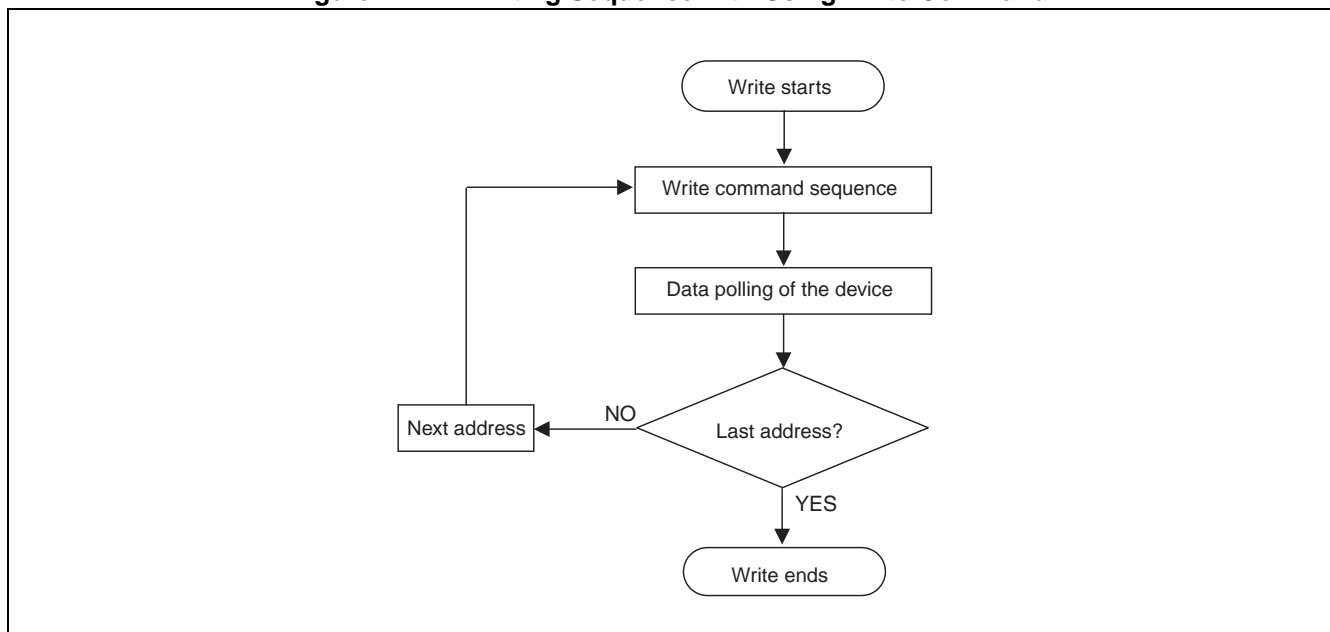
● Program (Write)

In CPU programming mode, data is basically written in halfword units. The write operation is performed in four cycles of bus operation. The command sequence has two "unlock" cycles, which are followed by a write setup command and a write data cycle. Writing to the memory starts in the last write cycle.

After an automatic write algorithm command sequence was executed, it becomes unnecessary to control the flash memory externally. The flash memory itself internally generates write pulses to check the margin of the cells to which data is written. The data polling function compares bit7 of the original data with bit7 of the written data, and if these bits are the same, the automatic write operation ends (see "■Hardware Sequence Flag" in section "17.5 Automatic Algorithm Execution Status"). The automatic write operation then returns to the read mode and accepts no more write addresses. After that, the flash memory requests the next valid address. In this manner, the data polling function indicates the memory is in a write operation.

During a write operation, all commands written to the flash memory are ignored. If a hardware reset starts during write operation, the data at the address for writing may become invalid. Writing operations can be performed in any address sequence and outside of sector boundaries. However, write operations cannot change a data item "0" to "1". If a "0" is overwritten with a "1", either the data polling algorithm determines that the elements are defective, or it looks as if "1" has been written. In the latter case, however, the respective data item is read as "0" in reset/read mode. A data item "0" can be changed to "1" only by an erase operation. Figure 17.4-1 shows a writing sequence with using write command.

Figure 17.4-1 Writing Sequence with Using Write Command



● Chip Erase

The chip erase ("erase all sectors simultaneously") is executed in six access cycles. First, two "unlock" cycles are executed, and then a "setup" command is written. Two more "unlock" cycles are executed to enter the chip erase command.

During the chip erase, the user does not have to write to flash memory before the erase operation. When the automatic erase algorithm is executed, flash memory checks cell states by writing a pattern of zeros before automatically erasing the contents of all cells (preprogram). In this operation, flash memory does not have to be controlled externally.

The automatic erase operation starts with the write operation of the command sequence and ends when bit7 is set to "1", where flash memory returns to the read mode. The chip erase time can be expressed as follows: time for sector erase \times number of all sectors + time for writing to the chip (preprogram).

Figure 17.4-2 shows chip erasing sequence with using chip erase command.

● Sector Erase

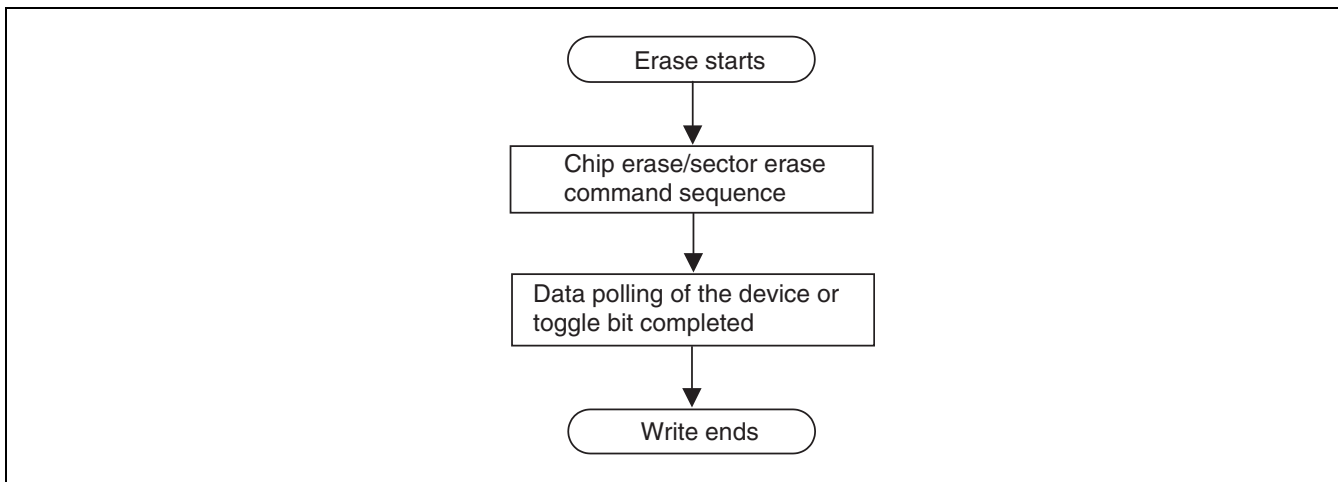
The sector erase is executed in six access cycles. First, two "unlock" cycles are executed, and then a "setup" command is written. After two more "unlock" cycles, the sector erase command is entered in the sixth cycle for starting the sector erase operation. The next sector erase command can be accepted within a time-out period of 50 μ s after the last sector erase command was written.

Multiple sector erase commands can be accepted concurrently during the six bus cycles of the writing operation as mentioned above. This command sequence is executed by writing sector erase commands (30_H) consecutively into the addresses for sectors whose contents are to be erased simultaneously. The sector erase operation itself starts from the end of the time-out period of 50 μ s after the last sector erase command is written. Therefore, when the contents of multiple sectors are erased simultaneously, the subsequent sector erase commands must be inputted within the 50 μ s time-out period to ensure that they are accepted. If the commands are inputted after the 50 μ s time-out period, the commands may not be accepted. For checking whether the succeeding sector erase command is valid, read bit3 (see "■Hardware Sequence Flag" in section "17.5 Automatic Algorithm Execution Status").

During the time-out period, any commands other than sector erase and temporary stop erase are reset to read, and the preceding command sequence is ignored. In the case, the erase operation is completed by erasing the contents of the sector again. Any combination and number (from 0 to 6) of sector addresses can be entered in the sector erase buffers. The user does not have to write to flash memory before the sector erase operation. Flash memory automatically writes to all cells in a sector whose data is automatically erased (preprogram). When the contents of a sector are erased, the other sectors remain intact. In these operations, flash memory does not have to be controlled externally.

The automatic sector erase operation starts from the end of the 50 μ s time-out period after the last sector erase command is written. When bit7 is set to "1", the automatic sector erase operation ends and flash memory returns to the read mode (see "■Hardware Sequence Flag" in section "17.5 Automatic Algorithm Execution Status"). At this time, other commands are ignored. The data polling function is enabled for any sector address in which data has been erased. The time required for erasing the data of multiple sectors can be expressed as follows: time for sector erase + time for sector write (preprogram) \times number of erased sectors.

Figure 17.4-2 shows chip erasing sequence with using chip erase command.

Figure 17.4-2 Chip Erasing Sequence with Using Chip Erase Command

● Temporary Stop Erase

The temporary stop erase command temporarily stops the automatic algorithm in flash memory when the user is erasing the data of a sector, thereby making it possible to write data to and read data from the other sectors. This command is valid only during the sector erase operation, and ignored during chip erase and write operations. The temporary stop erase command ($B0_H$) is effective only during sector erase operation that includes the sector erase time-out period after a sector erase command (30_H) is issued. When this command is entered within the time-out period, time-out ends immediately and the erase operation is suspended. The erase operation is restarted when a restart erase command was written. Temporary stop erase and restart erase commands can be entered with any addresses.

When a temporary stop erase command is entered during sector erase operation, the flash memory needs a maximum of 20 μs to stop the erase operation. When flash memory enters temporary erase stop mode, a ready/busy signal and bit7 output "1", and bit6 stops to toggle. For checking whether the erase operation has stopped, enter the address of the sector whose data is being erased and read the values of bit6 and bit7. At this time, another temporary stop erase command entry is ignored. When the erase operation stops, flash memory enters the temporary erase stop and read mode. Data reading is enabled in this mode for sectors that are not subject to temporary erase stop. Other than that, there is no difference from the standard read operation. In this mode, bit2 toggles for consecutive reading operations from sectors subject to temporary erase stop (see "■Hardware Sequence Flag" in section "17.5 Automatic Algorithm Execution Status").

After the temporary erase stop and read mode is entered, the user can write to flash memory by writing a write command sequence. The write mode in this case is the temporary erase stop and write mode. In this mode, data write operations become valid for sectors that are not subject to temporary erase stop. Other than that, there is no difference from the standard byte writing operation. In this mode, bit2 toggles for consecutive reading operations from sectors that are subject to temporary erase stop. The temporary erase stop bit (bit6) can be used to detect this operation.

Note that bit6 can be read from any addresses, but bit7 must be read from write addresses. To restart the sector erase operation, a restart erase command (30_H) must be entered. Another restart erase command entry is ignored in this point. On the other hand, a temporary stop erase command can be entered after flash memory restarts the erase operation.

17.5 Automatic Algorithm Execution Status

Flash memory is provided with hardware to notify the internal operation status of flash memory and the completion of the operation to the outside of the flash memory for executing write/erase operations in the automatic algorithm. One is a ready/busy signal and the other is a hardware sequence flag.

■ Ready/Busy Signal (RDY/ $\overline{\text{BUSY}}$)

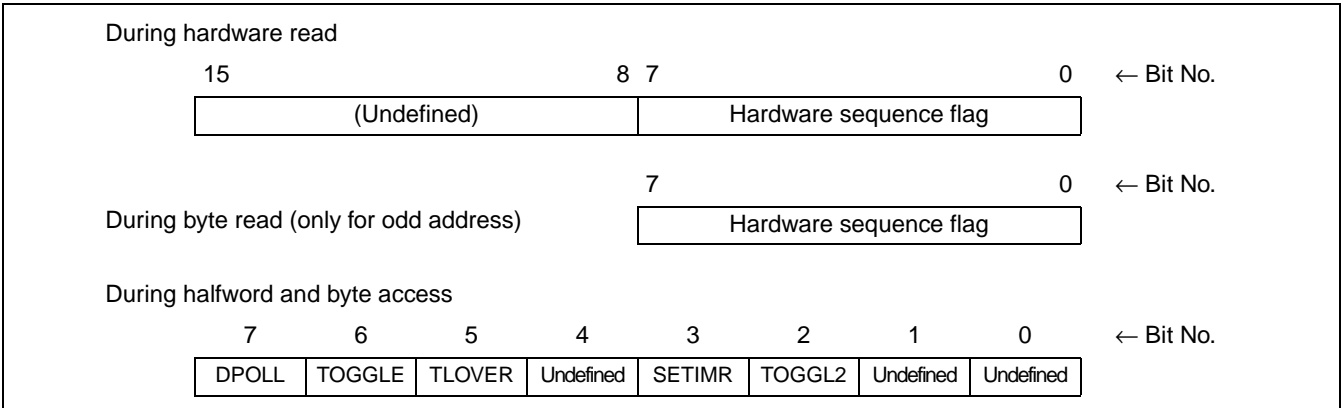
The flash memory has the ready/busy signal in addition to the hardware sequence flag for indicating whether the internal automatic algorithm is running or completed. This ready/busy signal is connected to the flash memory interface circuit, and it can be read as the RDY bit of the flash memory status register. Also, by starting this ready/busy signal, it can generate an interrupt request to the CPU (see "17.1 Overview of Flash Memory").

When the read value of the RDY bit is "0", the flash memory is executing a write or erase operation, where new write and erase commands are not accepted. When the value of the RDY bit is "1", the flash memory is in read/write state or in erase operation wait state.

■ Hardware Sequence Flag

Hardware sequence flag can be obtained as data by reading an arbitrary address (or odd number address in byte access) of the flash memory during automatic algorithm operation. In the data, valid bits are five bits and each of them indicates the status of the automatic algorithm. Figure 17.5-1 shows the structure of the hardware sequence flag.

Figure 17.5-1 Structure of the Hardware Sequence Flag



The hardware sequence flag becomes invalid in FR-CPU ROM mode. Always use FR-CPU programming mode and read only in halfwords or bytes.

Table 17.5-1 shows a hardware sequence flag status list.

Table 17.5-1 Hardware Sequence Flag Status List

State		DPOLL (bit7)	TOGLLE (bit6)	TLOVER (bit5)	SETIMR (bit3)	TOGGL2 (bit2)
Executing	Automatic write operation	Inverted data	Toggle	0	0	1
	Write/erase operation in automatic erase	0	Toggle	0	1	Toggle
	Temporary erase stop	Read (from the sector being erased)	1	1	0	0
		Read (from the sector not being erased)	Data	Data	Data	Data
		Write (to the sector not being erased)	Inverted data	Toggle	0	0
Time limit exceeded	Automatic write operation	Inverted data	Toggle	1	0	1
	Write/erase operation in automatic erase	0	Toggle	1	1	*2

*1: During write operation in the temporary erase stop status, bit2 outputs logic "1" to the read of the address that is being written to.

However, bit2 toggles to the continuous read operations from sectors in temporary erase stop status.

*2: While bit5 is "1" (the time limit is exceeded), bit2 toggles to the continuous read operations for sectors under write/erase operation but does not toggle to the read operations for other sectors.

Each bit is explained below.

[bit7] DPOLL: Data polling

- Automatic write operation status

When read access is performed while the automatic write algorithm is being executed, flash memory outputs the inverted data of the last data written into bit7. When read access is performed at the end of the automatic write algorithm, flash memory outputs bit7 of the read data of the address indicated by the address signal.

- Automatic erase operation status

When read access is performed while the automatic erase algorithm is being executed, flash memory outputs "0" regardless of the address indicated by the address signal. In the same way, "1" is output when it ends.

- Temporary sector erase stop status

When read access is performed during temporary sector erase stop status, flash memory outputs "1" if the address indicated by the address signal belongs to the sector in erase status. If not, flash memory outputs bit7 of the read value of that address. To check whether a sector is currently in temporary sector erase stop status or not, and which sector is in erase status, read this bit together with the toggle bit or bit6 explained later.

Note:

When automatic algorithm operation is close to its end, bit7 (data polling) is changed asynchronously during read operation. This means that the flash memory sends the operation status information to bit7 and the bit7 sends the defined data to the next. When the flash memory ends the automatic algorithm and even if the bit7 outputs the defined data, other bits are undefined. Defined data of other bits is read when executing consecutive read operation.

[bit6] TOGGLE: Toggle bit

- Automatic write/erase operation status
When continuous read operations are performed while the automatic write or erase algorithm is being executed, flash memory outputs "1" and "0" toggling results to bit6. When automatic write or erase algorithm ends, flash memory stops bit6 from toggling to continuous read operations and outputs valid data.
Toggle bit becomes valid after the last write cycle of each command sequence.
If sectors to be written are rewrite assured sectors, the toggle operation ends after about 2 μ s toggling period without rewriting data. When erasing, if the all selected sectors are write assured sectors, the operation returns into the read mode after about 100 μ s toggling period without rewriting data.
- Temporary sector erase stop status
When a read operation is performed during a temporary sector erase stop operation, flash memory outputs "1" if the address indicated by the address signal belongs to the sector in erase state. If not, flash memory outputs bit6 of the read value at the address indicated by the address signal.

[bit5] TLOVER: Time limit over

- Automatic write/erase operation status
bit5 indicates that automatic algorithm is executed beyond the time (internal pulse count) specified inside the flash memory. In this case, bit5 outputs "1". In other words, if this flag outputs "1" during automatic algorithm execution, it indicates that write or erase operation fails.
In addition, bit5 fails when writing into the non-blank without erasing. In this case, defined data cannot be read from bit7 (data polling), and bit6 keeps toggling. If the operation exceeds the time limit in this condition, "1" is output to bit5. This status indicates that flash memory was not used correctly, not that it was defective. Be sure to execute the reset command if this state appears.

[bit3] SETIMR: Sector erase timer

Sector erase operation status

After executing the first sector erase command sequence, the operation enters sector erase wait period. bit3 outputs "0" during this period and outputs "1" if the operation exceeds the sector erase wait period. Data polling and toggle bits become valid after the first sector erase command sequence is executed.

When this flag is "1" while the data polling or toggle bit function is indicating that the erase algorithm is being executed, an internally controlled erase operation has started. Any subsequent written commands are ignored (except for the temporary erase stop code input) until the data polling or toggle bit indicates that the erasing is completed. When this flag is "0", flash memory accepts additional sector erase code entry. In this case, it is recommended to check the status of this flag by software before writing the succeeding sector erase code. If this flag is "1" at the second time of status check, the additional sector erase code may have not been accepted. When a read operation is performed during a temporary sector erase stop operation, flash memory outputs "1" if the address indicated by the address signal belongs to the sector that is subject to the erase operation. If not, flash memory outputs the bit3 of the read value at the address indicated by the address signal.

[bit2] TOGGL2: Toggle bit 2

Sector erase operation status

Together with toggle bit6, this toggle bit is used to report whether flash memory is under automatic erase operation or in temporary erase stop status. bit2 toggles when reading continuously from a sector in erase status during automatic erase operation. When reading continuously from a sector in temporary erase stop read mode while flash memory is in temporary erase stop status mode, bit2 toggles.

When reading addresses continuously from a sector not subject to a temporary erase stop operation while flash memory is in temporary erase stop write mode, bit2 becomes "1". Unlike bit2, bit6 only toggles in normal write and erase operations, or in temporary erase stop status write operation.

For example, bit2 and bit6 are used together to detect a temporary erase stop read mode (bit2 toggles but bit6 does not). bit2 is also used to detect sectors that are subject to erase operations. When the flash memory is in the erase operation and if data is read from a sector that is subject to an erase operation, bit2 toggles.

■ Example of Hardware Sequence Flag Usage

By using hardware sequence flags described above, automatic algorithm status inside the flash memory can be confirmed. Figure 17.5-2 shows a write/erase confirm flow chart for using data polling function and Figure 17.5-3 shows a write/erase confirm flow chart for using toggle bit function as examples.

