



---

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.

**F<sup>2</sup>MC-16LX**  
**16-BIT MICROCONTROLLER**  
**MB90820B Series**  
**HARDWARE MANUAL**



# **F<sup>2</sup>MC-16LX**

## **16-BIT MICROCONTROLLER**

# **MB90820B Series**

# **HARDWARE MANUAL**

The information for microcontroller supports is shown in the following homepage.  
Be sure to refer to the "Check Sheet" for the latest cautions on development.

**"Check Sheet" is seen at the following support page**

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

<http://edevic.fujitsu.com/micom/en-support/>

**FUJITSU MICROELECTRONICS LIMITED**



# PREFACE

## ■ Objectives and intended reader

Thank you for purchasing Fujitsu semiconductor products.

The MB90820B series was developed as a group of general-purpose models in the F<sup>2</sup>MC-16LX series, which is a family of original 16-bit single-chip microcontrollers that can be used for application specific IC(ASIC).

This manual is intended for engineers who design products using the MB90820B series of microcontrollers. The manual describes the functions and operation of the MB90820B series.

Note: F<sup>2</sup>MC is the abbreviation of FUJITSU Flexible Microcontroller.

## ■ Trademarks

The company names and brand names herein are the trademarks or registered trademarks of their respective owners.

## ■ License

Purchase of Fujitsu I<sup>2</sup>C components conveys a license under the Philips I<sup>2</sup>C Patent Rights to use, these components in an I<sup>2</sup>C system provided that the system conforms to the I<sup>2</sup>C Standard Specification as defined by Philips.

## ■ Organization of this manual

This manual consists of the following 23 chapters and 3 appendices:

### CHAPTER 1 OVERVIEW

This chapter describes the main features and basic specifications of the MB90820B series.

### CHAPTER 2 CPU

This chapter describes memory space for the MB90820B series.

### CHAPTER 3 RESET

This chapter describes the function and operation of the reset for the MB90820B series microcontrollers.

### CHAPTER 4 CLOCK

This chapter describes the function and operation of the clock used by MB90820B series microcontrollers.

### CHAPTER 5 CLOCK SUPERVISOR

This chapter describes the functions and operations of the clock supervisor.

### CHAPTER 6 LOW-POWER CONSUMPTION MODE

This chapter describes the low-power consumption mode of MB90820B series microcontrollers.

### CHAPTER 7 INTERRUPT

This chapter explains the function and operation of the interrupt and extended intelligent I/O service (EI<sup>2</sup>OS) in the MB90820B series.

### CHAPTER 8 MODE SETTING

This chapter describes the operating modes and memory access modes supported by the MB90820B series.

## **CHAPTER 9 I/O PORTS**

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

## **CHAPTER 10 TIME-BASE TIMER**

This chapter describes the functions and operation of the time-base timer.

## **CHAPTER 11 WATCHDOG TIMER**

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

## **CHAPTER 12 16-BIT RELOAD TIMER**

This chapter describes the functions and operations of the 16-bit reload timer.

## **CHAPTER 13 PWC Timer**

This chapter explains the activation and operations of the PWC timer.

## **CHAPTER 14 16-BIT PPG TIMER**

This chapter describes the activation and operation of the 16-bit PPG Timer.

## **CHAPTER 15 MULTI-FUNCTIONAL TIMER**

This chapter describes the functions and operation of the multi-functional timer.

## **CHAPTER 16 DELAYED INTERRUPT GENERATOR MODULE**

This chapter describes the functions and operation of the delayed interrupt generator module.

## **CHAPTER 17 DTP/EXTERNAL INTERRUPT CIRCUIT**

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

## **CHAPTER 18 8/10-Bit A/D Converter**

This chapter explains the function and operation of the A/D converter.

## **CHAPTER 19 D/A CONVERTER**

This chapter explains the functions and operation of the digital/analog (D/A) converter.

## **CHAPTER 20 UART**

This chapter explains the functions and operation of UART.

## **CHAPTER 21 ROM CORRECTION FUNCTION**

This chapter describes the functions and operation of the ROM correction function.

## **CHAPTER 22 ROM MIRRORING FUNCTION SELECTION MODULE**

This chapter explains the function and operation of the MB90820B series ROM mirroring function selection module.

## **CHAPTER 23 512K / 1024K BIT FLASH MEMORY**

The following explains the functions and operations of the 512K / 1024K bit flash memory.

## **APPENDIX**

### **APPENDIX A I/O MAP**

### **APPENDIX B EXAMPLE OF F<sup>2</sup>MC-16LX MB90F822B/F823B CONNECTION FOR SERIAL WRITING**

### **APPENDIX C INSTRUCTIONS**

- The contents of this document are subject to change without notice.  
Customers are advised to consult with 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 MICROELECTRONICS device; FUJITSU MICROELECTRONICS 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 MICROELECTRONICS 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 MICROELECTRONICS or any third party or does FUJITSU MICROELECTRONICS warrant non-infringement of any third-party's intellectual property right or other right by using such information. FUJITSU MICROELECTRONICS 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 MICROELECTRONICS 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.
- Exportation/release of any products described in this document may require necessary procedures in accordance with the regulations of the Foreign Exchange and Foreign Trade Control Law of Japan and/or US export control laws.
- The company names and brand names herein are the trademarks or registered trademarks of their respective owners.





# CONTENTS

<b>CHAPTER 1</b>	<b>OVERVIEW .....</b>	<b>1</b>
1.1	MB90820B Series Features .....	2
1.2	MB90820B Series Product Line-up .....	5
1.3	Block Diagram of MB90820B Series .....	7
1.4	Pin Assignment .....	8
1.5	Package Dimensions .....	10
1.6	I/O Pins and Pin Functions .....	13
1.7	I/O Circuit Types .....	17
1.8	Notes on Handling Devices .....	21
<b>CHAPTER 2</b>	<b>CPU .....</b>	<b>25</b>
2.1	CPU .....	26
2.2	Memory Space .....	27
2.3	Memory Maps .....	29
2.4	Addressing .....	31
2.4.1	Address Specification by Linear Addressing .....	32
2.4.2	Address Specification by Bank Addressing .....	33
2.5	Memory Location of Multibyte Data .....	35
2.6	Registers .....	37
2.7	Dedicated Registers .....	38
2.7.1	Accumulator (A) .....	40
2.7.2	Stack Pointers (USP, SSP) .....	43
2.7.3	Processor Status (PS) .....	46
2.7.4	Condition Code Register (PS: CCR) .....	47
2.7.5	Register Bank Pointer (PS: RP) .....	49
2.7.6	Interrupt Level Mask Register (PS: ILM) .....	50
2.7.7	Program Counter (PC) .....	51
2.7.8	Direct Page Register (DPR) .....	52
2.7.9	Bank Registers (PCB, DTB, USB, SSB, ADB) .....	53
2.8	General-purpose Registers .....	54
2.9	Prefix Codes .....	56
2.9.1	Bank Select Prefix (PCB, DTB, ADB, SPB) .....	57
2.9.2	Common Register Bank Prefix (CMR) .....	59
2.9.3	Flag Change Suppression Prefix (NCC) .....	60
2.9.4	Restrictions on Prefix Codes .....	61
<b>CHAPTER 3</b>	<b>RESET .....</b>	<b>63</b>
3.1	Reset .....	64
3.2	Reset Causes and Oscillation Stabilization Wait Intervals .....	66
3.3	External Reset Pin .....	68
3.4	Reset Operation .....	69
3.5	Reset Cause Bits .....	71
3.6	Status of Pins in a Reset .....	73

<b>CHAPTER 4</b>	<b>CLOCK</b>	<b>75</b>
4.1	Clock	76
4.2	Block Diagram of the Clock Generation Block	78
4.3	Clock Selection Registers	80
4.3.1	Clock Selection Register (CKSCR)	81
4.3.2	PLL Clock Control Register (PCKCR)	83
4.4	Clock Mode	85
4.5	Oscillation Stabilization Wait Interval	87
4.6	Connection of an Oscillator or an External Clock to the Microcontroller	88
<b>CHAPTER 5</b>	<b>CLOCK SUPERVISOR</b>	<b>89</b>
5.1	Overview of Clock Supervisor	90
5.2	Configuration of Clock Supervisor	91
5.3	Registers of Clock Supervisor	93
5.3.1	Clock Supervisor Control Register (CSVCR)	94
5.4	Operations of Clock Supervisor	96
5.5	Precautions when Using Clock Supervisor	99
<b>CHAPTER 6</b>	<b>LOW-POWER CONSUMPTION MODE</b>	<b>101</b>
6.1	Low-Power Consumption Mode	102
6.2	Block Diagram of the Low-Power Consumption Control Circuit	105
6.3	Low-Power Consumption Mode Control Register (LPMCR)	107
6.4	CPU Intermittent Operation Mode	110
6.5	Standby Mode	111
6.5.1	Sleep Mode	112
6.5.2	Time-base Timer Mode	115
6.5.3	Stop Mode	117
6.6	State Change Diagram	119
6.7	State of Pins in Standby Mode and During Reset	122
6.8	Usage Notes on Low-Power Consumption Mode	123
<b>CHAPTER 7</b>	<b>INTERRUPT</b>	<b>127</b>
7.1	Interrupt	128
7.2	Interrupt Causes and Interrupt Vectors	130
7.3	Interrupt Control Registers and Peripheral Functions	133
7.3.1	Interrupt Control Registers (ICR00 to ICR15)	135
7.3.2	Interrupt Control Register Functions	137
7.4	Hardware Interrupt	140
7.4.1	Operation of Hardware Interrupt	143
7.4.2	Processing for Interrupt Operation	145
7.4.3	Procedure for Using Hardware Interrupt	146
7.4.4	Multiple Interrupts	147
7.4.5	Hardware Interrupt Processing Time	149
7.5	Software Interrupt	151
7.6	Interrupt of Extended Intelligent I/O Service (EI <sup>2</sup> OS)	153
7.6.1	Extended Intelligent I/O Service (EI <sup>2</sup> OS) Descriptor (ISD)	155
7.6.2	Registers of EI <sup>2</sup> OS Descriptor (ISD)	156

7.6.3	Operation of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	159
7.6.4	Procedure for Using the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	160
7.6.5	Processing Time of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	161
7.7	Exception Processing Interrupt .....	163
7.8	Stack Operations for Interrupt Processing .....	164
<b>CHAPTER 8</b>	<b>MODE SETTING .....</b>	<b>167</b>
8.1	Mode Setting .....	168
8.2	Mode Pins (MD2 to MD0) .....	169
8.3	Mode Data .....	170
<b>CHAPTER 9</b>	<b>I/O PORTS .....</b>	<b>173</b>
9.1	Overview of I/O Ports .....	174
9.2	Registers of I/O Ports .....	176
9.3	Port 0 .....	177
9.3.1	Port 0 Registers (PDR0, DDR0, and RDR0) .....	180
9.3.2	Operation of Port 0 .....	182
9.4	Port 1 .....	184
9.4.1	Port 1 Registers (PDR1, DDR1, and RDR1) .....	187
9.4.2	Operation of Port 1 .....	189
9.5	Port 2 .....	191
9.5.1	Port 2 Registers (PDR2, DDR2, and RDR2) .....	194
9.5.2	Operation of Port 2 .....	196
9.6	Port 3 .....	198
9.6.1	Port 3 Registers (PDR3, DDR3, and RDR3) .....	201
9.6.2	Operation of Port 3 .....	203
9.7	Port 4 .....	205
9.7.1	Port 4 Registers (PDR4 and DDR4) .....	208
9.7.2	Operation of Port 4 .....	209
9.8	Port 5 .....	211
9.8.1	Port 5 Registers (PDR5 and DDR5) .....	214
9.8.2	Operation of Port 5 .....	215
9.9	Port 6 .....	217
9.9.1	Port 6 Registers (PDR6, DDR6, and ADER0) .....	219
9.9.2	Operation of Port 6 .....	221
9.10	Port 7 .....	223
9.10.1	Port 7 Registers (PDR7, DDR7, and ADER1) .....	228
9.10.2	Operation of Port 7 .....	230
9.11	Port 8 .....	232
9.11.1	Port 8 Registers (PDR8 and DDR8) .....	235
9.11.2	Operation of Port 8 .....	236
<b>CHAPTER 10</b>	<b>TIME-BASE TIMER .....</b>	<b>239</b>
10.1	Overview of the Time-base Timer .....	240
10.2	Configuration of the Time-base Timer .....	242
10.3	Time-base Timer Control Register (TBTC) .....	244
10.4	Time-base Timer Interrupts .....	246

10.5	Operation of the Time-base Timer .....	247
10.6	Usage Notes on the Time-base Timer .....	249
<b>CHAPTER 11</b>	<b>WATCHDOG TIMER .....</b>	<b>251</b>
11.1	Overview of the Watchdog Timer .....	252
11.2	Configuration of the Watchdog Timer .....	253
11.3	Watchdog Timer Control Register (WDTC) .....	255
11.4	Operation of the Watchdog Timer .....	257
11.5	Usage Notes on the Watchdog Timer .....	259
<b>CHAPTER 12</b>	<b>16-BIT RELOAD TIMER .....</b>	<b>261</b>
12.1	Overview of 16-Bit Reload Timer .....	262
12.2	Block Diagram of 16-Bit Reload Timer .....	265
12.3	Pins of 16-Bit Reload Timer .....	267
12.4	Registers of 16-Bit Reload Timer .....	268
12.4.1	Upper Bits of Timer Control Status Registers (TMCSRH0/TMCSRH1) .....	269
12.4.2	Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1) .....	271
12.4.3	16-Bit Timer Registers (TMR0/TMR1) .....	273
12.4.4	16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) .....	274
12.5	Interrupts of 16-Bit Reload Timer .....	275
12.6	Operation of 16-Bit Reload Timer .....	276
12.6.1	Internal Clock Mode (Reload Mode) .....	278
12.6.2	Internal Clock Mode (One-Shot Mode) .....	281
12.6.3	Event Count Mode .....	284
12.7	Notes on Using the 16-Bit Reload Timer .....	286
<b>CHAPTER 13</b>	<b>PWC Timer .....</b>	<b>287</b>
13.1	Overview of the PWC Timer .....	288
13.2	Block Diagram of the PWC Timer .....	289
13.3	PWC Timer Pins .....	290
13.4	PWC Timer Registers .....	293
13.4.1	PWC Control Status Register (PWCSH0/PWCSH1, PWCSL0/PWCSL1) .....	294
13.4.2	PWC Data Buffer Register (PWC0/PWC1) .....	299
13.4.3	Division Ratio Control Register (DIV0/DIV1) .....	301
13.5	PWC Timer Interrupts .....	302
13.6	Operation of the PWC Timer .....	304
13.6.1	Operation Mode Selection .....	307
13.6.2	Starting and Stopping the Timer and Pulse-width Measurement and Clearing the Timer .....	309
13.6.3	Timer Mode Operation .....	311
13.6.4	Pulse Width Measurement Mode Operation .....	314
13.7	Usage Notes on the PWC Timer .....	319
<b>CHAPTER 14</b>	<b>16-BIT PPG TIMER .....</b>	<b>321</b>
14.1	Overview of 16-bit PPG Timer .....	322
14.2	Block Diagram of 16-bit PPG Timer .....	323
14.3	16-bit PPG Timer Pins .....	324
14.4	16-bit PPG Timer Registers .....	326

14.4.1	PPG Down Counter Register (PDCR0 to PDCR2)	328
14.4.2	PPG Period Setting Buffer Register (PCSR0 to PCSR2)	329
14.4.3	PPG Duty Setting Buffer Register (PDUT0 to PDUT2)	330
14.4.4	PPG Control Status Register (PCNTL0 to PCNTL2, PCNTH0 to PCNTH2)	331
14.5	16-bit PPG Timer Interrupts	336
14.6	Operation of 16-bit PPG Timer	338
14.7	Usage Notes on the 16-bit PPG Timer	342
<b>CHAPTER 15 MULTI-FUNCTIONAL TIMER</b>		<b>343</b>
15.1	Overview of Multi-functional Timer	344
15.2	Block Diagram of Multi-functional Timer	346
15.3	Multi-functional Timer Pins	350
15.4	Registers of Multi-functional Timer	353
15.4.1	Compare Clear Buffer Register (CPCLRB) and Compare Clear Register (CPCLR)	357
15.4.2	Timer Data Register (TCDT)	358
15.4.3	Timer Control Status Register (TCCSH, TCCSL)	359
15.4.4	Output Compare Buffer Registers (OCCPB0 to OCCPB5)/Output Compare Registers (OCCP0 to OCCP5)	364
15.4.5	Compare Control Registers (OCS0 to OCS5)	366
15.4.6	Input Capture Register (IPCP0 to IPCP3)	371
15.4.7	Input Capture Control Status Registers (ICS23, PICS01)	372
15.4.8	16-bit Timer Register (TMRR0 to TMRR2)	380
15.4.9	16-bit Timer Control Register (DTCR0 to DTCR2)	381
15.4.10	Waveform Control Register (SIGCR)	385
15.5	Multi-functional Timer Interrupts	387
15.6	Operation of Multi-functional Timer	391
15.6.1	Operation of 16-bit Free-run Timer	392
15.6.2	Operation of 16-bit Output Compare	398
15.6.3	Operation of 16-bit Input Capture	403
15.6.4	Operation of Waveform Generator	405
15.7	Usage Notes on the Multi-functional Timer	416
<b>CHAPTER 16 DELAYED INTERRUPT GENERATOR MODULE</b>		<b>419</b>
16.1	Overview of the Delayed Interrupt Generator Module	420
16.2	Delayed Interrupt Generator Module Register	421
16.3	Operation of the Delayed Interrupt Generator Module	422
16.4	Usage Notes on the Delayed Interrupt Generator Module	423
<b>CHAPTER 17 DTP/EXTERNAL INTERRUPT CIRCUIT</b>		<b>425</b>
17.1	Overview of the DTP/External Interrupt Circuit	426
17.2	Block Diagram of the DTP/External Interrupt Circuit	428
17.3	DTP/External Interrupt Circuit Pins	430
17.4	DTP/External Interrupt Circuit Registers	432
17.4.1	DTP/external interrupt cause register (EIRR)	433
17.4.2	DTP/external interrupt enable register (ENIR)	434
17.4.3	Request Level Setting Register (ELVR)	436
17.5	Operation of the DTP/External Interrupt Circuit	438

17.5.1	External Interrupt Function .....	441
17.5.2	DTP Function .....	442
17.6	Usage Notes on the DTP/External Interrupt Circuit .....	444
<b>CHAPTER 18</b>	<b>8/10-Bit A/D Converter .....</b>	<b>447</b>
18.1	Overview of 8/10-Bit A/D Converter .....	448
18.2	Block Diagram of 8/10-Bit A/D Converter .....	450
18.3	Configuration of 8/10-Bit A/D Converter .....	453
18.3.1	A/D Control Status Registers High Order (ADCS1) .....	455
18.3.2	A/D Control Status Register Low Order (ADCS0) .....	459
18.3.3	A/D Data Register (ADCR0/ADCR1) .....	461
18.3.4	A/D Setting Register (ADSR0/ADSR1) .....	462
18.3.5	Analog Input Enable Resister (ADER0/ADER1) .....	467
18.4	Interrupt of 8/10-Bit A/D Converter .....	468
18.5	Operation of 8/10-Bit A/D Converter .....	469
18.5.1	Single Conversion Mode .....	470
18.5.2	Continuous Conversion Mode .....	472
18.5.3	Stop Conversion Mode .....	474
18.5.4	Conversion Using EI <sup>2</sup> OS .....	476
18.5.5	A/D Converted Data Protection Function .....	478
18.6	Precautions for Using the 8/10-Bit A/D Converter .....	482
<b>CHAPTER 19</b>	<b>D/A CONVERTER .....</b>	<b>483</b>
19.1	Overview of D/A Converter .....	484
19.2	Block Diagram of D/A Converter .....	485
19.3	D/A Converter Pins .....	486
19.4	D/A Converter Registers .....	487
19.4.1	D/A Converter Register 1 (DAT1) .....	488
19.4.2	D/A Converter Register 0 (DAT0) .....	489
19.4.3	D/A Control Register 1 (DACR1) .....	490
19.4.4	D/A Control Register 0 (DACR0) .....	491
<b>CHAPTER 20</b>	<b>UART .....</b>	<b>493</b>
20.1	Overview of UART .....	494
20.2	Block Diagram of UART .....	496
20.3	UART Pins .....	499
20.4	UART Registers .....	502
20.4.1	Serial Control Register (SCR0/SCR1) .....	503
20.4.2	Serial Mode Register (SMR0/SMR1) .....	505
20.4.3	Serial Status Register (SSR0/SSR1) .....	508
20.4.4	Serial Input Data Register (SIDR0/SIDR1) and Serial Output Data Register (SODR0/SODR1) .....	510
20.4.5	Communication Prescaler Control Register (CDCR) .....	512
20.5	UART Interrupts .....	514
20.5.1	Reception Interrupt Request Generation and Flag Set Timing .....	516
20.5.2	Transmission Interrupt Request Generation and Flag Set Timing .....	518
20.6	UART Baud Rates .....	520

20.6.1	Baud Rates Determined Using the Dedicated Baud Rate Generator .....	522
20.6.2	Baud Rates Determined Using the Internal Timer (16-bit Reload Timer) .....	525
20.6.3	Baud Rates Determined Using the External Clock .....	527
20.7	Operation of UART .....	528
20.7.1	Operation in Asynchronous Mode (Operation Modes 0 and 1) .....	530
20.7.2	Operation in Clock Synchronous Mode (Operation Mode 2) .....	532
20.7.3	Bidirectional Communication Function (Normal Mode) .....	534
20.7.4	Master-slave Communication Function (Multiprocessor Mode) .....	536
20.8	Usage Notes on UART .....	539
<b>CHAPTER 21</b>	<b>ROM CORRECTION FUNCTION .....</b>	<b>541</b>
21.1	Overview of the ROM Correction Function .....	542
21.2	Block Diagram of ROM Correction Function .....	543
21.3	ROM Correction Function Registers .....	544
21.3.1	Program Address Detection Register (PADR0/PADR1) .....	545
21.3.2	Program Address Detection Control Status Register (PACSR) .....	546
21.4	Operation of the ROM Correction Function .....	548
21.5	Example of Using ROM Correction Function .....	549
<b>CHAPTER 22</b>	<b>ROM MIRRORING FUNCTION SELECTION MODULE .....</b>	<b>553</b>
22.1	Overview of the ROM Mirroring Function Selection Module .....	554
22.2	ROM Mirroring Function Selection Register (ROMM) .....	555
<b>CHAPTER 23</b>	<b>512K / 1024K BIT FLASH MEMORY .....</b>	<b>557</b>
23.1	Overview of the 512K / 1024K Bit Flash Memory .....	558
23.2	512K / 1024K Bit Flash Memory Sector Configuration .....	559
23.3	Flash Memory Control Status Register (FMCS) .....	560
23.4	Method of Starting the Automatic Algorithm in Flash Memory .....	563
23.5	Verifying Automatic Algorithm Execution Status .....	564
23.5.1	Data Polling Flag (DQ7) .....	566
23.5.2	Toggle Bit Flag (DQ6) .....	568
23.5.3	Time limit Exceeded Flag (DQ5) .....	570
23.5.4	Sector Deletion Timer Flag (DQ3) .....	571
23.6	Detailed Explanation on the Flash Memory Write/Delete .....	572
23.6.1	Setting the Read/Reset Status .....	573
23.6.2	Writing the Data .....	574
23.6.3	Deleting All Data (Chip Deletion) .....	576
23.6.4	Deleting Arbitrary Data (Sector Deletion) .....	577
23.6.5	Temporarily Stopping the Sector Deletion .....	579
23.6.6	Restarting the Sector Deletion .....	580
23.7	Flash Security Feature .....	581
<b>APPENDIX</b>	<b>.....</b>	<b>583</b>
APPENDIX A	I/O Map .....	584
APPENDIX B	Example of F <sup>2</sup> MC-16LX MB90F822B/F823B Connection for Serial Writing .....	590
B.1	Standard Configuration for Serial On-board Writing (Fujitsu Standard) .....	591
B.2	Example of Connection for Serial Writing (When Power Supplied by User) .....	594



B.3	Example of Connection for Serial Writing (When Power Supplied from Writer)	596
B.4	Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied by User)	598
B.5	Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied from Writer)	600
APPENDIX C Instructions		602
C.1	Instruction Types	603
C.2	Addressing	604
C.3	Direct Addressing	606
C.4	Indirect Addressing	612
C.5	Execution Cycle Count	620
C.6	Effective address field	623
C.7	How to Read the Instruction List	624
C.8	F <sup>2</sup> MC-16LX Instruction List	627
C.9	Instruction Map	641
<b>INDEX</b>		<b>663</b>

# Main changes in this edition

Page		Changes (For details, refer to main body.)
-		Added Part number (MB90F828B) Added "CHAPTER 5 CLOCK SUPERVISOR" Changed Package ("FPT-80P-M05" → "FPT-80P-M21") ("FPT-80P-M11" → "FPT-80P-M22")
7	CHAPTER 1 OVERVIEW 1.3 Block Diagram of MB90820B Series	Changed "Figure 1.3-1 MB90820B Series Overall Block Diagram"
64	CHAPTER 3 RESET 3.1 Reset	Changed "Table3.1-1 Reset causes" (Added "Clock supervisor reset")
65		Changed "■ Reset Causes" (Added "● Clock supervisor reset")
66	CHAPTER 3 RESET 3.2 Reset causes and oscillation stabilization wait intervals	Changed "Table3.2-1 Reset causes and oscillation stabilization wait intervals" (Added "Clock supervisor reset")
69	CHAPTER 3 RESET 3.4 Reset Operation	Changed "Figure 3.4-1 Reset Operation flow" (Added "Clock supervisor reset")
71	CHAPTER 3 RESET 3.5 Reset Cause Bits	Changed "Figure 3.5-1 Block Diagram of reset cause bits"
72		Changed "Table3.5-1 Correspondence between reset cause flag bits and reset causes" (Added "Clock supervisor reset")
77	CHAPTER 4 CLOCK 4.1 Clock	Changed "Figure 4.1-1 Clock Supply Map"
78	CHAPTER 4 CLOCK 4.2 Block Diagram of the Clock Generation Block	Changed "● Clock selector → • Operating Clock selector" Changed "Figure 4.2-1 Block Diagram of the clock generation block"
79		Changed "● Clock selector → ● Operating Clock selector"
105	CHAPTER 6 LOW-POWER CONSUMPTION MODE 6.2 Block Diagram of the Low-Power Consumption Control Circuit	Changed "Figure 6.2-1 Block Diagram of the low-power consumption control circuit"
548	CHAPTER 21 ROM CORRECTION FUNCTION 21.4 Operation of the ROM Correction Function	Changed "■ Operation of the ROM Correction Function" (Added "Notes")

Page		Changes (For details, refer to main body.)
556	CHAPTER 22 ROM MIRRORING FUNCTION SELECTION MODULE 22.2 ROM mirroring function selection regis- ter(ROMM)	Changed Table(Address 1, Address 2) in "■ ROM mirroring function selection register (ROMM)"
587	APPENDIX APPENDIX A I/O Map	Changed "Address 00008A <sub>H</sub> " in "TableA-1 I/O map"
602 to 662	APPENDIX APPENDIX C Instructions	Changed the entire part of "APPENDIX C Instructions"

The vertical lines marked in the left side of the page show the changes.

# **CHAPTER 1**

---

# **OVERVIEW**

**This chapter describes the main features and basic specifications of the MB90820B series.**

- 1.1 MB90820B Series Features
- 1.2 MB90820B Series Product Line-up
- 1.3 Block Diagram of MB90820B Series
- 1.4 Pin Assignment
- 1.5 Package Dimensions
- 1.6 I/O Pins and Pin Functions
- 1.7 I/O Circuit Types
- 1.8 Notes on Handling Devices

## 1.1 MB90820B Series Features

---

The MB90820B series is a line of general-purpose, 16-bit microcontrollers designed for those applications which require high-speed real-time processing, proving to be suitable for various industrial machines and motor control (AC induction motor and brushless DC motor). These microcontrollers consist of a multi-functional timer for AC/DC motor control and a multi-pulse generator for DC motor control, which can generate various type of waveform.

The instruction set is designed to be optimized for controller applications which inheriting the AT architecture of F<sup>2</sup>MC-16LX series and allow a wide range of control tasks to be processed efficiently at high speed.

---

### ■ MB90820B Series Features

- Clock
  - Embedded PLL clock multiplication circuit
  - Operating clock (PLL clock) can selected from divided-by-2 of oscillation, one to four times or six times the oscillation (at oscillation of 4 MHz, 4 MHz to 16 MHz or 24MHz)
  - Minimum instruction execution time of 42 ns (at oscillation of 4 MHz, six times the PLL clock, operation at Vcc of 5.0 V)
- CPU addressing space of 16 Mbytes
  - Internal 24-bit addressing
- Instruction set optimized for controller applications
  - Rich data types (bit, byte, word, long word)
  - Rich addressing mode (23 types)
  - High code efficiency
  - Enhanced precision calculation realized by the 32-bit accumulator
- Instruction set designed for H level language (C) and multi-task operations
  - Adoption of system stack pointer
  - Enhanced pointer indirect instructions
  - Barrel shift instructions
- Program patch function (2 address pointer)
- Improved execution speed
  - 4-byte instruction queue

## MB90820B Series

- Powerful interrupt function
  - Priority level programmable: 8 levels
  - 32 factors of stronger interrupt function
- Automatic data transmission function independent of CPU operation
  - Extended intelligent I/O service function (EI<sup>2</sup>OS)
  - Maximum 16 channels
- Low-power consumption (standby) mode
  - Sleep mode (mode in which CPU operating clock is stopped)
  - Time-base timer mode (mode in which other than oscillation and time-base timer are stopped)
  - Stop mode (mode in which oscillation is stopped)
  - CPU intermittent operation mode
- Package
  - LQFP-80 (FPT-80P-M21: 0.50 mm pitch)
  - LQFP-80 (FPT-80P-M22: 0.65 mm pitch)
  - QFP-80 (FPT-80P-M06: 0.80 mm pitch)
- Process

CMOS technology

### ■ Internal Peripheral Features

- I/O port

Maximum of 66 ports
- 18-bit time-base counter/watchdog timer: 1 channel
- Watchdog timer: 1 channel
- PWC: 2 channels
- 16-bit reload timer: 2 channels
- 16-bit PPG timer: 3 channels
- Multi-functional timer (for AC/DC motor control): 1 channel
  - 16-bit free-run timer with up or up-down mode selection and buffer: 1 channel
  - 16-bit output compare with buffer: 6 channels
  - 16-bit input capture: 4 channels
  - 16-bit PPG timer: 1 channel
  - Waveform generator (16-bit timer: 3 channels, 3-phase waveform or dead time)

● UART: 2 channels

- With full-duplex double buffer (8-bit length)
- Clock asynchronized or clock synchronized transmission (with start and stop bits) can be selectively used

● DTP/External interrupt circuit: 8 channels

A module for starting extended intelligent I/O service (EI<sup>2</sup>OS) and generating an external interrupt triggered by an external input

● Delayed interrupt generation module

Generates an interrupt request for switching tasks

● 8/10-bit A/D converter: 16 channels

Selectable 8/10-bit resolution

● 8-bit D/A converter: 2 channels

● Clock supervisor

# MB90820B Series

## 1.2 MB90820B Series Product Line-up

The MB90820B series contains 3 different devices. Table 1.2-1 lists the product line-up.

### ■ MB90820B Series Product Line-up

Table 1.2-1 MB90820B Series Product Line-up (1 / 2)

<div>Part number</div> <div>Item</div>	MB90V820B	MB90F822B	MB90F823B	MB90F828B	MB90822B	MB90823B
Classification	Development / evaluation product	Mass-produced products (Flash ROM with flash security)			Mass-produced product (Mask ROM)	
ROM size	—	64K Bytes	128K Bytes		64K Bytes	128K Bytes
RAM size	16K Bytes	4K Bytes		8K Bytes	4K Bytes	
CPU function	Number of instruction : 351					
I/O port	Minimum execution time : 42 ns / 4 MHz (PLL: 4MHz x 6)					
	Addressing mode : 23					
	Data bit length : 1, 8, 16 bits					
	Maximum memory space : 16 MBytes					
	I/O port (CMOS) : 66					
PWC	Pulse width counter timer : 2 channels					
	Timer function (select the counter timer from three internal clocks) Various pulse width measuring function (H pulse width, L pulse width, rising edge to falling edge period, falling edge to rising edge period, rising edge to rising edge period and falling edge to falling edge period)					
UART	UART : 2 channels With full-duplex double buffer (8-bit length) Clock asynchronized or clock synchronized transmission (with start and stop bits) can be selected and used. Transmission can be one-to-one (bidirectional commuication) or one-to-n (master-slave communication)					
16-bit reload timer	Reload timer : 2 channels Reload mode, single-shot mode or event count mode selectable					
16-bit PPG timer	PPG timer : 3 channels					
	PWM mode or single-shot mode selectable Channel 0 can be worked with multi-functional timer or individually					
Multi-functional timer (for AC/DC motor control)	16-bit free-run timer with up or up-down mode selection and buffer: 1 channel 16-bit output compare : 6 channels 16-bit input capture : 4 channels 16-bit PPG timer : 1 channel Waveform generator (16-bit timer: 3 channels, 3-phase waveform or dead time)					
8/10-bit A/D converter	8/10-bit resolution (16 channels) Conversion time : Min. 3 μs (24 MHz internal clock, including sampling time)					
Clock supervisor	No			Yes	No	
8-bit D/A converter	8-bit resolution (2 channels)					



**Table 1.2-1 MB90820B Series Product Line-up (2 / 2)**

<div>Part number</div> <div>Item</div>	MB90V820B	MB90F822B	MB90F823B	MB90F828B	MB90822B	MB90823B
DTP/External interrupt	8 independent channels Selectable causes : Rising edge, falling edge, "L" level or "H" level					
Low-power consumption	Stop mode / Sleep mode / CPU intermittent operation mode					
Package	PGA299	LQFP-80 (FPT-80P-M21 : 0.50 mm pitch) LQFP-80 (FPT-80P-M22 : 0.65 mm pitch) QFP-80 (FPT-80P-M06 : 0.80 mm pitch)				
Power supply voltage for operation	4.5 V to 5.5 V <sup>*1</sup>	3.5 V to 5.5 V : other than conditions listed below 4.0 V to 5.5 V : if A/D converter is used 4.5 V to 5.5 V : if D/A converter is used / writing to FLASH				
Process	CMOS					
Emulator power supply <sup>*2</sup>	Yes	—				

\*1: The guaranteed operating temperature range for the MB90V820B is 0 to +25°C.

\*2: These are the jumper switch (TOOL, VCC) settings for using the emulator (MB2147-01). See "3.3 Selecting the Power Supply for the Emulator" in the hardware manual for the MB2147-01 and MB2147-20 for details.

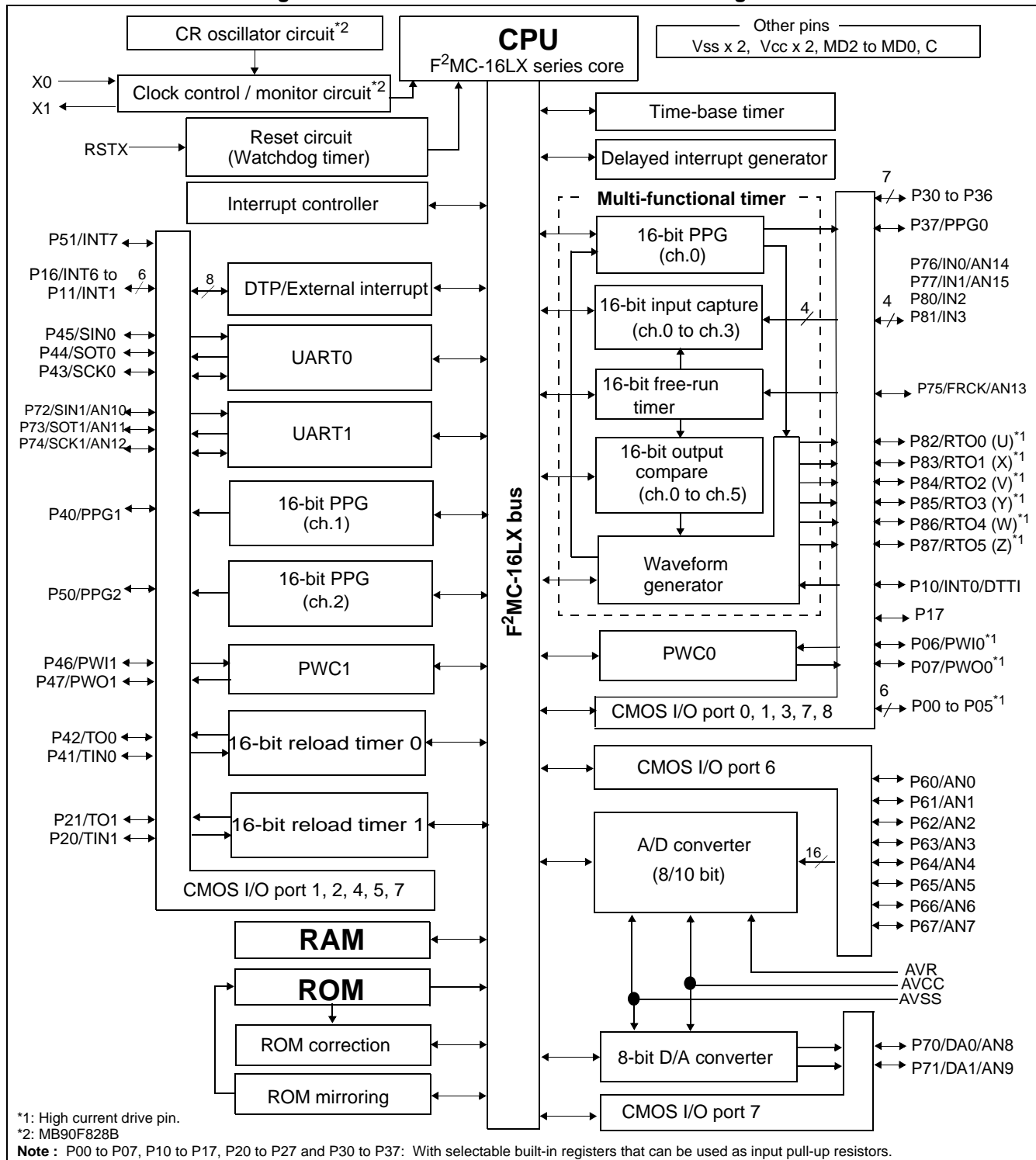
# MB90820B Series

## 1.3 Block Diagram of MB90820B Series

Figure 1.3-1 show the MB90820B series overall block diagram.

### ■ MB90820B Series Block Diagram

Figure 1.3-1 MB90820B Series Overall Block Diagram

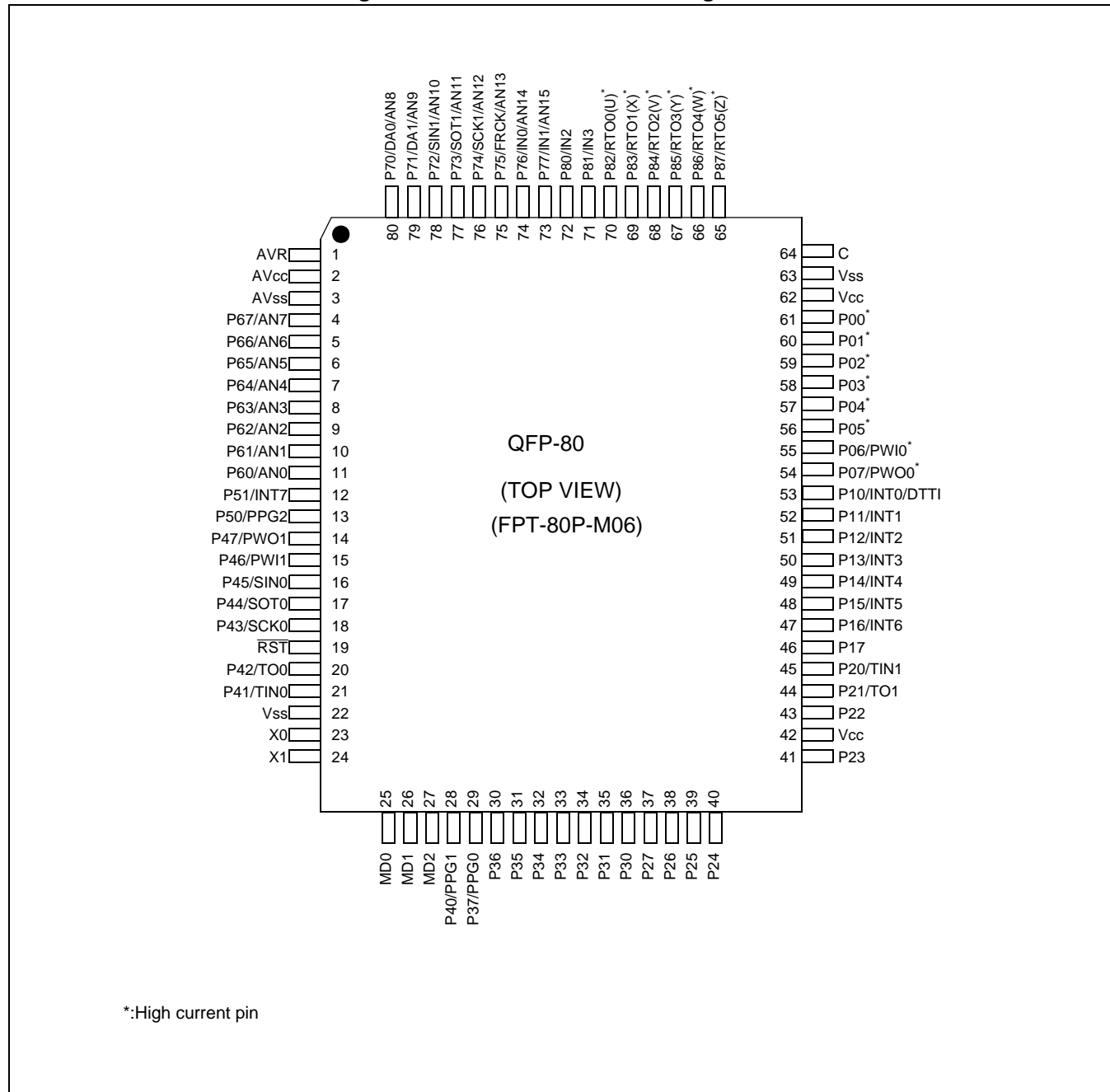


## 1.4 Pin Assignment

Figure 1.4-1, Figure 1.4-2 show the pin assignment diagrams for the MB90820B series.

### ■ FPT-80P-M06 Pin Assignment

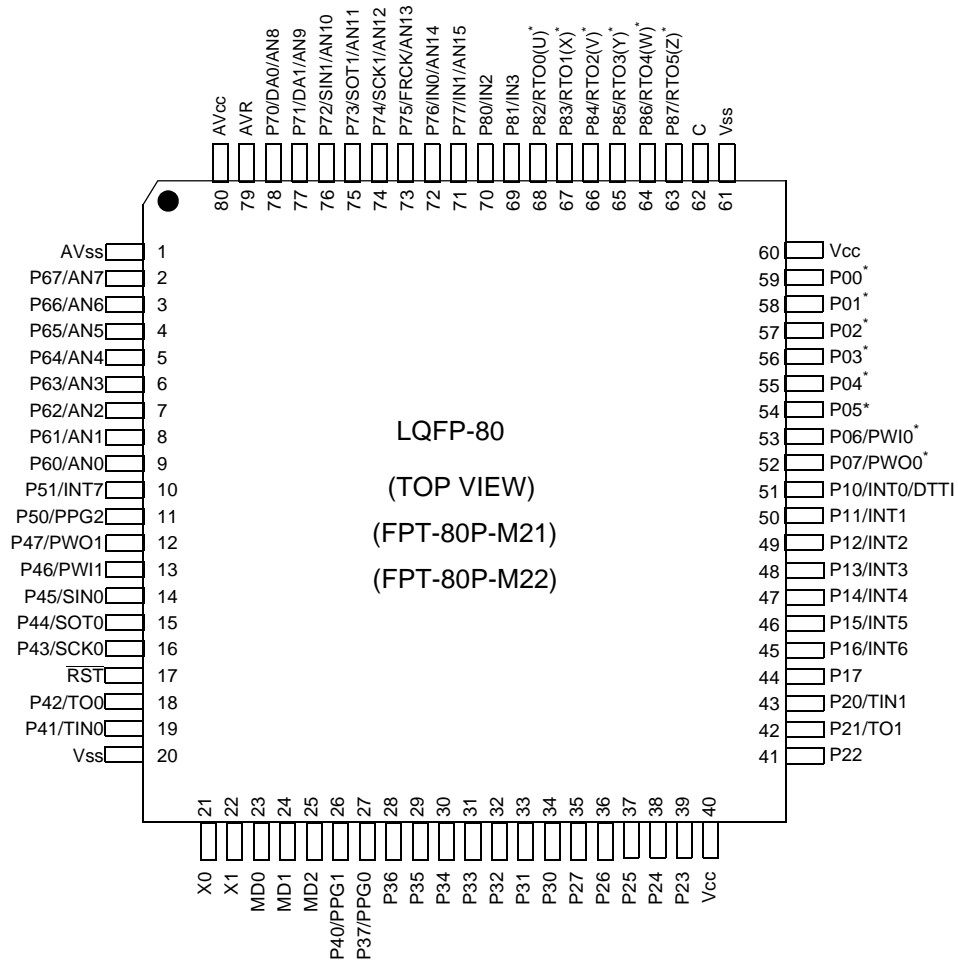
Figure 1.4-1 FPT-64P-M06 Pin Assignment



# MB90820B Series

## ■ FPT-80P-M21/FPT-80P-M22 Pin Assignment

Figure 1.4-2 FPT-80P-M21/FPT-80P-M22 Pin Assignment



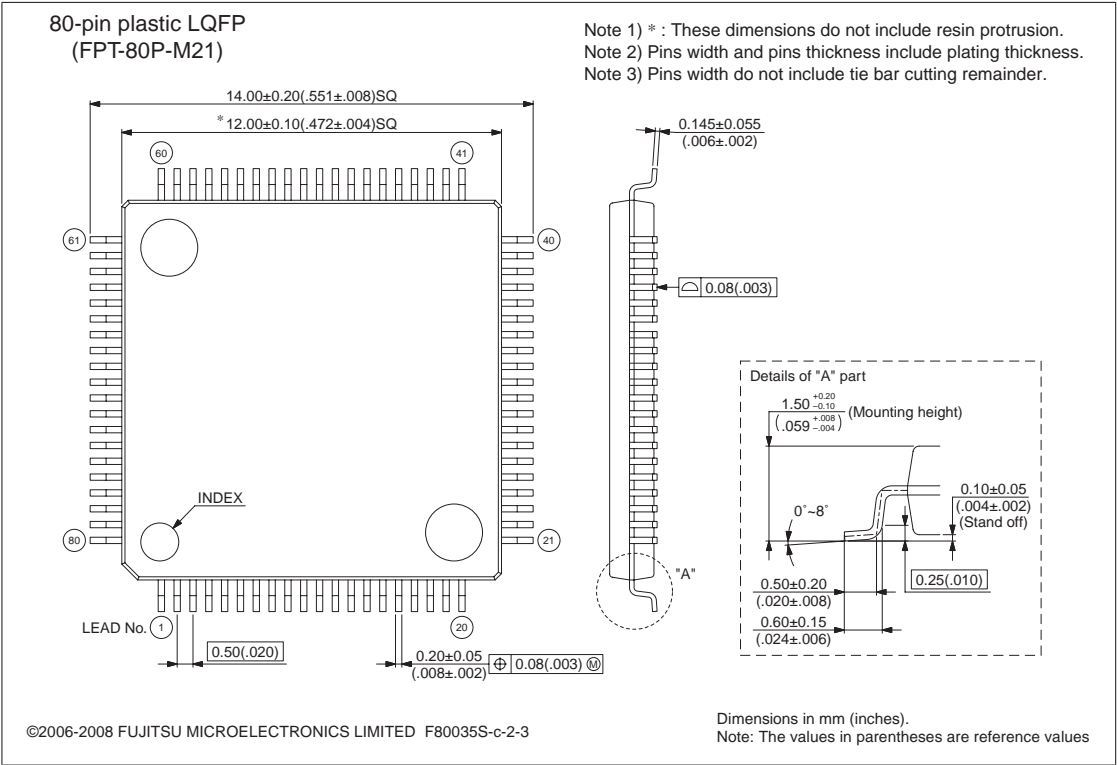
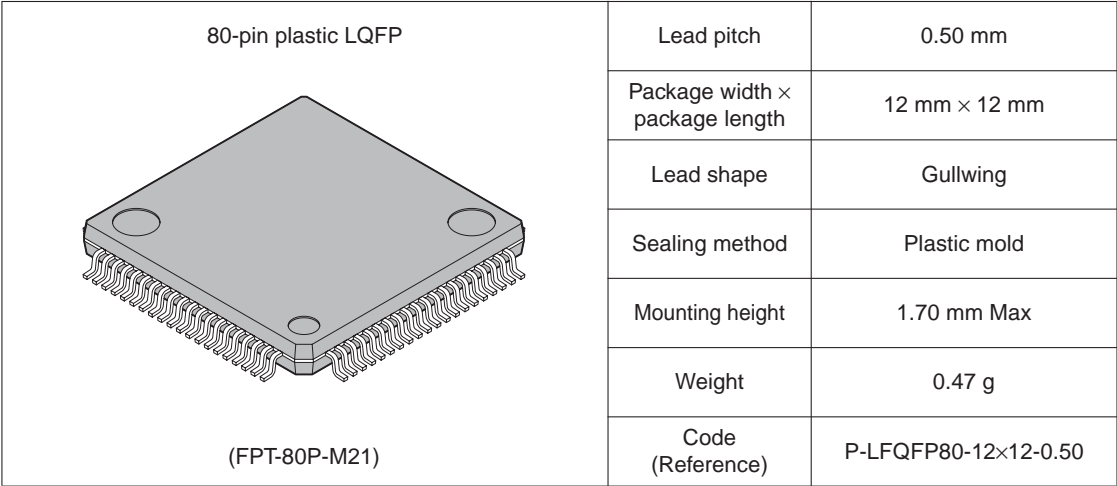
\*:High current pin

1.5 Package Dimensions

Three types of packages are available for MB90820B series.  
Figure 1.5-1 to Figure 1.5-3 show the package dimensions.

■ FPT-80P-M21 Package Dimensions

Figure 1.5-1 FPT-80P-M21 Package Dimensions

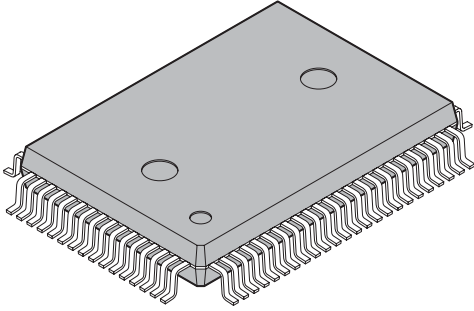


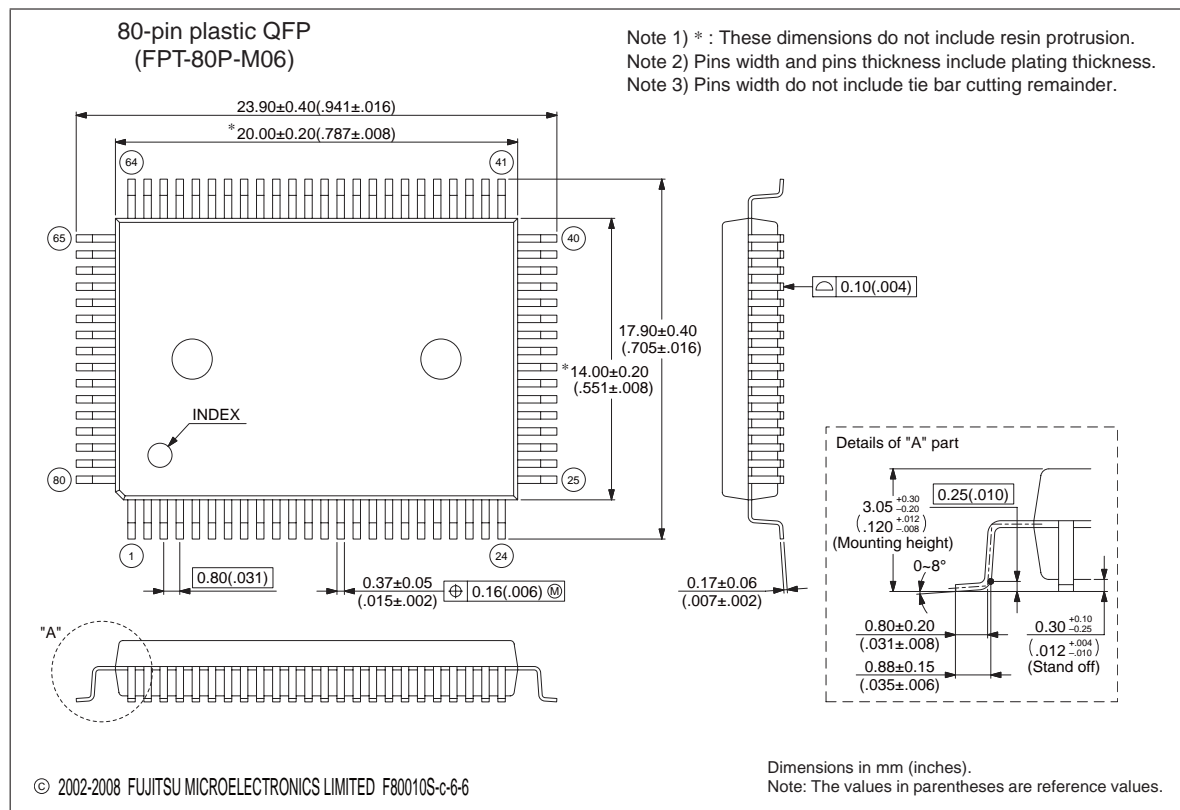
Please confirm the latest Package dimension by following URL.  
<http://edevic.fujitsu.com/package/en-search/>

# MB90820B Series

## ■ FPT-80P-M06 Package Dimensions

Figure 1.5-2 FPT-80P-M06 Package Dimensions

 <p>80-pin plastic QFP</p> <p>(FPT-80P-M06)</p>	Lead pitch	0.80 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-QFP80-14×20-0.80

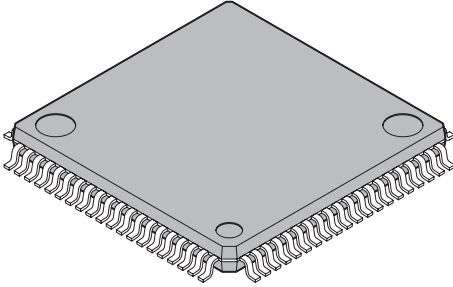


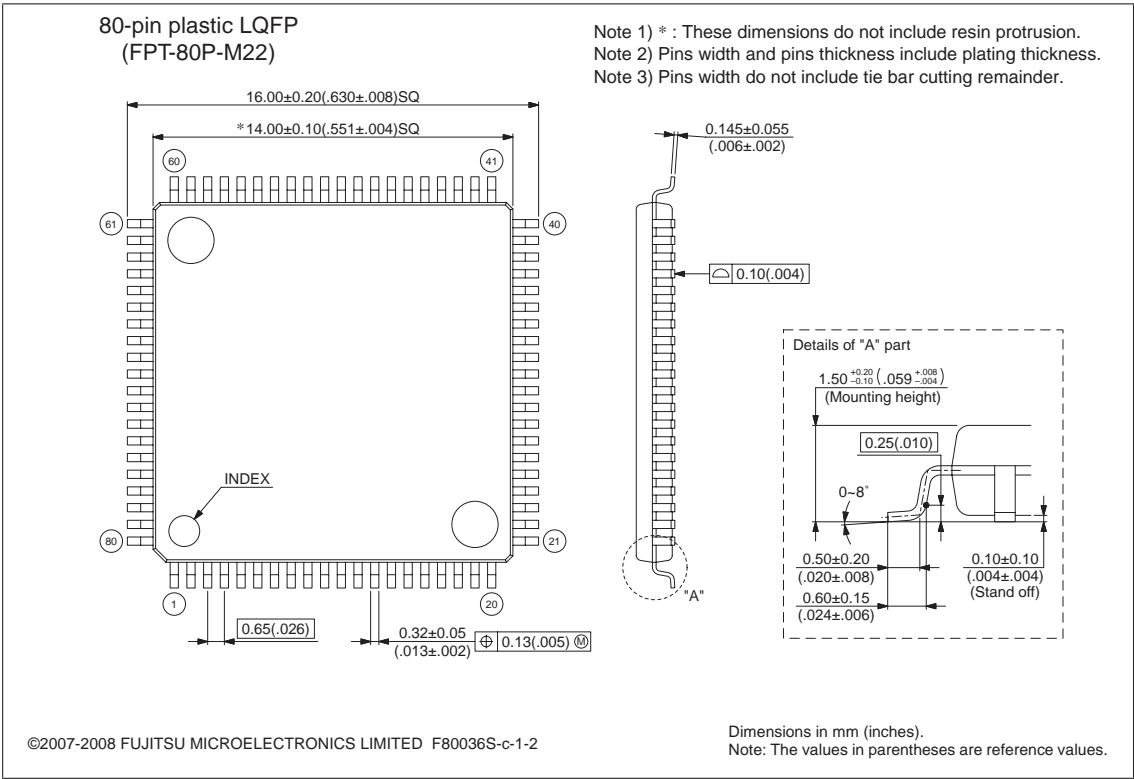
Please confirm the latest Package dimension by following URL.

<http://edevice.fujitsu.com/package/en-search/>

■ FPT-80P-M22 Package Dimensions

Figure 1.5-3 FPT-80P-M22 Package Dimensions

 <p>80-pin plastic LQFP</p> <p>(FPT-80P-M22)</p>	Lead pitch	0.65 mm
	Package width × package length	14.00 mm × 14.00 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm Max
	Weight	0.62 g
	Code (Reference)	P-LFQFP80-14×14-0.65



Please confirm the latest Package dimension by following URL.  
<http://edevice.fujitsu.com/package/en-search/>

# MB90820B Series

## 1.6 I/O Pins and Pin Functions

Table 1.6-1 lists the MB90820B series I/O pins and their functions. Table 1.7-1 lists the I/O circuit types. The letter in the “I/O circuit type” column in Table 1.6-1 refers to the letter in the “Type” column in Table 1.7-1.

### ■ I/O Pins and Pin Functions

Table 1.6-1 Pin Description (1 / 4)

Pin no.		Pin name	I/O circuit	Pin status during reset	Function
LQFP*1	QFP*2				
21, 22	23, 24	X0, X1	A	Oscillating	Oscillation input pins.
17	19	$\overline{\text{RST}}$	B	Reset input	External reset input pin.
59 to 54	61 to 56	P00 to P05	C	Port input	General-purpose I/O ports.
53	55	P06	C		General-purpose I/O ports.
		PWIO			PWC 0 signal input pin.
52	54	P07	C		General-purpose I/O ports.
		PWO0			PWC 0 signal output pin.
51	53	P10	D		General-purpose I/O ports.
		INT0			Can be used as interrupt request input channel 0. Input is enabled when 1 is set in EN0 in standby mode.
		DTTI			RTO0 to RTO5 pins for fixed-level input. This function is enabled when the waveform generator enables its input bits.
50 to 45	52 to 47	P11 to P16	D		General-purpose I/O ports.
		INT1 to INT6			Can be used as interrupt request input channels 1 to 6. Input is enabled when 1 is set in EN1 to EN6 in standby mode.
44	46	P17	D		General-purpose I/O ports.
43	45	P20	D		General-purpose I/O ports.
		TIN1			External clock input pin for reload timer 1.
42	44	P21	D		General-purpose I/O ports.
		TO1			Event output pin for reload timer 1.
41, 39 to 35	43, 41 to 37	P22 to P27	D		General-purpose I/O ports.
34 to 28	36 to 30	P30 to P36	E		General-purpose I/O ports.
27	29	P37	E		General-purpose I/O ports.
		PPG0			Output pins for PPG channel 0. This function is enabled when PPG channel 0 enables output.
26	28	P40	F		General-purpose I/O ports.
		PPG1			Output pins for PPG channel 1. This function is enabled when PPG channel 1 enables output.



**Table 1.6-1 Pin Description (2 / 4)**

Pin no.		Pin name	I/O circuit	Pin status during reset	Function
LQFP*1	QFP*2				
19	21	P41	F	Port Input	General-purpose I/O ports.
		TIN0			External clock input pin for reload timer 0.
18	20	P42	F		General-purpose I/O ports.
		TO0			Event output pin for reload timer 0.
16	18	P43	F		General-purpose I/O ports.
		SCK0			Serial clock I/O pin for UART0. This function is enabled when UART0 enables clock output.
15	17	P44	F		General-purpose I/O ports.
		SOT0			Serial data output pin for UART0. This function is enabled when UART 0 enables data output.
14	16	P45	G		General-purpose I/O ports.
		SIN0			Serial data input pin for UART0. While UART0 is operating for input, the input of this pin is used as required and must not be used for any other input. CMOS input can be selected by user program.
13	15	P46	F		General-purpose I/O ports.
		PWI1			PWC 1 signal input pin.
12	14	P47	F		General-purpose I/O ports.
		PWO1			PWC 1 signal output pin.
11	13	P50	F		General-purpose I/O ports.
		PPG2			Output pins for PPG channel 2. This function is enabled when PPG channel 2 enables output.
10	12	P51	F		General-purpose I/O ports.
		INT7			Usable as interrupt request input channel 7. Input is enabled when 1 is set in EN7 in standby mode.
9 to 2	11 to 4	P60 to P67	H	Analog input	General-purpose I/O ports.
		AN0 to AN7			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER0).
78, 77	80, 79	P70, P71	I		General-purpose I/O ports.
		DA0, DA1			D/A converter analog output pins. This function is enabled when D/A converter is enabled.
		AN8, AN9			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).

**MB90820B Series****Table 1.6-1 Pin Description (3 / 4)**

Pin no.		Pin name	I/O circuit	Pin status during reset	Function
LQFP*1	QFP*2				
76	78	P72	J	Analog input	General-purpose I/O ports.
		SIN1			Serial data input pin for UART1. While UART1 is operating for input, the input of this pin is used as required and must not be used for any other input. CMOS input can be selected by user program.
		AN10			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).
75	77	P73	K	Analog input	General-purpose I/O ports.
		SOT1			Serial data output pin for UART1. This function is enabled when UART1 enables data output.
		AN11			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).
74	76	P74	K	Analog input	General-purpose I/O ports.
		SCK1			Serial clock I/O pin for UART1. This function is enabled when UART1 enables clock output.
		AN12			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).
73	75	P75	K	Analog input	General-purpose I/O ports.
		FRCK			External clock input pin for free-run timer.
		AN13			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).
72, 71	74, 73	P76, P77	K	Analog input	General-purpose I/O ports.
		IN0, IN1			Trigger input pins for input capture channels 0 to 1. When input capture channels 0 to 1 are used for input operation, these pins are enabled as required and must not be used for any other input capture.
		AN14, AN15			A/D converter analog input pins. This function is enabled when the analog input is enabled (ADER1).
70, 69	72, 71	P80, P81	F	Port input	General-purpose I/O ports.
		IN2, IN3			Trigger input pins for input capture channels 2 to 3. When input capture channels 2 to 3 are used for input operation, these pins are enabled as required and must not be used for any other input capture.
68 to 63	70 to 65	P82 to P87	L	Port input	General-purpose I/O ports.
		RTO0 to RTO5			Waveform generator output pins. These pins output the waveforms specified at the waveform generator. Output is generated when waveform generator output is enabled.

**Table 1.6-1 Pin Description (4 / 4)**

Pin no.		Pin name	I/O circuit	Pin status during reset	Function
LQFP*1	QFP*2				
23	25	MD0	M	Mode input	Input pin for operation mode specification. Connect this pin directly to Vcc or Vss.
24, 25	26, 27	MD1, MD2	N		Input pins for operation mode specification. Connect these pins directly to Vcc or Vss.
80	2	AVCC	—	Power	Vcc power input pin for analog circuits.
79	1	AVR	—		Vref+ input pin for the A/D converter. This voltage must not exceed AVcc. Vref- is fixed to AVss.
1	3	AVSS	—		Vss power input pin for analog circuits.
20, 61	22, 63	Vss	—	Power	Power (0 V) input pin.
40, 60	42, 62	Vcc	—		Power (5 V) input pin.
62	64	C	—	—	Capacity pin for power stabilization. Please connect to an approximately 0.1 $\mu$ F ceramic capacitor.

\*1: FPT-80P-M21, FPT-80P-M22

\*2: FPT-80P-M06

\*3: See Section "1.7 I/O Circuit Types" for information on the circuit types.

## MB90820B Series

## 1.7 I/O Circuit Types

Table 1.7-1 summarize the I/O circuit types of MB90820B series

## ■ I/O Circuit Types

Table 1.7-1 I/O Circuit Type (1 / 4)

Type	Circuit	Remarks
A		<p>Main clock (main clock crystal oscillator)</p> <ul style="list-style-type: none"> <li>Oscillation feedback resistor of approximately 1 M<math>\Omega</math></li> </ul>
B		<ul style="list-style-type: none"> <li>Hysteresis input</li> <li>Resistor approximately 50 k<math>\Omega</math></li> </ul>
C		<ul style="list-style-type: none"> <li>CMOS output</li> <li>Hysteresis input</li> <li>Selectable pull-up resistor approximately 50 k<math>\Omega</math></li> <li><math>I_{OL} = 12</math> mA</li> </ul>
D		<ul style="list-style-type: none"> <li>CMOS output</li> <li>Hysteresis input</li> <li>Selectable pull-up resistor approximately 50 k<math>\Omega</math></li> <li><math>I_{OL} = 4</math> mA</li> </ul>

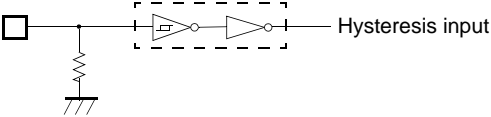
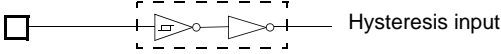
**Table 1.7-1 I/O Circuit Type (2 / 4)**

Type	Circuit	Remarks
E		<ul style="list-style-type: none"> <li>CMOS output</li> <li>CMOS input</li> <li>Selectable pull-up resistor approximately 50 k<math>\Omega</math></li> <li><math>I_{OL} = 4 \text{ mA}</math></li> </ul>
F		<ul style="list-style-type: none"> <li>CMOS output</li> <li>Hysteresis input</li> <li><math>I_{OL} = 4 \text{ mA}</math></li> </ul>
G		<ul style="list-style-type: none"> <li>CMOS output</li> <li>Hysteresis input</li> <li>CMOS input(selectable for UART 0 data input pin)</li> <li><math>I_{OL} = 4 \text{ mA}</math></li> </ul>
H		<ul style="list-style-type: none"> <li>CMOS output</li> <li>CMOS input</li> <li>Analog input</li> <li><math>I_{OL} = 4 \text{ mA}</math></li> </ul>

Table 1.7-1 I/O Circuit Type (3 / 4)

Type	Circuit	Remarks
I		<ul style="list-style-type: none"> <li>• CMOS output</li> <li>• CMOS input</li> <li>• Analog output</li> <li>• Analog input</li> <li>• <math>I_{OL} = 4 \text{ mA}</math></li> </ul>
J		<ul style="list-style-type: none"> <li>• CMOS output</li> <li>• Hysteresis input</li> <li>• CMOS input (selectable for UART1 data input pin)</li> <li>• <math>I_{OL} = 4 \text{ mA}</math></li> <li>• Analog input</li> </ul>
K		<ul style="list-style-type: none"> <li>• CMOS output</li> <li>• Hysteresis input</li> <li>• Analog input</li> <li>• <math>I_{OL} = 4 \text{ mA}</math></li> </ul>
L		<ul style="list-style-type: none"> <li>• CMOS output</li> <li>• Hysteresis input</li> <li>• <math>I_{OL} = 12 \text{ mA}</math></li> </ul>

Table 1.7-1 I/O Circuit Type (4 / 4)

Type	Circuit	Remarks
M		Mask ROM/evaluation product • Hysteresis input • Selectable pull-up resistor approximately 50 kΩ Flash ROM • CMOS input • No pull-down resistor
N		Mask ROM/evaluation product • Hysteresis input Flash ROM • CMOS input

# MB90820B Series

## 1.8 Notes on Handling Devices

---

When handling devices, pay special attention to the following items or procedures:

- Preventing latch-up
  - Stabilization of supply voltage
  - Notes on energization
  - Handling unused pins
  - Connection of unused pins of A/D converter and D/A converter
  - External clock
  - Power supply pin ( $V_{CC}/V_{SS}$ )
  - Turning-on sequence of A/D converter and D/A converter
  - Initialization
  - Return from standby state
- 

### ■ Notes on Handling Devices

- Be sure that the maximum rated voltage is not exceeded (latch-up prevention)

CMOS ICs may cause latch-up in the following situations:

- When a voltage higher than  $V_{CC}$  or lower than  $V_{SS}$  is applied to input or output pins.
- When a voltage exceeding the rating is applied between  $V_{CC}$  and  $V_{SS}$ .
- When the  $AV_{CC}$  power supply is applied before the  $V_{CC}$  voltage.

If latch-up occurs, the power supply current increases rapidly, sometimes resulting in thermal breakdown of the device. Use meticulous care not to exceed the rating.

For the same reason, also be careful not to let the analog power-supply voltage exceed the digital power-supply voltage.

- Stabilize the supply voltages

If the power supply voltage varies actually even within the operation assurance range of the  $V_{CC}$  power supply voltage, a malfunction may occur. The  $V_{CC}$  power supply voltage must therefore be stabilized.

As stabilization guidelines, stabilize the power supply voltage so that  $V_{CC}$  ripple fluctuations (peak to peak value) in the commercial frequencies (50 to 60 Hz) fall within 10% of the standard  $V_{CC}$  power supply voltage and transient fluctuation rate becomes 0.1 V/ms or less in instantaneous fluctuation for power supply switching.

- Notes on energization

To prevent the internal regulator circuit from malfunctioning, set the voltage rise time during energization at 50  $\mu$ s or more.



● Handling unused pins

Unused input pins left open may cause abnormal operations, or latch-up leading to permanent damage. Unused input pins should be pulled up or pulled down through at least 2 k $\Omega$  resistance.

Unused input/output pins may be left open in output state, but if such pins are in input state they should be handled in the same way as input pins.

If any output pins are unused, set them to open.

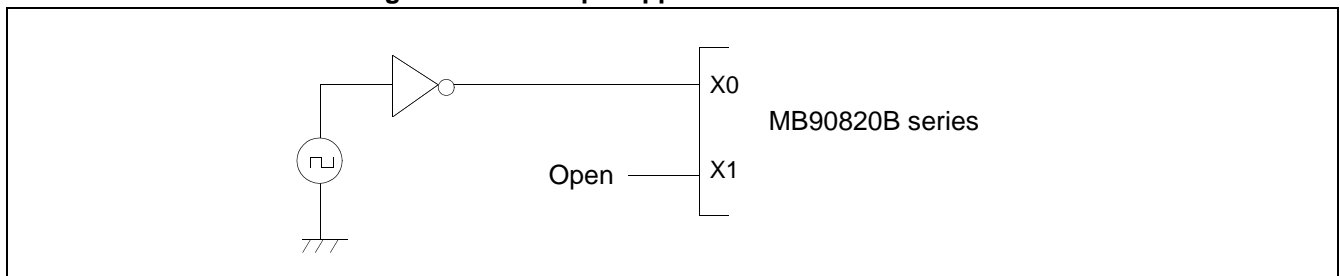
● Connection of unused pins of A/D converter and D/A converter

When the A/D converter and D/A converter are not used, connect the pins as follows:  $AV_{CC} = V_{CC}$ ,  $AV_{SS} = AVR = V_{SS}$ .

● Notes on external clock

To use an external clock, drive only the X0 pin and leave the X1 pin open (See the illustration below).

**Figure 1.8-1 Sample application of external clock**



● Power supply pins ( $V_{CC}/V_{SS}$ )

In products with multiple  $V_{CC}$  or  $V_{SS}$  pins, the pins of the same potential are internally connected in the device to avoid abnormal operations including latch-up. However, you must connect the pins to external power supply and a ground lines to lower the electro-magnetic emission level, to prevent abnormal operation of strobe signals caused by the rise in the ground level, and to conform to the total output current rating.

Moreover, connect the current supply source with the  $V_{CC}$  and  $V_{SS}$  pins of this device at the low impedance.

It is also advisable to connect a ceramic bypass capacitor of approximately 0.1  $\mu$ F between  $V_{CC}$  and  $V_{SS}$  near this device.

● Turning-on sequence of A/D converter and D/A converter

Make sure to turn on the A/D converter and D/A converter power supply ( $AV_{CC}$ ,  $AV_{SS}$ ,  $AVR$ ) and analog inputs ( $AN0$  to  $AN15$ ) after turning-on the digital power supply ( $V_{CC}$ ).

Turn-off the digital power after turning off the A/D converter and D/A converter supply and analog inputs. In this case, make sure that the voltage of  $AVR$  does not exceed  $AV_{CC}$ .

- Initialization

In the device, there are internal registers which are initialized only by a power-on reset. To initialize these registers turning on the power again.

- Return from standby state

If the power supply voltage goes below the standby RAM holding voltage in the standby state, the device may fail to return from the standby state. In this case, reset the device via the external reset pin to return to the normal state.



# CHAPTER 2

---

## CPU

**This chapter describes memory space for the MB90820B series.**

- 2.1 CPU
- 2.2 Memory Space
- 2.3 Memory Maps
- 2.4 Addressing
- 2.5 Memory Location of Multibyte Data
- 2.6 Registers
- 2.7 Dedicated Registers
- 2.8 General-purpose Registers
- 2.9 Prefix Codes

## **2.1 CPU**

---

The F<sup>2</sup>MC-16LX CPU is a 16-bit CPU designed for use in applications, such as welfare and mobile equipment, which require high-speed real-time processing. The instruction set of the F<sup>2</sup>MC-16LX was designed for controllers so that it can perform various types of control at high speed and efficiency.

The F<sup>2</sup>MC-16LX CPU process not only 16-bit data but also 32-bit data using a built-in 32-bit accumulator. Memory space, which can be extended up to 16M bytes, can be accessed in either linear or bank access mode. The instruction set inherits the AT architecture of F<sup>2</sup>MC-8L, and has additional instructions supporting high-level languages. In addition, it has an extended addressing mode, enhanced multiply/divide instructions and reinforced bit manipulation instructions. The features of the F<sup>2</sup>MC-16LX CPU are shown below:

---

### **■ CPU**

- Minimum instruction execution time: 42 ns (when the source oscillation is 4 MHz and the PLL clock is multiplied by 6)
- Maximum memory address space: 16M bytes. Access in linear or bank addressing mode
- Instruction set optimum for controller applications
  - Many data types (bit, byte, word and long word)
  - As many as 23 addressing modes
  - Enhanced high-precision arithmetic operation by a 32-bit accumulator
  - Enhanced signed multiply/divide instructions and RETI instruction function
- Enhanced interrupt function
  - Eight programmable priority levels
- Automatic transfer function independent on CPU
  - Extended intelligent I/O service using up to 16 channels
- Instruction set supporting high-level language (C) and multitasking
  - System stack pointer, instruction set symmetry and barrel shift instructions
- Increased execution speed: 4-byte instruction queue

---

**Note :**

The MB90820B series runs only in single-chip mode so only internal ROM and RAM and internal peripheral address space can be accessed.

---

# MB90820B Series

## 2.2 Memory Space

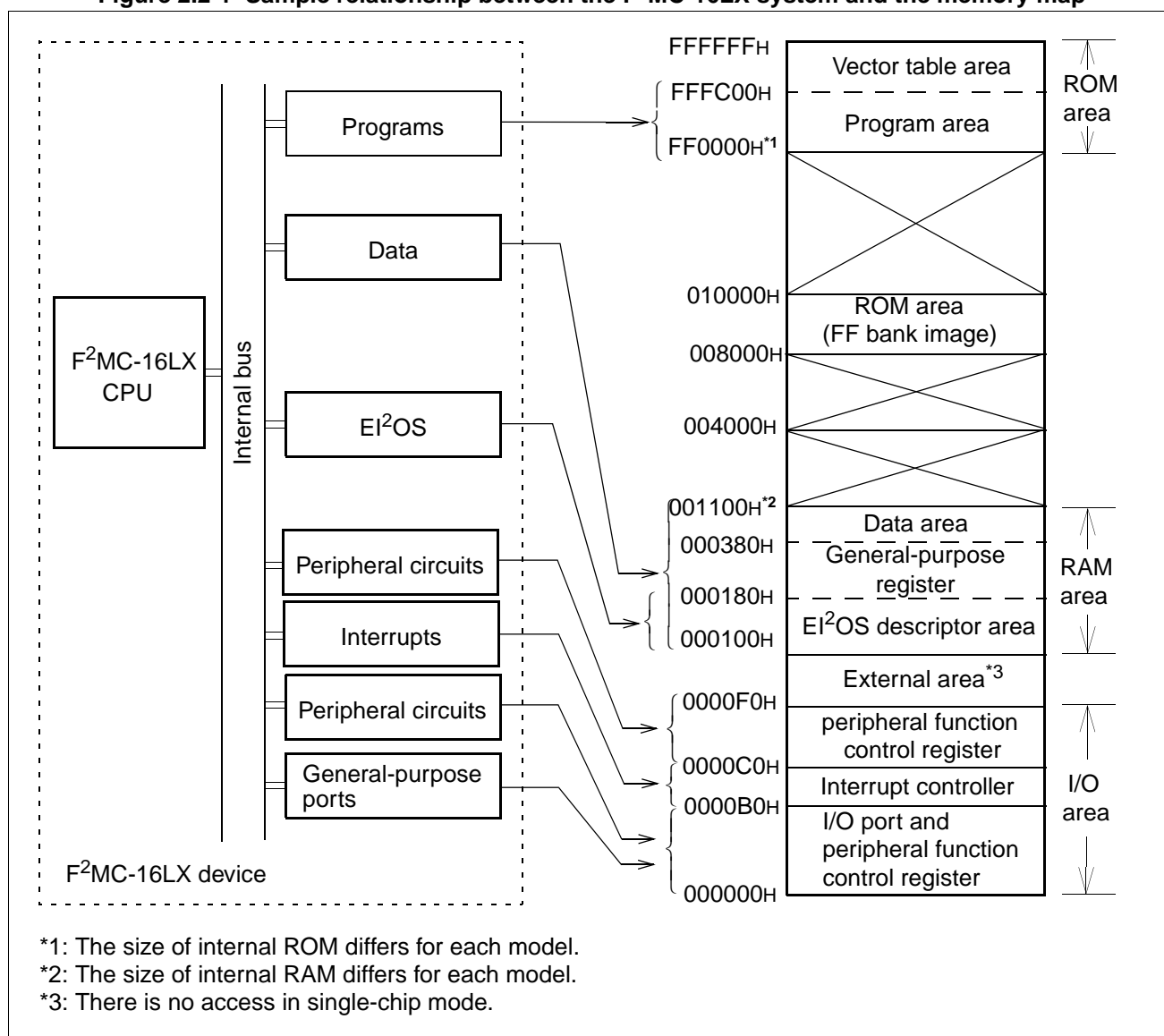
All I/O, programs and data are located in the 16-megabyte memory space of the F<sup>2</sup>MC-16LX. A part of the memory space is used for special purposes, such as extended intelligent I/O service (EI<sup>2</sup>OS) descriptors, general-purpose registers and vector tables.

### ■ Memory Space

All I/O, programs, and data are located in the 16-megabyte memory space of the F<sup>2</sup>MC-16LX CPU. The CPU is able to access each resource through an address indicated by the 24-bit address bus.

Figure 2.2-1 shows a sample relationship between the F<sup>2</sup>MC-16LX system and the memory map.

**Figure 2.2-1 Sample relationship between the F<sup>2</sup>MC-16LX system and the memory map**



## ■ ROM Area

- Vector table area (address: FFFC00<sub>H</sub> to FFFFFFF<sub>H</sub>)
  - This area is used as a vector table for vector call instructions, interrupt vectors and reset vectors.
  - This area is allocated at the highest addresses of the ROM area. The start address of the corresponding processing routine is set as data in each vector table address.
- Program area (address: up to FFFBFF<sub>H</sub>)
  - ROM is built in as an internal program area.
  - The size of internal ROM differs for each model.

## ■ RAM Area

- Data area (address: from 000100<sub>H</sub>)
  - The static RAM is built in as an internal data area.
  - The size of internal RAM differs for each model.
- General-purpose register area (address: 000180<sub>H</sub> to 00037F<sub>H</sub>)
  - Auxiliary registers used for 8-bit, 16-bit and 32-bit arithmetic operations and transfer are allocated in this area.
  - Since this area is allocated to a part of the RAM area, it can be used as ordinary RAM.
  - When this area is used as a general-purpose register, general-purpose register addressing enables high-speed access with short instructions.
- Extended intelligent I/O service (EI<sup>2</sup>OS) descriptor area (address: 000100<sub>H</sub> to 00017F<sub>H</sub>)
  - This area retains the transfer modes, I/O addresses, transfer count and buffer addresses.
  - Since this area is allocated to a part of the RAM area, it can be used as ordinary RAM.

## ■ I/O Area

- Interrupt control register area (address: 0000B0<sub>H</sub> to 0000BF<sub>H</sub>)

The interrupt control registers (ICR00 to ICR15) correspond to all peripheral functions that have an interrupt function. These registers set interrupt levels and control the extended intelligent I/O service (EI<sup>2</sup>OS).
- Peripheral function control register area (address: 000020<sub>H</sub> to 0000AF<sub>H</sub>, 0000C0<sub>H</sub> to 0000EF<sub>H</sub>)

These registers control the built-in peripheral functions, input and output data. Instruction using I/O addressing e.g. MOV A, io, is not supported for registers area 003FE0<sub>H</sub> to 003FFF<sub>H</sub>
- I/O port control register area (address: 000000<sub>H</sub> to 00001F<sub>H</sub>)

These registers controls I/O ports, input and output data.