Figure 17.5-2 Write/Erase Confirmation Flow Chart Using Data Polling Function

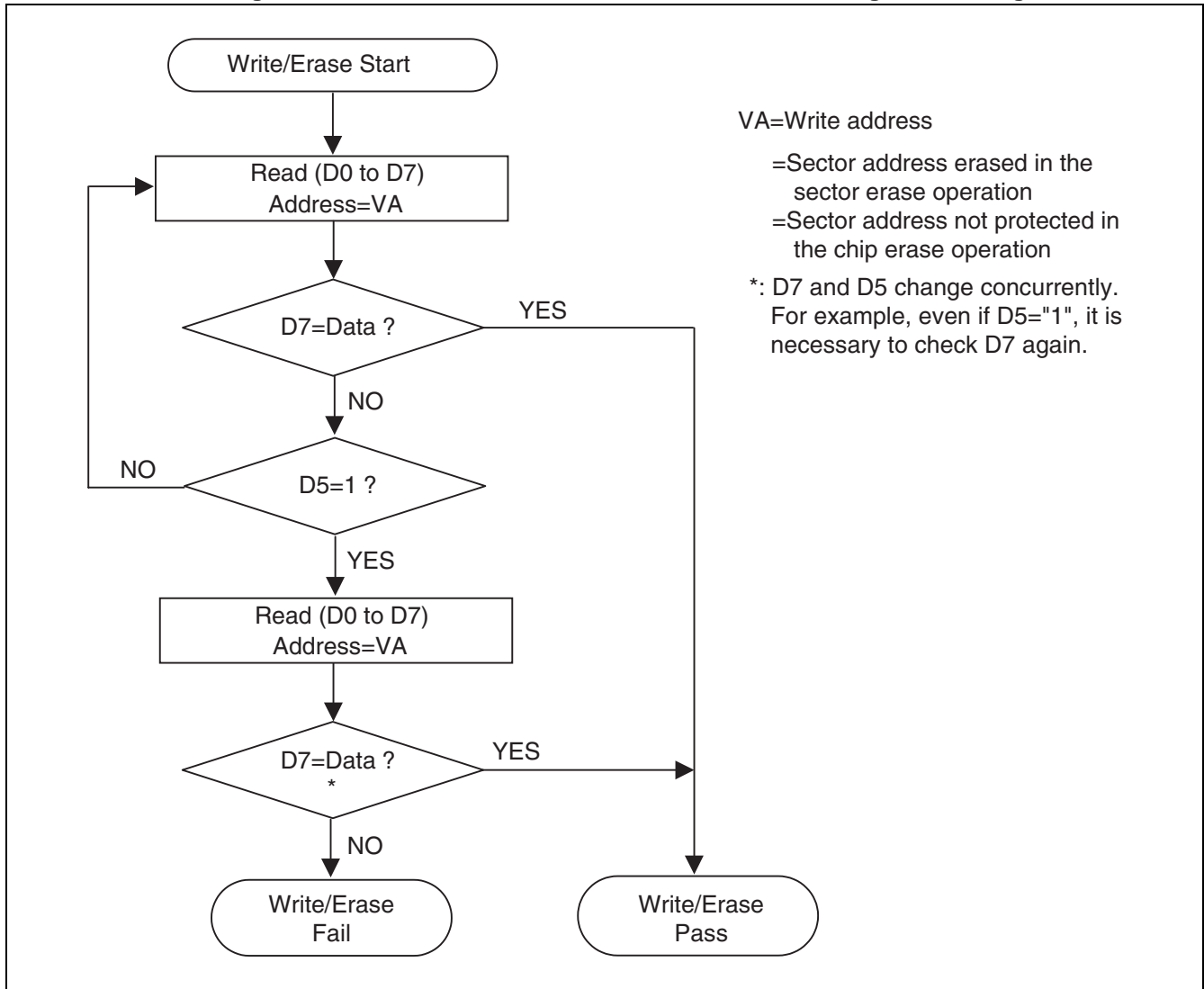
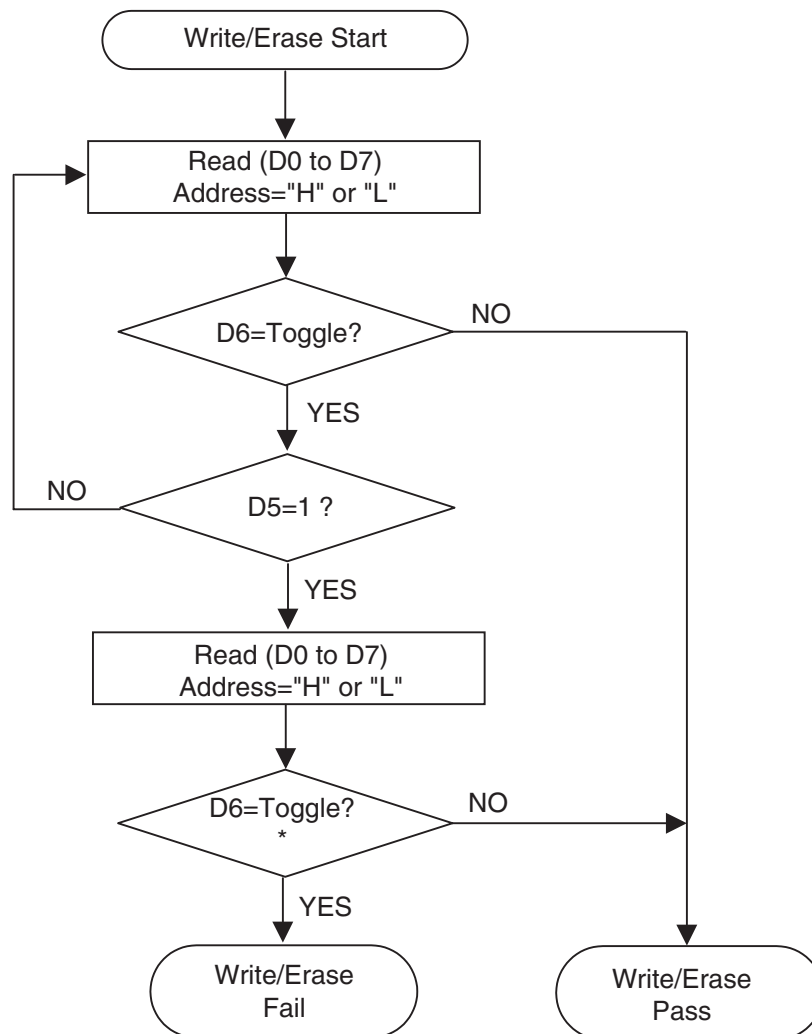


Figure 17.5-3 Write/Erase Confirmation Flow Chart Using Toggle Bit Function

*: D6 stops toggle operation when D5 changes to "1", so even if D5=1, it is necessary to check D6 again.

17.6 Sector Protect Operation

This flash memory has a sector protection function that disables illegal write/erase operation on a sector basis. Once the sector is protected, it will keep the protection function unless it breaks down. However, write/erase operation can be performed on the protected sector by canceling the protection temporarily. This canceling is performed through the sector protect operation.

The automatic algorithms such as write/erase operations are not available in the sector protect operation. The sector protect operation is executable in flash memory mode, but not supported in the normal mode.

For this reason, this operation must be controlled mainly by an external pin using a flash memory writer.

■ Sector Protect Operation List

The following three types of sector protect operations are available:

- Enable sector protect
- Verify sector protect
- Temporary sector protect cancel

Table 17.6-1 shows pin configurations for each type.

Table 17.6-1 Pin Configuration

Operation	CEX	OEX	WEX	A1	A2	A7	A17 to A133	D0 to D15	RSTX	MD2	MD1	MD0
Enable sector protect	L	H	L	L	H	L	Sector address	–	H	V _{ID}	H	V _{ID}
Verify sector protect	L	L	H	L	H	L	Sector address	Code output *	H	H	H	V _{ID}
Temporary sector protect cancel	–	–	–	–	–	–	–	–	H	H	V _{ID}	H

*: Outputs "01_H" during sector protect and "00_H" during sector unprotect

■ Enable Sector Protect

Enable sector protect operation writes into the protection circuit in the flash memory.

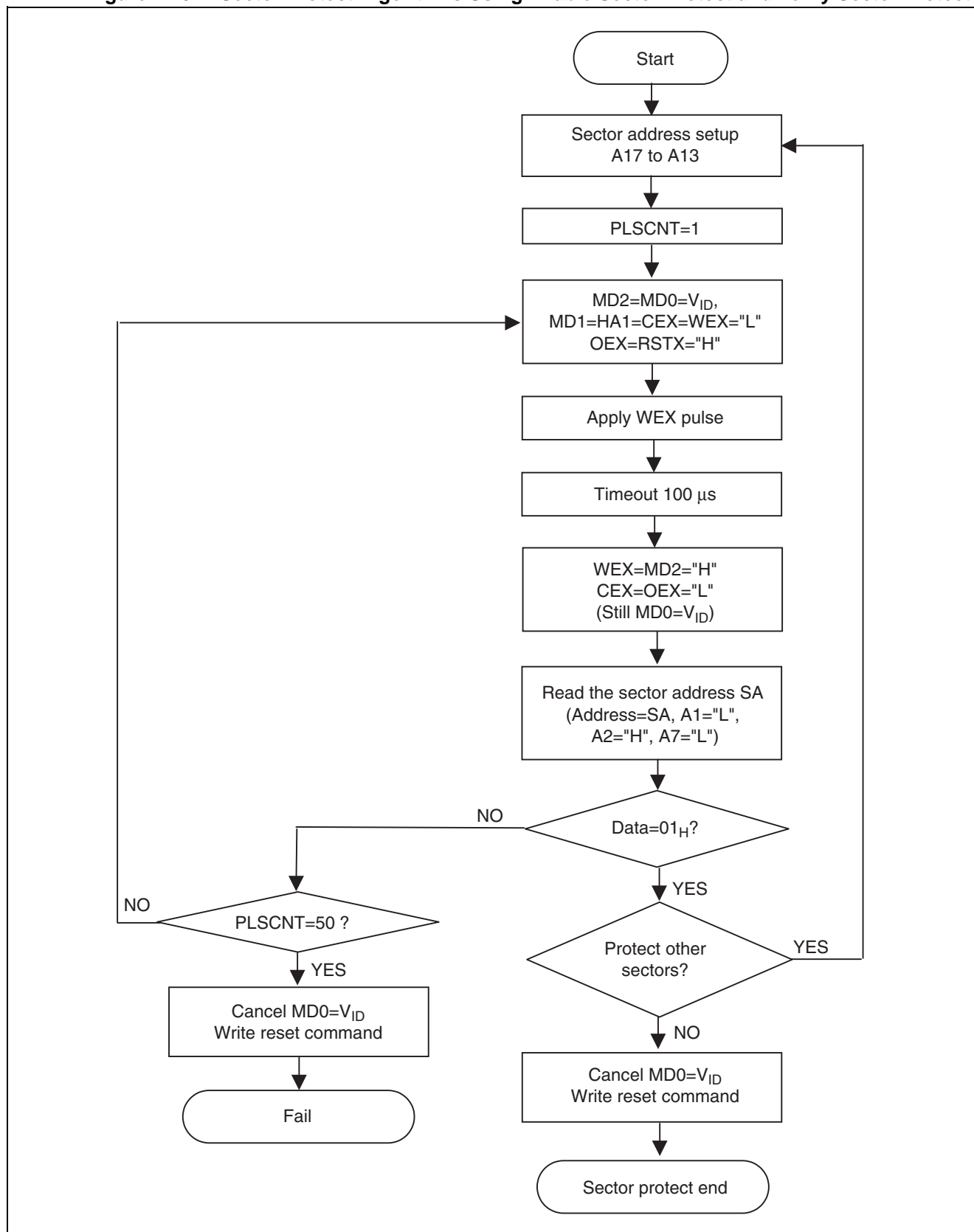
By this operation, write and erase operations are invalidated for any combination of ten sectors. For MB91F264B, the protection is cancelled in all sectors at the factory setting. First in this operation, set sector addresses (A17, A16, A15, A14, A13) of the sector to be protected to the address signal. For the sector and sector address, see Table 17.1-1.

Writing to the protection circuit starts by trailing edge of WEX pulse after applying V_{ID} (=12 V) to MD2 and MD0 to set CEX=0, and ends by leading edge.

Keep sector addresses constant during WEX pulse. Once the sector protection is set, it cannot be canceled. Also, write/erase operations are disabled on the protected sector afterward.

■ Verify Sector Protect

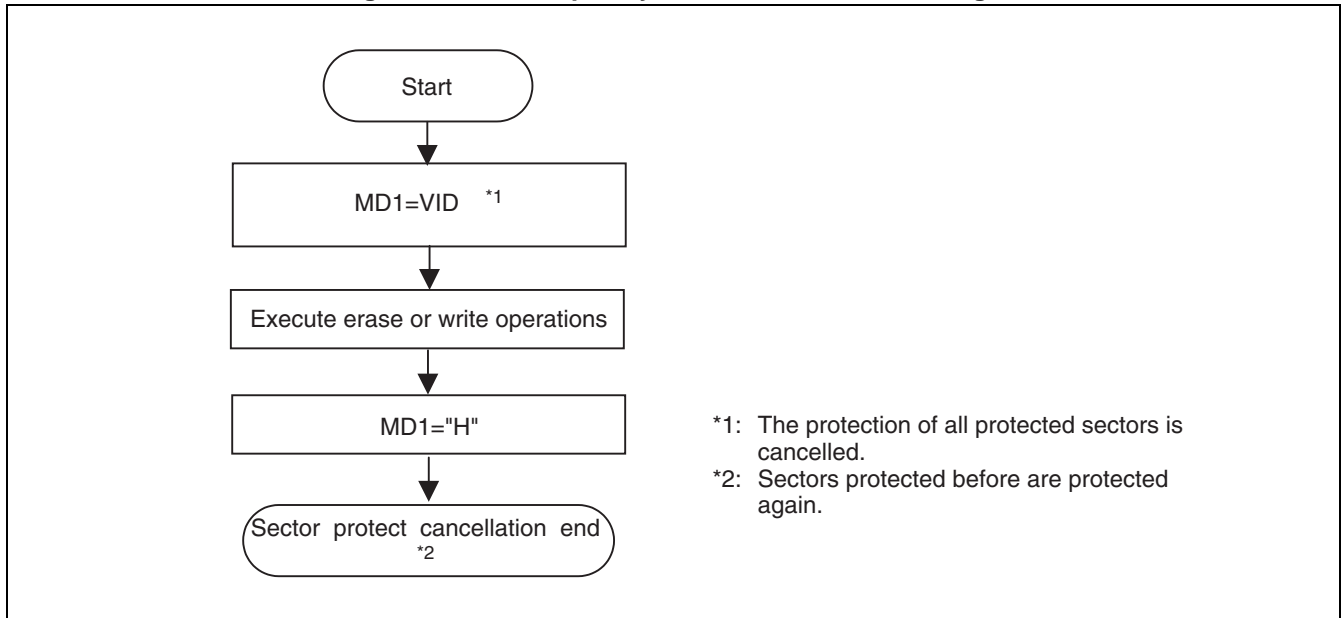
Verify sector protect operation verifies the writing to the protection circuit in the flash memory. First, this operation sets CEX and OEX to "0" and applies V_{ID} (margin mode) to MD0 while WEX is "1". When setting a sector address to the address signal and reading under (A7, A2, A1) = (0, 1, 0) condition, it outputs "1" to output DQ0 in the protected sector. The value "00_H" is read from the unprotected sector. Table 17.6-1 shows the sector protect algorithms using enable sector protect and verify sector protect.

Figure 17.6-1 Sector Protect Algorithms Using Enable Sector Protect and Verify Sector Protect

■ Temporary Sector Protect Cancel

Once a sector is protected by enable sector protect, write/erase operations are disabled unless it breaks down. However, the temporary sector protect cancel operation can cancel the protection information in the protected sector temporarily. This operation is set by keeping an application of V_{ID} to MD1. In this period, the sector protection information that is set before is ignored, and so write/erase operations can be performed in all sectors. When MD1 returns to "1" (=3.3 V), this operation is canceled and all sectors protected before are protected again. Figure 17.6-2 shows the temporary sector protect cancel algorithm.

Figure 17.6-2 Temporary Sector Protect Cancel Algorithm



CHAPTER 18

SERIAL PROGRAMMING CONNECTION

This chapter describes basic configuration of serial programming and examples of the connection.

18.1 Overview

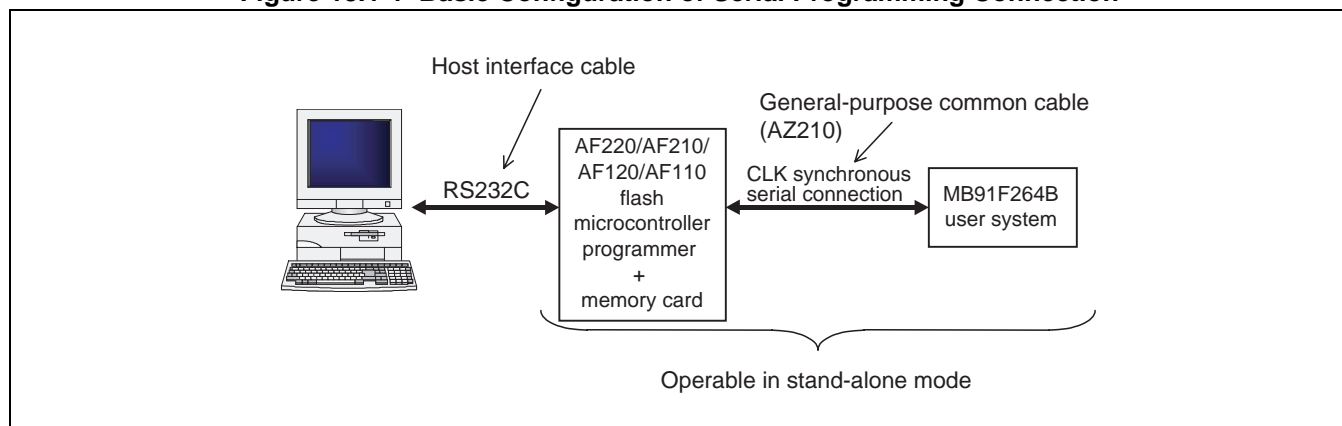
18.1 Overview

The MB91F264B supports the serial onboard writing (Fujitsu standard) of the flash memory. The following explains its specification.

■ Basic Configuration of the Serial Programming Connection

Fujitsu standard serial onboard writing uses the AF220/AF210/AF120/AF110 flash microcontroller programmer by Yokogawa Digital Computer Corporation. The writing can be done by the program of the single-chip mode operation. Figure 18.1-1 shows the basic configuration of MB91F264B serial programming connection.

Figure 18.1-1 Basic Configuration of Serial Programming Connection



Note:

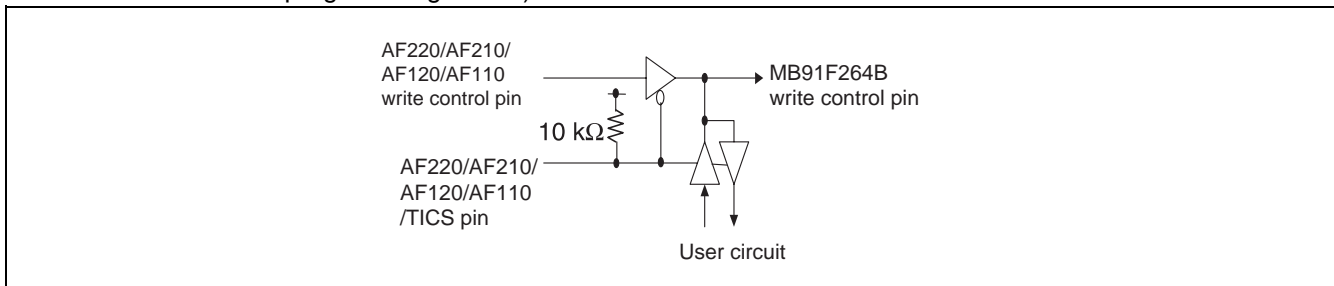
For information about the functions of and operational procedures related to AF210 flash microcontroller programmer, the general-purpose common cable (AZ210) for connection, and the connector, please contact Yokogawa Digital Computer Corporation.

■ Pins Used for Fujitsu Standard Serial Onboard Writing

Pin	Function	Description
MD2, MD1, MD0	Mode pin	Control to set the flash serial programming mode. Flash serial programming mode: MD2, MD1, MD0=1, 0, 0 Reference: Single-chip mode: MD2, MD1, MD0=0, 0, 0
P44, P45	Writing program start pin	Set "L" level to P44, and "H" level to P45.
INIT	Reset pin	—
SIN0	Serial data input pin	Use UART ch.0 resource as clock synchronous mode.
SOT0	Serial data output pin	
SCK0	Serial clock input pin	
V _{CC}	Power voltage supply pin	Supply programming voltage from the user system. When connecting, do not short with power supply of the user side.
V _{SS}	GND pin	Must be shared with GND of the flash microcontroller programmer.

Notes:

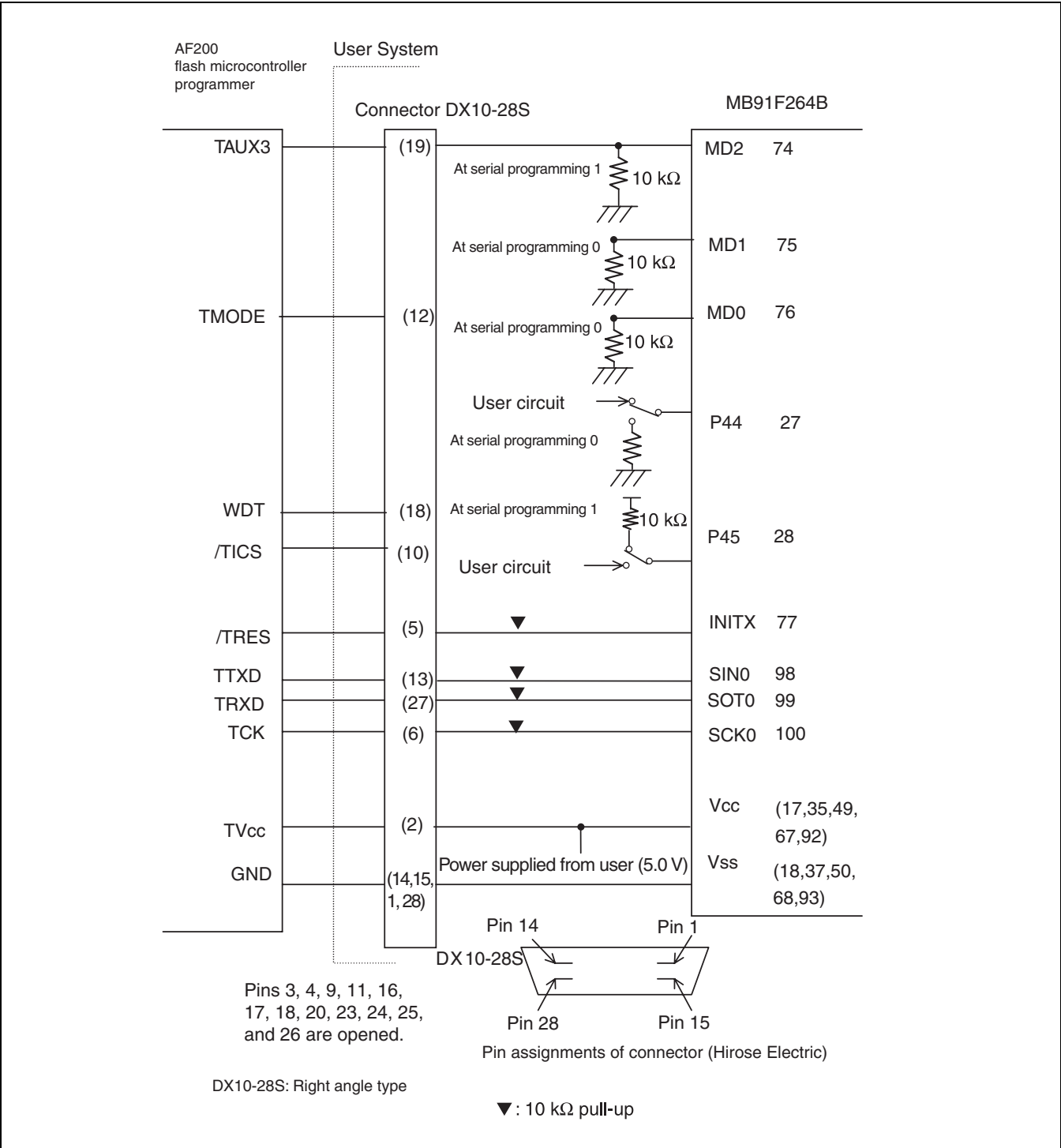
- To use the P44, P45, SIN0, SOT0, and SCK0 pins within the user system as well, the control circuit shown in the following figure is required.
(Using the flash microcontroller programmer's /TICS signal, the user circuit can be disconnected in serial programming mode.)