# MB90820B Series

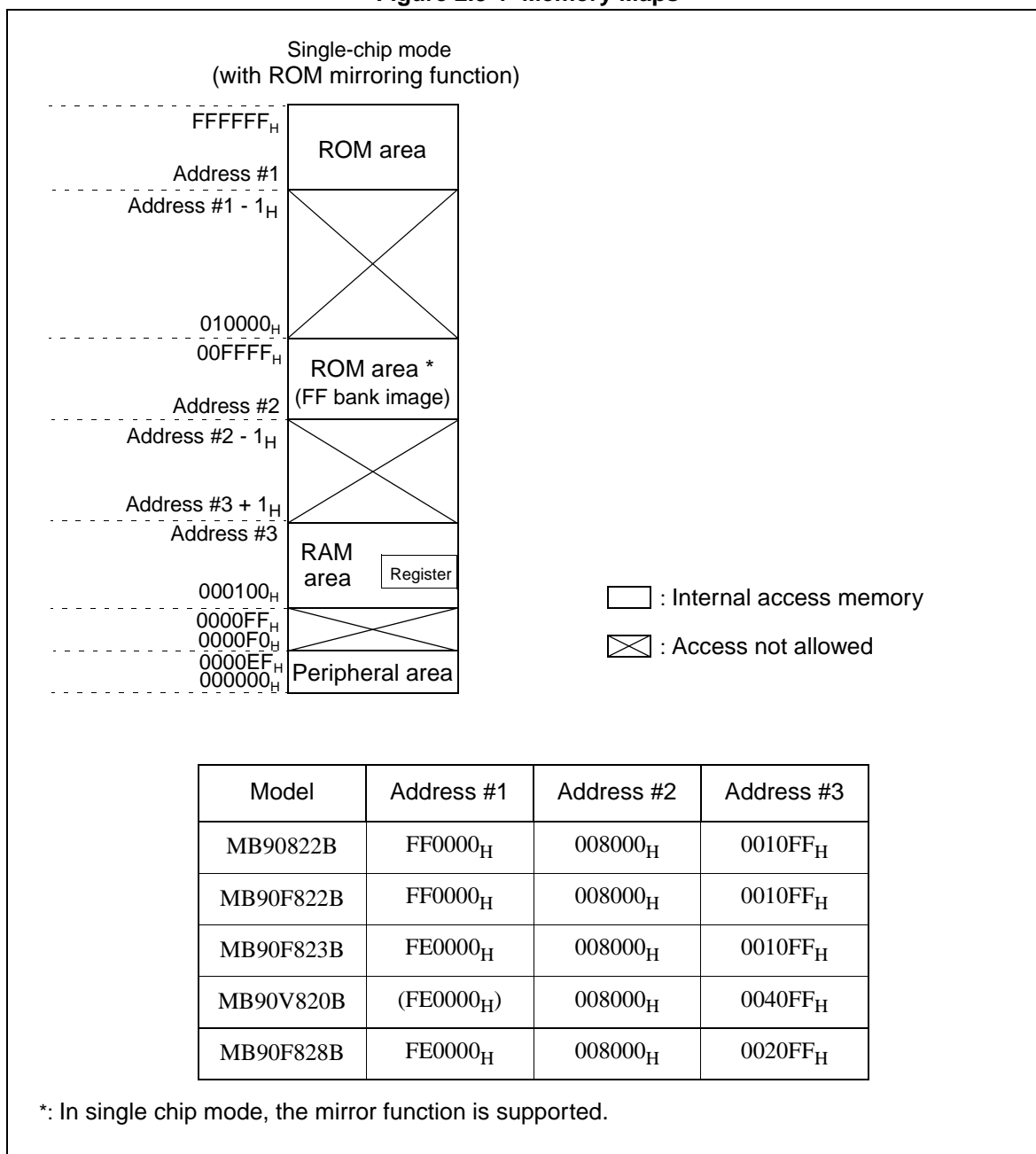
## 2.3 Memory Maps

This section shows the memory map for each MB90820B series model.

### ■ Memory Maps

Figure 2.3-1 shows the memory maps for the MB90820B series.

**Figure 2.3-1 Memory Maps**





---

Reference:

When the single-chip mode (without ROM mirroring function), see "CHAPTER 22 ROM MIRRORING FUNCTION SELECTION MODULE".

---

---

Note :

The ROM data of bank FF is reflected to the upper address of bank 00, realizing effective use of the C compiler small model. The lower 16-bit is assigned to the same address, enabling reference of the table on the ROM without stating "far". For example, if an attempt has been made to access 00C000<sub>H</sub>, the contents of the ROM at FFC000<sub>H</sub> are accessed actually. Since the ROM area of the FF bank exceeds 32K bytes, the whole area cannot be reflected in the image for the 00 bank. The ROM data at FF8000<sub>H</sub> to FFFFFFF<sub>H</sub> looks, therefore, as if it were the image for 008000<sub>H</sub> to 00FFFF<sub>H</sub>. Thus, it is recommended that the ROM data table be stored in the area of FF8000<sub>H</sub> to FFFFFFF<sub>H</sub>.

---

## MB90820B Series

### 2.4 Addressing

The methods for generating addresses are linear addressing and bank addressing. In linear addressing, the complete 24-bit address is specified directly by an instruction. In bank addressing, the upper 8 bits of the address are specified by a bank register for the required purpose, and the lower 16 bits of the address are specified by the instruction.

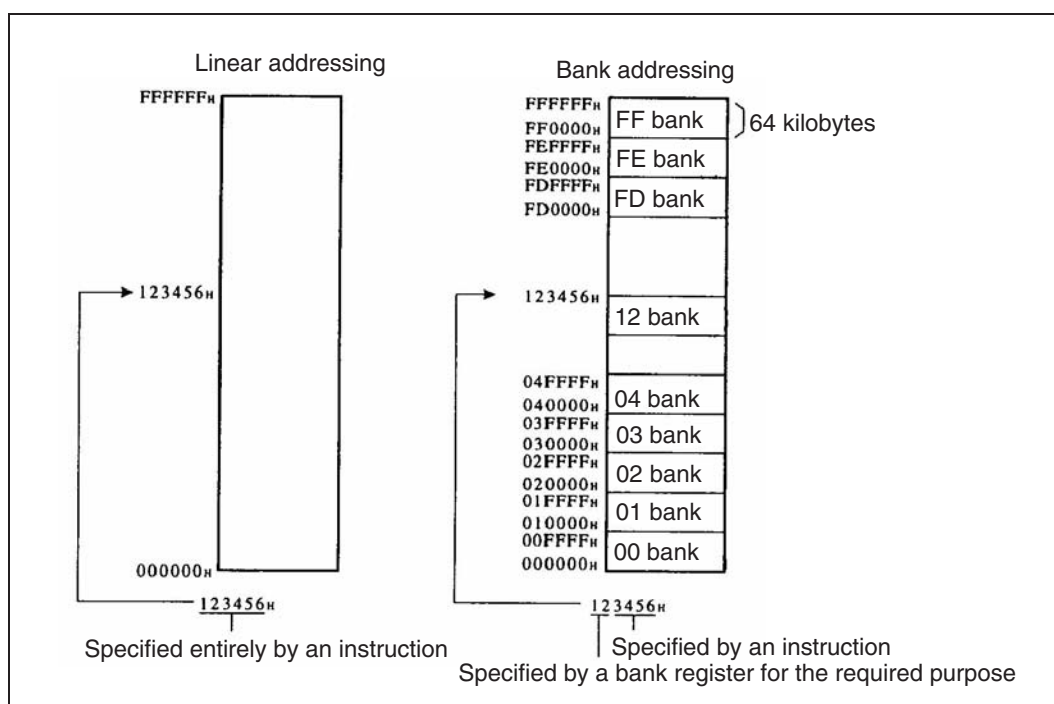
The F<sup>2</sup>MC-16LX series generally uses bank addressing.

#### ■ Linear Addressing and Bank Addressing

In linear addressing, the 16-megabyte space is accessed as consecutive address spaces. In bank addressing, the 16-megabyte space is divided into and managed as 256 64-kilobyte banks.

Figure 2.4-1 is an overview of linear addressing and bank addressing memory management.

**Figure 2.4-1 Linear addressing and bank addressing memory management**

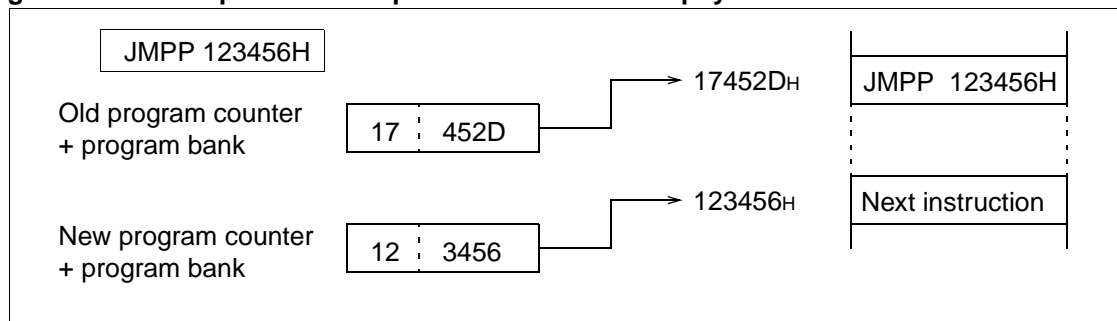


## 2.4.1 Address Specification by Linear Addressing

The two types of address specification by linear addressing are specification of a 24-bit address directly in the operand and specification of the lower 24 bits of a 32-bit general-purpose register.

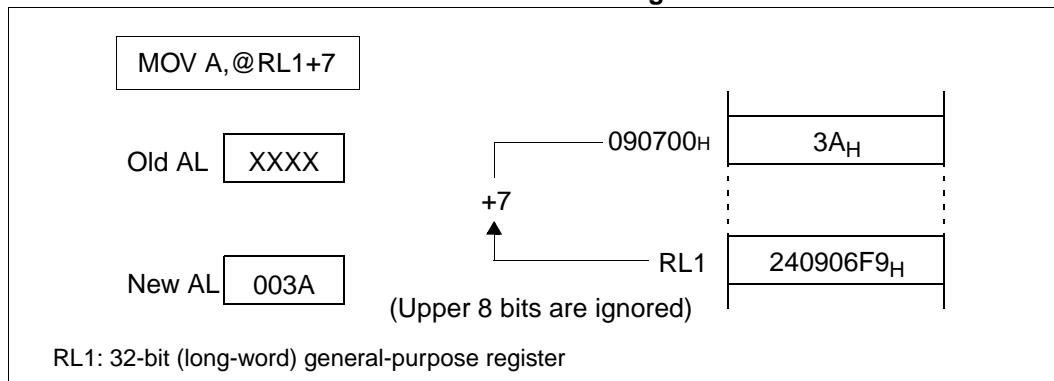
### ■ Linear Addressing by 24-bit Operand Specification

Figure 2.4-2 Example of direct specification of a 24-bit physical address in linear addressing



### ■ Addressing by Indirect Specification with a 32-bit Register

Figure 2.4-3 Example of indirect specification with a 32-bit general-purpose register in linear addressing



## MB90820B Series

### 2.4.2 Address Specification by Bank Addressing

In address specification by bank addressing, the 16-megabyte memory space is divided into 256 64-kilobyte banks. A bank address that corresponds to each space is specified in the bank register to determine the upper 8 bits of the address. The lower 16 bits of the address are specified by the instruction.

The five types of bank registers classified by function are as follows:

- Program bank register (PCB)
- Data bank register (DTB)
- User stack bank register (USB)
- System stack bank register (SSB)
- Additional bank register (ADB)

#### ■ Bank Registers and Access Space

Table 2.4-1 lists the access space and main function of each bank register.

**Table 2.4-1 Access space and main function of each bank register**

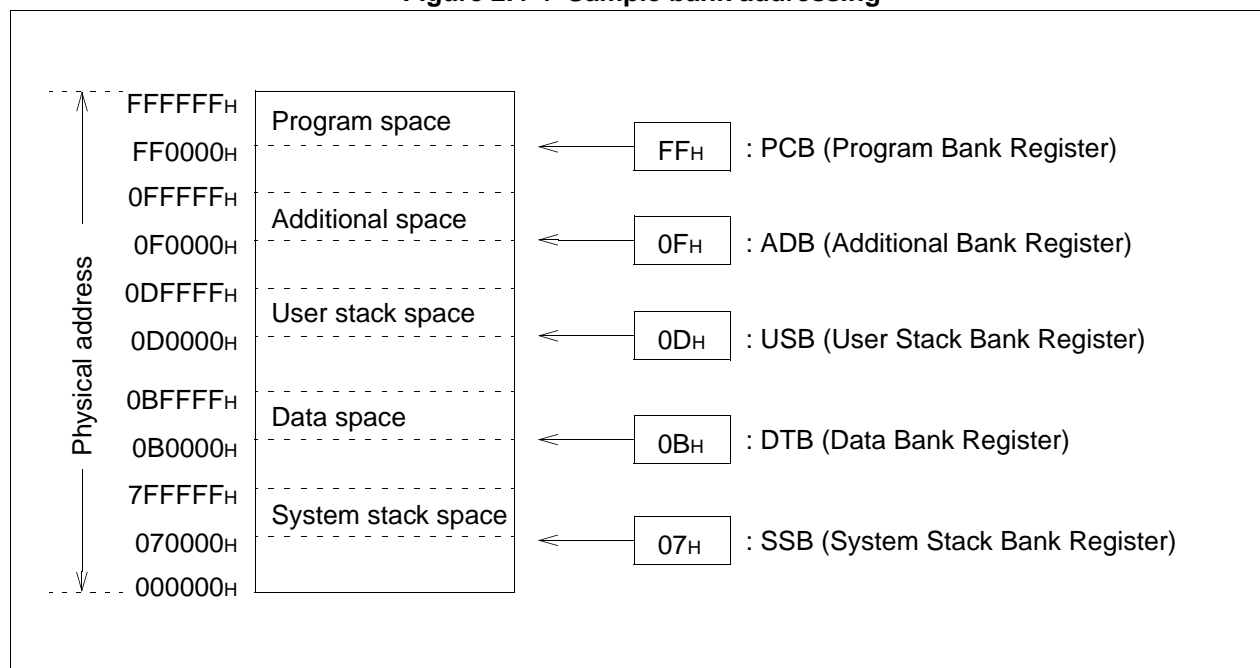
Bank register name	Access space	Main function	Initial value after a reset
Program bank register (PCB)	Program (PC) space	Instruction codes, vector tables and immediate-value data are stored.	FF <sub>H</sub>
Data bank register (DTB)	Data (DT) space	Read/write data is stored. Internal or external peripheral control registers and data registers are accessed.	00 <sub>H</sub>
User stack bank register (USB)	Stack (SP) space	This area is used for stack accesses such as when PUSH/POP instructions and interrupt registers are saved. The SSB is used when the stack flag in the condition register (CCR: S) is 1. The USB is used when the stack flag in the condition register (CCR: S) is 0. *	00 <sub>H</sub>
System stack bank register (SSB) *			00 <sub>H</sub>
Additional bank register (ADB)	Additional (AD) space	Data that overflows from the data (DT) space is stored.	00 <sub>H</sub>

\* : The SSB is always used as an interrupt stack.

Figure 2.4-4 shows the relationship between the memory space divisions and each register.

See Section "2.7.9 Bank Registers (PCB, DTB, USB, SSB, ADB)", for details.

**Figure 2.4-4 Sample bank addressing**



## ■ Bank Addressing and Default Space

To improve instruction coding efficiency, each instruction has a defined default space for each addressing method, as shown in Table 2.4-2. To use a space other than the default space, specify a prefix code for a bank before the instruction. This enables the bank space that corresponds to the specified prefix code to be accessed. See Section "2.9 Prefix Codes", for details about prefix codes.

**Table 2.4-2 Addressing and default spaces**

Default space	Addressing
Program space	PC indirect, program access, branching
Data space	Addressing using @RW0, @RW1, @RW4, and @RW5, @A, addr16, dir
Stack space	Addressing using PUSHW, POPW, @RW3, and @RW7
Additional space	Addressing using @RW2 and @RW6

# MB90820B Series

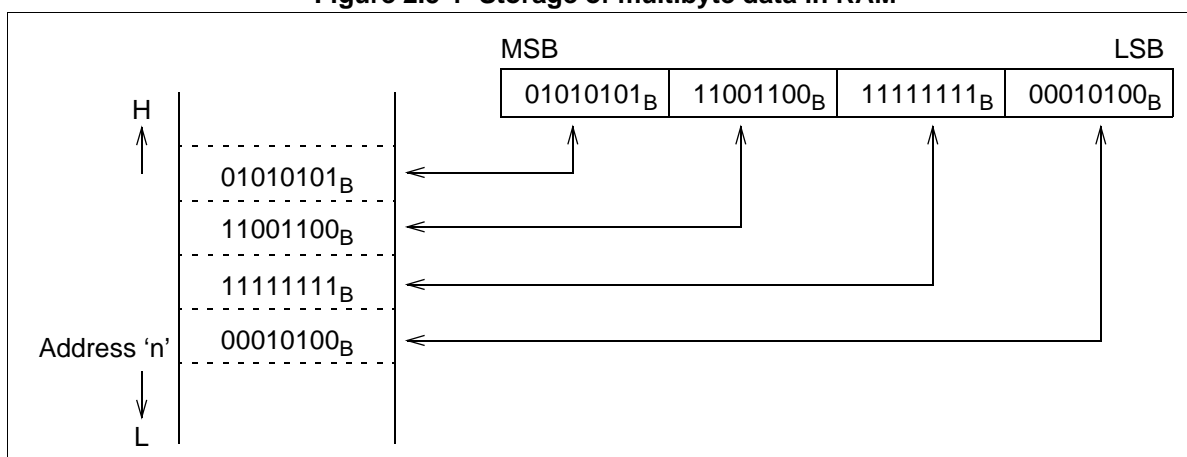
## 2.5 Memory Location of Multibyte Data

Multibyte data is written to memory sequentially from the lower address. If multibyte data is 32-bit data, the lower 16 bits are transferred followed by the upper 16 bits. If an external signal is input immediately after the low-order data is written, the high-order data may not be written.

### ■ Storage of Multibyte Data on Memory

Figure 2.5-1 shows the data configuration of multibyte data in memory. The lower 8 bits of the data is located at address  $n$ , and subsequent data is located at address  $n + 1$ , address  $n + 2$ , address  $n + 3$  and so on, in this sequence.

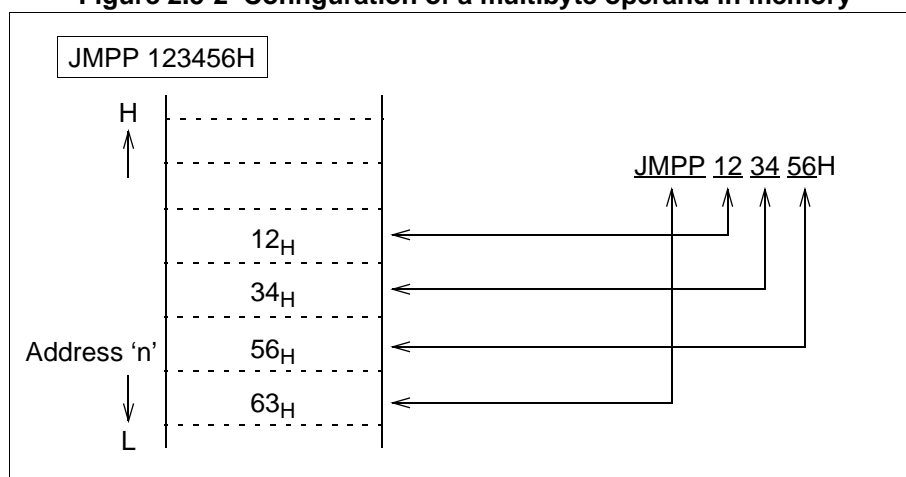
Figure 2.5-1 Storage of multibyte data in RAM



### ■ Storage of Multibyte Operand

Figure 2.5-2 shows the configuration of a multibyte operand in memory.

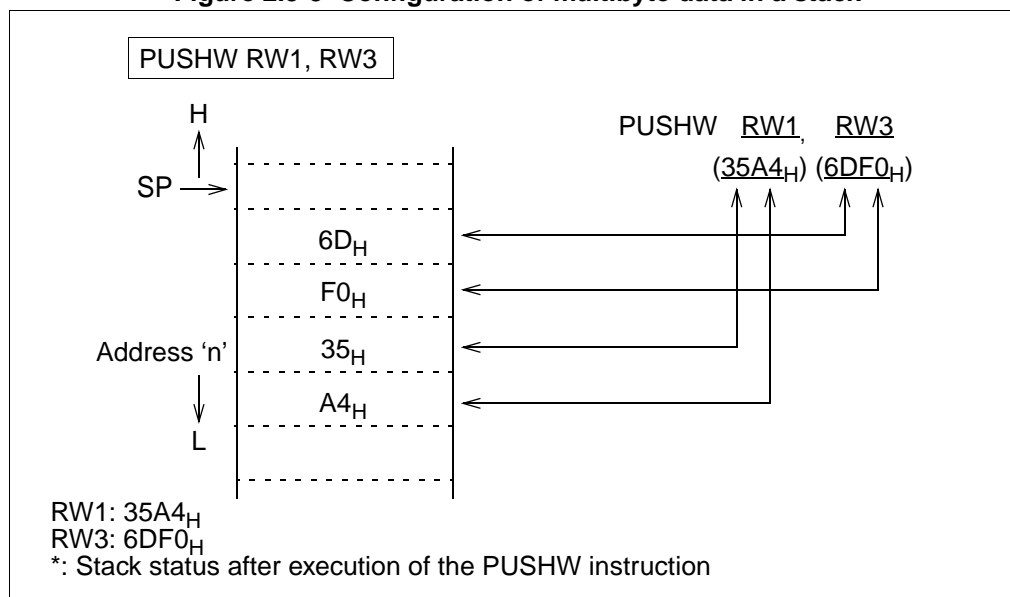
Figure 2.5-2 Configuration of a multibyte operand in memory



## ■ Storage of Multibyte Data in a Stack

Figure 2.5-3 shows the configuration of multibyte data in a stack.

Figure 2.5-3 Configuration of multibyte data in a stack

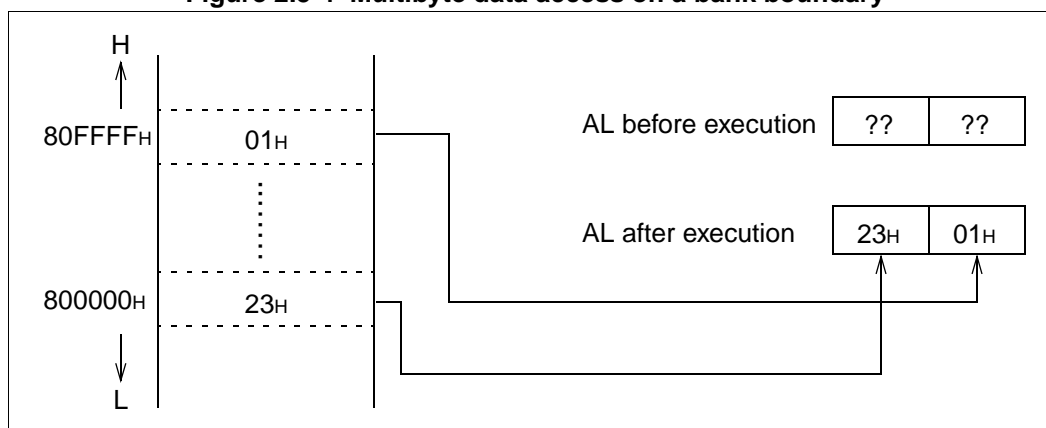


## ■ Multibyte Data Access

Accessing is generally performed within a bank. For an instruction that accesses to multibyte data, the address following `FFFFH` is `0000H` in the same bank.

Figure 2.5-4 shows an example of executing an instruction that accesses multibyte data on a bank boundary.

Figure 2.5-4 Multibyte data access on a bank boundary



## 2.6 Registers

F<sup>2</sup>MC-16LX registers are classified into internal dedicated CPU registers and built-in RAM general-purpose registers.

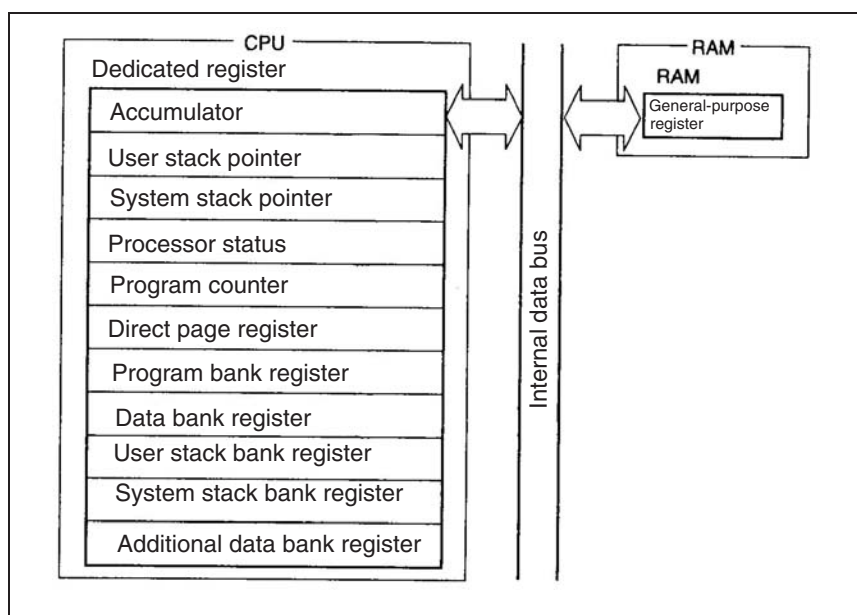
### ■ Dedicated Registers and General-purpose Registers

Dedicated registers are dedicated hardware inside the CPU with limited use in the CPU architecture.

General-purpose registers exist together with RAM in the CPU address space. Just like dedicated registers, general-purpose registers can be accessed without addressing. Just like ordinary memory, the user can specify how the register is used.

Figure 2.6-1 shows the location of the dedicated registers and general-purpose registers in the device.

**Figure 2.6-1 Dedicated registers and general-purpose registers**





## 2.7 Dedicated Registers

The following 11 registers are dedicated registers in the CPU.

- Accumulator (A)
- System stack pointer (SSP)
- Program counter (PC)
- Program bank register (PCB)
- User stack bank register (USB)
- Additional data bank register (ADB)
- User stack pointer (USP)
- Processor status (PS)
- Direct page register (DPR)
- Data bank register (DTB)
- System stack bank register (SSB)

### ■ Configuration of Dedicated Registers

Figure 2.7-1 shows the configuration of dedicated registers; Table 2.7-1 lists the initial values of the dedicated registers.

Figure 2.7-1 Configuration of dedicated registers

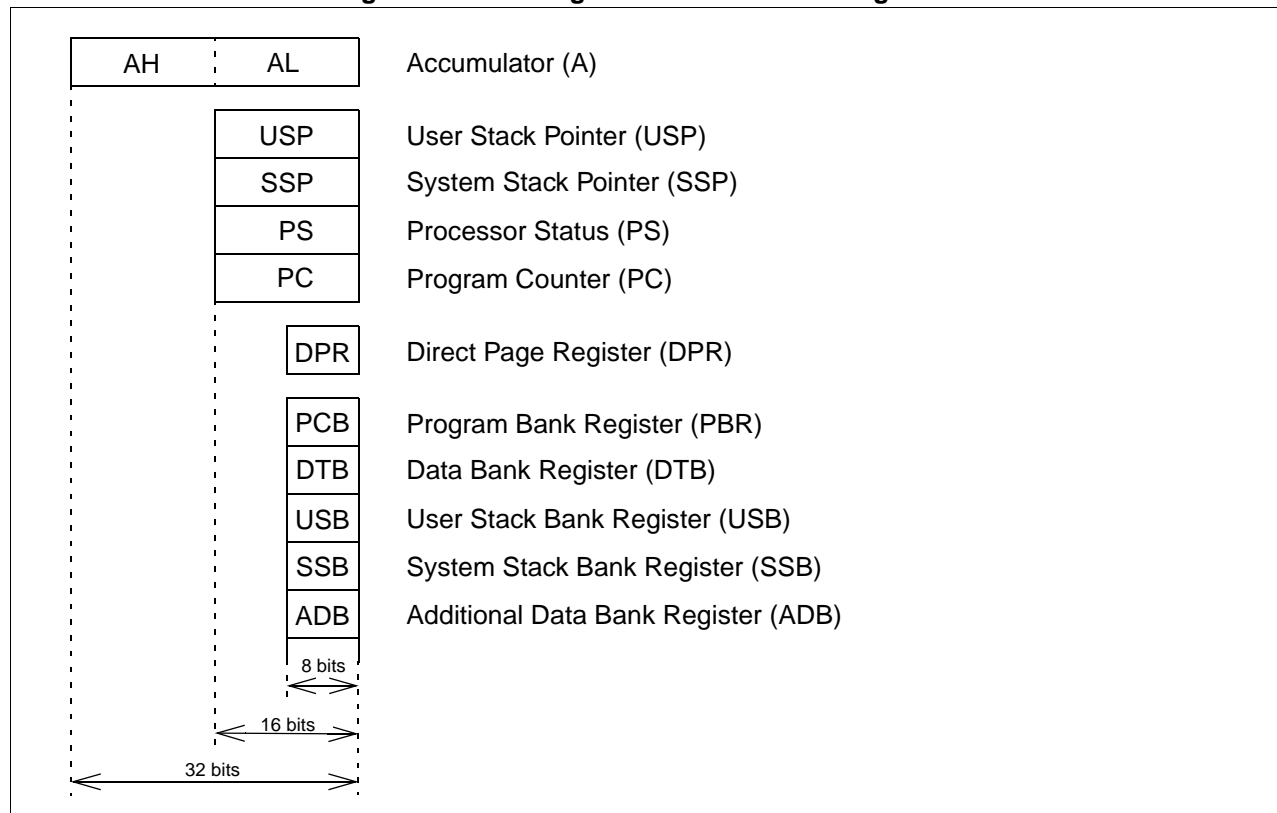


Table 2.7-1 Initial values of the dedicated registers

Dedicated register	Initial value
Accumulator (A)	Undefined
User stack pointer (USP)	Undefined
System stack pointer (SSP)	Undefined
Processor status (PS)	<div style="text-align: center;">           bit 15 <math>\longleftrightarrow</math> 13 12 <math>\longleftrightarrow</math> 8 7 <math>\longleftrightarrow</math> 0  <div style="display: flex; justify-content: center; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin: 0 5px;">PS</div> <div style="border: 1px solid black; padding: 2px; margin: 0 5px;">ILM</div> <div style="border: 1px solid black; padding: 2px; margin: 0 5px;">RP</div> <div style="border: 1px solid black; padding: 2px; margin: 0 5px;">CCR</div> </div>           Default value <math>\rightarrow</math>    000        00000        -01xxxxx         </div>
Program counter (PC)	Value in reset vector (Values of FFFFDC <sub>H</sub> , FFFFDD <sub>H</sub> )
Direct page register (DPR)	01 <sub>H</sub>
Program bank register (PCB)	Value in reset vector (Value of FFFFDE <sub>H</sub> )
Data bank register (DTB)	00 <sub>H</sub>
User stack bank register (USB)	00 <sub>H</sub>
System stack bank register (SSB)	00 <sub>H</sub>
Additional data bank register (ADB)	00 <sub>H</sub>

- : Not used

x: Undefined

**Note :**

The above initial values are the initial values for the device. They are different from the ICE (emulator, etc.) values.

## 2.7.1 Accumulator (A)

The accumulator (A) consists of two 16-bit arithmetic operation registers (AH and AL). The accumulator is used to temporarily store the results of an arithmetic operation and data. The A register can be used as a 32-bit, 16-bit or 8-bit register. Various arithmetic operations can be performed between memory and other registers or between the AH register and the AL register. The A register has a data retention function that automatically transfers pre-transfer data from the AL register to the AH register when data of word length or less is transferred to the AL register. (Data is not retained with some instructions.)

### ■ Accumulator (A)

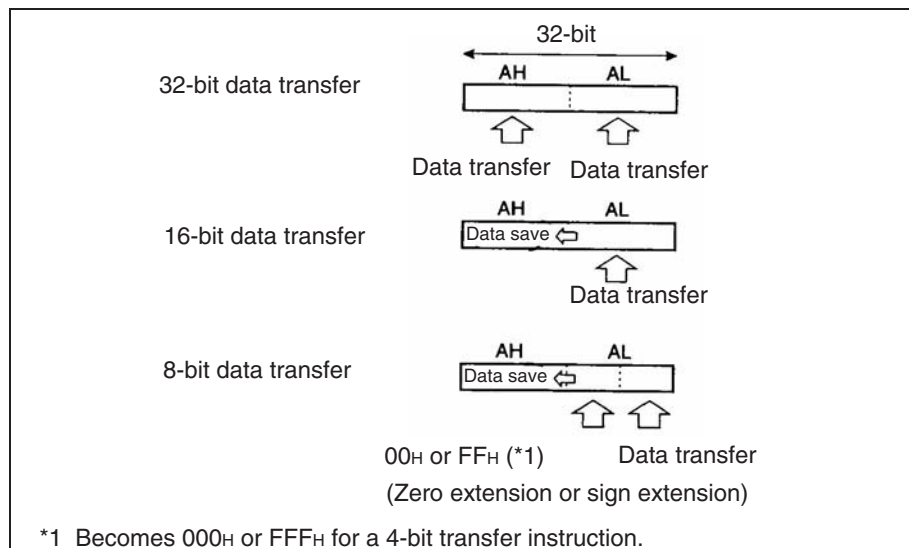
#### ● Data transfer to the accumulator

The accumulator can process 32-bit (long word), 16-bit (word) and 8-bit (byte) data. The 4-bit data transfer instruction (MOVN) is an exception. The explanation of 8-bit data also applies to 4-bit data.

- For 32-bit data processing, the AH register and AL register are combined.
- For 16-bit data and 8-bit data, only the AL register is used.
- When data of byte length or less is transferred to the AL register, data becomes 16 bits long by sign extension or zero extension, and is stored in the AL register. Data in the AL register can be handled as word-length or byte-length data.

Figure 2.7-2 shows data transfer to the accumulator. Figure 2.7-3 to Figure 2.7-6 show specific transfer examples.

Figure 2.7-2 Data transfer to the accumulator



**MOV A, 3000H** (An instruction that zero-extends the contents at address 3000<sub>H</sub> and stores the result in the AL register)

A before execution

XXXX <sub>H</sub>	2456 <sub>H</sub>
-------------------	-------------------

DTB B5<sub>H</sub>

A after execution

2456 <sub>H</sub>	0088 <sub>H</sub>
AH	AL

Memory space

MSB	Memory space		LSB
B53000 <sub>H</sub>	77 <sub>H</sub>	88 <sub>H</sub>	

**MOVX A, 3000H** (An instruction that stores the contents at address 3000<sub>H</sub> in the AL register)

A before execution: XXXX<sub>H</sub> | 2456<sub>H</sub>

DTB: B5<sub>H</sub>

A after execution: 2456<sub>H</sub> | FF88<sub>H</sub>  
 AH AL

Memory space: B53000<sub>H</sub> | 77<sub>H</sub> | 88<sub>H</sub>  
 MSB | | LSB

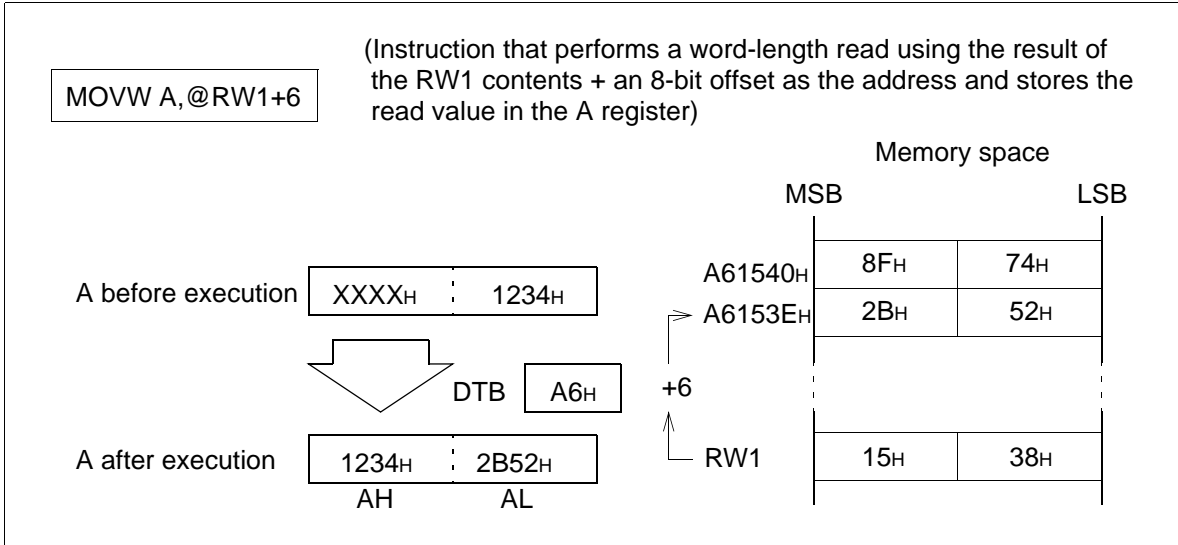
**MOVL A, @RW1+6**

(Instruction that perform a long-word-length read using the result of the RW1 contents + an 8-bit offset as the address and stores the read value in the A register)

The diagram illustrates the execution of the **MOVL A, @RW1+6** instruction. It shows the register **A** before and after execution, the data bus (DTB) with value **A6H**, and the memory space. The instruction performs a long-word read from memory address **A6153EH + 6**, resulting in the value **8F74H 2B52H** being stored in register **A**.

Memory space	
MSB	LSB
<b>A61540H</b>	8FH 74H
<b>A6153EH</b>	2BH 52H
...	
<b>RW1</b>	15H 38H

**Figure 2.7-6 Example of AL-AH transfer in the accumulator (A) (16 bits, register indirect)**



- Accumulator byte-processing arithmetic operation

When a byte-processing arithmetic operation instruction is executed for the AL register, the upper 8 bits of the AL register before the arithmetic operation is executed are ignored. The upper 8 bits of the arithmetic operation results are all zeros.

- Initial value of the accumulator

The initial value after a reset is undefined.

## MB90820B Series

### 2.7.2 Stack Pointers (USP, SSP)

There are two types of stack pointers: a user stack pointer (USP) and a system stack pointer (SSP). Each stack pointer is a register that indicates the memory address of the location of the destination for saved data or a return address when PUSH instructions, POP instructions and subroutines are executed. The upper 8 bits of the stack address (24 bits) are specified by the user stack bank register (USB) or system stack bank register (SSB).

When the S flag of the condition code register (CCR) is "0", the USP and USB registers are valid. When the S flag is "1", the SSP and SSB registers are valid.


#### ■ Stack Selection

The F<sup>2</sup>MC-16LX uses two types of stack: a system stack and a user stack.

The stack address is determined, as shown in Table 2.7-2, by the S flag in the processor status register (PS: CCR).

**Table 2.7-2 Stack address specification**

S flag	Stack address	
	Upper 8 bits	Lower 16 bits
0	User stack bank register (USB)	User stack pointer (USP)
1	System stack bank register (SSB)	System stack pointer (SSP)

 : Initial value

Because the S flag is initialized to "1" by a reset, the system stack is used as the default.

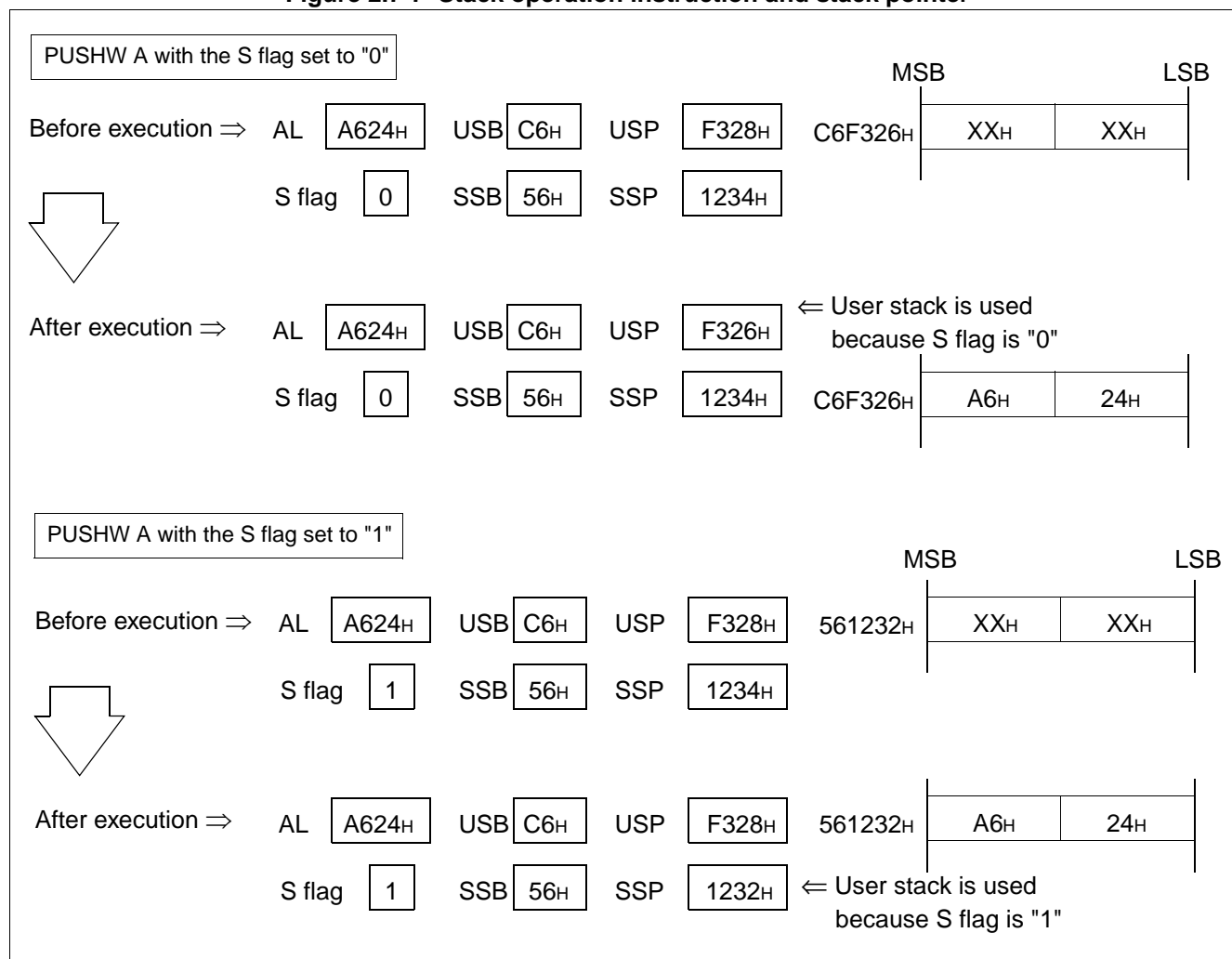
Ordinarily, the system stack is used for interrupt routine stack operations, and the user stack is used for all other types of stack operation. When separation of the stack space is not particularly necessary, only the system stack should be used.

#### Note :

Since the S flag is set to "1" when an interrupt is accepted, the system stack is always used for interrupts.

Figure 2.7-7 shows an example of stack operation instruction and stack pointer.

**Figure 2.7-7 Stack operation instruction and stack pointer**



References:

- To set the stack address for the stack pointer, generally use an even-numbered address. If an odd-numbered address is used, a word access is split into two parts, lowering efficiency.
- The initial values of the USP register and SSP register after a reset are undefined.

## MB90820B Series

### ■ System Stack Pointer (SSP)

To use the system stack pointer (SSP), set the S flag in the condition code register (CCR) of the processor status (PS) to "1". The upper 8 bits of the address that will be used for the stack operation are indicated by the system stack bank register (SSB).

### ■ User Stack Pointer (USP)

To use the user stack pointer (USP), set the S flag in the condition code register (CCR) of the processor status (PS) to "0". The upper 8 bits of the address that will be used for the stack operation are indicated by the user stack bank register (USB).



## 2.7.3 Processor Status (PS)

The processor status (PS) consists of CPU control bits and bits that indicate the CPU status. The PS register consists of the following three registers:

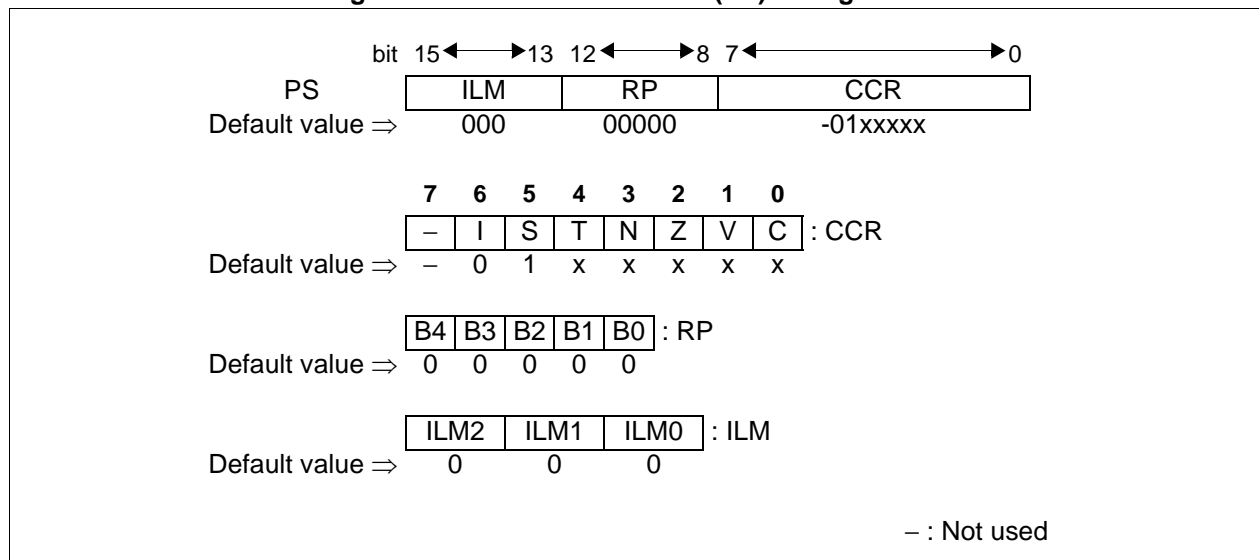
- Interrupt level mask register (ILM)
- Register bank pointer (RP)
- Condition code register (CCR)

### ■ Processor Status (PS) Configuration

The processor status (PS) consists of CPU control bits and bits that indicate the CPU status.

Figure 2.7-8 shows the configuration of the processor status (PS).

Figure 2.7-8 Processor status (PS) configuration



#### ● Interrupt level mask register (ILM)

This register indicates the level of the interrupt currently accepted by the CPU. The value is compared with the value of the interrupt level setting bits (ICR: IL0 to IL2) in the interrupt control register set for the peripheral resource interrupt request.

#### ● Register bank pointer (RP)

This pointer points to the first address of the memory block (register bank) used as the general-purpose register in the RAM area.

There are 32 banks for general-purpose registers. Values 0 to 31 are set in the RP to specify a bank.

#### ● Condition code register (CCR)

This register consists of flags that are set to "1" or reset to "0" by instruction execution results and by interrupt outputs.

## MB90820B Series

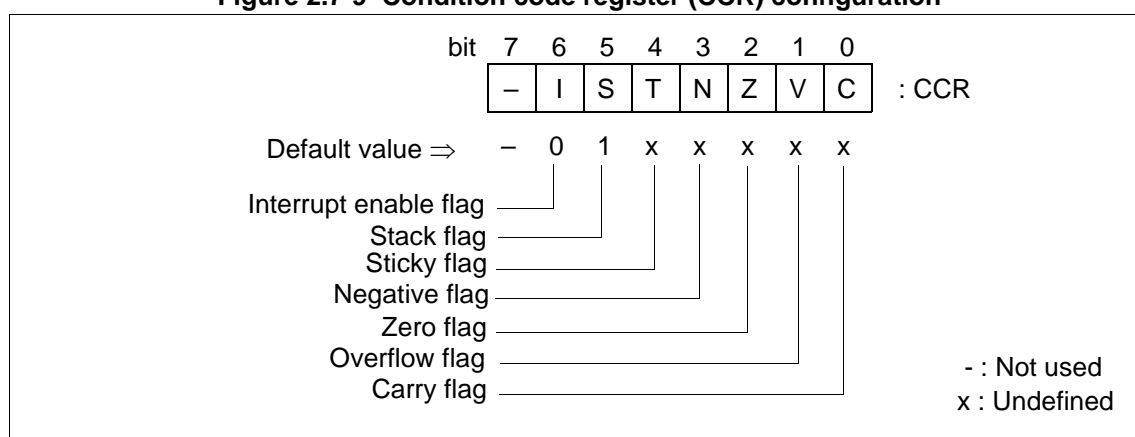
### 2.7.4 Condition Code Register (PS: CCR)

The condition code register (CCR) is an 8-bit register that consists of the bits that indicate the results of an arithmetic operation and the contents of transfer data and bits that control interrupt request acceptance.

#### ■ Condition Code Register (CCR) Configuration

Refer to the "F<sup>2</sup>MC-16LX Family Programming Manual" for details about the status of the condition code register (CCR) during instruction execution. Figure 2.7-9 shows the configuration of the CCR register.

**Figure 2.7-9 Condition code register (CCR) configuration**



#### ● Interrupt enable flag (I)

In response to all interrupt requests other than software interrupts, when the I flag is "1", interrupt requests are enabled. When the I flag is "0", interrupt requests are disabled. Cleared to "0" by a reset.

#### ● Stack flag (S)

This flag indicates the pointer used for a stack operation.

When the S flag is "0", the user stack pointer (USP) is valid. When the S flag is "1", the system stack pointer (SSP) is valid. Set to "1" when an interrupt is accepted or when a reset occurs.

#### ● Sticky bit flag (T)

Set to "1" when the data shifted out by the carry contains at least one "1" during execution of a logical right shift instruction or arithmetic right shift instruction. Otherwise, set to "0". Also set to "0" when the shift amount is zero.

#### ● Negative flag (N)

Set to "1" when the MSB is "1" as the result of an arithmetic calculation. Cleared to "0" when the MSB is "0".

● Zero flag (Z)

Set to "1" when the result of an arithmetic calculation is all zeros. Otherwise, set to "0".

● Overflow flag (V)

Set to "1" if a signed numeric value overflows because of an arithmetic calculation. Cleared to "0" if no overflow occurs.

● Carry flag (C)

Set to "1" when there is an overflow from the MSB or an underflow from the LSB because of an arithmetic calculation. Cleared to "0" when there is no overflow or underflow because of an arithmetic calculation.

## MB90820B Series

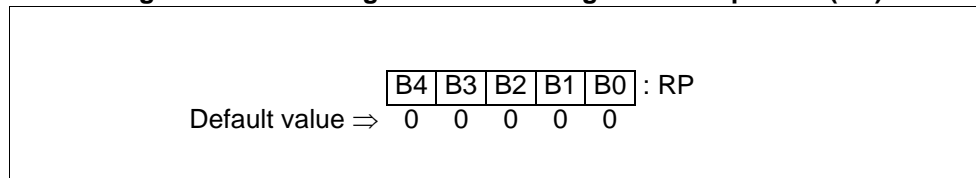
### 2.7.5 Register Bank Pointer (PS: RP)

The register bank pointer (RP) is a register that indicates the first address of the general-purpose register bank currently being used. The RP is used for real address conversion when general-purpose register addressing is used.

#### ■ Register Bank Pointer (RP)

Figure 2.7-10 shows the configuration of the register bank pointer (RP) register.

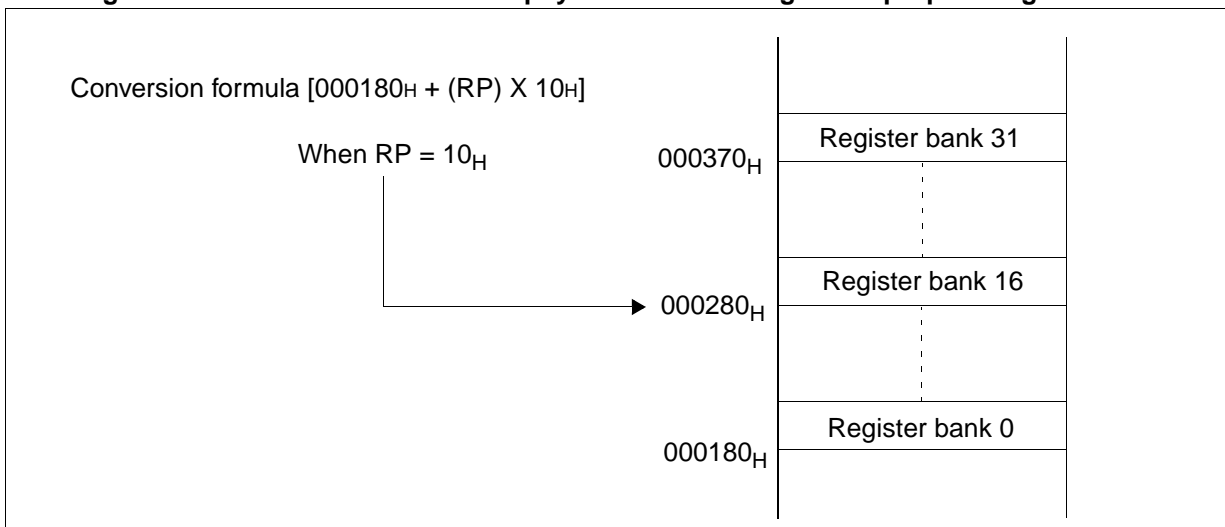
**Figure 2.7-10 Configuration of the register bank pointer (RP)**



#### ■ General-purpose Register Area and Register Bank Pointer (RP)

The register bank pointer points to the relationship between the general-purpose register of the F<sup>2</sup>MC-16LX and the address in internal RAM where the general-purpose register exists. Figure 2.7-11 shows the conversion rules used for the relationship between the contents of the RP and the real address.

**Figure 2.7-11 Conversion rules for physical address of general-purpose register area**



- Since the RP takes a value from 00<sub>H</sub> to 1F<sub>H</sub>, the first address of the register bank can be set in the range from 000180<sub>H</sub> to 00037<sub>H</sub>.
- Although an assembler instruction can use an 8-bit immediate value transfer instruction for transfer to the RP, in actuality only the lower 5 bits of the data are used.
- The initial value of the RP register after a reset is 00<sub>H</sub>.

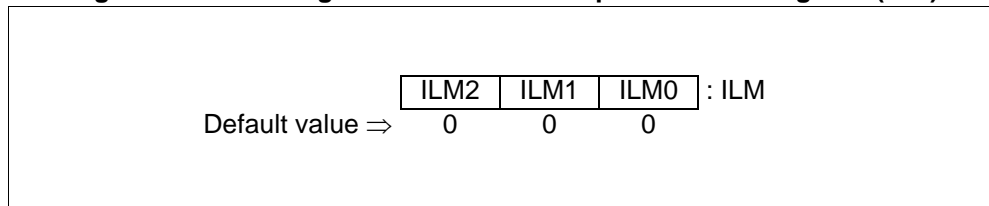
## 2.7.6 Interrupt Level Mask Register (PS: ILM)

The interrupt level mask register (ILM) is a 3-bit register that indicates the level of the interrupt currently accepted by the CPU.

### ■ Interrupt Level Mask Register (ILM)

Figure 2.7-12 shows the configuration of the interrupt level mask register (ILM). See "CHAPTER 7 INTERRUPT", for details about interrupts.

**Figure 2.7-12 Configuration of the interrupt level mask register (ILM)**



The interrupt level mask register (ILM) indicates the level of the interrupt currently accepted by the CPU. The level is compared with the value of the IL0 to IL2 bits of the interrupt control register (ICR00 to ICR15) set according to the interrupt request from the peripheral function. If the interrupt enable flag has been set to enable (CCR: I = 1), the CPU processes the instruction only when the value (interrupt level) of the interrupt request is smaller than the value indicated by these bits.

- When an interrupt is accepted, the interrupt level value is set in the interrupt level mask register (ILM). Thereafter, interrupts with the same or lower level are not accepted.
- The interrupt level is set to the highest level, which is the interrupts disabled status, because the interrupt level mask register (ILM) is initialized to all 0's by a reset.
- Although an assembler instruction can use an 8-bit immediate value transfer instruction for transfer to the interrupt level mask register (ILM), only the lower 3 bits of the data are enabled.

**Table 2.7-3 Interrupt level mask register (ILM) and interrupt level priority**

ILM2	ILM1	ILM0	Interrupt level	Interrupt level priority
0	0	0	0	<div style="text-align: center;"> Highest (interrupts disabled)  ↑  ↓  Lowest </div>
0	0	1	1	
0	1	0	2	
0	1	1	3	
1	0	0	4	
1	0	1	5	
1	1	0	6	
1	1	1	7	

## MB90820B Series

### 2.7.7 Program Counter (PC)

The program counter (PC) is a 16-bit counter that indicates the lower 16 bits of the memory address of the next instruction code to be executed by the CPU.

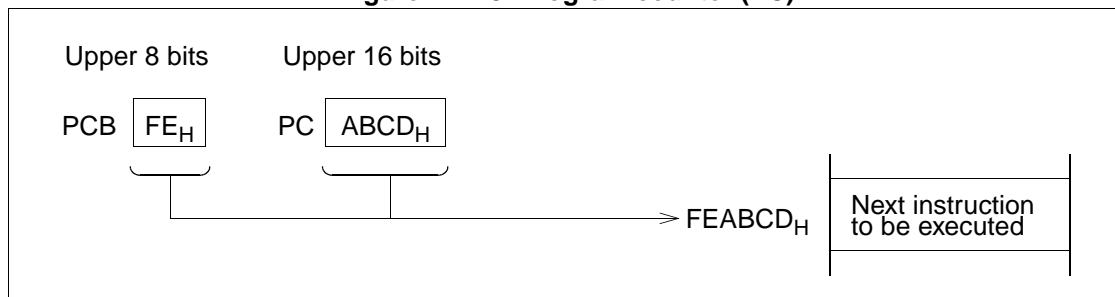
#### ■ Program Counter (PC)

The program bank register (PCB) specifies the upper 8 bits of the address where the next instruction code to be executed by the CPU is stored. The PC specifies the lower 16 bits. Before being used, the actual address is combined to become 24 bits, as shown in Figure 2.7-13.

The contents of the PC are updated by conditional branch instructions, subroutine call instructions, interrupts and resets.

The PC can be used as a bus pointer for reading operands.

**Figure 2.7-13 Program counter (PC)**



Note :

The PC and PCB cannot be rewritten directly by a program (such as by MOV PC and #FF).

2.7.8 Direct Page Register (DPR)

The direct page register (DPR) is an 8-bit register that specifies bits 8 to 15 (addr8 to addr15) of the operand address when a short direct addressing instruction is executed.

■ Direct Page Register (DPR)

As shown in Figure 2.7-14 , the DPR specifies bits 8 to 15 (addr8 to addr15) of the operand address when a short direct addressing instruction is executed. The DPR is 8-bits length. The DPR is initialized to 01<sub>H</sub> by a reset. The DPR can be read and written using an instruction.

Figure 2.7-14 Physical address generation by the direct page register (DPR)

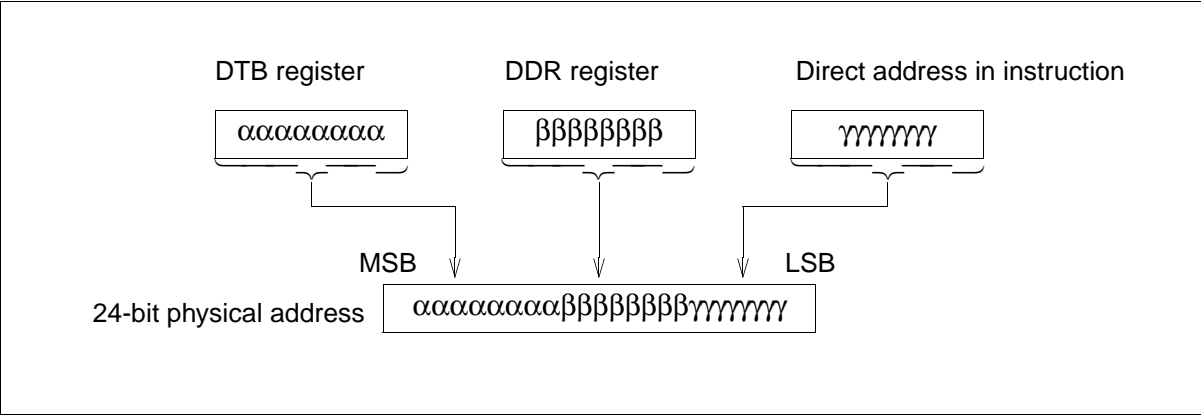
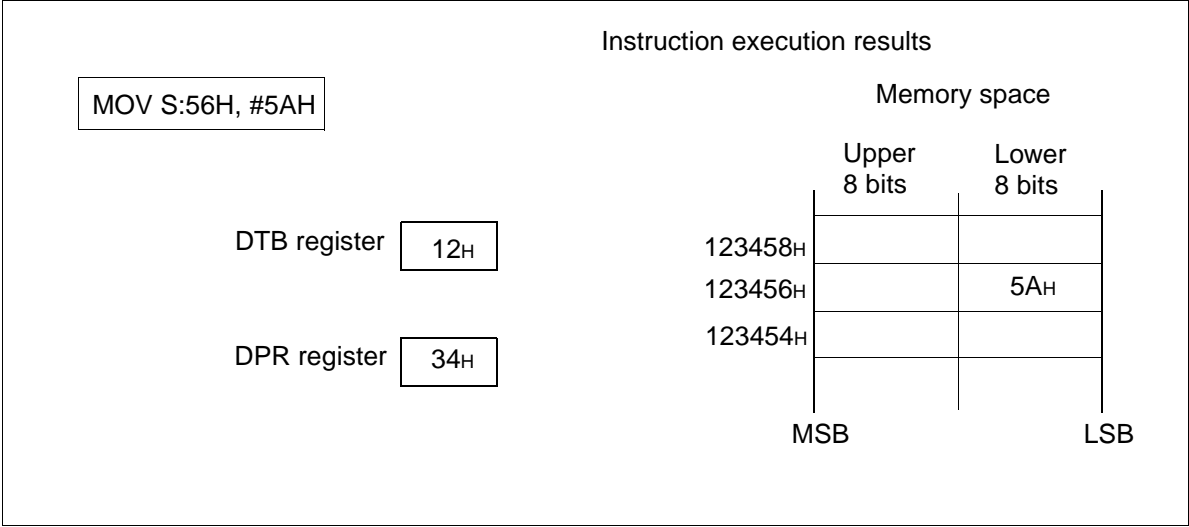


Figure 2.7-15 shows an example of direct page register (DPR) setting and data access.

Figure 2.7-15 Example of direct page register (DPR) setting and data access



## MB90820B Series

### 2.7.9 Bank Registers (PCB, DTB, USB, SSB, ADB)

---

Bank registers specify the highest 8-bit address by bank addressing. The five bank registers are as follows:

- Program bank register (PCB)
- Data bank register (DTB)
- User stack bank register (USB)
- System stack bank register (SSB)
- Additional bank register (ADB)

The PCB, DTB, USB, SSB and ADB registers indicate the individual memory banks where the program space, data space, user stack space, system stack space and additional space are located.

---

#### ■ Bank Registers (PCB, DTB, USB, SSB, ADB)

##### ● Program bank register (PCB)

The PCB is a bank register that specifies the program (PC) space.

The PCB is rewritten when a software interrupt instruction is executed, when the JMPP, CALLP, RETP and RETI instructions that branch anywhere within the 16-megabyte space are executed, or when a hardware interrupt or exception occurs.

##### ● Data bank register (DTB)

The DTB is a bank register that specifies the data (DT) space.

##### ● User stack bank register (USB), system stack bank register (SSB)

The USB and SSB are bank registers that specify the stack (SP) space. Whether the USB or the SSB is used depends on the S flag value in the processor status (PS: CCR). See Section "2.7.2 Stack Pointers (USP, SSP)", for details.

##### ● Additional bank register (ADB)

The ADB is a bank register that specifies the additional (AD) space.

##### ● Bank setting and data access

All bank registers are byte length. The PCB is initialized to FF<sub>H</sub> by a reset. The other bank registers are initialized to 00<sub>H</sub> by a reset. The PCB can be read, but cannot be written to.

The other bank registers can be read and written to.

---

#### Note :

The MB90820B series supports up to the memory space contained in the device.

See Section "2.4.2 Address Specification by Bank Addressing", for the operation of each register.

---



## 2.8 General-purpose Registers

The general-purpose registers are a memory block allocated in RAM at 000180<sub>H</sub> to 00037F<sub>H</sub> as banks, each of which consists of eight 16-bit segments.

The general-purpose registers can be used as general-purpose 8-bit registers (byte registers R0 to R7), 16-bit registers (word registers RW0 to RW7) or 32-bit registers (long-word registers RL0 to RL3).

General-purpose registers can access RAM with a short instruction at high speed. Since general-purpose registers are blocked into register banks, protection of register contents and division into function units can readily be performed. When a general-purpose register is used as a long-word register, it can be used as a linear pointer that directly accesses the entire space.

### ■ Configuration of a General-purpose Register

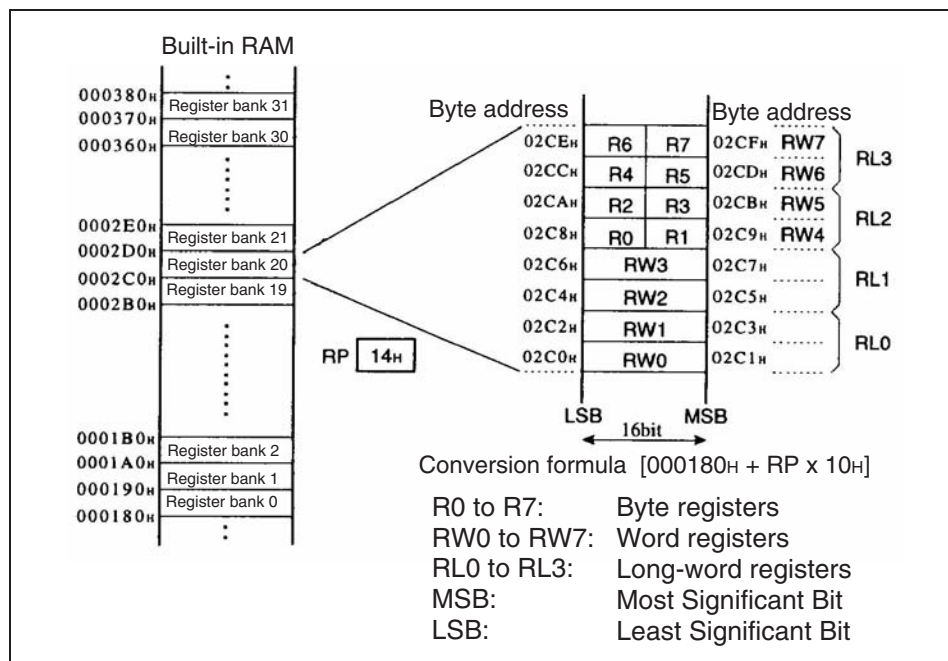
All general-purpose registers exist in RAM at 000180<sub>H</sub> to 00037F<sub>H</sub> and are configured as 32 banks. The register bank pointer (RP) specifies the bank that is to be used for a general-purpose register. The RP points to the bank currently being used.

The RP determines the first address of each bank with the following formula:

$$\text{Address of first general-purpose register} = 000180_{\text{H}} + \text{RP} \times 10_{\text{H}}$$

Figure 2.8-1 shows the location and configuration of the general-purpose register banks in the memory space.

Figure 2.8-1 Location and configuration of the general-purpose register banks in the memory space



---

Note :

The register bank pointer (RP) is initialized to 00<sub>H</sub> after a reset.

---

## ■ Register Bank

A register bank can be used as general-purpose registers (byte registers R0 to R7, word registers RW0 to RW7, long-word registers RL0 to RL3) for various arithmetic operations and pointers. A long-word register can be used as a linear pointer that directly accesses the entire memory space.

The contents of the register bank, like ordinary RAM, are not initialized by a reset. The status before a reset is retained. At power-on reset, however, the contents are undefined.

Table 2.8-1 lists the typical functions of general-purpose registers.

**Table 2.8-1 Typical functions of general-purpose registers**

Register name	Function
R0 to R7	Used as an operand in various instructions <b>Note :</b> R0 is also used as a barrel shift counter and an instruction normalization counter
RW0 to RW7	Used as a pointer Used as an operand in various instructions <b>Note :</b> RW0 is used also as a string instruction counter
RL0 to RL3	Used as a long pointer Used as an operand in various instructions

## **2.9 Prefix Codes**

---

**Prefix codes are placed before an instruction to partially change the operation of the instruction. The three types of prefix codes are as follows:**

- **Bank select prefix (PCB, DTB, ADB, SPB)**
  - **Common register bank prefix (CMR)**
  - **Flag change suppression prefix (NCC)**
- 

### **■ Prefix Codes**

#### ● Bank select prefix (PCB, DTB, ADB, SPB)

A bank select prefix is placed before an instruction to select the memory space to be accessed by the instruction regardless of the addressing method.

#### ● Common register bank prefix (CMR)

The common register bank prefix is placed before an instruction that accesses a register bank to change the register accessed by the instruction to the common bank (register bank selected when  $RP = 00_H$ ) at  $000180_H$  to  $00018F_H$  regardless of the current register bank pointer (RP) value.

#### ● Flag change suppression prefix (NCC)

The flag change suppression prefix code is placed before an instruction to suppress a flag change accompanying the execution of the instruction.

**MB90820B Series****2.9.1 Bank Select Prefix (PCB, DTB, ADB, SPB)**

A bank select prefix is placed before an instruction to select the memory space accessed by the instruction regardless of the addressing method.

**■ Bank Select Prefixes (PCB, DTB, ADB, SPB)**

The memory space used for data access is defined for each addressing method. If a bank select prefix is placed before an instruction, the memory space accessed by the instruction can be selected regardless of the addressing method. Table 2.9-1 lists the bank select prefix codes and selected memory spaces.

**Table 2.9-1 Bank select prefix codes and selected memory spaces**

Bank select prefix	Selected space
PCB	Program space
DTB	Data space
ADB	Additional space
SPB	When the value of the S flag in the condition code register (CCR) is 0, the user stack space is used, when the S flag is 1, the system stack space is used.

If a bank select prefix is used, some instructions perform an unexpected operation.

Table 2.9-2 lists the instructions that are not affected by bank select prefix codes. Table 2.9-3 lists instructions that require caution when they are used.

**Table 2.9-2 Instructions not affected by bank select prefixes**

Instruction type	Instruction	Effect of bank select prefix
String instruction	MOVS MOVSW SCEQ SCWEQ FILS FILSW	The bank register specified by the operand is used whether or not a prefix is used.
Stack operation	PUSHW POPW	When the S flag is 0, the user stack bank (USB) is used whether or not there is a prefix. When the S flag is 1, the system stack bank (SSB) is used regardless of whether a prefix is used.
I/O access instruction	MOV A MOVX A, io MOVW A, io MOV io, A MOVW io, A MOV io, #imm8 MOVW io, #imm16 MOVB A, io : bp MOVB io : bp, A SETB io : bp CLRB io : bp BBC io : bp, rel BBS io : bp, rel WBTC io, bp WBTS io : bp	The I/O space (000000 <sub>H</sub> to 0000FF <sub>H</sub> ) is accessed whether or not there is a prefix.
Interrupt return instruction	RETI	The system stack bank (SSB) is used whether or not a prefix is used.

**Table 2.9-3 Instructions whose use requires caution when bank select prefixes are used**

Instruction type	Instruction	Explanation
Flag change instruction	AND CCR, #imm8 OR CCR, #imm8	The effect of the prefix extends to the next instruction.
ILM setting instruction	MOV ILM, #imm8	The effect of the prefix extends to the next instruction.
PS return instruction	POPW PS	Do not place a bank select prefix before the PS return instruction.

**MB90820B Series****2.9.2 Common Register Bank Prefix (CMR)**

The common register bank prefix (CMR) is placed before an instruction that accesses a register bank to change the register accessed by the instruction to the common bank (register bank selected when RP = 0) at 000180<sub>H</sub> to 00018F<sub>H</sub> regardless of the current register bank pointer (RP) value.

**■ Common Register Bank Prefix (CMR)**

To facilitate data exchange between multiple tasks, a relatively simple means of accessing a fixed register bank regardless of the current register bank pointer (RP) value is necessary. This is the reason that the F<sup>2</sup>MC-16LX provides a common register bank for tasks, which is called the common bank. The common bank is located at address 000180<sub>H</sub> to 00018F<sub>H</sub>.

If the common register bank prefix (CMR) is placed before an instruction that accesses a register bank, registers accessed by the instruction can be changed to the common bank (register bank selected when RP = 0) at 000180<sub>H</sub> to 00018F<sub>H</sub> regardless of the current register bank pointer (RP) value.

Note that caution is required when this prefix is used with the instructions listed in Table 2.9-4 .

**Table 2.9-4 Instructions whose use requires caution when the common register bank prefix (CMR) is used**

Instruction type	Instruction	Explanation
String instruction	MOVS                      MOVSW SCEQ                      SCWEQ FILS                      FILSW	Do not place the CMR prefix before the string instruction.
Flag change instruction	AND    CCR, #imm8    OR   CCR, #imm8	The effect of the prefix extends to the next instruction.
PS return instruction	POPW   PS	The effect of the prefix extends to the next instruction.
ILM setting instruction	MOV    ILM, #imm8	The effect of the prefix extends to the next instruction.

## 2.9.3 Flag Change Suppression Prefix (NCC)

The flag change suppression prefix (NCC) code is placed before an instruction to suppress a flag change accompanying the execution of the instruction.

### ■ Flag Change Suppression Prefix (NCC)

The flag change suppression prefix (NCC) is used to suppress unnecessary flag changes. If a flag change suppression prefix code is placed before an instruction, a flag change accompanying the execution of the instruction is suppressed. Changes of the T, N, Z, V and C flags are suppressed.

Note that caution is required when this prefix is used with the instructions listed in Table 2.9-5.

**Table 2.9-5 Instructions whose use requires caution when the flag change suppression prefix (NCC) is used**

Instruction type	Instruction	Explanation
String instruction	MOVS MOVSW SCEQ SCWEQ FILS FILSW	Do not place the NCC prefix before the string instruction.
Flag change instruction	AND CCR, #imm8 OR CCR, #imm8	The condition code register (CCR) changes as defined in the instruction specification whether or not a prefix is used. The effect of prefix extends to the next instruction.
PS return instruction	POPW PS	The condition code register (CCR) changes as defined in the instruction specification whether or not a prefix is used. The effect of prefix extends to the next instruction.
ILM setting instruction	MOV ILM, #imm8	The effect of prefix extends to the next instruction.
Interrupt instruction Interrupt return instruction	INT #vct8 INT9 INT adder16 INTP addr24 RETI	The condition code register (CCR) changes as defined in the instruction specification whether or not a prefix is used.
Context switch instruction	JCTX @A	The condition code register (CCR) changes as defined in the instruction specification whether or not a prefix is used.

## MB90820B Series

### 2.9.4 Restrictions on Prefix Codes

The following three restrictions are imposed on the use of prefix codes:

- Interrupt/hold requests are not accepted during the execution of prefix codes and interrupt/hold suppression instructions.
- If a prefix code is placed before an interrupt/hold instruction, the effect of the prefix code is delayed.
- If consecutively placed prefix codes conflict, the last prefix code is valid.

#### ■ Prefix Codes and Interrupt/hold Suppression Instructions

Table 2.9-6 lists the interrupt/hold suppression instructions and prefix codes that have restrictions.

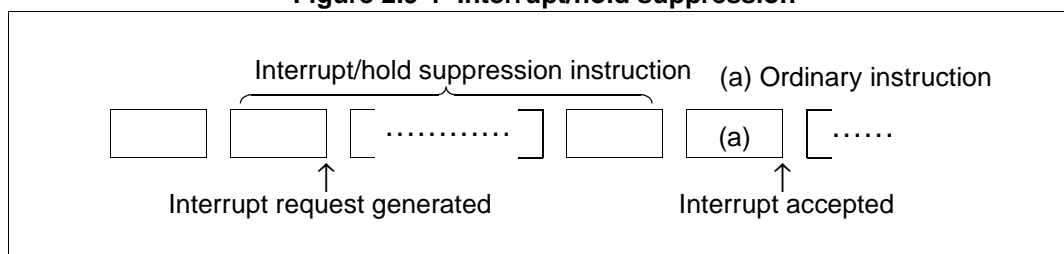
**Table 2.9-6 Prefix codes and interrupt suppression instructions sub sleep mode**

	Prefix codes	Interrupt/hold suppression instructions (instructions that delay the effect of prefix codes)
Instructions that do not accept interrupt and hold requests	PCB DTB ADB SPB CMR NCC	MOV ILM, #imm8 OR CCR, #imm8 AND CCR, #imm8 POPW PS

#### ● Interrupt/hold suppression

As shown in Figure 2.9-1, an interrupt or hold request generated during the execution of prefix codes and interrupt/hold instructions is not accepted. The interrupt/hold is not processed until the first instruction that is not governed by a prefix code or that is not an interrupt/hold suppression instruction is executed.

**Figure 2.9-1 Interrupt/hold suppression**

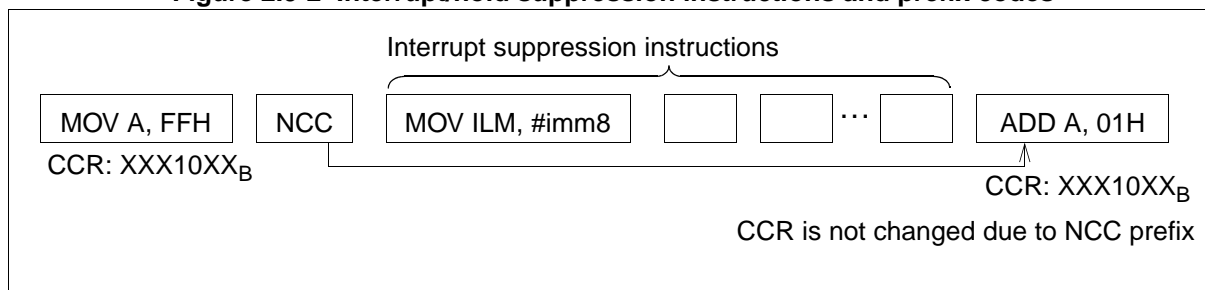


#### ● Delay of the effect of prefix codes

As shown in Figure 2.9-2, if a prefix code is placed before an interrupt/hold suppression instruction, the prefix code takes effect with the first instruction executed after the interrupt/hold suppression instruction.



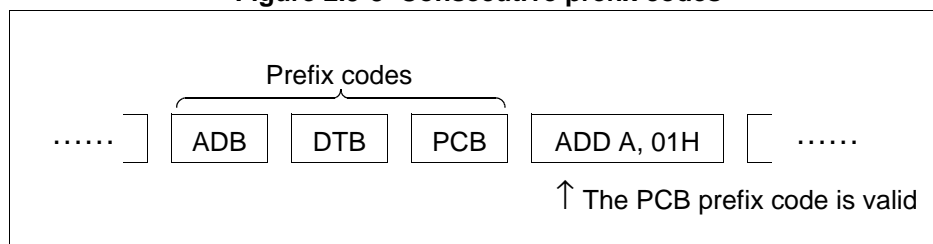
**Figure 2.9-2 Interrupt/hold suppression instructions and prefix codes**



## ■ Consecutive Prefix Codes

As shown in Figure 2.9-3, when consecutive conflicting prefix codes (PCB, ADB, DTB and SPB) are specified, the last prefix code is valid.

**Figure 2.9-3 Consecutive prefix codes**



# **CHAPTER 3**

---

# ***RESET***

**This chapter describes the function and operation of the reset for the MB90820B series microcontrollers.**

- 3.1 Reset
- 3.2 Reset Causes and Oscillation Stabilization Wait Intervals
- 3.3 External Reset Pin
- 3.4 Reset Operation
- 3.5 Reset Cause Bits
- 3.6 Status of Pins in a Reset

## 3.1 Reset

If a reset cause is generated, the CPU immediately stops the current execution process and waits for the reset to be cleared. When the reset is cleared, the CPU begins processing at the address indicated by the reset vector.

There are five causes of a reset:

- Power-on reset
- Watchdog timer overflow
- External reset request via the  $\overline{\text{RST}}$  pin
- Software reset request
- Clock supervisor reset request (MB90F828B only)

### ■ Reset Causes

Table 3.1-1 on page 64 lists the reset causes.

**Table 3.1-1 Reset causes**

Type of reset	Cause	Machine clock	Watchdog timer	Oscillation stabilization wait
External pin	L level input to $\overline{\text{RST}}$ pin	Previous state retained	Previous state retained	No
Software	“0” is written to the internal reset signal generation bit (RST) bit of the low-power consumption mode control register (LPMCR)	Previous state retained	Previous state retained	No
Watchdog timer	Watchdog timer overflow	MCLK	Stop count	Yes
Power-on	When the power is turned on	MCLK	Stop count	Yes
Clock Supervisor reset*	Main clock failure detected	Internal CR oscillator clock	Stop	No

MCLK: Main clock frequency (oscillation clock frequency divided by 2)

\*:MB90F828B only

## ● External reset

An external reset is generated by the "L" level input to an external reset pin ( $\overline{\text{RST}}$  pin). The minimum required period of the "L" level input to the  $\overline{\text{RST}}$  pin is 16 machine cycles ( $16/\phi$ ). The oscillation stabilization wait interval is not required for external resets.

---

### Reference:

For external reset requests via the  $\overline{\text{RST}}$  pin, if the reset cause is generated during a write operation (during the execution of a transfer instruction such as MOV), the CPU waits for the reset to be cleared after the instruction is completed. The normal write operation is therefore completed even though a reset is input concurrently.

Note, however, that waiting for the reset to be cleared may start before the transfer of the contents for a counter specified by a string-processing instruction is completed.

---

## ● Software reset

A software reset is an internal reset of three machine cycles ( $3/\phi$ ) generated by writing 0 to the RST bit of the low-power consumption mode control register (LPMCR). The oscillation stabilization wait interval is not required for software resets.

## ● Watchdog reset

A watchdog reset is generated by a watchdog timer overflow that occurs when 0 is not written to the watchdog control bit (WTE) bit of the watchdog timer control register (WDTC) within a given time after the watchdog timer is activated. The oscillation stabilization wait interval can be set by the clock selection register (CKSCR).

## ● Power-on reset

A power-on reset is generated when the power is turned on. The oscillation stabilization wait interval is fixed at  $2^{16}$  oscillation clock cycles ( $2^{16}/\text{HCLK}$ ). After the oscillation stabilization wait interval has elapsed, the reset is executed.

See Section "4.1 Clock", for details.

## ● Clock supervisor reset

A reset is generated upon detecting a main clock failure.

Clock supervisor reset does not wait for elapsing of the oscillation stabilization wait time.

---

### Reference:

- Definition of clocks
    - HCLK: Oscillation clock frequency (clock supplied from oscillation pin)
    - MCLK: Main clock frequency (clock of oscillation clock divided by 2)
    - $\phi$ : Machine clock (CPU operating clock) frequency
    - $1/\phi$ : Machine cycle (CPU operating clock cycle)
-

## 3.2 Reset Causes and Oscillation Stabilization Wait Intervals

The F<sup>2</sup>MC-16LX has five reset causes. The oscillation stabilization wait interval for a reset depends on the reset cause.

### ■ Reset Causes and Oscillation Stabilization Wait Intervals

Table 3.2-1 summarizes reset causes and oscillation stabilization wait intervals.