- Connect with AF200 under the power-off state of the user power supply.

■ Example of Serial Programming Connection

Figure 18.1-2 Example of MB91F264B Serial Programming Connection



■ System Configuration of AF200 Flash Microcontroller Programmer (Yokogawa Digital Computer Corporation)

Type		Function
Main body	AF220 /AC4P	Model with built-in Ethernet interface /100 V to 220 V power adapter
	AF210 /AC4P	Standard model /100 V to 220 V power adapter
	AF120 /AC4P	One-touch key model with built-in Ethernet interface /100 V to 220 V power adapter
	AF110 /AC4P	One-touch key model /100 V to 220 V power adapter
AZ221		Programmer dedicated RS232C cable for PC/AT
AZ210		Standard target probe (a) length: 1 m
FF201		Fujitsu FR flash microcontroller control module
AZ290		Remote controller
/P4		4 MB PC Card (Option) FLASH memory capacity of up to 512 KB supported

For inquiries: Yokogawa Digital Computer Corporation
Tel: +81-42-333-6224

■ Oscillation Clock Frequency

For the write operations on flash memory, the oscillation clock that can be used is 4 MHz.

■ Other Precautionary Information

The port state for flash memory writing via a serial programmer is the same as the reset state except the pin used for writing operation.

APPENDIX

This appendix contains the following items: I/O map, interrupt vector, pin status list, notes when little endian area is used, instruction lists, and the precautions on handling.

APPENDIX A I/O Map

APPENDIX B Vector Table

APPENDIX C Pin Status In Each CPU State

APPENDIX D Notes When Little Endian Area Is Used

APPENDIX E Instruction Lists

APPENDIX F Precautions on Handling

APPENDIX A I/O Map

The correspondence between the memory space area and each register of the peripheral resources is shown below.

I/O Map

[How to read the table]

Address	Register				Block
	+0	+1	+2	+3	
000000 H	PDR0 [R/W] *XXXXXX*	PDR1 [R/W] XXXXXXXX	PDR2 [R/W] XXXXXXXX	PDR3 [R/W] XXXXXXXX	Port Data Register

Read/write attribute

Initial value of register after reset

Register name (Column 1 of the register is at address 4n. Column 2 is at address 4n+2...)

Leftmost register address (For word-length access, column 1 of the register becomes the MSB side of the data.)

Notes:

- The initial value of bits in a register are indicated as follows:
"1": Initial value "1"
"0": Initial value "0"
"X": Initial value "X"
"—": No physical register at the location
- Low-order 16 bits (DTC [15:0]) of the DMACA0 to DMACA4 cannot be accessed by Byte.

Appendix Table A-1 I/O Map (1 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000000 _H	PDR0 [R/W] B XXXXXXXX	PDR1 [R/W] B XXXXXXXX	PDR2 [R/W] B XXXXXXXX	PDR3 [R/W] B XXXXXXXX	Port Data Register
000004 _H	PDR4 [R/W] B XXXXXXXX	PDR5 [R/W] B XXXXXXXX	PDR6 [R/W] B ----XXXX	PDR7 [R/W] B XXXXXXXX	
000008 _H					
00000C _H	PDRC [R/W] B XXXXXXXX	PDRD [R/W] B -----XX	PDRE [R/W] B -----XX	-----	
000010 _H	PDRG [R/W] B --XXXXXX	-----	-----	-----	
000014 _H to 00003C _H					Reserved
000040 _H	EIRR0 [R/W] B,H,W 00000000	ENIR0 [R/W] B,H,W 00000000	ELVR0 [R/W] B,H,W 00000000 00000000		Ext int (INT0 to INT7)
000044 _H	DICR [R/W] B,H,W -----0	HRCL [R/W,R] B,H,W 0--11111	-----	-----	DLYI/Hold Request
000048 _H	TMRLR0 [W] H,W XXXXXXXX XXXXXXXX		TMR0 [R] H,W XXXXXXXX XXXXXXXX		Reload Timer 0
00004C _H	-----		TMCSR0 [R/W,R] B,H,W ---00000 00000000		
000050 _H	TMRLR1 [W] H,W XXXXXXXX XXXXXXXX		TMR1 [R] H,W XXXXXXXX XXXXXXXX		Reload Timer 1
000054 _H	-----		TMCSR1 [R/W,R] B,H,W ---00000 00000000		
000058 _H	TMRLR2 [W] H,W XXXXXXXX XXXXXXXX		TMR2 [R] H,W XXXXXXXX XXXXXXXX		Reload Timer 2
00005C _H	-----		TMCSR2 [R/W,R] B,H,W ---00000 00000000		
000060 _H	SSR0 [R/W,R] B,H,W 00001000	SIDR0[R] / SODR0[W] B,H,W XXXXXXXX	SCR0 [R/W] B,H,W 00000100	SMR0 [R/W,W] B,H,W 00--0-0-	UART0
000064 _H	UTIM0 [R] H / UTIMR0 [W] H 00000000 00000000		DRCL0 -----*3	UTIMC0 [R/W] B 0--00001	U-TIMER0
000068 _H	SSR1 [R/W,R] B,H,W 00001000	SIDR1,SODR1[R/W] B,H,W XXXXXXXX	SCR1 [R/W] B,H,W 00000100	SMR1 [R/W] B,H,W 00--0-0-	UART1
00006C _H	UTIM1 [R] H / UTIMR1 [W] H 00000000 00000000		DRCL1 -----*3	UTIMC1 [R/W] B 0--00001	U-TIMER1
000070 _H	SSR2 [R/W,R] B,H,W 00001000	SIDR2,SODR2 [R/W] B,H,W XXXXXXXX	SCR2 [R/W] B,H,W 00000100	SMR2 [R/W] B,H,W 00--0-0-	UART2
000074 _H	UTIM2 [R] H / UTIMR2 [W] H 00000000 00000000		DRCL2 -----*3	UTIMC2 [R/W] B 0--00001	U-TIMER2
000078 _H	ADCH0 [R/W] B,H,W XX000000	ADMD0 [R/W] B,H,W 00001111	ADCD01 [R] B,H,W XXXXXXXX	ADCD00 [R] B,H,W XXXXXXXX	A/D Converter 0 / Analog Input Control 0
00007C _H	ADCS0[R/W,W] B,H,W 0000X00	-----	AICR0 [R/W] B,H,W 00000000	-----	
000080 _H	ADCH1 [R/W] B,H,W XXXX0XX0	ADMD1 [R/W] B,H,W 00001111	ADCD11 [R] B,H,W XXXXXXXX	ADCD10 [R] B,H,W XXXXXXXX	A/D Converter 1 / Analog Input Control 1
000084 _H	ADCS1[R/W,W] B,H,W 0000X00	-----	AICR1 [R/W] B,H,W -----00	-----	

Appendix Table A-1 I/O Map (2 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000088 _H	ADCH2 [R/W] B,H,W XXXX0XX0	ADMD2 [R/W] B,H,W 00001111	ADCD21 [R] B,H,W XXXXXXXXXX	ADCD20 [R] B,H,W XXXXXXXXXX	A/D Converter 2 / Analog Input Control 2
00008C _H	ADCS2[R/W,W] B,H,W 00000X00	—————	AICR2 [R/W] B,H,W -----00	—————	
000090 _H	OCCPBH0, OCCPBL0[W]/OCCPH0, OCCPL0 [R] H,W 00000000 00000000		OCCPBH1, OCCPBL1[W]/OCCPH1, OCCPL1 [R] H,W 00000000 00000000		16bit OCU
000094 _H	OCCPBH2, OCCPBL2[W]/OCCPH2, OCCPL2 [R] H,W 00000000 00000000		OCCPBH3, OCCPBL3[W]/OCCPH3, OCCPL3 [R] H,W 00000000 00000000		
000098 _H	OCCPBH4, OCCPBL4[W]/OCCPH4, OCCPL4 [R] H,W 00000000 00000000		OCCPBH5, OCCPBL5[W]/OCCPH5, OCCPL5 [R] H,W 00000000 00000000		
00009C _H	OCSH1 [R/W] B,H,W X1100000	OCSL0 [R/W] B,H,W 00001100	OCSH3 [R/W] B,H,W X1100000	OCSL2 [R/W] B,H,W 00001100	
0000A0 _H	OCSH5 [R/W] B,H,W X1100000	OCSL4 [R/W] B,H,W 00001100	OCMOD [R/W] B,H,W XX000000	—————	
0000A4 _H	CPCLRBH, CPCLRBL[W]/CPCLRH, CPCLRL[R] H,W 11111111 11111111		TCDTH, TCDTL [R/W] H,W 00000000 00000000		16bit Free run Timer
0000A8 _H	TCCSH [R/W] B,H,W 00000000	TCCSL [R/W] B,H,W 01000000	—————	ADTRGC [R/W] B,H,W XXXX0000	
0000AC _H	IPC PH0, IPC PL0 [R] H,W XXXXXXXXXX XXXXXXXXXX		IPC PH1, IPC PL1 [R] H,W XXXXXXXXXX XXXXXXXXXX		16bit ICU
0000B0 _H	IPC PH2, IPC PL2 [R] H,W XXXXXXXXXX XXXXXXXXXX		IPC PH3, IPC PL3 [R] H,W XXXXXXXXXX XXXXXXXXXX		
0000B4 _H	PICSH01 [W] B,H,W 000000--	PICSL01 [R/W] B,H,W 00000000	ICSH23 [R] B,H,W XXXXXXXX00	ICSL23 [R/W] B,H,W 00000000	
0000B8 _H	EIRR1 [R/W] B,H,W -----00	ENIR1 [R/W] B,H,W -----00	ELVR1 [R/W] B,H,W -----0000		Ext int (INT8, INT9)
0000BC _H	TMRRH0, TMRRL0 [R/W] H,W XXXXXXXXXX XXXXXXXXXX		TMRRH1, TMRRL1 [R/W] H,W XXXXXXXXXX XXXXXXXXXX		Waveform Generator
0000C0 _H	TMRRH2, TMRRL2 [R/W] H,W XXXXXXXXXX XXXXXXXXXX		—————	—————	
0000C4 _H	DTCR0 [R/W] B,H,W 00000000	DTCR1 [R/W] B,H,W 00000000	DTCR2 [R/W] B,H,W 00000000	—————	
0000C8 _H	—————	SIGCR1 [R/W] B,H,W 00000000	—————	SIGCR2 [R/W] B,H,W XXXXXXXXX1	
0000CC _H	ADCOMP0 [R/W] H,W 00000000 00000000		ADCOMP1 [R/W] H,W 00000000 00000000		A/D COMP
0000D0 _H	ADCOMP2 [R/W] H,W 00000000 00000000		—————	ADCOMPC [R/W] B,H,W XXXXX000	
0000D4 _H to 0000DC _H	—————				Reserved
0000E0 _H	PWCSR0 [R/W,R] B,H,W 00000000 00000000		PWCR0 [R] H,W 00000000 00000000		PWC
0000E4 _H	PWCSR1 [R/W,R] B,H,W 00000000 00000000		PWCR1 [R] H,W 00000000 00000000		
0000E8 _H	—————	PDIVR0 [R/W] B,H,W XXXXXX000	—————	PDIVR1 [R/W] B,H,W XXXXXX000	
0000EC _H to 0000FC _H	—————				Reserved

Appendix Table A-1 I/O Map (3 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000100 _H	PRLH0 [R/W] B,H,W XXXXXXXXXX	PRL0 [R/W] B,H,W XXXXXXXXXX	PRLH1 [R/W] B,H,W XXXXXXXXXX	PRLL1 [R/W] B,H,W XXXXXXXXXX	PPG0 to PPG15
000104 _H	PRLH2 [R/W] B,H,W XXXXXXXXXX	PRLL2 [R/W] B,H,W XXXXXXXXXX	PRLH3 [R/W] B,H,W XXXXXXXXXX	PRLL3 [R/W] B,H,W XXXXXXXXXX	
000108 _H	PPGC0 [R/W] B,H,W 0000000X	PPGC1 [R/W] B,H,W 0000000X	PPGC2 [R/W] B,H,W 0000000X	PPGC3 [R/W] B,H,W 0000000X	
00010C _H	PRLH4 [R/W] B,H,W XXXXXXXXXX	PRLL4 [R/W] B,H,W XXXXXXXXXX	PRLH5 [R/W] B,H,W XXXXXXXXXX	PRLL5 [R/W] B,H,W XXXXXXXXXX	
000110 _H	PRLH6 [R/W] B,H,W XXXXXXXXXX	PRLL6 [R/W] B,H,W XXXXXXXXXX	PRLH7 [R/W] B,H,W XXXXXXXXXX	PRLL7 [R/W] B,H,W XXXXXXXXXX	
000114 _H	PPGC4 [R/W] B,H,W 0000000X	PPGC5 [R/W] B,H,W 0000000X	PPGC6 [R/W] B,H,W 0000000X	PPGC7 [R/W] B,H,W 0000000X	
000118 _H	PRLH8 [R/W] B,H,W XXXXXXXXXX	PRLL8 [R/W] B,H,W XXXXXXXXXX	PRLH9 [R/W] B,H,W XXXXXXXXXX	PRLL9 [R/W] B,H,W XXXXXXXXXX	
00011C _H	PRLH10 [R/W] B,H,W XXXXXXXXXX	PRLL10 [R/W] B,H,W XXXXXXXXXX	PRLH11 [R/W] B,H,W XXXXXXXXXX	PRLL11 [R/W] B,H,W XXXXXXXXXX	
000120 _H	PPGC8 [R/W] B,H,W 0000000X	PPGC9 [R/W] B,H,W 0000000X	PPGC10 [R/W] B,H,W 0000000X	PPGC11 [R/W] B,H,W 0000000X	
000124 _H	PRLH12 [R/W] B,H,W XXXXXXXXXX	PRLL12 [R/W] B,H,W XXXXXXXXXX	PRLH13 [R/W] B,H,W XXXXXXXXXX	PRLL13 [R/W] B,H,W XXXXXXXXXX	
000128 _H	PRLH14 [R/W] B,H,W XXXXXXXXXX	PRLL14 [R/W] B,H,W XXXXXXXXXX	PRLH15 [R/W] B,H,W XXXXXXXXXX	PRLL15 [R/W] B,H,W XXXXXXXXXX	
00012C _H	PPGC12 [R/W] B,H,W 0000000X	PPGC13 [R/W] B,H,W 0000000X	PPGC14 [R/W] B,H,W 0000000X	PPGC15 [R/W] B,H,W 0000000X	
000130 _H	TRG [R/W] B,H,W 00000000 00000000		—————	GATEC [R/W] B,H,W XXXXXX00	
000134 _H	REVC [R/W] B,H,W 00000000 00000000		—————	—————	
000138 _H to 0001FC _H	—————				Reserved
000200 _H	DMACA0 [R/W] B,H,W ^{*1} 00000000 00000000 00000000 00000000				DMAC
000204 _H	DMACB0 [R/W] B,H,W 00000000 00000000 00000000 00000000				
000208 _H	DMACA1 [R/W] B,H,W ^{*1} 00000000 00000000 00000000 00000000				
00020C _H	DMACB1 [R/W] B,H,W 00000000 00000000 00000000 00000000				
000210 _H	DMACA2 [R/W] B,H,W ^{*1} 00000000 00000000 00000000 00000000				
000214 _H	DMACB2 [R/W] B,H,W 00000000 00000000 00000000 00000000				
000218 _H	DMACA3 [R/W] B,H,W ^{*1} 00000000 00000000 00000000 00000000				
00021C _H	DMACB3 [R/W] B,H,W 00000000 00000000 00000000 00000000				
000220 _H	DMACA4 [R/W] B,H,W ^{*1} 00000000 00000000 00000000 00000000				

Appendix Table A-1 I/O Map (4 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000224 _H	DMACB4 [R/W] B,H,W 00000000 00000000 00000000 00000000				DMAC
000228 _H to 00023C _H	_____				Reserved
000240 _H	DMACR [R/W] B 0XX00000 XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX				DMAC
000244 _H to 000398 _H	_____				Reserved
00039C _H	_____	_____	_____	_____	Multiplier Accumulator
0003A0 _H	DSP-PC [R/W] XXXXXXXX	DSP-CSR [R/W,R,W] 00000000	DSP-LY [R/W] XXXXXXXX XXXXXXXX		
0003A4 _H	DSP-OT0 [R] XXXXXXXX XXXXXXXX		DSP-OT1 [R] XXXXXXXX XXXXXXXX		
0003A8 _H	DSP-OT2 [R] XXXXXXXX XXXXXXXX		DSP-OT3 [R] XXXXXXXX XXXXXXXX		
0003AC _H	_____	_____	_____	_____	
0003B0 _H	DSP-OT4 [R] XXXXXXXX XXXXXXXX		DSP-OT5 [R] XXXXXXXX XXXXXXXX		
0003B4 _H	DSP-OT6 [R] XXXXXXXX XXXXXXXX		DSP-OT7 [R] XXXXXXXX XXXXXXXX		
0003B8 _H to 0003EC _H	_____				Reserved
0003F0 _H	BSD0 [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				Bit Search
0003F4 _H	BSD1 [R/W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
0003F8 _H	BSDC [W] W XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
0003FC _H	BSRR [R] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX				
000400 _H	DDR0 [R/W] B 00000000	DDR1 [R/W] B 00000000	DDR2 [R/W] B 00000000	DDR3 [R/W] B 00000000	Data Direction Register
000404 _H	DDR4 [R/W] B 00000000	DDR5 [R/W] B 00000000	DDR6 [R/W] B ----0000	DDR7 [R/W] B 00000000	
000408 _H	_____	_____	_____	_____	
00040C _H	DDRC [R/W] B 00000000	DDRD [R/W] B -----00	DDRE [R/W] B -----00	_____	
000410 _H	DDRG [R/W] B --000000	_____	_____	_____	
000414 _H to 00041C _H	_____				Reserved

Appendix Table A-1 I/O Map (5 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000420 _H	PFR0 [R/W] B 00000000	PFR1 [R/W] B -0000000	PFR2 [R/W] B --00-00-	_____	Port Function Register
000424 _H	_____	_____	_____	PFR7 [R/W] B -----00	
000428 _H	_____	_____	_____	_____	
00042C _H	_____	_____	_____	_____	
000430 _H	PFRG [R/W] B --00--0-	_____	_____	_____	
000434 _H to 00043C _H	_____				Reserved
000440 _H	ICR00 [R/W,R] B,H,W ---1111	ICR01 [R/W,R] B,H,W ---1111	ICR02 [R/W,R] B,H,W ---1111	ICR03 [R/W,R] B,H,W ---1111	Interrupt Control unit
000444 _H	ICR04 [R/W,R] B,H,W ---1111	ICR05 [R/W,R] B,H,W ---1111	ICR06 [R/W,R] B,H,W ---1111	ICR07 [R/W,R] B,H,W ---1111	
000448 _H	ICR08 [R/W,R] B,H,W ---1111	ICR09 [R/W,R] B,H,W ---1111	ICR10 [R/W,R] B,H,W ---1111	ICR11 [R/W,R] B,H,W ---1111	
00044C _H	ICR12 [R/W,R] B,H,W ---1111	ICR13 [R/W,R] B,H,W ---1111	ICR14 [R/W,R] B,H,W ---1111	ICR15 [R/W,R] B,H,W ---1111	
000450 _H	ICR16 [R/W,R] B,H,W ---1111	ICR17 [R/W,R] B,H,W ---1111	ICR18 [R/W,R] B,H,W ---1111	ICR19 [R/W,R] B,H,W ---1111	
000454 _H	ICR20 [R/W,R] B,H,W ---1111	ICR21 [R/W,R] B,H,W ---1111	ICR22 [R/W,R] B,H,W ---1111	ICR23 [R/W,R] B,H,W ---1111	
000458 _H	ICR24 [R/W,R] B,H,W ---1111	ICR25 [R/W,R] B,H,W ---1111	ICR26 [R/W,R] B,H,W ---1111	ICR27 [R/W,R] B,H,W ---1111	
00045C _H	ICR28 [R/W,R] B,H,W ---1111	ICR29 [R/W,R] B,H,W ---1111	ICR30 [R/W,R] B,H,W ---1111	ICR31 [R/W,R] B,H,W ---1111	
000460 _H	ICR32 [R/W,R] B,H,W ---1111	ICR33 [R/W,R] B,H,W ---1111	ICR34 [R/W,R] B,H,W ---1111	ICR35 [R/W,R] B,H,W ---1111	
000464 _H	ICR36 [R/W,R] B,H,W ---1111	ICR37 [R/W,R] B,H,W ---1111	ICR38 [R/W,R] B,H,W ---1111	ICR39 [R/W,R] B,H,W ---1111	
000468 _H	ICR40 [R/W,R] B,H,W ---1111	ICR41 [R/W,R] B,H,W ---1111	ICR42 [R/W,R] B,H,W ---1111	ICR43 [R/W,R] B,H,W ---1111	
00046C _H	ICR44 [R/W,R] B,H,W ---1111	ICR45 [R/W,R] B,H,W ---1111	ICR46 [R/W,R] B,H,W ---1111	ICR47 [R/W,R] B,H,W ---1111	
000470 _H to 00047C _H	_____				Reserved
000480 _H	RSRR [R/W] B,H,W 10000000	STCR [R/W] B,H,W 00110011	TBCR [R/W] B,H,W 00XXXX00	CTBR [W] B,H,W XXXXXXXXXX	Clock Control unit
000484 _H	CLKR [R/W] B,H,W 00000000	WPR -----*3	DIVR0 [R/W] B,H,W 00000011	DIVR1 [R/W] B,H,W 00000000	
000488 _H to 0005FC _H	_____				Reserved