**Table 3.2-1 Reset causes and oscillation stabilization wait intervals**

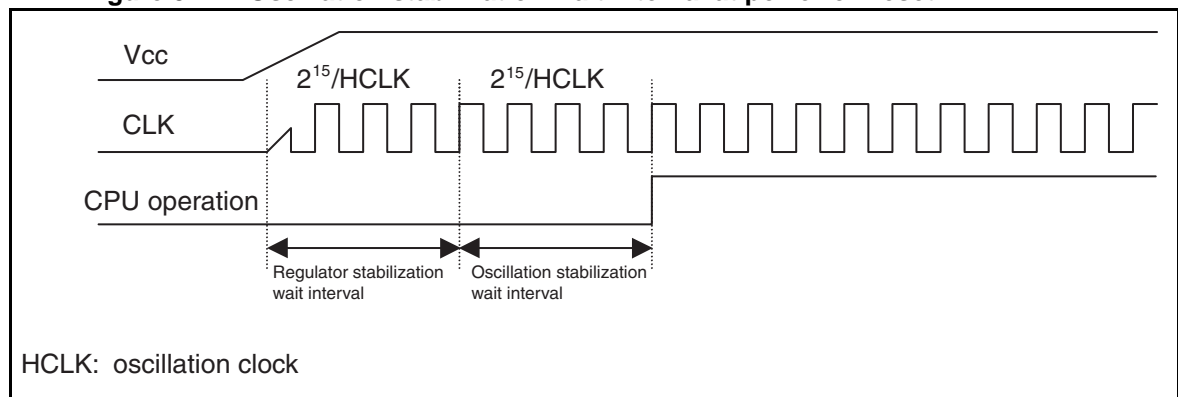
Reset cause	Oscillation stabilization wait interval The corresponding time interval for an oscillation clock frequency of 4 MHz is given in parentheses.
Power-on reset	$2^{16}/\text{HCLK}$ (approximately 16.39 ms)
Watchdog timer	$2^{16}/\text{HCLK}$ (approximately 16.39 ms)
External reset via the $\overline{\text{RST}}$ pin	None. (However the WS1 & WS0 bits are initialized to “11”.)
Software reset	None. (However the WS1 & WS0 bits are initialized to “11”.)
Clock supervisor reset*	None. (However the WS1 & WS0 bits are initialized to “11”.)

HCLK: Oscillation clock frequency

\*: MB90F828B only

Figure 3.2-1 shows the oscillation stabilization wait interval of the product at power-on reset.

**Figure 3.2-1 Oscillation stabilization wait interval at power-on reset**



Note :

Ceramic and crystal oscillators generally require an oscillation stabilization wait interval of a few to several dozen milliseconds after the start of oscillation until they stabilize at their natural frequency. Be sure to set a proper oscillation stabilization wait interval for the specific oscillator used.

See Section "3.2 Oscillation Stabilization Wait Interval", for detail about oscillation stabilization wait interval.

#### ■ Oscillation Stabilization Wait and Reset State

A reset operation in response to a power-on reset and other externally activated resets during stop mode is performed after the oscillation stabilization wait interval has elapsed. This time interval is generated by the time-base timer. If the external reset input has not been cleared after the interval, the reset operation is performed after the external reset is cleared.

### 3.3 External Reset Pin

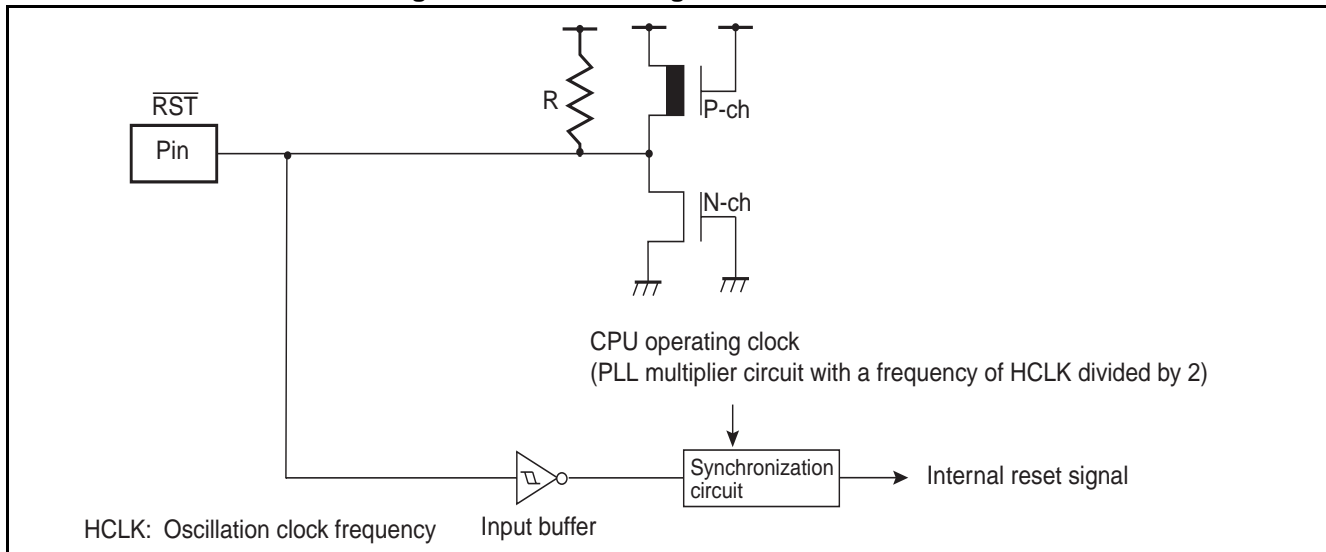
The external reset pin ( $\overline{\text{RST}}$  pin) is a dedicated pin for inputting, with an L level signal, a reset and generating an internal reset by the L level input.

For MB90820B series microcontrollers, resets are generated in synchronization with the CPU operating clock. Asynchronous resets are generated only for the external terminals.

#### ■ Block Diagram of the External Reset Pin

Figure 3.3-1 shows a block diagram of the external reset pin.

Figure 3.3-1 Block diagram of external reset



Note :

Inputs to the  $\overline{\text{RST}}$  pin are accepted during cycles in which memory is not affected to prevent memory from being destroyed by a reset during a write operation.

A clock is required to initialize the internal circuit. In particular, an operation with an external clock requires clock input together with reset input.

## 3.4 Reset Operation

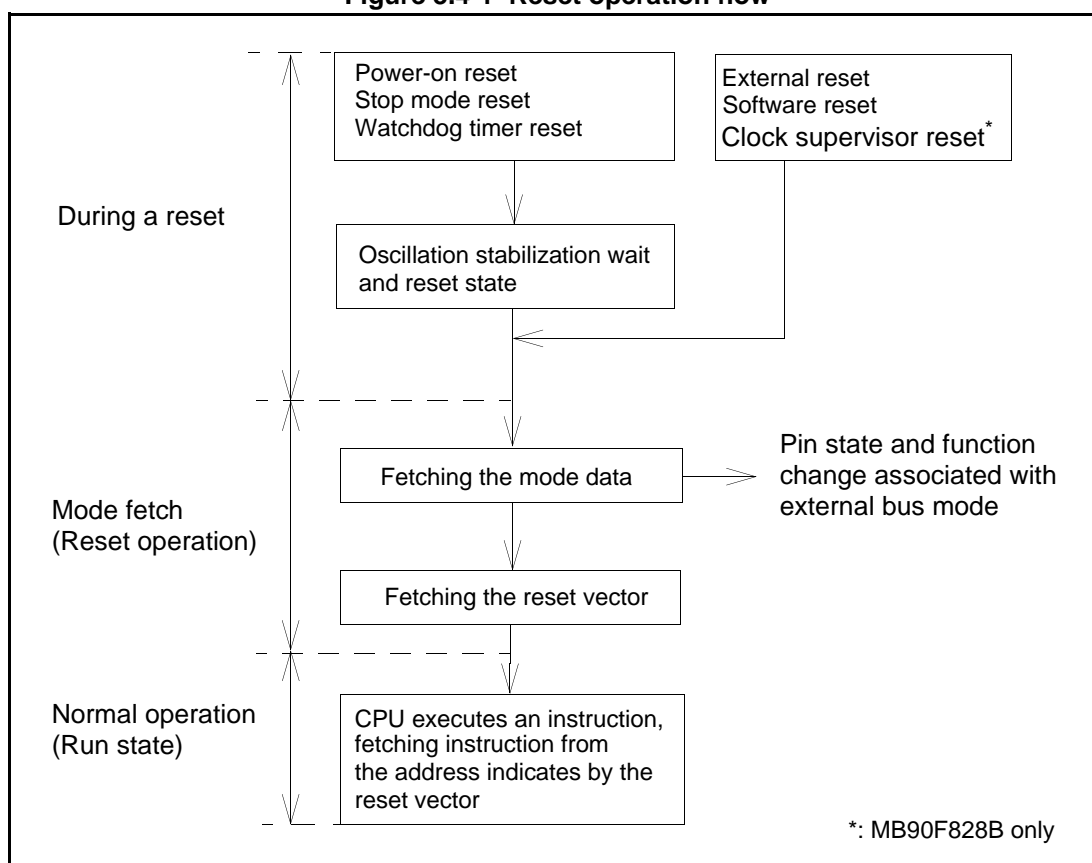
When a reset is cleared, the memory locations from which the mode data and the reset vector are read are selected according to the setting of the mode pins, and the mode setting data is fetched. Mode setting data determines the CPU operating mode and the execution start address after a reset operation ends.

For power-on or recovery from stop mode by a reset, the mode is fetched after the oscillation stabilization wait time has elapsed.

### ■ Overview of Reset Operation

Figure 3.4-1 shows the reset operation flow.

Figure 3.4-1 Reset operation flow



### ■ Mode Pins

Setting the mode pins (MD0 to MD2) specifies how to fetch the reset vector and the mode data. Fetching the reset vector and the mode data is performed in the reset sequence. See Section "8.1 Mode Setting", for details about mode pins.

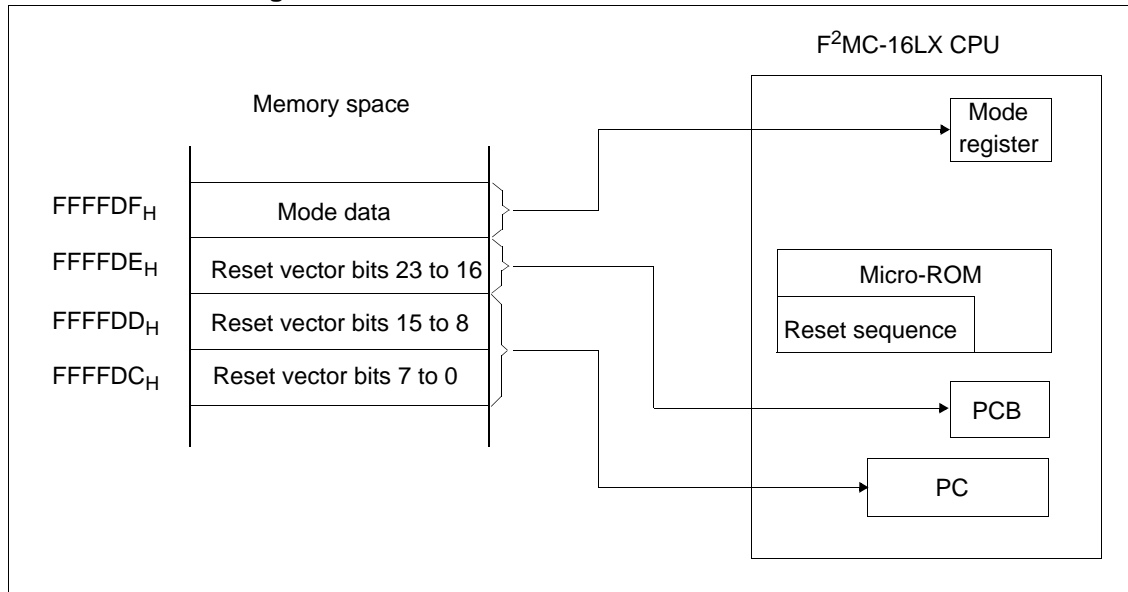


## ■ Mode Data Fetch

When the reset is cleared, the CPU transfers the reset vector and the mode data stored in the hardware memory to the appropriate registers in the CPU core. The reset vector and mode data are allocated to the four bytes from FFFFDC<sub>H</sub> to FFFFDF<sub>H</sub>. The CPU outputs these addresses to the bus immediately after the reset is cleared and fetches the reset vector and mode data. Using mode fetching, the CPU can begin processing at the address indicated by the reset vector.

Figure 3.4-2 shows the transfer of the reset vector and mode data.

**Figure 3.4-2 Transfer of reset vector and mode data**



### ● Mode data (address: FFFFDF<sub>H</sub>)

Only the reset operation changes the contents of the mode register. The mode register setting is valid after a reset operation. See Section "8.1 Mode Setting", for details about mode data.

### ● Reset vector (address: FFFFDC<sub>H</sub> to FFFFDE<sub>H</sub>)

The execution start address after the reset operation ends is written as the reset vector. Execution starts at the address contained in the reset vector.

# MB90820B Series

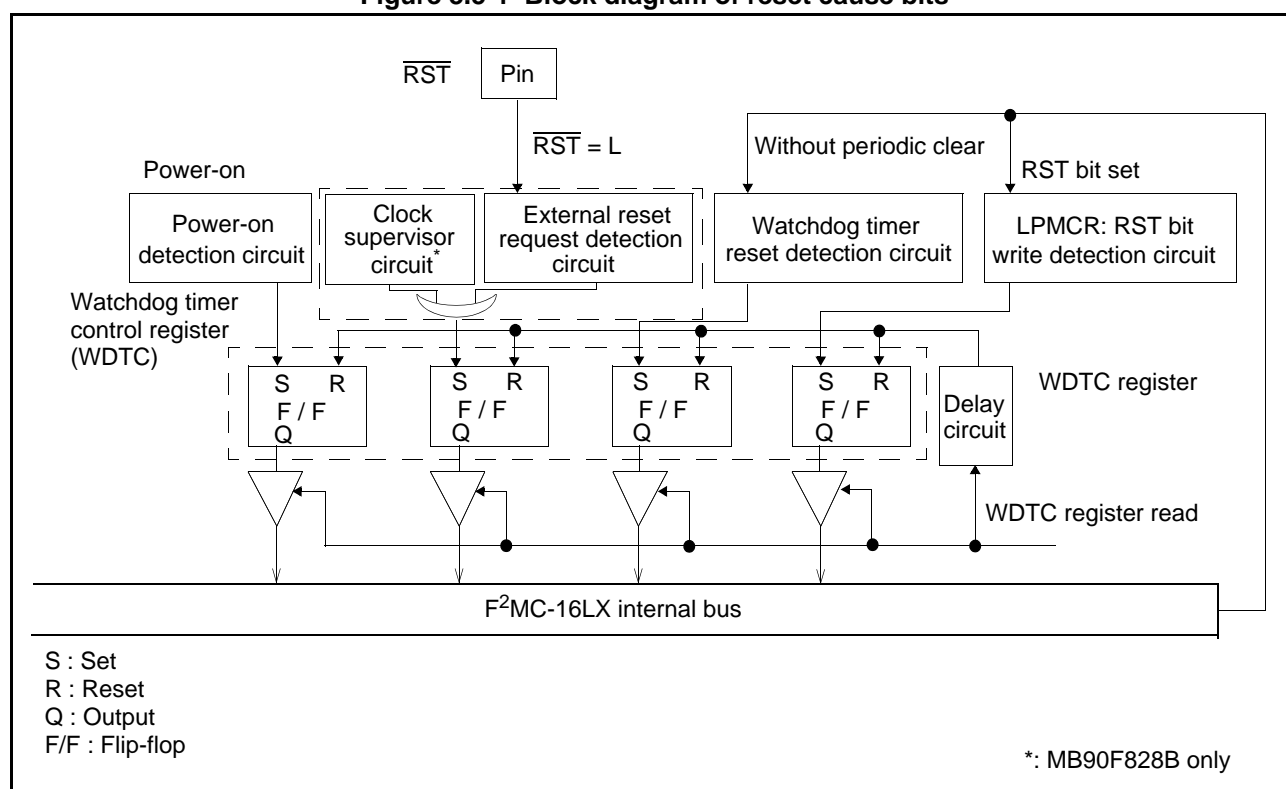
## 3.5 Reset Cause Bits

A reset cause can be identified by reading the flag in the watchdog timer control register (WDTC).

### Reset Cause Bits

As shown in Figure 3.5-1, a flip-flop is associated with each reset cause. The contents of the flip-flops are obtained by reading the watchdog timer control register (WDTC). If it is necessary to identify the cause of a reset after the reset has been cleared, the value read from the WDTC should be processed by the software and a branch should be made to the appropriate program.

Figure 3.5-1 Block diagram of reset cause bits



## ■ Correspondence Between Reset Cause Bits and Reset Causes

Figure 3.5-2 shows the configuration of the reset cause flag bits of the watchdog timer control register (WDTC). Table 3.5-1 on page 72 shows the correspondence between the reset cause flag bits and reset causes.

**Figure 3.5-2 Configuration of reset cause flag bits (watchdog timer control register)**

Watchdog timer control register (WDTC)									
	bit	7	6	5	4	3	2	1	0
Address : 0000A8 <sub>H</sub>		PONR	-	WRST	ERST	SRST	WTE	WT1	WT0
Read/write ⇒		(R)	(-)	(R)	(R)	(R)	(W)	(W)	(W)
Default value ⇒		(X)	(X)	(X)	(X)	(X)	(1)	(1)	(1)

**Table 3.5-1 Correspondence between reset cause flag bits and reset causes**

Reset cause	PONR	WRST	ERST	SRST
Power-on reset	1	X	X	X
Watchdog timer overflow	*	1	*	*
External reset request via $\overline{\text{RST}}$ pin Clock supervisor reset (MB90F828B only)	*	*	1	*
Software reset request	*	*	*	1

\*:Previous state retained

X: Undefined

## ■ Notes About Reset Cause Bits

### ● Multiple reset causes generated at the same time

When multiple reset causes are generated at the same time, the corresponding reset cause bits of the watchdog timer control register (WDTC) are set to "1".

If, for example, an external reset request via the  $\overline{\text{RST}}$  pin and the watchdog timer overflow occur at the same time, both the ERST bit and the WRST bit are set to "1".

### ● Power-on reset

For a power-on reset, the PONR bit is set to "1", but all other reset cause bits are undefined.

Consequently, program the software so that it will ignore all reset cause bits except the PONR bit if it is "1".

### ● Clearing the reset cause bits

The reset cause bits are cleared only when the watchdog timer control register (WDTC) is read. Any bit that corresponds to a reset cause that has already been generated once is not cleared even though another reset is generated (its setting of "1" is retained).

## MB90820B Series

### 3.6 Status of Pins in a Reset

---

**This section describes the status of pins when a reset occurs.**

---

#### ■ Status of Pins During a Reset

The status of pins during a reset depends on the settings of mode pins ( $MD2$  to  $MD0 = 011_B$ ).

- When internal vector mode has been set:

All I/O pins (resource pins) are high impedance, and mode data is read from the internal ROM.

#### ■ Status of Pins After Mode Data is Read

The status of pins after mode data has been read depends on the mode data ( $M1$  and  $M0 = 00_B$ ).

- When single-chip mode has been selected ( $M1, M0 = 00_B$ ):

All I/O pins (resource pins) are high impedance, and mode data is read from the internal ROM.

---

Note :

For those pins that change to high impedance when a reset cause is generated, take care that devices connected to them do not malfunction.

---

See Table 5.7-1 for information about the state of pins during a reset.



# **CHAPTER 4**

---

# **CLOCK**

**This chapter describes the function and operation of the clock used by MB90820B series microcontrollers.**

- 4.1 Clock
- 4.2 Block Diagram of the Clock Generation Block
- 4.3 Clock Selection Registers
- 4.4 Clock Mode
- 4.5 Oscillation Stabilization Wait Interval
- 4.6 Connection of an Oscillator or an External Clock to the Microcontroller

## 4.1 Clock

---

**The clock generation block controls the operation of the internal clock that controls operation of the CPU and peripheral functions. This internal clock is called the machine clock ( $\phi$ ). One internal clock cycle is regarded as one machine cycle. Other clocks include a clock generated by source oscillation, called an oscillation clock, and a clock generated by the internal PLL oscillation, called a PLL clock.**

---

### ■ Clock

The clock generation block contains the oscillation circuit that generates the oscillation clock. An external oscillator is attached to this circuit. The oscillation clock can also be supplied by inputting an external clock to the clock generation block.

The clock generation block also contains the PLL clock multiplier circuit, which generates five clocks that are multiples of the oscillation clock.

The clock generation block controls the oscillation stabilization wait interval and PLL clock multiplication as well as controls internal clock operation by changing the clock with a clock selector.

#### ● Oscillation clock (HCLK)

The oscillation clock is generated either from an external oscillator attached to the oscillation circuit or by input of an external clock.

#### ● Main clock (MCLK)

The main clock, which is the oscillation clock divided by 2, supplies the clock input to the time-base timer and the clock selector.

#### ● PLL clock (PCLK)

The PLL clock is obtained by multiplying the oscillation clock with the internal PLL multiplier circuit (PLL oscillation circuit). Selection can be made from among four different PLL clocks.

#### ● Machine clock ( $\phi$ )

The machine clock controls the operation of the CPU and peripheral functions. One clock cycle is regarded as one machine cycle ( $1/\phi$ ). An operating machine clock can be selected from among the main clock that is generated from the source clock frequency divided by 2 and the five clocks that are multiples of the source clock frequency.

---

#### Note :

Although an oscillation clock of 3 MHz to 48 MHz can be generated when the operating voltage is 5 V, the maximum operating frequency for the CPU and peripheral functions is 24 MHz. If a frequency multiplier rate exceeding the maximum operating frequency is specified, devices will not operate correctly. If, for example, a source oscillation of 12 MHz is generated, only a multiplier of 2 can be specified.

---

A PLL clock oscillation of 4 to 24 MHz is possible, but this range depends on the operating voltage and multiplier. See "Data Sheet", for details.

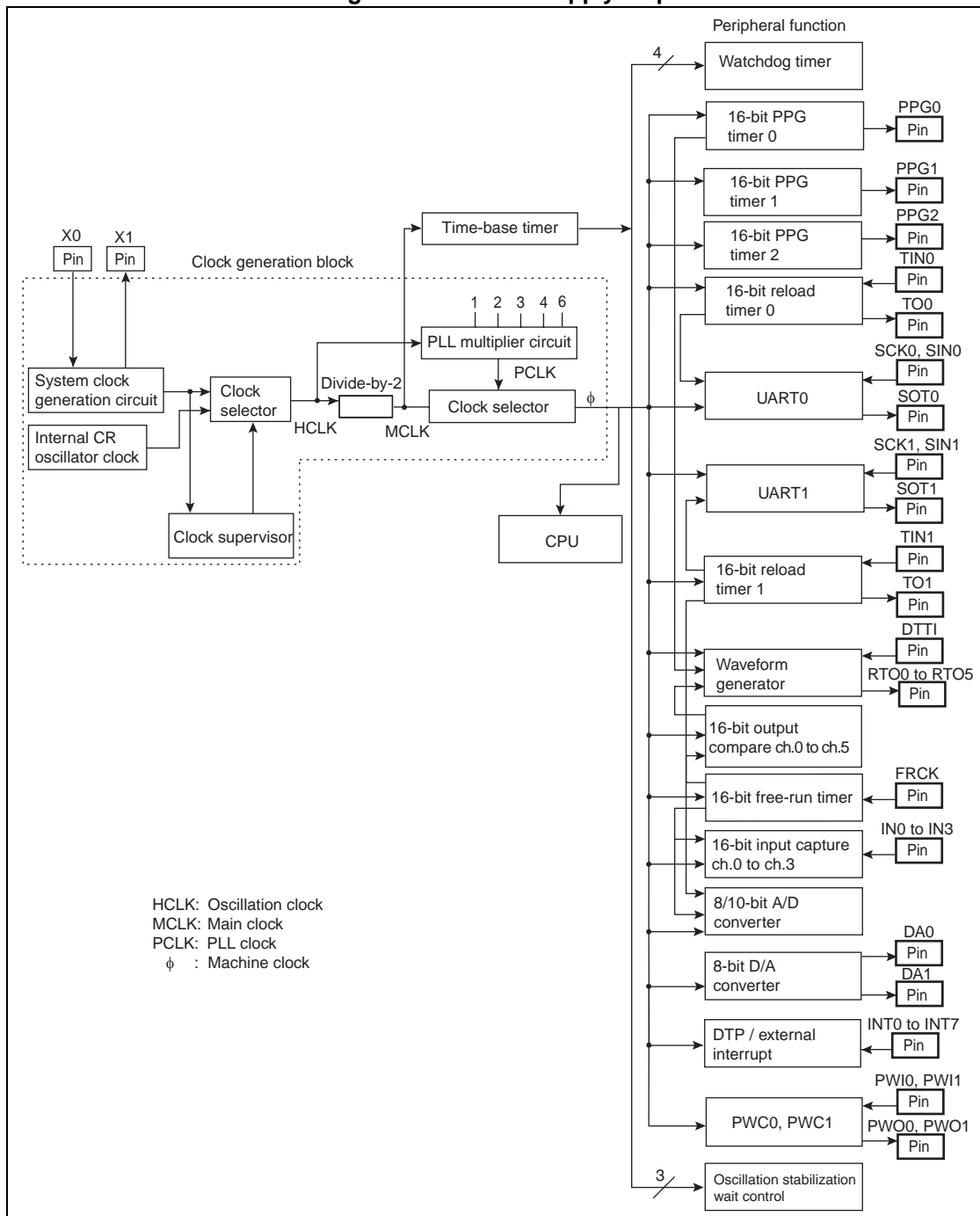
#### ■ Clock Supply Map

Since the machine clock generated in the clock generation block is supplied as the clock that controls operation of the CPU and peripheral functions, the operation of the CPU and peripheral functions is affected by switching of the main clock and the PLL clock (clock mode) and a changing in the PLL clock multiplier rate.

Since some peripheral functions receive frequency-divided output from the time-base timer, a peripheral unit can select the clock best suited for its operation.

Figure 4.1-1 shows the clock supply map.

**Figure 4.1-1 Clock supply map**





## 4.2 Block Diagram of the Clock Generation Block

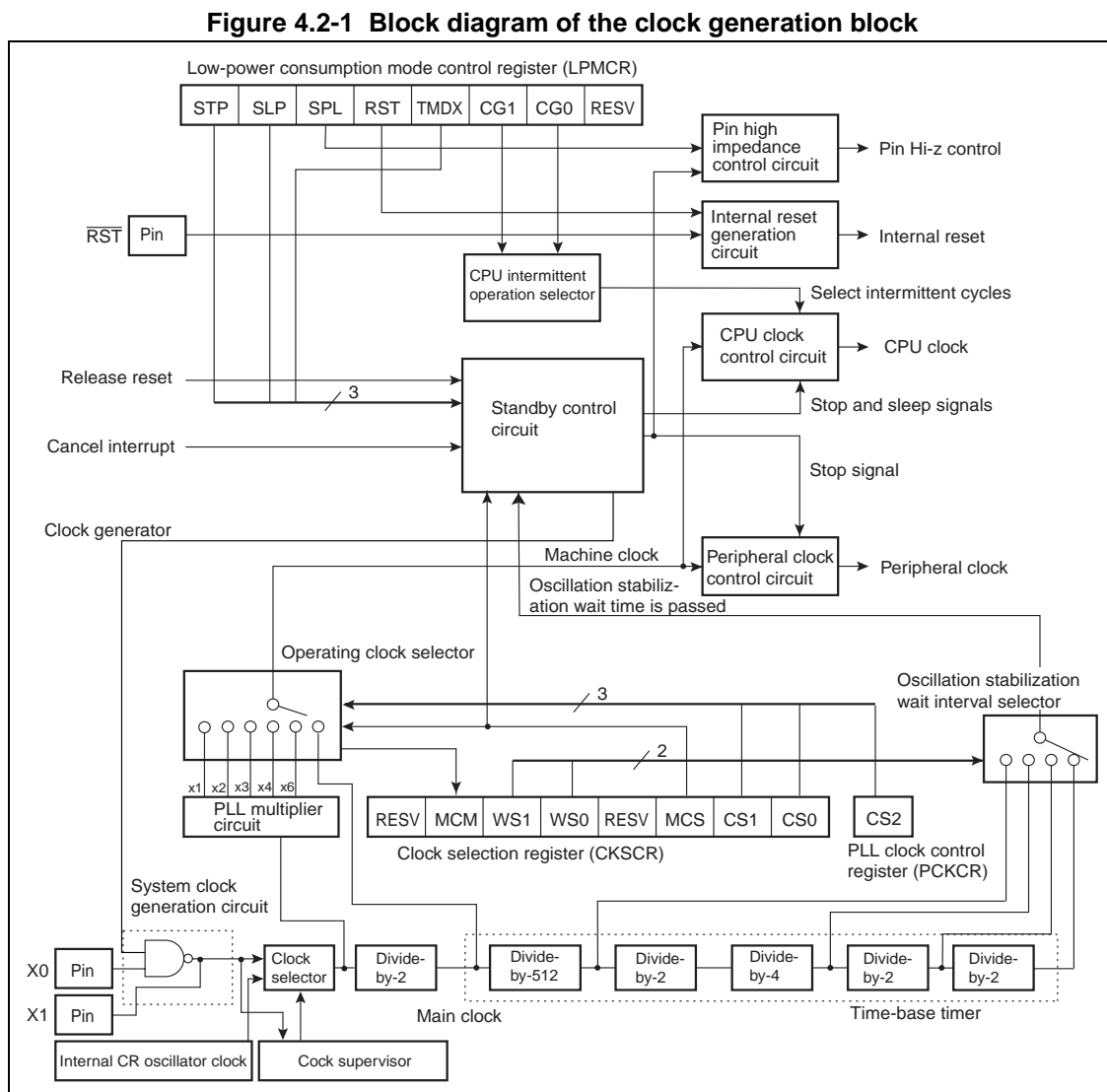
The clock generation block consists of five blocks:

- System clock generation circuit
- PLL multiplier circuit
- Operating clock selector
- Clock selection register (CKSCR)
- Oscillation stabilization wait interval selector

### ■ Block Diagram of the Clock Generation Block

Figure 4.2-1 shows a block diagram of the clock generation block.

Figure 4.2-1 also includes the standby control circuit and time-base timer circuit.



- System clock generation circuit

The system clock generation circuit generates an oscillation clock (HCLK) from an external oscillator attached to it. Alternatively, an external clock can be input to this circuit.

- PLL multiplier circuit

The PLL multiplier circuit multiplies the oscillation clock (HCLK) through PLL oscillation and supplies a clock that is a multiple of the frequency to the CPU clock selector.

- Operating clock selector

From among the main clock and five different PLL clocks, the operating clock selector selects the clock that is supplied to the CPU and peripheral clock control circuits.

- Clock selection register (CKSCR) and PLL clock control register (PCKCR)

The clock selection register and PLL clock control register are used to set switching between the oscillation clock and a PLL clock, selection of an oscillation stabilization wait interval, and selection of a PLL clock multiplier rate.

- Oscillation stabilization wait interval selector

This selector selects an oscillation stabilization wait interval for the oscillation clock when stop mode is released or when a watchdog timer reset occurs. Selection is made from among three kinds of time-base timer output. In all other cases, an oscillation stabilization wait interval is not selected.

## 4.3 Clock Selection Registers

The clock selection registers consist of clock selection register (CKSCR) and PLL clock control register (PCKCR).

### ■ Clock Selection Registers

Figure 4.3-1 shows the clock selection register (CKSCR) and PLL clock control register (PCKCR).

Figure 4.3-1 Clock selection registers

<b>CKSCR</b>									
	Bit	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
Address: 0000A1 <sub>H</sub>		RESV	MCM	WS1	WS0	RESV	MCS	CS1	CS0
Read/write		(R)	(R)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)
Initial value: 11111100 <sub>B</sub>									
<b>PCKCR</b>									
	Bit	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
Address: 00002F <sub>H</sub>		—	—	—	—	RESV	RESV	RESV	CS2
Read/write		—	—	—	—	(W)	(W)	(W)	(W)
Initial value : xxxx0000 <sub>B</sub>									
(R): Read only									
(W): Write only									
(R/W): Read / write									
-: Undefined									

## MB90820B Series

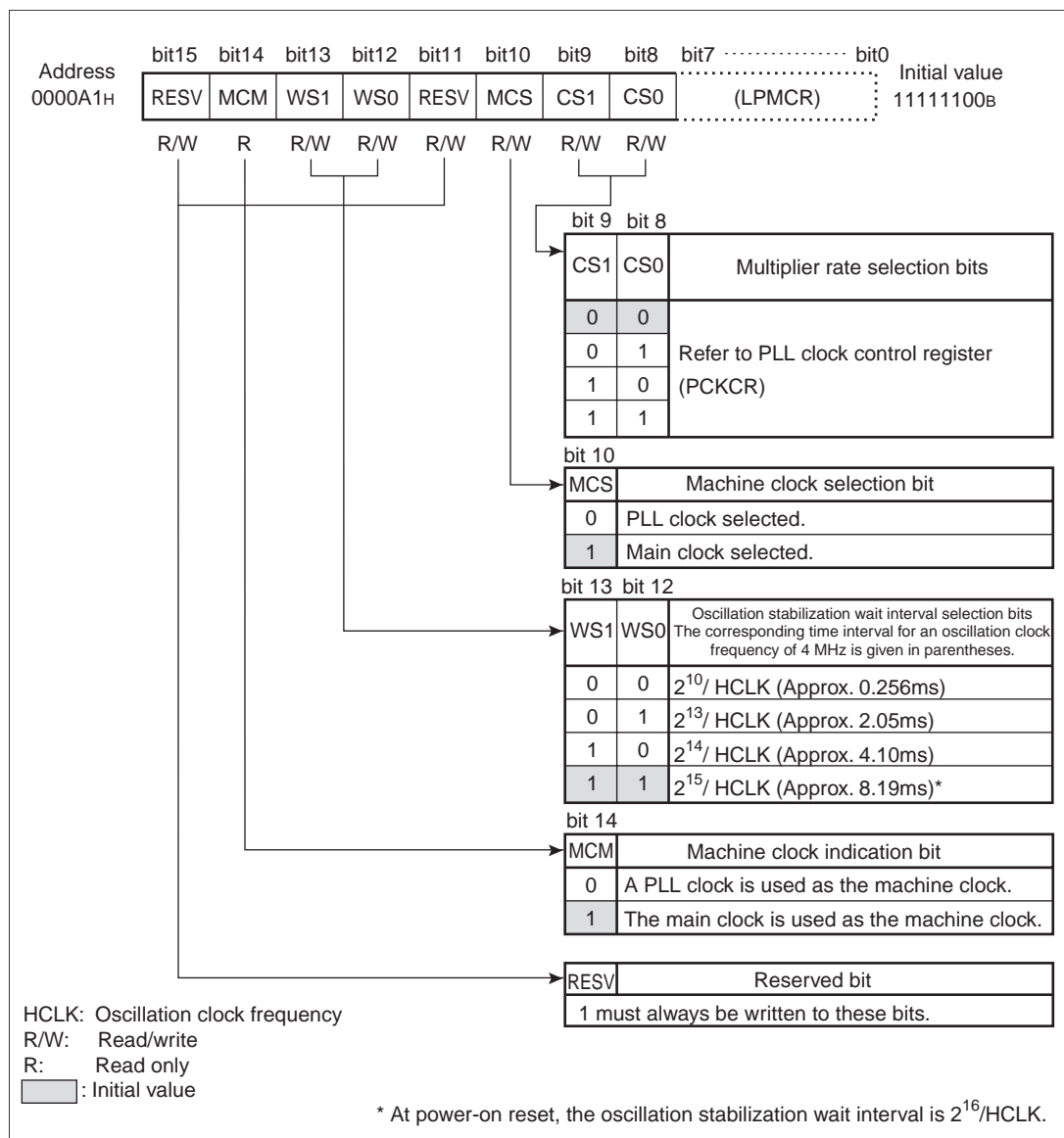
### 4.3.1 Clock Selection Register (CKSCR)

The clock selection register (CKSCR) is used to set switching between the main clock and a PLL clock, selection of an oscillation stabilization wait interval, and selection of a PLL clock multiplier rate.

#### ■ Configuration of the Clock Selection Register (CKSCR)

Figure 4.3-2 shows the configuration of the clock selection register (CKSCR). Table 4.3-1 describes the function of each bit in the clock selection register (CKSCR).

Figure 4.3-2 Configuration of the clock selection register (CKSCR)



Note :

If the machine clock selection bit is not set, the main clock is used as the machine clock.

**Table 4.3-1 Function description of each bit in the clock selection register (CKSCR)**

Bit name		Function
bit15, bit11	RESV: Reserved bits	<b>Note :</b> <ul style="list-style-type: none"> <li>"1" must always be written to these bits.</li> </ul>
bit14	MCM: Machine clock indication bit	<ul style="list-style-type: none"> <li>Indicates whether the main clock or PLL clock is used as the machine clock. MCM = "0": PLL clock is selected MCM = "1": Main clock is selected</li> <li>If MCS = "0" and MCM = "1", this indicates the PLL clock oscillation stabilization wait state.</li> <li>Writing has no effect on the operation.</li> </ul>
bit13, bit12	WS1, WS0: Oscillation stabilization wait interval selection bits	<ul style="list-style-type: none"> <li>These bits select an oscillation stabilization wait interval of the oscillation clock after stop mode has been released.</li> <li>These bits are initialized to "11<sub>B</sub>" by all reset causes.</li> </ul> <b>Note :</b> The oscillation stabilization wait interval must be set to a value appropriate for the oscillator used. See Section "3.2 Reset Causes and Oscillation Stabilization Wait Intervals". The oscillation stabilization wait period for all PLL clocks is fixed at $2^{14}/\text{HCLK}$ .
bit10	MCS: Machine clock selection bit	<ul style="list-style-type: none"> <li>This bit specifies whether the main clock or a PLL clock is selected as the machine clock.</li> <li>When this bit is "0", a PLL clock is selected. When this bit is "1", the main clock is selected.</li> <li>If this bit has been set to "1" and "0" is written to it, the oscillation stabilization wait interval for the PLL clock starts. As a result, the time-base timer is automatically cleared, and the TBOF bit of the time-base timer control register (TBTC) is also cleared.</li> <li>For PLL clocks, the oscillation stabilization wait period is fixed at <math>2^{14}/\text{HCLK}</math> (the oscillation stabilization wait interval is approx. 2 ms for an oscillation clock frequency of 4 MHz).</li> <li>When the main clock has been selected, the operating clock frequency is the frequency of the oscillation clock divided by 2 (e.g., the operating clock is 2 MHz when the oscillation clock frequency is 4 MHz).</li> <li>This bit is initialized to "1" by power-on or watchdog reset.</li> </ul> <b>Note :</b> When the MCS bit is "1", write "0" to it only when the time-base timer interrupt is masked by the TBIE bit of the time-base timer control register (TBTC) or the interrupt level register (ILM).
bit9, bit8	CS1, CS0: Multiplier rate selection bits	<ul style="list-style-type: none"> <li>These bits, combine with CS2 bit of PCKCR, select a PLL clock multiplier rate.</li> <li>Selection can be made from among five different multiplier rates.</li> <li>These bits are initialized to 00<sub>B</sub> by all reset causes.</li> <li>Refer to PCKCR for the relationship between setting CS2, CS1 and CS0 bits and the PLL clock multiplier rate selection,</li> </ul> <b>Note :</b> When the MCS bit is "0", writing to these bits is not allowed. Write to the CS1 and CS0 bits only after setting the MCS bit to "1" (main clock mode).

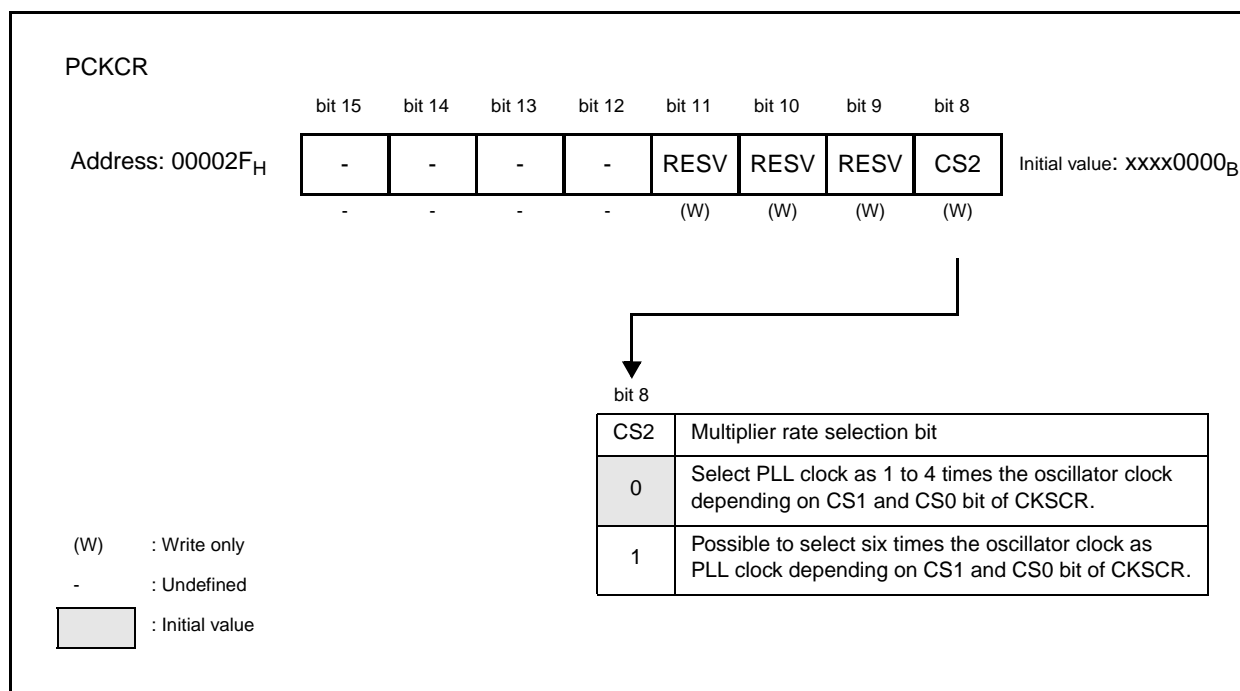
### 4.3.2 PLL Clock Control Register (PCKCR)

The PLL clock control register (PCKCR), combine with CS1 and CS0 bits in CKSCR, is used to select a PLL clock multiplier rate.

#### ■ Configuration of the PLL Clock Control Register (PCKCR)

Figure 4.3-3 shows the configuration of the PLL clock control register (PCKCR). Table 4.3-2 describes the function of each bit in the PLL clock control register (PCKCR).

**Figure 4.3-3 Configuration of the PLL clock control register (PCKCR)**



**Table 4.3-2 Function description of each bit in the PLL clock control register (PCKCR)**

Bit name		Function																												
bit 15 to bit 12	Not used bits	<ul style="list-style-type: none"><li>When read, the value is undefined.</li><li>Writing has no effect on operation.</li></ul>																												
bit 11 to bit 9	Reserved bits	<ul style="list-style-type: none"><li>When read, the value is undefined.</li><li>Always write "0" to these bits</li></ul>																												
bit8	CS2: Multiplier rate selection bit	<ul style="list-style-type: none"><li>These bits, and CS2 bit of PCKCR, select a PLL clock multiplier rate.</li><li>Selection can be made from among five different multiplier rates.</li><li>This bit is initialized to "0" by all reset causes.</li><li>The read value is undefined.</li><li>Recommended setting of CS2, CS1 and CS0 bits:</li></ul>																												
		<table><tr><td>CS2</td><td>CS1</td><td>CS0</td><td>PLL clock multiplex time</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1 × HCLK (4 MHz)</td></tr><tr><td>0</td><td>0</td><td>1</td><td>2 × HCLK (8 MHz)</td></tr><tr><td>0</td><td>1</td><td>0</td><td>3 × HCLK (12 MHz)</td></tr><tr><td>0</td><td>1</td><td>1</td><td>4 × HCLK (16 MHz)</td></tr><tr><td>1</td><td>1</td><td>0</td><td>6 × HCLK (24 MHz)</td></tr><tr><td colspan="3">( other )</td><td>Setting not allowed</td></tr></table>	CS2	CS1	CS0	PLL clock multiplex time	0	0	0	1 × HCLK (4 MHz)	0	0	1	2 × HCLK (8 MHz)	0	1	0	3 × HCLK (12 MHz)	0	1	1	4 × HCLK (16 MHz)	1	1	0	6 × HCLK (24 MHz)	( other )			Setting not allowed
		CS2	CS1	CS0	PLL clock multiplex time																									
		0	0	0	1 × HCLK (4 MHz)																									
		0	0	1	2 × HCLK (8 MHz)																									
		0	1	0	3 × HCLK (12 MHz)																									
		0	1	1	4 × HCLK (16 MHz)																									
		1	1	0	6 × HCLK (24 MHz)																									
		( other )			Setting not allowed																									
		<b>Note :</b>																												
When the MCS bit of CKSCR is "0", writing to this bit is not allowed. Write to the CS2 bits only after setting the MCS bit of CKSCR to "1" (main clock mode).																														

## 4.4 Clock Mode

---

Two clock modes are provided: main clock mode and PLL clock mode.

---

### ■ Main Clock Mode and PLL Clock Mode

- Main clock mode

In main clock mode, the main clock, whose frequency is the oscillation clock (HCLK) divided by 2, is used as the operating clock for the CPU and peripheral resources, and the PLL clocks are disabled.

- PLL clock mode

In PLL clock mode, a PLL clock is used as the operating clock for the CPU and peripheral resources. A PLL clock multiplier rate is selected with the clock selection register (CKSCR: CS1 and CS0) and PLL clock control register (PCKCR: CS2).

### ■ Clock Mode Transition

Switching between main clock mode and PLL clock mode is done by writing to the MCS bit of the clock selection register (CKSCR).

- Switching from main clock mode to PLL clock mode

When the MCS bit of CKSCR is "1" and "0" is written to it, the switch from the main clock to a PLL clock occurs after the PLL clock oscillation stabilization wait period ( $2^{14}/\text{HCLK}$ ).

- Switching from PLL clock mode to main clock mode

When the MCS bit of CKSCR is "0" and "1" is written to it, the switch from the PLL clock to the main clock occurs when the edges of the PLL clock and the main clock coincide (after 1 to 12 PLL clocks).

---

Note :

Even though the MCS bit of CKSCR is rewritten, machine clock switching does not occur immediately. When operating a resource that depends on the machine clock, make sure that machine clock switching has been done by referring to the MCM bit of CKSCR before operating the resource.

---

### ■ Selection of a PLL Clock Multiplier Rate

Writing a value from "000<sub>B</sub>" to "011<sub>B</sub>" or "110<sub>B</sub>" to the CS2 bit of PCKCR, the CS1 and CS0 bits of CKSCR selects one to the five PLL clock multiplier rates.

### ■ Machine Clock

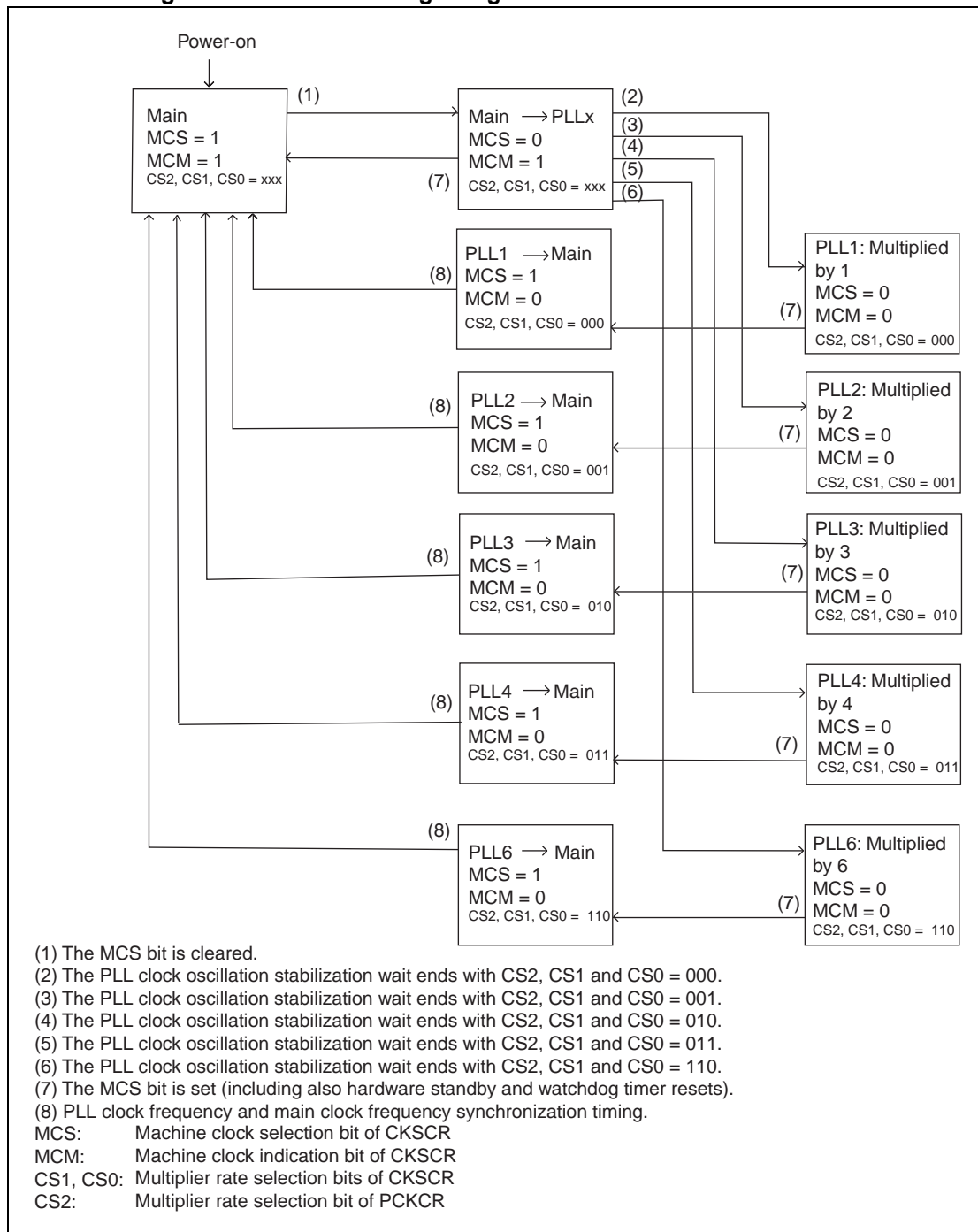
The machine clock may be either a PLL clock output from the PLL multiplier circuit or the clock that is the source oscillation frequency divided by 2. This machine clock is supplied to the CPU and peripheral functions.

Either the main clock or a PLL clock can be selected by writing to the MCS bit of CKSCR.

Figure 4.4-1 shows the status change caused by the machine clock switching.



Figure 4.4-1 Status change diagram for machine clock selection



Note :

The initial value for the machine clock setting is main clock (MCS of CKSCR = 1).

## 4.5 Oscillation Stabilization Wait Interval

When the power is turned on, when stop mode is released, or when a watchdog timer reset occurs, the oscillation clock starts, oscillation is unstable initially. Therefore, an oscillation stabilization wait interval is required. When the switch from the main clock to a PLL clock occurs, an oscillation stabilization wait interval is also required after PLL oscillation starts.

### ■ Oscillation Stabilization Wait Interval

Ceramic and crystal oscillators generally require an oscillation stabilization wait interval of a few to several dozen milliseconds until they stabilize at their natural frequency when oscillation starts.

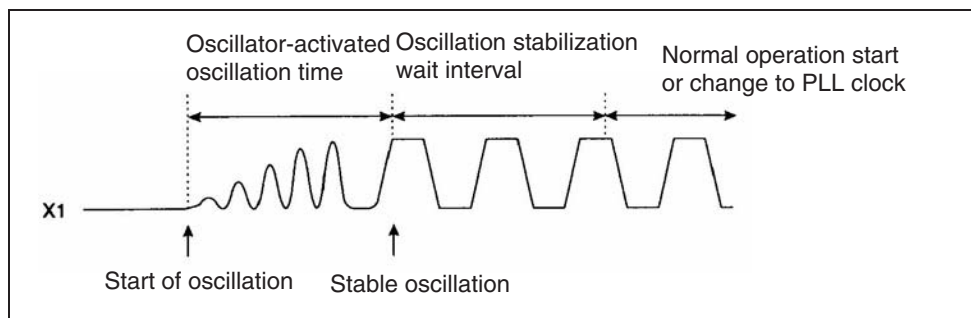
For this reason, CPU operation is not allowed as soon as oscillation starts and is allowed only after full stabilization of oscillation. After the oscillation stabilization wait interval has elapsed, the clock is supplied to the CPU.

Because the oscillation stabilization wait time depends on the type of the oscillator (crystal, ceramic, etc.), the proper oscillation stabilization wait interval for the oscillator used must be selected. An oscillation stabilization wait interval is selected by setting the clock selection register (CKSCR).

In a switch from the main clock to a PLL clock, the CPU continues to operate on the main clock during the oscillation stabilization wait interval of the PLL. After this interval, the operating clock switches to the PLL clock.

Figure 4.5-1 shows the operation immediately after oscillation starts.

**Figure 4.5-1 Operation immediately after oscillation starts**



## 4.6 Connection of an Oscillator or an External Clock to the Microcontroller

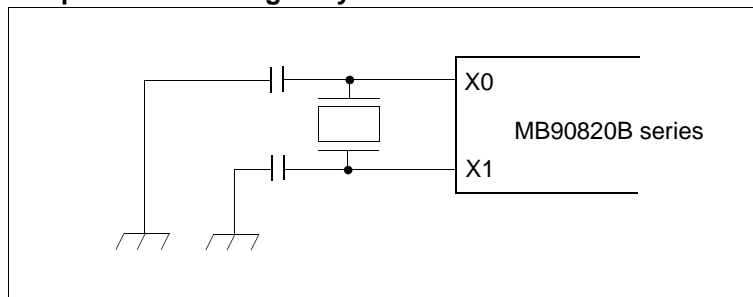
The F<sup>2</sup>MC-16LX microcontroller contains a system clock generation circuit. Connecting an external oscillator to this circuit generates the system clock. Alternatively, an externally generated clock can be input to the microcontroller.

### ■ Connection of an Oscillator or an External Clock to the Microcontroller

- Example of connecting a crystal or ceramic oscillator to the microcontroller

Connect a crystal or ceramic oscillator as shown in the example in Figure 4.6-1.

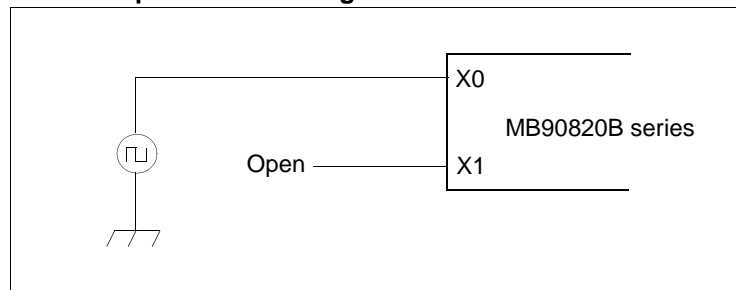
Figure 4.6-1 Example of connecting a crystal or ceramic oscillator to the microcontroller



- Example of connecting an external clock to the microcontroller

As shown in Figure 4.6-2, connect an external clock to pin X0. Pin X1 must be open.

Figure 4.6-2 Example of connecting an external clock to the microcontroller



# **CHAPTER 5**

---

# **CLOCK SUPERVISOR**

**This chapter describes the functions and operations of the clock supervisor.**

**(This feature is available for MB90F828B only )**

- 5.1 Overview of Clock Supervisor
- 5.2 Configuration of Clock Supervisor
- 5.3 Registers of Clock Supervisor
- 5.4 Operations of Clock Supervisor
- 5.5 Precautions when Using Clock Supervisor

## 5.1 Overview of Clock Supervisor

---

**The clock supervisor prevents the situation which is out of control, when main clock oscillation has halted. This function switches to an CR clock generated with internal CR oscillator circuit, if main clock oscillation has halted.**

---

### ■ Overview of Clock Supervisor

- The clock supervisor monitors the main clock oscillation and generates an internal reset if it detects that the oscillation has halted. In this case, the clock supervisor switches to the internal CR clock.

The reset source register (RSRR) can be used to determine whether a reset was triggered by the clock supervisor.

- A main clock oscillation halt is detected if the rising edge of the main clock is not detected for 4 CR clock cycles. The clock supervisor may detect incorrectly, if main clock is longer than 4 CR clock cycles.
  - Setting registers enable to prohibit the reset output.
  - While the clock stops in main clock stop mode, clock monitoring is disabled.
- 

#### Note:

Refer to the data sheet for the period and other details about the CR clock.

---

**MB90820B Series****5.2 Configuration of Clock Supervisor**

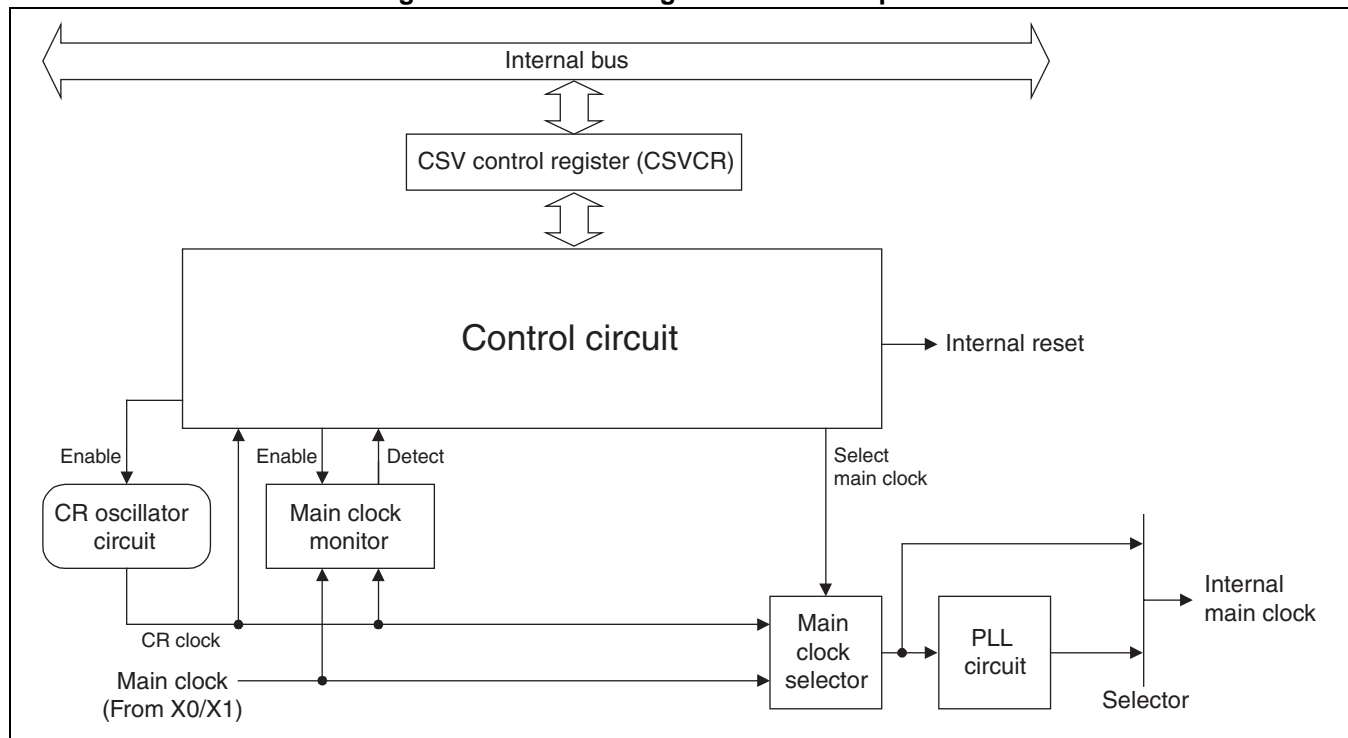
The clock supervisor consists of the following blocks:

- Control circuit
- CR oscillator circuit
- Main clock monitor
- Main clock selector
- CSV control register (CSVCR)

### ■ Block Diagram of Clock Supervisor

Figure 5.2-1 shows a block diagram of the clock supervisor.

**Figure 5.2-1 Block Diagram of Clock Supervisor**



- **Control circuit**

This block controls the clocks, resets, and other settings based on the information in the CSV control register (CSVCR).

- **CR oscillator circuit**

This block is a internal CR oscillator circuit. The oscillation can be turned on or off via a control signal from the control circuit. This also serves as an internal clock after a clock halt is detected.

- **Main clock monitor**

This block monitors whether the main clock halts.

- **Main clock selector**

This block outputs the CR clock as the internal main clock upon detection of a main clock halt.

- **CSV control register (CSVCR)**

This block is used to control clock monitoring and CR clock and to check information on halt detection.

MB90820B Series

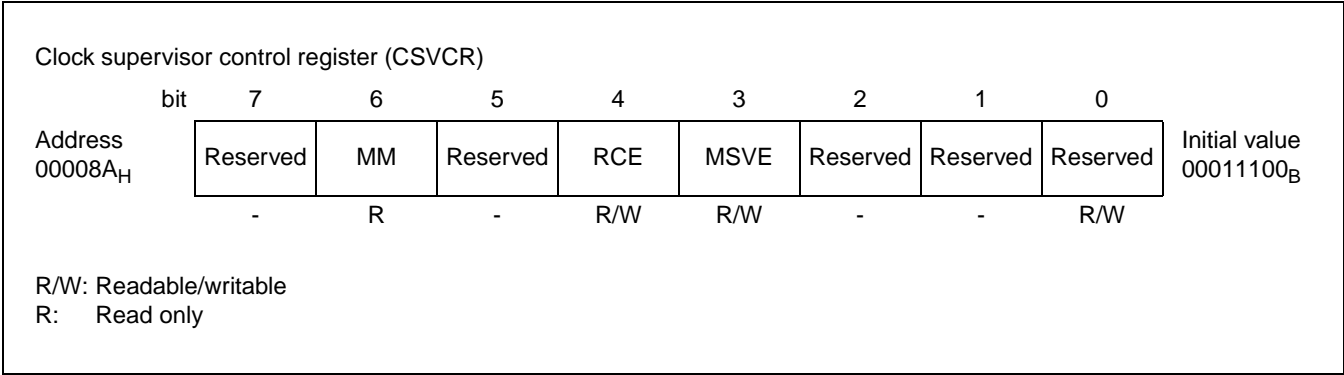
5.3 Registers of Clock Supervisor

This section describes the clock supervisor registers.

■ Clock Supervisor Register

Figure 5.3-1 shows the register of the clock supervisor.

Figure 5.3-1 Clock Supervisor Register





## 5.3.1 Clock Supervisor Control Register (CSVCR)

The clock supervisor control register (CSVCR) is used to enable the various functions and to check the status.

### ■ Clock Supervisor Control Register (CSVCR)

Figure 5.3-2 Clock Supervisor Control Register (CSVCR)

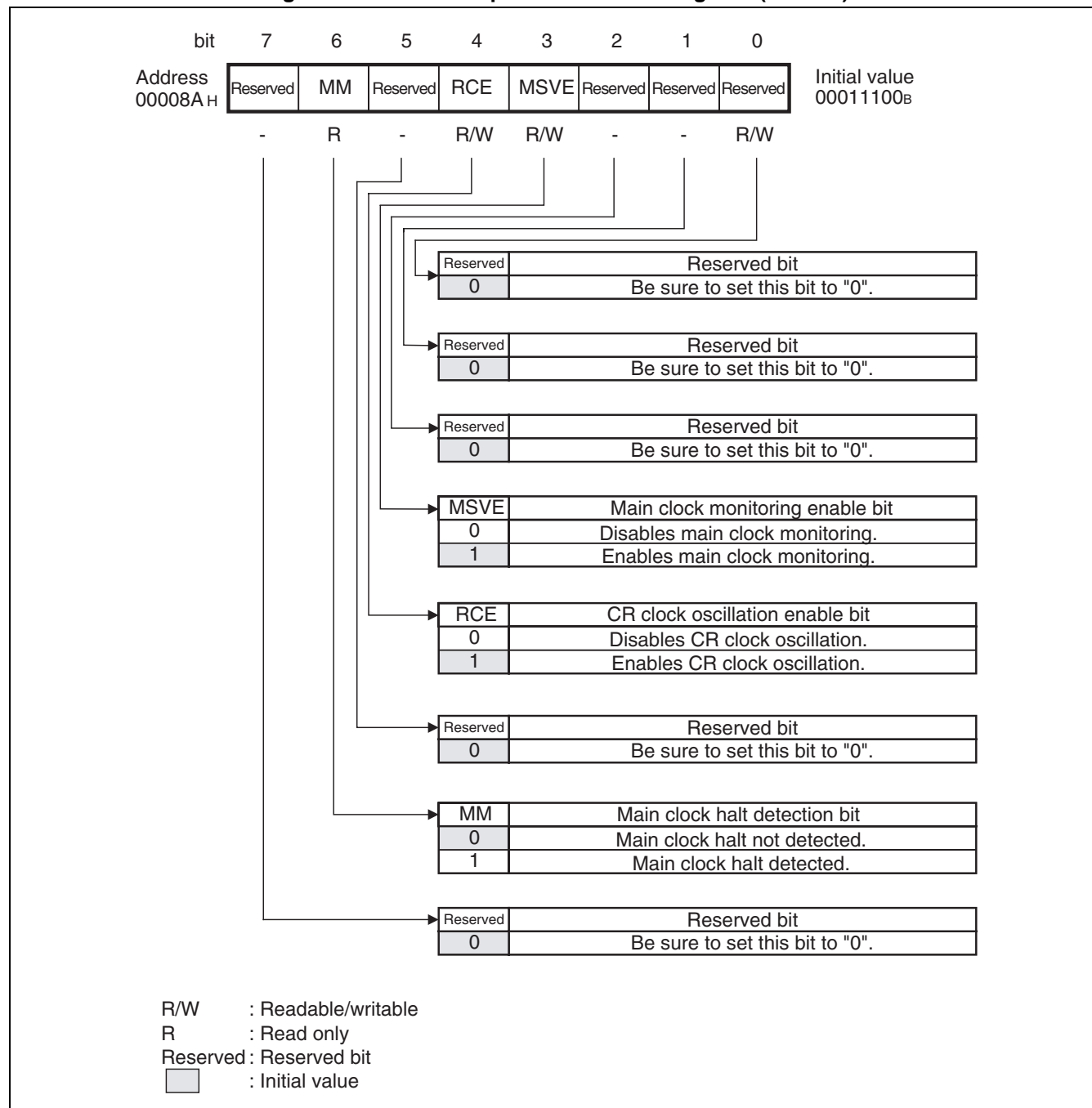


Table 5.3-1 Functions of Bits in Clock Supervisor Control Register (CSVCR)

Bit name		Function
bit7	Reserved bit	This bit is reserved. Write "0" to this bit. The read value is always "0".
bit6	MM: Main clock halt detection bit	This bit is read-only, and this bit indicates that a main clock oscillation halt has been detected. <b>When set to "1"</b> : The bit indicates that a main clock oscillation halt has been detected. <b>When set to "0"</b> : The bit indicates that no main clock oscillation halt has been detected. Writing "1" to this bit does not affect the operation.
bit5	Reserved bit	This bit is reserved. Write "0" to this bit. The read value is always "0".
bit4	RCE: CR clock oscillation enable bit	This bit enables CR oscillation. <b>When set to "1"</b> : The bit enables oscillation. <b>When set to "0"</b> : The bit disables oscillation. Before writing "0" to this bit, make sure that the clock monitor function has been disabled with the MM and SM bits set to "0".
bit3	MSVE: Main clock monitoring enable bit	This bit enables the monitoring of main clock oscillation. <b>When set to "1"</b> : The bit enables main clock monitoring. <b>When set to "0"</b> : The bit disables main clock monitoring. This bit is set to "1" only when a power-on reset occurs.
bit2 to bit0	Reserved bit	This bit is reserved. Write "0" to this bit. The read value is always "0".

## Note:

When the power is turned on, the clock supervisor starts monitoring after the oscillation stabilization wait time for the main clock elapses. The oscillation stabilization wait time of the main clock must therefore be longer than the time required for the clock supervisor to start operating.

## 5.4 Operations of Clock Supervisor

---

**This section describes the operations of the clock supervisor.**

---

### ■ Operations of Clock Supervisor

The clock supervisor monitors the main clock oscillation. If main clock oscillation has halted, the device switches to an CR clock and generates a reset.

The following describes the operation in each clock mode.

#### ● Main clock oscillation halt in main clock mode

The clock supervisor detect that main clock oscillation has halted, if no rising edge is detected on the main clock for 4 CR clock cycles in main clock mode.

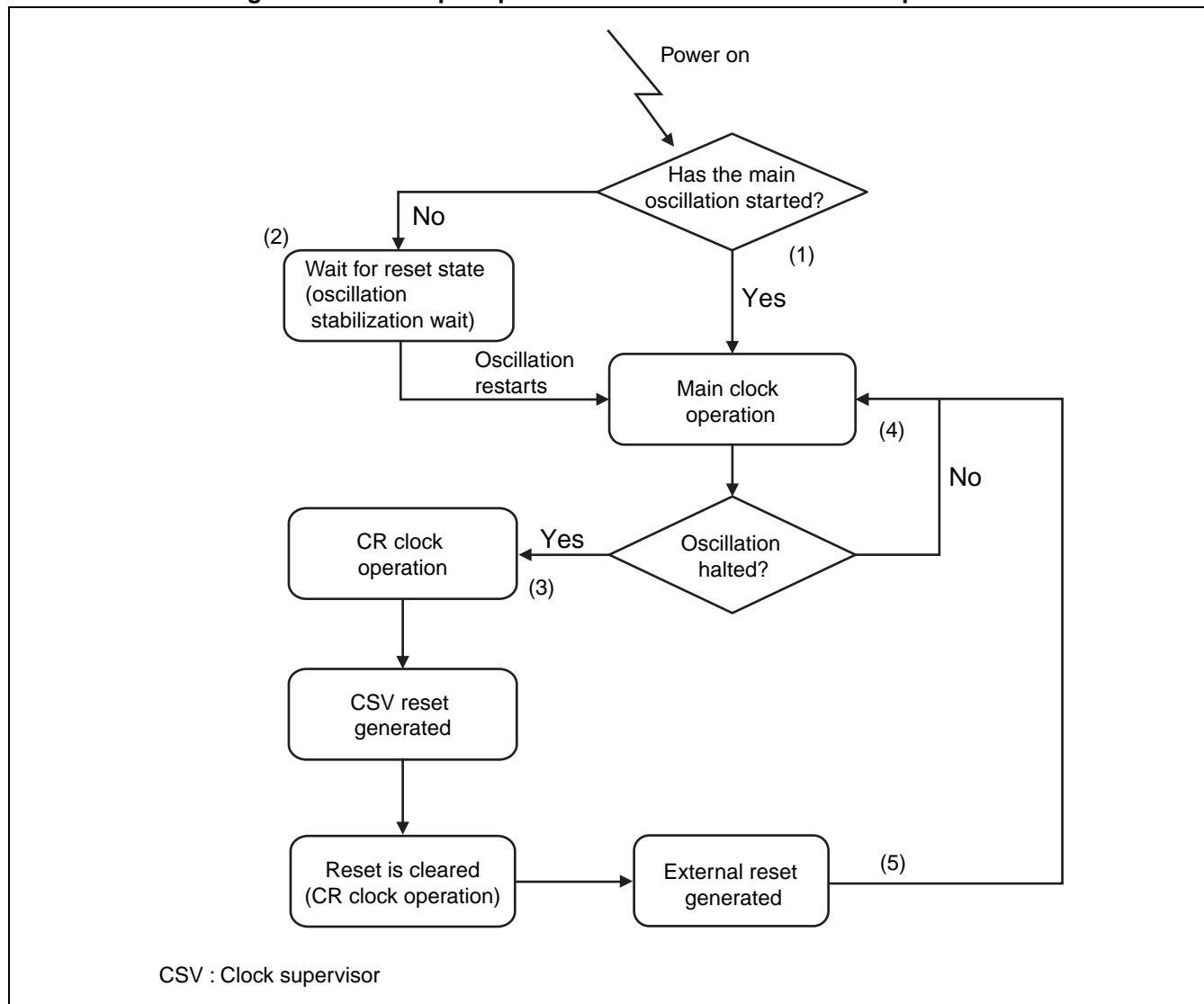
If a main clock halt is detected, a reset is generated and the main clock switches to the CR clock.

The clock supervisor may detect incorrectly, if main clock is a low speed (longer than 4 CR clock cycles). It results from using the CR clock for detecting that main clock oscillation have halted.

The clock supervisor does not detect the main clock during stop mode.

■ Example Operation Flowchart for the Clock Supervisor

Figure 5.4-1 Example Operation Flowchart for the Clock Supervisor



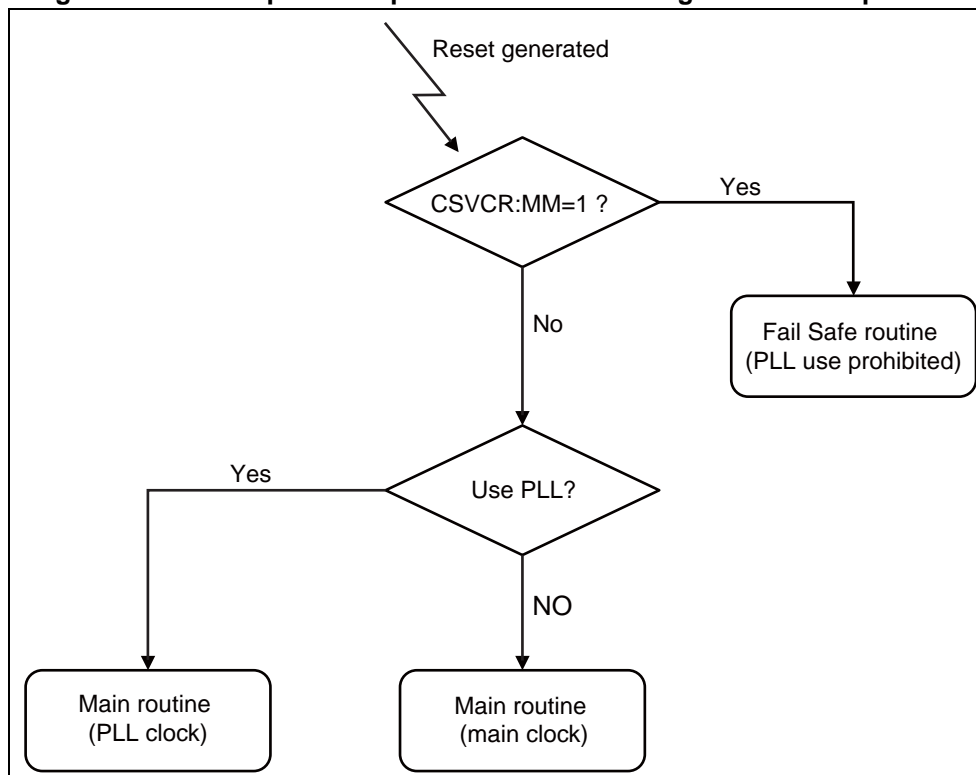
1. After the power is turned on, the main clock operation starts after the oscillation stabilization wait time generated by the main oscillation has elapsed.
2. If the main clock halts at power on, the device remains in the reset state (oscillation stabilization wait state). The operation changes to the main clock, after the oscillation restarts and the oscillation stabilization wait time elapsed.
3. If an oscillation halt is detected during main clock operation, the operating clock is switched to the CR clock and a reset is generated.
4. If the main oscillation continues (oscillation does not halt), the device continues to run using the main clock.
5. If an external reset occurs during the CR clock operation, operation changes to the main clock. However, if the oscillation is halted at this time, another CSV reset is generated and the device returns to CR clock operation.

### ■ Example Startup Flowchart when using the Clock Supervisor

Inserting checking process of the main clock stop detection bit (CSVCR:MM) enables user programs to control the Fail Safe routine.

Figure 5.4-2 shows the example startup flowchart when using the clock supervisor.

**Figure 5.4-2 Example Startup Flowchart when using the Clock Supervisor**



## 5.5 Precautions when Using Clock Supervisor

---

Take note of the following points when using the clock supervisor.

---

### ■ Precautions when using the Clock Supervisor

- Operation of the clock supervisor at power on

When the power is turned on, the clock supervisor starts monitoring after the oscillation stabilization wait time for the main clock has elapsed. Therefore, unless the operation continues for longer than the oscillation stabilization wait time for the main clock, the clock supervisor will not operate.

- Transition to CR clock mode

Do not turn on the PLL after changing to CR clock mode.

As the frequency is below the lower limit for the input frequency of the PLL circuit, the PLL operation will not be guaranteed.

- Disabling the CR oscillation

Do not use the CR oscillation enable bit (CSVCR:RCE) to disable the CR oscillation during CR clock mode.

As this halts the internal clock, it may result in deadlock.

- Initializing the main clock halt detection bit

The main clock halt detection bit (CSVCR:MM) is initialized by a power-on reset nor external reset only. The bit is not initialized by neither a watchdog reset, software reset, nor CSV reset. Accordingly, the device remains in CR clock mode if one of these resets occurs during CR clock mode.

- Verifying reset execution using clock supervisor function

To verify if a reset was executed using the clock monitoring function, read the WDTC register with software to check the reset factor. When the ERST (bit 4 of WDTC) is set, generation of a reset via the external pin or clock supervisor reset can be verified.

If the MM bit (bit 6 of CSVCR) is "0", the reset factor is external reset. When the MM is "1", it is the loss of main clock.



# **CHAPTER 6**

---

# ***LOW-POWER CONSUMPTION MODE***

**This chapter describes the low-power consumption mode of MB90820B series microcontrollers.**

- 6.1 Low-Power Consumption Mode
- 6.2 Block Diagram of the Low-Power Consumption Control Circuit
- 6.3 Low-Power Consumption Mode Control Register (LPMCR)
- 6.4 CPU Intermittent Operation Mode
- 6.5 Standby Mode
- 6.6 State Change Diagram
- 6.7 State of Pins in Standby Mode and During Reset
- 6.8 Usage Notes on Low-Power Consumption Mode



## 6.1 Low-Power Consumption Mode

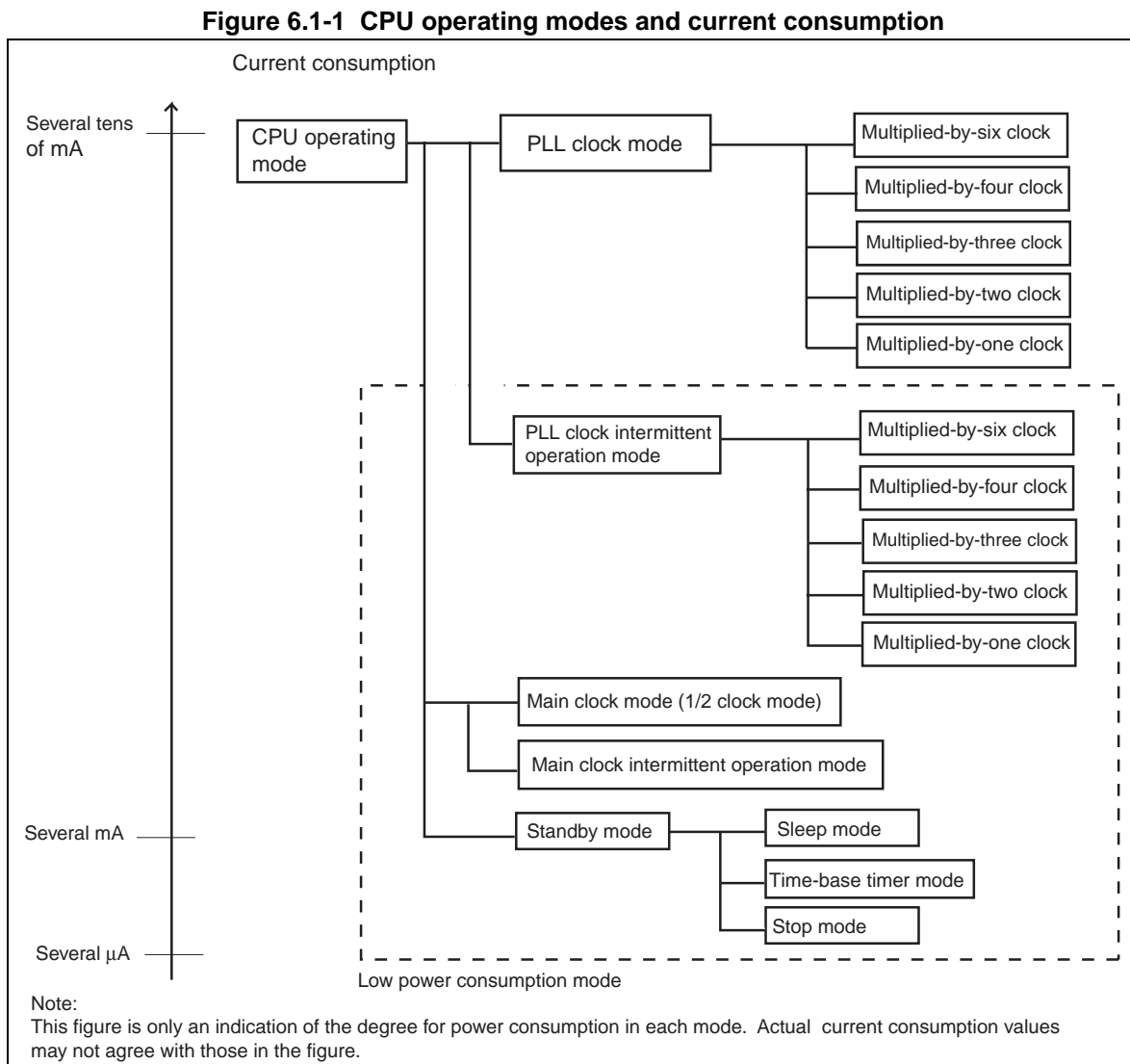
F<sup>2</sup>MC-16LX microcontrollers have the following CPU operating modes, any of which can be used depending on the operating clock selection and clock operation control:

- Clock mode (PLL clock mode and main clock mode)
- CPU intermittent operation mode (PLL clock intermittent operation mode and main clock intermittent operation mode)
- Standby mode (sleep, time-base timer and stop modes)

All modes other than PLL clock mode are low-power consumption mode.

### ■ CPU Operating Modes and Current Consumption

Figure 6.1-1 shows the relation between the CPU operating modes and current consumption



## MB90820B Series

### ■ Clock Mode

#### ● PLL clock mode

A PLL clock that is a multiple of the oscillation clock (HCLK) frequency is used to operate the CPU and peripheral functions.

#### ● Main clock mode

The main clock, with a frequency one-half that of the oscillation clock (HCLK), is used to operate the CPU and peripheral functions. In main clock mode, the PLL multiplier circuit is inactive.

---

Reference:

See Section "4.1 Clock", for details about clock mode.

---

### ■ CPU Intermittent Operation Mode

CPU intermittent operation mode causes the CPU to operate intermittently, while high-speed clock pulses are supplied to peripheral functions, reducing power consumption. In CPU intermittent operation mode, intermittent clock pulses are only applied to the CPU when it is accessing a register, an internal memory, a peripheral function, or an external unit.

### ■ Standby Mode

In standby mode, the low-power consumption control circuit stops supplying the clock to the CPU (sleep mode) or the CPU and peripheral functions (time-base timer mode), or stops the oscillation clock itself (stop mode), reducing power consumption.

#### ● PLL sleep mode

PLL sleep mode is activated to stop the CPU operating clock when the microcontroller enters PLL clock mode; other components continue to operate on the PLL clock.