Appendix Table A-1 I/O Map (6 / 7)

Address	Register				Block
	+0	+1	+2	+3	
000600 _H	PCR0 [R/W] B 00000000	PCR1 [R/W] B 00000000	PCR2 [R/W] B 00000000	PCR3 [R/W] B 00-----	Pull Up Control
000604 _H	PCR4 [R/W] B 00000000	PCR5 [R/W] B 00000000	PCR6 [R/W] B ----0000	PCR7 [R/W] B 00000000	
000608 _H	_____	_____	_____	_____	
00060C _H	_____	_____	_____	_____	
000610 _H	PCRG [R/W] B --000000	_____	_____	_____	
000614 _H to 000FFC _H	_____				Reserved
001000 _H	DMASA0 [R/W] W 00000000 00000000 00000000 00000000				DMAC
001004 _H	DMADA0 [R/W] W 00000000 00000000 00000000 00000000				
001008 _H	DMASA1 [R/W] W 00000000 00000000 00000000 00000000				
00100C _H	DMADA1 [R/W] W 00000000 00000000 00000000 00000000				
001010 _H	DMASA2 [R/W] W 00000000 00000000 00000000 00000000				
001014 _H	DMADA2 [R/W] W 00000000 00000000 00000000 00000000				
001018 _H	DMASA3 [R/W] W 00000000 00000000 00000000 00000000				
00101C _H	DMADA3 [R/W] W 00000000 00000000 00000000 00000000				
001020 _H	DMASA4 [R/W] W 00000000 00000000 00000000 00000000				
001024 _H	DMADA4 [R/W] W 00000000 00000000 00000000 00000000				
001028 _H to 006FFC _H	_____				Reserved
007000 _H	FLCR [R,R/W] 0110X000	_____	_____	_____	FLASH
007004 _H	FLWC [R,R/W] ^{*2} 00000011	_____	_____	_____	
007008 _H	_____	_____	_____	_____	
00700C _H	_____	_____	_____	_____	
007010 _H	_____	_____	_____	_____	
007014 _H to 00BFFC _H	_____				Reserved

Appendix Table A-1 I/O Map (7 / 7)

Address	Register				Block
	+0	+1	+2	+3	
00C000 _H to 00C07C _H	X-RAM (Coefficient RAM) [R/W] 64 × 16 bit				Multiplier Accumulator
00C080 _H to 00C0FC _H	Y-RAM (Variable RAM) [R/W] 64 × 16 bit				
00C100 _H to 00C2FC _H	I-RAM (Instruction RAM) [R/W] 256 × 16 bit				
00C300 _H to 00FFFC _H	—————				Reserved

*1: Low-order 16 bits (DTC [15:0]) of the DMACA0 to DMACA4 cannot be accessed by Byte.

*2: The initial value of FLWC (7004_H) is "00010011_B" on the EVA tool. Writing "00000011_B" to the EVA product does not affect the operation.

*3: These are reserved registers. These access are prohibited.

Notes:

- Do not use RMW instructions to the register having a write-only bit.
 - Data indicated as reserved or (–) is undefined.
-

APPENDIX B Vector Table

Appendix Table B-1 shows an interrupt vector table. In this table, interrupt factors and interrupt vector/interrupt control register allocations for MB91260B series are listed.

ICR: ICR is a register installed among the interruption controllers. ICR sets the interruption level to each demand of the interruption. ICR is prepared for corresponding to each interruption demand.

TBR: TBR is a register that shows the first address of the vector table for EIT.
The address where TBR and the defined offset value of each EIT factor were added is a vector address.

The area of 1 KB from an address that TBR shows is the vector area for EIT.

The size of one vector is four bytes. The relationship between the vector number and the vector address is shown as follows.

$$\begin{aligned} \text{vctadr} &= \text{TBR} + \text{vctofs} \\ &= \text{TBR} + (3\text{FC}_{\text{H}} - 4 \times \text{vct}) \end{aligned}$$

vctadr: Vector address

vctofs: Vector offset

vct: Vector number

Appendix Table B-1 Vector Table (1 / 3)

Interrupt factor	Interrupt No.		Interrupt level	Offset	TBR default address	RN
	Decimal number	Hexadecimal number				
Reset	0	00	–	3FC _H	000FFFFC _H	–
Mode vector	1	01	–	3F8 _H	000FFFF8 _H	–
System reserved	2	02	–	3F4 _H	000FFFF4 _H	–
System reserved	3	03	–	3F0 _H	000FFFF0 _H	–
System reserved	4	04	–	3EC _H	000FFFE4 _H	–
System reserved	5	05	–	3E8 _H	000FFFE8 _H	–
System reserved	6	06	–	3E4 _H	000FFFE4 _H	–
Coprocessor absence trap	7	07	–	3E0 _H	000FFFE0 _H	–
Coprocessor error trap	8	08	–	3DC _H	000FFFD4 _H	–
INTE instruction	9	09	–	3D8 _H	000FFFD8 _H	–
System reserved	10	0A	–	3D4 _H	000FFFD4 _H	–
System reserved	11	0B	–	3D0 _H	000FFFD0 _H	–
Step trace trap	12	0C	–	3CC _H	000FFFCC _H	–
NMI request (tool)	13	0D	–	3C8 _H	000FFF8 _H	–
Undefined instruction exception	14	0E	–	3C4 _H	000FFF4 _H	–
NMI request	15	0F	15(F _H) Fixed	3C0 _H	000FFF0 _H	–
External interrupt 0	16	10	ICR00	3BC _H	000FFFBC _H	6
External interrupt 1	17	11	ICR01	3B8 _H	000FFFB8 _H	7
External interrupt 2	18	12	ICR02	3B4 _H	000FFFB4 _H	–
External interrupt 3	19	13	ICR03	3B0 _H	000FFFB0 _H	–
External interrupt 4	20	14	ICR04	3AC _H	000FFFAC _H	–
External interrupt 5	21	15	ICR05	3A8 _H	000FFFA8 _H	–
External interrupt 6	22	16	ICR06	3A4 _H	000FFFA4 _H	–
External interrupt 7	23	17	ICR07	3A0 _H	000FFFA0 _H	–
Reload timer 0	24	18	ICR08	39C _H	000FFF9C _H	8
Reload timer 1	25	19	ICR09	398 _H	000FFF98 _H	9
Reload timer 2	26	1A	ICR10	394 _H	000FFF94 _H	10
UART0 (Reception completed)	27	1B	ICR11	390 _H	000FFF90 _H	0
UART0 (Transmission completed)	28	1C	ICR12	38C _H	000FFF8C _H	3
DTTI	29	1D	ICR13	388 _H	000FFF88 _H	–
DMAC0 (End or error)	30	1E	ICR14	384 _H	000FFF84 _H	–
DMAC1 (End or error)	31	1F	ICR15	380 _H	000FFF80 _H	–
DMAC2/3/4 (End or error)	32	20	ICR16	37C _H	000FFF7C _H	–

Appendix Table B-1 Vector Table (2 / 3)

Interrupt factor	Interrupt No.		Interrupt level	Offset	TBR default address	RN
	Decimal number	Hexadecimal number				
UART1 (Reception completed)	33	21	ICR17	378 _H	000FFF78 _H	1
UART1 (Transmission completed)	34	22	ICR18	374 _H	000FFF74 _H	4
UART2 (Reception completed)	35	23	ICR19	370 _H	000FFF70 _H	2
UART2 (Transmission completed)	36	24	ICR20	36C _H	000FFF6C _H	5
Multiplier Accumulator	37	25	ICR21	368 _H	000FFF68 _H	—
PPG0	38	26	ICR22	364 _H	000FFF64 _H	—
PPG1	39	27	ICR23	360 _H	000FFF60 _H	—
PPG2/3	40	28	ICR24	35C _H	000FFF5C _H	—
PPG4/5/6/7	41	29	ICR25	358 _H	000FFF58 _H	—
PPG8/9/10/11/12/13/14/15	42	2A	ICR26	354 _H	000FFF54 _H	—
External interrupt 8/9	43	2B	ICR27	350 _H	000FFF50 _H	—
Waveform 0(Underflow)	44	2C	ICR28	34C _H	000FFF4C _H	—
Waveform 1(Underflow)	45	2D	ICR29	348 _H	000FFF48 _H	—
Waveform 2(Underflow)	46	2E	ICR30	344 _H	000FFF44 _H	—
Time base timer overflow	47	2F	ICR31	340 _H	000FFF40 _H	—
Free-run timer (Compare clear)	48	30	ICR32	33C _H	000FFF3C _H	—
Free-run timer (Zero detection)	49	31	ICR33	338 _H	000FFF38 _H	—
A/D0	50	32	ICR34	334 _H	000FFF34 _H	—
A/D1	51	33	ICR35	330 _H	000FFF30 _H	—
A/D2	52	34	ICR36	32C _H	000FFF2C _H	—
PWC0 (Measurement completed)	53	35	ICR37	328 _H	000FFF28 _H	—
PWC1 (Measurement completed)	54	36	ICR38	324 _H	000FFF24 _H	—
PWC0 (Overflow)	55	37	ICR39	320 _H	000FFF20 _H	—
PWC1 (Overflow)	56	38	ICR40	31C _H	000FFF1C _H	—
ICU0 (Capture)	57	39	ICR41	318 _H	000FFF18 _H	—
ICU1 (Capture)	58	3A	ICR42	314 _H	000FFF14 _H	—
ICU2/3 (Capture)	59	3B	ICR43	310 _H	000FFF10 _H	—
OCU0/1 (Match)	60	3C	ICR44	30C _H	000FFF0C _H	—
OCU2/3 (Match)	61	3D	ICR45	308 _H	000FFF08 _H	—
OCU4/5 (Match)	62	3E	ICR46	304 _H	000FFF04 _H	—
Delayed interrupt factor bit	63	3F	ICR47	300 _H	000FFF00 _H	—

Appendix Table B-1 Vector Table (3 / 3)

Interrupt factor	Interrupt No.		Interrupt level	Offset	TBR default address	RN
	Decimal number	Hexadecimal number				
System reserved (used by REALOS)	64	40	–	2FC _H	000FFEFC _H	–
System reserved (used by REALOS)	65	41	–	2F8 _H	000FFEF8 _H	–
System reserved	66	42	–	2F4 _H	000FFEF4 _H	–
System reserved	67	43	–	2F0 _H	000FFEF0 _H	–
System reserved	68	44	–	2EC _H	000FFEEC _H	–
System reserved	69	45	–	2E8 _H	000FFEE8 _H	–
System reserved	70	46	–	2E4 _H	000FFEE4 _H	–
System reserved	71	47	–	2E0 _H	000FFEE0 _H	–
System reserved	72	48	–	2DC _H	000FFEDC _H	–
System reserved	73	49	–	2D8 _H	000FFED8 _H	–
System reserved	74	4A	–	2D4 _H	000FFED4 _H	–
System reserved	75	4B	–	2D0 _H	000FFED0 _H	–
System reserved	76	4C	–	2CC _H	000FFECC _H	–
System reserved	77	4D	–	2C8 _H	000FFEC8 _H	–
System reserved	78	4E	–	2C4 _H	000FFEC4 _H	–
System reserved	79	4F	–	2C0 _H	000FFEC0 _H	–
Used in INT instruction	80 to 255	50 to FF	–	2BC _H to 000 _H	000FFEBC _H to 000FFC00 _H	–

APPENDIX C Pin Status In Each CPU State

This appendix describes the pin status in each CPU state.

Words and phrases used for the pin status have the following meanings.

1. Input enabled
It means that the input function is allowed to be used.
2. Input 0 fix
It means that the pin is sending "0" to the internal by blocking the external input at the input gate near the pin.
3. Output Hi-Z
It means that the transistor for pin drive is disabled and the pin is set to high impedance.
4. Output Retention
It means that the output status used immediately before becoming this mode is output as it is.
That is, when internal peripherals are operating, the pin will output by following the peripherals in which the output occurs. When the pin outputs as a port, it will retain the output.
5. Retention of the Status Immediately Before
It means that the output status immediately before becoming this mode is output as it is, or for input, it means that the input is enabled.

[Single-Chip Mode]

Appendix Table C-1 Pin Status In Each CPU State (1 / 2)

Pin No.		Pin name	Function	At initialize		At sleep	At stop	
QFP	LQFP			$\overline{\text{INIT}}=\text{L}^{*1}$	$\overline{\text{INIT}}=\text{H}^{*2}$		HIZ=0	HIZ=1
1	99	P23	SIN1	Output Hi-Z/ Input disabled	Output Hi-Z/ Input disabled	Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
2	100	P24	SOT1					
3	1	P25	SCK1					
4, 5	2, 3	P26, P27	INT6-INT7			Input enabled	Input enabled	Input enabled
6 to 9	4 to 7	P50 to 53	Port			Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
10	8	P54	INT0			Input enabled	Input enabled	Input enabled
11	9	P55	INT1					
12	10	P56	INT2					
13	11	P57	INT3					
14	12	PG0	CKI/INT4					
15	13	PG1	PPG0/INT5			Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
16	14	PG2	Port					
20	18	PG3	SIN2					
21	19	PG4	SOT2					
22	20	PG5	SCK2					
23 to 30	21 to 28	P40 to P47	Port			Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
31, 32	29, 30	PE1, PE0	AN11, AN10					
38, 39	36, 37	PD1, PD0	AN9, AN8					
41 to 48	39 to 46	PC7 to PC0	AN7 to AN0					
51 to 56	49 to 54	P30 to P35	RTO0 to RTO5					
57, 58	55, 56	P36, P37	IC0, IC1					
59, 60	57, 58	P60, P61	IC2, IC3					
61, 62	59, 60	P62, P63	INT8, INT9					
63, 64	61, 62	P70, P71	TOT1, TOT2					
65	63	P72	DTTI					
66	64	P73	PWI0					
69	67	P74	PWI1					
70	68	P75	ADTG0			Input enabled	Input enabled	Input enabled
71	69	P76	ADTG1			Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
72	70	P77	ADTG2					

Appendix Table C-1 Pin Status In Each CPU State (2 / 2)

Pin No.		Pin name	Function	At initialize		At sleep	At stop	
QFP	LQFP			$\overline{\text{INIT}}=\text{L}^{*1}$	$\overline{\text{INIT}}=\text{H}^{*2}$		HIZ=0	HIZ=1
73	71	$\overline{\text{NMI}}$	$\overline{\text{NMI}}$	Input enabled	Input enabled	Input enabled	Input enabled	Input enabled
78	76	P00	PPG1	Output Hi-Z/ Input disabled	Output Hi-Z/ Input disabled	Retention of the status immediately before	Retention of the status immediately before	Output Hi-Z/ Input 0 fix
79	77	P01	PPG2					
80	78	P02	PPG3					
81	79	P03	PPG4					
82	80	P04	PPG5					
83	81	P05	PPG6					
84	82	P06	PPG7					
85	83	P07	PPG8					
86	84	P10	PPG9					
87	85	P11	PPG10					
88	86	P12	PPG11					
89	87	P13	PPG12					
90	88	P14	PPG13					
91	89	P15	PPG14					
96	94	P16	PPG15					
97	95	P17	Port					
98	96	P20	SIN0					
99	97	P21	SOT0					
100	98	P22	SCK0					

*1: $\overline{\text{INIT}}=\text{L}$: Pin status while $\overline{\text{INIT}}$ is "L".

*2: $\overline{\text{INIT}}=\text{H}$: Pin status immediately after $\overline{\text{INIT}}$ changes from "L" to "H".

APPENDIX D Notes When Little Endian Area Is Used

This appendix explains operating suggestions for the following items when using the little endian area.

- C compiler
 - Assembler
 - Linker
 - Debugger
-

■ C Compiler (fcc911)

Please note that operations cannot be guaranteed, if the following operations are performed to the little endian area when programming by C language.

- Arrangement of variable with initial value
- Assignment of structure
- Operations other than character type array that uses character string operation function
- Specification of -K lib option when using character string operation function
- Use of double type and long double type
- Arrangement of stack in little endian area

● Arrangement of variable with initial value

The variable with an initial value cannot be arranged in the little endian area.

The compiler does not have a function to generate an initial value of the little endian. The variable can be arranged in the little endian area but an initial value cannot be set to it.

Please run a process to set an initial value at the head of the program.

[Example] When setting an initial value to the variable `little_data` of the little endian area.

```
extern int little_data;

void little_init(void) {
    little_data = initial value;
}

void main(void) {
    little_init();
    ...
}
```

● Assignment of structure

When assigning between structures, the compiler selects the best method to transfer and transfers on a byte, halfword, and word basis. Therefore, when the structure assignment is done between a structure variable allocated in the normal area and a structure variable allocated in the little endian area, a correct result cannot be obtained.

Please assign the structure member respectively.

[Example] When assigning the structure to the structure variable `little_st` of the little endian area.

```
struct tag { char c; int i; } normal_st;
extern struct tag little_st;

#define STRMOVE(DEST, SRC) DEST.c=SRC.c; DEST.i=SRC.i;

void main(void) {
    STRMOVE(little_st, normal_st);
}
```

In addition, the arrangement of the structure member is different in each compiler. Therefore, the member's arrangement might be different from that of the structure compiled by other compilers. In this case, a correct result is not obtained from the above-mentioned method.

Please do not arrange the structure variable in the little endian area when the arrangement of the structure member is different.

● Operations other than character type array that uses character string operation function

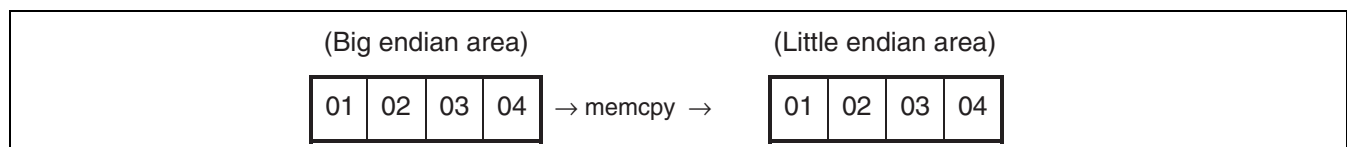
The character string operation function prepared as a standard library processes on a byte basis. Therefore, when processing by using the character string operation function is done to the area with types other than `char`, `unsigned char`, and the signed `char` type arranged in the little endian area, a correct result cannot be obtained.

Please do not use such processing.

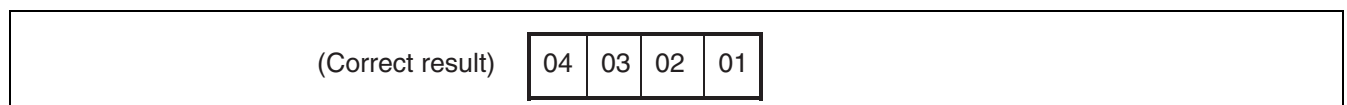
[Example of trouble] Transferring of word data with `memcpy`

```
int big = 0x01020304;    /* Big endian area      */
extern int little;        /* Little endian area    */
memcpy(&little, &big, 4); /* Transferring by memcpy */
```

The above-mentioned execution result is as follows:



The result of transferring the word data is incorrect.



● Specification of -K lib option when using character string operation function

When -K lib option is specified, the compiler performs inline expansion to some character string operation functions. At this time, since the best processing method is selected, the processing may be changed to the halfword or the word basis processing.

Therefore, processing to the little endian area is not executed correctly.

Please do not specify -K lib option when a process is executed to the little endian area by using the character string operation function.

Also, please do not specify either -O4 option or -K speed option that including -K lib option.

● Use of double type and long double type

The access to the double type and the long double type is done by using a method of accessing to one high-level word and one low-level word respectively. Therefore, accessing to the double type and the long double type variables arranged in the little endian area yields an incorrect result.

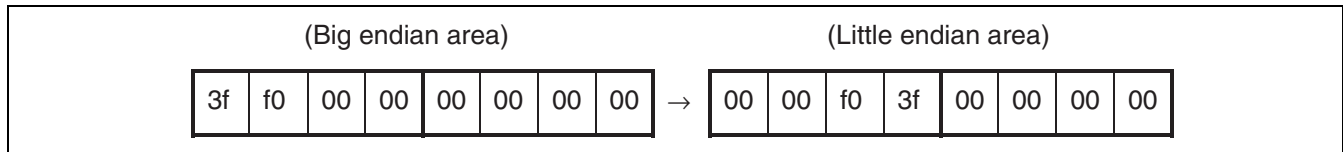
The substitution of the variables of the same type allocated in the little endian area is possible. However, these substitutions are occasionally replaced with the substitution of the constant as a result of optimization.