#### ● Main sleep mode

Main sleep mode is activated to stop the CPU operating clock when the microcontroller enters main clock mode; other components continue to operate on the main clock.

#### ● PLL time-base timer mode

PLL time-base timer mode causes microcontroller operation, with the exception of the oscillation clock, PLL clock, and time-base timer, to stop. All functions other than the time-base timer are deactivated.

#### ● Main time-base timer mode

Main time-base timer mode causes microcontroller operation, with the exception of the oscillation clock, main clock, and the time-base timer, to stop. All functions other than the time-base timer are deactivated.

#### ● Stop mode

Stop mode causes the source oscillation (HCLK) to stop. All functions are deactivated.

---

Note :

Because stop mode turns the oscillation clock off, this mode saves most power while data is being retained.

---

# MB90820B Series

## 6.2 Block Diagram of the Low-Power Consumption Control Circuit

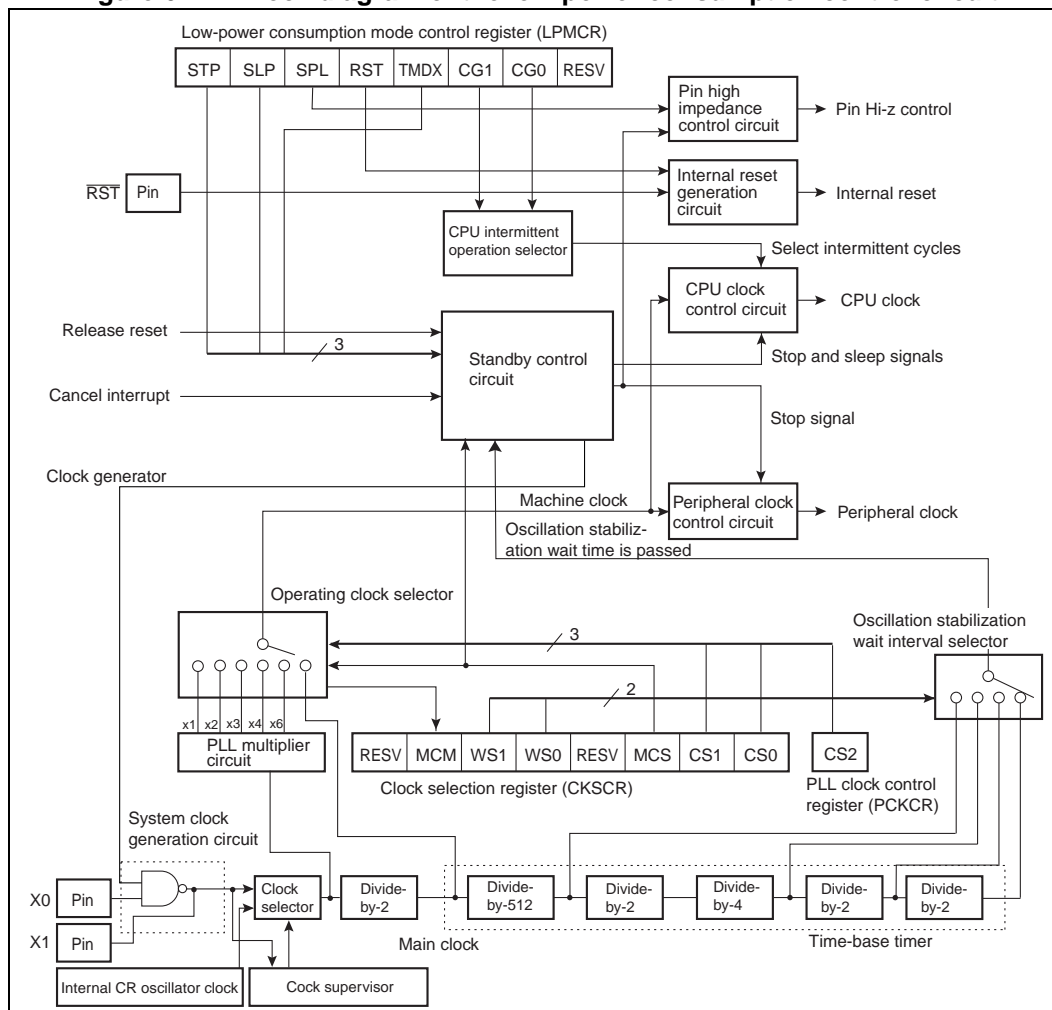
The low-power consumption control circuit consists of the following seven blocks:

- CPU intermittent operation selector
- Standby clock control circuit
- CPU clock control circuit
- Peripheral clock control circuit
- Pin high-impedance control circuit
- Internal reset generation circuit
- Low-power consumption mode control register (LPMCR)

### ■ Block Diagram of the Low-Power Consumption Control Circuit

Figure 6.2-1 shows the block diagram of the low-power consumption control circuit.

**Figure 6.2-1 Block diagram of the low-power consumption control circuit**



- CPU intermittent operation selector

This selector selects the number of clock pulses the CPU is to be halted during CPU intermittent operation mode.

- Standby control circuit

The standby control circuit controls the CPU clock control circuit and the peripheral clock control circuit, and turns the low-power consumption mode on and off.

- CPU clock control circuit

This circuit controls the clocks supplied to the CPU. This circuit controls the clocks supplied to peripheral functions for the peripheral clock control.

- Peripheral clock control circuit

This circuit controls the clocks supplied to peripheral functions.

- Pin high-impedance control circuit

This circuit makes the external pins high-impedance when the microcontroller enters time-base timer mode and stop mode.

For the pins with the pull-up option, this circuit disconnects the pull-up resistor when the microcontroller enters stop mode.

- Internal reset generation circuit

This circuit generates an internal reset signal.

- Low-power consumption mode control register (LPMCR)

This register is used to switch to and release standby mode and to set the CPU intermittent operation function.

## MB90820B Series

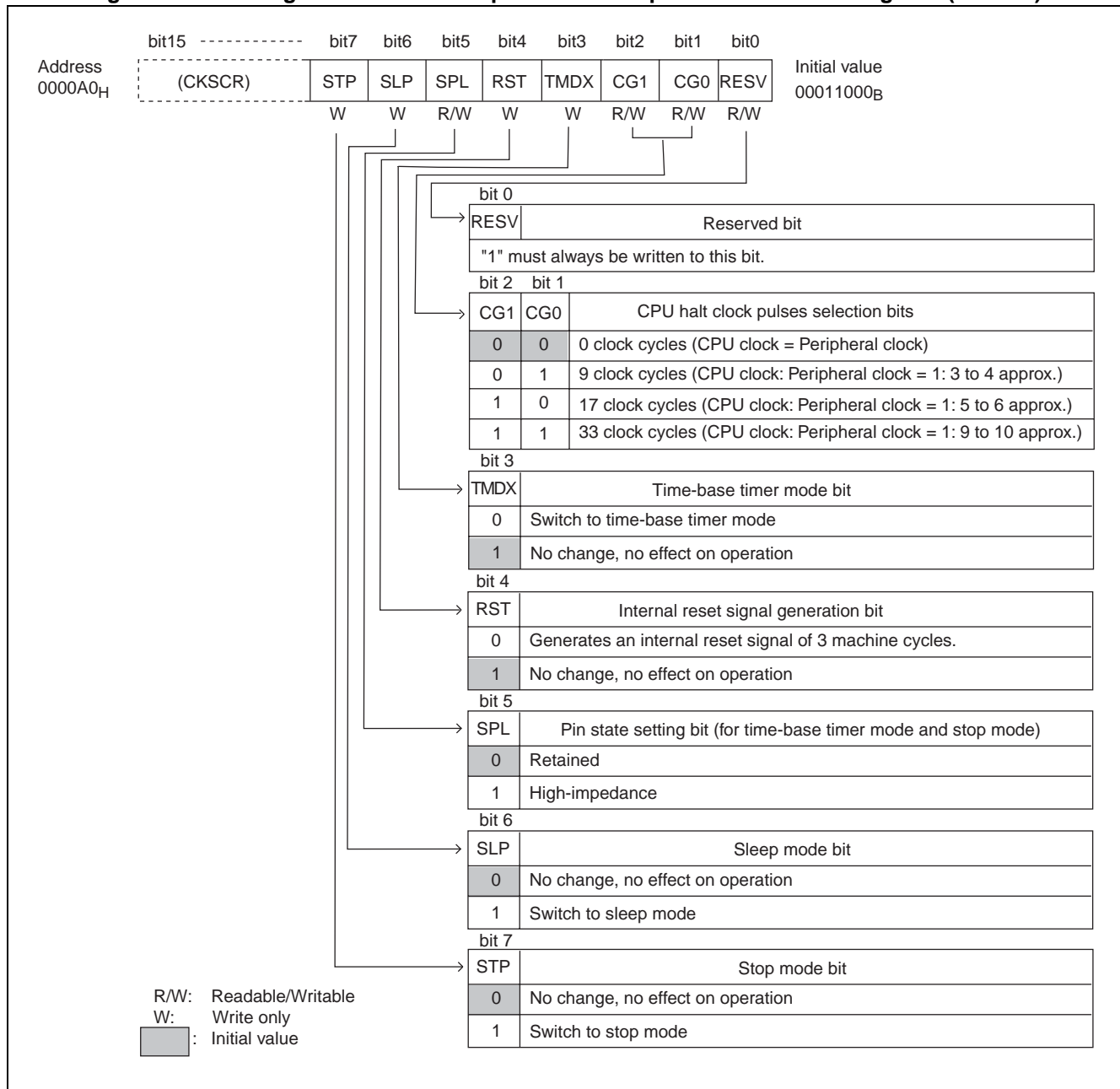
## 6.3 Low-Power Consumption Mode Control Register (LPMCR)

The low-power consumption mode control register (LPMCR) switches to or releases low-power consumption mode. It is also used to set the number of CPU clock pulses the CPU is to be halted during CPU intermittent mode.

### Low-Power Consumption Mode Control Register (LPMCR)

Figure 6.3-1 shows the configuration of the low-power consumption mode control register (LPMCR).

Figure 6.3-1 Configuration of the low-power consumption mode control register (LPMCR)



**Table 6.3-1 Function description of each bit in the low-power consumption mode control register (LPMCR)**

Bit name		Function
bit7	STP: Stop mode bit	<ul style="list-style-type: none"> <li>This bit indicates switching to stop mode.</li> <li><b>When "1" is written to this bit:</b> the mode switches to stop mode.</li> <li><b>Writing "0" to this bit:</b> no effect on operation.</li> <li>This bit is cleared to "0" by a reset or by release of stop state.</li> <li>The read value of this bit is always "0".</li> </ul>
bit6	SLP: Sleep mode bit	<ul style="list-style-type: none"> <li>This bit indicates switching to sleep mode.</li> <li><b>When "1" is written to this bit:</b> the mode switches to sleep mode.</li> <li><b>Writing "0" to this bit:</b> no effect on operation.</li> <li>This bit is cleared to "0" by a reset or by release of sleep mode.</li> <li>The read value of this bit is always "0".</li> </ul>
bit5	SPL: Pin state setting bit (for time-base timer mode and stop mode)	<ul style="list-style-type: none"> <li>This bit is enabled while either time-base timer mode or stop mode is in effect.</li> <li><b>When this bit is "0":</b> the level of the external pins is retained.</li> <li><b>When this bit is "1":</b> the status of the external pins changes to high-impedance.</li> <li>This bit is initialized to "0" by a reset.</li> </ul>
bit4	RST: Internal reset signal generation bit	<ul style="list-style-type: none"> <li>When "0" is written to this bit, an internal reset signal of 3 machine cycles is generated.</li> <li>Writing "1" to this bit has no effect on operation.</li> <li>The read value of this bit is always "1".</li> </ul>
bit3	TMDX: Time-base timer mode bit	<ul style="list-style-type: none"> <li>This bit indicates switching to time-base timer mode.</li> <li>When "0" is written to this bit, the mode switches to time-base timer mode.</li> <li>Writing "1" to this bit has no effect on operation.</li> <li>This bit is set to "1" by a reset or by release of time-base timer mode.</li> <li>The read value of this bit is always "1".</li> </ul>
bit2, bit1	CG1, CG0: CPU halt clock pulses selection bits	<ul style="list-style-type: none"> <li>These bits set the number of CPU halt clock pulses for the CPU intermittent operation function.</li> <li>The clock supplied to the CPU is stopped after the execution of every instruction for the specified number of clock pulses.</li> <li>Selection can be made from among four different clock pulses.</li> <li>These bits are initialized to "00<sub>B</sub>" by a power-on or watchdog timer reset. Other resets do not initialize these bits.</li> </ul>
bit0	RESV: Reserved bit	<p><b>Note:</b></p> <p>"1" must always be written to this bit.</p>

**Note:**

If "1" is written to the STP bit and SLP bit, and "0" is written to TMDX bit at the same time, switching to stop mode takes the highest priority, then time-base timer mode and sleep mode have the lowest priority.

## MB90820B Series

### ■ Access to the Low-Power Consumption Mode Control Register

Writing to the low-power consumption mode control register causes a change to a low-power consumption mode (stop mode, sleep mode, time-base timer mode, or watch mode). In this case, please use one of the instructions specified in Table 6.3-2.

Always insert the following instructions (indicated by the box below) immediately after the instruction from Table 6.3-2 for switching to a low-power consumption mode.

MOV LPMCR,#H'xx	; Instruction from Table 6.3-2 for switching to a low-power consumption mode
NOP	
NOP	
JMP \$+3	; Jump to next instruction
MOV A,#H'10	; Next instruction

The operation when recovering from low-power consumption mode is not guaranteed unless the instructions shown in the box above are used.

If accessing the low-power consumption control register from a C program, refer to the notes in "■ Points to Note when Accessing the Low-power Consumption Mode Control Register (LPMCR) to Switch to Standby Mode" in "6.8 Usage Notes on Low-Power Consumption Mode".

When writing to the low-power consumption mode control register (LPMCR) using a word-length instruction, ensure that you write to an even-numbered address. Writing to an odd-numbered address to switch to low-power consumption mode may result in misoperation.

No restrictions apply on what instructions to use when performing operations on functions other than those listed in Table 6.3-2.

**Table 6.3-2 Instructions to be used for switching to low-power consumption mode**

MOV io, #imm8	MOV dir, #imm8	MOV eam, #imm8	MOV eam, Ri
MOV io, A	MOV dir, A	MOV addr16, A	MOV eam, A
MOV @RLi+disp8, A			
MOVW io, #imm16	MOVW dir, #imm16	MOVW eam, #imm16	MOVW eam, RWi
MOVW io, A	MOVW dir, A	MOVW addr16, A	MOVW eam, A
MOVW @RLi+disp8, A			
SETB io:bp	SETB dir:bp	SETB addr16:bp	
CLRB io:bp	CLRB dir:bp	CLRB addr16:bp	



## 6.4 CPU Intermittent Operation Mode

**CPU intermittent operation mode is used for intermittent operation of the CPU while external buses and peripheral functions continue to operate at high speed. Its purpose is to reduce power consumption.**

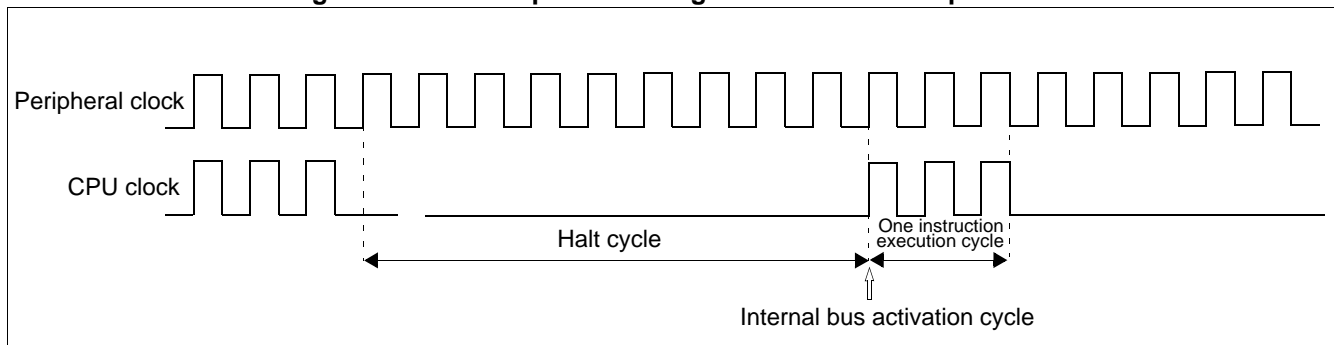
### ■ CPU Intermittent Operation Mode

CPU intermittent operation mode halts the supply of the clock to the CPU for a certain period. The halt occurs after the execution of every instruction that accesses a register, internal memory (ROM and RAM), I/O, peripheral functions, and the external bus. Internal bus cycle activation is therefore delayed. While a high rate of peripheral clock pulses are supplied to the peripheral functions, the rate of CPU execution is reduced, enabling processing with low-power consumption.

- The CG1 and CG0 bits of the low-power consumption mode control register (LPMCR) are used to select the number for clock pulses per halt cycle of the clock supplied to the CPU.
- External bus operation uses the same clock as that used for peripheral functions.
- Instruction execution time in CPU intermittent mode can be calculated. A correction value should be obtained by multiplying the number of times instructions that access a register, internal memory, peripheral functions, and the external bus are executed by the number of clock pulses per halt cycle. Add this correction value to the normal execution time.

Figure 6.4-1 shows the operating clock pulses during CPU intermittent operation mode.

**Figure 6.4-1 Clock pulses during CPU intermittent operation**



**MB90820B Series****6.5 Standby Mode**

**Standby mode includes the sleep (PLL sleep and main sleep), time-base timer, and stop modes.**

### ■ Operating Status During Standby Mode

Figure 6.5-1 summarizes the operating statuses during standby mode.

**Table 6.5-1 Operation statuses during standby mode**

Standby mode		Condition for switch	Oscillation	Clock	CPU	Peripheral	Pin	Release event
Sleep mode	PLL sleep mode	MCS = 0 SLP = 1	Active	Active	Inactive	Active	Active	Reset or Interrupt
	Main sleep mode	MCS = 1 SLP = 1						
Time-base timer mode	PLL time-base timer mode (SPL = 0)	MCS = 0 TMDX = 0				Hold		
	PLL time-base timer mode (SPL = 1)						Hi-Z	
	Main time-base timer mode (SPL = 0)	MCS = 1 STP = 1						
	Main time-base timer mode (SPL = 1)						Hi-Z	
Stop mode	Main/PLL stop mode (SPL = 0)	MCS = x STP = 1	Inactive	Inactive		Inactive		
	Main/PLL stop mode (SPL = 1)						Hi-Z	

\*: Only the time-base timer is active.

SPL: Pin state setting bit of low-power consumption mode control register (LPMCR)

SLP: Sleep mode bit of LPMCR

STP: Stop mode bit of LPMCR

TMDX: Time-base timer mode bit of LPMCR

MCS: Machine clock selection bit of clock selection register (CKSCR)

Hi-Z: High-impedance

### 6.5.1 Sleep Mode

---

**Sleep mode causes the CPU operating clock to stop while other components continue to operate.**

**When the low-power consumption mode control register (LPMCR) indicates a switch to sleep mode, a switch to PLL sleep mode occurs if PLL clock mode has been set. Alternatively, a switch to main sleep mode occurs if main clock mode has been set.**

---

#### ■ Switching to Sleep Mode

Writing "1" to the SLP and TMDX bits of LPMCR and "0" to the STP bit of LPMCR triggers a switch to sleep mode.

At this time, if the MCS bit of the clock selection register (CKSCR) is "0", the microcontroller enters PLL sleep mode. If the MCS bit of CKSCR is "1", the microcontroller enters main sleep mode.

---

#### Note :

Since the STP/TMDX bit setting overrides the SLP bit setting when "1" is written to the SLP and STP, and "0" to TMDX bit at the same time, the mode switches to stop/time-base timer mode.

---

#### ● Data retention function

In sleep mode, the contents of dedicated registers, such as accumulators and internal RAM, are retained.

#### ● Operation during an interrupt request

Writing "1" to the SLP bit of LPMCR during an interrupt request does not trigger a switch to sleep mode. If the CPU does not accept the interrupt, the CPU executes the next instruction. If the CPU accepts the interrupt, CPU operation immediately branches to the interrupt processing routine.

#### ● Status of pins

During sleep mode, all pins retain the state they had immediately before the switch to sleep mode. The once exceptions are the pins used for bus input/output or bus control.

#### ■ Release of Sleep Mode

The low-power consumption control circuit releases sleep mode. Releasing is caused by the input of a reset or by an interrupt.

#### ● Return to normal mode by a reset

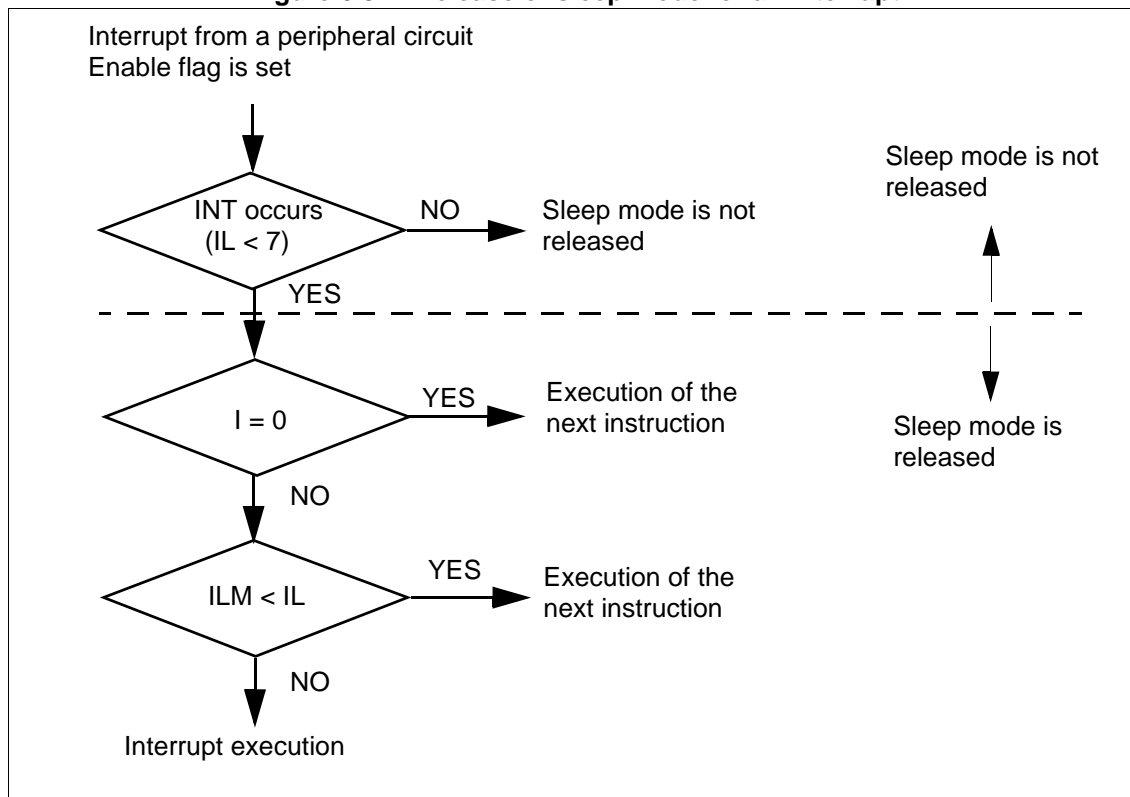
When sleep mode is released by a reset, the microcontroller is placed in the reset state on release from sleep mode.

● Return to normal mode by an interrupt

If an interrupt request higher than interrupt level 7 is issued from a peripheral circuit during sleep mode, sleep mode is released. After release, the CPU handles the interrupt as it would any other interrupt. The CPU executes processing according to the settings of the I flag of the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR). If that interrupt is accepted, the CPU executes interrupt processing. If the interrupt is not accepted, the CPU resumes execution with the instruction that follows the instruction in which switching to sleep mode was specified.

Figure 6.5-1 shows the release of sleep mode for an interrupt.

**Figure 6.5-1 Release of sleep mode for an interrupt**



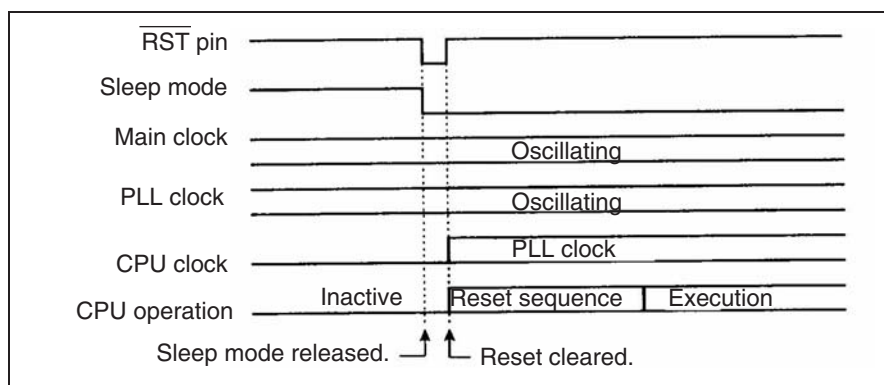
Note :

When interrupt processing is executed normally, the CPU first executes the instruction that follows the instruction in which switching to sleep mode was specified. The CPU then proceeds to interrupt processing.

● Return to normal mode from PLL sleep mode by an external reset

During PLL sleep mode, the main clock and the PLL clock generate clock pulses. Since an external reset does not initialize the MCS bit in the clock selection register (CKSCR) to "1", PLL clock mode remains selected (MCS of CKSCR = 0). On return from PLL sleep mode by an external reset, the CPU starts operation using the PLL clock immediately after PLL sleep mode is released as shown in Figure 6.5-2.

**Figure 6.5-2 Release of PLL sleep mode (by external reset)**



**MB90820B Series****6.5.2 Time-base Timer Mode**


---

**Time-base timer mode causes the microcontroller operation to stop with the exception of the source oscillation and the time-base timer. All functions other than time-base timer are deactivated.**

---

**■ Switching to Time-base Timer Mode**

Writing "0" to the TMDX and STP bits of LPMCR triggers a switch to time-base timer mode.

At this time, if the MCS bit of the clock selection register (CKSCR) is "0", the microcontroller enters PLL time-base timer mode. If the MCS bit of CKSCR is "1", the microcontroller enters main time-base timer mode.

---

Note :

Since the STP bit setting overrides the TMDX bit setting when "0" is written to the TMDX and STP bits at the same time, the mode switches to stop mode.

---

- Data retention function

In time-base timer mode, the contents of dedicated registers, such as accumulators and internal RAM, are retained.

- Operation during an interrupt request

Writing "0" to the TMDX bit of LPMCR during an interrupt request does not trigger switching to time-base timer mode.

- Status of pins

Selection of whether the external pins retain the state they had immediately before switching to time-base timer mode or go to high-impedance with switching to this mode can be controlled by the SPL bit of LPMCR.

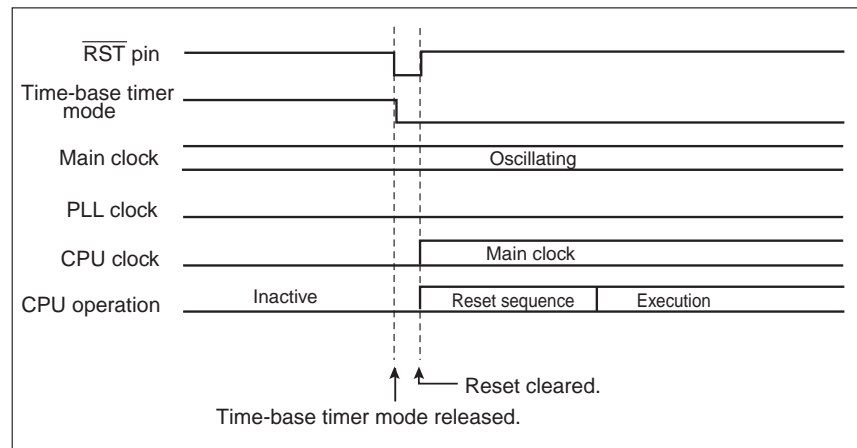
**■ Release of Time-base Timer Mode**

The low-power consumption control circuit releases time-base timer mode. Release is caused by input of a reset or an interrupt. If time-base timer mode is released by a reset, the microcontroller is placed in the reset state after its release from time-base timer mode.

- Return to normal mode by a reset

If time-base timer mode is released by a reset, the microcontroller is placed in the reset state after release from time-base timer mode. The timerbase timer mode is initialized to the main clock mode by a reset.

Figure 6.5-3 shows the operation for return to normal mode from time-base timer mode triggered by an external reset.

**Figure 6.5-3 Release of time-base timer mode (by an external reset)**

#### ● Return to normal mode by an interrupt

If an interrupt request higher interrupt than level 7 is issued from a peripheral circuit in time-base timer mode (when IL2, IL1 and IL0 of the interrupt control register (ICR) are set to a value other than "111<sub>B</sub>"), the low-power consumption control circuit releases time-base timer mode. After the release, the CPU handles the interrupt as it would normal interrupt. The CPU executes processing according to the settings of the I flag of the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR). If the interrupt is accepted, the CPU executes interrupt processing. If the interrupt is not accepted, the CPU resumes execution with the instruction that follows the instruction in which switching to time-base timer mode was specified.

#### Note :

When interrupt processing is executed normally, the CPU first executes the instruction that follows the instruction in which switching to sleep mode was specified. The CPU then proceeds to interrupt processing.

**MB90820B Series****6.5.3 Stop Mode**


---

**Stop mode causes the source oscillation to stop and deactivates all functions. It therefore saves the most power saving while data is being retained.**

---

**■ Switching to Stop Mode**

Writing "1" to the STP bit of LPMCR triggers a switch to stop mode.

At this time, if the MCS bit of the clock selection register (CKSCR) is "0", the microcontroller enters PLL stop mode. If the MCS bit of CKSCR is "1", the microcontroller enters main stop mode.

- Data retention function

In stop mode, the contents of dedicated registers, such as accumulators and internal RAM, are retained.

- Operation during an interrupt request

Writing "1" to the STP bit of LPMCR during an interrupt request does not trigger switching to stop mode.

- Pin state setting

Selection of whether the external pins retain the state they had immediately before switching to stop mode or go to high-impedance with switching to stop mode can be controlled by the SPL bit of LPMCR.

**■ Release of Stop Mode**

The low-power consumption control circuit releases stop mode. The release is caused by input of a reset or by an interrupt.

Because the oscillation of the operating clock is halted before return to normal mode from stop mode, the low-power consumption control circuit puts the microcontroller into the oscillation stabilization wait state, then releases stop mode.

- Return to normal mode by a reset

When stop mode is released by a reset cause, the microcontroller is placed in the oscillation stabilization wait and reset state after release from stop mode. The reset sequence proceeds after the oscillation stabilization wait interval has elapsed.

- Return to normal mode by a interrupt

If an interrupt request higher than interrupt level 7 is issued from a peripheral circuit during stop mode (when IL2, IL1, and IL0 of the interrupt control register (ICR) are set to a value other than "111<sub>B</sub>"), the low-power consumption control circuit releases stop mode. After release, the CPU handles the interrupt as it would any other interrupts. However, the CPU starts after the main clock oscillation stabilization wait interval specified by the WS1 and WS0 bits of the clock selection register (CKSCR) has elapsed. The CPU executes processing according to the settings of the I flag in the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR). If the interrupt is accepted, the CPU executes interrupt processing. If the interrupt is not accepted, the CPU resumes the execution with the instruction that follows the instruction in which switching to stop mode was specified.

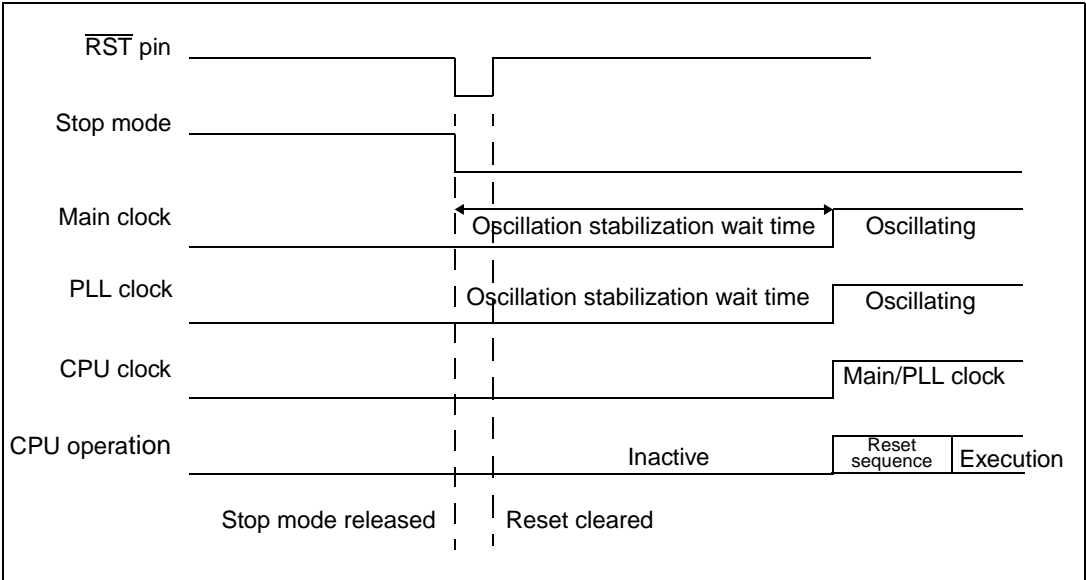


Note :

When interrupt processing is executed normally, the CPU first executes the instruction that follows the instruction in which switching to stop mode was specified. The CPU then proceeds to interrupt processing.

Figure 6.5-4 shows the operation of return to normal mode from stop mode.

Figure 6.5-4 Release of stop mode (by external reset)



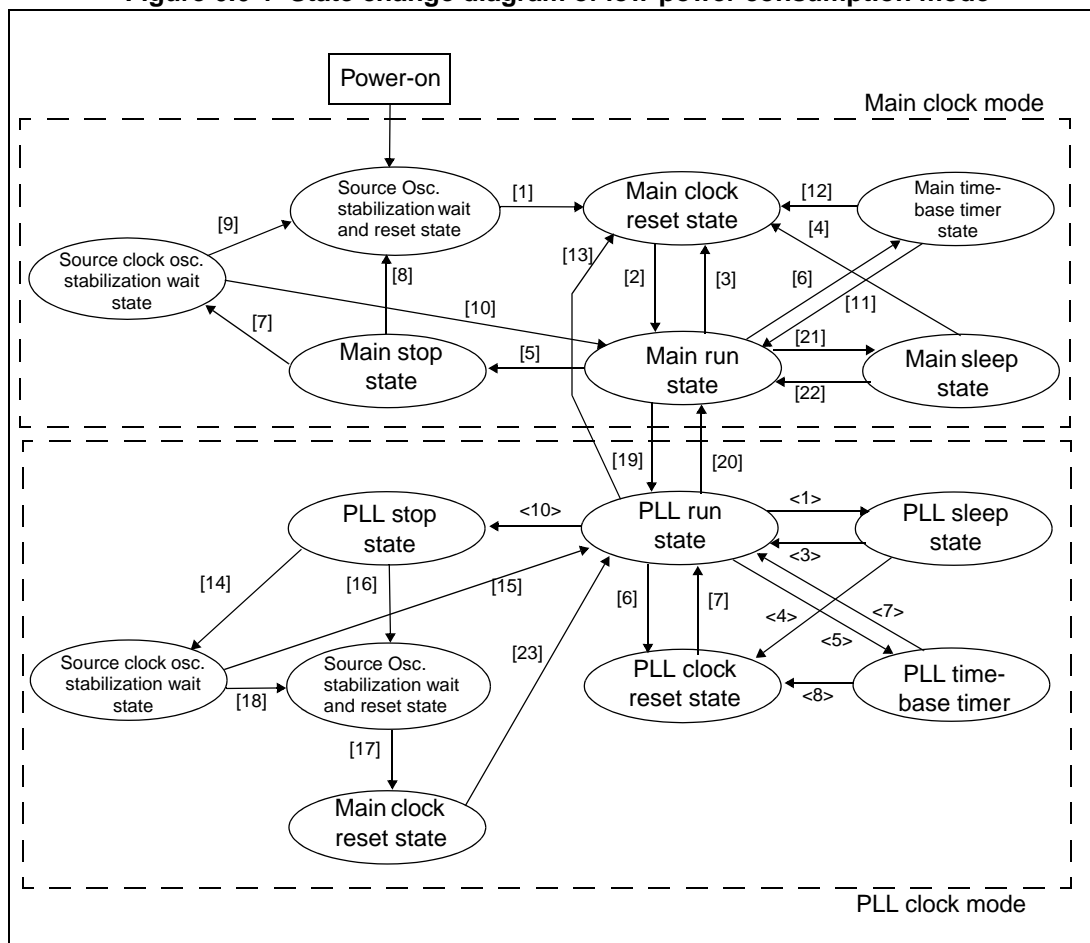
## MB90820B Series

## 6.6 State Change Diagram

Figure 6.6-1 shows the state change diagram of F<sup>2</sup>MC-16LX operation and gives change conditions.

### ■ State Change Diagram

Figure 6.6-1 State change diagram of low-power consumption mode



## ■ Low-Power Consumption Mode Operating States

Table 6.6-1 lists the operating states of low-power consumption mode.

**Table 6.6-1 Low-power consumption mode operating states**

Low-power consumption mode	Condition for transition	Oscillation	Clock	CPU	Peripheral	Pin	Release event
Main sleep	MCS = 1 SLP = 1	Active	Active	Inactive	Active	Active	Reset or interrupt
PLL sleep	MCS = 0 SLP = 1	Active	Active	Inactive	Active	Active	Reset or interrupt
Main/PLL time-base timer (SPL = 0)	MCS = x TMDX = 0	Active	Active	Inactive	Inactive	Hold	Reset or interrupt
Main/PLL time-base timer (SPL = 1)	MCS = x TMDX = 0	Active	Active	Inactive	Inactive	Hi-Z	Reset or interrupt
Main/PLL stop (SPL = 0)	MCS = x STP = 1	Inactive	Inactive	Inactive	Inactive	Hold	Reset or interrupt
Main/PLL stop (SPL = 1)	MCS = x STP = 1	Inactive	Inactive	Inactive	Inactive	Hi-Z	Reset or interrupt

## ● Clock mode switching and release (excluding standby mode)

Table 6.6-2 lists clock mode switching and release.

**Table 6.6-2 Clock mode switching and release**

Transition	Conditions
After power-on, transition to the main run state	[1] Source clock oscillation stabilization wait interval ends. (Time-base timer output) [2] Reset input has been cleared.
Reset during main run state	[3] External reset, software reset, or watchdog timer reset
Transition from main run state to PLL run state	[19] MCS = 0 (After PLL clock oscillation stabilization wait interval, switch to PLL clock) *
Return to main run state from PLL run state	[20] MCS = 1 (PLL clock deactivated)
Reset during PLL run state	[6] External reset or software reset ([7] After reset, return to PLL run state) [13] Watchdog reset ([2] After reset, return to main run state)

\*: The microcontroller operates using the main clock during the PLL clock oscillation stabilization wait state.

● Switching to and release of standby mode

Table 6.6-3 lists switching to and release of standby mode.

**Table 6.6-3 Switching to and release of standby mode**

Transition	Conditions
Transition to main sleep mode	[21] SLP = 1, MCS = 1 (Transition from main run state) [2] SLP = 1, MCS = 1 (Transition from PLL run state)
Release of main sleep mode	[22] Interrupt input [4] External reset
Transition to main stop mode	[5] STP = 1, MCS = 1 (Transition from main run state)
Transition to PLL stop mode	<10> STP = 1, MCS = 0 (Transition from PLL run state)
Release of main stop mode	[7] Interrupt input ([10] indicates return to main run state after oscillation stabilization wait) [8] External reset ([9] indicates external reset during oscillation stabilization wait state)
Release of PLL stop mode	[14] Interrupt input ([15] indicates return to PLL run state after oscillation stabilization wait) [16] External reset ([18] indicates external reset during oscillation stabilization wait state)
Transition to PLL sleep mode	<1> SLP = 1, MCS = 0 (Transition from PLL run state) <2> SLP = 1, MCS = 0 (Transition from main run state, switch to PLL clock after PLL clock oscillation stabilization wait) *
Release of PLL sleep mode	<3> Interrupt input <4> External reset
Transition to main time-base timer mode	[6] STP = 1, MCS = 1 (Transition from main run state)
Transition to PLL time-base timer mode	<5> STP = 1, MCS = 0 (Transition from PLL run state)
Release of main time-base timer mode	[11] Interrupt input [12] External reset ([2] After reset, return to main run state)
Release of PLL time-base timer mode	<7> Interrupt input <8> External reset ([7] After reset, return to PLL run state)

\* The microcontroller operates using the main clock during the PLL clock oscillation stabilization wait state.

## 6.7 State of Pins in Standby Mode and During Reset

The state of pins in standby mode and during reset is summarized below for each memory access mode.

### ■ Software Pull-up Resistor

For pins with a pull-up resistor selected by software, the pull-up resistor is disconnected during "L" level output.

### ■ State of Pins in Single-chip Mode

Table 6.7-1 lists the state of pins in single-chip mode.

**Table 6.7-1 State of pins in single-chip mode**

Pin name	Standby mode			When reset
	Sleep	Stop mode		
		SPL = 0	SPL = 1	
P00 to P07, P17, P20 to P27, P30 to P37, P40 to P47, P50, P60 to P63, P70 to P77, P80 to P87	The preceding state is retained*2	The preceding state is retained*2	Input shut off*3 / output Hi-Z	output Hi-Z
P10 to P16 P63		Input enabled*1		

\*1 "Input enabled" means that the input function is enabled when corresponding external interrupt pin is enable. Select either the pull-up or the pull-down option. Alternatively, an external input is required. Pins used as output ports are the same as other ports.

\*2 "The preceding state is retained" means that the state of the pin output existing immediately before switching to this mode is retained. Note that input is disabled if the preceding state was input.

- "State of the pin output is retained" means that the pin retains the value output from an operating internal peripheral unit or the value output from the port if the pin is used as a port.
- "Input disabled" means that the input to the pin is not accepted because the internal circuit is inactive, although operation of the input gate adjacent to the pin is enabled.

\*3 When in the input shut off state, an "L" level is passed to the internal circuit. "Output Hi-Z" means that the pin state is high-impedance because driving of the pin driving transistor is disabled.

**MB90820B Series****6.8 Usage Notes on Low-Power Consumption Mode**


---

**Note the following six items to use low-power consumption mode:**

- **Switching to standby mode and interrupts**
  - **Release of standby mode by an interrupt**
  - **Setting of standby mode**
  - **Release of stop mode**
  - **Release of time-base timer mode**
  - **Oscillation stabilization wait time**
  - **Points to note when accessing the low-power consumption mode control register (LPMCR) to switch to standby mode**
- 

**■ Notes on Standby Mode**

● **Switching to standby mode and interrupts**

During an interrupt request to the CPU from a peripheral function, the CPU ignores the STP and SLP bits of the low-power consumption mode control register (LPMCR) even though "1" has been written to these bits. Thus, switching to any standby mode is disabled (even after processing the interrupt is completed, there is no switch to standby mode). If the interrupt level is higher than 7, this action does not depend on whether the interrupt request is accepted by the CPU.

However, during execution of interrupt processing by the CPU, if the interrupt request flag bit is cleared and no other interrupt requests have been issued, switching to standby mode can be done.

● **Release of standby mode caused by an interrupt**

If an interrupt request higher than interrupt level 7 is issued from a peripheral function during the sleep, time-base timer, or stop modes, the standby mode is released. This action does not depend on whether the CPU accepts that interrupt.

After the release of standby mode, normal interrupt processing is performed. The CPU branches to the interrupt handling routine provided that the priority of the interrupt request indicated by the interrupt level setting bits (IL2, IL1, and IL0 of ICR) is higher than the interrupt level mask register (ILM); and the interrupt enable flag (I) of the condition code register (CCR) is set to "1" (enabled). If the interrupt is not accepted, the CPU starts the execution with the instruction that follows the instruction in which switching to standby mode was specified.

When interrupt processing is executed normally, the CPU first executes the instruction that follows the instruction in which switching to standby mode was specified. The CPU then proceeds to interrupt processing. Depending on the condition when switching to standby mode was performed, however, the CPU may proceed to interrupt processing before executing the next instruction.

If the CPU should not branch to the interrupt processing routine immediately after return to normal mode from standby mode, action must be taken to disable interrupts before standby mode is set.

- Setting of standby mode

When "1" is written to the STP bit and SLP bit of LPMCR at the same time, switching to standby mode is performed. If the MCS bit of the clock selection register (CKSCR) is "0", switching to time-base timer mode is performed; if this bit is "1", switching to stop mode is performed.

- Release of Stop Mode

To use an external interrupt for releasing stop mode, use an input that has been set as an interrupt input cause before the system enters stop mode. As an input cause, "H" level, "L" level, rising edge, or falling edge can be selected.

- Release of Time-base Timer Mode

When time-base timer mode is released, the microcontroller is placed in the PLL clock oscillation stabilization wait state. If the PLL clock is not used, change the MCS bit of the clock selection register (CKSCR) to "1" with the instruction that is to be executed immediately after a reset or on return from an interrupt.

If an external interrupt is used to release time-base timer mode, the input cause can be selected as "H" level, "L" level, rising edge, or falling edge.

- Oscillation Stabilization Wait Interval

- Source clock oscillation stabilization wait interval

Because the oscillator for source oscillation is halted in stop mode, an oscillation stabilization wait interval is required. A time period selected by the WS1 and WS0 bits of CKSCR is used as the oscillation stabilization wait interval.

- PLL clock oscillation stabilization wait interval

The CPU may be working with the main clock, and the PLL clock may be stopped. If the microcontroller will enter a mode in which the CPU and peripheral functions work with the PLL clock, the PLL clock initially enters the oscillation stabilization wait state. In this state, the CPU still operates using the main clock.

The PLL clock oscillation stabilization wait interval is fixed at  $2^{14}/\text{HCLK}$  (HCLK: oscillation clock frequency).

However, this interval may range from  $2^{14}/\text{HCLK}$  to  $2 \times 2^{14}/\text{HCLK}$  depending on the status of the time-base timer, if the time-base timer is not cleared before the PLL clock oscillation stabilization wait state is entered. (For example, return to the PLL run state from time-base timer mode occurs because of an external reset.)

## ■ Points to Note when Accessing the Low-power Consumption Mode Control Register (LPMCR) to Switch to Standby Mode

- When using assembly language to access the low-power consumption mode control register (LPMCR)

When using the low-power consumption mode control register (LPMCR) to switch to a low-power consumption mode, use one of the instructions listed in Table 6.3-2.

Always insert the instructions enclosed in the box below immediately after the instruction from Table 6.3-2 for switching to a low-power consumption mode.

MOV LPMCR,#H'xx ; Instruction from Table 6.3-2 for switching to a low-power consumption mode	
NOP NOP JMP \$+3 ; Jump to next instruction MOV A,#H'10 ; Next instruction	

The operation when recovering from low-power consumption mode is not guaranteed if instructions other than those shown in the box are used.

- When accessing the low-power consumption control register (LPMCR) from a C program

Use one of the methods listed in (1) to (3) below to set the low-power consumption mode control register (LPMCR) to switch to standby mode.

- (1) Implement the instructions for switching to the standby mode in a function and insert two calls to the `_wait_nop()` function immediately after the instruction that switches to standby mode. If it is possible that an interrupt for recovering from standby mode may occur while the function is executing, use optimization when compiling and prevent generation of the LINK and ULINK instructions.

Example: (function for switching to watch mode or time-base timer mode)

```
void enter_watch(){
    IO_LPMCR.byte = 0x10;          /* Set the TMDX bit in the LPMCR register to 0 */
    _wait_nop();
    _wait_nop();
}
```

- (2) Use `_asm` statements for the instructions for switching to standby mode and insert two NOP instructions and a JMP instruction after the instruction for switching to standby mode.

Example: (switching to sleep mode)

```
_asm(" MOV I:_IO_LPMCR, #H'58"); /* Set the SLP bit in the LPMCR register to 1 */
_asm(" NOP");
_asm(" NOP");
_asm(" JMP $+3");                /* Jump to next instruction */
```



- (3) Enclose the instructions for switching to standby mode between "#pragma asm" and "#pragma endasm" and insert two NOP instructions and a JMP instruction after the instruction for switching to standby mode.

Example: (switching to stop mode)

```
#pragma asm
MOV I:_IO_LPMCR, #H'98      /* Set the STP bit in the LPMCR register to 1 */
NOP
NOP
JMP $+3                     /* Jump to next instruction */
#pragma endasm
```

# **CHAPTER 7**

---

# **INTERRUPT**

**This chapter explains the function and operation of the interrupt and extended intelligent I/O service (EI<sup>2</sup>OS) in the MB90820B series.**

- 7.1 Interrupt
- 7.2 Interrupt Causes and Interrupt Vectors
- 7.3 Interrupt Control Registers and Peripheral Functions
- 7.4 Hardware Interrupt
- 7.5 Software Interrupt
- 7.6 Interrupt of Extended Intelligent I/O Service (EI<sup>2</sup>OS)
- 7.7 Exception Processing Interrupt
- 7.8 Stack Operations for Interrupt Processing

## **7.1 Interrupt**

---

**This chapter explains the function and operation of the interrupt and extended intelligent I/O service (EI<sup>2</sup>OS) in the MB90820B series.**

- **Hardware interrupt**
  - **Software interrupt**
  - **Interrupt from extended intelligent I/O service (EI<sup>2</sup>OS)**
  - **Exception processing**
- 

### **■ Interrupt Types and Functions**

#### ● **Hardware interrupt**

A hardware interrupt transfers control to a user-defined interrupt processing program in response to an interrupt request from a peripheral function.

#### ● **Software interrupt**

A software interrupt transfers control to a user-defined interrupt processing program triggered by the execution of a dedicated software interrupt instruction (such as the INT instruction).

#### ● **Interrupt from extended intelligent I/O service (EI<sup>2</sup>OS)**

The EI<sup>2</sup>OS function automatically transfers data between a peripheral function and memory. Data transfer, which has ordinarily been executed by an interrupt processing program, can be handled like a direct memory access (DMA). When the specified number of data transfers has been terminated, the interrupt processing program is automatically executed.

An instruction from EI<sup>2</sup>OS is a type of hardware interrupt.

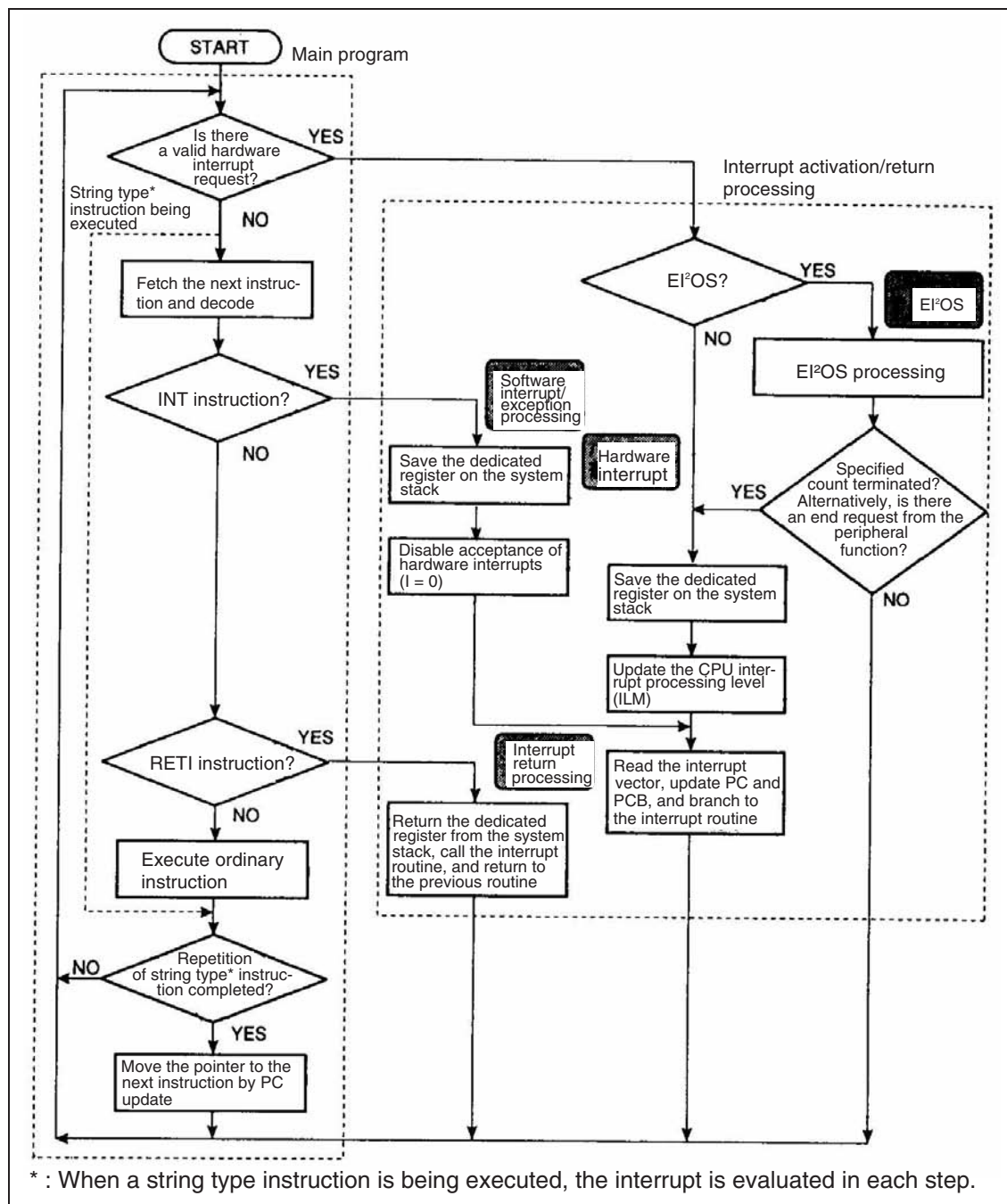
#### ● **Exception processing**

Exception processing is basically the same as an interrupt. When an exception event (execution of an undefined instruction) is detected on the instruction boundary, ordinary processing is interrupted and exception processing is performed. This is equivalent to software interrupt instruction INT10.

## ■ Interrupt Operation

Figure 6.1-1 shows the activation and return processing for the four types of interrupt functions.

Figure 7.1-1 Overall flow of interrupt operation



## 7.2 Interrupt Causes and Interrupt Vectors

The F<sup>2</sup>MC-16LX has functions for handling 256 types of interrupt causes. The 256 interrupt vector tables are allocated to the memory at the highest addresses. These interrupt vectors are shared by all interrupts.

Software interrupt can use all these interrupt vectors (INT0 to INT255). Software interrupt shares same interrupt vectors with the hardware interrupt and exception processing interrupt. Hardware interrupt uses a fixed interrupt vector and interrupt control register (ICR) for each peripheral function.

### ■ Interrupt Vectors

#### ● Interrupt vectors

Interrupt vector tables referred during interrupt processing are allocated to the highest addresses in the memory area (FFFC00<sub>H</sub> to FFFFFFF<sub>H</sub>). Interrupt vectors share the same area with EI<sup>2</sup>OS, exception processing, hardware, and software interrupt.

Table 6.2-1 shows the assignment of interrupt numbers and interrupt vectors.

**Table 7.2-1 Interrupt vectors**

Software interrupt instruction	Vector address L	Vector address M	Vector address H	Mode data	Interrupt no.	Hardware interrupt
INT0	FFFFFC <sub>H</sub>	FFFFFD <sub>H</sub>	FFFFFE <sub>H</sub>	Not used	#0	None
:	:	:	:	:	:	:
INT7	FFFFE0 <sub>H</sub>	FFFFE1 <sub>H</sub>	FFFFE2 <sub>H</sub>	Not used	#7	None
INT8	FFFFDC <sub>H</sub>	FFFFDD <sub>H</sub>	FFFFDE <sub>H</sub>	FFFFDF <sub>H</sub>	#8	(RESET vector)
INT9	FFFFD8 <sub>H</sub>	FFFFD9 <sub>H</sub>	FFFFDA <sub>H</sub>	Not used	#9	None
INT10	FFFFD4 <sub>H</sub>	FFFFD5 <sub>H</sub>	FFFFD6 <sub>H</sub>	Not used	#10	<Exception processing>
INT11	FFFFD0 <sub>H</sub>	FFFFD1 <sub>H</sub>	FFFFD2 <sub>H</sub>	Not used	#11	Hardware interrupt #0
INT12	FFFFCC <sub>H</sub>	FFFFCD <sub>H</sub>	FFFFCE <sub>H</sub>	Not used	#12	Hardware interrupt #1
INT13	FFFFC8 <sub>H</sub>	FFFFC9 <sub>H</sub>	FFFFCA <sub>H</sub>	Not used	#13	Hardware interrupt #2
INT14	FFFFC4 <sub>H</sub>	FFFFC5 <sub>H</sub>	FFFFC6 <sub>H</sub>	Not used	#14	Hardware interrupt #3
:	:	:	:	:	:	:
INT254	FFFC04 <sub>H</sub>	FFFC05 <sub>H</sub>	FFFC06 <sub>H</sub>	Not used	#254	None
INT255	FFFC00 <sub>H</sub>	FFFC01 <sub>H</sub>	FFFC02 <sub>H</sub>	Not used	#255	None

Reference:

Unused interrupt vectors should be set as the exception processing address.

## ■ Interrupt Causes and Interrupt Vectors/Interrupt Control Registers

Table 6.2-2 shows the relationship among interrupt causes (excluding software interrupt), interrupt vectors, and interrupt control registers.

**Table 7.2-2 Interrupt causes, interrupt vectors, and interrupt control registers**

Interrupt cause	EI <sup>2</sup> OS support	Interrupt vector		Interrupt control register		Priority *2	
		Number	Address	ICR	Address		
Reset	X	#08	08 <sub>H</sub>	FFFFDC <sub>H</sub>	-	-	<div>↑</div> <div>↓</div> <div>High</div> <div>Low</div>
INT9 instruction	X	#09	09 <sub>H</sub>	FFFFD8 <sub>H</sub>	-	-	
Exception processing	X	#10	0A <sub>H</sub>	FFFFD4 <sub>H</sub>	-	-	
A/D converter conversion termination	O	#11	0B <sub>H</sub>	FFFFD0 <sub>H</sub>	ICR00	0000B0 <sub>H</sub> *1	
Output compare channel 0 match	O	#12	0C <sub>H</sub>	FFFFCC <sub>H</sub>			
End of measurement by PWC timer 0 / PWC timer 0 overflow	O	#13	0D <sub>H</sub>	FFFFC8 <sub>H</sub>	ICR01	0000B1 <sub>H</sub> *1	
16-bit PPG timer 0	O	#14	0E <sub>H</sub>	FFFFC4 <sub>H</sub>			
Output compare channel 1 match	O	#15	0F <sub>H</sub>	FFFFC0 <sub>H</sub>	ICR02	0000B2 <sub>H</sub> *1	
16-bit PPG timer 1	O	#16	10 <sub>H</sub>	FFFFBC <sub>H</sub>			
Output compare channel 2 match	O	#17	11 <sub>H</sub>	FFFFB8 <sub>H</sub>	ICR03	0000B3 <sub>H</sub> *1	
16-bit reload timer 1 underflow	O	#18	12 <sub>H</sub>	FFFFB4 <sub>H</sub>			
Output compare channel 3 match	O	#19	13 <sub>H</sub>	FFFFB0 <sub>H</sub>	ICR04	0000B4 <sub>H</sub> *1	
DTP/ext. interrupt channels 0/1 detection	O	#20	14 <sub>H</sub>	FFFFAC <sub>H</sub>			
DTTI	Δ						
Output compare channel 4 match	O	#21	15 <sub>H</sub>	FFFA8 <sub>H</sub>	ICR05	0000B5 <sub>H</sub> *1	
DTP/ext. interrupt channels 2/3 detection	O	#22	16 <sub>H</sub>	FFFA4 <sub>H</sub>			
Output compare channel 5 match	O	#23	17 <sub>H</sub>	FFFA0 <sub>H</sub>	ICR06	0000B6 <sub>H</sub> *1	
End of measurement by PWC timer 1 / PWC timer 1 overflow	O	#24	18 <sub>H</sub>	FFFF9C <sub>H</sub>			
DTP/ext. interrupt channel 4 detection	O	#25	19 <sub>H</sub>	FFFF98 <sub>H</sub>	ICR07	0000B7 <sub>H</sub> *1	
DTP/ext. interrupt channel 5 detection	O	#26	1A <sub>H</sub>	FFFF94 <sub>H</sub>			
DTP/ext. interrupt channel 6 detection	O	#27	1B <sub>H</sub>	FFFF90 <sub>H</sub>	ICR08	0000B8 <sub>H</sub> *1	
DTP/ext. interrupt channel 7 detection	O	#28	1C <sub>H</sub>	FFFF8C <sub>H</sub>			
Waveform generator 16-bit timer 0/1/2 underflow	Δ	#29	1D <sub>H</sub>	FFFF88 <sub>H</sub>	ICR09	0000B9 <sub>H</sub> *1	
16-bit reload timer 0 underflow	O	#30	1E <sub>H</sub>	FFFF84 <sub>H</sub>			
16-bit free-run timer 0 detect	Δ	#31	1F <sub>H</sub>	FFFF80 <sub>H</sub>	ICR10	0000BA <sub>H</sub> *1	
16-bit PPG timer 2	O	#32	20 <sub>H</sub>	FFFF7C <sub>H</sub>			
Input capture channels 0/1	O	#33	21 <sub>H</sub>	FFFF78 <sub>H</sub>	ICR11	0000BB <sub>H</sub> *1	
16-bit free-run timer compare clear	Δ	#34	22 <sub>H</sub>	FFFF74 <sub>H</sub>			
Input capture channels 2/3	O	#35	23 <sub>H</sub>	FFFF70 <sub>H</sub>	ICR12	0000BC <sub>H</sub> *1	
Time-base timer	Δ	#36	24 <sub>H</sub>	FFFF6C <sub>H</sub>			
UART1 receive completed	⊙	#37	25 <sub>H</sub>	FFFF68 <sub>H</sub>	ICR13	0000BD <sub>H</sub> *1	
UART1 send start	Δ	#38	26 <sub>H</sub>	FFFF64 <sub>H</sub>			
UART0 receive completed	⊙	#39	27 <sub>H</sub>	FFFF60 <sub>H</sub>	ICR14	0000BE <sub>H</sub> *1	
UART0 send start	Δ	#40	28 <sub>H</sub>	FFFF5C <sub>H</sub>			
Flash memory status	Δ	#41	29 <sub>H</sub>	FFFF58 <sub>H</sub>	ICR15	0000BF <sub>H</sub> *1	
Delayed interrupt generator module	Δ	#42	2A <sub>H</sub>	FFFF54 <sub>H</sub>			

O: Can be used and interrupt request flag is cleared by EI<sup>2</sup>OS interrupt clear signal.

X: Cannot be used.

⊙: Can be used and support the EI<sup>2</sup>OS stop request.

Δ: Usable when an interrupt cause that shares the ICR is not used.

- \*1: - For peripheral functions that share the ICR register, the interrupt level will be the same.
  - If the extended intelligent I/O service is to be used with a peripheral function that shares the ICR register with another peripheral function, the service can be started by either of the function. And if EI<sup>2</sup>OS clear is supported, both interrupt request flags for the two interrupt causes are cleared by EI<sup>2</sup>OS interrupt clear signal. It is recommended to mask either of the interrupt requests during the use of EI<sup>2</sup>OS.
  - EI<sup>2</sup>OS service cannot be started multiple times simultaneously. Interrupt other than the operating interrupt is masked during EI<sup>2</sup>OS operation. It is recommended to mask either of the interrupt requests during the use of EI<sup>2</sup>OS.
- \*2: This priority is applied when interrupts of the same level occur simultaneously.

**MB90820B Series****7.3 Interrupt Control Registers and Peripheral Functions**

Interrupt control registers (ICR00 to ICR15) are located inside the interrupt controller. The interrupt control registers correspond to all peripheral functions that have the interrupt function. These registers control interrupts and the extended intelligent I/O service (EI<sup>2</sup>OS).

**■ Interrupt Control Registers**

Table 6.3-1 lists the interrupt control registers and corresponding peripheral functions.

**Table 7.3-1 Interrupt control registers**

Address	Register	Abbreviation	Corresponding peripheral function
0000B0 <sub>H</sub>	Interrupt control register 00	ICR00	A/D converter conversion termination, Output compare channel 0 match
0000B1 <sub>H</sub>	Interrupt control register 01	ICR01	End of measurement by PWC timer 0 /PWC timer 0 overflow
0000B2 <sub>H</sub>	Interrupt control register 02	ICR02	Output compare channel 1 match, 16-bit PPG timer 1
0000B3 <sub>H</sub>	Interrupt control register 03	ICR03	Output compare channel 2 match, 16-bit reload timer 1 underflow
0000B4 <sub>H</sub>	Interrupt control register 04	ICR04	Output compare channel 3 match, DTP/ext. interrupt channels 0/1 detection, DTTI
0000B5 <sub>H</sub>	Interrupt control register 05	ICR05	Output compare channel 4 match, DTP/ext. interrupt channels 2/3 detection
0000B6 <sub>H</sub>	Interrupt control register 06	ICR06	Output compare channel 5 match, End of measurement by PWC timer 1 /PWC timer 1 overflow
0000B7 <sub>H</sub>	Interrupt control register 07	ICR07	DTP/ext. interrupt channels 4 detection, DTP/ext. interrupt channels 5 detection
0000B8 <sub>H</sub>	Interrupt control register 08	ICR08	DTP/ext. interrupt channels 6 detection, DTP/ext. interrupt channels 7 detection
0000B9 <sub>H</sub>	Interrupt control register 09	ICR09	Waveform generator 16-bit reload timer 0/1/2 underflow, 16-bit reload timer 0 underflow
0000BA <sub>H</sub>	Interrupt control register 10	ICR10	16-bit free-run timer zero detect, 16-bit PPG timer 2
0000BB <sub>H</sub>	Interrupt control register 11	ICR11	Input capture channels 0/1, 16-bit free-run timer compare clear
0000BC <sub>H</sub>	Interrupt control register 12	ICR12	Input capture channels 2/3, Time-base timer
0000BD <sub>H</sub>	Interrupt control register 13	ICR13	UART1 receive, UART1 send
0000BE <sub>H</sub>	Interrupt control register 14	ICR14	UART0 receive, UART0 send
0000BF <sub>H</sub>	Interrupt control register 15	ICR15	Flash memory status, Delayed interrupt generator module



## ■ Interrupt Control Register Functions

All interrupt control registers (ICR) do the following:

- Set the interrupt level of the corresponding peripheral function
- Select ordinary interrupt or the extended intelligent I/O service as interrupt of the corresponding peripheral function
- Select an extended intelligent I/O service (EI<sup>2</sup>OS) channel
- Display the status of the extended intelligent I/O service (EI<sup>2</sup>OS)

Some of the functions for the interrupt control registers (ICR) differ during writing and reading, as shown in Figure 7.3-1 and Figure 7.3-2 .

---

### Note:

Do not use a read-modify-write instruction to access the interrupt control registers (ICR), since operation will not be correct.

---

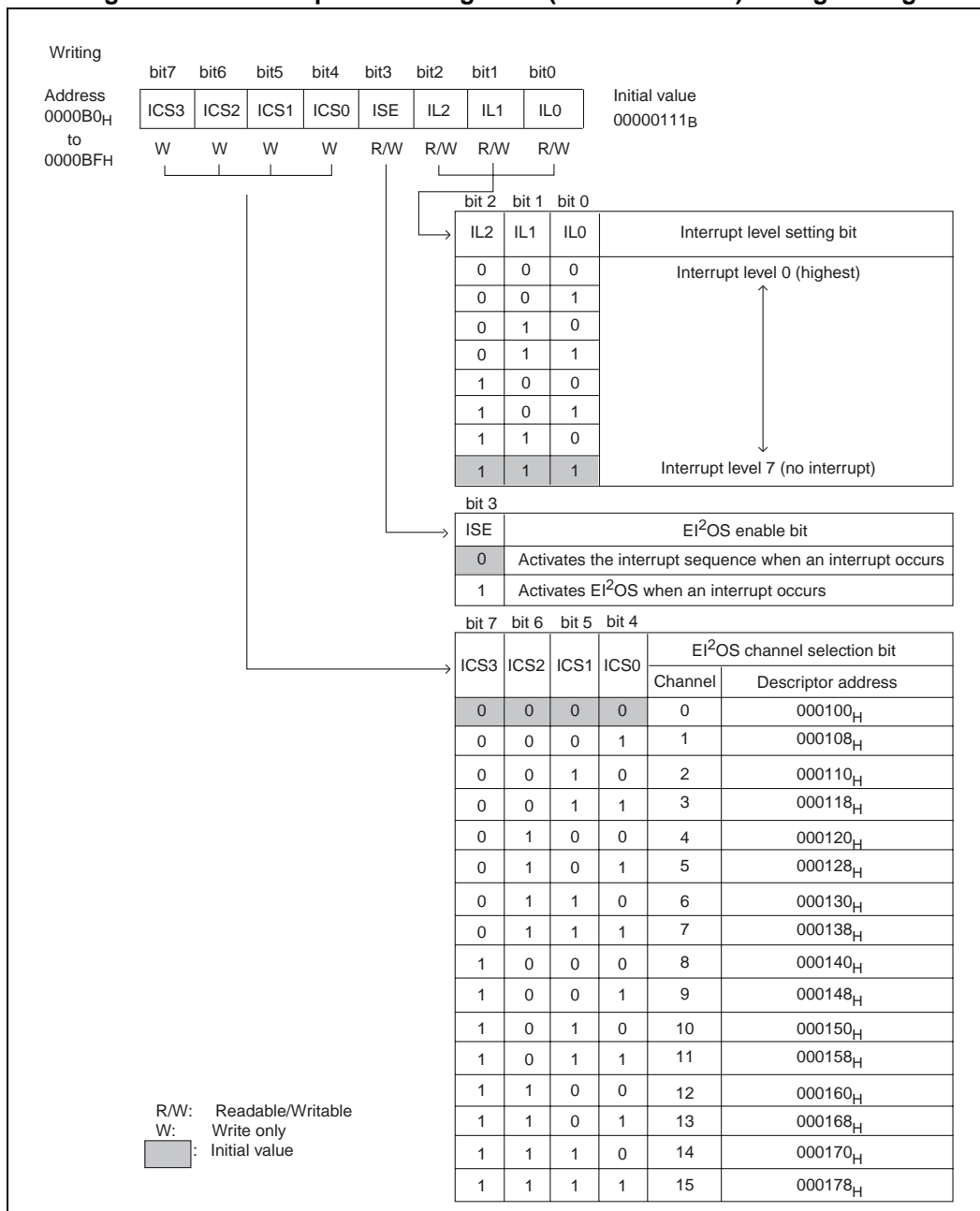
## MB90820B Series

## 7.3.1 Interrupt Control Registers (ICR00 to ICR15)

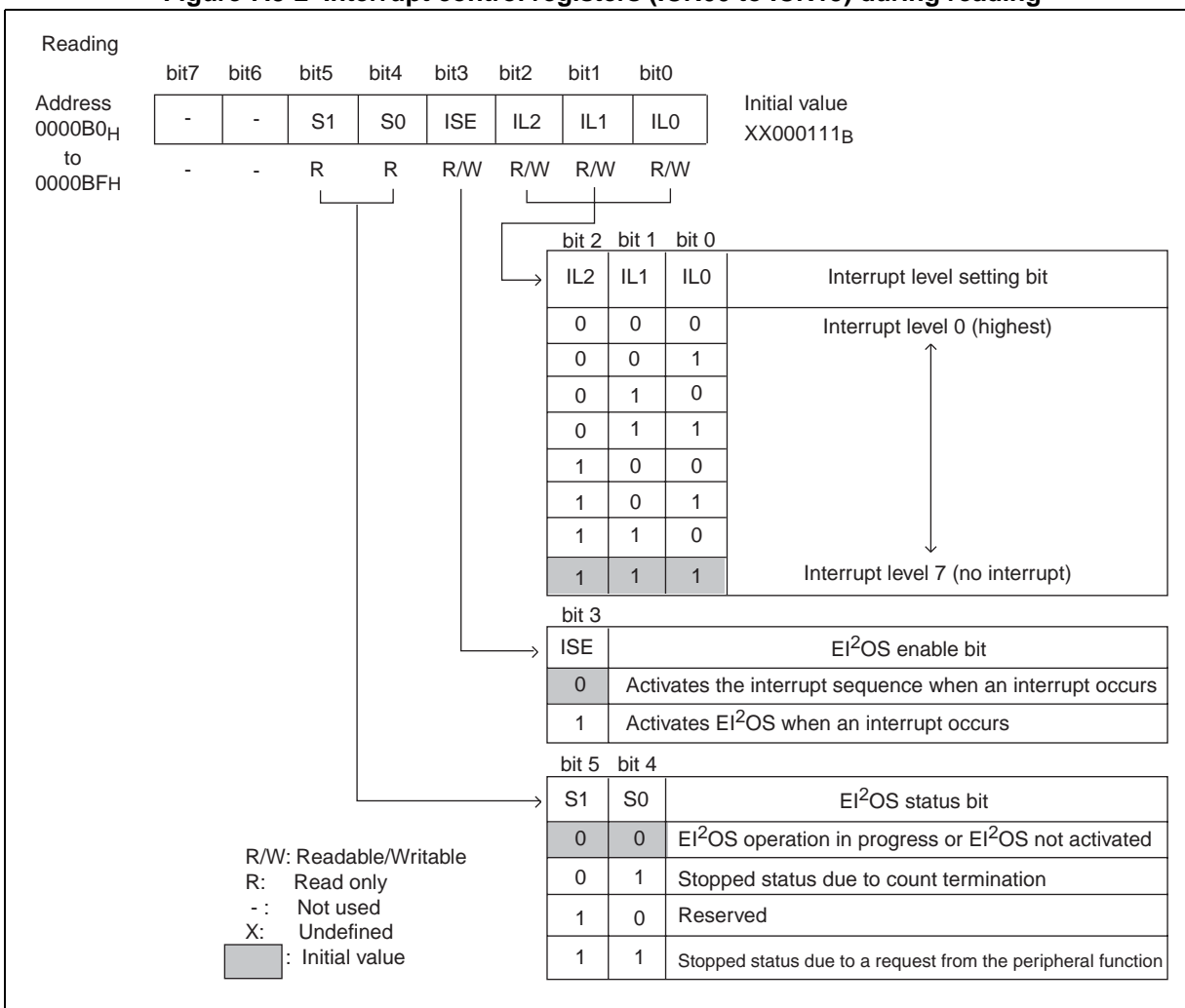
Interrupt control registers correspond to all peripheral functions that have the interrupt function. The interrupt control registers control the processing when an interrupt request occurs. The functions of these registers partially differ at writing and reading.

### ■ Interrupt Control Registers (ICR00 to ICR15)

Figure 7.3-1 Interrupt control registers (ICR00 to ICR15) during writing



**Figure 7.3-2 Interrupt control registers (ICR00 to ICR15) during reading**



## MB90820B Series

### 7.3.2 Interrupt Control Register Functions

The interrupt control registers (ICR00 to ICR15) consist of the following four functional bits:

- Interrupt level setting bits (IL2 to IL0)
- Extended intelligent I/O service (EI<sup>2</sup>OS) enable bit (ISE)
- Extended intelligent I/O service (EI<sup>2</sup>OS) channel selection bits (ICS3 to ICS0)
- Extended intelligent I/O service (EI<sup>2</sup>OS) status bits (S1, S0)

#### ■ Bit Configuration of Interrupt Control Registers (ICR)

Figure 7.3-3 shows the configuration of the interrupt control register (ICR) bits.

**Figure 7.3-3 Configuration of interrupt control registers (ICR)**

Writing to interrupt control register (ICR)								
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0000B0 <sub>H</sub>	ICS3	ICS2	ICS1	ICS0	ISE	IL2	IL1	IL0
to	W	W	W	W	W	W	W	W
0000BF <sub>H</sub>								
Reading of interrupt control register (ICR)								
Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0000B0 <sub>H</sub>	—	—	S1	S0	ISE	IL2	IL1	IL0
to	—	—	R	R	R	R	R	R
0000BF <sub>H</sub>								
Initial value								
00000111 <sub>B</sub>								
XX000111 <sub>B</sub>								
R: Read only								
W: Write only								
- : Undefined								

Reference:

- The ICS3 to ICS0 bits are valid only when the extended intelligent I/O service (EI<sup>2</sup>OS) has been activated. To activate EI<sup>2</sup>OS, set the ISE bit to "1". To not activate EI<sup>2</sup>OS, set the ISE bit to 0. When EI<sup>2</sup>OS is not activated, setting ICS3 to ICS0 is optional.
- ICS1 and ICS0 are valid only for writing. S1 and S0 are valid only for reading.

## ■ Interrupt Control Register Functions

### ● Interrupt level setting bits (IL2 to IL0)

These bits set the interrupt level of the corresponding peripheral function. These bits are initialized to level 7 (no interrupt) by a reset.

Table 7.3-2 shows the correspondence between the interrupt level setting bits and interrupt levels.

**Table 7.3-2 Correspondence between the interrupt level setting bits and interrupt levels**

IL2	IL1	IL0	Interrupt level
0	0	0	<div>0 (highest priority)</div> <div>↑</div> <div>↓</div> <div>6 (lowest priority)</div>
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	7 (no interrupt)
1	1	1	

### ● Extended intelligent I/O service (EI<sup>2</sup>OS) enable bit (ISE)

If this bit is "1" when an interrupt request is generated, EI<sup>2</sup>OS is activated. If this bit is "0" when an interrupt request is generated, the interrupt sequence is activated. When the EI<sup>2</sup>OS termination condition is met (when the S1 and S0 bits are not "00<sub>B</sub>"), the ISE bit is cleared. If the corresponding peripheral function does not have the EI<sup>2</sup>OS function, the ISE bit must be set to "0" by software. The ISE bit is initialized to "0" by a reset.

● Extended intelligent I/O service (EI<sup>2</sup>OS) channel selection bits (ICS3 to ICS0)

These write-only bits specify the EI<sup>2</sup>OS channel. The EI<sup>2</sup>OS descriptor address is determined based on the value set here. The ICS bit is initialized to "0000<sub>B</sub>" by a reset.

Table 7.3-3 shows the correspondence between the EI<sup>2</sup>OS channel selection bits and descriptor addresses.

**Table 7.3-3 Correspondence between the EI<sup>2</sup>OS channel selection bits and descriptor addresses**

ICS3	ICS2	ICS1	ICS0	Selected channel	Descriptor address
0	0	0	0	0	000100 <sub>H</sub>
0	0	0	1	1	000108 <sub>H</sub>
0	0	1	0	2	000110 <sub>H</sub>
0	0	1	1	3	000118 <sub>H</sub>
0	1	0	0	4	000120 <sub>H</sub>
0	1	0	1	5	000128 <sub>H</sub>
0	1	1	0	6	000130 <sub>H</sub>
0	1	1	1	7	000138 <sub>H</sub>
1	0	0	0	8	000140 <sub>H</sub>
1	0	0	1	9	000148 <sub>H</sub>
1	0	1	0	10	000150 <sub>H</sub>
1	0	1	1	11	000158 <sub>H</sub>
1	1	0	0	12	000160 <sub>H</sub>
1	1	0	1	13	000168 <sub>H</sub>
1	1	1	0	14	000170 <sub>H</sub>
1	1	1	1	15	000178 <sub>H</sub>

● Extended intelligent I/O service (EI<sup>2</sup>OS) status bits (S1 and S0)

These are read-only bits. When this value is checked at EI<sup>2</sup>OS termination, the operating status and termination status can be distinguished. These bits are initialized to "00<sub>B</sub>" by a reset.

Table 7.3-4 shows the relationship between the S0 and S1 bits and the EI<sup>2</sup>OS status.

**Table 7.3-4 Relationship between EI<sup>2</sup>OS status bits and the EI<sup>2</sup>OS status**

S1	S0	EI <sup>2</sup> OS status
0	0	EI <sup>2</sup> OS operation in progress or EI <sup>2</sup> OS not activated
0	1	Stopped status due to count termination
1	0	Reserved
1	1	Stopped status due to a request from the peripheral function

## 7.4 Hardware Interrupt

---

The hardware interrupt function temporarily interrupts the program being executed by the CPU and transfers control to a user-defined interrupt processing program in response to an interrupt signal from a peripheral function.

The extended intelligent I/O service (EI<sup>2</sup>OS) and external interrupt are executed as a type of hardware interrupt.

---

### ■ Hardware Interrupt

#### ● Hardware interrupt function

The hardware interrupt function compares the interrupt level of the interrupt request signal output by a peripheral function with the interrupt level mask register (ILM) in the CPU processor status (PS). The function then refers the contents of the I flag in the processor status (PS) through the hardware and decides if the interrupt can be accepted.

When the hardware interrupt is accepted, the CPU internal registers are automatically saved on the system stack. The currently requested interrupt level is stored in the interrupt level mask register (ILM), and the function branches to the corresponding interrupt vector.

#### ● Multiple interrupts

Multiple hardware interrupts can be activated.

#### ● Extended intelligent I/O service (EI<sup>2</sup>OS)

EI<sup>2</sup>OS is an automatic transfer function between memory and I/O. When the specified transfer count has been completed, a hardware interrupt is activated. Multiple EI<sup>2</sup>OS activation does not occur. During EI<sup>2</sup>OS processing, all other interrupt requests and EI<sup>2</sup>OS requests are held.

#### ● External interrupt

An external interrupt (including wake-up interrupt) is accepted from a peripheral function (interrupt request detection circuit) as a hardware interrupt.

#### ● Interrupt vector

Interrupt vector tables referred during interrupt processing are allocated to memory at FFFC00<sub>H</sub> to FFFFFFF<sub>H</sub>. These tables are shared by software interrupts.

See Section "7.2 Interrupt Causes and Interrupt Vectors", for more information about the allocation of interrupt numbers and interrupt vectors.

## ■ Hardware Interrupt Structure

Table 6.4-1 lists four mechanisms used for hardware interrupt. These four mechanisms must be included in the program before hardware interrupt can be used.

**Table 7.4-1 Mechanisms used for hardware interrupt**

	Hardware interrupt mechanism	Function
Peripheral function	Interrupt enable bit, interrupt request bit	Controls interrupt requests from a peripheral function
Interrupt controller	Interrupt control register (ICR)	Sets the interrupt level and controls EI <sup>2</sup> OS
CPU	Interrupt enable flag (I)	Identifies the interrupt enable status
	Interrupt level mask register (ILM)	Compares the request interrupt level and current interrupt level
	Microcode	Executes the interrupt processing routine
FFFC00 <sub>H</sub> to FFFFFFF <sub>H</sub> in memory	Interrupt vector table	Stores the branch destination address for interrupt processing

## ■ Hardware Interrupt Suppression

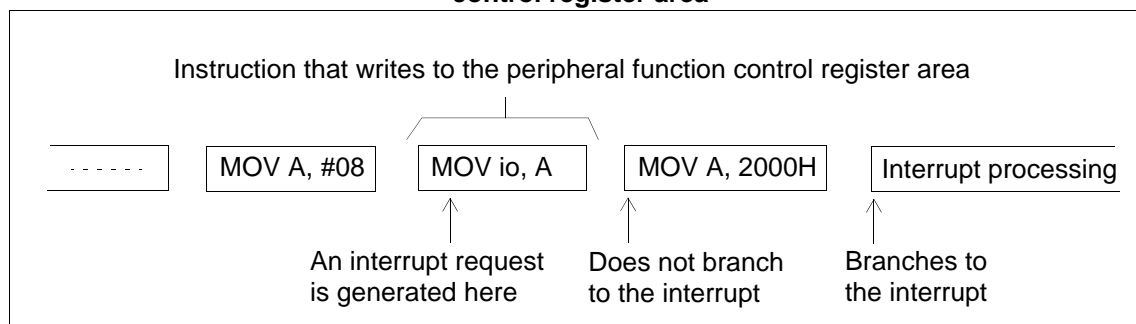
Acceptance of hardware interrupt requests is suppressed under the following conditions.

### ● Hardware interrupt suppression during writing to the peripheral function control register area

When data is being written to the peripheral function control register area, hardware interrupt requests are not accepted. This prevents the CPU from making operational mistakes. The mistakes may be caused if an interrupt request is generated during data is written to the interrupt control registers for a resource. The peripheral function control register area is not the I/O addressing area at 000000<sub>H</sub> to 0000FF<sub>H</sub>, but the area allocated to the peripheral function control register and data register.

Figure 6.4-1 shows hardware interrupt operation during writing to built-in resource area.

**Figure 7.4-1 Hardware interrupt request while writing to the peripheral function control register area**



### ● Hardware interrupt suppression by interrupt suppression instruction

The ten types of hardware interrupt suppression instructions listed in Table 6.4-2 ignore interrupt requests without detecting whether a hardware interrupt request exists.



**Table 7.4-2 Hardware interrupt suppression instruction**

	Prefix code	Interrupt/hold suppression instructions (instructions that delay the effect of the prefix code)
Instructions that do not accept interrupt and hold requests	PCB DTB ADB SPB CMR NCC	MOV ILM, #imm8 OR CCR, #imm8 AND CCR, #imm8 POPW PS

Even if a valid hardware interrupt request is generated during execution for one of these instructions, the interrupt is not processed until the first time an instruction of a different type is executed.

● **Hardware interrupt suppression during execution of software interrupt**

When a software interrupt is activated, the I flag is cleared to "0". In this state, other interrupt requests cannot be accepted.

## 7.4.1 Operation of Hardware Interrupt

---

**This section explains hardware interrupt operation from generation of a hardware interrupt request to the completion of interrupt processing.**

---

### ■ Hardware Interrupt Activation

- Peripheral function operation (generation of an interrupt request)

A peripheral function that has a hardware interrupt request function also has an interrupt request flag that indicates the presence of interrupt requests and an interrupt enable flag that determines whether CPU interrupt requests are enabled or disabled. The interrupt request flag is set when an event specific to the peripheral function occurs.

- Interrupt controller operation (interrupt request control)

The interrupt controller compares the interrupt levels (IL) of interrupt requests received at the same time. The interrupt controller selects the request with the highest level (with the smallest IL value) and posts it to the CPU. When multiple requests have the same interrupt level, the request with the smallest interrupt number has the highest priority.

- CPU operation (interrupt request acceptance and interrupt processing)

The CPU compares the received interrupt level (ICR: IL2 to IL0) and the interrupt level mask register (ILM). If  $IL < ILM$  and interrupts are enabled (PS: CCR: I = 1), the CPU activates the interrupt processing microcode after the instruction currently being executed terminates.

At the beginning of the interrupt processing microcode, the CPU refers the ISE bit. If ISE = 0, the CPU continues the execution of interrupt processing. (If ISE = 1, EI<sup>2</sup>OS is activated.)

Interrupt processing saves the contents of the dedicated registers (12 bytes including A, DPR, ADB, DTB, PCB, PC, and PS) on the system stack (the system stack space indicated by the SSB and SSP).

The CPU then loads data into the interrupt vector program counters (PCB and PC), updates the ILM, and sets the stack flag (S) (sets CCR: S = 1 and activates the system stack).

### ■ Returning from a Hardware Interrupt

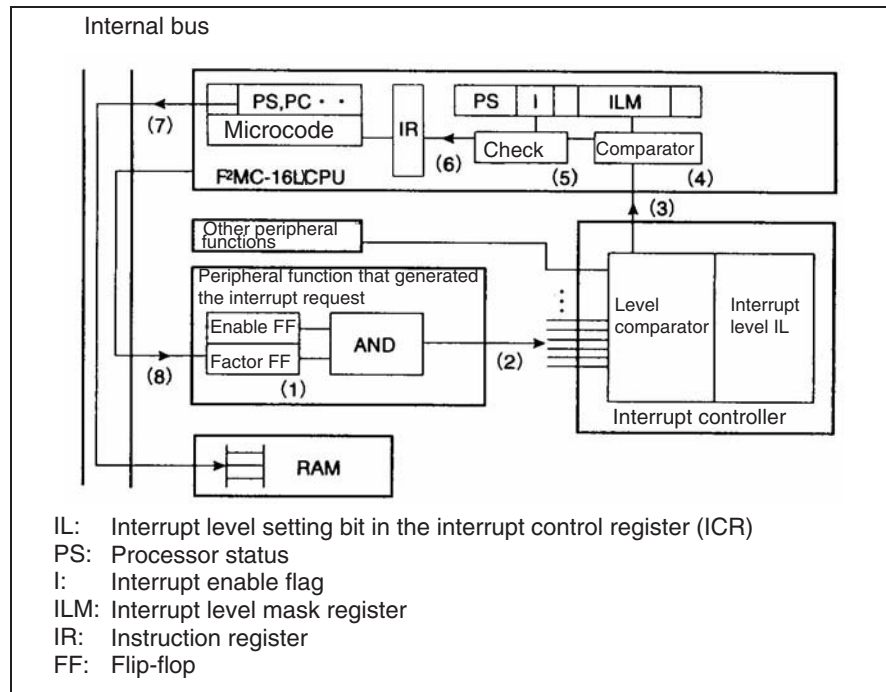
In an interrupt processing program, when the interrupt request flag of the peripheral function that generates the interrupt cause is cleared and the RETI instruction is executed, 12-byte data saved on the system stack is restored to the dedicated registers and the processing that was being executed before branching for the interrupt is resumed.

When the interrupt request flag is cleared, interrupt requests output by the peripheral function to the interrupt controller are automatically canceled.

## ■ Hardware Interrupt Operation

Figure 7.4-2 shows hardware interrupt operation from generation of a hardware interrupt to the completion of interrupt processing.

**Figure 7.4-2 Hardware interrupt operation**



- (1) An interrupt request is generated within the peripheral function.
- (2) The interrupt enable bit of the peripheral function is referred. If the interrupt is enabled, the interrupt request is output from the peripheral function to the interrupt controller.
- (3) The interrupt controller that receives the interrupt request determines the priority of simultaneous interrupt requests, then transfers the interrupt level (IL) that matches the corresponding interrupt request to the CPU.
- (4) The CPU compares the interrupt level (IL) requested by the interrupt controller with the interrupt level mask register (ILM).
- (5) If the comparison indicates a higher priority than the current interrupt processing level, the CPU checks the contents of the I flag in the condition code register (CCR).
- (6) If in the check in (5) the I flag is interrupt enabled (I = 1), the CPU waits until the execution of the instruction currently being executed terminates. At termination, the CPU sets the requested level (IL) in the ILM.
- (7) Registers are saved, and processing branches to the interrupt processing routine.
- (8) The interrupt cause that was generated in (1) is cleared by software in the interrupt processing routine. Execution of the RETI instruction terminates the interrupt processing.

## MB90820B Series

### 7.4.2 Processing for Interrupt Operation

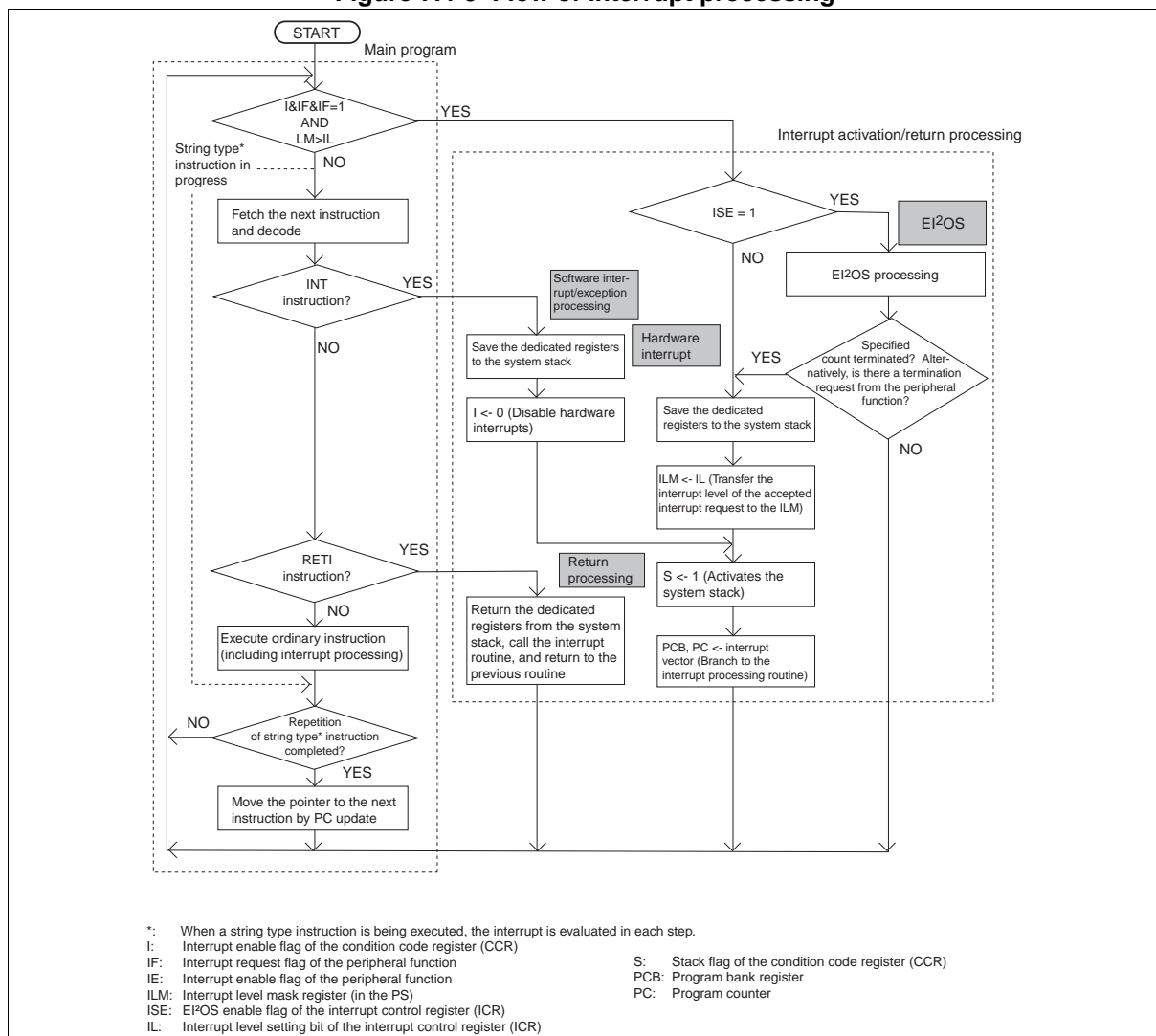
When an interrupt request is generated by the peripheral function, the interrupt controller transmits the interrupt level to the CPU. If the CPU is able to accept interrupt, the interrupt controller temporarily interrupts the instruction currently being executed. The interrupt controller then executes the interrupt processing routine or activates the extended intelligent I/O service (EI<sup>2</sup>OS).

If a software interrupt is generated by the INT instruction, the interrupt processing routine is executed regardless of the CPU status. In this case, hardware interrupt is not allowed.

#### ■ Processing for Interrupt Operation

Figure 7.4-3 shows the flow of processing for interrupt operation.

Figure 7.4-3 Flow of interrupt processing



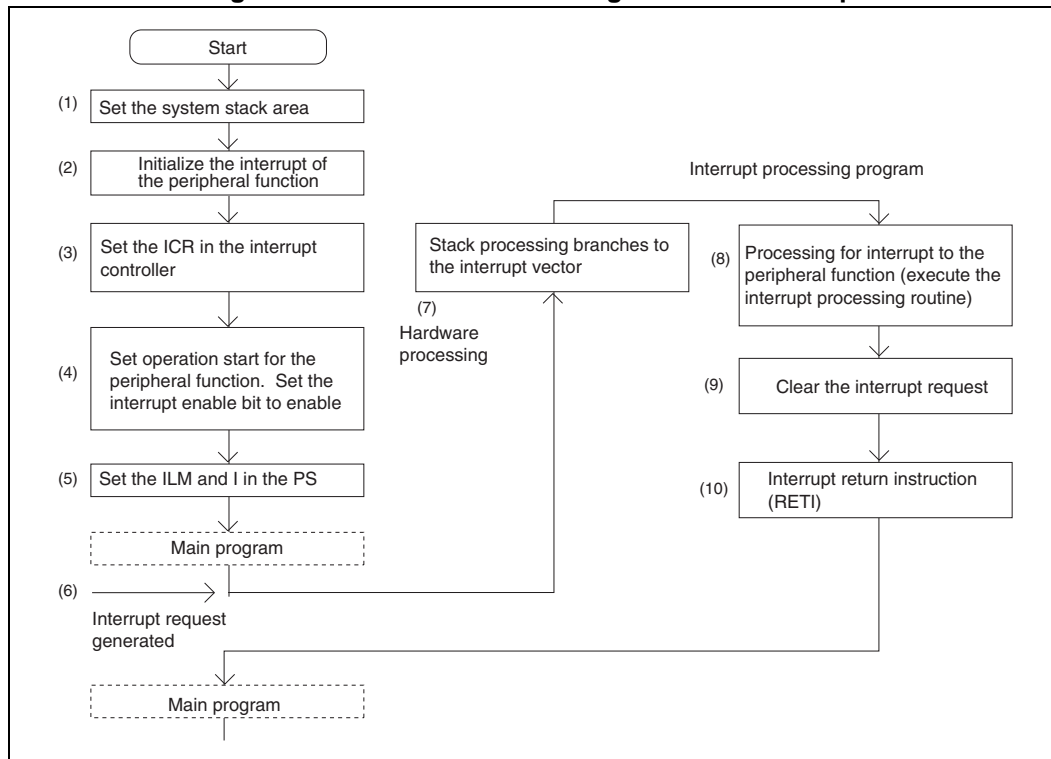
### 7.4.3 Procedure for Using Hardware Interrupt

Before hardware interrupt can be used, the system stack area, peripheral function, and interrupt control register (ICR) must be set.

#### ■ Procedure for Using Hardware Interrupt

Figure 7.4-4 shows an example of the procedure for using hardware interrupt.

Figure 7.4-4 Procedure for using hardware interrupt



- (1) Set the system stack area.
- (2) Initialize a peripheral function that can generate interrupt requests.
- (3) Set the interrupt control register (ICR) in the interrupt controller.
- (4) Set the peripheral function to the operation start status and set the interrupt enable bit to enable.
- (5) Set the interrupt level mask register (ILM) and interrupt enable flag (I) to interrupt acceptable.
- (6) An interrupt generated in the peripheral function causes a hardware interrupt request.
- (7) The interrupt processing hardware saves the registers and branches to the interrupt processing program.
- (8) The interrupt processing program processes the peripheral function in response to the generated interrupt.
- (9) Clear the interrupt request from peripheral function.
- (10) Execute the interrupt return instruction and return to the program before branching.

## 7.4.4 Multiple Interrupts

---

Multiple hardware interrupts can be implemented by setting different interrupt levels in the interrupt level setting bits (IL0, IL1, IL2) of the interrupt control register (ICR) in response to multiple interrupt requests from peripheral functions. Use of multiple interrupts, however, is not possible with the extended intelligent I/O service.

---

### ■ Multiple Interrupts

#### ● Operation of multiple interrupts

During execution of an interrupt processing routine, if an interrupt request with a higher-priority interrupt level is generated, the current interrupt processing is interrupted and the interrupt request with the higher-priority interrupt level is accepted. When the interrupt request with the higher-priority interrupt level terminates, the CPU returns to the previous interrupt processing.

0 to 7 can be set as the interrupt level (IL). If level 7 is set, the CPU does not accept interrupt requests.

During execution of interrupt processing, if an interrupt request with the same or lower-priority interrupt level is generated, the new interrupt request is held until the current interrupt returns unless the I flag in the code condition register (CCR) or ILM is changed.

Other multiple interrupts to be activated during an interrupt can be temporarily disabled by setting the I flag in the condition code register (CCR) in the interrupt processing routine to interrupts not allowed (CCR: I = 0) or the interrupt level mask register (ILM) to interrupts not allowed (ILM = 000<sub>B</sub>).

---

#### Note :

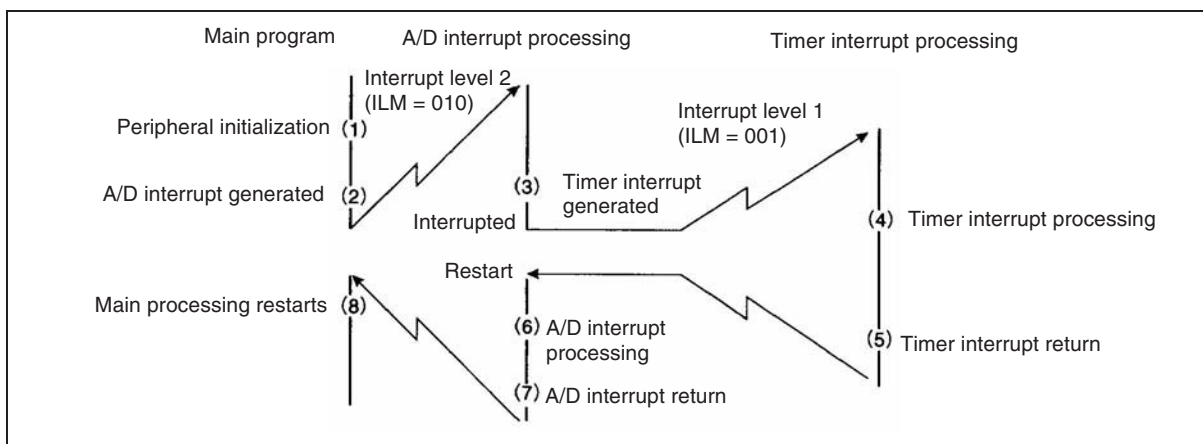
The extended intelligent I/O service (EI<sup>2</sup>OS) cannot be used for the activation of multiple interrupts. During processing of the extended intelligent I/O service (EI<sup>2</sup>OS), all other interrupt requests and extended intelligent I/O service requests are held.

---

#### ● Example of multiple interrupts

This example of multiple interrupt processing assumes that a timer interrupt is given a higher priority than an A/D converter interrupt. In this example, the A/D converter interrupt level is set to 2, and the timer interrupt level is set to 1. If a timer interrupt is generated during processing of the A/D converter interrupt, the processing shown in Figure 7.4-5 is performed.

Figure 7.4-5 Example of multiple interrupts



1) A/D interrupt generated

When the A/D converter interrupt processing starts, the interrupt level mask register (ILM) automatically has the same value (2 in the example) as the A/D converter interrupt level (ICR: IL2 to IL0). If a level-1 or level-0 interrupt request is generated, this interrupt processing has priority.

2) Interrupt processing terminated

When the interrupt processing terminates and the return instruction (RETI) is executed, the values of the dedicated registers (A, DPR, ADB, DTB, PCB, PC, and PS) are returned from the stack, and the interrupt level mask register (ILM) has the value that it had before the interrupt.

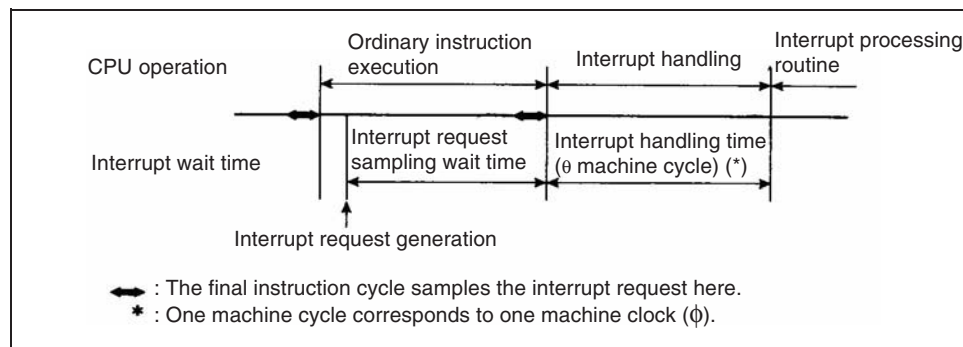
### 7.4.5 Hardware Interrupt Processing Time

From the generation of a hardware interrupt request to the execution of an interrupt processing routine, the time for the instruction currently being executed to terminate and the time required to handle an interrupt are necessary.

#### ■ Hardware Interrupt Processing Time

From the generation of a hardware interrupt request to the acceptance of the interrupt and to the execution of an interrupt processing routine, the time to wait for sampling of an interrupt request and the time required to handle an interrupt (time to prepare for interrupt processing) are necessary. Figure 7.4-6 shows the interrupt processing time.

Figure 7.4-6 Hardware interrupt processing time



#### ● Interrupt request sampling wait time

The interrupt request sampling wait time is the time from the generation of interrupt request and to the termination of the instruction currently being executed.

Whether an interrupt request has been generated is determined by sampling the instruction for an interrupt request in the final cycle of the instruction. Consequently, the CPU cannot identify an interrupt request during execution of each instruction creating a delay time.

The interrupt request sampling wait time is the maximum when an interrupt request is generated as soon as the POPW RW0, ... RW7 instructions (45 machine cycles), which takes the longest to execute, starts.

#### ● Interrupt handling time ( $\phi$ machine cycle)

The CPU saves dedicated registers to the system stack and fetches interrupt vectors after it receives an interrupt request. The required interrupt time for this processing is  $\phi$  machine cycles. The interrupt handling time is calculated with the following formula:

When an interrupt is activated:  $\theta = 24 + 6 + Z$  machine cycles

When control is returned from an interrupt:  $\theta = 11 + 6 + Z$  machine cycles (RETI instruction)

The interrupt handling time is different for each address pointed to by the stack pointer.

Table 7.4-3 shows the interpolation values (Z) for the interrupt handling time.



**Table 7.4-3 Interpolation values (Z) for the interrupt handling time**

Address pointed to by the stack pointer	Interpolation value (Z)
External 8-bit	+4
External even-numbered address	+1
External odd-numbered address	+4
Internal even-numbered address	0
Internal odd-numbered address	+2

---

Reference:

One machine cycle corresponds to one clock cycle of the machine clock ( $\phi$ ).

---

## 7.5 Software Interrupt

---

**When the software interrupt instruction (INT instruction) is executed, the software interrupt function transfers control from the program being executed by the CPU to the user-defined interrupt processing program. Hardware interrupt is disabled during execution of a software interrupt.**

---

### ■ Software Interrupt Activation

#### ● Software interrupt activation

The INT instruction is used to activate a software interrupt. There is no interrupt request flag or enable flag for software interrupt requests. When the INT instruction is executed, an interrupt request is always generated.

#### ● Hardware interrupt suppression

Since the INT instruction does not have interrupt levels, the interrupt level mask register (ILM) is not updated. During the execution of the INT instruction, the I flag of the condition code register (CCR) is set to "0", and hardware interrupts are masked.

To enable hardware interrupts during software interrupt processing, set the I flag to "1" in the software interrupt processing routine.

#### ● Software interrupt operation

When the CPU fetches the INT instruction, the software interrupt processing microcode is activated. This microcode saves the internal CPU registers on the system stack, masks hardware interrupts (CCR: I = 0), and branches to the corresponding interrupt vector.

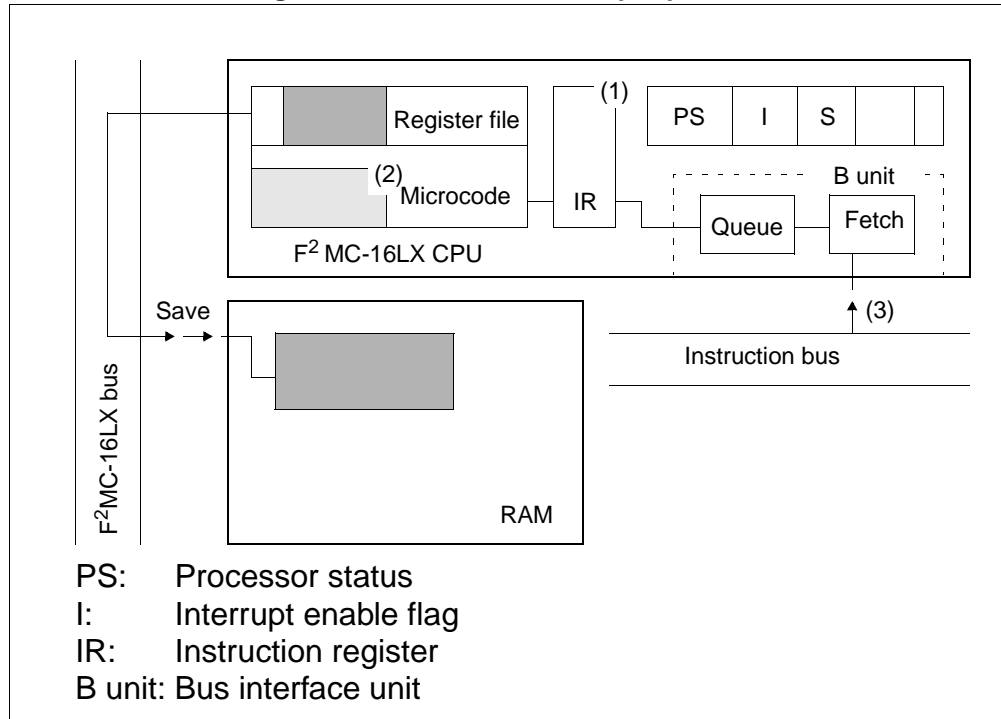
### ■ Returning from a Software Interrupt

In the interrupt processing program, when the interrupt return instruction (RETI instruction) is executed, the 12-byte data saved to the system stack is restored to the dedicated registers and the processing that was being executed before branching for the interrupt is resumed.

## ■ Software Interrupt Operation

Figure 6.5-1 shows software interrupt operation from the generation of a software interrupt to the completion of interrupt processing.

**Figure 7.5-1 Software interrupt operation**



- (1) A software interrupt instruction is executed.
- (2) The dedicated registers are saved according to the microcode that corresponds to the software interrupt instruction, and other necessary processing is performed. Branch processing is then executed.
- (3) The RETI instruction in the user interrupt processing routine terminates the interrupt processing.

### Note :

When the program bank register (PCB) is FF<sub>H</sub>, the vector area of the CALLV instruction overlaps the INT #vct8 instruction table. When creating the software, be careful of the duplicated address of the CALLV instruction and INT #vct8 instruction.

## 7.6 Interrupt of Extended Intelligent I/O Service (EI<sup>2</sup>OS)

---

The extended intelligent I/O service (EI<sup>2</sup>OS) automatically transfers data between a peripheral function (I/O) and memory. When the data transfer terminates, a hardware interrupt is generated.

---

### ■ Extended Intelligent I/O Service (EI<sup>2</sup>OS)

The extended intelligent I/O service is a type of hardware interrupt. It automatically transfers data between a peripheral function (I/O) and a memory. Traditionally, data transfer between a peripheral function (I/O) and a memory has been performed by the interrupt processing program. EI<sup>2</sup>OS performs this data transfer in the same way as direct memory access (DMA). At termination, EI<sup>2</sup>OS sets the termination condition and automatically branches to the interrupt processing routine. The user creates programs only for EI<sup>2</sup>OS activation and termination. Data transfer programs in between are not required.

#### ● Advantages of extended intelligent I/O service (EI<sup>2</sup>OS)

Compared to data transfer performed by the interrupt processing routine, EI<sup>2</sup>OS has the following advantages.

- Coding a data transfer program is not necessary, reducing program size.
- Because data transfer can be stopped depending on the peripheral function (I/O) status, unnecessary data transfer can be eliminated.
- Update or no update can be selected for the buffer address.
- Update or no update can be selected for the I/O register address.

#### ● Extended intelligent I/O service (EI<sup>2</sup>OS) termination interrupt

When data transfer by EI<sup>2</sup>OS terminates, a termination condition is set in the S1 and S0 bits in the interrupt control register (ICR). Processing then automatically branches to the interrupt processing routine.

The EI<sup>2</sup>OS termination factor can be determined by checking the EI<sup>2</sup>OS status (ICR: S1 and S0) with the interrupt processing program.

Interrupt numbers and interrupt vectors are permanently set for each peripheral. See Section "7.2 Interrupt Causes and Interrupt Vectors", in "CHAPTER 7 INTERRUPT" for more information.

#### ● Interrupt control register (ICR)

This register, which is located in the interrupt controller, activates EI<sup>2</sup>OS, specifies the EI<sup>2</sup>OS channel, and displays the EI<sup>2</sup>OS termination status.

#### ● Extended intelligent I/O service (EI<sup>2</sup>OS) descriptor (ISD)

This descriptor, which is located in internal RAM at 000100<sub>H</sub> to 00017F<sub>H</sub>, is an eight-byte data that retains the transfer mode, I/O address, transfer count, and buffer address. The descriptor handles 16 channels. The channel is specified by the interrupt control register (ICR).

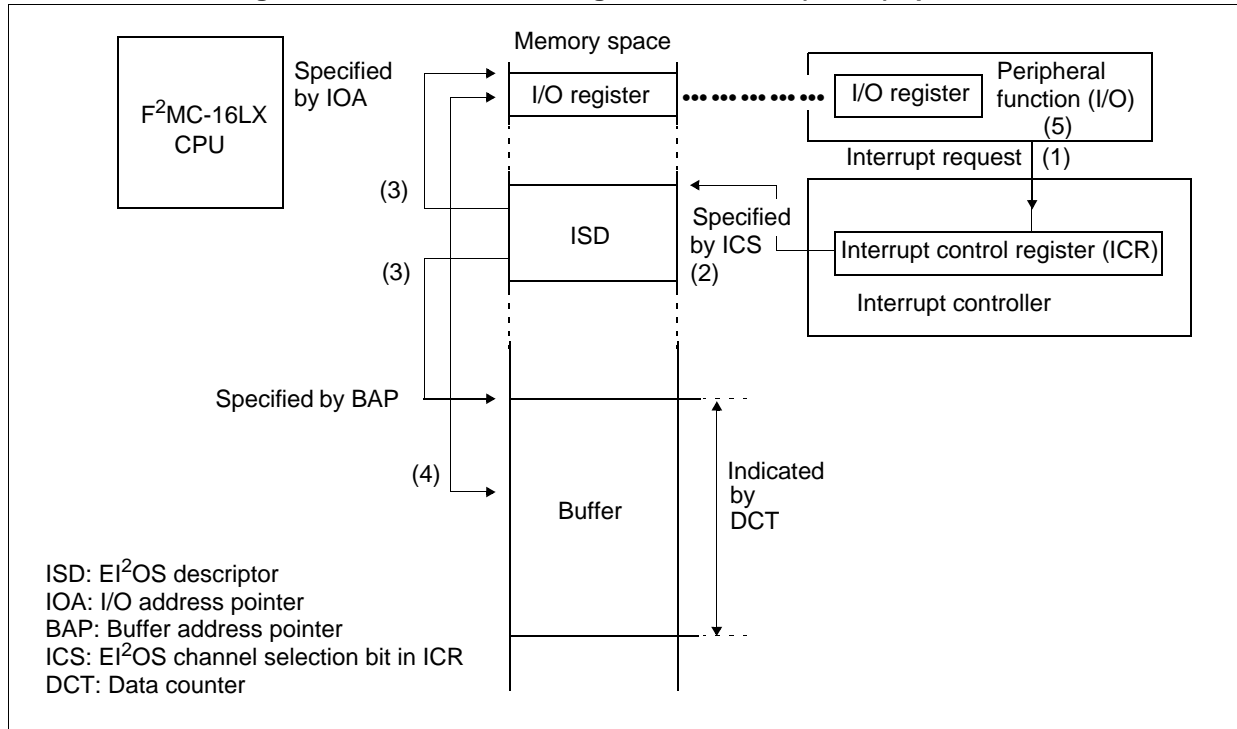
Note:

When the extended intelligent I/O service (EI<sup>2</sup>OS) is operating, execution of the CPU program stops.

## ■ Operation of the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

Figure 6.6-1 shows EI<sup>2</sup>OS operation.

Figure 7.6-1 Extended intelligent I/O service (EI<sup>2</sup>OS) operation



- (1) I/O requests transfer.
- (2) The interrupt controller selects the descriptor.
- (3) The transfer source and transfer destination are read from the descriptor.
- (4) Transfer is performed between I/O and memory.
- (5) The interrupt cause is automatically cleared.

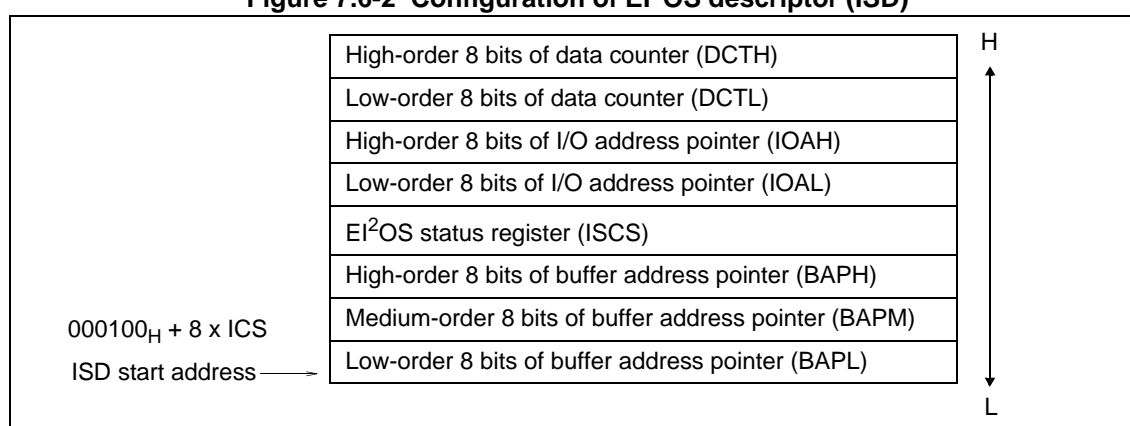
## 7.6.1 Extended Intelligent I/O Service (EI<sup>2</sup>OS) Descriptor (ISD)

The extended intelligent I/O service (EI<sup>2</sup>OS) descriptor (ISD) resides in internal RAM at 000100<sub>H</sub> to 00017F<sub>H</sub>. The ISD consists of 8 bytes x 16 channels.

### ■ Configuration of the Extended Intelligent I/O Service (EI<sup>2</sup>OS) Descriptor (ISD)

The ISD consists of 8 bytes x 16 channels. Each ISD has the structure shown in Figure 7.6-2. Table 7.6-1 shows the correspondence between channel numbers and ISD addresses.

**Figure 7.6-2 Configuration of EI<sup>2</sup>OS descriptor (ISD)**



**Table 7.6-1 Correspondence between channel numbers and ISD addresses**

Channel	Descriptor address
0	000100 <sub>H</sub>
1	000108 <sub>H</sub>
2	000110 <sub>H</sub>
3	000118 <sub>H</sub>
4	000120 <sub>H</sub>
5	000128 <sub>H</sub>
6	000130 <sub>H</sub>
7	000138 <sub>H</sub>
8	000140 <sub>H</sub>
9	000148 <sub>H</sub>
10	000150 <sub>H</sub>
11	000158 <sub>H</sub>
12	000160 <sub>H</sub>
13	000168 <sub>H</sub>
14	000170 <sub>H</sub>
15	000178 <sub>H</sub>

## 7.6.2 Registers of EI<sup>2</sup>OS Descriptor (ISD)

- Data counter (DCT)
- I/O register address pointer (IOA)
- EI<sup>2</sup>OS status register (ISCS)
- Buffer address pointer (BAP)

Note that the initial value of each register is undefined after a reset.

### ■ Data Counter (DCT)

The DCT is a 16-bit register that serves as a counter for the data transfer count. After each data transfer, the counter is decremented by 1. When the counter reaches zero, EI<sup>2</sup>OS terminates.

Figure 7.6-3 shows the configuration of the DCT.

Figure 7.6-3 Configuration of DCT

<b>DCTH</b>										
	bit	15	14	13	12	11	10	9	8	Initial value: xxxxxxxx <sub>B</sub>
Upper byte of data counter		B15	B14	B13	B12	B11	B10	B09	B08	
<b>DCTL</b>										
	bit	7	6	5	4	3	2	1	0	Initial value: xxxxxxxx <sub>B</sub>
Lower byte of data counter		B07	B06	B05	B04	B03	B02	B01	B00	

### ■ I/O Register Address Pointer (IOA)

The IOA is a 16-bit register that indicates the lower address (A15 to A00) of the I/O register used to transfer data to and from the buffer. The upper address (A23 to A16) is all zeros. Any I/O from 000000<sub>H</sub> to 00FFFF<sub>H</sub> can be specified by address.

Figure 7.6-4 shows the configuration of the IOA.

Figure 7.6-4 Configuration of I/O register address pointer (IOA)

<b>IOAH</b>										
	bit	15	14	13	12	11	10	9	8	Initial value:
Upper address pointer		A15	A14	A13	A12	A11	A10	A09	A08	xxxxxxxx <sub>B</sub>

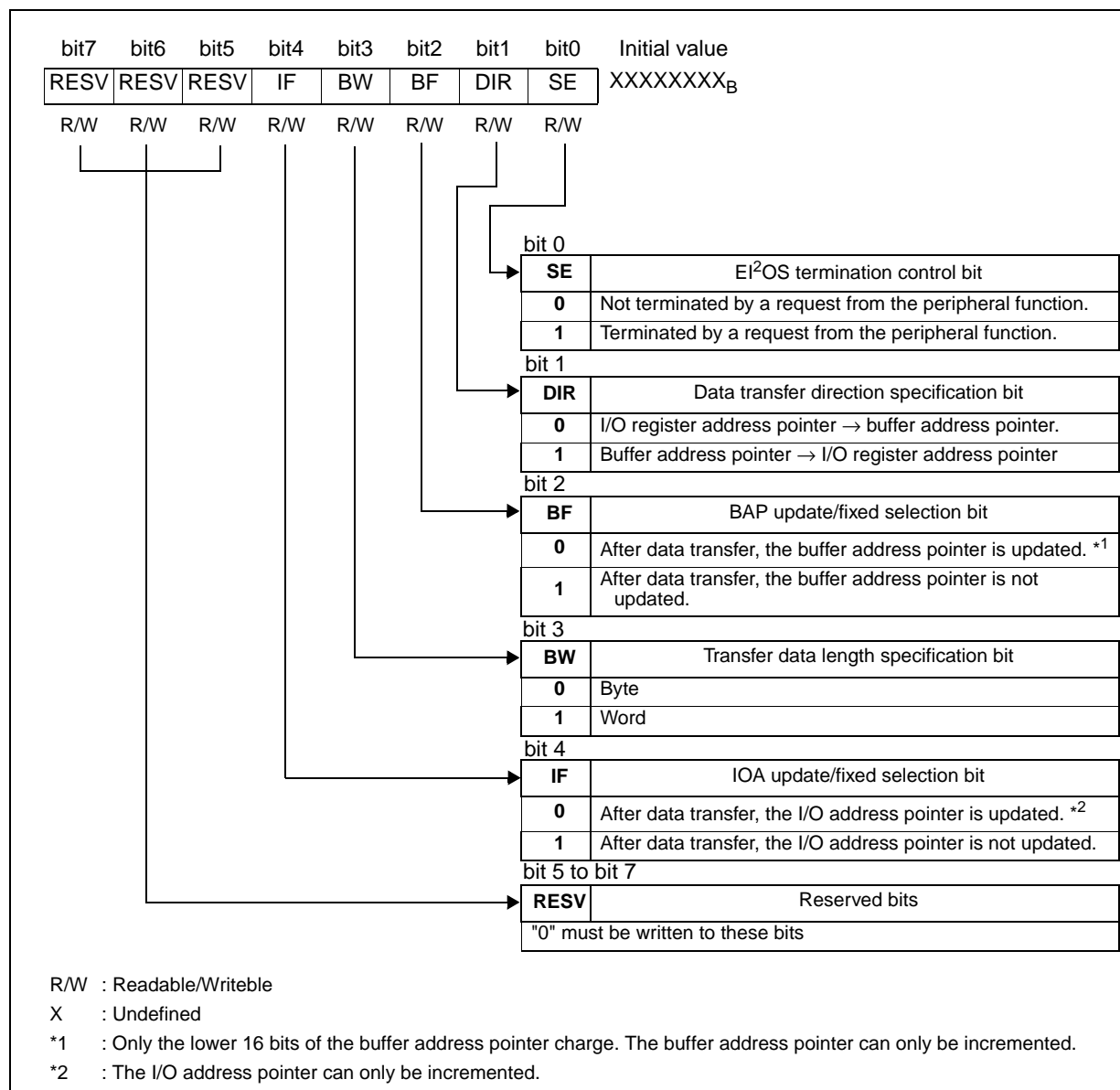
<b>IOAL</b>										
	bit	7	6	5	4	3	2	1	0	Initial value:
Lower address pointer		A07	A06	A05	A04	A03	A02	A01	A00	xxxxxxxx <sub>B</sub>

## Extended Intelligent I/O Service (EI<sup>2</sup>OS) Status Register (ISCS)

The ISCS is an 8-bit register. The ISCS indicates the update/fixed for the buffer address pointer and I/O register address pointer, transfer data format (byte or word), and transfer direction.

Figure 7.6-5 shows the configuration of the ISCS.

Figure 7.6-5 Configuration of EI<sup>2</sup>OS status register (ISCS)

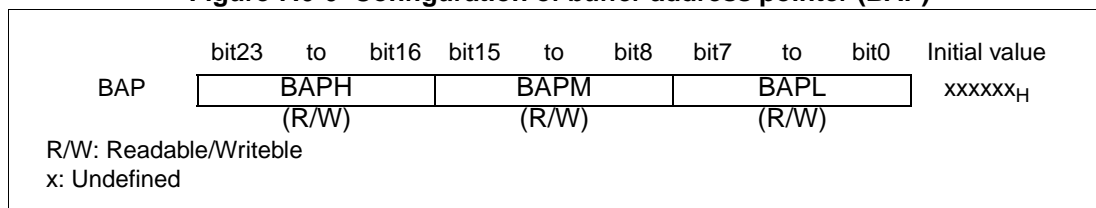


## Buffer address Pointer (BAP)

The BAP is a 24-bit register that retains the address used by EI<sup>2</sup>OS for the next transfer. Since one independent BAP exists for each EI<sup>2</sup>OS channel, each EI<sup>2</sup>OS channel can transfer data between any address in the 16-megabyte space and the I/O. If the BF bit (BAP update/fixed selection bit) in the EI<sup>2</sup>OS status register (ISCS) is set to "update yes," only the lower 16 bits (BAPM and BAPL) of the BAP change; the upper 8 bits (BAPH) do not change. Figure 7.6-6 shows the configuration of the BAP.



**Figure 7.6-6 Configuration of buffer address pointer (BAP)**



References:

- The area that can be specified by the I/O address pointer (IOA) extends from 000000<sub>H</sub> to 00FFFF<sub>H</sub>.
- The area that can be specified with the buffer address pointer (BAP) extends from 000000<sub>H</sub> to FFFFFFF<sub>H</sub>.
- The maximum transfer count that can be specified by the data counter (DCT) is 65536 (64 kilobytes).

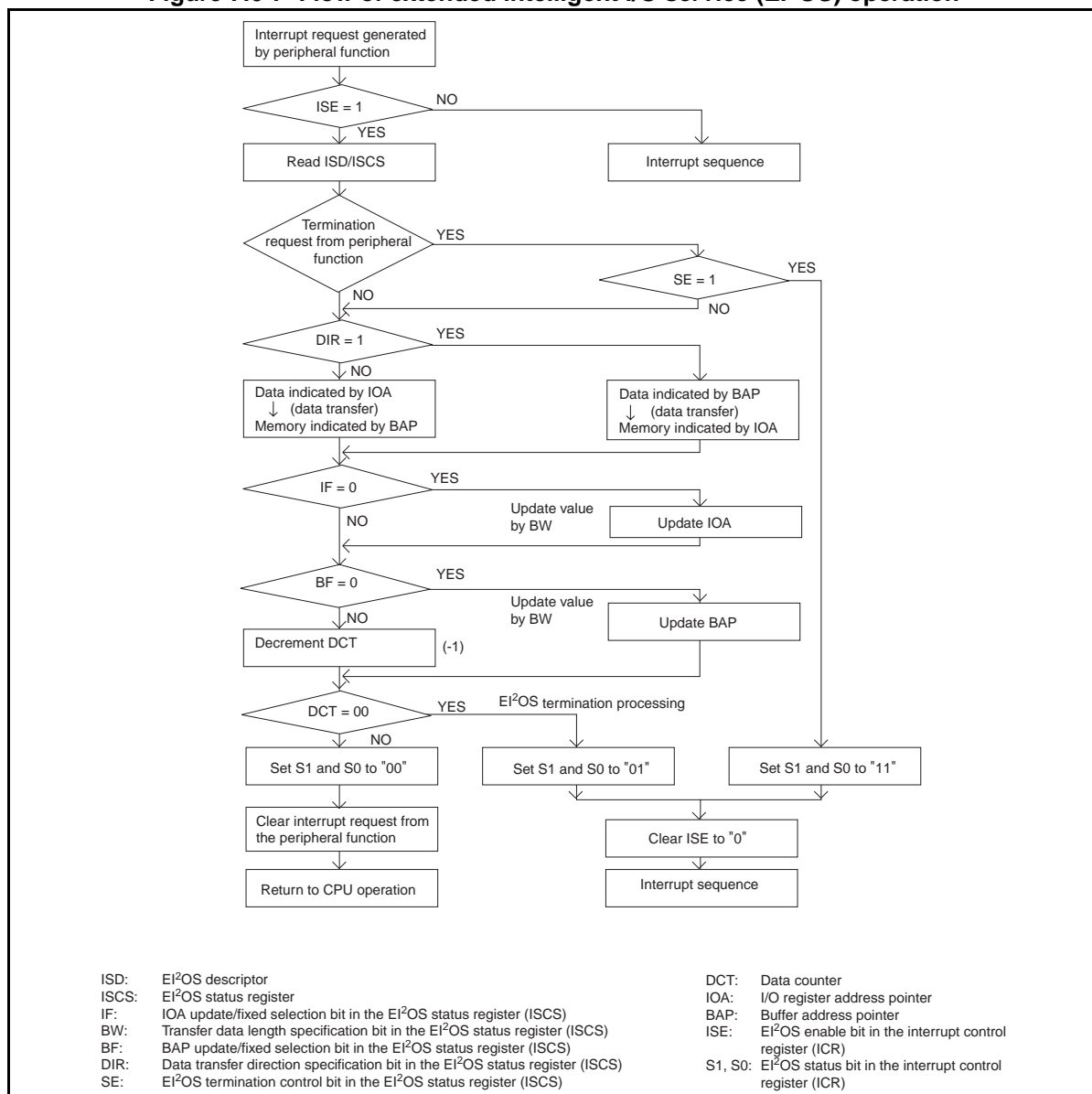
### 7.6.3 Operation of the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

If an interrupt request is generated by a peripheral function, EI<sup>2</sup>OS activation is set in the corresponding interrupt control register (ICR) that the CPU uses EI<sup>2</sup>OS to transfer data. When the specified data transfer count terminates, the hardware interrupt is automatically processed.

#### ■ Operation Flow of the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

Figure 7.6-7 shows the flow of EI<sup>2</sup>OS operation based on the internal microcode of the CPU.

Figure 7.6-7 Flow of extended intelligent I/O service (EI<sup>2</sup>OS) operation



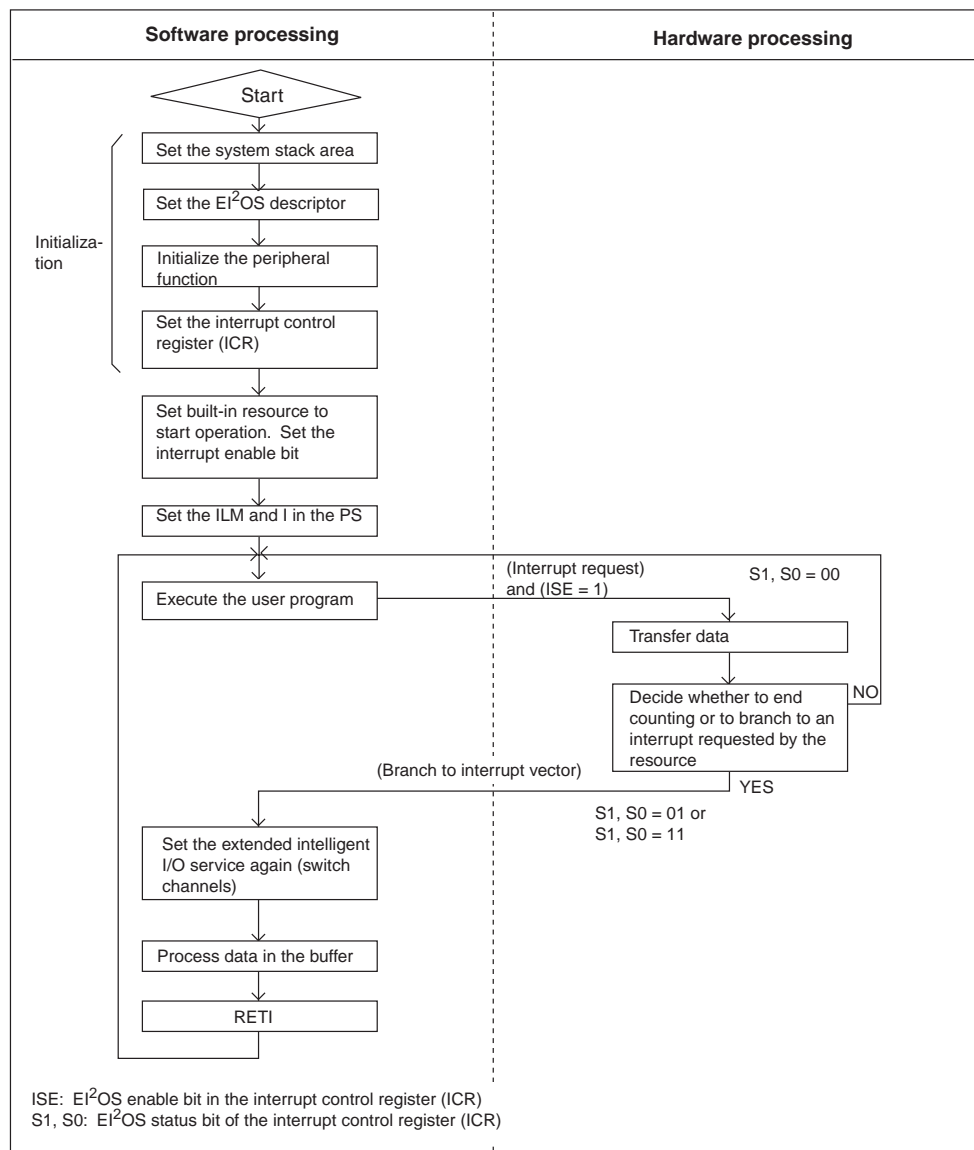
## 7.6.4 Procedure for Using the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

Before the extended intelligent I/O service (EI<sup>2</sup>OS) can be used, the system stack area, extended intelligent I/O service (EI<sup>2</sup>OS) descriptor, interrupt function, and interrupt control register (ICR) must be set.

### ■ Procedure for Using the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

Figure 7.6-8 shows the EI<sup>2</sup>OS software and hardware processing.

Figure 7.6-8 Procedure for using the extended intelligent I/O service (EI<sup>2</sup>OS)



## MB90820B Series

### 7.6.5 Processing Time of the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

The time required for processing the extended intelligent I/O service (EI<sup>2</sup>OS) changes according to the following factors:

- EI<sup>2</sup>OS status register (ISCS) setting
- Address (area) pointed to by the I/O register address pointer (IOA)
- Address (area) pointed to by the buffer address pointer (BAP)
- External data bus length for external access
- Transfer data length

Because the hardware interrupt is activated when data transfer by EI<sup>2</sup>OS terminates, the interrupt handling time is added.

#### ■ Processing Time (One Transfer Time) of the Extended Intelligent I/O Service (EI<sup>2</sup>OS)

- When data transfer continues

The EI<sup>2</sup>OS processing time for data transfer continuation is shown in Table 7.6-2 based on the EI<sup>2</sup>OS status register (ISCS) setting.

**Table 7.6-2 Extended intelligent I/O service execution time**

EI <sup>2</sup> OS termination control bit (SE) setting		Terminates due to termination request from the peripheral function		Ignores termination request from the peripheral function	
IOA update/fixed selection bit (IF) setting		Fixed	Update	Fixed	Update
BAP address update/fixed selection bit (BF) setting	Fixed	32	34	33	35
	Update	34	36	35	37

Unit: Machine cycle (One machine cycle corresponds to one clock cycle of the machine clock,  $\phi$ ).

As shown in Table 7.6-3, interpolation is necessary depending on the EI<sup>2</sup>OS execution condition.

**Table 7.6-3 Data transfer interpolation value for EI<sup>2</sup>OS execution time**

I/O register address pointer			Internal access		External access	
			B/Even	Odd	B/Even	8/Odd
Buffer address pointer	Internal access	B/Even	0	+2	+1	+4
		Odd	+2	+4	+3	+6
	External access	B/Even	+1	+3	+2	+5
		8/Odd	+4	+6	+5	+8

B: Byte data transfer

8: External bus width using the 8-bit word transfer

Even: Even-numbered address word transfer

Odd: Odd-numbered address word transfer

● When the data counter (DCT) count terminates (final data transfer)

Because the hardware interrupt is activated when data transfer by EI<sup>2</sup>OS terminates, the interrupt handling time is added. The EI<sup>2</sup>OS processing time when counting terminates is calculated with the following formula:

EI<sup>2</sup>OS processing time when counting terminates = EI<sup>2</sup>OS processing time when data is transferred +  $(21 + 6 \times Z)$  Machine cycles

↑  
Interrupt handling time

The interrupt handling time is different for each address pointed to by the stack pointer. Table 7.6-4 shows the interpolation value (Z) for the interrupt handling time.

**Table 7.6-4 Interpolation value (Z) for the interrupt handling time**

Address pointed to by the stack pointer	Interpolation value (Z)
External 8-bit	+4
External even-numbered address	+1
External odd-numbered address	+4
Internal even-numbered address	0
Internal odd-numbered address	+2

● For termination by a termination request from the peripheral function (I/O)

When data transfer by EI<sup>2</sup>OS is terminated before completion due to a termination request from the peripheral function (I/O) (ICR: S1, S0 = 11<sub>B</sub>), the data transfer is not performed and a hardware interrupt is activated. The EI<sup>2</sup>OS processing time is calculated with the following formula. Z in the formula indicates the interpolation value for the interrupt handling time (Table 6.6.5-3).

EI<sup>2</sup>OS processing time for termination of data transfer =  $36 + 6 \times Z$  Machine cycle

---

Reference:

One machine cycle corresponds to one clock cycle of the machine clock ( $\phi$ ).

---

## MB90820B Series

### 7.7 Exception Processing Interrupt

---

In the F<sup>2</sup>MC-16LX, the execution of an undefined instruction results in exception processing.

Exception processing is basically the same as an interrupt. When the generation of an exception processing is detected on the instruction boundary, ordinary processing is interrupted and exception processing is executed.

Generally, exception processing occurs as the result of an unexpected operation. Exception processing should be used only to activate recovery software required for debugging or an emergency.

---

#### ■ Exception Processing

##### ● Exception processing operation

The F<sup>2</sup>MC-16LX handles all codes that are not defined in the instruction map as undefined instructions. When an undefined instruction is executed, processing equivalent to the INT #10 software interrupt instruction is executed.

The following processing is executed before exception processing branches to the interrupt routine:

- The A, DPR, ADB, DTB, PCB, PC, and PS registers are saved to the system stack.
- The I flag of the condition code register (CCR) is cleared to "0", and hardware interrupts are masked.
- The S flag of the condition code register (CCR) is set to "1", and the system stack is activated.

The program counter (PC) value saved to the system stack is the exact address where the undefined instruction is stored. For 2-byte or longer instruction codes, the code identified as undefined is stored at this address. When the exception factor type must be determined within the exception processing routine, use this PC value.

##### ● Return from exception processing

When the RETI instruction returns control from exception processing, exception processing restarts because the PC is pointing to the undefined instruction. Provide a solution such as resetting the software.

## 7.8 Stack Operations for Interrupt Processing

Once an interrupt is accepted, the contents of the dedicated registers are automatically saved to the system stack before a branch to interrupt processing. When the interrupt processing terminates, the previous processing is automatically restored from the system stack.

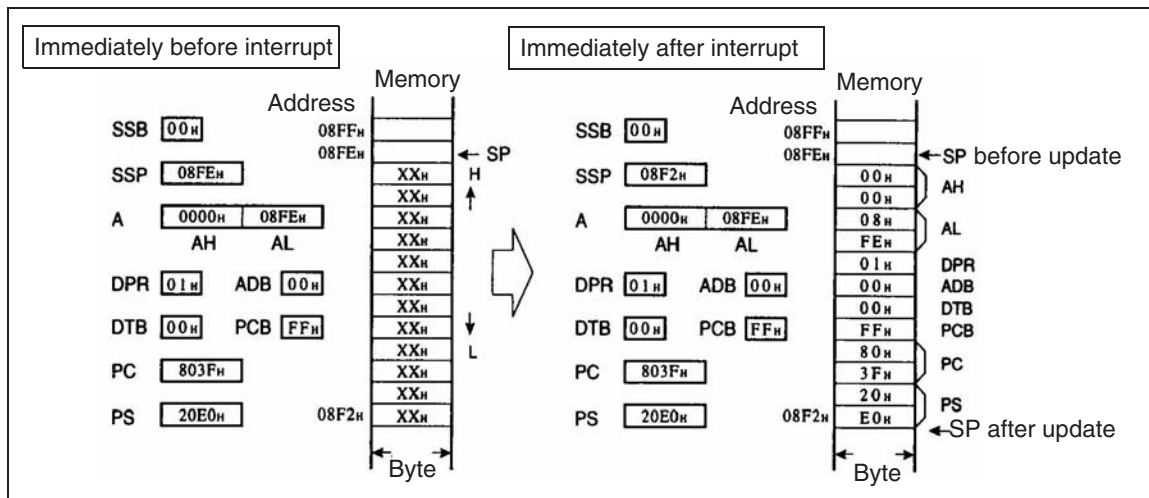
### ■ Stack Operations at the Start of Interrupt Processing

Once an interrupt is accepted, the CPU automatically saves the contents of the current dedicated registers to the system stack in the order given below:

- Accumulator (A)
- Direct page register (DPR)
- Additional data bank register (ADB)
- Data bank register (DTB)
- Program bank register (PCB)
- Program counter (PC)
- Processor status (PS)

Figure 7.8-1 shows the stack operations at the start of interrupt processing.

Figure 7.8-1 Stack operations at the start of interrupt processing



### ■ Stack Operations on Return from Interrupt Processing

When the interrupt return instruction (RETI) is executed at the termination of interrupt processing, dedicated register (the PS, PC, PCB, DTB, ADB, DPR, and A) values are returned from the stack in reverse order from the order they were placed on the stack. The dedicated registers are restored to the status they had immediately before the start of interrupt processing.

## ■ Stack Area

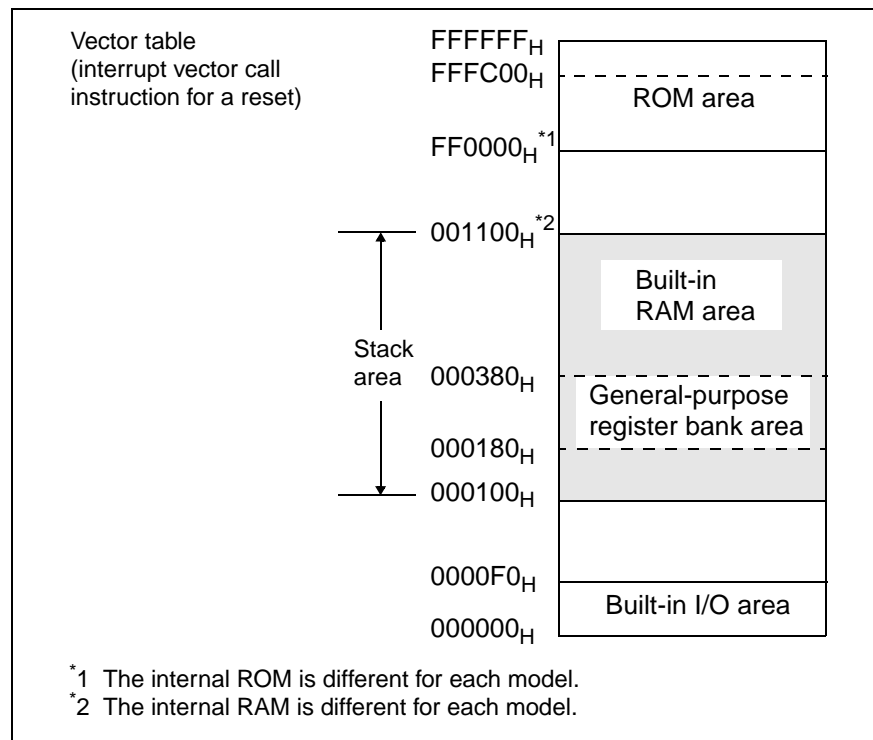
### ● Stack area allocation

The stack area is used for saving and restoring the program counter (PC) when the subroutine call instruction (CALL) and vector call instruction (CALLV) are executed in addition to interrupt processing. The stack area is used for temporary saving and restoring of registers by the PUSHW and POPW instructions.

The stack area is allocated together with the data area in RAM.

Figure 7.8-2 shows the stack area.

**Figure 7.8-2 Stack area**



### Notes:

- Generally set an even-numbered address in the stack pointers (SSP and USP).
- Allocate the system stack area, user stack area, and data area so that they do not overlap.

### ● System stack and user stack

The system stack area is used for interrupt processing. When an interrupt occurs, the user stack area being used is forcibly switched to the system stack. The system stack area must be set correctly even in a system that mainly uses the user stack area.

If division of the stack space is not particularly necessary, use only the system stack.





# **CHAPTER 8**

---

# **MODE SETTING**

**This chapter describes the operating modes and memory access modes supported by the MB90820B series.**

- 8.1 Mode Setting
- 8.2 Mode Pins (MD2 to MD0)
- 8.3 Mode Data

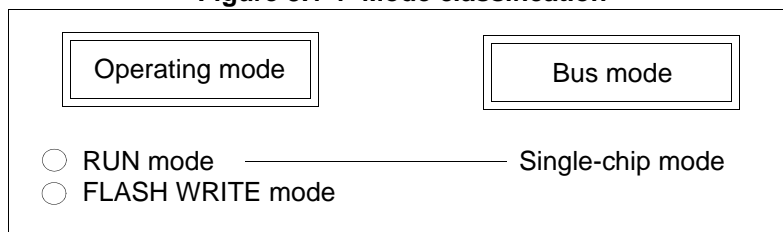
## 8.1 Mode Setting

The F<sup>2</sup>MC-16LX supports the modes for access method and access areas. A mode is determined based on the settings by the mode pin at a reset as well as the mode data fetched.

### ■ Mode Setting

The F<sup>2</sup>MC-16LX supports the modes for access method and access areas, classified as shown in Figure 8.1-1 in this module.

**Figure 8.1-1 Mode classification**



### ■ Operating Modes

The operating modes control the operating state of the device and are specified by the mode setting pin (MDx) and Mx bit contents in mode data.

#### ● Bus mode

The bus mode controls the operation of internal ROM and external access functions and is specified by the mode setting pin (MDx) and Mx bit contents in mode data. The mode setting pin (MDx) specifies bus mode when reset vector and mode data are read. The Mx bit in mode data specifies bus mode during normal operation.

#### ● RUN mode

The RUN mode means CPU operating mode. The RUN mode includes main clock mode, PLL clock mode, and various low-power consumption modes. See "CHAPTER 6 LOW-POWER CONSUMPTION MODE", for details.

# MB90820B Series

## 8.2 Mode Pins (MD2 to MD0)

Three external pins, MD2 to MD0, are supported as the mode pins. These are used to specify how the reset vector and mode data are fetched.

### ■ Mode Pins (MD2 to MD0)

The mode pins are used to select the data bus (external or internal) used for reading the reset vector and to specify the bus width when the external data bus is selected.

For built-in FLASH memory, the mode pins are also used to specify FLASH programming mode, which is used to write programs and other data to internal FLASH memory.

Table 7.2-1 shows the mode pin settings.

**Table 8.2-1 Mode pin settings**

MD2	MD1	MD0	Mode name	Reset vector access area	External data bus width	Remarks
0	0	0	Setting not allowed			
0	0	1				
0	1	0				
0	1	1	Internal vector mode	Internal	Mode data	The reset sequence and subsequent sequences are controlled by mode data.
1	0	0	Setting not allowed			
1	0	1				
1	1	0	FLASH serial write mode	-	-	-
1	1	1	FLASH memory mode	-	-	Mode when the parallel writer is used.

MD2 to MD0: Connect the pins to Vss for "0" and to Vcc for "1".

\*: The flash serial write mode cannot be executed by just setting the mode pins. Other terminal also need to be set. For details, see "APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing".

#### Note:

Because the MB90820B series is only used in single-chip mode, set MD2, MD1, and MD0 to "011<sub>B</sub>" and set M1 and M0 to "00<sub>B</sub>".

## 8.3 Mode Data

The mode data is at memory location  $\text{FFFFDF}_\text{H}$  and is used to specify the operation after a reset sequence. The mode data is automatically fetched to the CPU.

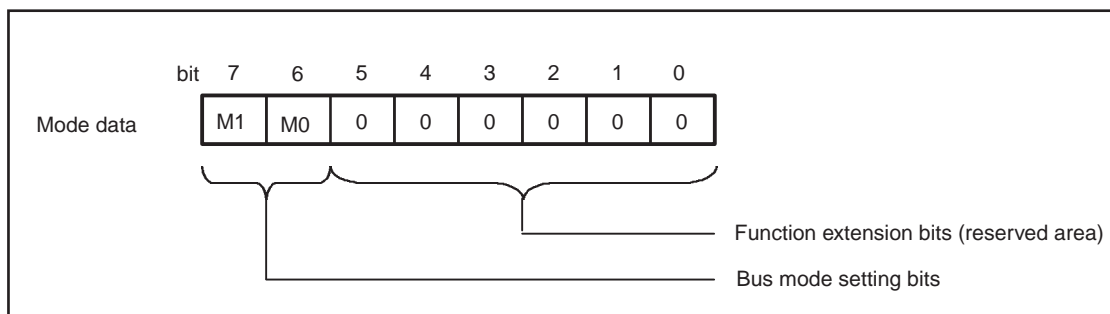
### ■ Mode Data

During a reset sequence, the mode data at address  $\text{FFFFDF}_\text{H}$  is fetched to the mode data register in the CPU. The CPU uses the mode data to set the memory access mode.

The contents of the mode data register can only be changed during the reset sequence. The memory access mode set by the mode data takes effect after the reset sequence.

Figure 7.3-1 shows the mode data configuration.

Figure 8.3-1 Mode data configuration



### ■ Bus Mode Setting Bits

The bus mode setting bits specify operating mode after a reset sequence. Table 7.3-1 lists the relationship between the bits and functions.

Table 8.3-1 Bus mode setting bits and functions

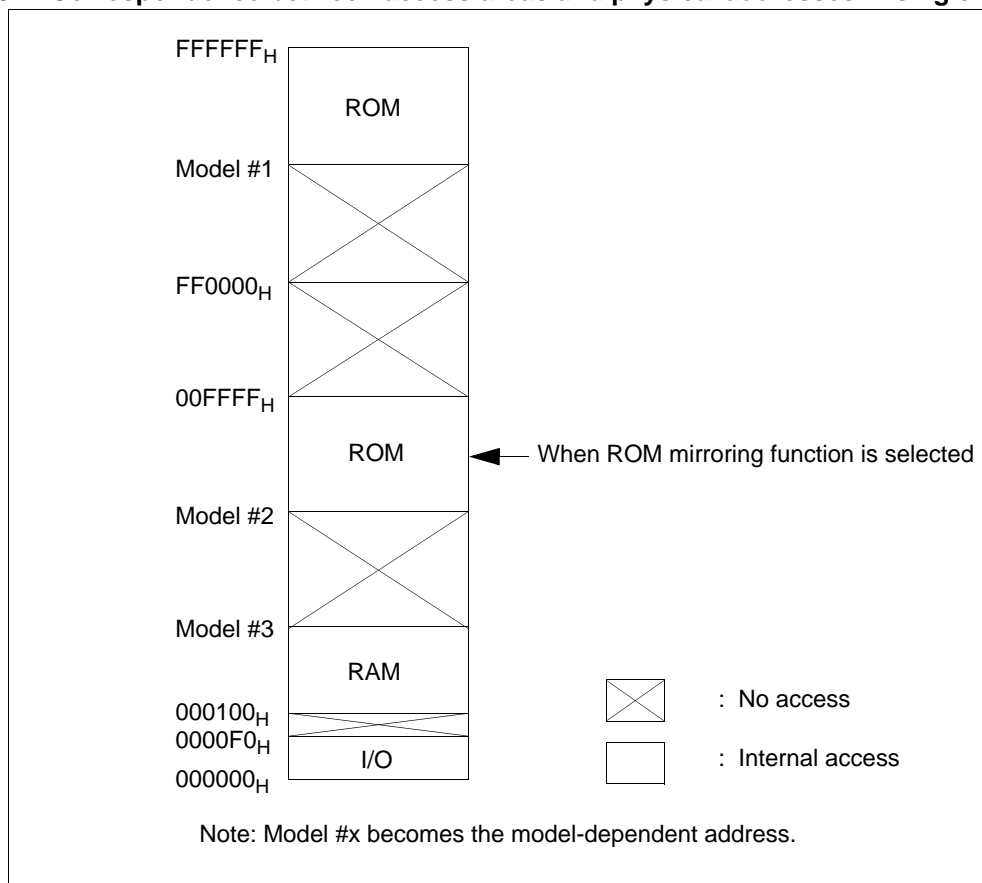
M1	M0	Function
0	0	Single-chip mode
0	1	(Setting not allowed)
1	0	
1	1	

Note:

Because the MB90820B series is only used in single-chip mode, set MD2, MD1, and MD0 to "011<sub>B</sub>" and set M1 and M0 to "00<sub>B</sub>".

Figure 7.3-2 shows the correspondence between access areas and physical addresses in single-chip mode.

**Figure 8.3-2 Correspondence between access areas and physical addresses in single-chip mode**



## ■ Relationship Between Mode Pins and Mode Data

Table 7.3-2 lists the relationship between mode pins and mode data.

**Table 8.3-2 Relationship between mode pins and mode data**

Mode	Mode pins			Mode data	
	MD2	MD1	MD0	M1	M0
Single-chip mode	0	1	1	0	0

Note :

Because the MB90820B series is only used in single-chip mode, set MD2, MD1, and MD0 to "011<sub>B</sub>" and set M1 and M0 to "00<sub>B</sub>".



# **CHAPTER 9**

---

## ***I/O PORTS***

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

- 9.1 Overview of I/O Ports
- 9.2 Registers of I/O Ports
- 9.3 Port 0
- 9.4 Port 1
- 9.5 Port 2
- 9.6 Port 3
- 9.7 Port 4
- 9.8 Port 5
- 9.9 Port 6
- 9.10 Port 7
- 9.11 Port 8



## 9.1 Overview of I/O Ports

---

All I/O ports can be used as general-purpose I/O ports (parallel I/O ports). The MB90820B series has 9 ports (66 pins). The ports are also used for resource I/O pins (peripheral function I/O pins).

---

### ■ I/O Ports Functions

Each I/O port outputs data from CPU to I/O pins or inputs signals from I/O pins to CPU through port data register (PDR). Direction of the data flow (input or output) for each I/O pin can be designated in bit unit by port direction register (DDR). The function of each port and the resource I/O multiplexed with it are described below:

- Port 0 : General-purpose I/O port/resource (PWC)
- Port 1 : General-purpose I/O port/resources (DTP / Multi-functional timer)
- Port 2 : General-purpose I/O port/resource (16-bit reload timer)
- Port 3 : General-purpose I/O port/resource (16-bit PPG timer)
- Port 4 : General-purpose I/O port/resources (16-bit PPG timer / 16-bit reload timer / UART / PWC)
- Port 5 : General-purpose I/O port/resources (16-bit PPG timer / DTP)
- Port 6 : General-purpose I/O port/resource (8/10-bit A/D converter)
- Port 7 : General-purpose I/O port/resources (8/10-bit A/D converter / 8-bit D/A converter / UART / 16-bit free-run timer / 16-bit input capture)
- Port 8 : General-purpose I/O port/resources (16-bit input capture / Multi-functional timer)

# MB90820B Series

Table 9.1-1 summarizes the functions of individual port.

**Table 9.1-1 Functions of individual port**

Port	Pin	Input type	Output type	Function	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Port 0	P00 to P07/ PWO0	CMOS (hysteresis)	CMOS pull-up resistor selectable	General I/O port	–	–	–	–	–	–	–	–	P07	P06	P05	P04	P03	P02	P01	P00
				Resource	–	–	–	–	–	–	–	–	PW00	PW10	–	–	–	–	–	–
Port 1	P10/INT0/ DTT1 to P17			General I/O port	P17	P16	P15	P14	P13	P12	P11	P10	–	–	–	–	–	–	–	–
				Resource	–	INT6	INT5	INT4	INT3	INT2	INT1	INT0 DTT1	–	–	–	–	–	–	–	–
Port 2	P20/TIN1 to P27	CMOS (hysteresis)	CMOS pull-up resistor selectable	General I/O port	–	–	–	–	–	–	–	–	P27	P26	P25	P24	P23	P22	P21	P20
				Resource	–	–	–	–	–	–	–	–	–	–	–	–	–	TO1	TIN1	–
Port 3	P30 to P37/ PPG0			General I/O port	P37	P36	P35	P34	P33	P32	P31	P30	–	–	–	–	–	–	–	–
				Resource	PPG0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Port 4	P40/PPG1 to P47/ PWO1	CMOS (hysteresis)	CMOS	General I/O port	–	–	–	–	–	–	–	–	P47	P46	P45	P44	P43	P42	P41	P40
				Analog output	–	–	–	–	–	–	–	–	PW01	PW11	SIN0*	SOT0	SCK0	TO0	TIN0	PPG1
Port 5	P50/PPG2 to P51/ INT7			General I/O port	–	–	–	–	–	–	P51	P50	–	–	–	–	–	–	–	–
				Resource	–	–	–	–	–	–	INT7	PPG2	–	–	–	–	–	–	–	–
Port 6	P60/AN0 to P67/ AN7	CMOS	CMOS	General I/O port	–	–	–	–	–	–	–	–	P67	P66	P65	P64	P63	P62	P61	P60
				Analog Input	–	–	–	–	–	–	–	–	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
Port 7	P70/DA0/ AN8 to P77/IN1/ AN15			General I/O port	P77	P76	P75	P74	P73	P72	P71	P70	–	–	–	–	–	–	–	–
				Resource	IN1	IN0	FRCK	SCK1	SOT1	SIN1*	–	–	–	–	–	–	–	–	–	–
		CMOS (hysteresis)	CMOS	Analog output	–	–	–	–	–	–	DA1	DA0	–	–	–	–	–	–	–	–
				Analog input	AN15	AN14	AN13	AN12	AN11	AN10	AN9	AN8	–	–	–	–	–	–	–	–
Port 8	P80/IN2 to P87/ RTO5			General I/O port	–	–	–	–	–	–	–	–	P87	P86	P85	P84	P83	P82	P81	P80
				Resource	–	–	–	–	–	–	–	–	RTO5	RTO4	RTO3	RTO2	RTO1	RTO0	IN3	IN2

\*: UART0, UART1 data input pins SIN0 and SIN1 can be selected as CMOS input by user program.

## Note :

Port 6, Port 7 are also used as analog signal input pins. To use the port as a general-purpose I/O port, be sure to set the corresponding bit of the analog input enable register (ADER0 and ADER1) to "0". Resetting the MCU sets the ADER0 / ADER1 register bits to "1".

## 9.2 Registers of I/O Ports

This section provides a list of the registers related to the I/O port settings.

### ■ Registers for I/O Ports

Table 9.2-1 is a list of the registers corresponding to individual port.

**Table 9.2-1 Registers of corresponding port**

Register name	Read/Write	Address	Initial value
Port 0 data register (PDR0)	R/W	000000 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 1 data register (PDR1)	R/W	000001 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 2 data register (PDR2)	R/W	000002 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 3 data register (PDR3)	R/W	000003 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 4 data register (PDR4)	R/W	000004 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 5 data register (PDR5)	R/W	000005 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 6 data register (PDR6)	R/W	000006 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 7 data register (PDR7)	R/W	000007 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 8 data register (PDR8)	R/W	000008 <sub>H</sub>	XXXXXXXX <sub>B</sub>
Port 0 direction register (DDR0)	R/W	000010 <sub>H</sub>	00000000 <sub>B</sub>
Port 1 direction register (DDR1)	R/W	000011 <sub>H</sub>	00000000 <sub>B</sub>
Port 2 direction register (DDR2)	R/W	000012 <sub>H</sub>	00000000 <sub>B</sub>
Port 3 direction register (DDR3)	R/W	000013 <sub>H</sub>	00000000 <sub>B</sub>
Port 4 direction register (DDR4)	R/W	000014 <sub>H</sub>	00000000 <sub>B</sub>
Port 5 direction register (DDR5)	R/W	000015 <sub>H</sub>	XXXXXX00 <sub>B</sub>
Port 6 direction register (DDR6)	R/W	000016 <sub>H</sub>	00000000 <sub>B</sub>
Port 7 direction register (DDR7)	R/W	000017 <sub>H</sub>	00000000 <sub>B</sub>
Port 8 direction register (DDR8)	R/W	000018 <sub>H</sub>	00000000 <sub>B</sub>
Analog input enable register 0 (ADER0)	R/W	0000C5 <sub>H</sub>	11111111 <sub>B</sub>
Analog input enable register 1 (ADER1)	R/W	0000D0 <sub>H</sub>	11111111 <sub>B</sub>
Port 0 pull-up resistor setting register (RDR0)	R/W	00008C <sub>H</sub>	00000000 <sub>B</sub>
Port 1 pull-up resistor setting register (RDR1)	R/W	00008D <sub>H</sub>	00000000 <sub>B</sub>
Port 2 pull-up resistor setting register (RDR2)	R/W	00008E <sub>H</sub>	00000000 <sub>B</sub>
Port 3 pull-up resistor setting register (RDR3)	R/W	00008F <sub>H</sub>	00000000 <sub>B</sub>

R/W :Readable and writable

X :Undefined

## MB90820B Series

### 9.3 Port 0

Port 0 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function, also provides the configuration of port 0, lists of pins, shows a block diagram of the pins, and describes the corresponding registers.

#### ■ Port 0 Configuration

Port 0 consists of the following:

- General-purpose I/O pins/resources I/O pins (P00 to P07/PWO0)
- Port 0 data register (PDR0)
- Port 0 direction register (DDR0)
- Port 0 pull-up resistor setting register (RDR0)

#### ■ Port 0 Pins

The port 0 I/O pins are also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port pins when they are used as resource I/O pins. Table 9.3-1 lists the port 0 pins.

**Table 9.3-1 Port 0 pins**

Port	Pin	Port function (single-chip mode)	Resource function	I/O form		Circuit type
				Input	Output	
Port 0	P00	P00	General-purpose I/O	—	—	CMOS (hysteresis)  CMOS  C
	P01	P01		—	—	
	P02	P02		—	—	
	P03	P03		—	—	
	P04	P04		—	—	
	P05	P05		—	—	
	P06/PWI0	P06		PWI0	PWC0 input	
	P07/PWO0	P07		PWO0	PWC0 output	

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## ■ Block Diagram of Port 0 Pins

Figure 9.3-1 shows the block diagram of the P00 to P06/PW10 pins.

**Figure 9.3-1 Block diagram of P00 to P06/PW10 pins**

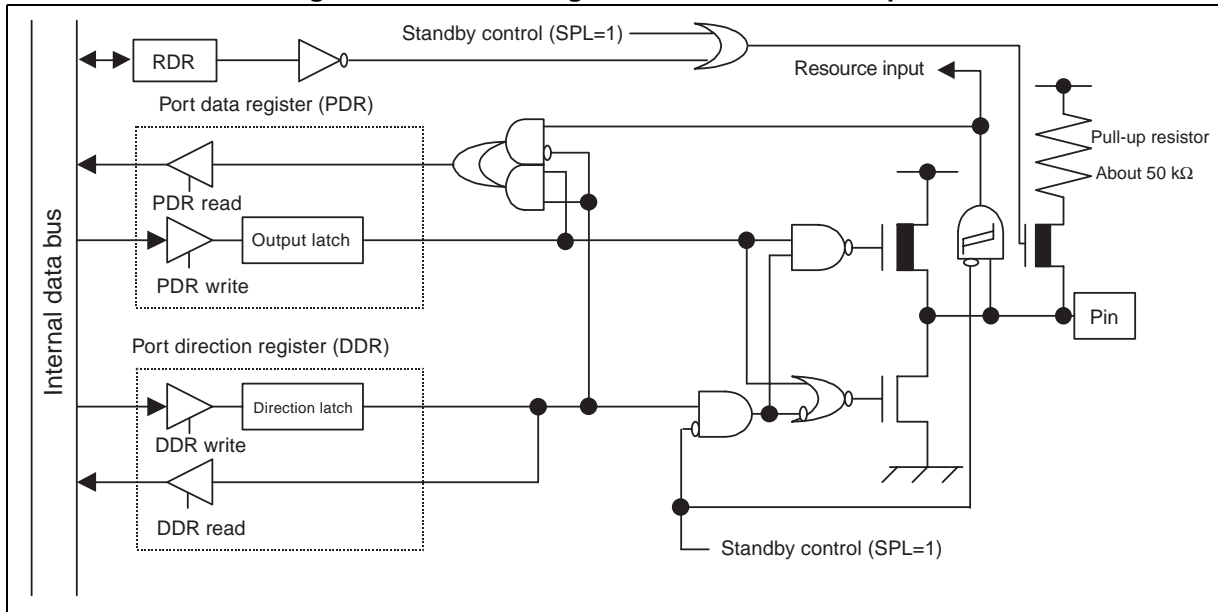
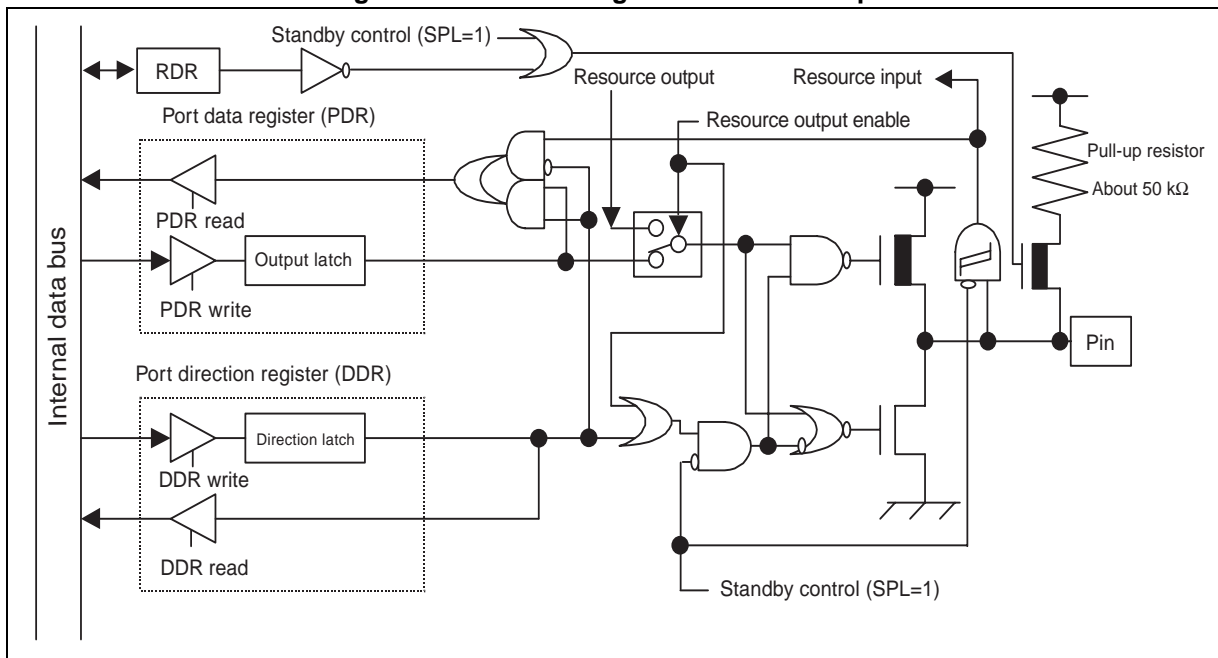


Figure 9.3-2 shows the block diagram of the P07/PWO0 pin.