Please do not arrange either double type or long double type variables in the little endian area.

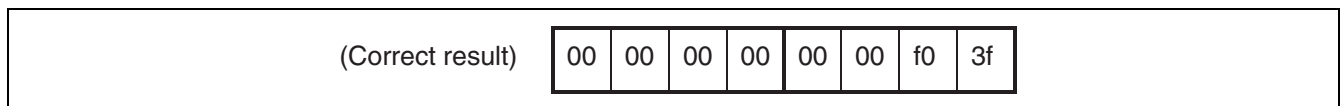
[Example of trouble] Transferring of double type data

```
double big = 1.0;      /* Big endian area      */
extern int little;     /* Little endian area    */
little = big;          /* Transferring of double type data */
```

The above-mentioned execution result is as follows:



The result of transferring the double type data is incorrect.



● Arrangement of stack in little endian area

When all or a part of the stack is arranged in the little endian area, operation is not guaranteed.

■ Assembler (fasm911)

For programming using the assembler language of FR family, operating suggestions relating to the little endian area are shown below.

● Section

The little endian area is used mainly for exchanging data to the little endian CPU. Therefore, please define the little endian area as a data section where an initial value does not exist.

When specifying a code, a stack, or a data section with an initial value to the little endian area, the access operation by MB91260B series cannot be guaranteed.

[Example]

```
/* Correct section definition in the little endian area */
.SECTION Little_Area, DATA, ALIGN=4
```

```
Little_Word:
    .RES.W 1
```

```
Little_Half:
    .RES.H 1
```

```
Little_Byte:
    .RES.B 1
```

● Data access

When accessing to the little endian area data, coding can be done to the value of the data without considering endians. However, please access to the little endian area data by using the same size access as the data size.

[Example]

```
LDI    #0x01020304, r0
LDI    #Little_Word, r1
```

```
LDI    #0x0102, r2
LDI    #Little_Half, r3
```

```
LDI    #0x01, r4
LDI    #Little_Byte, r5
```

```
/* Use a ST instruction (or a LD instruction, etc.) for 32-bit data access */
ST      r0, @r1
```

```
/* Use a STH instruction (or a LDH instruction, etc.) for 16-bit data access */
STH     r2, @r3
```

/* Use a STB instruction (or LDB instruction, etc.) for 8-bit data access */

STB r4, @r5

The value cannot be guaranteed when accessing it by a different data size in this MB91260B series. For example, when two consecutive 16-bit data are accessed at a time by using a 32-bit access instruction, the value of data cannot be guaranteed.

■ Linker (flink911)

For making the program that uses the little endian area, operating suggestions for the section arrangement when linking are shown below.

● Section type limitation

Only the data section without an initial value can be arranged in the little endian area.

When the data section with an initial value, stack section, and code section are arranged in the little endian area, since the arithmetic processing of the address solution etc. is performed in the big endian inside of the linker, so the program operation cannot be guaranteed.

● Undetected error

The linker does not recognize the little endian area. Therefore, the error message is not notified even if an arrangement that violates the above-mentioned limitations is done. Please confirm the content of the section arranged in the little endian area thoroughly before using the linker.

■ Debugger (sim911, eml911, mon911)

● Simulator debugger

There is no memory space specification command to indicate the little endian area.

Therefore, memory operation commands and instruction executions that operate memory are treated as big endian.

● Emulator debugger and monitor debugger

When the little endian area is accessed by the following commands, it is not treated as a normal value.

- set memory/show memory/enter/examine/set watch command
 - When using floating point (single/double) data, the specified value cannot be set or displayed.
- search memory command
 - When searching halfword and word data, the specified value cannot be searched.
- Line/reverse assemble (including reverse assemble display of the source window)
 - Normal instruction codes cannot be set or displayed (Please do not arrange the instruction code in the little endian area).
- call/show call command
 - When the stack area is placed in the little endian area, it does not operate properly (Please do not arrange the stack area in the little endian area).

APPENDIX E Instruction Lists

FR family instruction lists are shown below.

[How to read instruction lists]

Mnemonic	Type	OP	CYC	NZVC	Operation	Remarks
ADD Rj, Rj	A	AG	1	CCCC	Ri + Rj -> Rj	
*ADD #s5, Rj	C	A4	1	CCCC	Ri + s5 -> Ri	
,	,	,	,	,	,	
,	,	,	,	,	,	
(1)	(2)	(3)	(4)	(5)	(6)	(7)

(1) Instruction name

An asterisk (*) indicates an extended instruction that is not contained in the CPU specifications and is obtained by extension or addition by the assembler.

(2) Symbols indicating addressing modes that can be specified for the operand.

For the meaning of symbols, see "Addressing Mode Symbols (on the next page)".

(3) Instruction format

(4) Instruction code in hexadecimal notation

(5) Number of machine cycles

- a: Memory access cycle. It may be extended by the Ready function.
- b: Memory access cycle. It may be extended by the Ready function.
However, the cycle is interlocked if the instruction immediately after refers to a targeted register for LD operation, and the number of execution cycles is increased by 1.
- c: Interlocked if the instruction immediately after is an instruction that reads or writes to R15, SSP, or USP, or an instruction in instruction format A. The number of execution cycles is increased by 1 and so it becomes 2.
- d: Interlocked if the instruction immediately after refers to MDH/MDL. The number of execution cycles is increased to 2.
The minimum cycle number is 1 for each case a, b, c, and d.

(6) Indicating flag changes

Flag change
C: Change
-: No change
0: Clear
1: Set

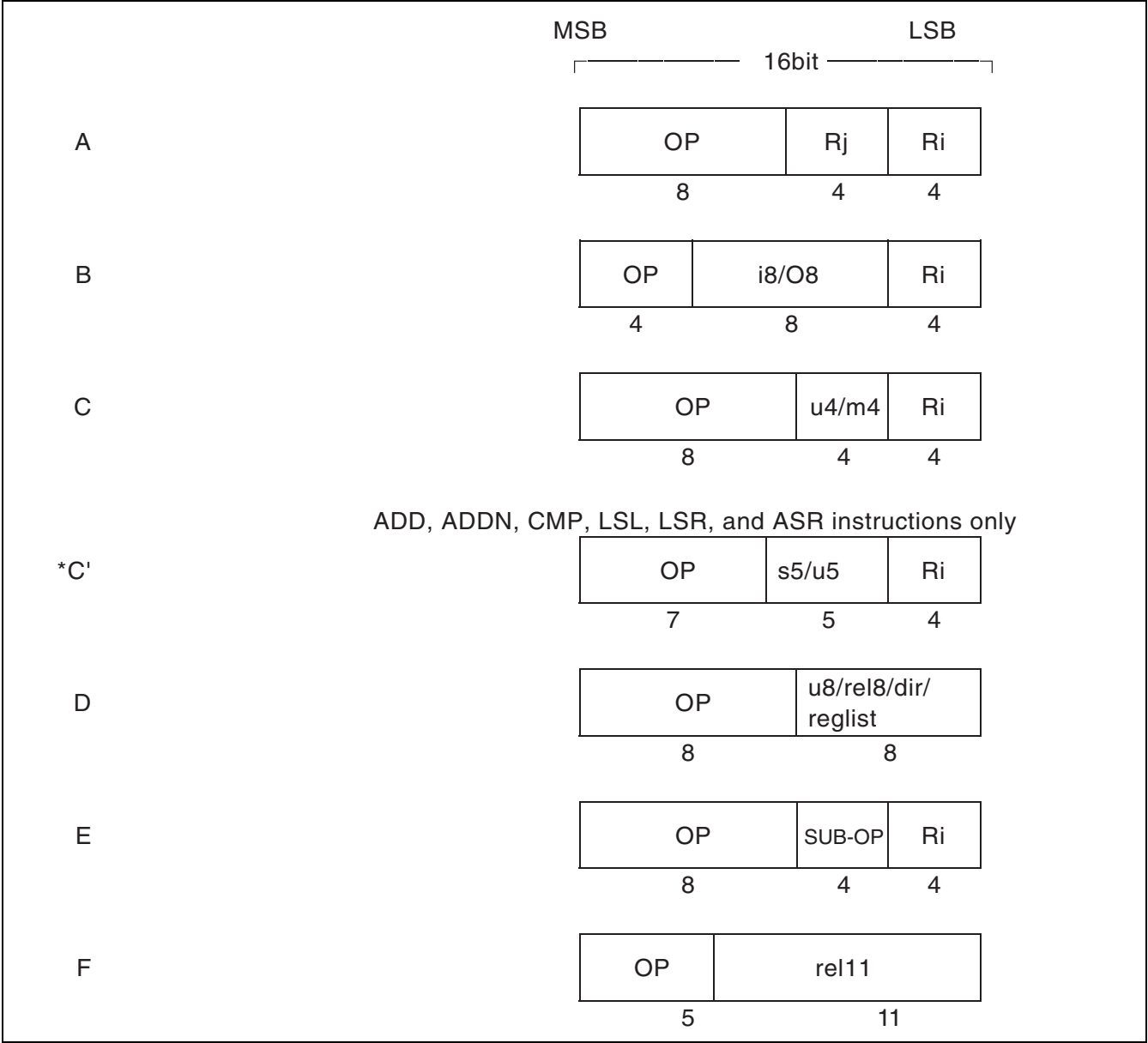
Flag meaning
N: Negative flag
Z: Zero flag
V: Over flag
C: Carry flag

(7) Instruction operation

● Addressing mode symbols

Ri	: Register direct (R0 to R15, AC, FP, SP)
Rj	: Register direct (R0 to R15, AC, FP, SP)
R13	: Register direct (R13, AC)
Ps	: Register direct (Program status register)
Rs	: Register direct (TBR, RP, SSP, USP, MDH, MDL)
Cri	: Register direct (CR0 to CR15)
CRj	: Register direct (CR0 to CR15)
#i8	: Unsigned 8-bit immediate (-128 to 255) Note: -128 to -1 is handled as 128 to 255.
#i20	: Unsigned 20-bit immediate (-0X80000 to 0XFFFFFF) Note: -0X7FFFF to -1 is handled as 0X7FFFF to 0XFFFFFF.
#i32	: Unsigned 32-bit immediate (-0X80000000 to 0xFFFFFFFF) Note: -0X80000000 to -1 is handled as 0X80000000 to 0xFFFFFFFF.
#s5	: Signed 5-bit immediate (-16 to 15)
#s10	: Signed 10-bit immediate (-512 to 508, multiples of 4 only)
#u4	: Unsigned 4-bit immediate (0 to 15)
#u5	: Unsigned 5-bit immediate (0 to 31)
#u8	: Unsigned 8-bit immediate (0 to 255)
#u10	: Unsigned 10-bit immediate (0 to 1020, multiples of 4 only)
@dir8	: Unsigned 8-bit direct address (0 to 0XFF)
@dir9	: Unsigned 9-bit direct address (0 to 0X1FE, multiples of 2 only)
@dir10	: Unsigned 10-bit direct address (0 to 0X3FC, multiples of 4 only)
label9	: Signed 9-bit branch address (-0X100 to 0XFC, multiples of 2 only)
label12	: Signed 12-bit branch address (-0X800 to 0X7FC, multiples of 2 only)
label20	: Signed 20-bit branch address (-0X80000 to 0X7FFFF)
label32	: Signed 32-bit branch address (-0X80000000 to 0X7FFFFFFF)
@Ri	: Register indirect (R0 to R15, AC, FP, SP)
@Rj	: Register indirect (R0 to R15, AC, FP, SP)
@(R13,Rj)	: Register relative indirect (Rj: R0 to R15, AC, FP, SP)
@(R14,disp10)	: Register relative indirect (disp10: -0X200 to 0X1FC, multiples of 4 only)
@(R14,disp9)	: Register relative indirect (disp9: -0X100 to 0XFE, multiples of 2 only)
@(R14,disp8)	: Register relative indirect (disp8: -0X80 to 0X7F)
@(R15,udisp6)	: Register relative indirect (udisp6: 0 to 60, multiples of 4 only)
@Ri+	: Register indirect with post-increment (R0 to R15, AC, FP, SP)
@R13+	: Register indirect with post-increment (R13, AC)
@SP+	: Stack pop
@-SP	: Stack push
(reglist)	: Register list

● Instruction format



Appendix Table E-1 Addition and Subtraction

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
ADD Rj, Ri	A	A6	1	CCCC	Ri+Rj->Ri	The assembler treats the highest-order 1 bit as the sign.
*ADD #s5, Ri	C'	A4	1	CCCC	Ri+s5->Ri	
ADD #u4, Ri	C	A4	1	CCCC	Ri+extu(i4)->Ri	Zero extension
ADD2 #u4, Ri	C	A5	1	CCCC	Ri+extu(i4)->Ri	Minus extension
ADDN Rj, Ri	A	A7	1	CCCC	Ri+Rj+c->Ri	Addition with carry
ADDN Rj, Ri	A	A2	1	----	Ri+Rj->Ri	The assembler treats the highest-order 1 bit as the sign.
*ADDN #s5, Ri	C'	A0	1	----	Ri+s5->Ri	
ADDN #u4, Ri	C	A0	1	----	Ri+extu(i4)->Ri	Zero extension
ADDN2 #u4, Ri	C	A1	1	----	Ri+extu(i4)->Ri	Minus extension
SUB Rj, Ri	A	AC	1	CCCC	Ri-Rj->Ri	
SUBC Rj, Ri	A	AD	1	CCCC	Ri-Rj-c->Ri	Subtraction with carry
SUBN Rj, Ri	A	AE	1	----	Ri-Rj->Ri	

Appendix Table E-2 Comparison Operation

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
CMP Rj, Ri	A	AA	1	CCCC	Ri-Rj	The assembler treats the highest-order 1 bit as the sign.
*CMP #s5, Ri	C'	A8	1	CCCC	Ri-s5	
CMP #u4, Ri	C	A8	1	CCCC	Ri-extu(i4)	Zero extension
CMP2 #u4, Ri	C	A9	1	CCCC	Ri-extu(i4)	Minus extension

Appendix Table E-3 Logic Operation

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMW	Remarks
AND Rj, Ri	A	82	1	CC--	$Ri \ \& \ = Rj$	-	Word
AND Rj, @Ri	A	84	1+2a	CC--	$(Ri) \ \& \ = Rj$	○	Word
ANDH Rj, @Ri	A	85	1+2a	CC--	$(Ri) \ \& \ = Rj$	○	Halfword
ANDB Rj, @Ri	A	86	1+2a	CC--	$(Ri) \ \& \ = Rj$	○	Byte
OR Rj, Ri	A	92	1	CC--	$Ri \ \ = Rj$	-	Word
OR Rj, @Ri	A	94	1+2a	CC--	$(Ri) \ \ = Rj$	○	Word
ORH Rj, @Ri	A	95	1+2a	CC--	$(Ri) \ \ = Rj$	○	Halfword
ORB Rj, @Ri	A	96	1+2a	CC--	$(Ri) \ \ = Rj$	○	Byte
EOR Rj, Ri	A	9A	1	CC--	$Ri \ \wedge \ = Rj$	-	Word
EOR Rj, @Ri	A	9C	1+2a	CC--	$(Ri) \ \wedge \ = Rj$	○	Word
EORH Rj, @Ri	A	9D	1+2a	CC--	$(Ri) \ \wedge \ = Rj$	○	Halfword
EORB Rj, @Ri	A	9E	1+2a	CC--	$(Ri) \ \wedge \ = Rj$	○	Byte

Appendix Table E-4 Bit Manipulation Instruction

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMW	Remarks
BANDL #u4, @Ri	C	80	1+2a	----	$(Ri) \ \& \ = (0xF0 + u4)$	○	Low-order 4 bits are manipulated.
BANDH #u4, @Ri	C	81	1+2a	----	$(Ri) \ \& \ = ((u4 \ll 4) + 0x0F)$	○	High-order 4 bits are manipulated.
*BAND #u8, @Ri _{*1}				----	$(Ri) \ \& \ = u8$	-	
BORL #u4, @Ri	C	90	1+2a	----	$(Ri) \ \ = u4$	○	Low-order 4 bits are manipulated.
BORH #u4, @Ri	C	91	1+2a	----	$(Ri) \ \ = (u4 \ll 4)$	○	High-order 4 bits are manipulated.
*BOR #u8, @Ri _{*2}				----	$(Ri) \ \ = u8$	-	
BEORL #u4, @Ri	C	98	1+2a	----	$(Ri) \ \wedge \ = u4$	○	Low-order 4 bits are manipulated.
BEORH #u4, @Ri	C	99	1+2a	----	$(Ri) \ \wedge \ = (u4 \ll 4)$	○	High-order 4 bits are manipulated.
*BEOR #u8, @Ri _{*3}				----	$(Ri) \ \wedge \ = u8$	-	
BTSTL #u4, @Ri	C	88	2+a	0C--	$(Ri) \ \& \ u4$	-	Low-order 4 bits are tested.
BTSTH #u4, @Ri	C	89	2+a	CC--	$(Ri) \ \& \ (u4 \ll 4)$	-	High-order 4 bits are tested.

*1: The assembler generates BANDL if the bit is set at $u8 \ \& \ 0x0F$, and BANDH if the bit is set at $u8 \ \& \ 0xF0$. In some cases, both BANDL and BANDH may be generated.

*2: The assembler generates BORL if the bit is set at $u8 \ \& \ 0x0F$, and BORH if the bit is set at $u8 \ \& \ 0xF0$. In some cases, both BORL and BORH are generated.

*3: The assembler generates BEORL if the bit is set at $u8 \ \& \ 0x0F$, and BEORH if the bit is set at $u8 \ \& \ 0xF0$. In some cases, both BEORL and BEORH are generated.

Appendix Table E-5 Multiplication and Division

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
MUL Rj,Ri	A	AF	5	CCC-	Ri * Rj -> MDH, MDL	32bit*32bit=64bit
MULU Rj,Ri	A	AB	5	CCC-	Ri * Rj -> MDH, MDL	No sign
MULH Rj,Ri	A	BF	3	CC--	Ri * Rj -> MDL	16bit*16bit=32bit
MULUH Rj,Ri	A	BB	3	CC--	Ri * Rj -> MDL	No sign
DIV0S Ri	E	97-4	1	----		Step operation
DIV0U Ri	E	97-5	1	----		32bit/32bit=32bit
DIV1 Ri	E	97-6	d	-C-C		
DIV2 Ri	E	97-7	1	-C-C		
DIV3	E	9F-6	1	----		
DIV4S	E	9F-7	1	----		
*DIV Ri	*1		36	-C-C	MDL / Ri -> MDL , MDL % Ri -> MDH	
*DIVU Ri	*2			-C-C	MDL / Ri -> MDL , MDL % Ri -> MDH	

Appendix Table E-6 Shift

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LSL Rj, Ri	A	B6	1	CC-C	Ri << Rj -> Ri	Logical shift
*LSL #u5, Ri(u5:0 to 31)	C'	B4	1	CC-C	Ri << u5 -> Ri	
LSL #u4, Ri	C	B4	1	CC-C	Ri << u4 -> Ri	
LSL2 #u4, Ri	C	B5	1	CC-C	Ri <<(u4+16) -> Ri	
LSR Rj, Ri	A	B2	1	CC-C	Ri >> Rj -> Ri	Logical shift
*LSR #u5, Ri(u5:0 to 31)	C'	B0	1	CC-C	Ri >> u5 -> Ri	
LSR #u4, Ri	C	B0	1	CC-C	Ri >> u4 -> Ri	
LSR2 #u4, Ri	C	B1	1	CC-C	Ri >>(u4+16) -> Ri	
ASR Rj, Ri	A	BA	1	CC-C	Ri >> Rj -> Ri	Arithmetic shift
*ASR #u5, Ri (u5:0 to 31)	C'	B8	1	CC-C	Ri >> u5 -> Ri	
ASR #u4, Ri	C	B8	1	CC-C	Ri >> u4 -> Ri	
ASR2 #u4, Ri	C	B9	1	CC-C	Ri >>(u4+16) -> Ri	

Appendix Table E-7 Immediate Value Set/16-Bit/32-Bit Immediate Value Transfer Instruction

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LDI:32 #i32, Ri	E	9F-8	3	----	i32 -> Ri	High-order 12 bits are zero-extended.
LDI:20 #i20, Ri	C	9B	2	----	i20 -> Ri	High-order 24 bits are zero-extended.
LDI:8 #i8, Ri	B	C0	1	----	i8 -> Ri	
*LDI # {i8 i20 i32} ,Ri	*3				{i8 i20 i32} -> Ri	

*1: DIV0S, DIV1 x 32, DIV2, DIV3, and DIV4S are generated. The instruction code length becomes 72 bytes.

*2: DIV0U and DIV1 x 32 are generated. The instruction code length becomes 66 bytes.

*3: If the immediate value is an absolute value, i8, i20, or i32 is selected automatically by the assembler.
If immediate value contains a relative value or an external reference symbol, i32 is selected.