**Figure 9.3-2 Block diagram of P07/PWO0 pin**



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR0 register.

#### ■ Port 0 Registers

Port 0 registers are PDR0, DDR0, and RDR0. The bits making up each register correspond to the port 0 pins on a one-to-one basis. Table 8.3-2 lists the port 0 pins and their corresponding register bits.

**Table 9.3-2 Port 0 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 0	PDR0, DDR0, RDR0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P07	P06	P05	P04	P03	P02	P01	P00

See Section "1.7 I/O Circuit Types", for information on the circuit types.

### 9.3.1 Port 0 Registers (PDR0, DDR0, and RDR0)

---

**This section describes the port 0 registers.**

---

#### ■ Functions of Port 0 Registers

- Port 0 data register (PDR0)

The PDR0 register indicates the state of each pin for port 0.

- Port 0 direction register (DDR0)

The DDR0 register specifies the direction of a data flow (input or output) at each pin (bit) of port 0. When a DDR0 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

Note :

- When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR0 register as long as the resource output enable bit corresponding to the pins is set to enable.
- To use a resource having input pins, set the port 0 direction register bit corresponding to each resource input pin to "0" to place the port in input mode.

Table 9.3-3 lists the functions of the port 0 registers.

---

Reference:

When the MCU is reset, the DDR0 register is cleared to "0" for general I/O port input.

---

- Port 0 pull-up resistor setting register (RDR0)

The RDR0 register specifies the selection of a pull-up resistor at each pin (bit) of port 0. When a RDR0 register bit is "1", a pull-up resistor is selected for the corresponding port (pin). When the bit is "0", the pull-up resistor is deselected.

Table 9.3-3 lists the functions of the port 0 registers.

**Table 9.3-3 Port 0 register functions**

Register name	Data	During reading	During writing	Read/Write	Address	Initial value
Port 0 data register (PDR0)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000000 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 0 direction register (DDR0)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000010 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
Port 0 pull-up resistor setting register (RDR0)	0	The setting latch is "0".	The pull-up resistor is cut and the port is placed in the Hi-Z state in input mode.	R/W	00008C <sub>H</sub>	00000000 <sub>B</sub>
	1	The setting latch is "1".	The pull-up resistor is selected and the port is held at the "H" level in input mode.			

R/W : Readable and writable

X : Undefined



## **9.3.2 Operation of Port 0**

---

**This section describes the operation of port 0.**

---

### **■ Operation of Port 0**

#### ● Port operation in output mode

- Setting a bit of the DDR0 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR0 register in output mode is held in the output latch of the PDR0 and output to the port pins.
- The PDR0 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR0).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port 0 data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write the output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the DDR0 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pin is in high-impedance state.
- However, when the corresponding bit in RDR0 register is set to "1" to select a pull-up resistor, the pin is held at the H level.
- Data written to the PDR0 register in input mode is held in the output latch of the PDR0 but is not output to the port pins.
- The PDR0 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

#### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR0 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

#### ● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the corresponding bit in DDR0 register to "0" to place the port in input mode.

#### ● Port operation after a reset

- When the MCU is reset, the DDR0 and RDR0 registers are initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input), the pull-up resistor is cut, and the pins are placed in a high-impedance state.
- The PDR0 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR0 register after the output data is set in the PDR0 register.

#### ● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR0 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.3-4 lists the states of the port 0 pins.

**Table 9.3-4 States of port 0 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1, RDR = 0)	Stop mode or time-base timer mode (SPL = 1, RDR = 1)
P00 to P07/ PWO0	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z	Input disabled/held at H level

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR)

Hi-Z : High impedance

## 9.4 Port 1

Port 1 is a general-purpose I/O port. It can also be used for resource input. Individual port pin can be switched between the general-purpose I/O port and resource input. This section focuses on the general I/O port function. The section also provides the configuration of port 1, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.

### ■ Port 1 Configuration

Port 1 consists of the following:

- General-purpose I/O pins/resources input pins (P10/INT0/DTTI to P17)
- Port 1 data register (PDR1)
- Port 1 direction register (DDR1)
- Port 1 pull-up resistor setting register (RDR1)

### ■ Port 1 Pins

The port 1 I/O pins are also used as resource input pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource input pins. Table 9.4-1 lists the port 1 pins.

**Table 9.4-1 Port 1 pins**

Port	Pin	Port function		Resource function		I/O form		Circuit type
						Input	Output	
Port 1	P10/INT0/ DTTI	P10	General- purpose I/O	INT0/ DTTI	External interrupt input / waveform generator input	CMOS (hysteresis)	CMOS	D
	P11/INT1	P11		INT1	External interrupt input			
	P12/INT2	P12		INT2	External interrupt input			
	P13/INT3	P13		INT3	External interrupt input			
	P14/INT4	P14		INT4	External interrupt input			
	P15/INT5	P15		INT5	External interrupt input			
	P16/INT6	P16		INT6	External interrupt input			
	P17	P17		—	—			

See Section "1.7 I/O Circuit Types", for information on the circuit types.

# MB90820B Series

## ■ Block Diagram of Port 1 Pins

Figure 9.4-1 shows the block diagram of P10/INT0/DTTI to P16/INT6 pins.

**Figure 9.4-1 Block diagram of P10/INT0/DTTI to P16/INT6 pins**

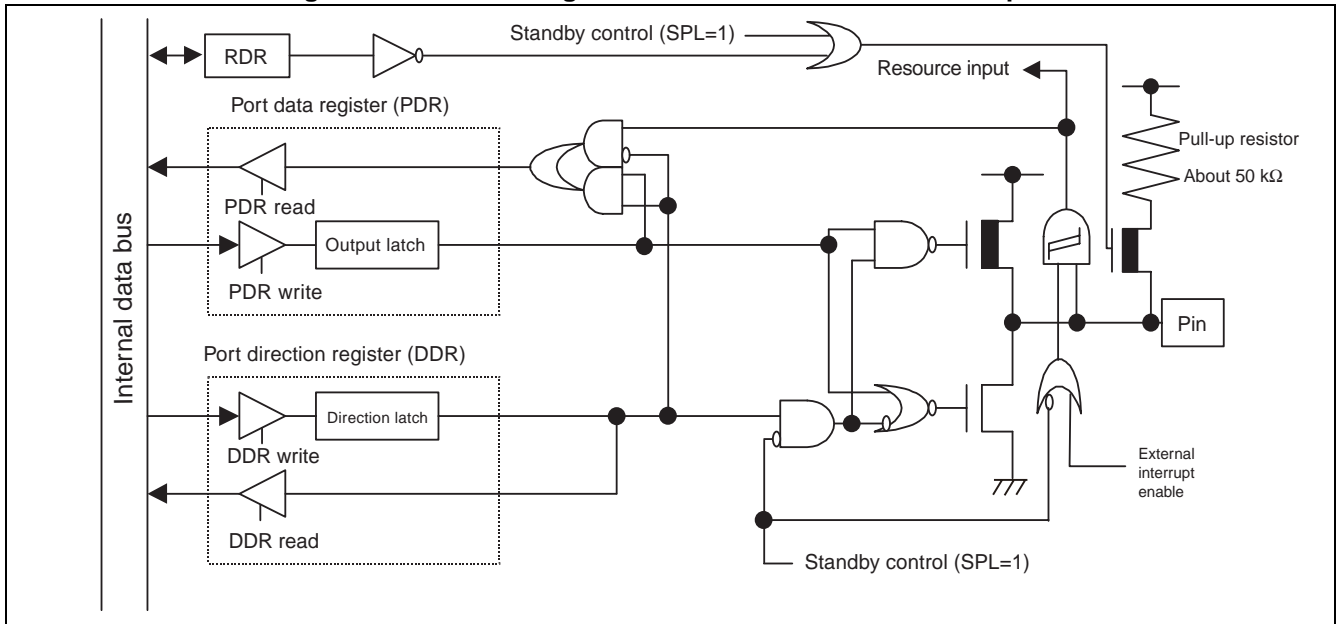
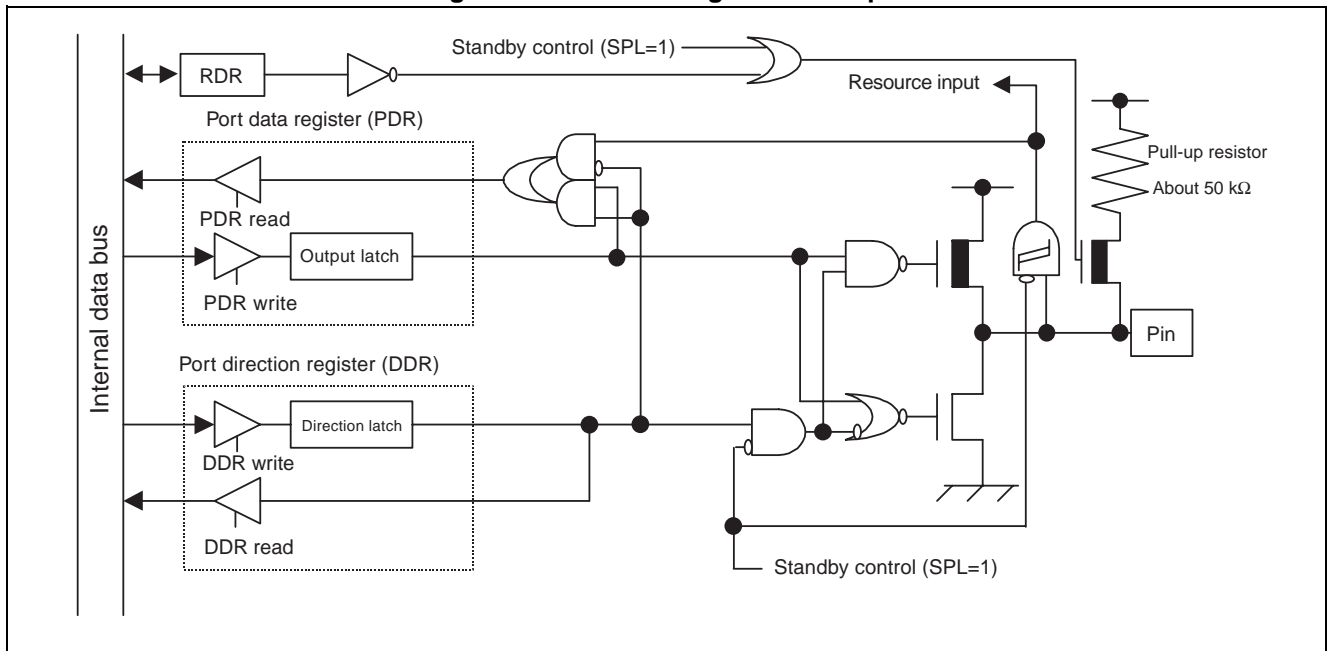


Figure 9.4-2 shows the block diagram of P17 pin.

**Figure 9.4-2 Block diagram of P17 pin**



■ Port 1 Registers

Port 1 registers are PDR1, DDR1, and RDR1. The bits making up each register correspond to the port 1 pins on a one-to-one basis. Table 9.4-2 lists the port 1 pins and their corresponding register bits.

Table 9.4-2 Port 1 pins and their corresponding register bits

Port	Register bits and corresponding port pins								
Port 1	PDR1, DDR1, RDR1	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
	Corresponding pin	P17	P16	P15	P14	P13	P12	P11	P10

## MB90820B Series

### 9.4.1 Port 1 Registers (PDR1, DDR1, and RDR1)

---

This section describes the port 1 registers.

---

#### ■ Functions of Port 1 Registers

- Port 1 data register (PDR1)

The PDR1 register indicates the state of each pin for port 1.

- Port 1 direction register (DDR1)

The DDR1 register specifies the direction of a data flow (input or output) at each pin (bit) of port 1. When a DDR1 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

---

Note :

To use a resource having input pins, reset the port direction register bit corresponding to each resource input pin to "0" to place the port in input mode.

---

Reference :

When the MCU is reset, the DDR1 register is cleared to "0" for general I/O port.

---

- Port 1 pull-up resistor setting register (RDR1)

The RDR1 register specifies the selection of a pull-up resistor at each pin (bit) of port 1. When a RDR1 register bit is "1", a pull-up resistor is selected for the corresponding port (pin). When the bit is "0", the pull-up resistor is deselected.

Table 9.4-3 lists the functions of the port 1 registers.

**Table 9.4-3 Port 1 register functions**

Register name	Data	During reading	During writing	Read/Write	Address	Initial value
Port 1 data register (PDR1)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000001 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 1 direction register (DDR1)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000011 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
Port 1 pull-up resistor setting register (RDR1)	0	The setting latch is "0".	The pull-up resistor is cut and the port is placed in the Hi-Z state in input mode.	R/W	00008D <sub>H</sub>	00000000 <sub>B</sub>
	1	The setting latch is "1".	The pull-up resistor is selected and the port is held at the "H" level in input mode.			

R/W : Readable and writable

X : Undefined

**MB90820B Series****9.4.2 Operation of Port 1**


---

**This section describes the operation of port 1.**

---

**■ Operation of Port 1**

● Port operation in output mode

- Setting a bit of the DDR1 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR1 register in output mode is held in the output latch of the PDR1 and output to the port pins.
- The PDR1 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR1).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write the output data to the PDR register, then specify output mode in the DDR register.

---

● Port operation in input mode

- Setting a bit of the DDR1 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pin is in high-impedance state.
- However, when the corresponding bit in RDR1 register is set to "1" to select a pull-up resistor, the pin is held at the H level.
- Data written to the PDR1 register in input mode is held in the output latch of the PDR1 but not output to the port pins.
- The PDR1 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the corresponding bit in DDR1 register to "0" to place the port in input mode.

● Port operation after a reset

- When the MCU is reset, the DDR1 registers is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input), and the pins are placed in a high-impedance state.
- The PDR1 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR1 register after the output data is set in the PDR1 register.



● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR1 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.4-4 lists the states of the port 1 pins.

**Table 9.4-4 States of port 1 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL =0)	Stop mode or time-base timer mode (SPL=1, RDR =0)	Stop mode or time-base timer mode (SPL=1, RDR =1)
P10/INT0/ DTTI to P16/INT6	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input enabled */ output in Hi-Z	Input enabled */ held at "H" level
P17	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/ output in Hi-Z	Input disabled/ held at "H" level

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance

\* : Only when P10/INT0/DTTI to P16/INT6 are configured as external interrupt pins, otherwise, input is disabled.

## MB90820B Series

### 9.5 Port 2

Port 2 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function. The section provides the configuration of port 2, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.

#### ■ Port 2 Configuration

Port 2 consists of the following:

- General-purpose I/O port/resource I/O pins (P20/TIN1 to P27)
- Port 2 data register (PDR2)
- Port 2 direction register (DDR2)
- Port 2 pull-up resistor setting register (RDR2)

#### ■ Port 2 Pins

The port 2 I/O pins are also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource I/O pins. Table 9.5-1 lists the port 2 pins.

**Table 9.5-1 Port 2 pins**

Port	Pin	Port function	Resource function	I/O form		Circuit type
				Input	Output	
Port 2	P20/TIN1	P20	TIN1	CMOS (hysteresis)	CMOS	D
	P21/TO1	P21	TO1			
	P22	P22	—			
	P23	P23	—			
	P24	P24	—			
	P25	P25	—			
	P26	P26	—			
	P27	P27	—			

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## ■ Block Diagram of Port 2 Pins

Figure 9.5-1 shows the block diagram of port 2 (excluding P21/TO1) pins.

**Figure 9.5-1 Block diagram of port 2 (excluding P21/TO1) pins**

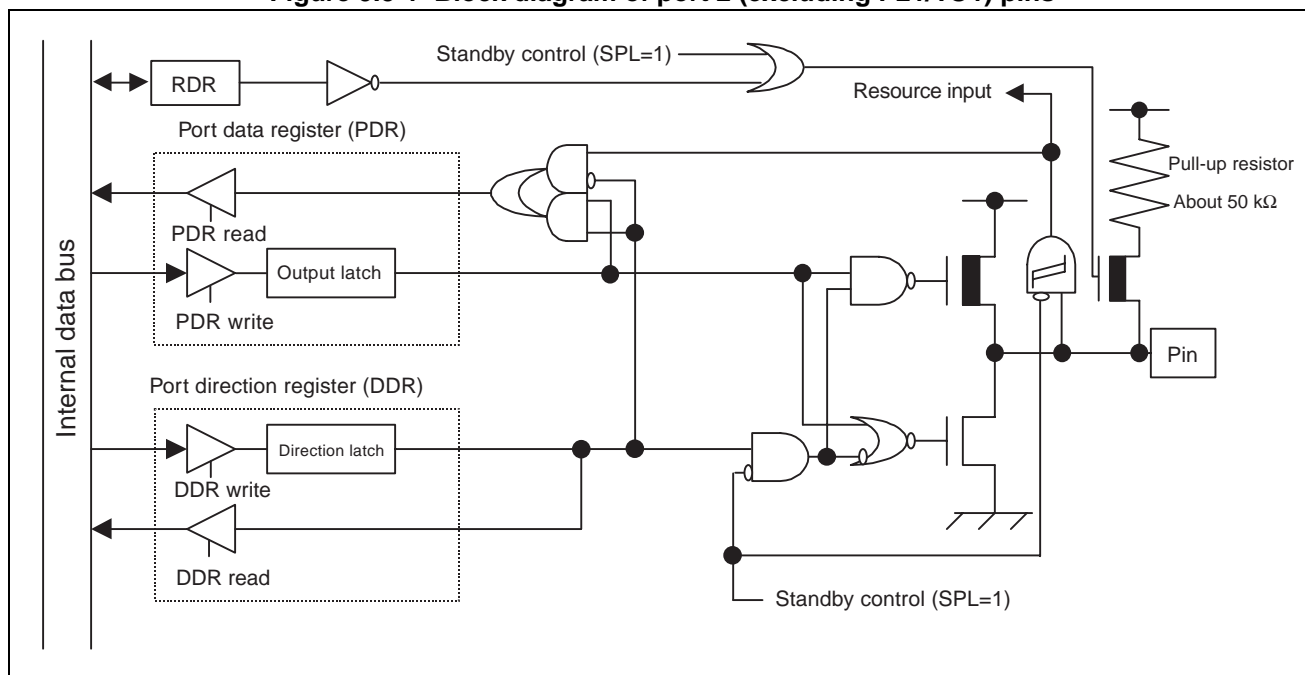
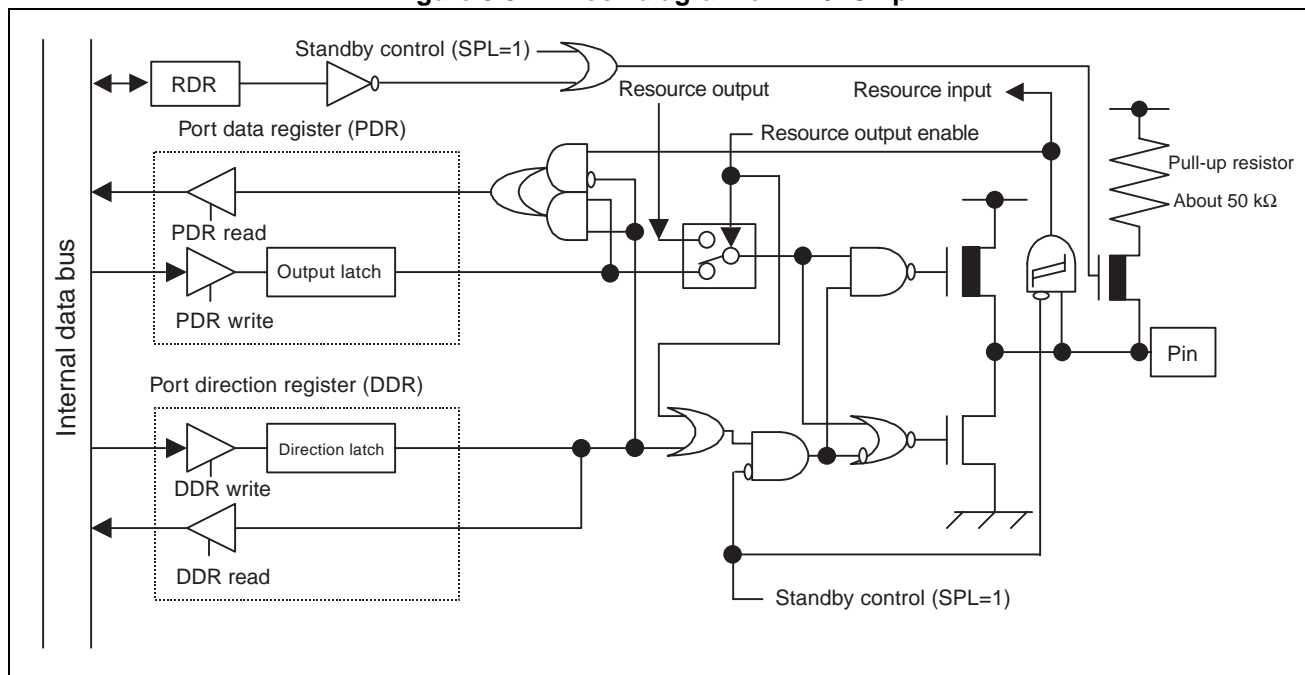


Figure 9.5-2 shows the block diagram of P21/TO1 pin.

**Figure 9.5-2 Block diagram of P21/TO1 pin**



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR2 register.

## ■ Port 2 Registers

Port 2 registers are PDR2, DDR2, and RDR2. The bits making up each register correspond to the port 2 pins on a one-to-one basis. Table 9.5-2 lists the port 2 pins and their corresponding register bits.

**Table 9.5-2 Port 2 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 2	PDR2, DDR2, RDR2	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P27	P26	P25	P24	P23	P22	P21	P20

## **9.5.1 Port 2 Registers (PDR2, DDR2, and RDR2)**

---

**This section describes the port 2 registers.**

---

### **■ Functions of Port 2 Registers**

- Port 2 data register (PDR2)

The PDR2 register indicates the state of each pin for port 2.

- Port 2 direction register (DDR2)

The DDR2 register specifies the direction of a data flow (input or output) at each pin (bit) of port 2. When a DDR2 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is 0, the port (pin) is set as an input port.

---

Notes:

- When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR2 register as long as the resource output enable bit corresponding to the pins is set.
  - To use a resource having input pins, set the port direction register bit corresponding to each resource input pin to "0" to place the port in input mode.
- 

- Port 2 pull-up resistor setting register (RDR2)

The RDR2 register specifies the selection of a pull-up resistor at each pin (bit) of port 2. When a RDR2 register bit is "1", a pull-up resistor is selected for the corresponding port (pin). When the bit is "0", the pull-up resistor is deselected.

Table 9.5-3 lists the functions of the port 2 registers.

**Table 9.5-3 Port 2 register functions**

Register name	Data	During reading	During writing	Read/Write	Address	Initial value
Port 2 data register (PDR2)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000002 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 2 direction register (DDR2)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000012 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
Port 2 pull-up resistor setting register (RDR2)	0	The setting latch is "0".	The pull-up resistor is cut and the port is placed in the Hi-Z state in input mode.	R/W	00008E <sub>H</sub>	00000000 <sub>B</sub>
	1	The setting latch is "1".	The pull-up resistor is selected and the port is held at the "H" level in input mode.			

R/W : Readable and writable

X : Undefined

## **9.5.2 Operation of Port 2**

---

**This section describes the operation of port 2.**

---

### **■ Operation of Port 2**

#### ● Port operation in output mode

- Setting a bit of the DDR2 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR2 register in output mode is held in the output latch of the PDR2 and output to the port pins.
- The PDR2 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR2).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write the output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the DDR2 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pin is in high-impedance state.
- However, when the corresponding bit in RDR2 register is set to "1" to select a pull-up resistor, the pin is held at the H level.
- Data written to the PDR2 register in input mode is held in the output latch of the PDR2 but not output to the port pins.
- The PDR2 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

#### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR2 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

#### ● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the corresponding bit in DDR2 register to "0" to place the port in input mode.

● Port operation after a reset

- When the MCU is reset, the DDR2 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input) and the pins are placed in a high-impedance state.
- The PDR2 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR2 register after the output data is set in the PDR2 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR2 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.5-4 lists the states of the port 2 pins.

**Table 9.5-4 States of port 2 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL =0)	Stop mode or time-base timer mode (SPL =1, RDR =0)	Stop mode or time-base timer mode (SPL =1, RDR =1)
P20/TIN1 to P27	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z	Input disabled/held at "H" level

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance



## 9.6 Port 3

**Port 3 is a general-purpose I/O port. It can also be used for resource output. Individual port pin can be switched between the I/O port and resource output. This section focuses on the general I/O port function. The section provides the configuration of port 3, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.**

### ■ Port 3 Configuration

Port 3 consists of the following:

- General-purpose I/O pins/resources output pin (P30 to P37/PPG0)
- Port 3 data register (PDR3)
- Port 3 direction register (DDR3)
- Port 3 pull-up resistor setting register (RDR3)

### ■ Port 3 pins

The port 3 is also used as resource output pin. Therefore, the pins cannot be used as general-purpose I/O port pins when they are used as resource output pin. Table 9.6-1 lists the port 3 pins.

**Table 9.6-1 Port 3 pins**

Port	Pin	Port function	Resource function	I/O form		Circuit type
				Input	Output	
Port 3	P30	P30	—	—	CMOS	CMOS
	P31	P31	—	—		
	P32	P32	—	—		
	P33	P33	—	—		
	P34	P34	—	—		
	P35	P35	—	—		
	P36	P36	—	—		
	P37/PPG0	P37	PPG0	PPG0 output		

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## ■ Block Diagram of Port 3 Pins

Figure 9.6-1 shows the block diagram of port 3 (excluding P37/PPG0) pins.

**Figure 9.6-1 Block diagram of port 3 (excluding P37/PPG0) pins**

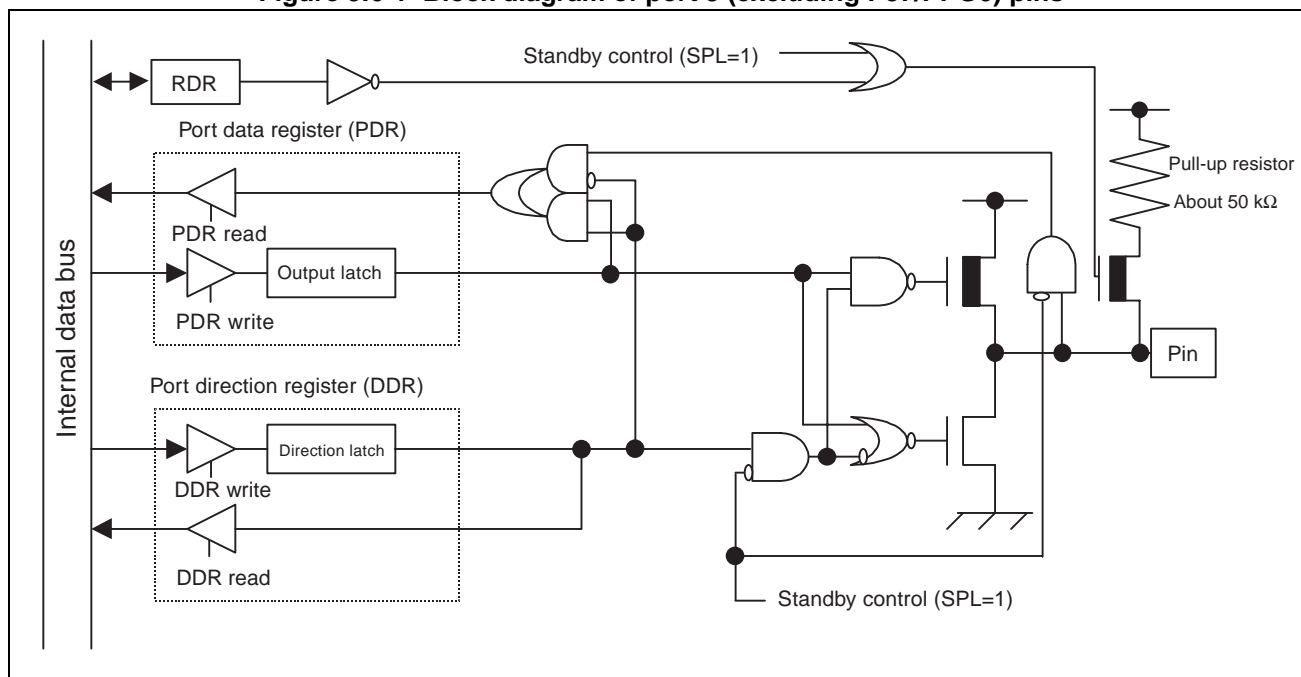
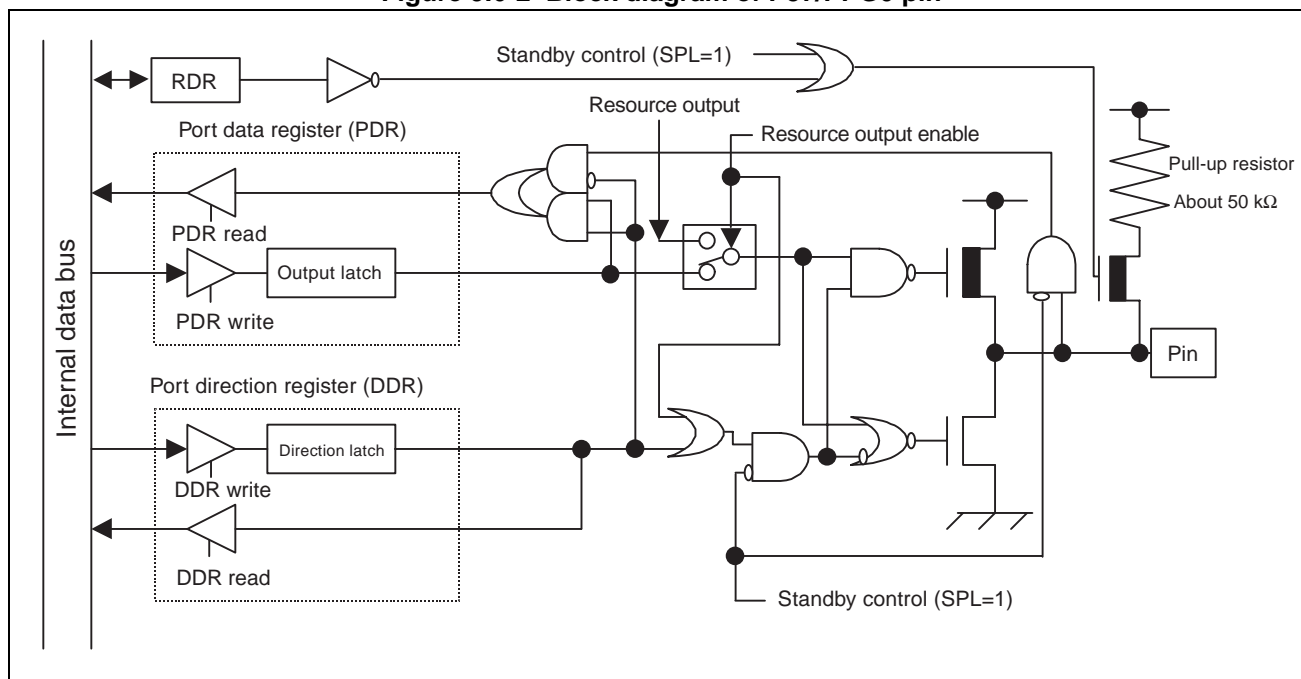


Figure 9.6-2 shows the block diagram of P37/PPG0 pin.

**Figure 9.6-2 Block diagram of P37/PPG0 pin**



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR3 register.

■ Port 3 Registers

Port 3 registers are PDR3, DDR3, and RDR3. The bits making up each register correspond to the port 3 pins on a one-to-one basis. Table 9.6-2 lists the port 3 pins and their corresponding register bits.

Table 9.6-2 Port 3 pins and their corresponding register bits

Port	Register bits and corresponding port pins								
Port 3	PDR3, DDR3, RDR3	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
	Corresponding pin	P37	P36	P35	P34	P33	P32	P31	P30

## MB90820B Series

### 9.6.1 Port 3 Registers (PDR3, DDR3, and RDR3)

---

**This section describes the port 3 registers.**

---

#### ■ Functions of Port 3 Registers

- Port 3 data register (PDR3)

The PDR3 register indicates the state of each pin for port 3.

- Port 3 direction register (DDR3)

The DDR3 register specifies the direction of a data flow (input or output) at each pin (bit) of port 3. When a DDR3 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

---

Note :

When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR3 register as long as the resource output enable bit corresponding to the pins is set.

---

- Port 3 pull-up resistor setting register (RDR3)

The RDR3 register specifies the selection of a pull-up resistor at each pin (bit) of port 3. When a RDR3 register bit is "1", a pull-up resistor is selected for the corresponding port (pin). When the bit is "0", the pull-up resistor is deselected. Table 9.6-3 lists the functions of the port 3 registers.

**Table 9.6-3 Port 3 register functions**

Register name	Data	During reading	During writing	Read /Write	Address	Initial value
Port 3 data register (PDR3)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000003 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 3 direction register (DDR3)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000013 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
Port 3 pull-up resistor setting register (RDR3)	0	The setting latch is "0".	The pull-up resistor is cut and the port is placed in the Hi-Z state in input mode.	R/W	00008F <sub>H</sub>	00000000 <sub>B</sub>
	1	The setting latch is "1".	The pull-up resistor is selected and the port is held at the "H" level in input mode.			

R/W : Readable and writable

X : Undefined

**MB90820B Series****9.6.2 Operation of Port 3**


---

**This section describes the operation of port 3.**

---

**■ Operation of Port 3**

● Port operation in output mode

- Setting a bit of the DDR3 register to “1” places the corresponding port pin in output mode.
- Data written to the PDR3 register in output mode is held in the output latch of the PDR3 and output to the port pins.
- The PDR3 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR3).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write the output data to the PDR register, then specify output mode in the DDR register.

---

● Port operation in input mode

- Setting a bit of the DDR3 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pin is in high-impedance state.
- However, when the corresponding bit in RDR3 register is set to “1” to select a pull-up resistor, the pin is held at the H level.
- Data written to the PDR3 register in input mode is held in the output latch of the PDR3 but not output to the port pins.
- The PDR3 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR3 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

● Port operation after a reset

- When the MCU is reset, the DDR3 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input) and the pins are placed in a high-impedance state.
- The PDR3 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR3 register after the output data is set in the PDR3 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR3 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.6-4 lists the states of the port 3 pins.

**Table 9.6-4 States of port 3 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL =0)	Stop mode or time-base timer mode (SPL =1, RDR =0)	Stop mode or time-base timer mode (SPL =1, RDR =1)
P30 to P37/PPG0	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z	Input disabled/held at "H" level

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance

# MB90820B Series

## 9.7 Port 4

Port 4 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function, also provides the configuration of port 4, lists of pins, shows a block diagram of the pins, and describes the corresponding registers.

### ■ Port 4 Configuration

Port 4 consists of the following:

- General-purpose I/O pins/resources I/O pins (P40/PPG1 to P47/PWO1)
- Port 4 data register (PDR4)
- Port 4 direction register (DDR4)

### ■ Port 4 Pins

The port 4 is also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource I/O pins. Table 9.7-1 lists the port 4 pins.

**Table 9.7-1 Port 4 pins**

Port	Pin	Port function	Resource function	I/O form		Circuit type
				Input	Output	
Port 4	P40/PPG1	P40	PPG1	CMOS (hysteresis)	CMOS	F
	P41/TIN0	P41	TIN0			
	P42/TO0	P42	TO0			
	P43/SCK0	P43	SCK0			
	P44/SOT0	P44	SOT0			G
	P45/SIN0	P45	SIN0			
	P46/PWI1	P46	PWI1			F
	P47/PWO1	P47	PWO1			

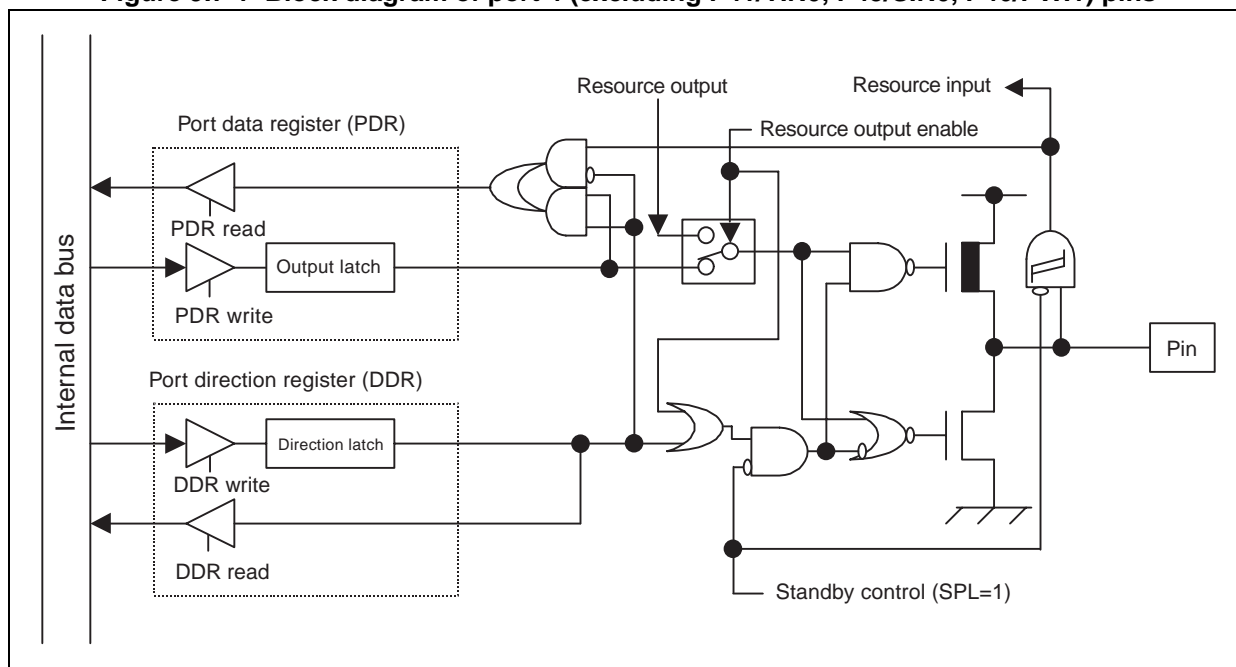
See Section "1.7 I/O Circuit Types", for information on the circuit types.



## ■ Block Diagram of Port 4 Pins

Figure 9.7-1 shows the block diagram of port 4 pins (excluding P41/TIN0, P45/SIN0, P46/PWI1).

**Figure 9.7-1 Block diagram of port 4 (excluding P41/TIN0, P45/SIN0, P46/PWI1) pins**



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR4 register.

Figure 9.7-2 shows the block diagram of P41/TIN0 and P46/PWI1 pins.

**Figure 9.7-2 Block diagram of P41/TIN0 and P46/PWI1 pins**

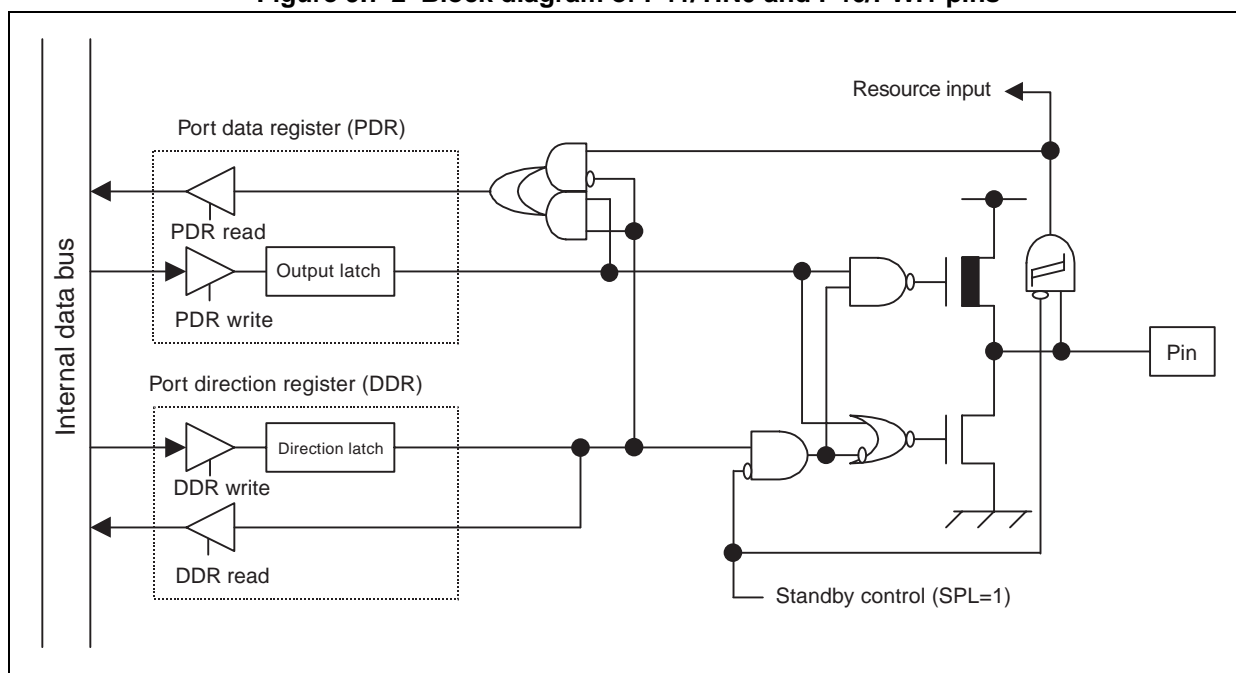
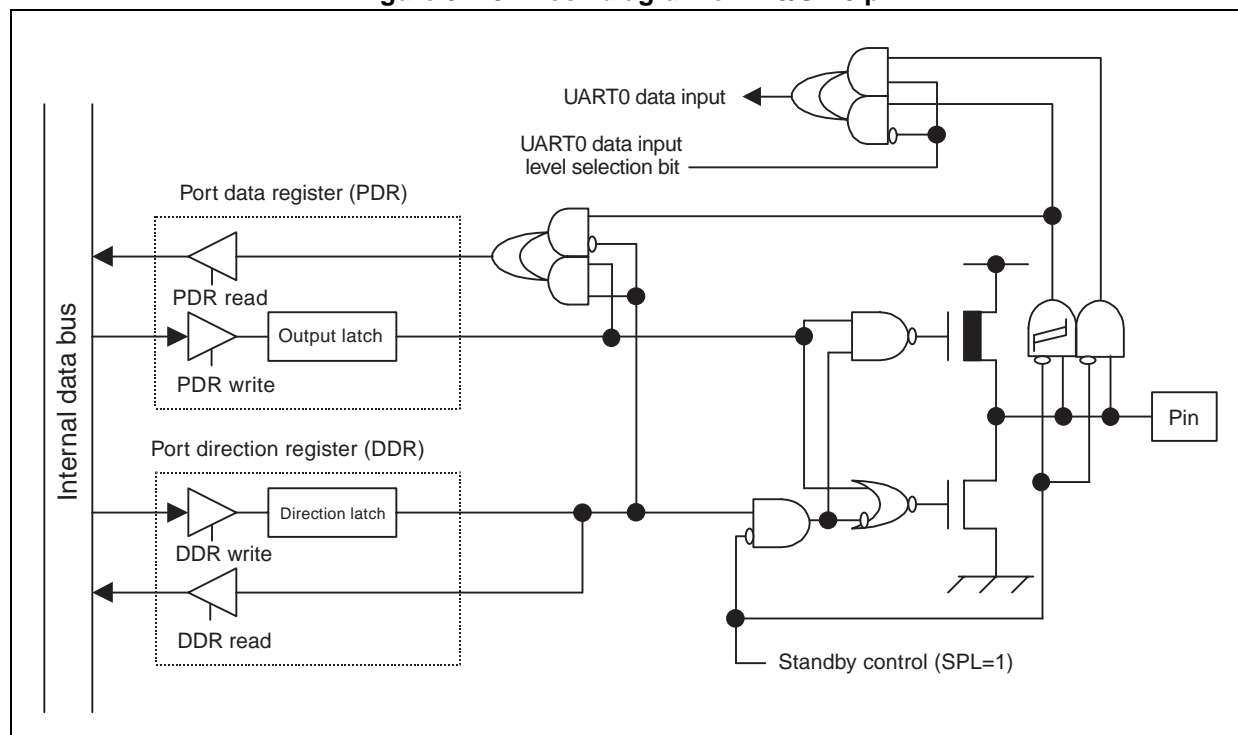


Figure 9.7-3 shows the block diagram of P45/SIN0 pin.

**Figure 9.7-3 Block diagram of P45/SIN0 pin**



## ■ Port 4 Registers

Port 4 registers are PDR4 and DDR4. The bits making up each register correspond to the port 4 pins on a one-to-one basis. Table 9.7-2 lists the port 4 pins and their corresponding register bits.

**Table 9.7-2 Port 4 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 4	PDR4, DDR4	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P47	P46	P45	P44	P43	P42	P41	P40

## 9.7.1 Port 4 Registers (PDR4 and DDR4)

This section describes the port 4 registers.

### ■ Functions of Port 4 registers

#### ● Port 4 data register (PDR4)

The PDR4 register indicates the state of each pin for port 4.

#### ● Port 4 direction register (DDR4)

The DDR4 register specifies the direction of a data flow (input or output) at each pin (bit) of port 4. When a DDR4 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

Notes:

- When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR4 register as long as the resource output enable bit corresponding to the pins is set.
- To use a resource having input pins, set the port direction register bit corresponding to each resource input pin to "0" to place the port in input mode.

Table 9.7-3 lists the functions of the port 4 register.

**Table 9.7-3 Port 4 register functions**

Register name	Data	During reading	During writing	Read /Write	Address	Initial value
Port 4 data register (PDR4)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000004 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 4 direction register (DDR4)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000014 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			

R/W : Readable and writable

X : Undefined

## MB90820B Series

### 9.7.2 Operation of Port 4

---

**This section describes the operation of port 4.**

---

#### ■ Operation of Port 4

##### ● Port operation in output mode

- Setting a bit of the DDR4 register to "1" places the corresponding port pin in output mode.
  - Data written to the PDR4 register in output mode is held in the output latch of the PDR4 and output to the port pins.
  - The PDR4 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR4).
- 

#### Note :

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write output data to the PDR register, then specify output mode in the DDR register.

---

##### ● Port operation in input mode

- Setting a bit of the DDR4 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pins are placed in a high-impedance state.
- Data written to the PDR4 register in input mode is held in the output latch of the PDR4 but not output to the port pins.
- The PDR4 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

##### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR4 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

##### ● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the corresponding bit in DDR4 register to "0" to place the port in input mode.

● Port operation after a reset

- When the MCU is reset, the DDR4 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input) and the pins are placed in a high-impedance state.
- The PDR4 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR4 register after the output data is set in the PDR4 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already 1 when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR4 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit.

Table 9.7-4 lists the states of the port 4 pins.

**Table 9.7-4 States of port 4 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1)
P40/PPG1 to P47/PWO1	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR)

Hi-Z : High impedance

## MB90820B Series

### 9.8 Port 5

Port 5 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function, also provides the configuration of port 5, lists of pins, shows a block diagram of the pins, and describes the corresponding registers.

#### ■ Port 5 Configuration

Port 5 consists of the following:

- General-purpose I/O pins/resources I/O pins (P50/PPG2 and P51/INT7)
- Port 5 data register (PDR5)
- Port 5 direction register (DDR5)

#### ■ Port 5 Pins

The port 5 is also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource I/O pins. Table 9.8-1 lists the port 5 pins.

**Table 9.8-1 Port 5 pins**

Port	Pin	Port function		Resource function		I/O form		Circuit type
						Input	Output	
Port 5	P50/PPG2	P50	General-purpose I/O	PPG2	PPG2 output	CMOS (hysteresis)	CMOS	F
	P51/INT7	P51		INT7	External interrupt input			

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## ■ Block Diagram of Port 5 Pins

Figure 9.8-1 shows the block diagram of P50/PPG2 pin.

Figure 9.8-1 Block diagram of P50/PPG2 pin

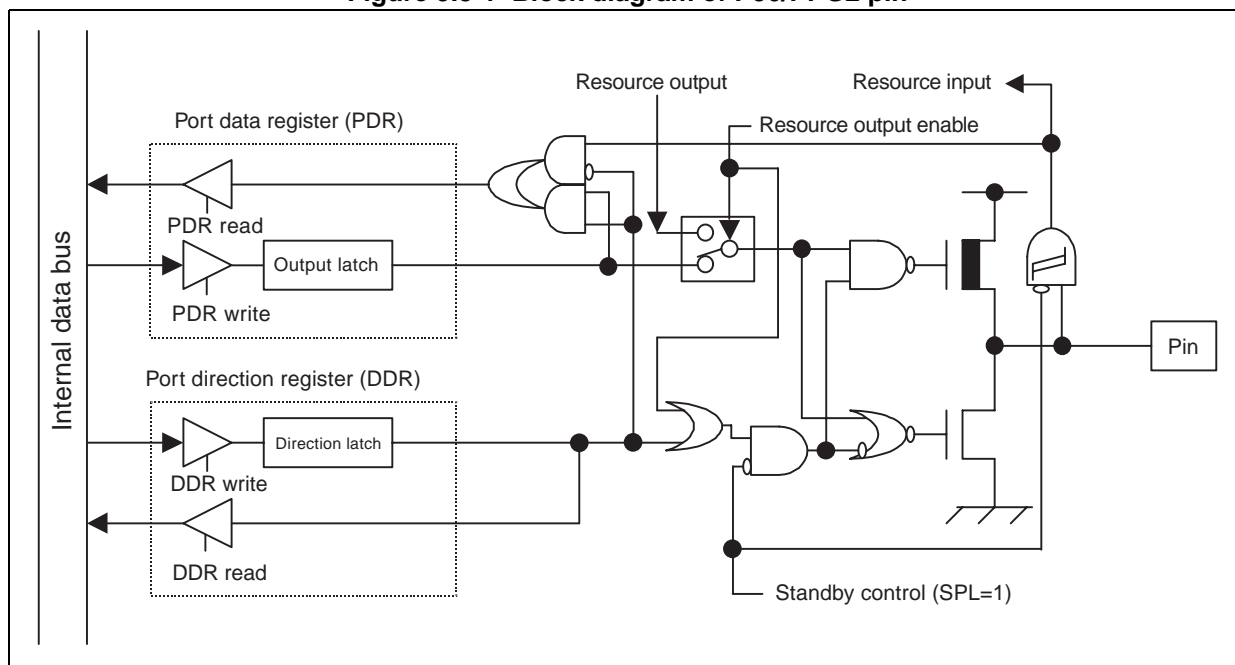
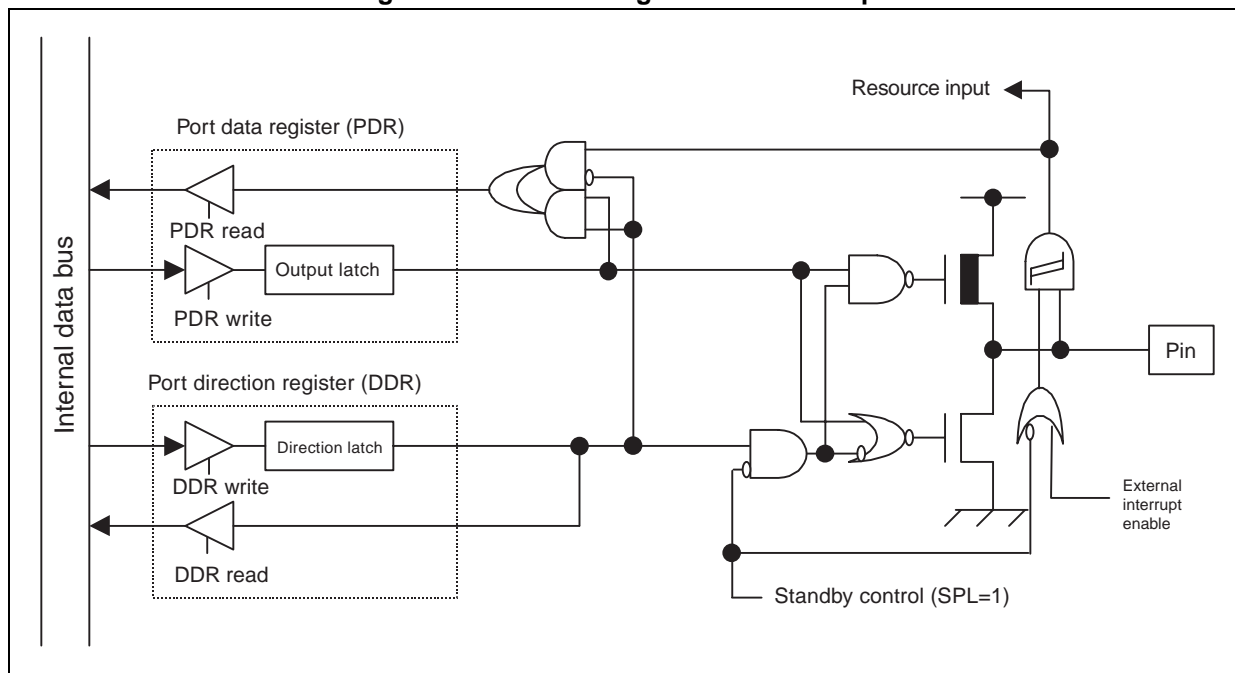


Figure 9.8-2 shows the block diagram of P51/INT7 pin.

Figure 9.8-2 Block diagram of P51/INT7 pin



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR5 register.

## ■ Port 5 Registers

Port 5 registers are PDR5 and DDR5. The bits making up each register correspond to the port 5 pins on a one-to-one basis. Table 9.8-2 lists the port 5 pins and their corresponding register bits.

**Table 9.8-2 Port 5 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 5	PDR5, DDR5	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
	Corresponding pin	-	-	-	-	-	-	P51	P50



## 9.8.1 Port 5 Registers (PDR5 and DDR5)

This section describes the port 5 registers.

### ■ Functions of Port 5 registers

#### ● Port 5 data register (PDR5)

The PDR5 register indicates the state of each pin for port 5.

#### ● Port 5 direction register (DDR5)

The DDR5 register specifies the direction of a data flow (input or output) at each pin (bit) of port 5. When a DDR5 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

Notes:

- When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR5 register as long as the resource output enable bit corresponding to the pins is set.
- To use a resource having input pins, set the port direction register bit corresponding to each resource input pin to "0" to place the port in input mode.

Table 9.8-3 lists the functions of the port 5 registers.

**Table 9.8-3 Port 5 register functions**

Register name	Data	During reading	During writing	Read /Write	Address	Initial value
Port 5 data register (PDR5)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000005 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 5 direction register (DDR5)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000015 <sub>H</sub>	XXXXXX00 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			

R/W : Readable and writable

X : Undefined

## 9.8.2 Operation of Port 5

---

**This section describes the operation of port 5.**

---

### ■ Operation of Port 5

#### ● Port operation in output mode

- Setting a bit of the DDR5 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR5 register in output mode is held in the output latch of the PDR5 and output to the port pins.
- The PDR5 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR5).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the DDR5 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pins are placed in a high-impedance state.
- Data written to the PDR5 register in input mode is held in the output latch of the PDR5 but not output to the port pins.
- The PDR5 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

#### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR5 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

#### ● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the corresponding bit in DDR5 register to "0" to place the port in input mode.

#### ● Port operation after a reset

- When the MCU is reset, the DDR5 register is initialized to "0". As a result, the output buffer is turned

off (I/O mode changes to input), and the pins are placed in a high-impedance state.

- The PDR5 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR5 register after the output data is set in the PDR5 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR5 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.8-4 lists the states of the port 5 pins.

**Table 9.8-4 States of port 5 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1)
P50/PPG2	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z
P51/INT7	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input enabled*/output in Hi-Z

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance

\* : Only when P51/INT7 are configured as external interrupt pins, otherwise, input is disabled.

# MB90820B Series

## 9.9 Port 6

Port 6 is a general-purpose I/O port. It can also be used for A/D converter analog input. Individual port pin can be switched between the I/O port and A/D converter analog input. This section focuses on the general I/O port function. It provides the configuration of port 6, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.

### ■ Port 6 Configuration

Port 6 consists of the following:

- General-purpose I/O port/analog input pins (P60/AN0 to P67/AN7)
- Port 6 data register (PDR6)
- Port 6 direction register (DDR6)
- Analog input enable register 0 (ADER0)

### ■ Port 6 Pins

The port 6 I/O is also used as analog input pins. Therefore, the pins cannot be used as general-purpose I/O port pins when they are used as analog input pins. Table 9.9-1 lists the port 6 pins.

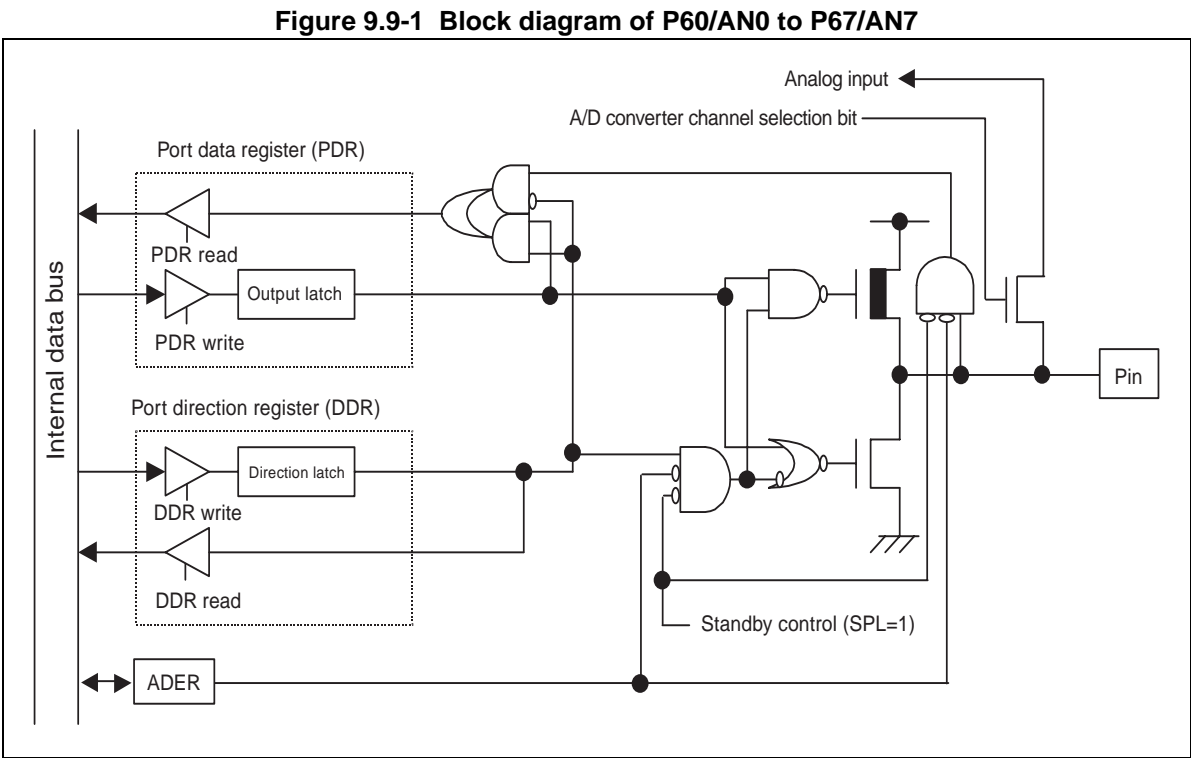
**Table 9.9-1 Port 6 pins**

Port	Pin	Port function		Resource function		I/O form		Circuit type
						Input	Output	
Port 6	P60/AN0	P60	General-purpose I/O	AN0	Analog input 0	Analog/CMOS	CMOS	H
	P61/AN1	P61		AN1	Analog input 1			
	P62/AN2	P62		AN2	Analog input 2			
	P63/AN3	P63		AN3	Analog input 3			
	P64/AN4	P64		AN4	Analog input 4			
	P65/AN5	P65		AN5	Analog input 5			
	P66/AN6	P66		AN6	Analog input 6			
	P67/AN7	P67		AN7	Analog input 7			

See Section "1.7 I/O Circuit Types", for information on the circuit types.

■ Block Diagram of Port 6 Pins

Figure 9.9-1 shows the block diagram of P60/AN0 to P67/AN7.



When the analog input enable bit is set to enable, the port is forcibly caused to function as A/D converter input pin regardless of the value in the DDR6 register.

■ Port 6 Registers

Port 6 registers are PDR6, DDR6, and ADER0. The bits making up PDR6, DDR6, and ADER0 registers correspond to the port 6 pin on one-to-one basis. Table 9.9-2 lists the port 6 pins and their corresponding register bits.

**Table 9.9-2 Port 6 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 6	PDR6, DDR6, ADER0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P67	P66	P65	P64	P63	P62	P61	P60

## MB90820B Series

### 9.9.1 Port 6 Registers (PDR6, DDR6, and ADER0)

---

**This section describes the port 6 registers.**

---

#### ■ Functions of Port 6 Registers

- Port 6 data register (PDR6)

The PDR6 register indicates the state of each pin for port 6.

- Port 6 direction register (DDR6)

The DDR6 register specifies the direction of a data flow (input or output) at each pin (bit) of port 6. When a DDR6 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is 0, the port (pin) is set as an input port.

---

**Notes:**

- When A/D input enable bit is set, the corresponding port functions as A/D converter input pins regardless of the value in the DDR6 register.
- To use as general-purpose I/O port, set the corresponding A/D converter input enable register bit to "0" to place the port in general-purpose I/O mode.

- Analog input enable register 0 (ADER0)

Each bit of the ADER0 register specifies whether the corresponding port 6 pin is to be used as a general-purpose I/O port or an analog input pin. Setting an ADE bit to "1" enables the corresponding pin for analog input. Setting the bit to 0 enables the pin for general-purpose I/O.

---

**Note :**

If a signal at an intermediate level is input in port I/O mode, input leak current flows. Therefore, for a pin used for analog input, be sure to set the corresponding ADE bits to "1".

---

**Reference:**

When the MCU is reset, the DDR6 register is cleared to "0" and the ADER0 register is set to "1" (used as the analog input).

---

Table 9.9-3 lists the functions of the port 6 registers.

**Table 9.9-3 Port 6 register functions**

Register	Data	During reading	During writing	Read /Write	Address	Initial value
Port 6 data register (PDR6)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000006 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 6 direction register (DDR6)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000016 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
A/D input enable register 0 (ADER0)	0	Port I/O mode		R/W	0000C5 <sub>H</sub>	11111111 <sub>B</sub>
	1	Analog input mode				

R/W : Readable and writable

X : Undefined

## 9.9.2 Operation of Port 6

---

**This section describes the operation of port 6.**

---

### ■ Operation of Port 6

#### ● Port operation in output mode

- Setting a bit of the ADER0 register to "0" places the corresponding port pin in port I/O mode.
- Setting a bit of the DDR6 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR6 register in output mode is held in the output latch of the PDR6 and output to the port pins.
- The PDR6 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR6).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the ADER0 register to "0" places the corresponding port pin in port I/O mode.
- Setting a bit of the DDR6 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pins are placed in a high-impedance state.
- Data written to the PDR6 register in input mode is held in the output latch of the PDR6 but not output to the port pins.
- The PDR6 register can be accessed in read mode to read the level value (0 or 1) at the port pins.

#### ● Port operation for analog input

To use a port pin for analog input, write "1" to the corresponding ADE bit. Doing so disables the pin from operating as a general-purpose port pin and enables it to function as an analog input pin. When PDR6 register is accessed in read mode in this situation, a value of "0" is read.

#### ● Port operation after a reset

- When the MCU is reset, the DDR6 register is initialized to "0" and the ADER0 register is initialized to "1" to place the port in analog input mode. To use the port as a general-purpose port, write "0" to the ADER0 register in advance to place the port in port I/O mode.
- When the MCU is reset, the DDR6 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input), and the pins are placed in a high-impedance state.
- The PDR6 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR6 register after the output data is set in the PDR6 register.



● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR6 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit.

Table 9.9-4 lists the states of the port 6 pins.

**Table 9.9-4 States of port 6 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1)
P60/AN0 to P67/AN7	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance

# MB90820B Series

## 9.10 Port 7

**Port 7 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function. It provides the configuration of port 7, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.**

### ■ Port 7 Configuration

Port 7 consists of the following:

- General-purpose I/O pins/resource I/O pins (P70/DA0/AN8 to P77/IN1/AN15)
- Port 7 data register (PDR7)
- Port 7 direction register (DDR7)
- Analog input enable register 1 (ADER1)

### ■ Port 7 Pins

The port 7 I/O is also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource I/O pins. Table 9.10-1 lists the port 7 pins.

**Table 9.10-1 Port 7 pins**

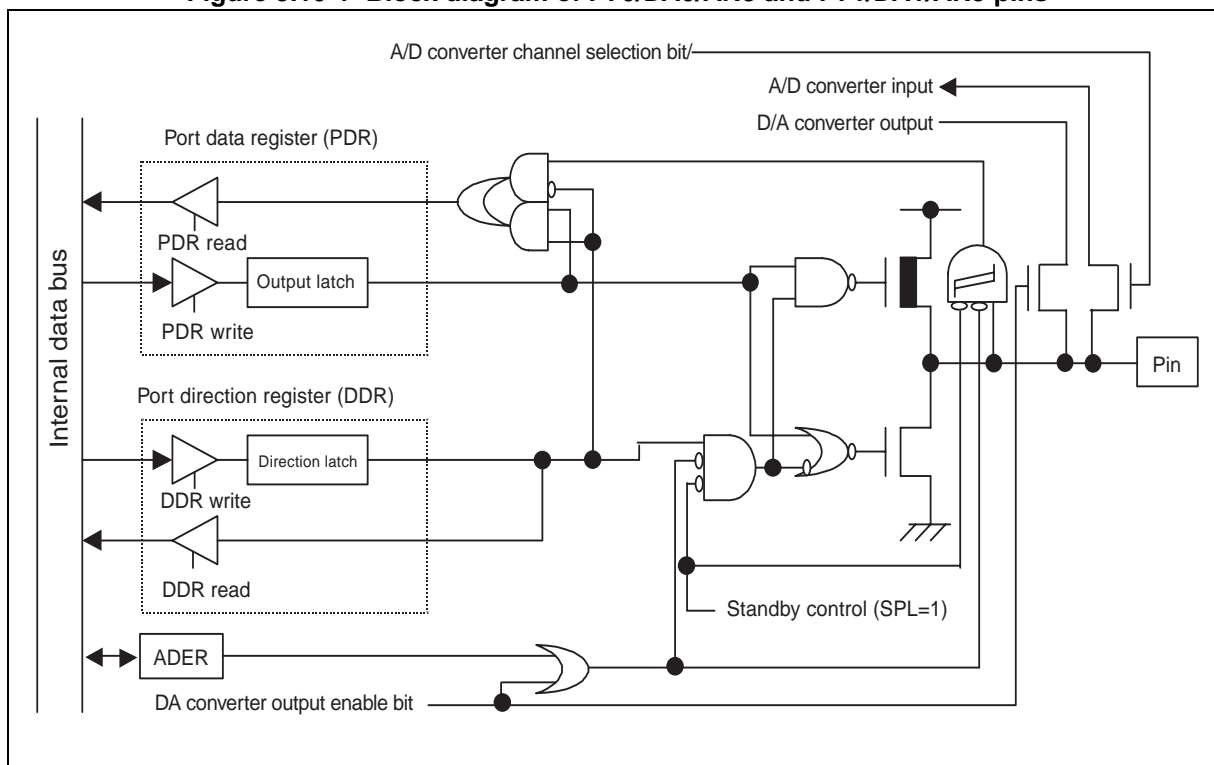
Port	Pin	Port function	Resource function	I/O form		Circuit type
				Input	Output	
Port 7	P70/DA0/ AN8	P70	DA0/ AN8	D/A converter output 0 / A/D converter channel 8 input	CMOS (hysteresis)/ Analog	I
	P71/DA1/ AN9	P71	DA1/ AN9	D/A converter output 1 / A/D converter channel 9 input		
	P72/SIN1/ AN10	P72	SIN1/ AN10	UART1 data input / A/D converter channel 10 input		J
	P73/SOT1/ AN11	P73	SOT1/ AN11	UART1 data output / A/D converter channel 11 input		K
	P74/SCK1/ AN12	P74	SCK1/ AN12	UART1 serial clock input / A/D converter channel 12 input		
	P75/ FRCK/ AN13	P75	FRCK/ AN13	Free-run timer clock input / A/D converter channel 13 input		
	P76/IN0/ AN14	P76	IN0/ AN14	Input capture channel 0 input / A/D converter channel 14 input		
	P77/IN1/ AN15	P77	IN1/ AN15	Input capture channel 1 input / A/D converter channel 15 input		

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## ■ Block Diagram of Port 7 Pins

Figure 9.10-1 shows the block diagram of P70/DA0/AN8 and P71/DA1/AN9 pins.

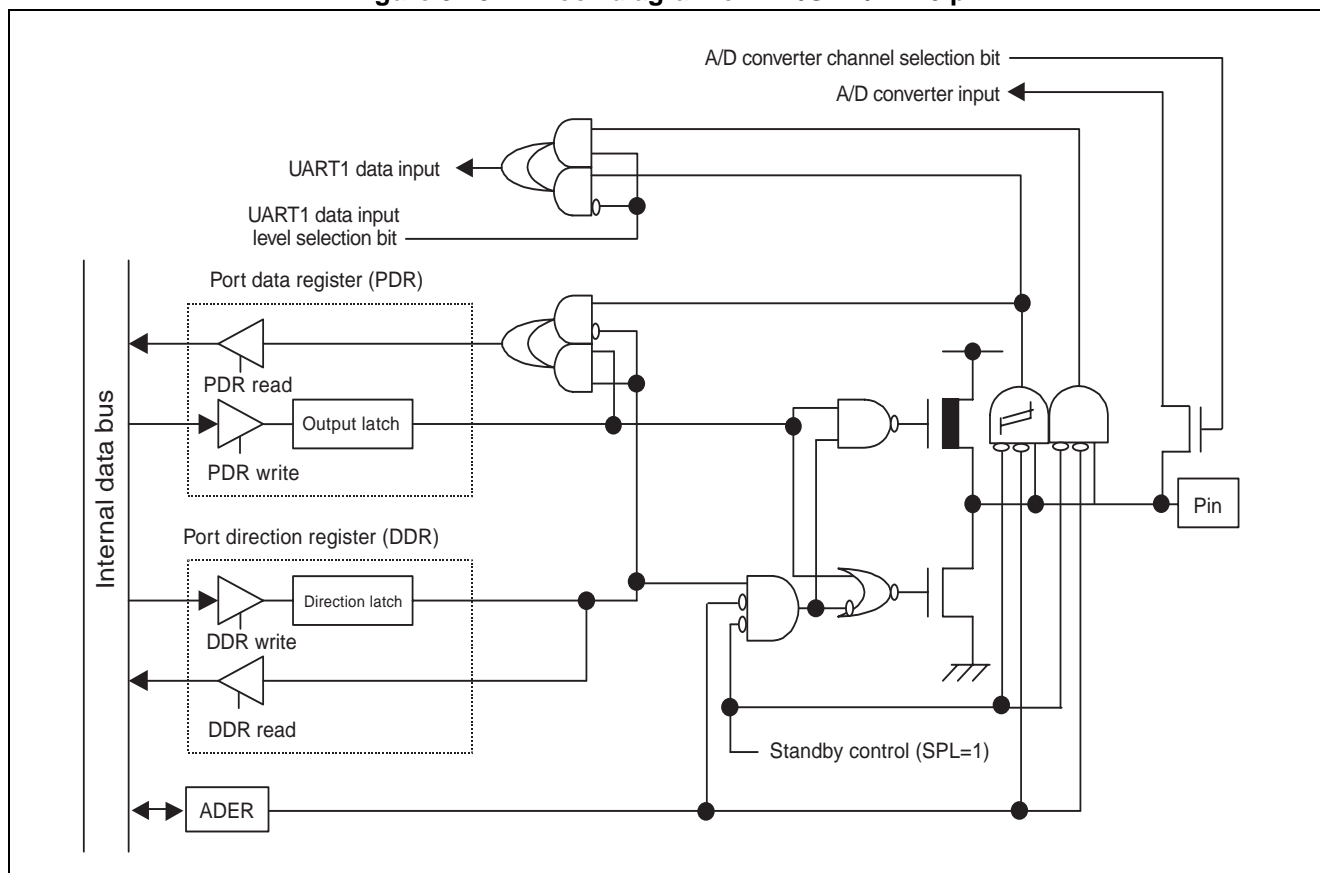
**Figure 9.10-1 Block diagram of P70/DA0/AN8 and P71/DA1/AN9 pins**



When the D/A converter output enable bit or analog input enable bit is set to enable, the port is forcibly caused to function as D/A converter output pin or A/D converter input pin regardless of the value in the DDR7 register.

Figure 9.10-2 shows the block diagram of P72/SIN1/AN10 pin.

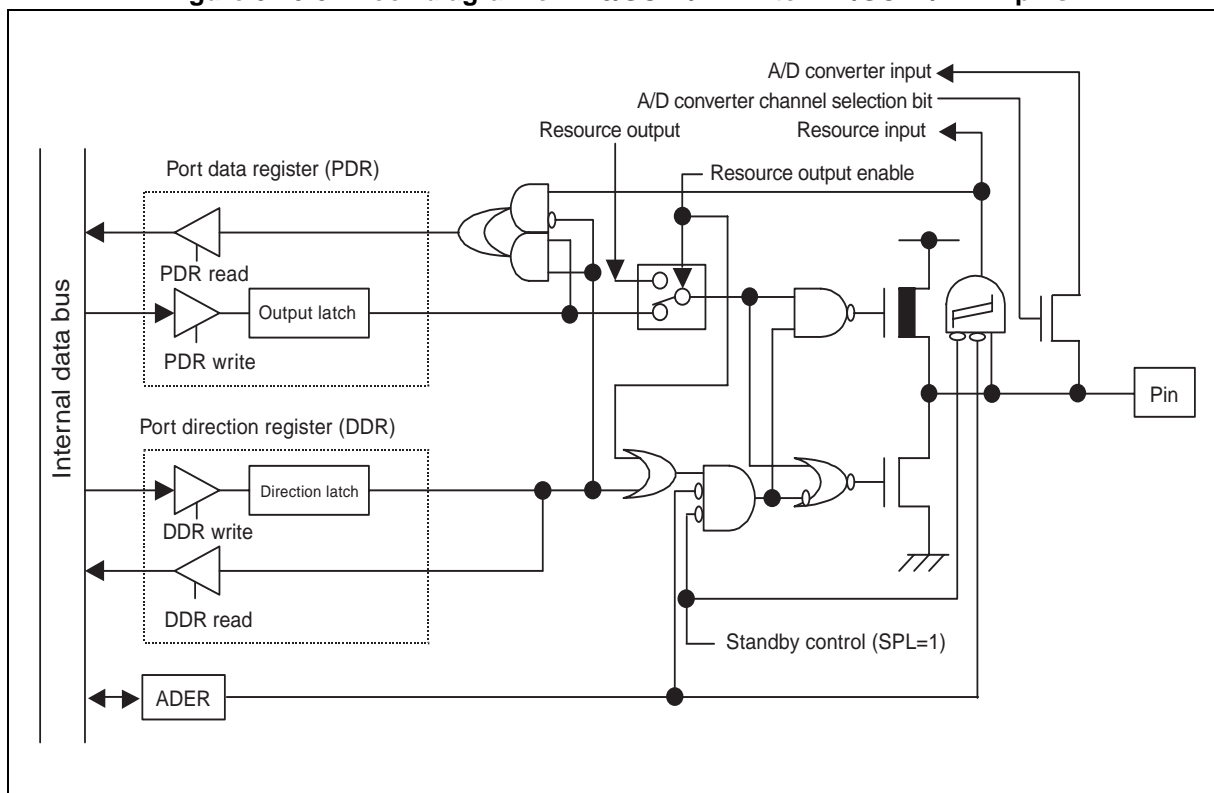
**Figure 9.10-2 Block diagram of P72/SIN1/AN10 pin**



When analog input enable bit is set to enable, the port is forcibly caused to function as A/D converter input pin regardless of the value in the DDR7 register.

Figure 9.10-3 shows the block diagram of port 7 (P73/SOT1/AN11 to P74/SCK1/AN12) pins.

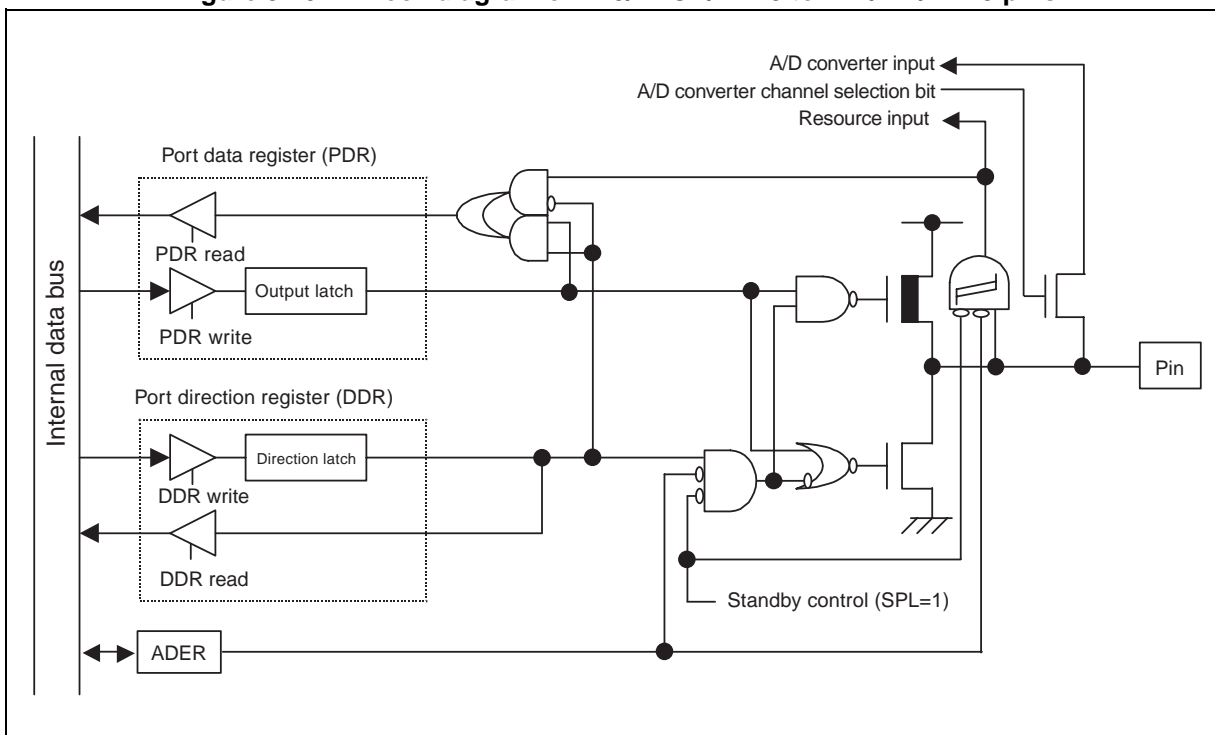
**Figure 9.10-3 Block diagram of P73/SOT1/AN11 to P74/SCK1/AN12 pins**



When the resource output enable bit is set to enable, the port is forcibly caused to function as resource output pin regardless of the value in the DDR7 register.

Figure 9.10-4 shows the block diagram of P75/FRCK/AN13 to P77/IN1/AN15 pins.

**Figure 9.10-4 Block diagram of P75/FRCK/AN13 to P77/IN1/AN15 pins**



## ■ Port 7 Registers

Port 7 registers are PDR7, DDR7, and ADER1. The bits making up each register correspond to the port 7 pins on a one-to-one basis. Table 9.10-2 lists the port 7 pins and their corresponding register bits.

**Table 9.10-2 Port 7 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 7	PDR7, DDR7	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
	ADER1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P77	P76	P75	P74	P73	P72	P71	P70

## **9.10.1 Port 7 Registers (PDR7, DDR7, and ADER1)**

---

**This section describes the port 7 registers.**

---

### **■ Functions of Port 7 Registers**

- Port 7 data register (PDR7)

The PDR7 register indicates the state of each pin for port 7.

- Port 7 direction register (DDR7)

The DDR7 register specifies the direction of a data flow (input or output) at each pin (bit) of port 7. When a DDR7 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

---

Notes:

- When D/A converter output enable bit or A/D input enable bit is set, the corresponding port functions as D/A converter output pin or A/D converter input pin regardless of the value in the DDR7 register.
  - To use as general-purpose I/O port, set the corresponding A/D converter input enable register bit to "0" to place the port in general-purpose I/O mode.
- 

- Analog input enable register 1 (ADER1)

Each bit of the ADER1 register specifies whether the corresponding port 7 pin is to be used as a general-purpose I/O port or an analog input pin. Setting an ADE bit to "1" enables the corresponding pin for analog input. Setting the bit to "0" enables the pin for general-purpose I/O.

---

Note:

If a signal at an intermediate level is input in port I/O mode, input leak current flows. Therefore, for a pin used for analog input, be sure to set the corresponding ADE bits to "1".

---

Reference:

When the MCU is reset, the DDR7 register is cleared to "0" and the ADER1 register is set to "1" (used as the analog input).

---

Table 9.10-3 lists the functions of the port 7 registers.

**Table 9.10-3 Port 7 register functions**

Register	Data	During reading	During writing	Read /Write	Address	Initial value
Port 7 data register (PDR7)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000007 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output latch is loaded with "1". When the pin functions as an output port, the pin is set to the "H" level.			
Port 7 direction register (DDR7)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000017 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned on to place the port in output mode.			
A/D input enable register 1 (ADER1)	0	Port I/O mode		R/W	0000D0 <sub>H</sub>	11111111 <sub>B</sub>
	1	Analog input mode				

R/W : Readable and writable

X : Undefined



## **9.10.2 Operation of Port 7**

---

**This section describes the operation of port 7.**

---

### **■ Operation of Port 7**

#### ● Port operation in output mode

- Setting a bit of the ADER1 register to "0" places the corresponding port pin in port I/O mode.
- Setting a bit of the DDR7 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR7 register in output mode is held in the output latch of the PDR7 and output to the port pins.
- The PDR7 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR7).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the ADER1 register to "0" places the corresponding port pin in port I/O mode.
- Setting a bit of the DDR7 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pins are placed in a high-impedance state.
- Data written to the PDR7 register in input mode is held in the output latch of the PDR7 but not output to the port pins.
- The PDR7 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

#### ● Port operation for analog input

To use a port pin for analog input, write "1" to the corresponding ADE bit. Doing so disables the pin from operating as a general-purpose port pin and enables it to function as an analog input pin. When PDR7 register is accessed in read mode in this situation, a value of "0" is read.

#### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR7 register bit is "0", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the DDR7 register to "0" to place the port in input mode.

● Port operation after a reset

- When the MCU is reset, the DDR7 register is initialized to "0" and the ADER1 register is initialized to "1" to place the port in analog input mode. To use the port as a general-purpose port, write "0" to the ADER1 register in advance to place the port in port I/O mode.
- When the MCU is reset, the DDR7 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input), and the pins are placed in a high-impedance state.
- The PDR7 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR7 register after the output data is set in the PDR7 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the DDR7 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.10-4 lists the states of the port 7 pins.

**Table 9.10-4 States of port 7 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1)
P70/DA0/AN8 to P77/IN1/AN15	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR: SPL)

Hi-Z : High impedance

## 9.11 Port 8

**Port 8 is a general-purpose I/O port. It can also be used for resource I/O. Individual port pin can be switched between the I/O port and resource I/O. This section focuses on the general I/O port function. This section also provides the configuration of port 8, lists its pins, shows a block diagram of the pins, and describes the corresponding registers.**

### ■ Port 8 Configuration

Port 8 consists of the following:

- General-purpose I/O pins/resource I/O pins (P80/IN2 to P87/RTO5)
- Port 8 data register (PDR8)
- Port 8 direction register (DDR8)

### ■ Port 8 Pins

The port 8 I/O is also used as resource I/O pins. Therefore, the pins cannot be used as general-purpose I/O port when they are used as resource I/O pins. Table 9.11-1 lists the port 8 pins.

**Table 9.11-1 Port 8 pins**

Port	Pin	Port function		Resource function		I/O form		Circuit type
						Input	Output	
Port 8	P80/IN2	P80	General-purpose I/O	IN2	Input capture channel 2	CMOS (hysteresis)	CMOS	F
	P81/IN3	P81		IN3	Input capture channel 3			
	P82/RTO0	P82		RTO0	Waveform generator output 0			L
	P83/RTO1	P83		RTO1	Waveform generator output 1			
	P84/RTO2	P84		RTO2	Waveform generator output 2			
	P85/RTO3	P85		RTO3	Waveform generator output 3			
	P86/RTO4	P86		RTO4	Waveform generator output 4			
	P87/RTO5	P87		RTO5	Waveform generator output 5			

See Section "1.7 I/O Circuit Types", for information on the circuit types.

## MB90820B Series

### ■ Block Diagram of Port 8 Pins

Figure 9.11-1 shows the block diagram of P80/IN2 and P81/IN3 pins.

**Figure 9.11-1 Block diagram of P80/IN2 and P81/IN3 pins**

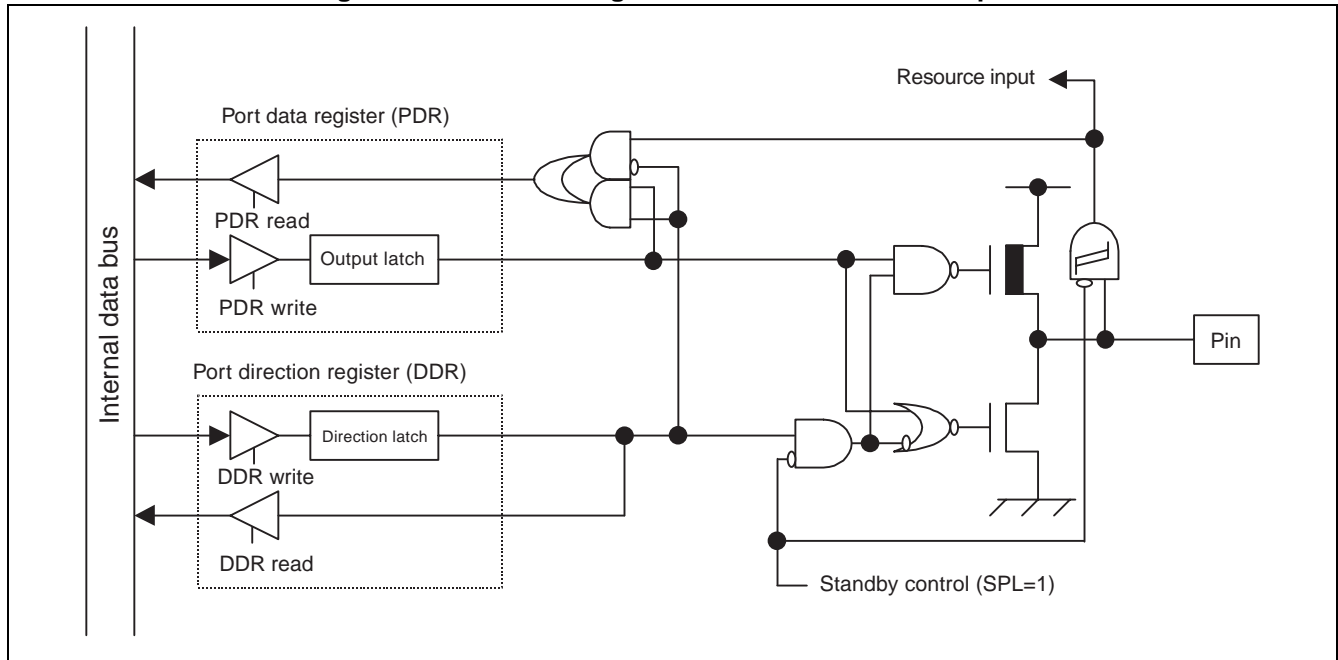
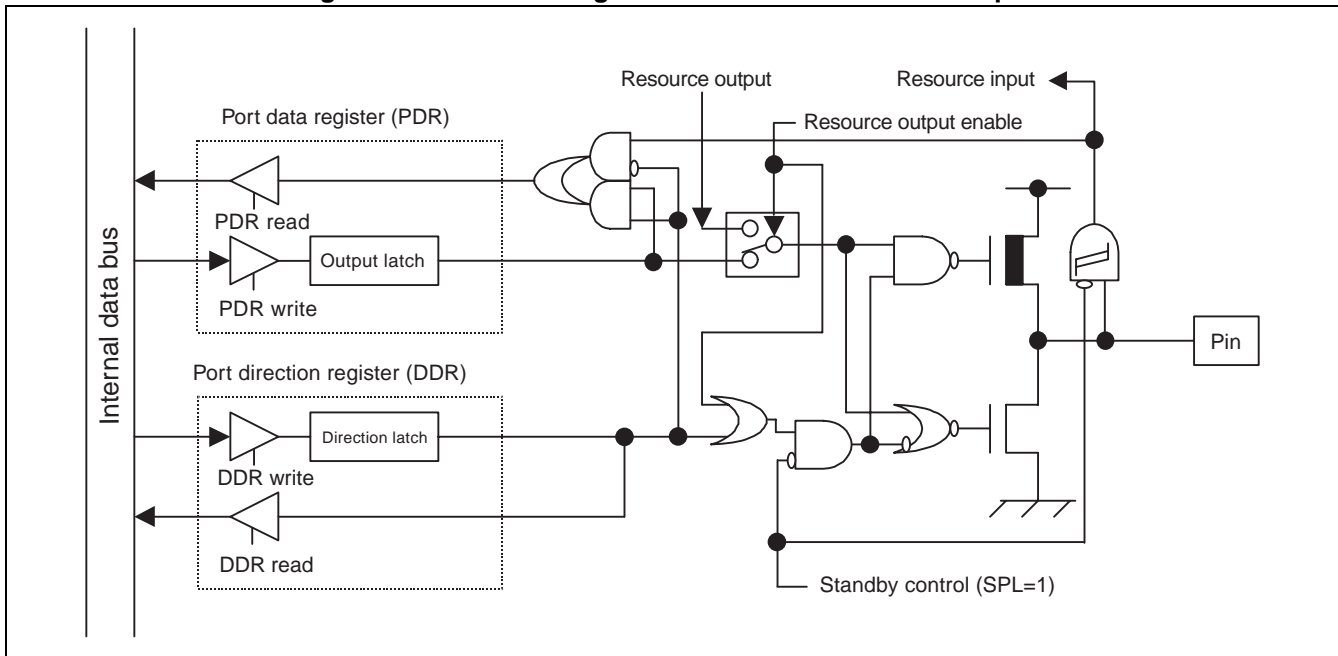


Figure 9.11-2 shows the block diagram of P82/RTO0 to P87/RTO5 pins.

**Figure 9.11-2 Block diagram of P82/RTO0 to P87/RTO5 pins**



When the resource output enable bit is set, the port is forcibly caused to function as resource output pin regardless of the value in the DDR8 register.

■ **Port 8 Registers**

Port 8 registers are PDR8 and DDR8. The bits making up each register correspond to the port 8 pins on a one-to-one basis. Table 9.11-2 lists the port 8 pins and their corresponding register bits.

**Table 9.11-2 Port 8 pins and their corresponding register bits**

Port	Register bits and corresponding port pins								
Port 8	PDR8, DDR8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P87	P86	P85	P84	P83	P82	P81	P80

## MB90820B Series

### 9.11.1 Port 8 Registers (PDR8 and DDR8)

This section describes the port 8 registers.

#### ■ Functions of Port 8 registers

##### ● Port 8 data register (PDR8)

The PDR8 register indicates the state of each pin for port 8.

##### ● Port 8 direction register (DDR8)

The DDR8 register specifies the direction of a data flow (input or output) at each pin (bit) of port 8. When a DDR8 register bit is "1", the corresponding port (pin) is set as an output port. When the bit is "0", the port (pin) is set as an input port.

Notes:

- When a resource having output pins is used, the port functions as resource output pins regardless of the value in the DDR8 register as long as the resource output enable bit corresponding to the pins is set.
- To use a resource having input pins, set the port direction register bit corresponding to each resource input pin to "0" to place the port in input mode.

Table 9.11-3 lists the functions of the port 8 registers.

**Table 9.11-3 Port 8 register functions**

Register name	Data	During reading	During writing	Read /Write	Address	Initial value
Port 8 data register (PDR8)	0	The pin is at the "L" level.	The output latch is loaded with "0". When the pin functions as an output port, the pin is set to the "L" level.	R/W	000008 <sub>H</sub>	XXXXXXXX <sub>B</sub>
	1	The pin is at the "H" level.	The output buffer is turned off to place the port in input mode.			
Port 8 direction register (DDR8)	0	The direction latch is "0".	The output buffer is turned off to place the port in input mode.	R/W	000018 <sub>H</sub>	00000000 <sub>B</sub>
	1	The direction latch is "1".	The output buffer is turned off to place the port in input mode.			

R/W : Readable and writable

X : Undefined

## 9.11.2 Operation of Port 8

---

**This section describes the operation of port 8.**

---

### ■ Operation of Port 8

#### ● Port operation in output mode

- Setting a bit of the DDR8 register to "1" places the corresponding port pin in output mode.
- Data written to the PDR8 register in output mode is held in the output latch of the PDR8 and output to the port pins.
- The PDR8 register can be accessed in read mode to read the value at the port pins (the same value as in the output latch of the PDR8).

---

**Note :**

If a read-modify-write instruction (such as an instruction that sets bits) is used with the port data register, the target bits of the register are set to the specified value. The bits that have been specified for output using the DDR register are not affected, but for the bits that have been specified for input, a value input from the pins is written to the output latch and output as it is. Before switching the mode for the bits from input to output, therefore, write the output data to the PDR register, then specify output mode in the DDR register.

---

#### ● Port operation in input mode

- Setting a bit of the DDR8 register to "0" places the corresponding port pin in input mode.
- In input mode, the output buffer is turned off, and the pins are placed in a high-impedance state.
- Data written to the PDR8 register in input mode is held in the output latch of the PDR8 but not output to the port pins.
- The PDR8 register can be accessed in read mode to read the level value ("0" or "1") at the port pins.

#### ● Port operation for resource output

The resource output enable bit is set to enable the port to be used for resource output. The state of the resource enable bit takes precedence when specifying a switch between input and output. Even if a DDR8 register bit is "1", the corresponding port pin is used for resource output if the resource has been enabled for output. Because the value at the pins can be read even if resource output is enabled, the resource output value can be read.

#### ● Port operation for resource input

When the port is also used for resource input, the value at the pins is always supplied as resource inputs. To use an external signal for the resource, set the PDR8 register to "0" to place the port in input mode.

#### ● Port operation after a reset

- When the MCU is reset, the DDR8 register is initialized to "0". As a result, the output buffer is turned off (I/O mode changes to input), and the pins are placed in a high-impedance state.

- The PDR8 register is not initialized when the MCU is reset. To use the port in output mode, therefore, output mode must be specified in the DDR8 register after the output data is set in the PDR8 register.

● Port operation in stop or time-base timer mode

If the pin state specification bit (SPL) in the low-power consumption mode control register (LPMCR) is already "1" when the port is shifted to stop or time-base timer mode, the port pins are placed in a high-impedance state. This is because the output buffer is turned off forcibly regardless of the value in the PDR8 register. Note that the inputs are fixed at "H" level or "L" level to prevent leakage due to an open circuit. Table 9.11-4 lists the states of the port 8 pins.

**Table 9.11-4 States of port 8 pins**

Pin	Normal operation	Sleep mode	Stop mode or time-base timer mode (SPL = 0)	Stop mode or time-base timer mode (SPL = 1)
P80/IN2 to P87/RTO5	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input disabled/output in Hi-Z

SPL : Pin state specification bit of low-power consumption mode control register (LPMCR)

Hi-Z : High impedance





# **CHAPTER 10**

---

## ***TIME-BASE TIMER***

**This chapter describes the functions and operation of the time-base timer.**

- 10.1 Overview of the Time-base Timer
- 10.2 Configuration of the Time-base Timer
- 10.3 Time-base Timer Control Register (TBTC)
- 10.4 Time-base Timer Interrupts
- 10.5 Operation of the Time-base Timer
- 10.6 Usage Notes on the Time-base Timer

## 10.1 Overview of the Time-base Timer

The time-base timer is an 18-bit free-run counter (time-base counter) that counts up in synchronization with the internal count clock (one-half of the source oscillation). The timer has an interval timer function that can select four interval times.

The time-base timer also has functions for timer of the oscillation stabilization wait time and for supplying the clocks for the watchdog timer.

### ■ Interval Timer Function

The interval timer function repeatedly generates an interrupt request at a given interval.

- An interrupt request is generated when the interval timer bit for the time-base counter overflows.

The interval time bit (interval) can be selected from four types. Table 9.1-1 lists the interval time for the time-base timer.

**Table 10.1-1 Interval time for the time-base timer**

Internal count clock cycle	Interval time
2 / HCLK (0.5 $\mu$ s)	$2^{12}$ / HCLK (Approx. 1.0 ms)
	$2^{14}$ / HCLK (Approx. 4.1 ms)
	$2^{16}$ / HCLK (Approx. 16.4 ms)
	$2^{19}$ / HCLK (Approx. 131.1 ms)

HCLK: Oscillation clock frequency

Values in parentheses are for a 4 MHz oscillation clock frequency.

## ■ Clock Supply Function

The clock supply function supplies clocks to the oscillation stabilization wait time timer and to some peripheral functions.

Table 9.1-2 lists the cycle times of clocks supplied from the time-base timer to each peripheral.

**Table 10.1-2 Clock cycle time supplied from the time-base timer**

Clock supply destination	Clock cycle time	Remarks
Oscillation stabilization wait time	$2^{13} / \text{HCLK}$ (Approx. 2.0 ms)	Oscillation stabilization wait time for ceramic vibrator
	$2^{15} / \text{HCLK}$ (Approx. 8.2 ms)	Oscillation stabilization wait time for crystal vibrator
	$2^{18} / \text{HCLK}$ (Approx. 65.4 ms)	
Watchdog timer	$2^{12} / \text{HCLK}$ (Approx. 1.0 ms)	Count-up clock for watchdog timer
	$2^{14} / \text{HCLK}$ (Approx. 4.1 ms)	
	$2^{16} / \text{HCLK}$ (Approx. 16.4 ms)	
	$2^{19} / \text{HCLK}$ (Approx. 131.1 ms)	

HCLK: Oscillation clock frequency

Values in parentheses occurs during operation of the 4 MHz oscillation clock frequency.

### Reference:

The oscillation stabilization wait time is the yardstick because the oscillation cycle time is unstable as soon as oscillation starts.

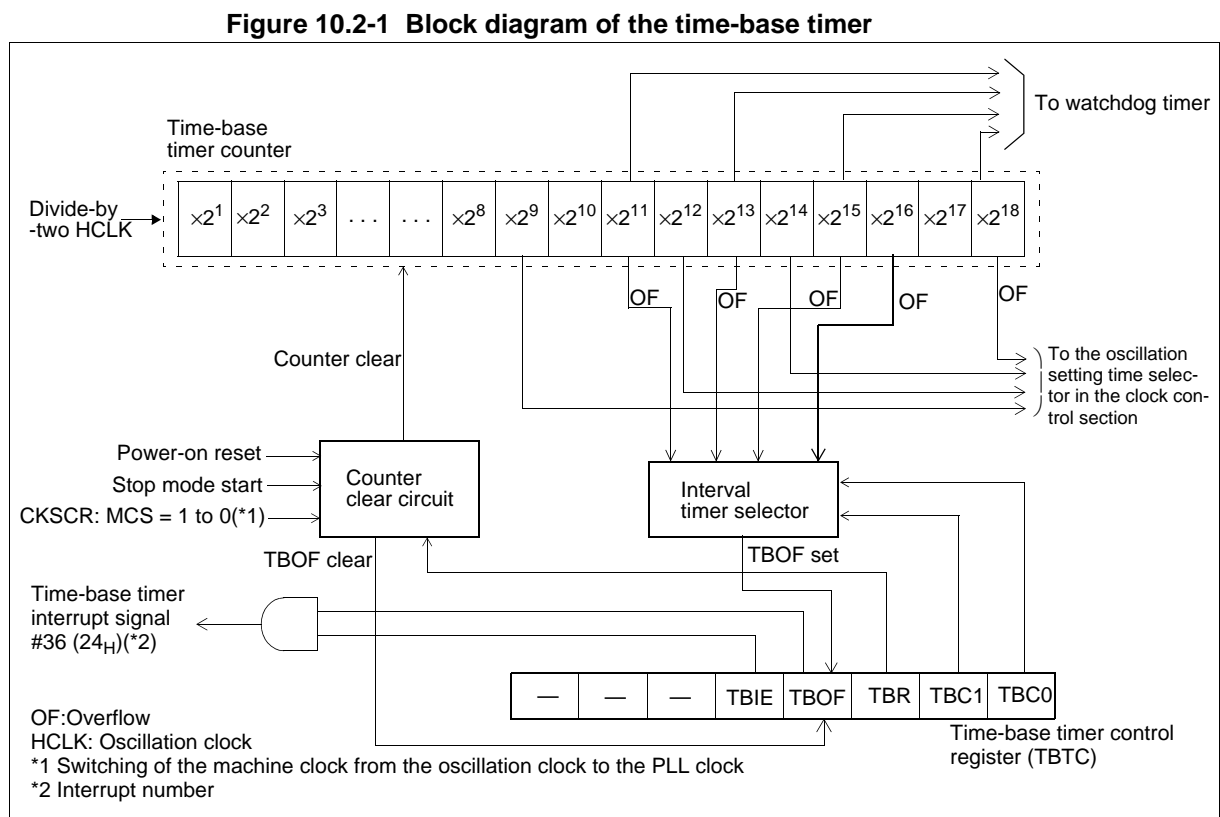
## 10.2 Configuration of the Time-base Timer

The time-base timer consists of the following four blocks:

- Time-base timer counter
- Counter clear circuit
- Interval timer selector
- Time-base timer control register (TBTC)

### ■ Block Diagram of the Time-base Timer

Figure 9.2-1 shows the block diagram of the time-base timer.



#### ● Time-base timer counter

This 18-bit up counter uses the divide-by-two clock of the oscillation clock (HCLK) as the count clock.

#### ● Counter clear circuit

Used to clear the counter by writing "0" to the TBTC:TBR bit, by a power-on reset, or by transition to stop mode (LPMCR: STP = 1).

#### ● Interval timer selector

Selects one of four outputs for the time-base timer counter. An overflow of the selected bit becomes an interrupt cause.

- Time-base timer control register (TBTC)

Selects the interval time, clears the counter of the time-base timer, controls an interrupt request, and checks the status.

## 10.3 Time-base Timer Control Register (TBTC)

The time-base timer control register (TBTC) selects the interval time, clears the counter, controls interrupts, and checks the status.

### ■ Time-base Timer Control Register (TBTC)

Figure 10.3-1 Time-base timer control register (TBTC)

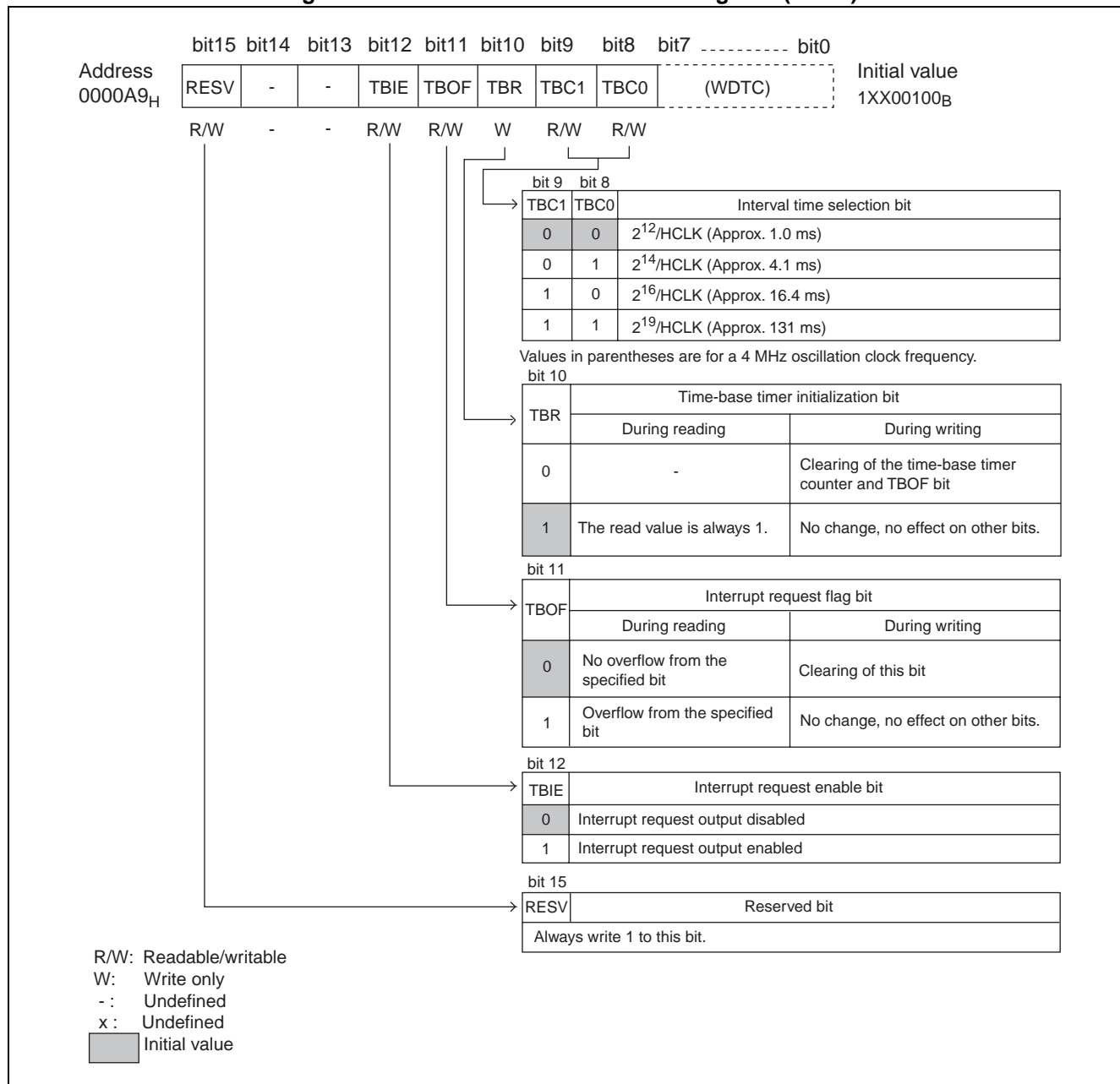


Table 10.3-1 Function description of each bit in the time-base timer control register (TBTC)

Bit name		Function
bit15	RESV: Reserved bit	<b>Note :</b> Always write "1" to this bit.
bit14, bit13	Not used bits	<ul style="list-style-type: none"> <li>When read, the value is undefined.</li> <li>Writing has no effect on operation.</li> </ul>
bit12	TBIE: Interrupt request enable bit	<ul style="list-style-type: none"> <li>Used to enable or disable an interrupt request to the CPU.</li> <li>When this bit and the interrupt request flag bit (TBOF) are "1", an interrupt request is output.</li> </ul>
bit11	TBOF: Interrupt request flag bit	<ul style="list-style-type: none"> <li>This bit is set to "1" when the bit specifying the time-base timer counter overflows.</li> <li>When this bit and the interrupt request enable bit (TBIE) are "1", an interrupt request is generated.</li> <li>During writing, this bit is cleared with "0". If "1" is written, the bit does not change and there is no effect.</li> </ul> <b>Note :</b> <ul style="list-style-type: none"> <li>To clear the TBOF bit, disable the time-base timer interrupt by specifying the TBIE bit or ILM bit of processor status (PS).</li> <li>The TBOF bit is cleared by writing "0", by a transition to stop mode, by clearing of the time-base timer with the TBR bit, or by a reset.</li> </ul>
bit10	TBR: Time-base timer initialization bit	<ul style="list-style-type: none"> <li>Used to clear the time-base timer counter.</li> <li>When "0" is written to this bit, the counter is cleared and the TBOF bit is cleared. If "1" is written, the bit does not change and there is no effect.</li> </ul> <b>Reference:</b> The read value is always "1".
bit9, bit8	TBC1, TBC0: Interval time selection bit	<ul style="list-style-type: none"> <li>Used to select an interval time.</li> <li>The bit for the interval timer of the time-base timer time is specified.</li> <li>Four types of interval time can be selected.</li> </ul>



## 10.4 Time-base Timer Interrupts

The time-base timer can generate an interrupt request when the bit specifying the time-base timer counter overflows.

### ■ Time-base Timer Interrupts

The interrupt request flag bit (TBTC: TBOF) is set to "1" when the time-base timer counter counts up with the internal count clock and when the selected interval timer bit overflows. In this case, if the interrupt request enable bit has been enabled (TBTC: TBIE = 1), an interrupt request (#36) is generated in the CPU. Write "0" to the TBOF bit in the interrupt handling routine to clear the interrupt request. When the specified bit overflows, the TBOF bit is set to "1" regardless of the TBIE bit value.

Note :

Clear the interrupt request flag bit (TBTC: TBOF) while a time-base timer interrupt is disabled by setting the TBIE bit or ILM bit of the processor status (PS).

Reference:

When the TBOF bit is "1", if the TBIE bit status is switched from disable to enable (0 → 1), an interrupt request occurs immediately.

### ■ Time-base Timer Interrupts and EI<sup>2</sup>OS

Table 9.4-1 lists the time-base timer interrupt and EI<sup>2</sup>OS.

**Table 10.4-1 Time-base timer interrupts and EI<sup>2</sup>OS**

Interrupt number	Interrupt level setting register		Vector table address			EI <sup>2</sup> OS
	Register name	Address	Lower	Medium	Upper	
#36 (24 <sub>H</sub> )	ICR12	0000BC <sub>H</sub>	FFFF6C <sub>H</sub>	FFFF6D <sub>H</sub>	FFFF6E <sub>H</sub>	Δ

Δ: Usable when an interrupt cause that shares the ICR is not used.

Note :

ICR12 is common to the time-base timer interrupt and input capture channels 2/3 interrupt. Interrupts can be used for two applications, but the interrupt level is the same.

## MB90820B Series

### 10.5 Operation of the Time-base Timer

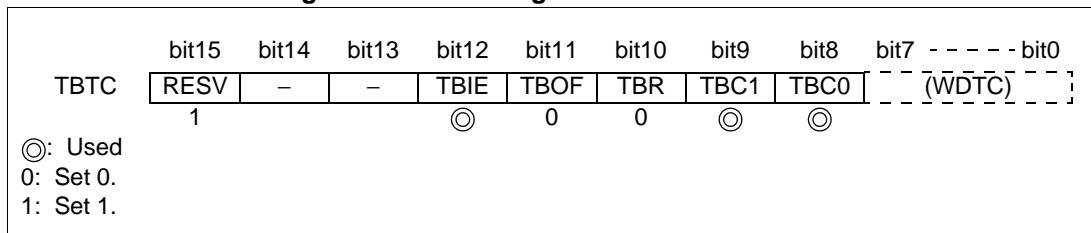
The time-base timer provides the interval timer function and the clock supply function that supplies clocks to some peripheral functions.

#### ■ Operation of the Interval Timer Function (Time-base Timer)

The interval timer function generates an interrupt request for each interval.

The setting in Figure 9.5-1 is required to all the timer to operate as an interval timer.

**Figure 10.5-1 Setting of the time-base timer**



- The time-base timer counter continues counting up in synchronization with the internal count clock (one-half of the oscillation clock) as long as the clock is being oscillated.
- When the counter is cleared (TBR = 0), it counts up from "0". When the interval timer bit overflows, the interrupt request flag bit (TBOF) is set to "1". At this time, if interrupt request output has been enabled (TBIE = 1), an interrupt is generated for each selected interval based on the cleared time.
- The interval time may become longer than the time set because of time-base timer clearing.

## ■ Oscillation Stabilization Time Timer Function

The time-base timer is also used as the oscillation wait time timer for oscillation and the PLL clocks.

The oscillation stabilization time is set for the interval time from the time the counter counts up from "0" (count clear) until the oscillation wait time bit overflows. When control returns from time-base timer mode to PLL clock mode, the oscillation wait time starts from the middle of counting because the time-base timer counter has not been cleared. Table 9.5-1 shows the clearing of the time-base counter and the oscillation wait times.

**Table 10.5-1 Time-base timer counter clearing and stabilization wait times**

Operation	Counter clear	TBOF clear	Oscillation stabilization wait time
TBTC: Writing of "0" to TBR	O	O	-
Power-on reset	O	O	Oscillation clock oscillation stabilization wait time
Watchdog reset			
Releasing of stop mode	O	O	Oscillation clock oscillation stabilization wait time (at return to main clock mode)
Transition from oscillation clock mode to PLL clock mode (MCS = 1 → 0)	O	O	PLL clock oscillation stabilization wait time
Releasing of time-base timer mode	X	X	PLL clock oscillation stabilization wait time (at return to PLL clock mode)
Releasing of sleep mode	X	X	Not available

O: Available

X: Not available

## ■ Supply of Operation Clock

The time-base timer supplies clocks to the watchdog timer. Clearing of the time-base counter affects operation of the watchdog timer.

**MB90820B Series****10.6 Usage Notes on the Time-base Timer**

---

**Notes about the effects on peripheral functions of clearing interrupt requests and the time-base timer counter are given below.**

---

**■ Time-base Timer Usage Notes****● Clearing interrupt requests**

The TBOF bit of the time-base timer control register must be cleared while a time-base timer interrupt is masked by the TBIE bit or the interrupt level mask register (ILM) of the processor status (PS).

**● Effects of time-base timer clearing**

Clearing of the time-base timer counter affects the following operations:

- When the time-base timer is using the interval timer function (interval interrupt)
- When the watchdog timer is being used

**● Use of the time-base timer as the oscillation stabilization wait time timer**

At power-on, the source oscillation of the main clock stops in main stop mode. After oscillator operation starts, the operating clock supplied by the time-base timer is used to take the oscillation stabilization wait time of the main clock. An appropriate oscillation stabilization wait time must be selected based on the type of oscillating element connected to the main clock oscillator (clock generation section). See Section "4.5 Oscillation Stabilization Wait Interval", for details.

**● Notes on peripheral functions to which clocks are supplied from the time-base timer**

In the mode in which the main clock source oscillation stops, the time-base timer counter is cleared and time-base timer operation stops. When the time-base timer counter is cleared, the clock supplied from the time-base timer is supplied from its initial state. As a result, the "H" level is shortened and the "L" level lengthened 1/2 cycle. Although the clock for the watchdog timer is also supplied from its initial state; the watchdog timer operates in normal cycles because the watchdog timer counter is cleared at the same time as the time-base timer counter is cleared.

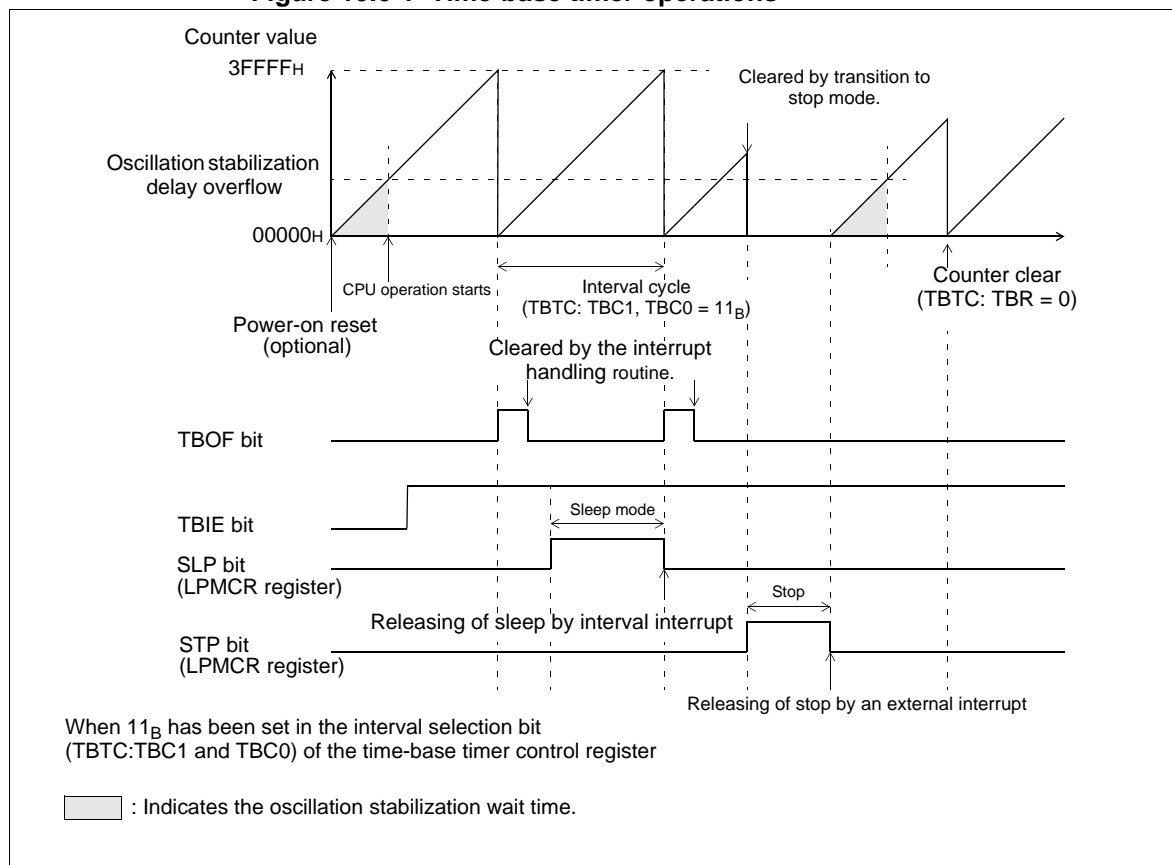
**■ Operation of the Time-base Timer**

The following operations are shown in Figure 9.6-1:

- A power-on reset occurs.
- Sleep mode is entered during operation of the interval timer function.
- A counter clear request is issued.

When stop mode is entered, the time-base timer is cleared and its operation stops. On return from stop mode, the time-base timer immediately counts the oscillation stabilization wait time.

Figure 10.6-1 Time-base timer operations



# **CHAPTER 11**

---

# **WATCHDOG TIMER**

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

- 11.1 Overview of the Watchdog Timer
- 11.2 Configuration of the Watchdog Timer
- 11.3 Watchdog Timer Control Register (WDTC)
- 11.4 Operation of the Watchdog Timer
- 11.5 Usage Notes on the Watchdog Timer

## 11.1 Overview of the Watchdog Timer

The watchdog timer is a 2-bit counter that uses the time-base timer supply clock as the count clock. After activation, if the watchdog timer is not cleared within a given time, the CPU is reset.

### ■ Watchdog Timer Function

The watchdog timer is a counter for handling program crashes. Once the watchdog timer is activated, it must be regularly cleared within a given time. If the program results in an endless loop and the watchdog timer is not cleared over a given time, a watchdog reset is generated for the CPU.

Table 10.1-1 lists the watchdog timer interval times. If the watchdog timer is not cleared, a watchdog reset is generated between the minimum time and maximum time. Clear the counter within the minimum time listed in this table.

**Table 11.1-1 Interval times for the watchdog timer**

Interval time		
Minimum*	Maximum*	Oscillation clock cycle count
Approx. 3.58 ms	Approx. 4.61 ms	$2^{14} \pm 2^{11}$
Approx. 14.33 ms	Approx. 18.3 ms	$2^{16} \pm 2^{13}$
Approx. 57.23 ms	Approx. 73.73 ms	$2^{18} \pm 2^{15}$
Approx. 458.75 ms	Approx. 589.82 ms	$2^{21} \pm 2^{18}$

\* Value during operation of the 4 MHz oscillation clock frequency

The maximum and minimum watchdog timer interval times and the oscillation clock cycle count depend on the clear timing of the watchdog timer.

The interval time is 3.5 to 4.5 times longer than the cycle of the count clock (time-base timer supply clock). See Section "11.4 Operation of the Watchdog Timer".

#### Note :

The watchdog timer consists of a 2-bit counter that uses the carry signals of the time-base timer as count clocks. Therefore, if the time-base timer is cleared, the watchdog reset generation time may become longer than the time set.

#### Reference:

At activation, the watchdog timer is initialized by a power-on or watchdog reset and is placed in stopped status. The watchdog timer is cleared by an external pin reset, software reset, writing to the WTE bit (watchdog timer control register), transition to sleep mode or stop mode. However, It is not stopped.

## MB90820B Series

### 11.2 Configuration of the Watchdog Timer

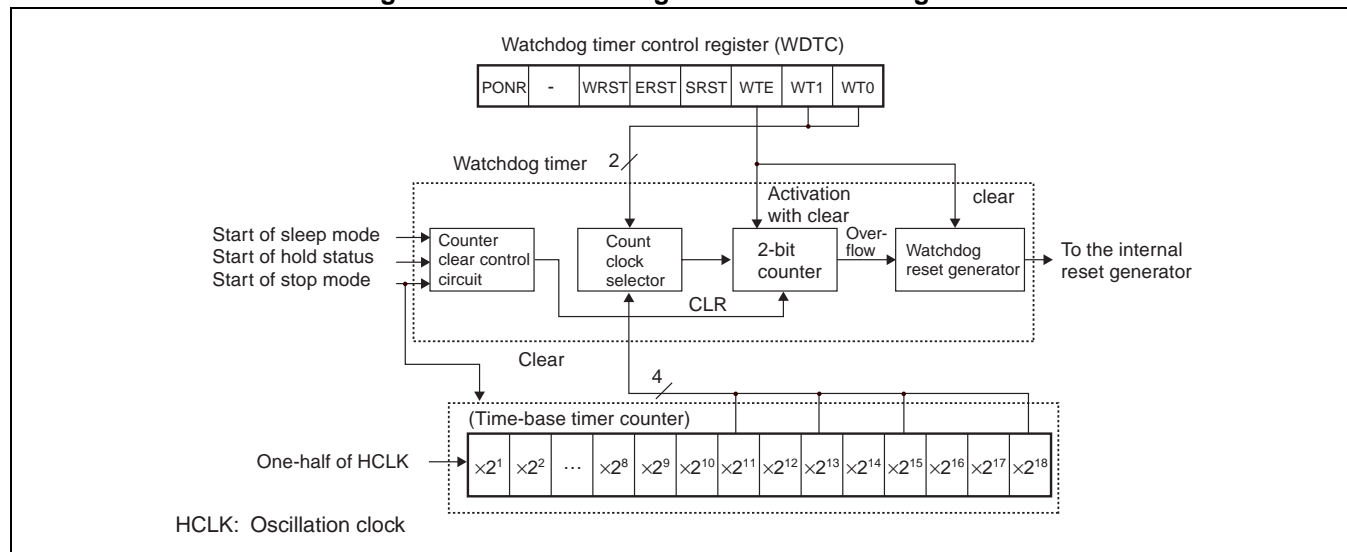
The watchdog timer consists of the following five blocks:

- Count clock selector
- Watchdog counter (2-bit counter)
- Watchdog reset generator
- Counter clear control circuit
- Watchdog timer control register (WDTC)

#### ■ Block Diagram of the Watchdog Timer

Figure 10.2-1 shows the block diagram of the watchdog timer.

Figure 11.2-1 Block diagram of the watchdog timer



#### ● Count clock selector

This circuit is used to select the count clock of the watchdog timer from four types of time-base timer outputs. This determines the watchdog reset generation time.

#### ● Watchdog counter (2-bit counter)

This 2-bit up counter uses the time-base timer output as the count clock.

#### ● Watchdog reset generator

Used to generate the reset signal by an overflow of the watchdog counter.

#### ● Counter clear circuit

Used to clear the watchdog counter and to control the operation or stopping of the counter.



● Watchdog timer control register (WDTC)

Used to activate or clear the watchdog timer; holds the reset generation cause.

## MB90820B Series

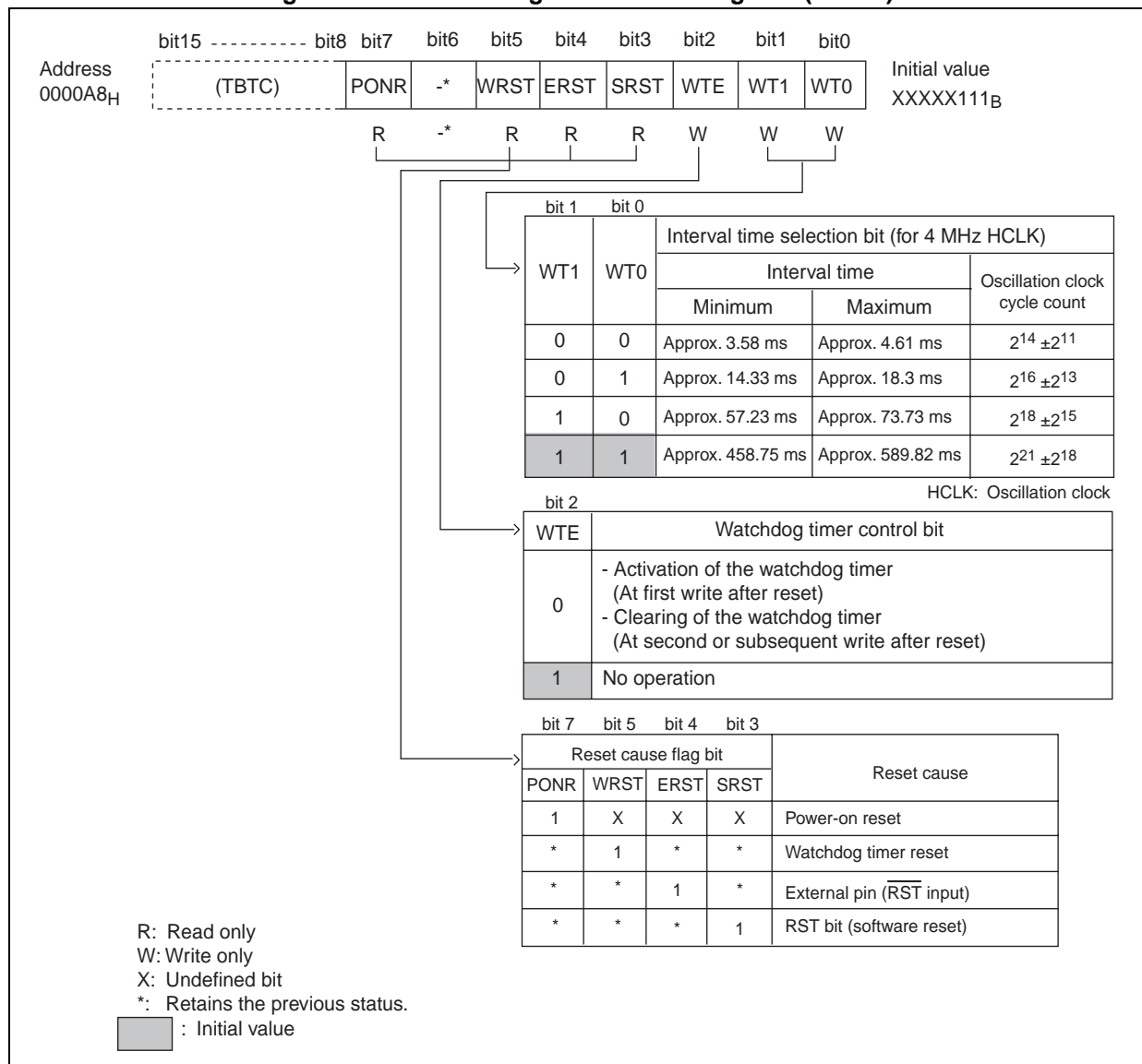
## 11.3 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) activates and clears the watchdog timer and displays the reset cause.

### ■ Watchdog Timer Control Register (WDTC)

Figure 10.3-1 shows the watchdog timer control register (WDTC). Table 10.3-1 describes the function of each bit in the watchdog timer control register (WDTC).

**Figure 11.3-1 Watchdog timer control register (WDTC)**



The interval time becomes 3.5 to 4.5 times longer than the count clock (time-base timer output value) cycle. For details, see Section "11.4 Operation of the Watchdog Timer".

**Table 11.3-1 Function description of each bit in the watchdog timer control register (WDTC)**

Bit name		Function
bit7 to bit3	PONR, WRST, ERST, SRST: Reset cause bits	<ul style="list-style-type: none"> <li>Read-only bits for indicating the reset cause. If more than one reset cause occurs, the bit for each reset cause occurring is set to "1".</li> <li>These bits are all cleared to "0" after the watchdog timer control register (WDTC) is read.</li> <li>At power-on, the contents of the bits other than the PONR bit are not guaranteed. Therefore, when the PONR bit is "1", ignore the contents of the bits other than the PONR bit.</li> </ul>
bit6	unused	<ul style="list-style-type: none"> <li>When read, the value is undefined. Writing has no effect on operation.</li> </ul>
bit2	WTE: Watchdog timer control bit	<ul style="list-style-type: none"> <li>When "0" is written to this bit, the watchdog timer is activated (first write after reset) or the 2-bit counter is cleared (second or subsequent write after reset).</li> <li>Writing "1" does not affect operation.</li> </ul>
bit1, bit0	WT1, WT0: Interval time selection bits	<ul style="list-style-type: none"> <li>Used to select the watchdog timer interval time.</li> <li>Only data at watchdog timer activation is valid. Data written after watchdog timer activation is ignored.</li> <li>These bits are write-only.</li> </ul>

MB90820B Series

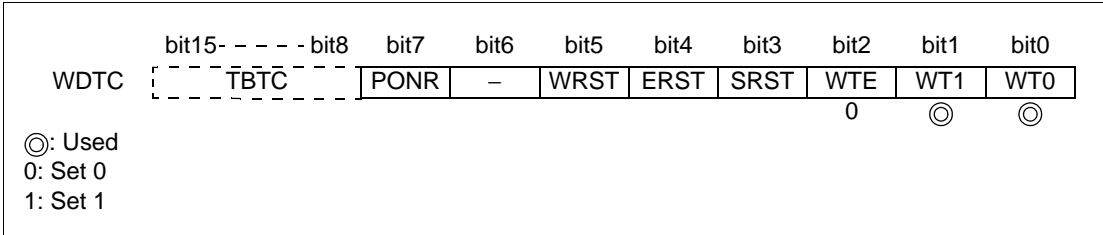
11.4 Operation of the Watchdog Timer

The watchdog timer generates a watchdog reset by an overflow of the watchdog counter.

■ Watchdog Timer Operation

Operation of the watchdog timer requires the setting in Figure 10.4-1.

Figure 11.4-1 Setting of the watchdog timer

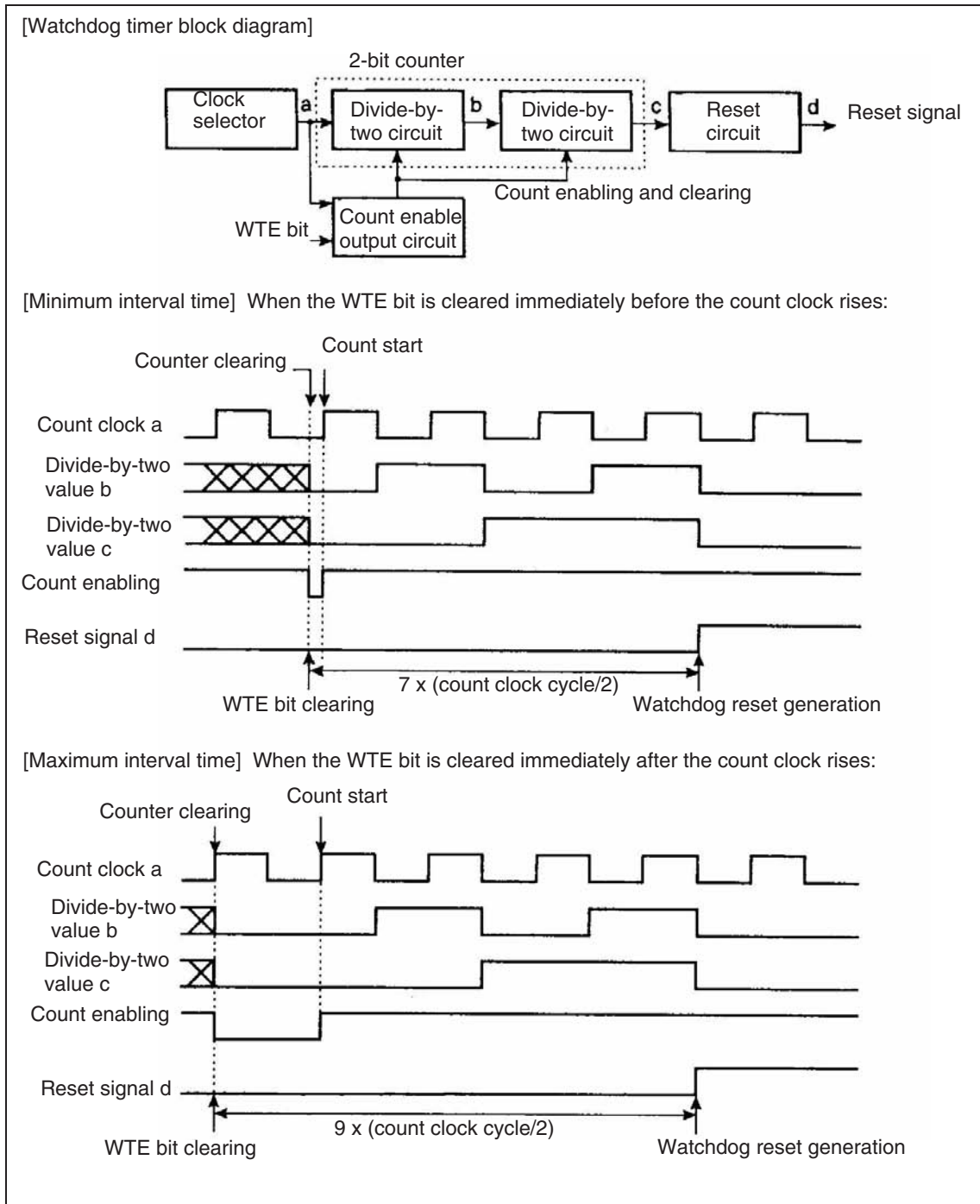


- Activating the watchdog timer
  - The watchdog timer is activated when the first "0" after reset is written to the WTE bit of the watchdog timer control register (WDTC). Specify the interval time by specifying the WT1 and WT0 bits of the watchdog timer control register at the same time.
  - When watchdog timer activation starts, it can be stopped only by a power-on or its own reset.
- Clearing the watchdog timer
  - When a second or subsequent "0" is written to the WTE bit, the 2-bit counter of the watchdog timer is cleared. If the counter is not cleared within the specified interval time, it overflows and a watchdog reset occurs.
  - The watchdog counter is cleared by reset generation, transition to sleep mode, stop mode, or clock mode.
- Intervals for the watchdog timer

Figure 10.4-2 shows the relationship between the clear timing of the watchdog timer and interval times. The interval time changes according to the clear timing of the watchdog timer and requires 3.5 to 4.5 times longer than the count clock cycle.
- Checking a reset cause

A reset cause can be determined by checking the PONR, WRST, ERST, and SRST bits of the watchdog timer control register (WDTC) after a reset.

Figure 11.4-2 Clear timing and interval times of watchdog timer



**MB90820B Series****11.5 Usage Notes on the Watchdog Timer**

---

Notes on using the watchdog timer are given below.

---

**■ Usage Notes on the Watchdog Timer****● Stopping the watchdog timer**

Once the watchdog timer is activated, it cannot stop until a power-on or watchdog reset occurs. The watchdog timer counter is cleared by an external reset or software reset; however, the watchdog timer does not stop.

**● Interval times**

Since a carry signal of the time-base timer is used as the count clock for the interval, the watchdog timer interval time may become longer than the setting time when the time-base timer is cleared.

**● Selecting the interval time**

The interval can be set when the watchdog timer is activated. Data written during operations other than activation is ignored.

**● Notes on program creation**

When a program that repeatedly clears the watchdog timer in the main loop is created, the processing time of the main loop including the interrupt processing must be equal to or less than the minimum watchdog timer interval time.

**● Watchdog timer operation in time-base timer mode**

The time-base timer operates while the time-base timer mode is set. The watchdog timer, however, is temporarily stopped.



# **CHAPTER 12**

---

## ***16-BIT RELOAD TIMER***

**This chapter describes the functions and operations of the 16-bit reload timer.**

- 12.1 Overview of 16-Bit Reload Timer
- 12.2 Block Diagram of 16-Bit Reload Timer
- 12.3 Pins of 16-Bit Reload Timer
- 12.4 Registers of 16-Bit Reload Timer
- 12.5 Interrupts of 16-Bit Reload Timer
- 12.6 Operation of 16-Bit Reload Timer
- 12.7 Notes on Using the 16-Bit Reload Timer



## 12.1 Overview of 16-Bit Reload Timer

The 16-bit reload timer has two modes: Internal clock mode (with countdown performed in synchronization with three types of internal clock), and event count mode (with countdown performed by detecting any pulse edge input to the external pin). Either mode may be selected. The timer defines an underflow when the counter value is in the range from "0000<sub>H</sub>" to "FFFF<sub>H</sub>". In other words, an underflow occurs at a count of [reload register's setting value +1].

The counter can be used to select either reload mode, in which an underflow causes the count set value to be reloaded for repeated counting, or one-shot mode, in which counting is stopped when an underflow occurs. Counter underflow may generate an interrupt and supports the extended intelligent I/O service (EI<sup>2</sup>OS).

### ■ Operation Mode of 16-bit Reload Timer

Table 12.1-1 lists the operation modes of the 16-bit reload timer.

**Table 12.1-1 Operation Modes of 16-bit Reload Timer**

Clock mode	Counter operation mode	Operation mode
Internal clock mode	Reload mode	Software trigger operation External trigger input operation External gate input operation
	One-Shot mode	
Event count mode (External clock mode)	Reload mode	Software trigger operation
	One-Shot mode	

### ■ Internal Clock Mode

One type of count clock is selected among three types of internal clock modes to operate as follows:

#### ● Software trigger operation

Sets the timer control status register (TMCSR0/TMCSR1): TRG bit to "1" to start count operation. Trigger input by using the TRG bit is also enabled for external trigger input and external gate input.

#### ● External trigger operation

Starts counting when the edge selected (leading, trailing, or both edges) is input to the TIN0/TIN1 pins.

#### ● External gate input operation

Continues counting when the signal level selected ("L" or "H") is input to the TIN0/TIN1.

## MB90820B Series

### ■ Event Count Mode (External Clock Mode)

Event count mode provides a function for starting countdown when a valid edge selected (leading, trailing, or both edges) is input to the TIN0/TIN1 pins. It is also used as an interval timer when using an external clock with a constant interval.

### ■ Counter Operation

#### ● Reload mode

If the countdown causes an underflow, and a transfer of the type 0000<sub>H</sub> --> FFFF<sub>H</sub> occurs, the setting value for counting is reloaded so that counting can continue. An underflow can trigger an interrupt request, which may be used for providing an interval timer. A toggled waveform, which reverses itself at every underflow, is output from the TO0/TO1 pins. Table 12.1-2 lists the interval time for the 16-bit reload timer.

**Table 12.1-2 Interval Time of 16-bit Reload Timer**

Count clock	Count clock interval	Interval time
Internal count clock	$2^1/\phi$ (0.125 $\mu$ s)	0.125 $\mu$ s to 8.192 ms
	$2^3/\phi$ (0.5 $\mu$ s)	0.5 $\mu$ s to 32.768 ms
	$2^5/\phi$ (2.0 $\mu$ s)	2.0 $\mu$ s to 131.1 ms
External clock	$2^3/\phi$ or more (0.5 $\mu$ s)	0.5 $\mu$ s or more

$\phi$ : Machine clock. The parenthesized value indicates the clock interval time applied when the machine clock frequency is 16 MHz and the FSEL bit is "1".

**Table 12.1-3 Interval Time of 16-bit Reload Timer**

Count clock	Count clock interval	Interval time
Internal count clock	$2^1/\phi$ (0.167 $\mu$ s)	0.167 $\mu$ s to 10.923 ms
	$2^3/\phi$ (0.667 $\mu$ s)	0.667 $\mu$ s to 43.690 ms
	$2^5/\phi$ (2.667 $\mu$ s)	2.667 $\mu$ s to 174.760 ms
External count clock	$2^3/\phi$ or more (0.667 $\mu$ s)	0.667 $\mu$ s or more

$\phi$ : Machine clock. The parenthesized value indicates the clock interval time applied when the machine clock frequency is 24 MHz and the FSEL bit is "0".

#### ● One-shot mode

If countdown leads to an underflow (0000<sub>H</sub> --> FFFF<sub>H</sub>), count operation will stop. Underflow may also trigger an interrupt. During counter operation, the square wave that indicates counting is output from the TO0/TO1 pins.

References:

- The 16-bit reload timer is used to generate the UART baud rate.
  - The 16-bit reload timer is used to trigger A/D converter operation.
-

# MB90820B Series

## 12.2 Block Diagram of 16-Bit Reload Timer

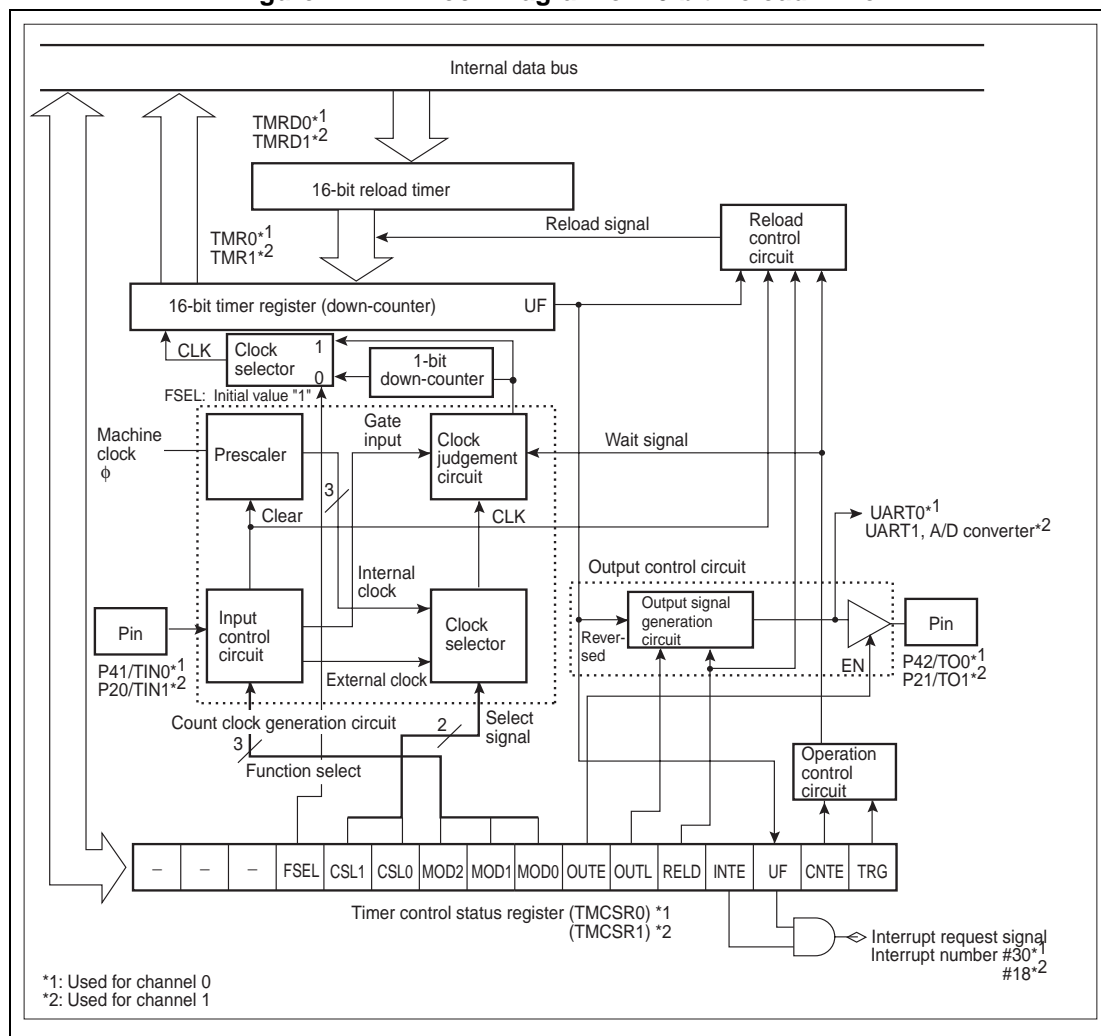
The 16-bit reload timer consists of the following seven blocks:

- Count clock generation circuit
- Reload control circuit
- Output control circuit
- Operation control circuit
- 16-bit timer registers (TMRL0/TMRL1, TMRH0/TMRH1)
- 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1)
- Timer control status registers (TMCSR0/TMCSR1, TMCSRH0/TMCSRH1)

### ■ Block Diagram of 16-bit Reload Timer

Figure 12.2-1 shows a block diagram of the 16-bit reload timer.

Figure 12.2-1 Block Diagram of 16-bit Reload Timer



- **Count clock generation circuit**

The count clock generation circuit generates the count clock for the 16-bit reload timer from the machine clock or external input clock.

- **Reload control circuit**

Controls reload operation when the timer starts and when underflow occurs.

- **Output control circuit**

Controls the reversal of TO0/TO1 pin output due to 16-bit reload timer underflow and the enable or disable states of TO0/TO1 pin output.

- **Operation control circuit**

Controls starting and stopping of the 16-bit reload timer.

- **16-bit timer registers (TMRL0/TMRL1, TMRH0/TMRH1)**

These registers are used to read the current counter value for the 16-bit down counter.

- **16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1)**

These registers are used to set the interval time of the 16-bit reload timer. The set value of the registers is loaded into the 16-bit timer registers for countdown.

- **Timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1)**

These registers are used to select the count clock and operation mode of the 16-bit reload timer, set operating conditions, activate a trigger by software, enable/disable count operation, select reload or one-shot mode, select the pin output level, enable or disable timer output, control clock division, control interrupt, and check the state of operation.

**MB90820B Series****12.3 Pins of 16-Bit Reload Timer**

This section describes the pins of the 16-bit reload timer.

**■ Pins of 16-bit Reload Timer**

The pins of the 16-bit reload timer can also be used for general-purpose I/O ports. Table 12.3-1 lists the pin functions, type of I/O, and settings for using the 16-bit reload timer.

**Table 12.3-1 Pins of the 16-bit Reload Timer**

Pin name	Pin function	Type of I/O	Pull-up operation	Standby control	Setting to use pin
P41/TIN0	I/O and timer input of port 4	CMOS output and CMOS hysteresis input	Not used	Provided	Set to input port. (DDR4:bit1=0)
P42/TO0	I/O and timer output of port 4				Setting to timer output enabled (TMCSRL0:OUTE=1) Sound generator output disabled
P20/TIN1	I/O and timer input of port 2		Selectable		Set to input port (DDR2:bit0=0) PPG1 output disabled
P21/TO1	I/O and timer output of port 2				Setting to timer output enabled (TMCSRL1:OUTE=1) PPG4 output disabled

Reference:

For pin block diagrams, refer to "CHAPTER 9 I/O PORTS".

## 12.4 Registers of 16-Bit Reload Timer

This section lists the registers of the 16-bit reload timer.

### ■ List of Registers for 16-bit Reload Timer

Figure 12.4-1 lists the registers of the 16-bit reload timer.

**Figure 12.4-1 Registers of 16-bit Reload Timer**

	Address	bit15 ..... bit8	bit7 ..... bit0
16-Bit Reload Timer 0	000082 <sub>H</sub> , 000083 <sub>H</sub>	TMCSR0 (Timer control status register)	
	000084 <sub>H</sub> , 000085 <sub>H</sub>	TMR0/TMRD0 (16-bit timer register/16-bit reload register)*	
16-Bit Reload Timer 1	000086 <sub>H</sub> , 000087 <sub>H</sub>	TMCSR1 (Timer control status register)	
	000088 <sub>H</sub> , 000089 <sub>H</sub>	TMR1/TMRD1 (16-bit timer register/16-bit reload register)*	

\*: Functions as a 16-bit timer register (TMR) for reading and as a 16-bit reload register (TMRLR) for writing.

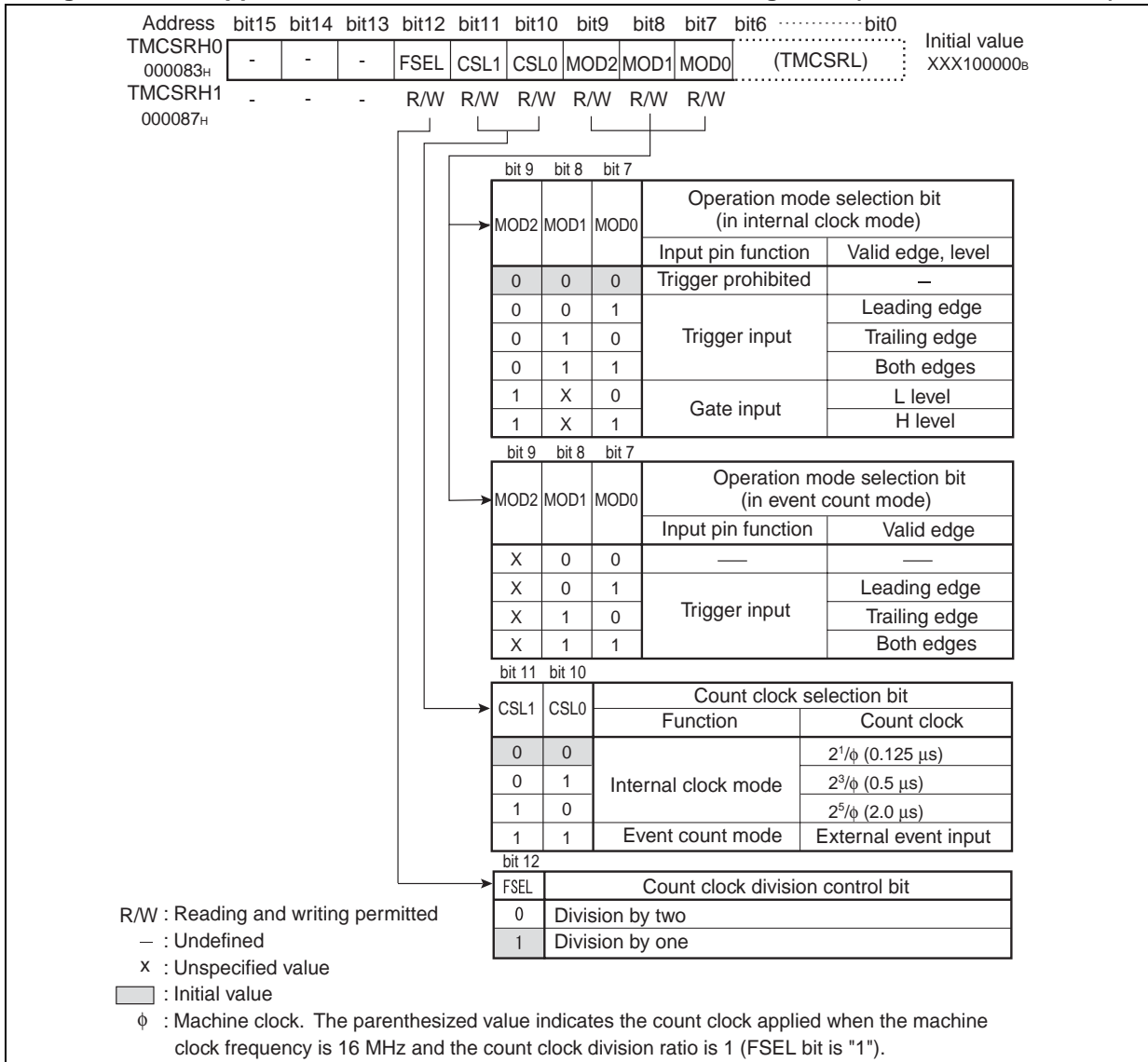
## MB90820B Series

### 12.4.1 Upper Bits of Timer Control Status Registers (TMCSRH0/TMCSRH1)

Upper bits 12 to 8 and lower bit 7 in the timer control status registers (TMCSRH0/TMCSRH1) are used to select the 16-bit reload timer operation mode and set the operating conditions. Use of the last lower bit 7 (MOD0 bit) is also described here.

#### ■ Upper Bits and Bit 7 of Timer Control Status Registers (TMCSRH0/TMCSRH1)

Figure 12.4-2 Upper Bits and Bit 7 of Timer Control Status Registers (TMCSRH0/TMCSRH1)





**Table 12.4-1 Function of the Upper Bits and Bit 7 in Timer Control Status Registers:  
(TMCSRH0, TMCSRH1)**

Bit name		Function
bit15 to bit13	Undefined bits	<ul style="list-style-type: none"> <li>Value at reading is not specified.</li> <li>If write, the bit value is always "1".</li> </ul>
bit12	FSEL: Count clock division control	<ul style="list-style-type: none"> <li>Specifies the count clock division ratio.</li> <li>If the FSEL bit is set to "0", the count clock specified by the count clock selection bits (CSL1 and CSL0) is divided by two.</li> </ul>
bit11, bit10	CSL1, CSL0: Count clock selection bits	<ul style="list-style-type: none"> <li>Selects the count clock of the 16-bit reload timer.</li> <li>Internal clock mode to count the internal clock is selected if the CSL1 and CSL0 bits are other than "11<sub>B</sub>".</li> <li>Event count mode to count external clock edges is selected if the CSL1 and CSL0 bits are "11<sub>B</sub>".</li> </ul>
bit9 to bit7	MOD2, MOD1, MOD0: Operation mode selection bits	<p><b>&lt;Internal clock mode&gt;</b></p> <ul style="list-style-type: none"> <li>The MOD2 bit is used to select the function of the input pin. When the MOD2 bit is set to "0", the input pin is used as a trigger input pin. When a valid edge is input, the 16-bit reload register data is loaded into the counter to continue with count operation. Valid edge types are selected by using the MOD1/0 bits.</li> <li>With the MOD2 bit set to "1", the input pin is used for gate input for counting only when a valid level signal is being input. The MOD0 bit enables selection of a valid level.</li> <li>Because the value of the MOD1 bit has no effect on operation, either value (0 or 1) can be set.</li> </ul> <p><b>&lt;Event count mode&gt;</b></p> <ul style="list-style-type: none"> <li>Because the value of the MOD2 bit has no effect on operation, either value (0 or 1) can be set.</li> <li>The input pin is used as a trigger input pin for event input. A valid edge is selected by using the MOD1/MOD0 bits.</li> </ul> <p><b>Note :</b> The operation mode selection must be counter operation stop mode (TMCSRL0/TMCSRL1: CNTE=0).</p>

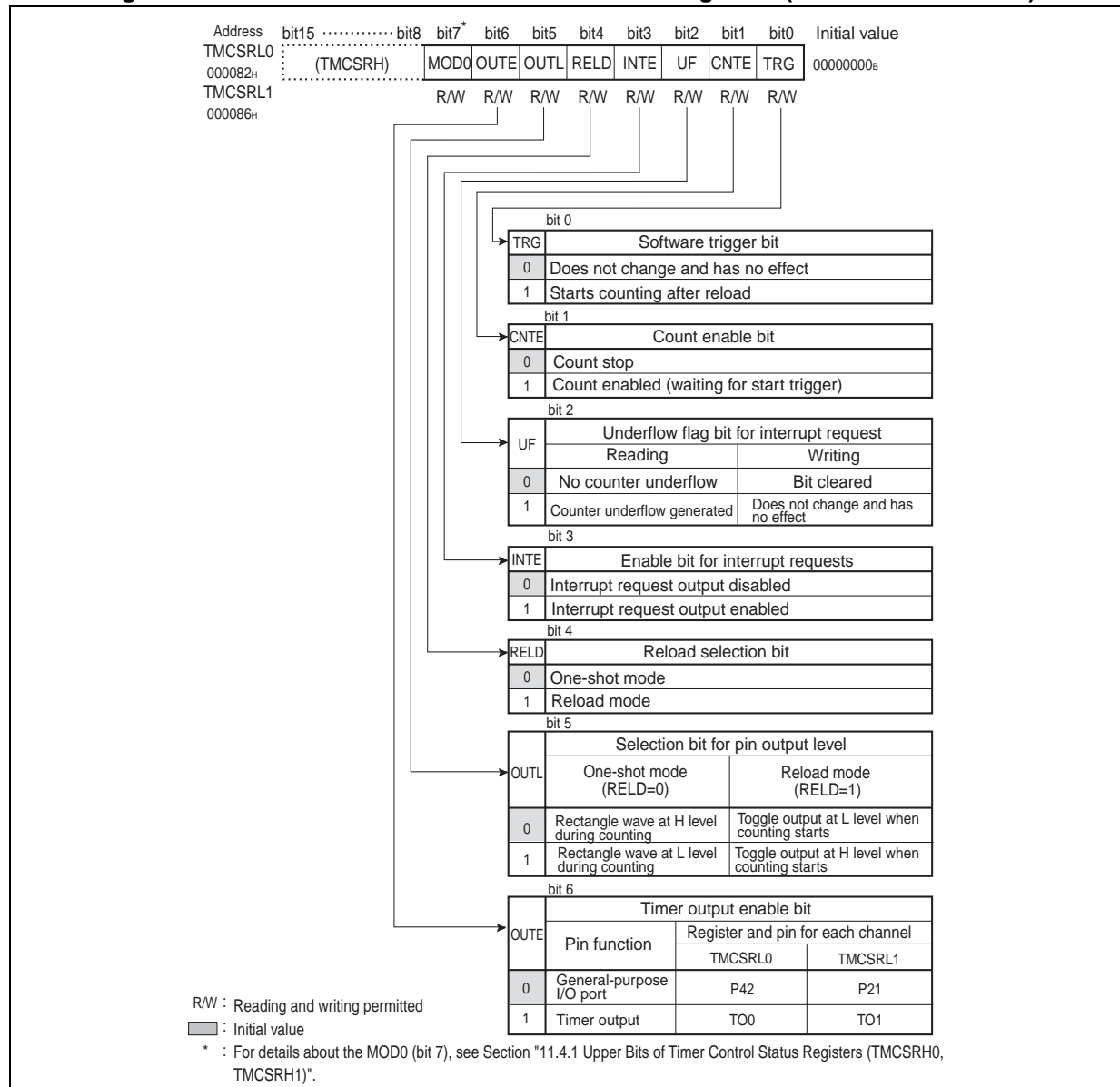
## MB90820B Series

## 12.4.2 Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1)

Bit 7 of the timer control status registers (TMCSRL0/TMCSRL1), which is part of the lower bits, is used to set the operating conditions of the 16-bit reload timer, enable or disable count operation, control interrupts, and check the state of operation.

### ■ Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1)

Figure 12.4-3 Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1)



**Table 12.4-2 Function of the Lower Bits in the Timer Control Status Registers (TMCSRL0/TMCSRL1)**

Bit name		Function
bit6	OUTE: Timer output enable bit	<ul style="list-style-type: none"> <li>Enables or disables output via the timer output pin. When this bit is "0", the pin is used as a general-purpose I/O port; when this bit is "1", the pin is used as a timer output pin.</li> <li>The waveform output from the timer output pin becomes toggle waveform output in reload mode. In one-shot mode, a rectangular wave is output, which indicates that counting is in progress.</li> </ul>
bit5	OUTL: Selection bit for pin output level	<ul style="list-style-type: none"> <li>Bit used to select the output level to the timer output pin.</li> <li>Toggle this bit to "0" or "1" to reverse the pin output level.</li> </ul>
bit4	RELD: Reload selection bit	<ul style="list-style-type: none"> <li>Enables reload operation. Reload mode is entered when this bit is set to "1". If underflow occurs, 16-bit reload register data is loaded into the 16-bit counter to continue count operation. One-shot mode is entered when this bit is set to "0". If underflow occurs, count operation will stop.</li> </ul>
bit3	INTE: Enable bit for interrupt requests	<ul style="list-style-type: none"> <li>Enables or disables interrupt requests to the CPU.</li> <li>When this bit and the flag bit for interrupt request (UF) are set to "1", an interrupt request is output.</li> </ul>
bit2	UF: Underflow flag bit for interrupt request	<ul style="list-style-type: none"> <li>Set to "1" if 16-bit counter underflow occurs. Cleared by writing "0". Writing "1" has no effect.</li> <li>Also cleared at EI<sup>2</sup>OS startup.</li> </ul>
bit1	CNTE: Count enable bit	<ul style="list-style-type: none"> <li>Enables or disables count operation. When this bit is set to "1", start trigger wait state is entered. As soon as the start trigger occurs, the actual counting will begin.</li> </ul>
bit0	TRG: Software trigger bit	<ul style="list-style-type: none"> <li>Used to start the interval timer function or counter function by software. Set this bit to "1" to activate the software trigger and load the 16-bit reload register value into the counter to start counting. Writing "0" has no effect.</li> <li>When CNTE=1, trigger input is always enabled by this but regardless of the mode.</li> <li>Reading always returns "0".</li> </ul>

**MB90820B Series****12.4.3 16-Bit Timer Registers (TMR0/TMR1)**

The 16-bit timer registers (TMR0/TMR1) are used to continuously read the current count value of the 16-bit down counter.

### ■ 16-bit Timer Registers (TMR0/TMR1)

Figure 12.4-4 shows the bit configuration of the 16-bit timer registers (TMR0/TMR1).

**Figure 12.4-4 Bit Configuration of 16-bit Timer Registers (TMR0/TMR1)**

Address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	Initial value
TMR0: 000053 <sub>H</sub>	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX <sub>B</sub>
TMR1: 000057 <sub>H</sub>	R	R	R	R	R	R	R	R	
Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	Initial value
TMR0: 000052 <sub>H</sub>	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX <sub>B</sub>
TMR1: 000056 <sub>H</sub>	R	R	R	R	R	R	R	R	

R: Read only  
X: Undefined value

These registers are used to read the current count value of the 16-bit down counter. When count operation is allowed (TMCSR0/TMCSR1: CNTE=1) to start counting, the value written to the 16-bit reload register is loaded into these registers to start the countdown. In counter stop mode (CNTE=0 for TMCSR0/TMCSR1), the register value is retained.

#### Note:

These registers may be read in counter operation mode by using a word transfer instruction (MOVW A, 003A<sub>H</sub>, etc).

The 16-bit timer registers (TMR0/TMR1) are read-only registers and assigned the same address as the write-only 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1). Therefore, writing does not affect TMR register values, though writing is performed to TMRDL0/TMRDL1 and TMRDH0/TMRDH1.

Be sure to perform word-access to the TMR0/TMR1 registers.

## 12.4.4 16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1)

The 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) are used to set a reload value to the 16-bit down counter. The value written to these registers is loaded into the down counter for countdown.

### ■ 16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1)

Figure 12.4-5 shows the bit configuration of the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1).

**Figure 12.4-5 Bit Configuration of 16-bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1)**

Address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	Initial value
TMRDH0: 000085 <sub>H</sub> TMRDH1: 000089 <sub>H</sub>	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX <sub>B</sub>
	W	W	W	W	W	W	W	W	
Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	Initial value
TMRDL0: 000084 <sub>H</sub> TMRDL1: 000088 <sub>H</sub>	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX <sub>B</sub>
	W	W	W	W	W	W	W	W	

R: Read only  
X: Undefined value

Regardless of the 16-bit reload timer operation mode, if counter operation is prohibited (TMCSR0/TMCSR1: CNTE=0), these registers are set to the initial value of the counter. When counter operation is allowed (TMCSR0/TMCSR1: CNTE=1) to start the counter, the countdown starts from the value written to these registers.

The value set in the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) is reloaded into the counter in reload mode if underflow occurs, then the countdown continues. In one-shot mode, the counter stops at "FFFF<sub>H</sub>" if underflow occurs.

Writing to the registers is always performed in counter stop mode (TMCSR0/TMCSR1: CNTE=0). Write operations always using a word transfer instruction (MOVW 003AH, A).

The 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) are functionally write-only registers that are allocated under the same address as the read-only 16-bit timer registers (TMR0/TMR1). Therefore, the value read is the value of TMR0/TMR1. Consequently, an instruction such as INC/DEC for read-modify-write (RMW) operation cannot be used.

## MB90820B Series

### 12.5 Interrupts of 16-Bit Reload Timer

The 16-bit reload timer may generate an interrupt due to 16-bit down counter underflow. The timer also supports the extended intelligent I/O service (EI<sup>2</sup>OS).