Appendix Table E-8 Memory Load

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LD @Rj, Ri	A	04	b	----	(Rj)->Ri	Rs: Special register *1
LD @(R13,Rj), Ri	A	00	b	----	(R13+Rj)->Ri	
LD @(R14,disp10), Ri	B	2	b	----	(R14+disp10)->Ri	
LD @(R15,udisp6), Ri	C	03	b	----	(R15+udisp6)->Ri	
LD @R15+, Ri	E	07-0	b	----	(R15)->Ri, R15+=4	
LD @R15+, Rs	E	07-8	b	----	(R15)->Rs, R15+=4	
LD @R15+, PS	E	07-9	1+a+b	CCCC	(R15)->PS, R15+=4	
LDUH @Rj, Ri	A	05	b	----	(Rj)->Ri	Zero extension
LDUH @(R13,Rj), Ri	A	01	b	----	(R13+Rj)->Ri	Zero extension
LDUH @(R14,disp9), Ri	B	4	b	----	(R14+disp9)->Ri	Zero extension
LDUB @Rj, Ri	A	06	b	----	(Rj)->Ri	Zero extension
LDUB @(R13,Rj), Ri	A	02	b	----	(R13+Rj)->Ri	Zero extension
LDUB @(R14,disp8), Ri	B	6	b	----	(R14+disp8)->Ri	Zero extension

*1: In the o8 and o4 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
disp10/4->o8, disp9/2->o8, disp8->o8, disp10, disp9, and disp8 have a sign, udisp6/4->o4 udisp6 has no sign.

Appendix Table E-9 Memory Store

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
ST Ri,@Rj	A	14	a	----	Ri->(Rj)	Word Word Word Rs Special register *1
ST Ri,@(R13,Rj)	A	10	a	----	Ri->(R13+Rj)	
ST Ri,@(R14,disp10)	B	3	a	----	Ri->(R14+disp10)	
ST Ri,@(R15,udisp6)	C	13	a	----	Ri->(R15+udisp6)	
ST Ri,@-R15	E	17-0	a	----	R15-=4, Ri->(R15)	
ST Rs,@-R15	E	17-8	a	----	R15-=4, Rs->(R15)	
ST PS,@-R15	E	17-9	a	----	R15-=4, PS->(R15)	
STH Ri,@Rj	A	15	a	----	Ri->(Rj)	Halfword
STH Ri,@(R13,Rj)	A	11	a	----	Ri->(R13+Rj)	Halfword
STH Ri,@(R14,disp9)	B	5	a	----	Ri->(R14+disp9)	Halfword
STB Ri,@Rj	A	16	a	----	Ri->(Rj)	Byte
STB Ri,@(R13,Rj)	A	12	a	----	Ri->(R13+Rj)	Byte
STB Ri,@(R14,disp8)	B	7	a	----	Ri->(R14+disp8)	Byte

*1: In the o8 and o4 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
disp10/4->o8, disp9/2->o8, disp8->o8, disp10, disp9, and disp8 have a sign, udisp6/4->o4 udisp6 has no sign.

Appendix Table E-10 Register-to-Register Transfer

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
MOV Rj, Ri	A	8B	1	----	Rj -> Ri	Transfer between general-purpose registers Rs: Special register Rs: Special register *1
MOV Rs, Ri	A	B7	1	----	Rs -> Ri	
MOV Ri, Rs	E	B3	1	----	Ri -> Rs	
MOV PS, Ri	E	17-1	1	----	PS -> Ri	
MOV Ri, PS	E	07-1	c	CCCC	Ri -> PS	

*1: Special register Rs: TBR, RP, USP, SSP, MDH, and MDL

Appendix Table E-11 Normal Branch (No Delay)

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
JMP @Ri	E	97-0	2	----	Ri -> PC	
CALL label12	E	D0	2	----	PC+2->RP , PC+2+(label12-PC-2)->PC	
CALL @Ri	F	97-1	2	----	PC+2->RP ,Ri->PC	
RET	E	97-2	2	----	RP -> PC	Return
INT #u8	D	1F	3+3a	----	SSP-=4, PS->(SSP), SSP-=4, PC+2->(SSP), 0->I Flag, 0->S Flag, (TBR+0x3FC-u8 × 4)->PC	
INTE	E	9F-3	3+3a	----	SSP-=4, PS->(SSP), SSP-=4, PC+2->(SSP), 0->S Flag,(TBR+0x3D8)->PC	For emulator
RETI	E	97-3	2+2A	CCCC	(R15)->PC,R15-=4,(R15)->PS,R15-=4	
BRA label9	D	E0	2	----	PC+2+(label9-PC-2)->PC	
BNO label9	D	E1	1	----	No branch	
BEQ label9	D	E2	2/1	----	if(Z==1) then PC+2+(label9-PC-2)->PC	
BNE label9	D	E3	2/1	----	↑ s/Z==0	
BC label9	D	E4	2/1	----	↑ s/C==1	
BNC label9	D	E5	2/1	----	↑ s/C==0	
BN label9	D	E6	2/1	----	↑ s/N==1	
BP label9	D	E7	2/1	----	↑ s/N==0	
BV label9	D	E8	2/1	----	↑ s/V==1	
BNV label9	D	E9	2/1	----	↑ s/V==0	
BLT label9	D	EA	2/1	----	↑ s/V xor N==1	
BGE label9	D	EB	2/1	----	↑ s/V xor N==0	
BLE label9	D	EC	2/1	----	↑ s/(V xor N) or Z==1	
BGT label9	D	ED	2/1	----	↑ s/(V xor N) or Z==0	
BLS label9	D	EE	2/1	----	↑ s/C or Z==1	
BHI label9	D	EF	2/1	----	↑ s/C or Z==0	

Notes:

- "2/1" under CYCLE indicates "2" cycles when branching occurs and "1" cycle when branching does not occur.
- In the rel11 and rel8 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
(label12-PC-2)/2->rel11, (label9-PC-2)/2->rel8, label12 and label9 have a sign.
- To execute the RETI instruction, the S flag must be set to "0".

Appendix Table E-12 Delayed Branch

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
JMP:D @Ri	E	9F-0	1	----	Ri -> PC	
CALL:D label12	F	D8	1	----	PC+4->RP , PC+2+(label12-PC-2)->PC	
CALL:D @Ri	E	9F-1	1	----	PC+4->RP ,Ri->PC	
RET:D	E	9F-2	1	----	RP -> PC	Return
BRA:D label9	D	F0	1	----	PC+2+(label9-PC-2)->PC	
BNO:D label9	D	F1	1	----	No branch	
BEQ:D label9	D	F2	1	----	if(Z==1) then PC+2+(label9-PC-2)->PC	
BNE:D label9	D	F3	1	----	↑ s/Z==0	
BC:D label9	D	F4	1	----	↑ s/C==1	
BNC:D label9	D	F5	1	----	↑ s/C==0	
BN:D label9	D	F6	1	----	↑ s/N==1	
BP:D label9	D	F7	1	----	↑ s/N==0	
BV:D label9	D	F8	1	----	↑ s/V==1	
BNV:D label9	D	F9	1	----	↑ s/V==0	
BLT:D label9	D	FA	1	----	↑ s/V xor N==1	
BGE:D label9	D	FB	1	----	↑ s/V xor N==0	
BLE:D label9	D	FC	1	----	↑ s/(V xor N) or Z==1	
BGT:D label9	D	FD	1	----	↑ s/(V xor N) or Z==0	
BLS:D label9	D	FE	1	----	↑ s/C or Z==1	
BHI:D label9	D	FF	1	----	↑ s/C or Z==0	

Notes:

- In the rel11 and rel8 fields of the hardware specifications, the assembler calculates values and sets them as shown below:
(label12-PC-2)/2->rel11, (label9-PC-2)/2->rel8, label12 and label9 have a sign.
- A delayed branch always occurs after the next instruction (delay slot) is executed.
- Instructions that can be placed in the delay slot are all 1-cycle, a-, b-, c-, and d-cycle instructions.
Multi-cycle instructions cannot be placed in the delay slot.

Appendix Table E-13 Other Instructions

Mnemonic	Type	OP	CYCLE	NZVC	Operation	RMW	Remarks
NOP	E	9F-A	1	----	No change	-	
ANDCCR #u8	D	83	c	cccc	CCR and u8 -> CCR	-	
ORCCR #u8	D	93	c	cccc	CCR or u8 -> CCR	-	
STILM #u8	D	87	1	----	i8 -> ILM	-	ILM Immediate set
ADDSP #s10 ^{*1}	D	A3	1	----	R15 += s10	-	ADD SP instruction
EXTSB Ri	E	97-8	1	----	Sign extension 8->32bit	-	
EXTUB Ri	E	97-9	1	----	Zero extension 8->32bit	-	
EXTSH Ri	E	97-A	1	----	Sign extension 16->32bit	-	
EXTUH Ri	E	97-B	1	----	Zero extension 16->32bit	-	
LDM0 (reglist)	D	8C		----	(R15)->reglist, R15 increment	-	Load multi R0-R7
LDM1 (reglist)	D	8D		----	(R15)->reglist, R15 increment	-	Load multi R8-R15
*LDM (reglist) ^{*2}				----	(R15)->reglist, R15 increment	-	Load multi R0-R15
STM0 (reglist)	D	8E		----	R15 decrement reglist->(R15)	-	Store multi R0-R7
STM1 (reglist)	D	8F		----	R15 decrement reglist->(R15)	-	Store multi R8-R15
*STM (reglist) ^{*3}				----	R15 decrement reglist->(R15)	-	Store multi R0-R15
ENTER #u10 ^{*4}	D	0F	1+a	----	R14 -> (R15 - 4), R15 - 4 -> R14, R15 - u10 -> R15	-	Entry processing of a function
LEAVE	E	9F-9	b	----	R14 + 4 -> R15, (R15 - 4) -> R14	-	Exit processing of a function
XCHB @Rj, Ri	A	8A	2a	----	Ri -> TEMP (Rj) -> Ri TEMP -> (Rj)	○	For semaphore management Byte data

*1: For s10, the assembler calculates s10/4 and then changes to s8 to set a value. s10 has a sign.

*2: If any of R0 to R7 is specified in reglist, LDM0 is generated, and if any of R8 to R15 is specified, LDM1 is generated. In some cases, both LDM0 and LDM1 are generated.

*3: If any of R0 to R7 is specified in reglist, STM0 is generated, and if any of R8 to R15 is specified, STM1 is generated. In some cases, both STM0 and STM1 are generated.

*4: For u10, the assembler calculates u10/4 and then changes to u8 to set a value. u10 has no sign.

Notes:

- The number of execution cycles of LDM0(reglist) and LDM1(reglist) can be calculated as $a \times (n-1) + b + 1$ cycles if the number of specified registers is n.
- The number of execution cycles of STM0(reglist) and STM1(reglist) can be calculated as $a \times n + 1$ cycles if the number of specified registers is n.

Appendix Table E-14 20-Bit Normal Branch Macro Instruction

Mnemonic	Operation	Remarks
*CALL20 label20,Ri	Address of the next instruction ->RP, label20->PC	Ri: Temporary register (See Reference 1)
*BRA20 label20,Ri	label20->PC	Ri: Temporary register (See Reference 2)
*BEQ20 label20,Ri	if(Z==1) then label20->PC	Ri: Temporary register (See Reference 3)
*BNE20 label20,Ri	↑ s/Z==0	↑
*BC20 label20,Ri	↑ s/C==1	↑
*BNC20 label20,Ri	↑ s/C==0	↑
*BN20 label20,Ri	↑ s/N==1	↑
*BP20 label20,Ri	↑ s/N==0	↑
*BV20 label20,Ri	↑ s/V==1	↑
*BNV20 label20,Ri	↑ s/V==0	↑
*BLT20 label20,Ri	↑ s/V xor N==1	↑
*BGE20 label20,Ri	↑ s/V xor N==0	↑
*BLE20 label20,Ri	↑ s/(V xor N) or Z==1	↑
*BGT20 label20,Ri	↑ s/(V xor N) or Z==0	↑
*BLS20 label20,Ri	↑ s/C or Z==1	↑
*BHI20 label20,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL20

(1) If label20-PC-2 is between -0x800 and +0x7fe, the following instruction will be generated:

CALL label12

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:20 #label20,Ri

CALL @Ri

[Reference 2] BRA20

(1) If label20-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

BRA label9

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:20 #label20,Ri

JMP @Ri

[Reference 3] Bcc20

(1) If label20-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

Bcc label9

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

Bxcc false xcc is the opposite condition of cc.

LDI:20 #label20,Ri

JMP @Ri

false:

Appendix Table E-15 20-Bit Delayed Branch Macro Instruction

Mnemonic	Operation	Remarks
*CALL20:D label20,Ri	Address of the next instruction +2->RP, label20->PC	RRi: Temporary register (See Reference 1)
*BRA20:D label20,Ri	label20->PC	Ri: Temporary register (See Reference 2)
*BEQ20:D label20,Ri	if(Z==1) then label20->PC	Ri: Temporary register (See Reference 3)
*BNE20:D label20,Ri	↑ s/Z==0	↑
*BC20:D label20,Ri	↑ s/C==1	↑
*BNC20:D label20,Ri	↑ s/C==0	↑
*BN20:D label20,Ri	↑ s/N==1	↑
*BP20:D label20,Ri	↑ s/N==0	↑
*BV20:D label20,Ri	↑ s/V==1	↑
*BNV20:D label20,Ri	↑ s/V==0	↑
*BLT20:D label20,Ri	↑ s/V xor N==1	↑
*BGE20:D label20,Ri	↑ s/V xor N==0	↑
*BLE20:D label20,Ri	↑ s/(V xor N) or Z==1	↑
*BGT20:D label20,Ri	↑ s/(V xor N) or Z==0	↑
*BLS20:D label20,Ri	↑ s/C or Z==1	↑
*BHI20:D label20,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL20:D

(1) If label20-PC-2 is between -0x800 and +0x7fe, the following instruction will be generated:

CALL:D label12

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:20 #label20,Ri

CALL:D @Ri

[Reference 2] BRA20:D

(1) If label20-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

BRA:D label9

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:20 #label20,Ri

JMP:D @Ri

[Reference 3] Bcc20:D

(1) If label20-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

Bcc:D label9

(2) If label20-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

Bxcc false xcc is the opposite condition of cc.

LDI:20 #label20,Ri

JMP:D @Ri

false:

Appendix Table E-16 32-Bit Normal Branch Macro Instruction

Mnemonic	Operation	Remarks
*CALL32 label32,Ri	Address of the next instruction ->RP, label32->PC	Ri: Temporary register (See Reference 1)
*BRA32 label32,Ri	label32->PC	Ri: Temporary register (See Reference 2)
*BEQ32 label32,Ri	if(Z==1) then label32->PC	Ri: Temporary register (See Reference 3)
*BNE32 label32,Ri	↑ s/Z==0	↑
*BC32 label32,Ri	↑ s/C==1	↑
*BNC32 label32,Ri	↑ s/C==0	↑
*BN32 label32,Ri	↑ s/N==1	↑
*BP32 label32,Ri	↑ s/N==0	↑
*BV32 label32,Ri	↑ s/V==1	↑
*BNV32 label32,Ri	↑ s/V==0	↑
*BLT32 label32,Ri	↑ s/V xor N==1	↑
*BGE32 label32,Ri	↑ s/V xor N==0	↑
*BLE32 label32,Ri	↑ s/(V xor N) or Z==1	↑
*BGT32 label32,Ri	↑ s/(V xor N) or Z==0	↑
*BLS32 label32,Ri	↑ s/C or Z==1	↑
*BHI32 label32,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL32

(1) If label32-PC-2 is between -0x800 and +0x7fe, the following instruction will be generated:

CALL label12

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:32 #label32,Ri

CALL @Ri

[Reference 2] BRA32

(1) If label32-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

BRA label9

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:32 #label32,Ri

JMP @Ri

[Reference 3] Bcc32

(1) If label32-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

Bcc label9

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

Bxcc false xcc is the opposite condition of cc.

LDI:32 #label32,Ri

JMP @Ri

false:

Appendix Table E-17 32-Bit Delayed Branch Macro Instruction

Mnemonic	Operation	Remarks
*CALL32D label32,Ri	Address of the next instruction +2->RP, label32->PC	Ri: Temporary register (See Reference 1)
*BRA32:D label32,Ri	label32->PC	Ri: Temporary register (See Reference 2)
*BEQ32:D label32,Ri	if(Z==1) then label32->PC	Ri: Temporary register (See Reference 3)
*BNE32:D label32,Ri	↑ s/Z==0	↑
*BC32:D label32,Ri	↑ s/C==1	↑
*BNC32:D label32,Ri	↑ s/C==0	↑
*BN32:D label32,Ri	↑ s/N==1	↑
*BP32:D label32,Ri	↑ s/N==0	↑
*BV32:D label32,Ri	↑ s/V==1	↑
*BNV32:D label32,Ri	↑ s/V==0	↑
*BLT32:D label32,Ri	↑ s/V xor N==1	↑
*BGE32:D label32,Ri	↑ s/V xor N==0	↑
*BLE32:D label32,Ri	↑ s/(V xor N) or Z==1	↑
*BGT32:D label32,Ri	↑ s/(V xor N) or Z==0	↑
*BLS32:D label32,Ri	↑ s/C or Z==1	↑
*BHI32:D label32,Ri	↑ s/C or Z==0	↑

[Reference 1] CALL32:D

(1) If label32-PC-2 is between -0x800 and +0x7fe, the following instruction will be generated:

CALL:D label12

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:32 #label32,Ri

CALL:D @Ri

[Reference 2] BRA32:D

(1) If label32-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

BRA:D label9

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

LDI:32 #label32,Ri

JMP:D @Ri

[Reference 3] Bcc32:D

(1) If label32-PC-2 is between -0x100 and +0xfe, the following instruction will be generated:

Bcc:D label9

(2) If label32-PC-2 is outside the range of (1) or contains an external reference symbol, the following instruction will be generated:

Bxcc false xcc is the opposite condition of cc.

LDI:32 #label32,Ri

JMP:D @Ri

false:

Appendix Table E-18 Direct Addressing

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
DMOV @dir10, R13	D	08	b	----	(dir10)-> R13	Word
DMOV R13, @dir10	D	18	a	----	R13 ->(dir10)	Word
DMOV @dir10, @R13+	D	0C	2a	----	(dir10)->(R13),R13+=4	Word
DMOV @R13+, @dir10	D	1C	2a	----	(R13)->(dir10),R13+=4	Word
DMOV @dir10, @-R15	D	0B	2a	----	R15-=4,(R15)->(dir10)	Word
DMOV @R15+, @dir10	D	1B	2a	----	(R15)->(dir10),R15+=4	Word
DMOVH @dir9, R13	D	09	b	----	(dir9)-> R13	Halfword
DMOVH R13, @dir9	D	19	a	----	R13 ->(dir9)	Halfword
DMOVH @dir9, @R13+	D	0D	2a	----	(dir9)->(R13),R13+=2	Halfword
DMOVH @R13+, @dir9	D	1D	2a	----	(R13)->(dir9),R13+=2	Halfword
DMOV B @dir8, R13	D	0A	b	----	(dir8)-> R13	Byte
DMOV B R13, @dir8	D	1A	a	----	R13 ->(dir8)	Byte
DMOV B @dir8, @R13+	D	0E	2a	----	(dir8)->(R13),R13++	Byte
DMOV B @R13+, @dir8	D	1E	2a	----	(R13)->(dir8),R13++	Byte

Note: In the dir8, dir9, and dir10 fields, the assembler calculates values and sets them as shown below:

dir8->dir, dir9/2->dir, dir10/4->dir dir8, dir9, and dir10 have no sign.

Appendix Table E-19 Resource Instruction

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
LDRES @Ri+, #u4	C	BC	a	----	Resource of u4 Ri+=4	u4: Channel No.
STRES #u4, @Ri+	C	BD	a	----	Resource of u4->(Ri) Ri+=4	u4: Channel No.

Note: These instructions cannot be used in this MB91260B series since resource having channel number is not installed.

Appendix Table E-20 Coprocessor Control Instruction

{CRi|CRj} := CR0 | CR1 | CR2 | CR3 | CR4 | CR5 | CR6 | CR7 | CR8 | CR9 | CR10 | CR11 | CR12 | CR13 | CR14 | | CR15

u4: := Specify channel

u8: := Specify command

Mnemonic	Type	OP	CYCLE	NZVC	Operation	Remarks
COPOP #u4, #u8, CRj, Cri	E	9F-C	2+a	----	Operation instruction	No error stop
COPLD #u4, #u8, Rj, Cri	E	9F-D	1+2a	----	Rj -> CRi	
COPST #u4, #u8, CRj, Ri	E	9F-E	1+2a	----	CRj -> Ri	
COPSV #u4, #u8, CRj, Ri	E	9F-F	1+2a	----	CRj -> Ri	

Note: These instructions cannot be used in this MB91260B series since coprocessor is not installed.

APPENDIX F Precautions on Handling

■ Common Items

● Clock controller

When inputting "L" to $\overline{\text{INIT}}$, the oscillation stabilization wait time must be secured.

● Switching functions between port and pin

When a port serves both as a port and a pin, port and pin functions are switched by PFR (Port Function Register). However, a bus pin is switched by external bus settings.

Note: External bus mode is not supported by the MB91260B series.

● D-bus memory

Do not set the code area in the D-bus memory.

No instruction fetch to the D-bus memory is executed. If instruction fetch to the D-bus memory is executed, incorrect data will be interpreted as a code, causing a runaway condition.

● Low-power consumption mode

(1) To switch to standby mode, use synchronous standby mode (set by the SYNCS bit, that is bit8 of the TBCR, time-base counter control register) and be sure to use the following sequence:

/* Writing STCR */

```
ldi    #_STCR, R0          ; STCR register (0x0481)
ldi    #Val_of_Stby, r1     ; Val_of_Stby is the write data to STCR.
stb    r1, @r0             ; Writing to STCR
```

/* Writing STBR */

```
ldi    #_CTBR, r2          ; CTBR register (0x0483)
ldi    #0xA5, r1           ; Clear command (1)
stb    r1, @r2             ; Writing A5 to CTBR
ldi    #0xA5, r1           ; Clear command (2)
stb    r1, @r2             ; Writing A5 to CTBR
```

/* Clearing the time base counter in here */

```
ldub    @r0, r1            ; Reading STCR
```

/* Starting synchronous standby transition */

```
ldub    @r0, r1            ; Reading dummy STCR
nop                                           ; NOP × 5 for timing adjustment
nop
nop
nop
nop
```

(2) When using the monitor debugger, DO NOT:

- Set a break point within the above sequence of instructions.
- Execute step within the above sequence of instructions.

● Notes on using PS register

PS register is processed by some instructions in advance so that exception operations as stated below may cause breaks during interruption processing routine when using debugger and may cause updates to the display contents of PS flags.

In either case, this device is designed to carry out reprocessing properly after returning from such EIT events. The operations before and after the events are performed as prescribed in the specification.

1. If the instruction immediately before DIV0U/DIV0S instruction (a) receives a user interrupt/NMI, (b) executes stepping or (c) breaks by data event or emulator menu, the following operations may be performed.
 - (1) D0 and D1 flags are updated in advance.
 - (2) EIT handling routine (user interrupt/NMI, or emulator) is executed.
 - (3) After returning from the EIT, DIV0U/DIV0S instruction is executed and the D0 and D1 flags are updated to the same values as in (1).
2. The following operations are performed if each instruction from ORCCR, STILM, MOV Ri and PS is executed to allow an interruption while user interrupt/NMI trigger exists.
 - (1) PS register is updated in advance.
 - (2) EIT handling routine (user interrupt/NMI) is executed.
 - (3) After returning from the EIT, the above instructions are executed and the PS register is updated to the same value as in (1).

● Watchdog timer function

The watchdog timer equipped in this model operates to monitor programs to ensure that they execute reset defer function within a certain period of time and to reset the CPU if the reset defer function is not executed due to the program runaway. Therefore, once the watchdog timer function is enabled, it keeps its operation until it is reset.

By way of exception, the watchdog timer automatically defers a reset under the conditions where the CPU program executions are stopped. For more detail about such conditions, refer to the description section of the watchdog timer function.

■ Notes on Using Debugger

● Stepping of the RETI instruction

In the environment where interruptions occur frequently during stepping, the RETI is executed repeatedly for the corresponding interrupt processing routines after the stepping. As the result of it, the main routine and low-interrupt-level programs are not executed.

To avoid this situation, do not step the RETI instruction. Otherwise, perform debugging by disabling the interruptions when the debug on the corresponding interrupt routines becomes unnecessary.

● Operand break

Do not set the access to the areas containing the address of system stack pointer as a target of data event break.

● Debugging on unused area of FLASH memory

If debug is performed on the unused area of FLASH memory (where data is 0XFFFF_H) by mistake, break will not be accepted. To avoid this situation, the use of address mask function of debugger code event is recommended to break when an instruction is accessing to the unused area.

● Power-on debug

When powering off by the power-on debug, be sure to power off under the condition that all the following three requirements are met.

(1) Time taken to decrease user power supply from 0.9 V_{CC} to 0.5 V_{CC} is 25 μs or more.

Note: If using two power supplies, V_{CC} is external I/O power supply voltage.

(2) CPU operating frequency is 1 MHz or higher.

(3) User program is in execution.

● Interrupt handler for NMI request (tool)

When ICE is unconnected, to prevent the malfunction caused by noise or other sources affecting DSU pin to set a flag which is only set by the break request from ICE, add the following programs to the interrupt handler.

ICE can be used even if this program is added.

Location to add

The following interrupt handler

Interrupt factor	: NMI request (tool)
Interrupt No.	: 13 (decimal), 0D (hexadecimal)
Offset	: 3C8 _H
TBR as a default address	: 000FFFC8 _H

Additional program

```

STM    (R0, R1)
LDI    #B00H, R0    ; B00H is the address of break factor register of DSU.
LDI    #0, R1
STB    R1, @R0      ; Clear break factor register.
LDM    (R0, R1)
RETI

```


INDEX

**The index follows on the next page.
This is listed in alphabetic order.**

Index

Numerics

0 Detection	
0 Detection	148
1 Detection	
1 Detection	148
16-bit Dead Timer Control Register	
16-bit Dead Timer Control Register (DTCR0)	245
16-bit Dead Timer Control Register (DTCR1)	247
16-bit Dead Timer Control Register (DTCR2)	249
16-bit Dead Timer Register	
16-bit Dead Timer Register	
(TMRRH: TMRRH0 to TMRRH2,	
TMRRL: TMRRL0 to TMRRL2)	244
16-bit Free-Run Timer	
16-bit Free-Run Timer Interrupt	256
16-bit Free-Run Timer Register	213
Notes on Using 16-bit Free-Run Timer	292
Program Example of 16-bit Free-Run Timer	294
16-bit Input Capture	
16-bit Input Capture Interrupt	258
16-bit Input Capture Registers	215
Cautions for Use of 16-bit Input Capture	292
Input Timing of 16-bit Input Capture	278
Operation of 16-bit Input Capture	277
16-bit Output Compare	
16-bit Output Compare and Free-run Timer Operation	
.....	273
16-bit Output Compare Interrupt	257
16-bit Output Compare Register	214
16-bit Output Compare Timing	272
Notes on Using 16-bit Output Compare	292
Operation of 16-bit Output Compare	
(Inverted Mode, MOD1x=0)	267
Operation of 16-bit Output Compare	
(Set/Reset Mode, MOD15 to MOD10=1)	
.....	271
Program Example of 16-bit Output Compare	295
16-bit Reload Register	
TMRLR Register (16-bit Reload Register)	157
16-bit Timer Register	
TMR Register (16-bit Timer Register)	157
8/10-bit A/D Converter	
Block Diagram of 8/10-bit A/D Converter	325
Block Diagram of 8/10-bit A/D Converter Pin	
.....	110, 329
Function of 8/10-bit A/D Converter	324
Interrupt of 8/10-bit A/D Converter	341
Pins of 8/10-bit A/D Converter	328
Precautions on Using 8/10-bit A/D Converter	347
Register List of 8/10-bit A/D Converter	330

A

A/D	
A/D Startup	291
A/D Startup Enable	291
A/D Activation	
A/D Activation Via Free-run Timer	266
A/D Activation Compare Registers	
A/D Activation Compare Registers	217
A/D Channel Control Register	
A/D Channel Control Register(ADCH: ADCH0 to ADCH2)	
.....	331
A/D Control Status Register	
A/D Control Status Register(ADCS: ADCS0 to ADCS2)	
.....	336
A/D Conversion Data	
A/D Conversion Data Protection Function	346
A/D Converter	
Block Diagram of 8/10-bit A/D Converter	325
Block Diagram of 8/10-bit A/D Converter Pin	
.....	110, 329
Function of 8/10-bit A/D Converter	324
Interrupt of 8/10-bit A/D Converter	341
Pins of 8/10-bit A/D Converter	328
Precautions on Using 8/10-bit A/D Converter	347
Register List of 8/10-bit A/D Converter	330
A/D Data Register	
A/D Data Register	
(ADCD: ADCD00, ADCD01, ADCD10, ADCD11,	
ADCD20, ADCD21)	339
A/D Mode Setting Register	
A/D Mode Setting Register(ADMD: ADMD0 to ADMD2)	
.....	333
A/D Trigger Control Register	
A/D Trigger Control Register (ADTRGC)	226
Absence Trap	
Coprocessor Absence Trap	60
ADCD	
A/D Data Register	
(ADCD: ADCD00, ADCD01, ADCD10, ADCD11,	
ADCD20, ADCD21)	339
ADCH	
A/D Channel Control Register(ADCH: ADCH0 to ADCH2)	
.....	331
ADCOMP	
Compare Register 0,1,2 (ADCOMP0, ADCOMP1,	
ADCOMP2)	254
ADCOMPC	
Control Register (ADCOMPC)	255
ADCS	
A/D Control Status Register	
(ADCS: ADCS0 to ADCS2)	336

Addressing	
Addressing Mode	391
Addressing Area	
Direct Addressing Area	26, 44
ADMD	
A/D Mode Setting Register(ADMD: ADMD0 to ADMD2)	
.....	333
ADTRGC	
A/D Trigger Control Register (ADTRGC)	226
AF200 Flash	
System Configuration of AF200 Flash Microcontroller	
Programmer (Yokogawa Digital Computer	
Corporation)	435
AICR	
Analog Input Control Register	
(AICR: AICR0 to AICR2)	340
Analog Input Control Register (AICR:AICR0 to AICR2)	
.....	111
Analog Input Control Register	
Analog Input Control Register	
(AICR: AICR0 to AICR2)	340
Analog Input Control Register (AICR:AICR0 to AICR2)	
.....	111
Architecture	
Internal Architecture.....	28
Assembler	
Assembler (fasm911)	456
Asynchronous	
Asynchronous (Start-stop Synchronization) Mode	
.....	314
Watchdog Timer Control Register	
RSRR: Reset Source Register/watchdog Timer Control	
Register.....	77
Automatic Algorithm	
Automatic Algorithm Execution Status	404
B	
Base Clock Divide Ratio Setting Register	
DIVR0: Base Clock Divide Ratio Setting Register 0	
.....	85
DIVR1: Base Clock Divide Ratio Setting Register 1	
.....	88
Basic Block Diagram	
Basic Block Diagram of I/O Ports.....	102
Basic Configuration	
Basic Configuration of the Serial Programming Connection	
.....	432
Basic Programming	
Basic Programming Model.....	33
Baud Rate	
Calculation of Baud Rate	302
Bit Ordering	
Bit Ordering.....	40
Block Diagram	
Basic Block Diagram of I/O Ports.....	102
Block Diagram.....	4, 117, 131, 142, 145, 168,
298, 306, 352, 370	
Block Diagram of 8/10-bit A/D Converter.....	325
Block Diagram of 8/10-bit A/D Converter Pin	
.....	110, 329
Block Diagram of Flash Memory	405
Block Diagram of Multifunctional Timer	206
PWC Block Diagram	185
Reload Timer Block Diagram.....	153
Block Size	
Block Size	390
Block Transfer	
Block Transfer	398
Branch Operation	
Branch Operation with Delay Slot.....	45
Branch Operation without Delay Slot	47
Branching Command	
JMP Instruction (Branching Command).....	366
BSD	
0 Detection Data Register (BSD0)	146
1 Detection Data Register (BSD1)	146
BSDC	
Change Point Detection Data Register (BSDC)	147
BSRR	
Detection Result Register (BSRR)	147
Burst Transfer	
Burst Transfer	399
Burst Two-cycle Transfer	
Burst Two-cycle Transfer	388
Bus Converter	
32-bit ↔ 16-bit Bus Converter	30
Harvard ↔ Princeton Bus Converter	30
Bus Modes	
Bus Modes	62
BUSY	
Ready/Busy Signal (RDY/ $\overline{\text{BUSY}}$).....	420
Busy Signal	
Ready/Busy Signal (RDY/ $\overline{\text{BUSY}}$).....	420
Byte Ordering	
Byte Ordering	40
C	
C Compiler	
C Compiler (fcc911)	453
Calculating	
Calculating Function.....	360
Cancel	
Temporary Sector Protect Cancel.....	429
Cancel Request	
Hold Request Cancel Request	124
Cascade Mode	
Cascade Mode.....	302
Change Point	
Change Point Detection	149

INDEX

Change Point Detection Data Register	
Change Point Detection Data Register (BSDC)	147
Channel	
Channel Selection and Control	397
CLK	
CLK Synchronous Mode	315
CLKB	
CPU Clock Signal (CLKB).....	74
CLKP	
Peripheral Clock Signal (CLKP)	74
CLKR	
CLKR: Clock Source Control Register	83
CLKT	
External Bus Clock Signal (CLKT).....	74
Clock	
Count Clock Selection	179, 193
CPU Clock Signal (CLKB).....	74
External Bus Clock Signal (CLKT).....	74
Internal Clock Operation	158
Oscillation Clock Frequency	435
Peripheral Clock Signal (CLKP)	74
Selected External Count Clock	266
Selecting a Clock for the UART	313
Selecting the Source Clock Signal.....	70
Clock Source Control Register	
CLKR: Clock Source Control Register	83
Combinations	
PPG Combinations.....	181
Common Items	
Common Items.....	475
Compare Clear Buffer	
Compare Clear Buffer.....	262
Compare Clear Register	
Compare Clear Register (CPCLRH,CPCLRL)	219
Compare Control Register	
Compare Control Register,Lower Byte	
(OCSL0,OCSL2,OCSL4)	232
Compare Control Register,Upper Byte	
(OCSH1,OCSH3,OCSH5).....	229
Compare Mode Control Register	
Compare Mode Control Register (OCMOD)	234
Compare Register	
Compare Register 0,1,2	
(ADCOMP0,ADCOMP1,ADCOMP2)	
.....	254
Compare-clear Buffer Register	
Compare-clear Buffer Register	
(CPCLRBH,CPCLRBL).....	218
Continuous Conversion	
Operation of Continuous Conversion Mode.....	343
Control Register	
Control Register (ADCOMPC)	255
Control Status Register	
Control Status Register	
(TMCSR:TMCSR0 to TMCSR2).....	154
Control/Status Registers	
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Control/Status Registers A	
.....	371
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Control/Status Registers B	
.....	375
DSP-CSR (Control/Status Registers).....	355
Conversion Mode	
Operation of Continuous Conversion Mode	343
Operation of Single Conversion Mode	342
Operation of Pause-conversion Mode.....	345
Coprocessor	
Coprocessor Absence Trap	60
Coprocessor Error Trap.....	61
Count	
Count Clock Selection.....	179, 193
Counter	
Counter Clear	195
Operating States of the Counter	161
CPCLRBH	
Compare-clear Buffer Register	
(CPCLRBH,CPCLRBL)	218
CPCLRBL	
Compare-clear Buffer Register	
(CPCLRBH,CPCLRBL)	218
CPCLRH	
Compare Clear Register (CPCLRH,CPCLRL).....	219
CPCLRL	
Compare Clear Register (CPCLRH,CPCLRL).....	219
CPU	
CPU	30
CPU Clock Signal (CLKB).....	74
CPU Control.....	392
FR-CPU Programming Mode (16-bit,Read/write)	
.....	414
FR-CPU ROM Mode (32/16/8-bit, Read Only).....	414
CTBR	
CTBR: Timebase Counter Clear Register.....	83
D	
Data Access	
Data Access.....	41
Data Direction Registers	
Data Direction Registers (DDR:DDR0 to DDR7, DDRC,	
DDRD, DDRE and DDRG)	105
Data Operation	
Data Operation at Two-cycle Transfer.....	400
Data Type	
Data Type.....	392
DDR	
Data Direction Registers (DDR:DDR0 to DDR7, DDRC,	
DDRD,DDRE and DDRG).....	105
Dead Time Timer Mode	
Operation During Dead Time Timer Mode	285
Debugger	
Debugger (sim911,eml911,mon911)	459
Notes on Using Debugger	477

Delay Slot	
Branch Operation with Delay Slot	45
Branch Operation without Delay Slot	47
Delayed Interrupt Control Register	
DICR (Delayed Interrupt Control Register)	143
Delayed Register	
DSP-LY (Delayed Register)	358
Delayed Write Feature	
Delayed Write Feature	360
Detection	
0 Detection	148
1 Detection	148
Change Point Detection	149
Detection Data Register	
0 Detection Data Register (BSD0)	146
1 Detection Data Register (BSD1)	146
Detection Result Register	
Detection Result Register (BSRR)	147
Device States	
Device States and State Transitions	93
DICR	
DICR (Delayed Interrupt Control Register)	143
DLYI Bit of DICR	144
Direct Addressing	
Direct Addressing Area	26, 44
DIVR	
DIVR0: Base Clock Divide Ratio Setting Register 0	85
DIVR1: Base Clock Divide Ratio Setting Register 1	88
DLYI Bit	
DLYI Bit of DICR	144
DMA	
DMA Transfer in Sleep Mode	396
Peripheral Interrupt Clear by DMA	394
DMAC	
DMAC Interrupt Control	396
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Control/Status Registers A	371
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Control/Status Registers B	375
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Overall DMAC Control Register	383
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Transfer Source/Transfer Destination Address Setting Registers	381
DSP-CSR	
DSP-CSR (Control/Status Registers)	355
DSP-LY	
DSP-LY (Delayed Register)	358
DSP-OT	
DSP-OT0 to DSP-OT7 (Variable Monitor Register)	358
DSP-PC	
DSP-PC (Program Counter)	357
DTCR	
16-bit Dead Timer Control Register (DTCR0)	245
16-bit Dead Timer Control Register (DTCR1)	247
16-bit Dead Timer Control Register (DTCR2)	249
DTTI	
DTTI Interrupts	290
DTTI Operation of Waveform Control Register 2 (SIGCR2)	290
DTTI Pin Input Operation	289
DTTI Pin Noise Cancel Feature	290
E	
EIRR	
External Interrupt Source Register (EIRR (EIRR0,EIRR1): External Interrupt Request Register)	132
EIT	
EIT (Exception,Interrupt,and Trap)	48
EIT Sources	48
EIT Vector Table	53
Reception Priority of EIT Source	56
Return from EIT	48
ELVR	
External Interrupt Request Level Setting Register (ELVR (ELVR0,ELVR1): External LeVel Register)	133
eml911	
Debugger (sim911,eml911,mon911)	459
ENable Interrupt Request Register	
Interrupt Enable Register (ENIR (ENIR0,ENIR1): ENable Interrupt Request Register)	132
End	
Operation End/Stopping	395
ENIR	
Interrupt Enable Register (ENIR (ENIR0,ENIR1): ENable Interrupt Request Register)	132
Error Trap	
Coprocessor Error Trap	61
Exception	
EIT (Exception,Interrupt,and Trap)	48
Processing of Undefined Instruction Exception	60
External Bus Clock	
External Bus Clock Signal (CLKT)	74
External Count Clock	
Selected External Count Clock	266
External Interrupt	
External Interrupt Request Level	135
External Interrupt Source Register (EIRR (EIRR0,EIRR1): External Interrupt Request Register)	132
Operating Procedure for an External Interrupt	134
Operation of an External Interrupt	134
Precautions when Returning from STOP State Using External Interrupt	137
External Interrupt Request Level Setting Register	
External Interrupt Request Level Setting Register (ELVR (ELVR0,ELVR1): External LeVel Register)	133
External Interrupt Source Register	
External Interrupt Source Register (EIRR (EIRR0,EIRR1): External Interrupt Request Register)	132

INDEX

F	
fasm911	
Assembler (fasm911).....	456
fcc911	
C Compiler (fcc911)	453
Features	
Features.....	2, 28, 48, 350
Flag	
Example of Hardware Sequence Flag Usage.....	424
Hardware Sequence Flag	420
I-flag.....	50
Occurrence of Interrupts and Timing for Setting Flags	316
Flash Memory	
Block Diagram of Flash Memory	405
Overview of Flash Memory	404
Register List of Flash Memory	409
Sector Configuration of Flash Memory	406
Flash Memory Status Register	
Configuration of Flash Memory Status Register (FLCR)	410
Flash Wait Register	
Configuration of the Flash Wait Register (FLWC)	412
FLCR	
Configuration of Flash Memory Status Register(FLCR)	410
flnk911	
Linker (flnk911).....	458
FLWC	
Configuration of the Flash Wait Register (FLWC)	412
FR-CPU Programming Mode	
FR-CPU Programming Mode (16-bit,Read/write)	414
FR-CPU ROM Mode	
FR-CPU ROM Mode (32/16/8-bit, Read Only)	414
Free-Run Timer	
16-bit Free-Run Timer Interrupt.....	256
16-bit Free-Run Timer Register.....	213
Notes on Using 16-bit Free-Run Timer	292
Program Example of 16-bit Free-Run Timer.....	294
Free-run Timer	
16-bit Output Compare and Free-run Timer Operation	273
A/D Activation Via Free-run Timer	266
Fujitsu Standard	
Pins Used for Fujitsu Standard Serial Onboard Writing	433
G	
GATE	
GATE Function.....	180
Output Status of RTO0 to RTO5 and GATE.....	279
GATE Function Control Register	
GATEC Register (GATE Function Control Register)	176
Gate Trigger	
PPG0 Output Via Gate Trigger	281
GATEC Register	
GATEC Register (GATE Function Control Register)	176
H	
Handling Devices	
Handling Devices	22
Hardware	
Hardware Configuration.....	114, 368
Initial Value of Each Hardware.....	180
Hardware Sequence Flag	
Example of Hardware Sequence Flag Usage.....	424
Hardware Sequence Flag.....	420
Harvard	
Harvard ↔ Princeton Bus Converter	30
Hold Request	
Hold Request Cancel Request.....	124
Hold Request Cancel Level	
Hold Request Cancel Level Register (HRCL).....	119
Hold Request Cancel Register	
Example of Using the Hold Request Cancel Register (HRCL)	125
HRCL	
Example of Using the Hold Request Cancel Register (HRCL)	125
Hold Request Cancel Level Register (HRCL)	119
I	
I/O Map	
I/O Map	438
I/O Ports	
Basic Block Diagram of I/O Ports	102
Mode for I/O Ports	102
ICR	
ICR Bit Configuration.....	51
Interrupt Control Register (ICR:ICR00 to ICR47)	118
ICR Mapping	
ICR Mapping.....	51
ICSH	
Input Capture State Control Register (ch.2, ch.3), Upper Byte (ICSH23)	237
ICSL	
Input Capture State Control Register (ch.2, ch.3), Lower Byte (ICSL23)	239
I-flag	
I-flag.....	50
ILM	
ILM Register	50

Initial Area	
Vector Table Initial Area.....	44
Initial Value	
Initial Value of Each Hardware.....	180
Initialization	
Wait Times after Setting Initialization.....	72
Input Capture	
16-bit Input Capture Interrupt.....	258
16-bit Input Capture Registers.....	215
Cautions for Use of 16-bit Input Capture.....	292
Input Timing of 16-bit Input Capture.....	278
Operation of 16-bit Input Capture.....	277
Input Capture Data Register	
Input Capture Data Register (IPCPH: IPCPH0 to IPCPL3,IPCPL: IPCPL0 to IPCPL3).....	236
Input Capture State Control Register	
Input Capture State Control Register (ch.2,ch.3),Upper Byte (ICSH23).....	237
Input Capture State Control Register (ch.2,ch.3),Lower Byte (ICSL23).....	239
Input/Output Circuit	
Input/Output Circuit Types.....	18
Instruction	
Instruction Definitions.....	354
Instruction Overview.....	31
JMP Instruction (Branching Command).....	366
MAC Instruction.....	363
Processing of RETI Instruction.....	61
STR Instruction (Transition Instruction).....	364
Instruction Operation	
Instruction Operation.....	359
INT	
Processing of INT Instruction.....	59
INTE	
Processing of INTE Instruction.....	59
Internal Architecture	
Internal Architecture.....	28
Internal Clock	
Internal Clock Operation.....	158
Internal Peripheral Request	
Internal Peripheral Request.....	387
Interrupt	
16-bit Free-Run Timer Interrupt.....	256
16-bit Input Capture Interrupt.....	258
16-bit Output Compare Interrupt.....	257
DMAC Interrupt Control.....	396
DTTI Interrupts.....	290
EIT (Exception,Interrupt,and Trap).....	48
External Interrupt Request Level.....	135
External Interrupt Source Register (EIRR (EIRR0,EIRR1): External Interrupt Request Register).....	132
Interrupt Levels.....	49
Interrupt Number.....	144
Interrupt of 8/10-bit A/D Converter.....	341
Interrupt Stack.....	52
Interrupt/NMI Level Masking.....	50
NMI (Non Maskable Interrupt).....	124
Occurrence of Interrupts and Timing for Setting Flags	316
Operating Procedure for an External Interrupt.....	134
Operation of an External Interrupt.....	134
Operation of User Interrupt/NMI.....	58
Peripheral Interrupt Clear by DMA.....	394
Precautions when Returning from STOP State Using External Interrupt.....	137
Timer Interrupt.....	264
Waveform Generator Interrupts.....	259
Interrupt Control Register	
Interrupt Control Register (ICR:ICR00 to ICR47)	118
Interrupt Enable Register	
Interrupt Enable Register (ENIR (ENIR0,ENIR1): ENable Interrupt Request Register).....	132
Interrupt Mask	
Interrupt Mask Function.....	264
Interruption	
Interruption.....	180
Inverted Mode	
Operation of 16-bit Output Compare (Inverted Mode,MOD1x=0).....	267
IPCPH	
Input Capture Data Register (IPCPH: IPCPH0 to IPCPL3,IPCPL: IPCPL0 to IPCPL3).....	236
IPCPL	
Input Capture Data Register (IPCPH: IPCPH0 to IPCPL3,IPCPL: IPCPL0 to IPCPL3).....	236
J	
JMP Instruction	
JMP Instruction (Branching Command).....	366
L	
Linker	
Linker (flnk911).....	458
Low-power Consumption	
Low-power Consumption Modes.....	96
M	
MAC Instruction	
MAC Instruction.....	363
Main Function	
Main Function.....	368
Major Functions	
Major Functions.....	114
Masking	
Interrupt/NMI Level Masking.....	50
Measurement	
Pulse Width Measurement Function.....	184
Measuring	
Details for Pulse Width Measuring Operation.....	195
Memory Map	
Memory Map.....	26, 42

INDEX

Microcontroller Programmer

System Configuration of AF200 Flash Microcontroller Programmer (Yokogawa Digital Computer Corporation)	435
--	-----

MOD

Operation of 16-bit Output Compare (Inverted Mode,MOD1x=0)	267
Operation of 16-bit Output Compare (Set/Reset Mode,MOD15 to MOD10=1)	271

Mode

Asynchronous (Start-stop Synchronization) Mode	314
Mode Setting	62
Returning from Standby Mode (Stop or Sleep Mode)	125

mon911

Debugger (sim911,eml911,mon911)	459
---------------------------------------	-----

Monitor Output

Variable Monitor Output	361
-------------------------------	-----

Multifunctional Timer

Block Diagram of Multifunctional Timer	206
Configuration of Multifunctional Timer	204
Operation of Multifunctional Timer	260
Pins of Multifunctional Timer	212

Multiplier

PLL Multiplier	71
Wait Times after the PLL Multiplier is Changed	72

N

NMI

NMI	136
NMI (Non Maskable Interrupt)	124
Operation of User Interrupt/NMI	58

NMI Level

Interrupt/NMI Level Masking	50
-----------------------------------	----

Noise Cancel

DTTI Pin Noise Cancel Feature	290
-------------------------------------	-----

Non Maskable Interrupt

NMI (Non Maskable Interrupt)	124
------------------------------------	-----

Note

Note	201
Notes	162
Notes on Setting Register	371

O

OCCPBH

Output Compare Buffer Register (OCCPBH: OCCPBH0 to OCCPBH5,OCCPBL: OCCPBL0 to OCCPBL5)	227
--	-----

OCCPBL

Output Compare Buffer Register (OCCPBH: OCCPBH0 to OCCPBH5,OCCPBL: OCCPBL0 to OCCPBL5)	227
--	-----

OCCPH

Output Compare Register (OCCPH: OCCPH0 to OCCPH5,OCCPL: OCCPL0 to OCCPL5)	228
---	-----

OCCPL

Output Compare Register (OCCPH: OCCPH0 to OCCPH5,OCCPL: OCCPL0 to OCCPL5)	228
---	-----

OCMOD

Compare Mode Control Register (OCMOD)	234
---	-----

OCSH

Compare Control Register,Upper Byte (OCSH1,OCSH3,OCSH5)	229
--	-----

OCSL

Compare Control Register,Lower Byte (OCSL0,OCSL2,OCSL4)	232
--	-----

Operating Mode

Operating Mode	177, 359
Operating Modes	313

Operating States

Operating States of the Counter	161
---------------------------------------	-----

Operation

Overview of Operation	177, 385
-----------------------------	----------

Operation Mode

Operation Modes	62
Selects Operation Mode	193

Operation Start

Operation Start	393
-----------------------	-----

Ordering

Bit Ordering	40
Byte Ordering	40

Oscillation Clock

Oscillation Clock Frequency	435
-----------------------------------	-----

Oscillation Stabilization Wait Time

Oscillation Stabilization Wait Time	67
---	----

Output Compare

16-bit Output Compare and Free-run Timer Operation	273
16-bit Output Compare Interrupt	257
16-bit Output Compare Register	214
16-bit Output Compare Timing	272
Notes on Using 16-bit Output Compare	292
Operation of 16-bit Output Compare (Inverted Mode,MOD1x=0)	267
Operation of 16-bit Output Compare (Set/Reset Mode,MOD15 to MOD10=1)	271
Program Example of 16-bit Output Compare	295

Output Compare Buffer Register

Output Compare Buffer Register (OCCPBH: OCCPBH0 to OCCPBH5,OCCPBL: OCCPBL0 to OCCPBL5)	227
--	-----

Output Compare Register

Output Compare Register (OCCPH: OCCPH0 to OCCPH5,OCCPL: OCCPL0 to OCCPL5)	228
---	-----

Output Invert Register	
REVC Register (Output Invert Register)	175
Output Status	
Output Status of RTO0 to RTO5 and GATE	279
Output Terminal	
Output Terminal Functions	160
Overall DMAC Control Register	
DMAC-ch.0,ch.1,ch.2,ch.3,ch.4 Overall DMAC Control Register.....	383
P	
Package Dimension	
Package Dimension	5
Pause-conversion Mode	
Operation of Pause-conversion Mode	345
PCR	
Pull-up Control Registers	
(PCR:PCR0 to PCR7 and PCRG)	106
PDIVR	
PDIVR (Ratio of Dividing Frequency Control Register)	
.....	191
PDR	
Port Data Registers (PDR:PDR0 to	
PDR7,PDRC,PDRD,PDRE and PDRG)	
.....	104
Peripheral Clock	
Peripheral Clock Signal (CLKP)	74
Peripheral Interrupt	
Peripheral Interrupt Clear by DMA.....	394
Peripheral Request	
Internal Peripheral Request	387
PFR	
Port Function Registers	
(PFR:PFR0 to PFR2,PFR7 and PFRG).....	107
PICSH	
PPG Output Control Register,Upper Byte (PICSH01)	
.....	241
PICSL	
PPG Control Register,Lower Byte (PICSL01).....	242
Pin Assignment	
Pin Assignment.....	7
Pin Description	
Pin Description	9
PLL	
Enabling/disabling PLL Operation.....	71
PLL Multiplier	71
Wait Times after PLL Operation is Enabled	72
Wait Times after the PLL Multiplier is Changed.....	72
Port Data Registers	
Port Data Registers (PDR:PDR0 to PDR7, PDRC, PDRD,	
PDRE and PDRG)	104
Port Function Registers	
Port Function Registers	
(PFR:PFR0 to PFR2,PFR7 and PFRG).....	107
Power	
Wait Times after the Power is Turned on.....	72
PPG	
Function of PPG	164
PPG Combinations	181
PPG Output Operation.....	178
PPG0 Output Control.....	281
PPG0 Output Via Gate Trigger	281
PPG Activation Register	
TRG Register (PPG Activation Register).....	175
PPG Control Register	
PPG Control Register,Lower Byte (PICSL01)	242
PPG Output Control Register	
PPG Output Control Register,Upper Byte (PICSH01)	
.....	241
PPGCn Register	
PPGCn Register (PPGn Operation Mode Control Register)	
.....	172
PPGn Operation Mode Control Register	
PPGCn Register (PPGn Operation Mode Control Register)	
.....	172
Precaution	
Other Precautionary Information	435
Precautions on Usage.....	318
Precautions when Returning from STOP State Using	
External Interrupt.....	137
Princeton	
Harvard ↔ Princeton Bus Converter	30
Priority	
Reception Priority of EIT Source	56
Priority Evaluation	
Priority Evaluation.....	120
PRL/PRLL Register	
PRL/PRLL Register (Reload Register)	174
Processing	
Save/Restore Processing	150
Program Access	
Program Access	41
Program Counter	
DSP-PC (Program Counter)	357
Program Example	
Program Example of 16-bit Free-Run Timer	294
Program Example of 16-bit Output Compare.....	295
Programming	
FR-CPU Programming Mode (16-bit,Read/write)	
.....	414
Programming Model	
Basic Programming Model	33
Protect	
Enable Sector Protect.....	427
Sector Protect Operation List	426
Verify Sector Protect	427
Protection Function	
A/D Conversion Data Protection Function.....	346
Pull-up Control Registers	
Pull-up Control Registers (PCR:PCR0 to PCR7 and PCRG)	
.....	106

INDEX

Pulse	
Control of Pulse Pin Output.....	179
Pulse Width	
Details for Pulse Width Measuring Operation.....	195
Pulse Width Measurement Function.....	184
Relation between Reload Value and Pulse Width	178
Starting and Stopping of Pulse Width Count.....	194
PWC	
PWC Block Diagram	185
PWC Control/Status Register	
PWCSR Register (PWC Control/Status Register).....	186
PWC Data Buffer Register	
PWCR0,PWCR1 Register (PWC Data Buffer Register)	
.....	190
PWCR	
PWCR0,PWCR1 Register (PWC Data Buffer Register)	
.....	190
PWCR1 Register	
PWCR0,PWCR1 Register	
(PWC Data Buffer Register).....	190
PWCSR Register	
PWCSR Register (PWC Control/Status Register).....	186
R	
Ratio of Dividing Frequency Control Register	
PDIVR (Ratio of Dividing Frequency Control Register)	
.....	191
RDY	
Ready/Busy Signal (RDY/ $\overline{\text{BUSY}}$).....	420
Ready	
Ready/Busy Signal (RDY/ $\overline{\text{BUSY}}$).....	420
Reception Priority	
Reception Priority of EIT Source	56
Register	
Notes on Setting Register	371
Register Description	369
Registers.....	34
Register List	
Register List	115, 130, 142, 145, 152, 164, 184, 298, 305, 351
Reload Operation	
Reload Operation	390
Reload Register	
PRL/PRLH Register (Reload Register).....	174
Reload Register (UTIMR: UTIMR0 to UTIMR2)	
.....	299
TMRLR Register (16-bit Reload Register)	157
Reload Timer	
Reload Timer Block Diagram.....	153
Reload Value	
Relation between Reload Value and Pulse Width	178
Request	
Internal Peripheral Request.....	387
Software Request	387
Transfer Request Acceptance and Transfer	394
Request Level	
External Interrupt Request Level	135
Reset	
Reset Operation Modes	69
Reset Sequence.....	66
Reset Sources	65
Reset Levels	
Reset Levels	64
Reset Source Register	
RSRR: Reset Source Register/watchdog Timer Control	
Register	77
Restore Processing	
Save/Restore Processing	150
RETI Instruction	
Processing of RETI Instruction	61
Return	
Return from EIT	48
Return from Standby	134
Return Operation	
Return Operation from STOP State	138
REVC Register	
REVC Register (Output Invert Register)	175
ROM	
FR-CPU ROM Mode (32/16/8-bit, Read Only).....	414
ROM Writer	
Writing by a ROM Writer	405
RSRR	
RSRR: Reset Source Register/watchdog Timer Control	
Register	77
RTO	
Output Status of RTO0 to RTO5 and GATE	279
S	
Save	
Save/Restore Processing	150
SCR	
SCR (Serial Control Register).....	308
Sector	
Enable Sector Protect	427
Sector Protect Operation List.....	426
Verify Sector Protect.....	427
Sector Configuration	
Sector Configuration of Flash Memory	406
Sector Protect	
Enable Sector Protect	427
Sector Protect Operation List.....	426
Temporary Sector Protect Cancel	429
Verify Sector Protect.....	427
Serial Control Register	
SCR (Serial Control Register).....	308
Serial Input Data Register	
SIDR: SIDR0 to SIDR2 (Serial Input Data Register)	
SODR: SODR0 to SODR2	
(Serial Output Data Register).....	310
Serial Mode Register	
SMR (Serial Mode Register).....	307

Serial Onboard Writing	
Pins Used for Fujitsu Standard Serial Onboard Writing	433
Serial Output Data Register	
SIDR: SIDR0 to SIDR2 (Serial Input Data Register)	
SODR: SODR0 to SODR2	
(Serial Output Data Register).....	310
Serial Programming Connection	
Basic Configuration of the Serial Programming Connection	432
Example of Serial Programming Connection.....	434
Serial Status Register	
SSR: SSR0 to SSR2 (Serial Status Register)	310
Set/Reset Mode	
Operation of 16-bit Output Compare	
(Set/Reset Mode,MOD15 to MOD10=1)	
.....	271
Setting	
Notes on Setting Register	371
SIDR	
SIDR: SIDR0 to SIDR2 (Serial Input Data Register)	
SODR: SODR0 to SODR2	
(Serial Output Data Register).....	310
SIGCR	
DTTI Operation of Waveform Control Register 2	
(SIGCR2)	290
Waveform Control Register 1 (SIGCR1)	251
Waveform Control Register 2 (SIGCR2)	253
Signal	
Ready/Busy Signal (RDY/BUSY)	420
sim911	
Debugger (sim911,eml911,mon911).....	459
Single Conversion	
Operation of Single Conversion Mode.....	342
Sleep Mode	
DMA Transfer in Sleep Mode.....	396
Returning from Standby Mode (Stop or Sleep Mode)	
.....	125
Slot	
Branch Operation with Delay Slot	45
Branch Operation without Delay Slot	47
SMR	
SMR (Serial Mode Register)	307
SODR	
SIDR: SIDR0 to SIDR2 (Serial Input Data Register)	
SODR: SODR0 to SODR2	
(Serial Output Data Register).....	310
Software Request	
Software Request.....	387
Source Clock	
Selecting the Source Clock Signal	70
SSR	
SSR: SSR0 to SSR2 (Serial Status Register)	310
Stack	
Interrupt Stack	52
Standby	
Return from Standby	134
Standby Control Register	
STCR: Standby Control Register	78
Standby Mode	
Returning from Standby Mode (Stop or Sleep Mode)	
.....	125
Start-stop Synchronization	
Asynchronous (Start-stop Synchronization) Mode	
.....	314
State Transitions	
Device States and State Transitions.....	93
STCR	
STCR: Standby Control Register	78
Step	
Processing of Step Trace Trap.....	59
Step/block Transfer	
Step/block Transfer Two-cycle Transfer	389
STOP	
Precautions when Returning from STOP State Using	
External Interrupt.....	137
Return Operation from STOP State.....	138
Stop	
Returning from Standby Mode (Stop or Sleep Mode)	
.....	125
Stop Mode	
Wait Times after Returning from Stop Mode.....	72
Stopping	
Operation End/Stopping	395
Stopping due to Error	395
STR Instruction	
STR Instruction (Transition Instruction)	364
Synchronous Mode	
CLK Synchronous Mode	315
System Configuration	
System Configuration of AF200 Flash Microcontroller	
Programmer (Yokogawa Digital Computer	
Corporation).....	435
T	
Table Base Register	
TBR (Table Base Register).....	53
TBCR	
TBCR: Timebase Counter Control Register.....	81
TBR	
TBR (Table Base Register).....	53
TCCSH	
Timer State Control Register,Upper Byte (TCCSH)	
.....	221
TCCSL	
Timer State Control Register,Lower Byte (TCCSL).....	224
TCDTH	
Timer Data Register (TCDTH,TCDTL).....	220
TCDTL	
Timer Data Register (TCDTH,TCDTL).....	220
Temporary Sector Protect Cancel	
Temporary Sector Protect Cancel.....	429

INDEX

Temporary Stop	
Temporary Stop	394
Timebase Counter	
Timebase Counter	90
Timebase Counter Clear Register	
CTBR: Timebase Counter Clear Register	83
Timebase Counter Control Register	
TBCR: Timebase Counter Control Register	81
Timer Clear	
Timer Clear	261
Timer Control Register	
Timer State Control Register, Lower Byte (TCCSL)	224
Timer Data Register	
Timer Data Register (TCDTH, TCDTL)	220
Timer Interrupt	
Timer Interrupt	264
Timer Mode	
Operation of Timer Mode	283
Timer Mode	261
Timer Register	
TMR Register (16-bit Timer Register)	157
Timer State Control Register	
Timer State Control Register, Upper Byte (TCCSH)	221
TMCSR	
Control Status Register (TMCSR: TMCSR0 to TMCSR2)	154
TMR Register	
TMR Register (16-bit Timer Register)	157
TMRLR Register	
TMRLR Register (16-bit Reload Register)	157
TMRRH	
16-bit Dead Timer Register (TMRRH: TMRRH0 to TMRRH2, TMRRL: TMRRL0 to TMRRL2)	244
TMRRL	
16-bit Dead Timer Register (TMRRH: TMRRH0 to TMRRL2, TMRRL: TMRRL0 to TMRRL2)	244
Trace	
Processing of Step Trace Trap	59
Transfer	
Block Transfer	398
Burst Transfer	399
Burst Two-cycle Transfer	388
Data Operation at Two-cycle Transfer	400
DMA Transfer in Sleep Mode	396
Step/block Transfer Two-cycle Transfer	389
Transfer of Operation Results	360
Transfer Request Acceptance and Transfer	394
Transfer Sequence Selection	388
Transfer Source/Transfer Destination Address Setting Registers	
DMAC-ch.0, ch.1, ch.2, ch.3, ch.4 Transfer Source/Transfer Destination Address Setting Registers	381
Transfers	
Control of Number of Transfers	392

Transition Instruction	
STR Instruction (Transition Instruction)	364
Trap	
Coprocessor Absence Trap	60
Coprocessor Error Trap	61
EIT (Exception, Interrupt, and Trap)	48
Processing of Step Trace Trap	59
TRG Register	
TRG Register (PPG Activation Register)	175
Trigger	
PPG0 Output Via Gate Trigger	281
Two-cycle Transfer	
Burst Two-cycle Transfer	388
Data Operation at Two-cycle Transfer	400
Step/block Transfer Two-cycle Transfer	389

U

UART	
Example of Using the UART	319
Features of UART	304
Selecting a Clock for the UART	313
Undefined Instruction	
Processing of Undefined Instruction Exception	60
Underflow Operation	
Underflow Operation	159
UTIM	
U-TIMER (UTIM: UTIM0 to UTIM2)	299
UTIMC	
U-TIMER Control Register (UTIMC: UTIMC0 to UTIMC2)	299
U-TIMER	
U-TIMER (UTIM: UTIM0 to UTIM2)	299
U-TIMER Control Register	
U-TIMER Control Register (UTIMC: UTIMC0 to UTIMC2)	299
UTIMR	
Reload Register (UTIMR: UTIMR0 to UTIMR2)	299

V

Variable Monitor Register	
DSP-OT0 to DSP-OT7 (Variable Monitor Register)	358
Vector Table	
EIT Vector Table	53
Vector Table Initial Area	44

W

Wait Times	
Wait Times after PLL Operation is Enabled	72
Wait Times after Returning from Stop Mode	72
Wait Times after Setting Initialization	72
Wait Times after the PLL Multiplier is Changed	72
Wait Times after the Power is Turned on	72

Waveform Control Register	
DTTI Operation of Waveform Control Register 2	
(SIGCR2)	290
Waveform Control Register 1 (SIGCR1)	251
Waveform Control Register 2 (SIGCR2)	253
Waveform Generator	
Notes on Using Waveform Generator	293
Waveform Generator Interrupts	259
Waveform Generator Registers	216
Writer	
Writing by a ROM Writer	405

CM71-10127-2E

Fujitsu Semiconductor Device. CONTROLLER MANUAL
FR60Lite
32-BIT MICROCONTROLLER
MB91260B Series
HARDWARE MANUAL

August 2006 the second edition

Published **FUJITSU LIMITED** Electronic Devices

Edited Business Promotion Dept.