#### ■ Interrupts Generated by 16-bit Reload Timer

Table 12.5-1 lists the interrupt control bits and interrupt sources of the 16-bit reload timer.

**Table 12.5-1 Interrupt Control Bits and Interrupt Sources of 16-bit Reload Timer**

Interrupt source	Lower bits of timer control status register (TMCSRL0/TMCSRL1)		
	Interrupt flag bit	Interrupt enable bit	Clearance of interrupt flag
Underflow of 16-bit down counter (TMR0/1) ("0000 <sub>H</sub> " --> "FFFF <sub>H</sub> ")	UF	INTE	<ul style="list-style-type: none"> <li>Writing "0" to the UF bit</li> <li>Resetting</li> <li>Starting EI<sup>2</sup>OS</li> </ul>

If the interrupt source listed in Table 12.5-1 is generated, the interrupt flag bit of the 16-bit reload timer is set to "1". If the interrupt enable bit of the 16-bit reload timer is "1" when the interrupt flag bit is set to "1", the 16-bit reload timer outputs an interrupt request to the interrupt controller.

#### ■ Interrupts of 16-bit Reload Timer and EI<sup>2</sup>OS

Table 12.5-2 lists the interrupts of the 16-bit reload timer and their relationship to EI<sup>2</sup>OS.

**Table 12.5-2 Interrupts of 16-bit Reload Timer and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower bits	Upper bits	Bank	
16-Bit Reload Timer 0	#30(1E <sub>H</sub> )	ICR09	0000B9 <sub>H</sub>	FFFF84 <sub>H</sub>	FFFF85 <sub>H</sub>	FFFF86 <sub>H</sub>	*
16-Bit Reload Timer 1	#18(12 <sub>H</sub> )	ICR03	0000B3 <sub>H</sub>	FFFFB4 <sub>H</sub>	FFFFB5 <sub>H</sub>	FFFFB6 <sub>H</sub>	*

\*: Available when not using interrupt sources sharing ICR03, ICR09, or the interrupt vector.

#### ■ EI<sup>2</sup>OS Function of 16-bit Reload Timer

The 16-bit reload timer has a circuit supporting EI<sup>2</sup>OS. Therefore, a counter underflow will start EI<sup>2</sup>OS. Note that EI<sup>2</sup>OS is only available when no other peripheral function that shares the interrupt control register (ICR) uses an interrupt. To use EI<sup>2</sup>OS by 16-bit reload timer 0, interrupts of waveform generator must be prohibited. To use EI<sup>2</sup>OS by 16-bit reload timer 1, interrupts of output compare 2 must be prohibited.

## 12.6 Operation of 16-Bit Reload Timer

This section describes how to set the 16-bit reload timer and counter operation state.

### ■ 16-bit Reload Timer Settings

#### ● Setting internal clock mode

To operate the interval timer, the settings listed in Figure 12.6-1 are required.

Figure 12.6-1 Internal Clock Mode Settings

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TMCSR	—	—	—	FSEL	CSL1	CSL0	MOD 2	MOD 1	MOD 0	OUTE	OUTL	RELD	INTE	UF	CNTE	TRG
				⊙	Other than "11"		⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	1	⊙
TMRD	Setting of the initial counter value (reload value)															
⊙: Bit used																
1: Set to "1".																

#### ● Setting event count mode

To operate the event counter, the settings listed in Figure 12.6-2 are required.

Figure 12.6-2 Event Count Mode Settings

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TMCSR	—	—	—	FSEL	CSL1	CSL0	MOD2	MOD1	MOD0	OUTE	OUTL	RELD	INTE	UF	CNTE	TRG
				⊙	1	1	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	1	⊙
TMRD	Setting of the initial counter value (reload value)															
DDR5	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div> <div>△</div>															
DDR0	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div> <div>△</div>															

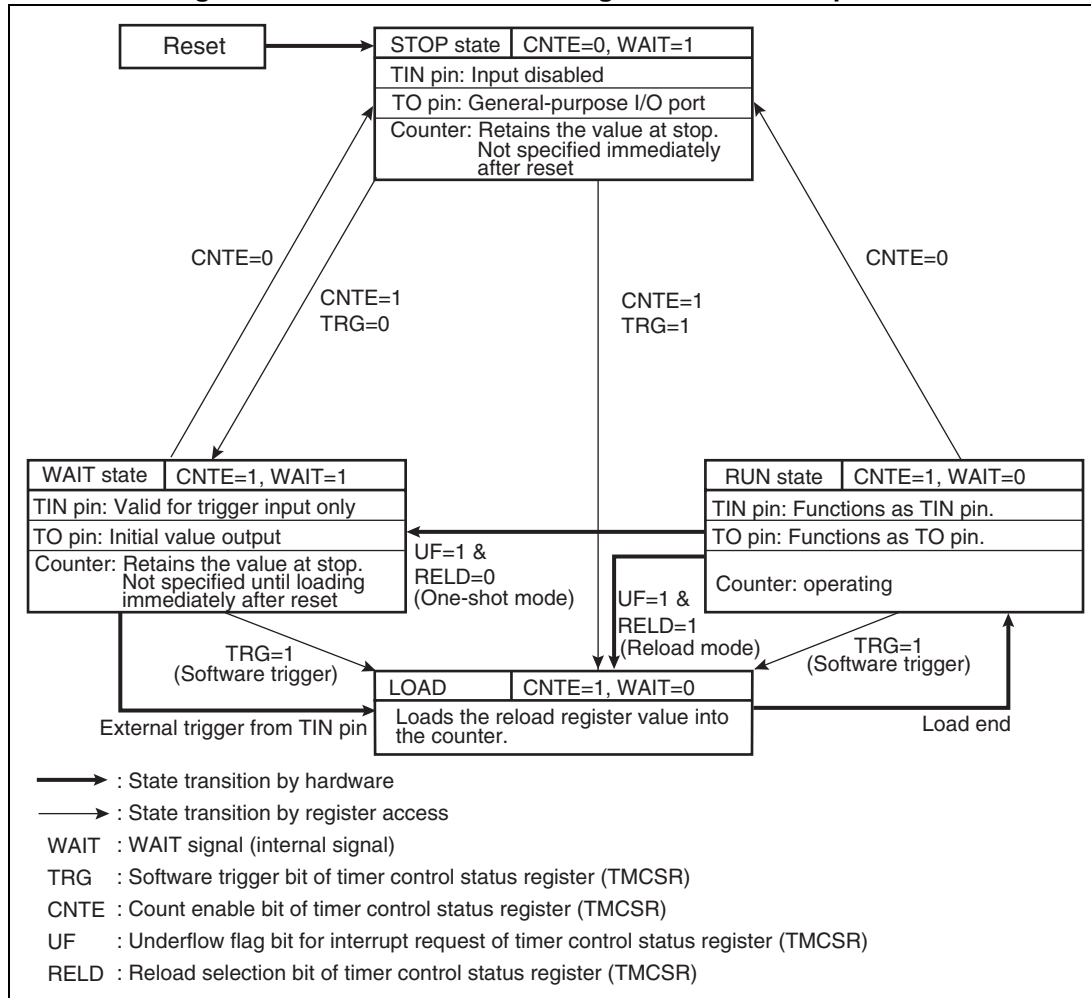
⊙: Bit used  
1: Set to "1".  
△: Set the bit corresponding to the pin used to "0".

# MB90820B Series

## ■ States of Counter Operation

The counter's operation state is determined by the CNTE bit of the timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1) and the internal WAIT signal. States that can be set include the stop state (STOP state), start trigger wait state (WAIT state), and operation state (RUN state). Figure 12.6-3 shows a state transition diagram for the counter.

**Figure 12.6-3 State Transition Diagram of Counter Operation**





## 12.6.1 Internal Clock Mode (Reload Mode)

The counter operates in sync with the internal count clock to count down the 16-bit counter and generate an interrupt request to the CPU in case of counter underflow. The counter also outputs a toggle waveform from the timer output pin.

### ■ Operation in Internal Clock Mode (Reload Mode)

When count operation is allowed (TMCSR0/TMCSR1: CNTE=1) and the timer is started by the software trigger bit (TMCSR: TRG) or external trigger, counter operation will start by reloading the data of the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) into the 16-bit down counter. When both the count enable bit and software trigger bit are set to "1", counting will begin as soon as the counter is enabled.

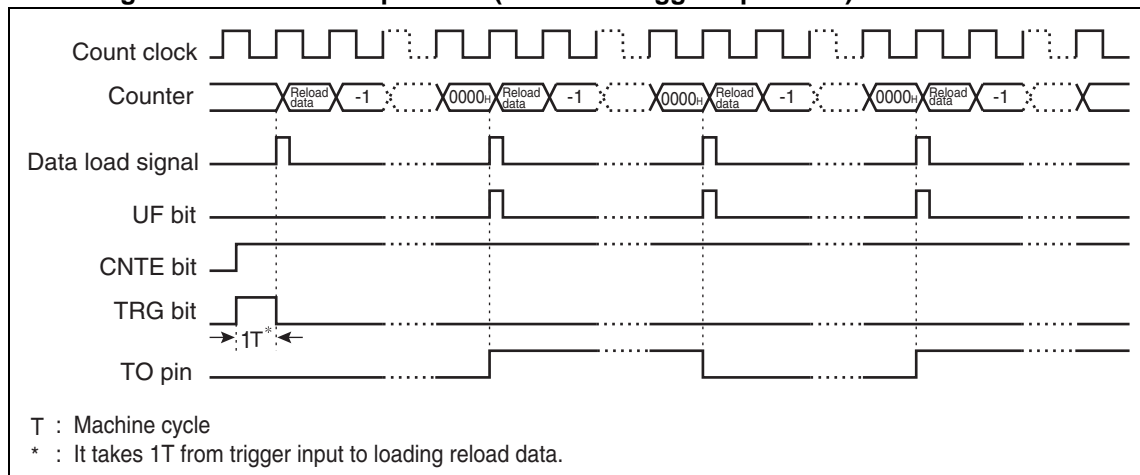
If the 16-bit counter value causes an underflow ("0000<sub>H</sub>" --> "FFFF<sub>H</sub>"), the value of the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) is reloaded into the 16-bit counter to continue counting. Note that if the underflow flag bit for interrupt request (UF) and enable bit for interrupt request (INTE) are set to "1", an interrupt request is generated.

The TO pin outputs a toggle waveform that is reversed at every underflow.

### ● Software trigger operation

When the TRG bit of the timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1) is set to "1", the counter starts operation. Figure 12.6-4 shows the software trigger operation in reload mode.

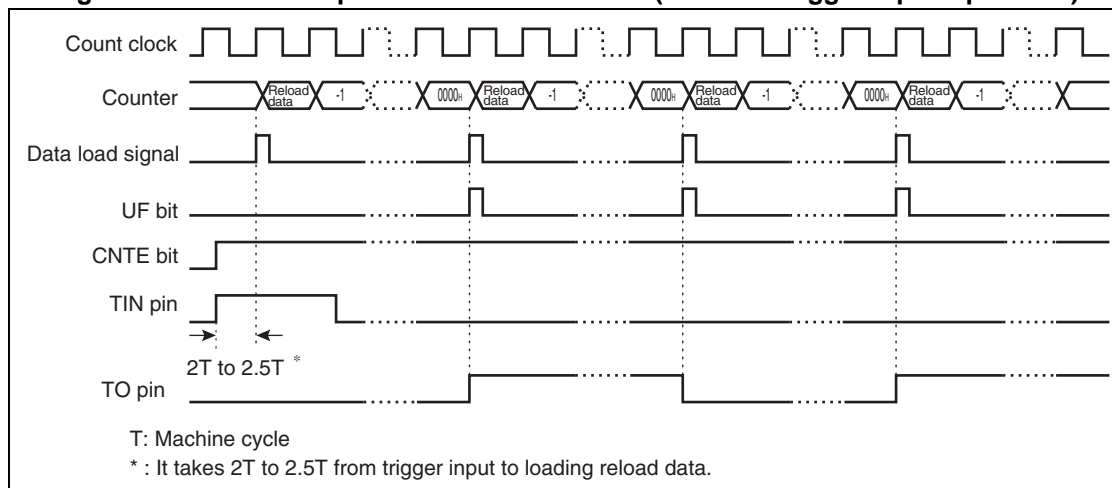
**Figure 12.6-4 Count Operation (Software Trigger Operation) in Reload Mode**



● External trigger input operation

When a valid edge (leading, trailing, or both edges can be selected) is input to the TIN pin, the count will start operation. Figure 12.6-5 shows the external trigger operation in reload mode.

**Figure 12.6-5 Count Operation in Reload Mode (External Trigger Input Operation)**



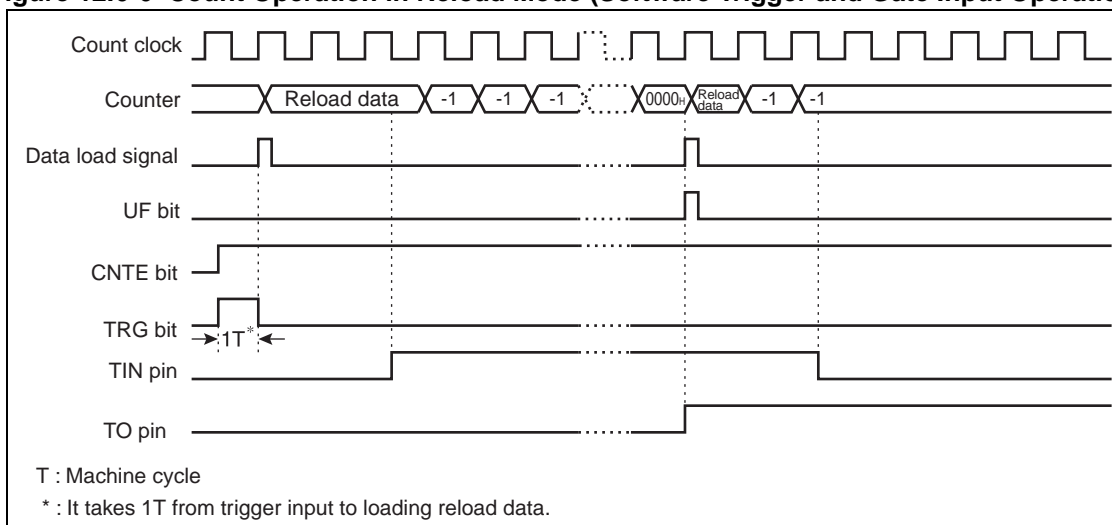
Note:

The width of trigger pulses input to the TIN pin must be  $2/\phi$  ( $\phi$ : machine clock) or more.

● Gate input operation

As soon as a valid level ("H" level or "L" level can be selected) is input to the TIN pin, the count will start operation. Figure 12.6-6 shows the gate input operation in reload mode.

**Figure 12.6-6 Count Operation in Reload Mode (Software Trigger and Gate Input Operation)**



---

Note:

The width of trigger pulses input to TIN pin must be  $2/\phi$  ( $\phi$ : machine clock) or more.

---

## MB90820B Series

### 12.6.2 Internal Clock Mode (One-Shot Mode)

The counter is in synchronization with the internal count clock in this mode to count down the 16-bit counter and generate an interrupt request to the CPU at counter underflow. It also outputs a square wave from the TO0/TO1 pin to indicate that counting is in progress.

#### ■ Operation of Internal Clock Mode (One-shot Mode)

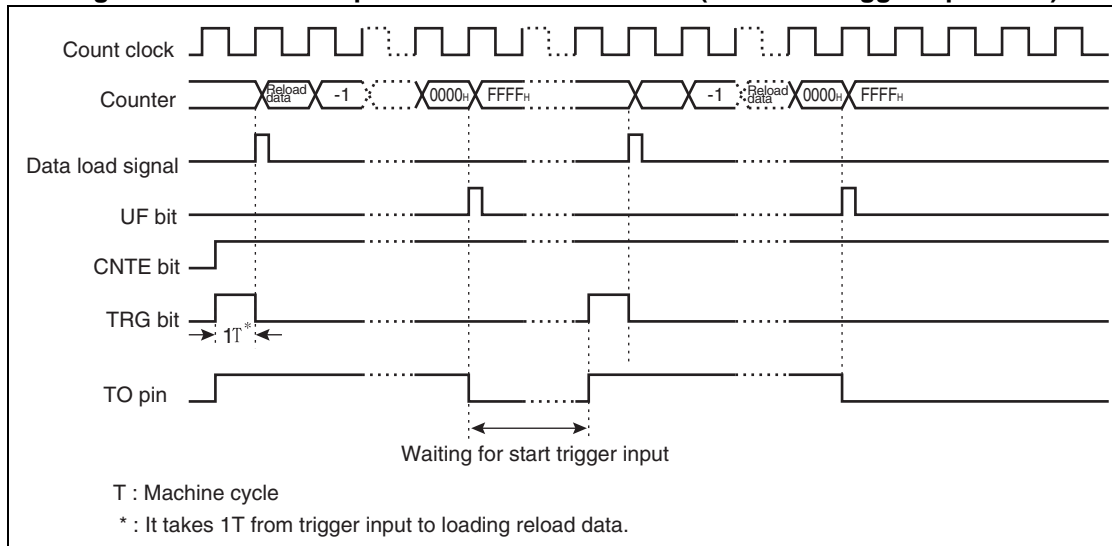
When count operation is allowed (TMCSR0/1: CNTE=1) and the timer is started by the software trigger bit (TMCSR0/TMCSR1: TRG) or external trigger, count operation will start. When both the count enable bit and software trigger bit are set to "1", counting will start at the same time counting becomes enabled. If the 16-bit counter value causes an underflow ("0000<sub>H</sub>" → "FFFF<sub>H</sub>"), the counter stops at "FFFF<sub>H</sub>", and the underflow flag bit for interrupt requests (UF) is set to "1". If the enable bit for interrupt request (INTE) is set to "1", an interrupt request is generated.

The TO pin outputs a square wave to indicate that counting is in progress.

#### ● Software trigger operation

The count will start as soon as the TRG bit of the timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1) is set to "1". Figure 12.6-7 shows the software trigger operation in one-shot mode.

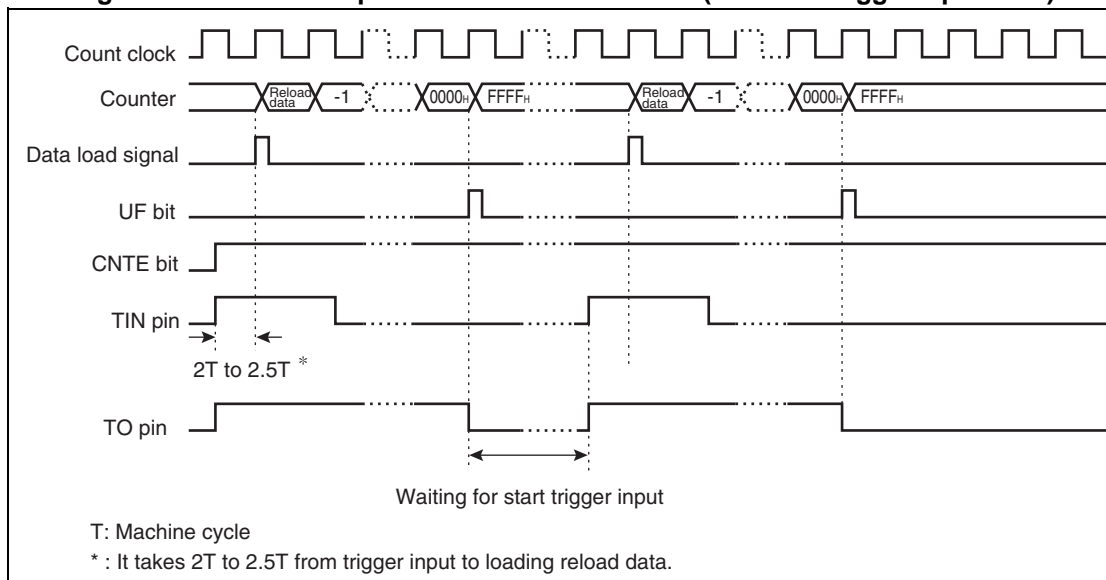
**Figure 12.6-7 Count Operation in One-shot Mode (Software Trigger Operation)**



● External trigger input operation

When a valid edge (leading, trailing, or both edges can be selected) is input to the TIN0/TIN1 pins, the count will start operation. Figure 12.6-8 shows the external trigger operation in one-shot mode.

**Figure 12.6-8 Count Operation in One-shot Mode (External Trigger Operation)**



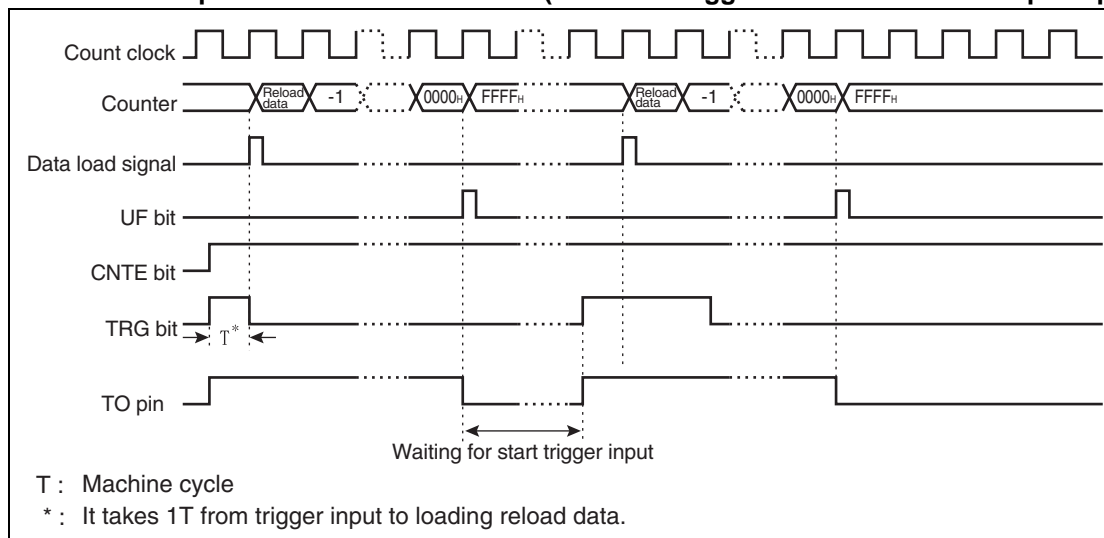
Note:

The width of trigger pulses input to the TIN pin must be  $2/\phi$  ( $\phi$ : machine clock) or more.

● Gate input operation

When a valid level ("H" or "L" level can be selected) is input to the TIN pin, the count starts operation.  
Figure 12.6-9 shows the gate input operation in one-shot mode.

**Figure 12.6-9 Count Operation in One-shot Mode (Software Trigger and External Gate Input Operation)**



**Note:**

The width for trigger pulse input to the TIN pin must be  $2/\phi$  ( $\phi$ : machine clock) or more.

### 12.6.3 Event Count Mode

In this mode, the counter counts input edges from the TIN pin to count down the 16-bit counter and generate an interrupt request to the CPU when a counter underflow occurs. The TO0/1 pin can output either a toggle waveform or a square wave.

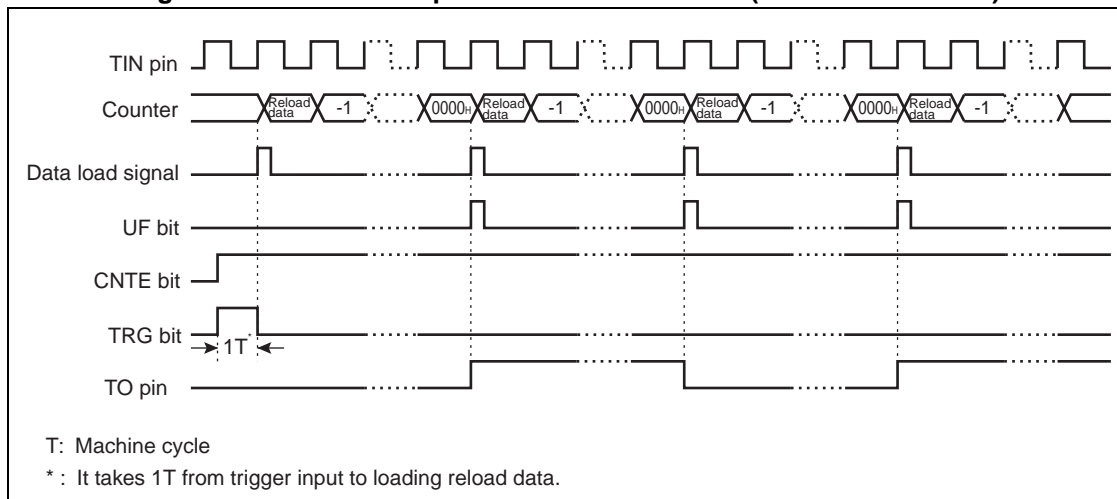
#### ■ Event Count Mode

When count operation is allowed (TMCSR0/TMCSR1: CNTE=1) to start the counter (TMCSR0/TMCSR1: TRG=1), data from the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) is loaded into the counter for a countdown whenever a valid edge (leading, trailing, or both edges can be selected) of pulses (external count clock) input to the TIN0/TIN1 pin is detected. When both the count enable bit and software trigger bit are set to "1", counting will start as soon as counting becomes enabled.

Operation in reload mode

If the counter value has an underflow ("0000<sub>H</sub>" --> "FFFF<sub>H</sub>"), data from the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) is loaded into the counter to continue counting. In this case, an interrupt request is issued when the underflow flag bit for interrupt requests (UF) and enable bit for interrupt requests (TMCSR0/TMCSR1: INTE) are both set to "1". The TO0/TO1 pin outputs a toggle waveform, which is reversed at every occurrence of underflow. Figure 12.6-10 shows the counting operation in reload mode.

**Figure 12.6-10 Count Operation in Reload Mode (Event Count Mode)**

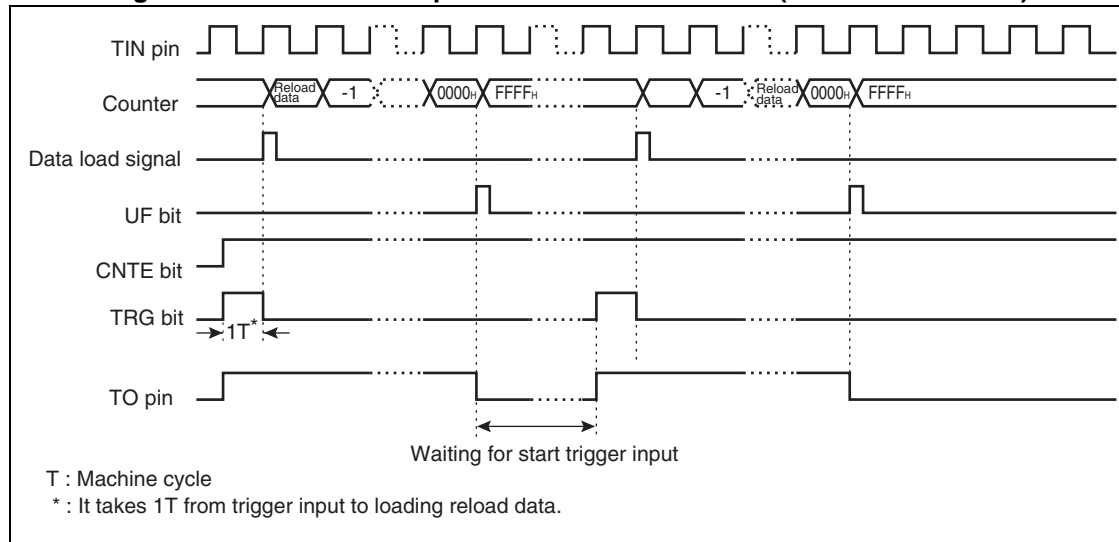


Note:

Both the "H" width and "L" width of clock input to TIN pin must be  $4/\phi$  ( $\phi$ : machine clock) or more.

If the counter value causes an underflow ("0000<sub>H</sub>" --> "FFFF<sub>H</sub>"), the 16-bit counter stops at "FFFF<sub>H</sub>". In this case, the underflow interrupt request flag bit (UF) is set to "1". If the interrupt request enable bit (INTE) is also set to "1", an interrupt request is generated. The TO0/TO1 pin outputs a square wave that indicates counting in progress. Figure 12.6-11 shows the counter operation in one-shot mode.

**Figure 12.6-11 Counter Operation in One-shot Mode (Event Count Mode)**



Note:

Both the "H" width and "L" width for the clock input to TIN pin must be  $4/\phi$  (machine clock) or more.



## 12.7 Notes on Using the 16-Bit Reload Timer

---

This section provides notes on using the 16-bit reload timer.

---

### ■ Notes on Using the 16-bit Reload Timer

#### ● Notes on setup by program

Writing a value to the 16-bit reload registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) must be performed in counter operation stop (TMCSR0/TMCSR1: CNTE=0) mode. Reading of the 16-bit timer registers (TMR0/TMR1) may be performed while the counter is in operation, but a word transfer instruction (MOVW A, dir, etc) must be used in this case.

The contents of the FSEL/CSL1/CSL0/MOD2/MOD1/MOD0 bits in the timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1) can only be changed in counter operation stop mode (TMCSRL0/TMCSRL1: CNTE=0).

#### ● Notes on interrupts

If the UF bit of the timer control status registers (TMCSRL0/TMCSRL1, TMCSRH0/TMCSRH1) is set to "1" in interrupt request enabled state (TMCSRL0/TMCSRL1:INTE=1), return from interrupt handling cannot be performed. Ensure that the UF bit is always cleared.

Since the 16-bit reload timer, waveform generator and output compare 2 share the same interrupt vector, interrupt sources must be checked in the interrupt-handling routine to enable using interrupts.

If the 16-bit reload timer uses EI<sup>2</sup>OS, "shared resource interrupts must be disabled".

# **CHAPTER 13**

---

## ***PWC Timer***

**This chapter explains the activation and operations of the PWC timer.**

- 13.1 Overview of the PWC Timer
- 13.2 Block Diagram of the PWC Timer
- 13.3 PWC Timer Pins
- 13.4 PWC Timer Registers
- 13.5 PWC Timer Interrupts
- 13.6 Operation of the PWC Timer
- 13.7 Usage Notes on the PWC Timer

## 13.1 Overview of the PWC Timer

---

The PWC timer (pulse-width measurement) is the multi-functional 16-bit up counter with the reload function and also has a function that calculates the pulse width of the input signal.

The PWC timer consists of a 16-bit counter, an input pulse divider, a division rate control register, a count input pin, a pulse output pin, and a 16-bit control register.

---

### ■ PWC Timer

The MB90820B series contains two PWC timer channels. The PWC timer has the following characteristics:

#### ● Timer function

- Generates an interrupt request at the specified time interval.
- Outputs the pulse signal that is synchronized with the timer period.
- Selects the counter clock from three internal clocks.

#### ● Pulse-width measurement function

- Measures the time between external pulse input events.
- Selects the counter clock from three internal clocks.
- Count mode
  - H pulse width (rising edge to falling edge) / L pulse width (falling edge to rising edge)
  - Rising edge period (rising edge to falling edge) / falling edge period (falling edge to rising edge)
  - Intermediate edge count (rising or falling edge to falling or rising edge)
- Uses the 8-bit input divider to divide the input pulse by  $2^2$ ,  $2^4$ ,  $2^6$ , and  $2^8$  to enable period measurement.
- Generates an interrupt request at completion of count.
- Selects single count or continuous count.

### ■ PWC Timer Operation

This block is a multi-functional timer that is based on the 16-bit up-count timer and contains a count input pin and an 8-bit input divider. The block has two main functions, a timer function and a pulse-width measurement function, both of which enable the selection for two types of count clocks.

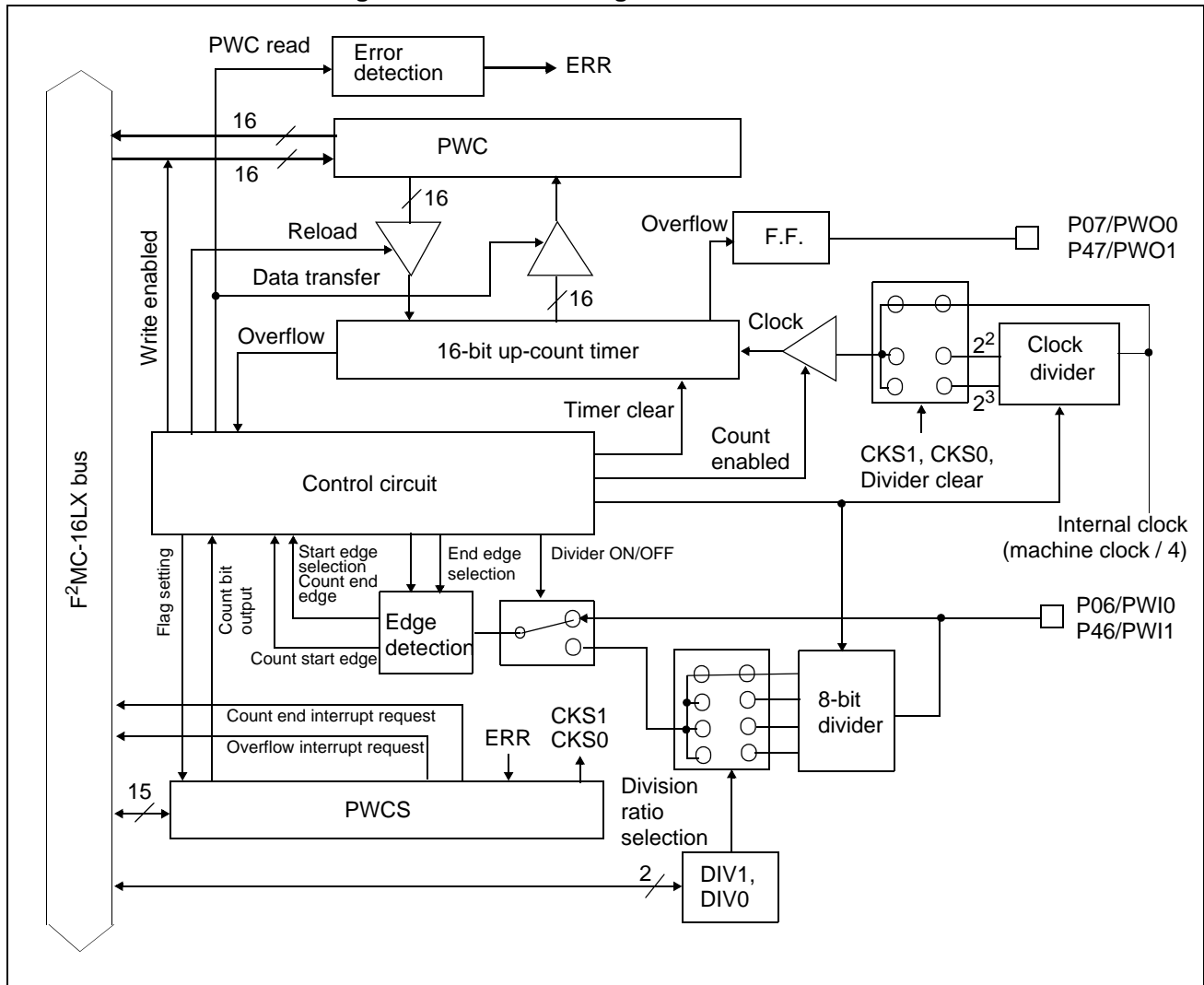
## MB90820B Series

### 13.2 Block Diagram of the PWC Timer

Figure 13.2-1 shows the PWC timer block diagram.

#### ■ Block Diagram of the PWC Timer

Figure 13.2-1 Block diagram of the PWC timer



13.3 PWC Timer Pins

This section describes the pins of the PWC timer and provides a pin block diagram.

PWC Timer Pins

The pins of the PWC timer are shared with the general-purpose I/O ports. Table 13.3-1 lists the functions of the pins, I/O format, and settings required to use the PWC timer.

Table 13.3-1 16-bit PWC timer pins

Pin name	Pin function	I/O format	Pull-up option	Standby control	Settings required for pins
P06/PWI0	Port 0 input-output / timer input	CMOS output / CMOS input	Selectable	Available	Setting for the input port (DDR0: bit 6 = 0)
P07/PWO0	Port 0 input-output / timer output				Setting for timer enable (PWCSL0: MOD2 to MOD0 not equal "0")
P46/PWI1	Port 4 input-output / timer input	CMOS output / CMOS hysteresis input	Not provided		Setting for the input port (DDR4: bit 6 = 0)
P47/PWO1	Port 4 input-output / timer output				Setting for timer enable (PWCSL1: MOD2 to MOD0 not equal "0")

Block Diagram of the PWC Timer Pins

Figure 13.3-1 shows the block diagram of the PWC timer 0 input pin.

Figure 13.3-1 Block diagram of the PWC timer 0 input pin (PWI0)

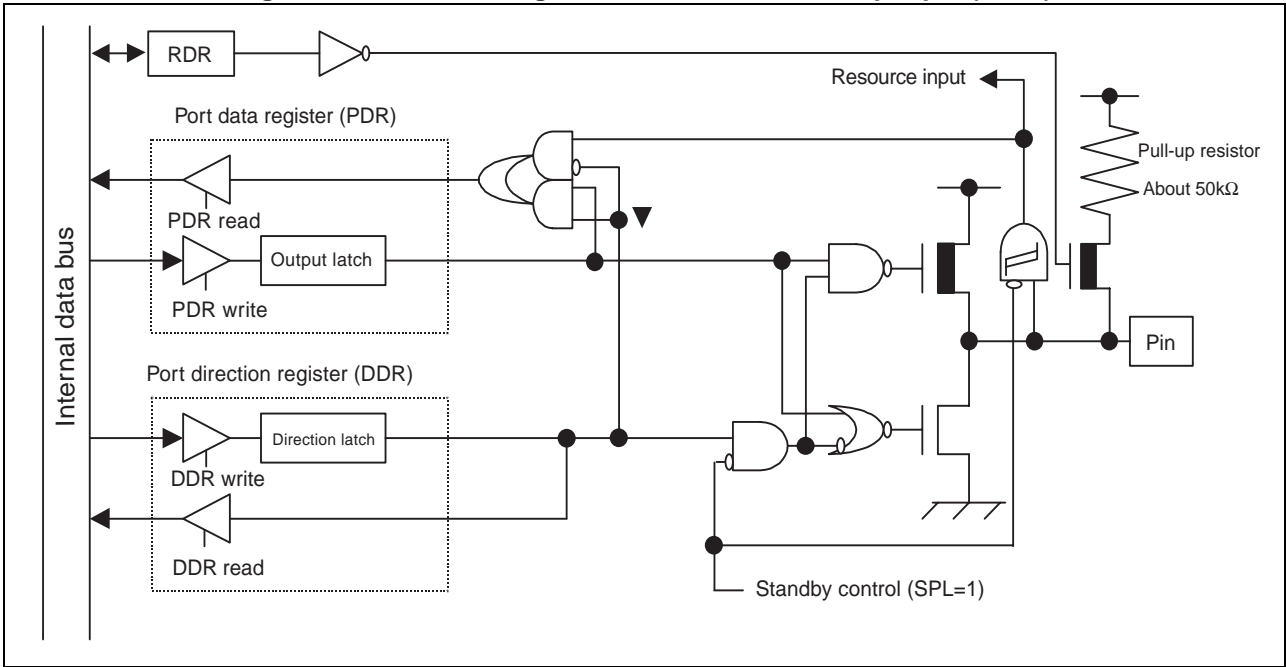


Figure 13.3-2 shows the block diagram of the PWC timer 0 output pin.

**Figure 13.3-2 Block diagram of the PWC timer 0 output pin (PWO0)**

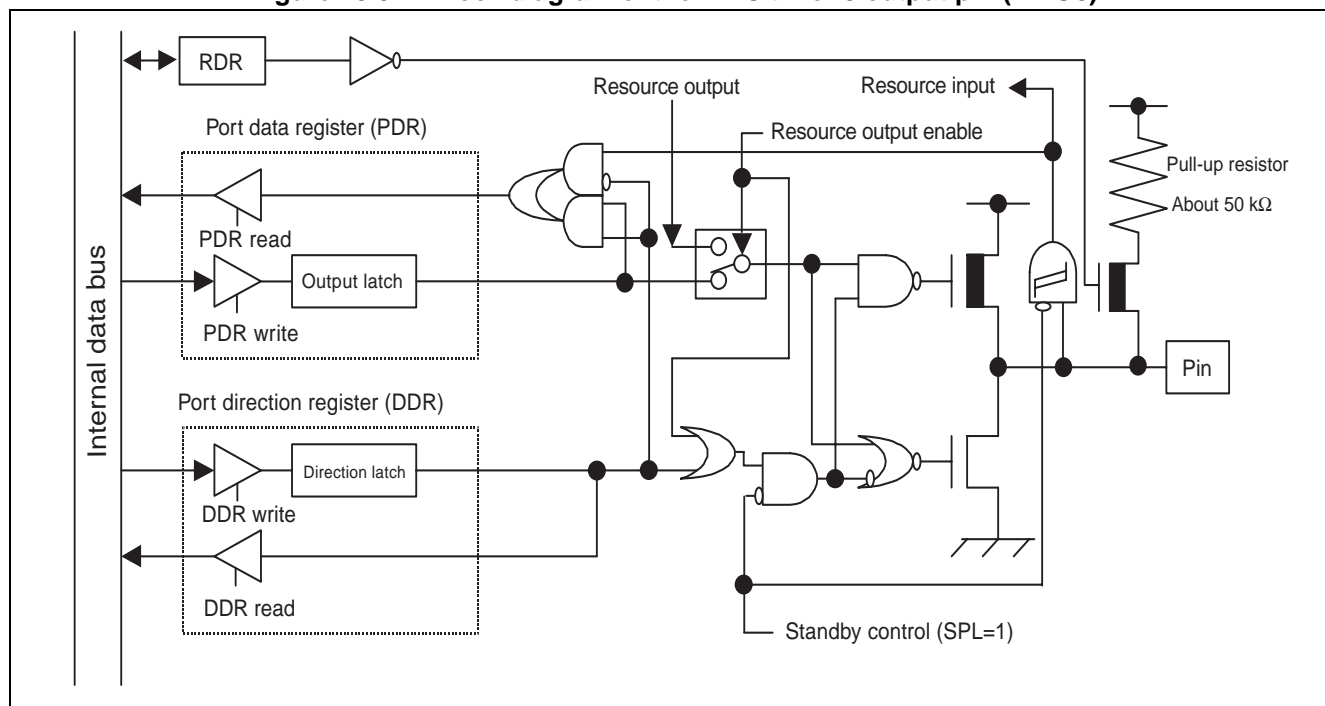


Figure 12.3-3 shows the block diagram of the PWC timer 1 input pin.

**Figure 13.3-3 Block diagram of the PWC timer 1 input pin (PWI1)**

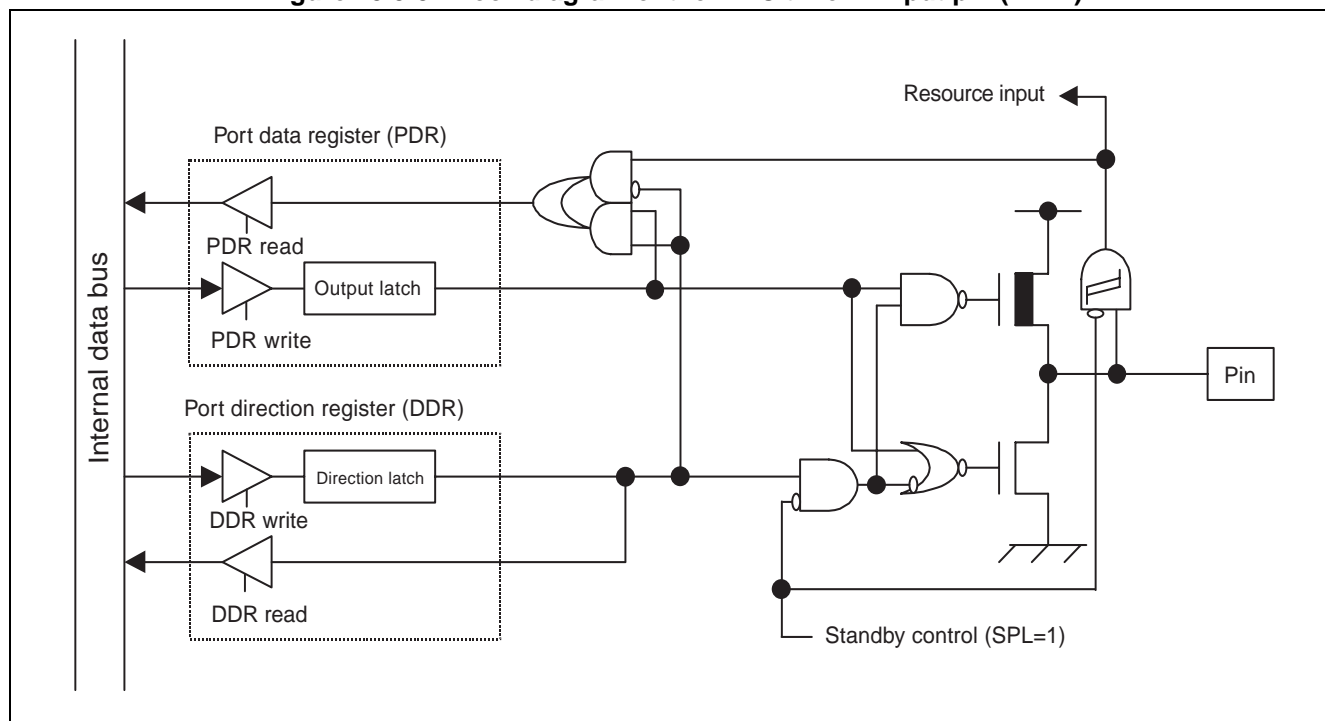
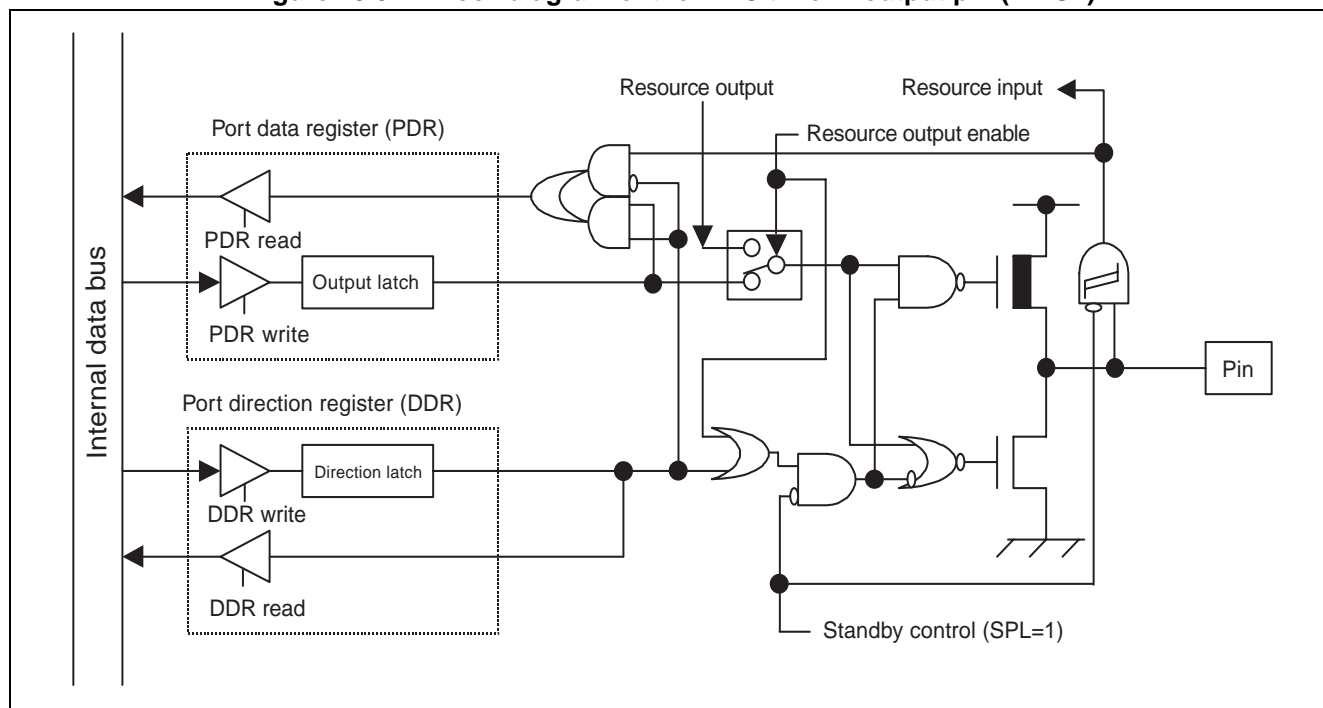


Figure 13.3-4 shows the block diagram of the PWC timer 1 output pin.

**Figure 13.3-4 Block diagram of the PWC timer 1 output pin (PW01)**



## MB90820B Series

### 13.4 PWC Timer Registers

Following are the PWC timer registers.

#### ■ PWC Timer Registers

Figure 13.4-1 PWC timer registers

<b>PWCSH0, PWCSH1</b> <b>PWC control status register (Upper)</b>								
Address: ch.0 0000C1 <sub>H</sub> ch.1 000029 <sub>H</sub> Read/write ⇒	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
	STRT	STOP	EDIR	EDIE	OVIR	OVIE	ERR	POUT
	R/W	R/W	R	R/W	R/W	R/W	R	R/W
Initial value:								
00000000 <sub>B</sub>								
<b>PWCSL0, PWCSL1</b> <b>PWC control status register (Lower)</b>								
Address: ch.0 0000C0 <sub>H</sub> ch.1 000028 <sub>H</sub> Read/write ⇒	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
	CKS1	CKS0	Reser ved	Reser ved	S/C	MOD2	MOD1	MOD0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value:								
00000000 <sub>B</sub>								
<b>PWC0, PWC1</b> <b>PWC data buffer register (Upper)</b>								
Address: ch.0 0000C3 <sub>H</sub> ch.1 00002B <sub>H</sub> Read/write ⇒	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
	PW15	PW14	PW13	PW12	PW11	PW10	PW09	PW08
	R/W	R/W	R	R/W	R/W	R/W	R	R/W
Initial value:								
xxxxxxxx <sub>B</sub>								
<b>PWC0, PWC1</b> <b>PWC data buffer register (Lower)</b>								
Address: ch.0 0000C2 <sub>H</sub> ch.1 00002A <sub>H</sub> Read/write ⇒	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
	PW07	PW06	PW05	PW04	PW03	PW02	PW01	PW00
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value:								
xxxxxxxx <sub>B</sub>								
<b>DIV0, DIV1</b> <b>Division ratio control register</b>								
Address: ch.0 0000C4 <sub>H</sub> ch.1 00002C <sub>H</sub> Read/write ⇒	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
	—	—	—	—	—	—	DIV1	DIV0
	—	—	—	—	—	—	R/W	R/W
Initial value:								
xxxxxx00 <sub>B</sub>								



## 13.4.1 PWC Control Status Register (PWCSH0/PWCSH1, PWCSL0/PWCSL1)

The PWC control status register (PWCSH0/PWCSH1, PWCSL0/PWCSL1) controls the PWC timer operation and reads the PWC timer state.

### ■ PWC Control Status Register, Upper Byte (PWCSH0/PWCSH1)

Figure 13.4-2 PWC control status register (PWCSH0/PWCSH1)

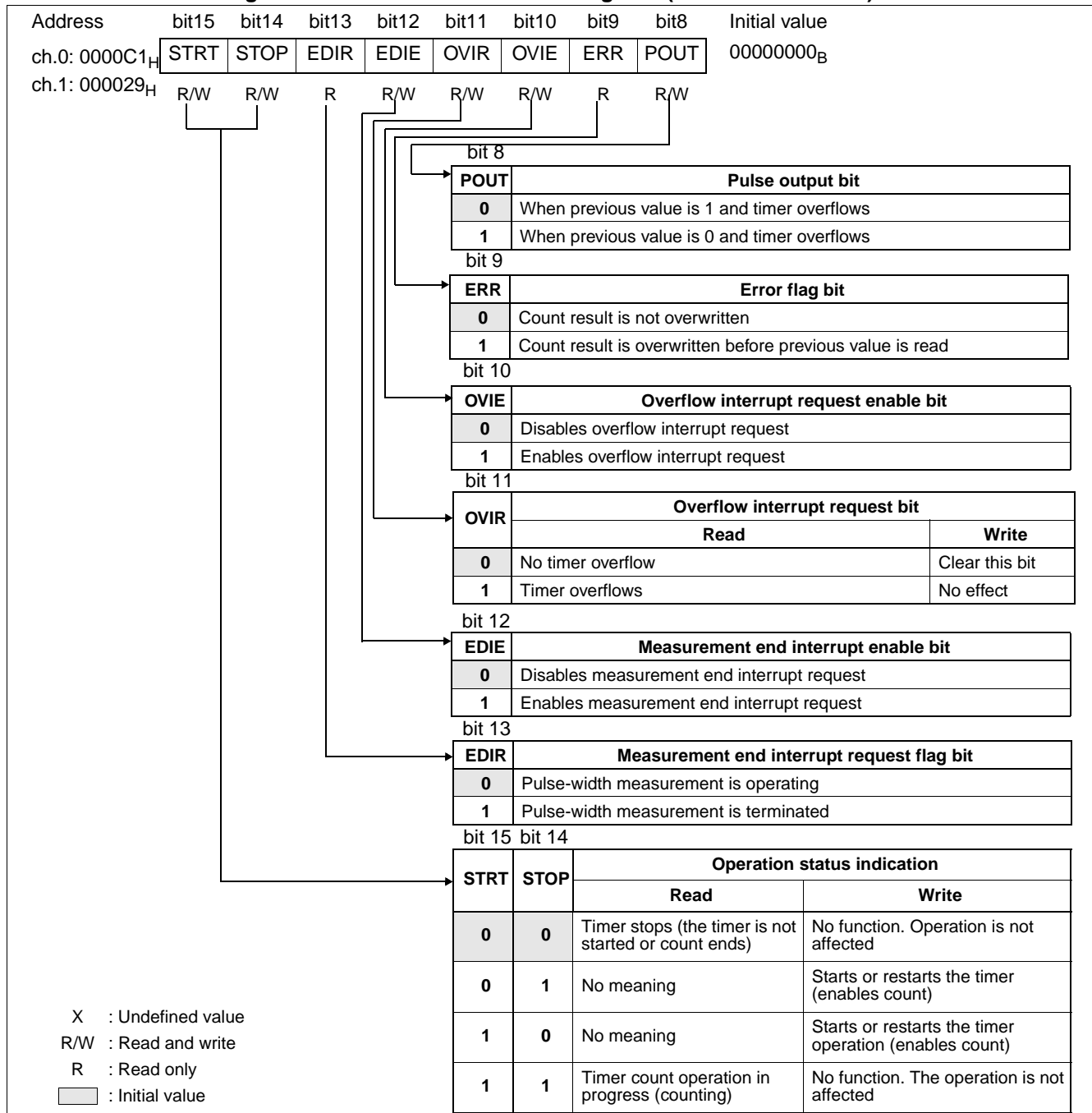


Table 13.4-1 PWC control status register (PWCSH0/PWCSH1) (1 / 2)

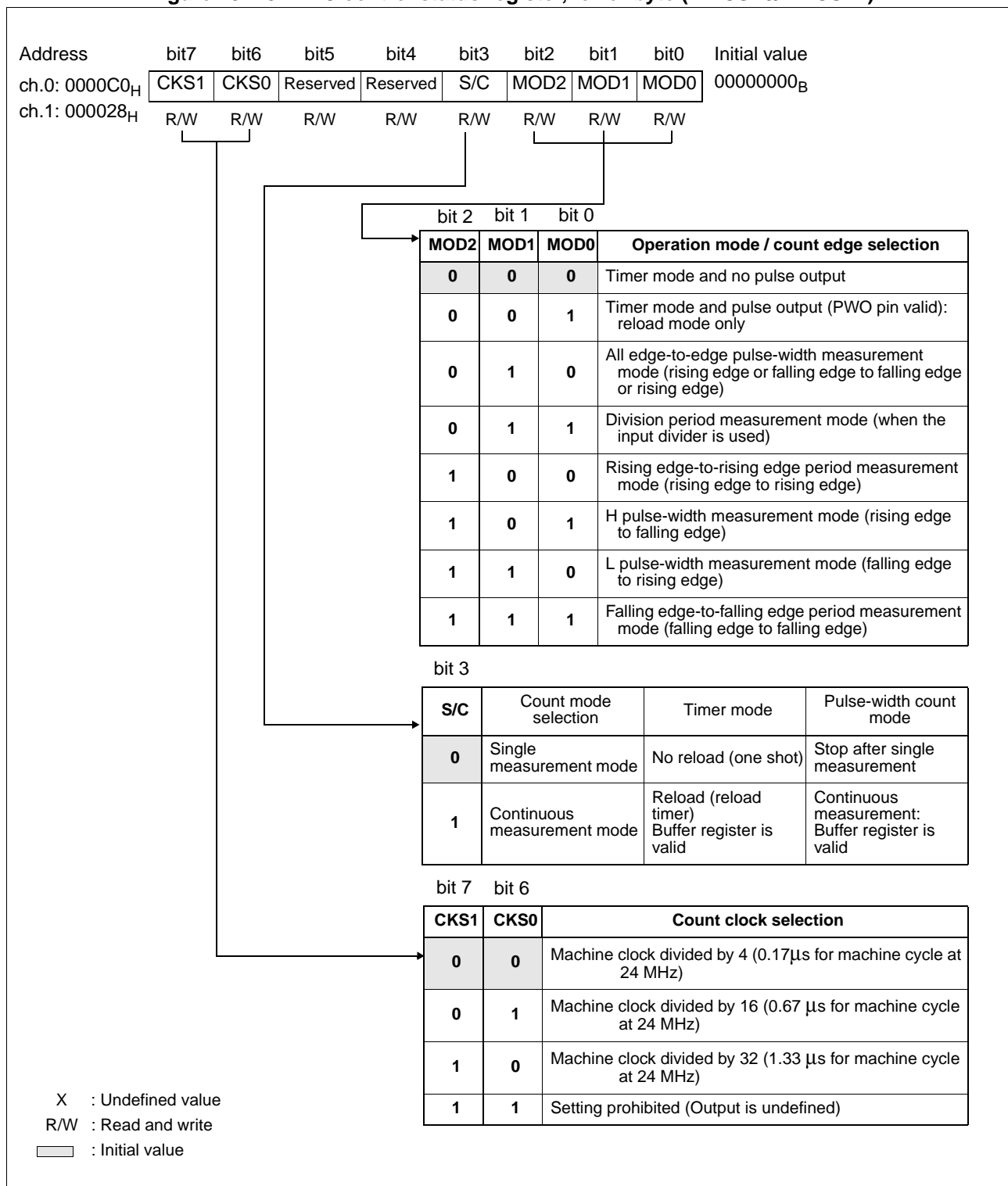
Bit name		Function
bit15, bit14	STRT, STOP: Start and Stop bits	<ul style="list-style-type: none"> <li>These bits are used to start, restart, and stop the 16-bit up-count timer.</li> <li>When these bits are read, the timer operation status is returned.</li> <li>These bits can be read and written. The meaning of bits depends on whether they are read or written.</li> <li>In read-modify-write operation, "11<sub>B</sub>" is always read.</li> <li>When the STRT and STOP bits are written to start and stop the timer, a bit manipulation instruction (such as bit clear instruction) can be used. However, when the operation status (which always indicates that the timer is operating, for example) is read, a bit manipulation instruction cannot be used.</li> </ul>
bit13	EDIR: Measurement end interrupt request flag bit	<ul style="list-style-type: none"> <li>This bit indicates that measurement terminated in pulse-width measurement mode.</li> <li>When pulse-width measurement terminates, the bit is set (PWC0/PWC1 contains the measurement result).</li> <li>This bit is cleared automatically when the measurement result in PWC data buffer register and PWC0/PWC1 are read.</li> <li>In timer mode, this bit is meaningless.</li> <li>This bit is read-only, writing this bit is meaningless.</li> </ul>
bit12	EDIE: Measurement end interrupt enable bit	<ul style="list-style-type: none"> <li>This bit is used to control a measurement termination interrupt request in pulse-width count mode.</li> <li>When this bit is "1" and EDIR bit is set to "1", the measurement end interrupt request will be generated to CPU.</li> <li>Always set "0" in timer mode.</li> </ul>
bit11	OVIR: Overflow interrupt request bit	<ul style="list-style-type: none"> <li>This bit is used to specify when the 16-bit up-count timer overflows. The operation affects all modes.</li> <li>When timer overflow occurs ("FFFF<sub>H</sub>" to "0000<sub>H</sub>"), the bit is set. Writing "0" will clear the bit. Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <p><b>Note:</b> In H/L pulse-width count mode, do not use this bit for pulse-width time measurement.</p>
bit10	OVIE: Overflow interrupt request enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable timer overflow interrupt request.</li> <li>When this bit is "1" and OVIR is set to "1", the overflow interrupt request will be generated to CPU.</li> </ul> <p><b>Note:</b> In the H/L pulse-width count mode, set this bit to "0".</p>

**Table 13.4-1 PWC control status register (PWCSH0/PWCSH1) (2 / 2)**

Bit name		Function
bit9	ERR: Error flag bit	<ul style="list-style-type: none"> <li>This bit is used to execute a continuous count in the pulse-width count mode. This flag indicates that the next count has been completed before the previous count result is read from PWC0/PWC1 register. If this state occurs, PWC0/PWC1 register is overwritten by new count result and the previous result is lost. The count operation continues regardless of the value for this bit.</li> <li>The bit is read-only. Writing to this bit is meaningless.</li> <li>When the count result that has not been read is overwritten by the next result, the bit is set.</li> <li>This bit is cleared automatically when the measurement result in PWC data buffer register and PWC0/PWC1 are read.</li> </ul>
bit8	POUT: Pulse output bit	<ul style="list-style-type: none"> <li>When the 16-bit up-count timer overflows in timer mode, this bit is reversed.</li> <li>In the pulse-width count mode, this bit is meaningless.</li> <li>The bit can be read and written. However, the bit can be written only if the timer stops (both bit 15: STRT and bit 14: STOP are set to "0"). If the bit is written during timer operation (both bit 15: STRT and bit 14: STOP are set to "1"), the bit value remains unchanged.</li> <li>When the POUT value is "0" and the timer overflows in the range from "FFFF<sub>H</sub>" to "0000<sub>H</sub>" or the timer stops and "1" is written, the bit is set.</li> <li>When the POUT value is "1" and the timer overflows in the range from "FFFF<sub>H</sub>" to "0000<sub>H</sub>" or the timer stops and "0" is written, the bit is cleared. The bit is also cleared by reset.</li> </ul>

## ■ PWC Control Status Register, Lower Byte (PWCSL0/PWCSL1)

Figure 13.4-3 PWC control status register, lower byte (PWCSL0/PWCSL1)



**Table 13.4-2 PWC control status register (PWCSL0/PWCSL1)**

Bit name		Function
bit7, bit6	CKS1, CKS0: Clock select bits	<ul style="list-style-type: none"> <li>CKS1 and CKS0 bits are used to select the internal count clock.</li> <li>After reset, the bits are initialized to "00<sub>B</sub>". The bits can be read and written. However, "11<sub>B</sub>" cannot be set.</li> </ul> <p><b>Note:</b> After the timer is started, changing the setting is prohibited. Write these bits before the timer is started or after the timer is stopped.</p>
bit5, bit4	Reserved bits	<ul style="list-style-type: none"> <li>These bits are undefined. Always write "00<sub>B</sub>" to these bits.</li> </ul>
bit3	S/C: Single/continuous bit	<ul style="list-style-type: none"> <li>The S/C bit is used to select the count mode.</li> <li>After reset, the bit is initialized to "0". The bit can be read and written.</li> </ul> <p><b>Note:</b> After the timer is started, changing the setting is prohibited. Write this bit before the timer is started or after the timer is stopped.</p>
bit2 to bit0	MOD2, MOD1, MOD0: Operation mode bits	<ul style="list-style-type: none"> <li>Setting these bits enables selection of the operating mode and the pulse edge that fits the pulse-width count.</li> <li>After reset, these bits are initialized to "000<sub>B</sub>". These bits can be read and written.</li> </ul> <p><b>Note:</b> After the timer is started, changing the setting is prohibited. Write these bits before the timer is started or after the timer is stopped.</p> <ul style="list-style-type: none"> <li>If the continuous measurement mode is set for the setting marked *, the number of edges are totaled and the divider for the internal count clock is not cleared at the end of count. In all other modes, the divider for the internal count clock is cleared at the end of the count.</li> </ul>



---

Note:

To access this register, always use the word transfer instruction.  
After reset, this register is initialized to "0000<sub>H</sub>".

---

## MB90820B Series

### 13.4.3 Division Ratio Control Register (DIV0/DIV1)

The division ratio control register (DIV0/DIV1) is used in the division period measurement mode (PWCSL:MOD2, MOD1, and MOD0 = 011<sub>B</sub>). This register has no meaning in other modes.

#### ■ Division Ratio Control Register (DIV0/DIV1)

Figure 13.4-5 Division ratio control register (DIV0/DIV1)

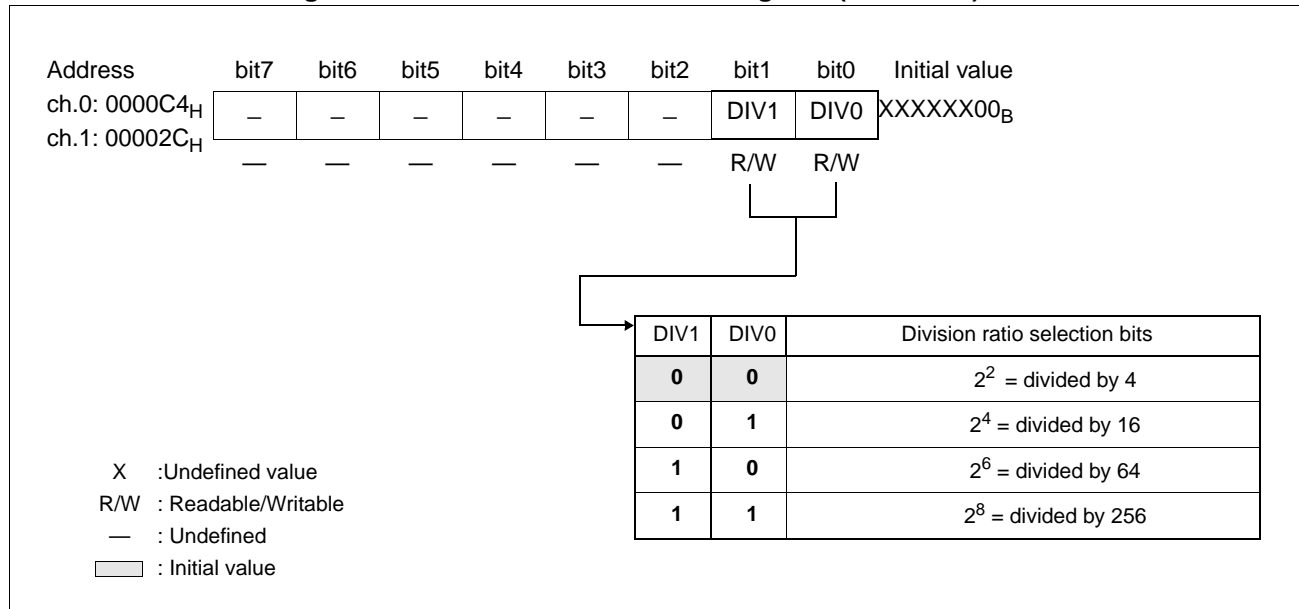


Table 13.4-3 Division ratio control register (DIV0/DIV1)

Bit name		Function
bit7 to bit2	Unused bits	<ul style="list-style-type: none"> <li>The read value is undefined.</li> <li>Writing to these bits has no effect on the operation.</li> </ul>
bit1, bit0	DIV1, DIV0: Division ratio selection bits	<ul style="list-style-type: none"> <li>In the division range measurement mode, this register is used to divide the pulse input from the measurement pin and measure the one-period width after division.</li> <li>After reset, these bits are initialized to "00<sub>B</sub>". These bits can be read and written.</li> </ul> <p><b>Note:</b> After the timer starts, the setting cannot be changed. Write these bits before the timer has started or after the timer has stopped.</p>



## 13.5 PWC Timer Interrupts

The PWC timer is enabled to generate an interrupt request in an overflow of the counter or measurement terminated in pulse-width measurement mode. It is also coordinated with the extended intelligent I/O service (EI<sup>2</sup>OS).

### ■ PWC Timer Interrupts

Table 13.5-1 lists the interrupt control bits and interrupt causes of the PWC timer.

**Table 13.5-1 Interrupt control bits and interrupt causes of the PWC timer**

	PWC timer 0		PWC timer 1	
Interrupt request flag bit	PWCSL0: OVIR	PWCSL0: EDIR	PWCSL1: OVIR	PWCSL1: EDIR
Interrupt request enable bit	PWCSL0: OVIE	PWCSL0: EDIE	PWCSL1: OVIE	PWCSL1: EDIE
Interrupt cause	Overflow of the 16-bit up counter	Measurement terminated in pulse-width measurement mode	Overflow of the 16-bit up counter	Measurement terminated in pulse-width measurement mode

In the PWC timer, the OVIR bit of the PWC control status register (PWCSL) is set to "1" by an overflow (from "FFFF<sub>H</sub>" to "0000<sub>H</sub>") of the up counter. If an interrupt request is enabled (PWCSL:OVIE = 1) in this operation, the interrupt request is output to the interrupt controller.

The EDIR bit of the PWC control status register (PWCSL) is set to "1" by measurement terminated in pulse-width measurement mode. If an interrupt request is enabled (PWCSL:EDIE = 1) in this operation, the interrupt request is output to the interrupt controller.

### ■ PWC Timer Interrupts and EI<sup>2</sup>OS

Table 13.5-2 lists the PWC timer interrupts and EI<sup>2</sup>OS.

**Table 13.5-2 16-bit PWC timer interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
PWC timer 0 <sup>*1</sup>	#13 (0D <sub>H</sub> )	ICR01	0000B1 <sub>H</sub>	FFFFC8 <sub>H</sub>	FFFFC9 <sub>H</sub>	FFFFCA <sub>H</sub>	O
PWC timer 1 <sup>*2</sup>	#24 (18 <sub>H</sub> )	ICR06	0000B6 <sub>H</sub>	FFFF9C <sub>H</sub>	FFFF9D <sub>H</sub>	FFFF9E <sub>H</sub>	

\*1: The same interrupt number as that for 16-bit PPG timer 0 is assigned to PWC timer 0.

\*2: The same interrupt number as that for output compare channel 5 match is assigned to PWC timer 1.

**■ EI<sup>2</sup>OS Function of the PWC Timer**

Since the PWC timer has a circuit that coordinates with EI<sup>2</sup>OS, the counter can start EI<sup>2</sup>OS when an overflow or measurement termination occurs.

However, EI<sup>2</sup>OS is available only when other peripheral functions sharing the interrupt control register (ICR) do not use interrupts. For example, when PWC timer 0 uses EI<sup>2</sup>OS, interrupts of the 16-bit PPG timer 0 must be disabled.

## 13.6 Operation of the PWC Timer

The PWC timer is the multi-functional timer based on the 16-bit up-count timer and contains the count input pin and 8-bit input divider. The block has two main functions: timer function and pulse-width measurement function. Both the timer function and the pulse-width measurement function enable the selection for two types of count clocks.

### ■ Timer Function

The timer function is the up-count timer that enables selection of the operation in single mode or reload mode.

When the timer is started, a timer count is performed at each count clock.

When an overflow occurs in the range from "FFFF<sub>H</sub>" to "0000<sub>H</sub>", an interrupt request is issued.

If an overflow occurs, the following occurs:

During single mode, count is discontinued (see Figure 13.6-1).

During reload mode, the reload register contents are reloaded to the timer, and the count is restarted (see Figure 13.6-2).

Figure 13.6-1 Timer operation (single mode)

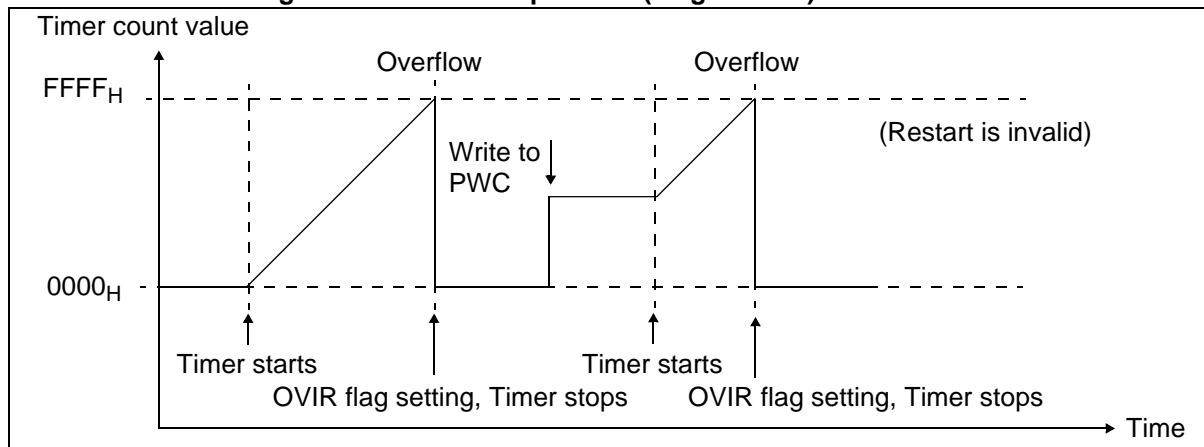
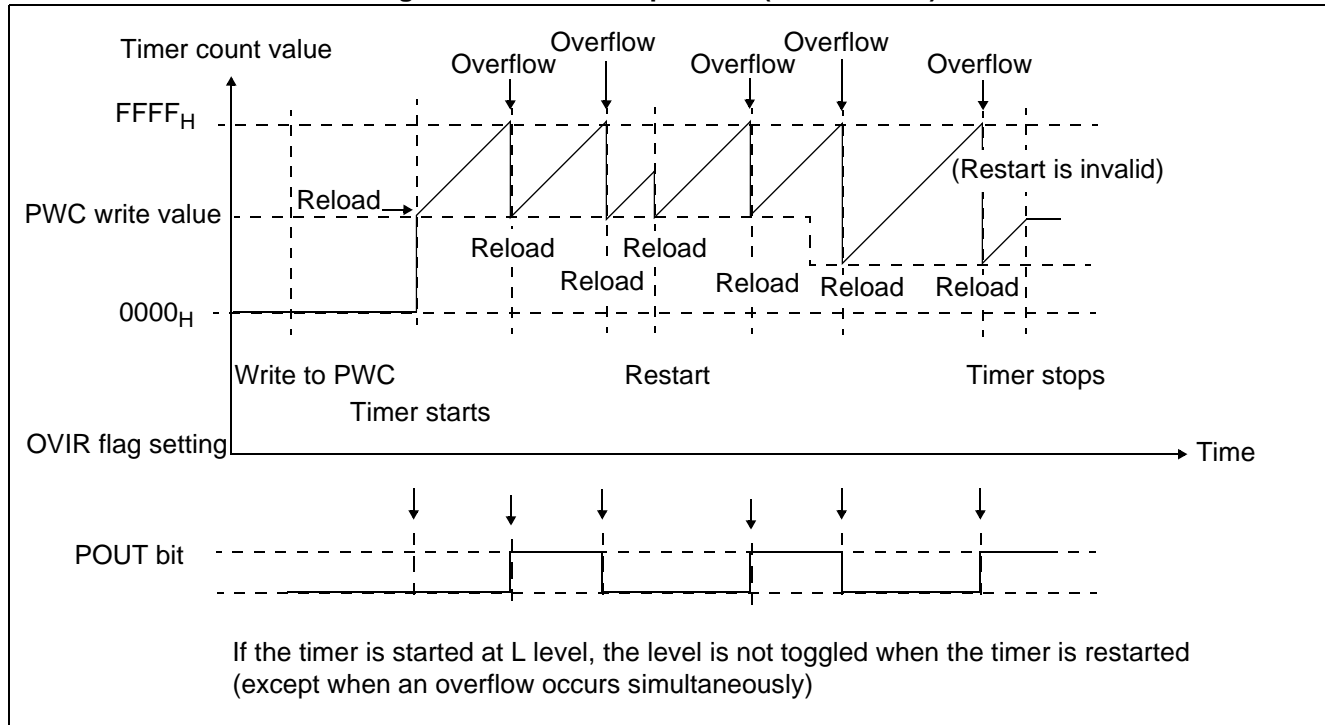


Figure 13.6-2 Timer operation (reload mode)



## ■ Pulse Width Measurement Function

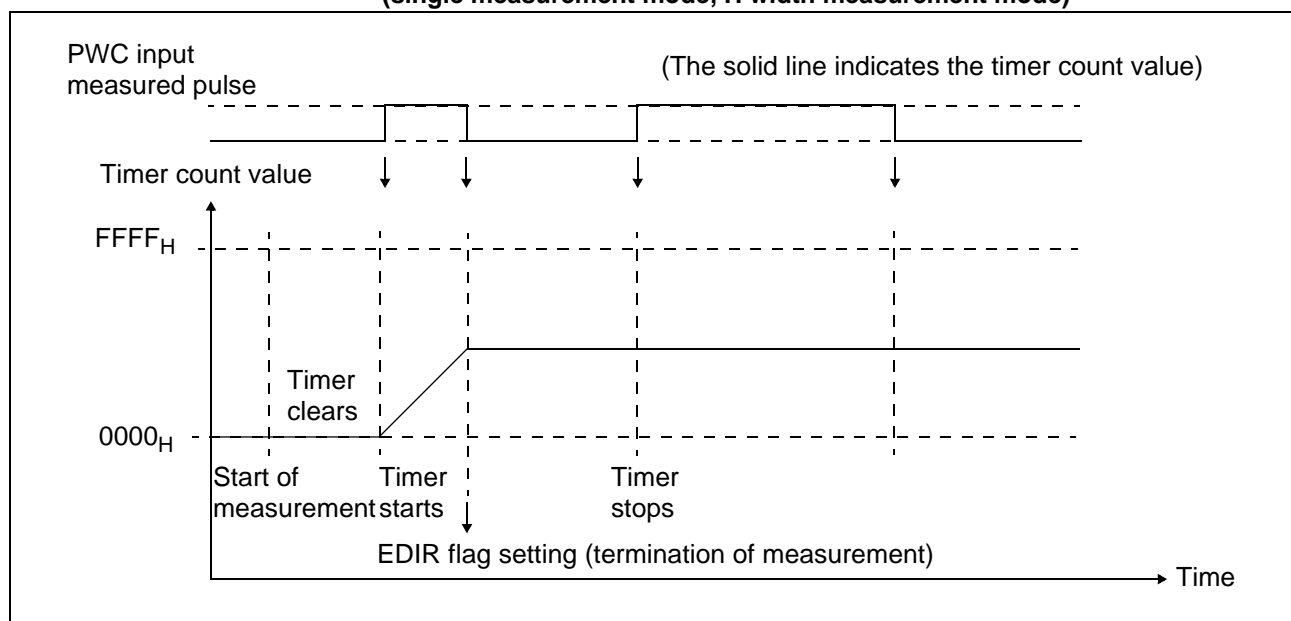
The pulse-width measurement function calculates the time between the specified events related to the input pulse.

When this function is activated, a count is started after the specified count start edge is input. If the counter is cleared to "0000<sub>H</sub>", a count is started when the start edge is detected, then the stop edge is detected. The count value during this period is held in the register as the pulse width.

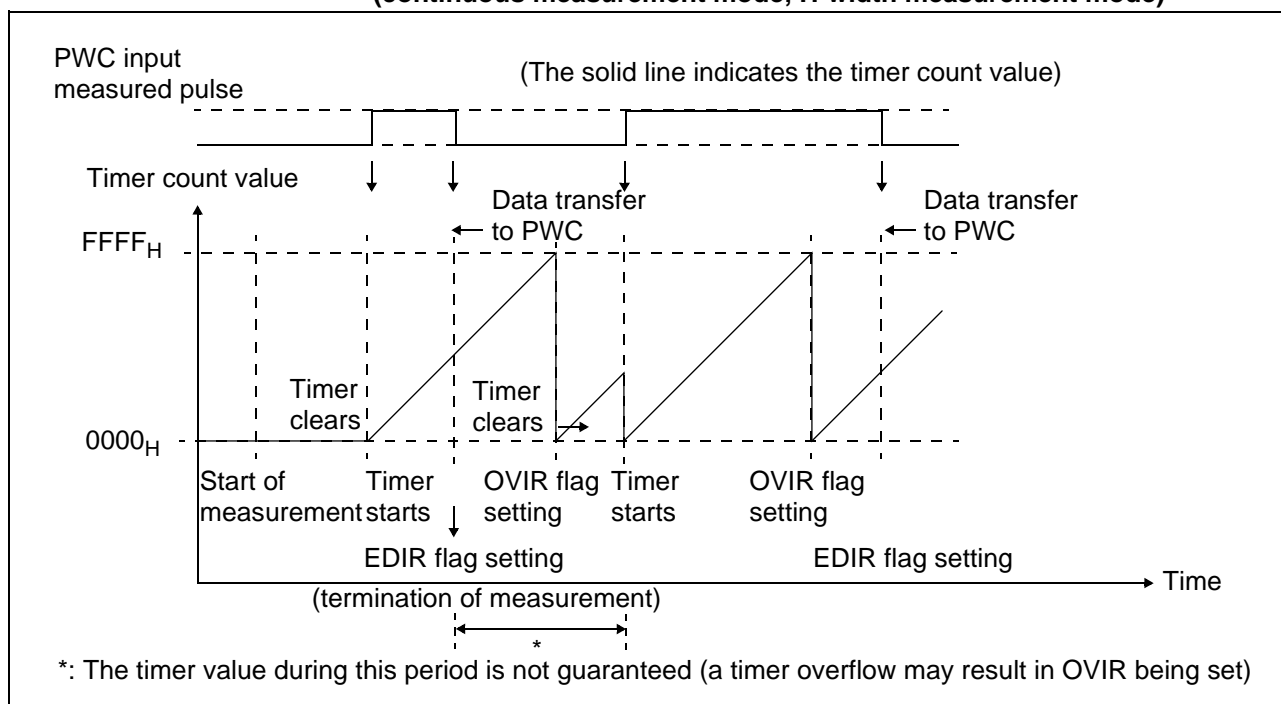
When the measurement terminates or an overflow occurs, an interrupt request can be generated. When the measurement is completed, the following occurs:

- Single measurement mode  
The operation is discontinued (see Figure 13.6-3).
- Continuous measurement mode  
The timer value is transferred to the buffer register, and the timer is in free-run state until the next edge is input (see Figure 13.6-4).

**Figure 13.6-3 Pulse-width measurement operation  
(single measurement mode, H width measurement mode)**



**Figure 13.6-4 Pulse-width measurement operation  
(continuous measurement mode, H-width measurement mode)**



## 13.6.1 Operation Mode Selection

---

Operation modes and count modes are selected according to the setting of PWCSL register.

---

### ■ Operation Mode Selection

The following registers are used to set the selection of operation modes and count modes:

- Operation mode setting: PWCSL:MOD2, MOD1, and MOD0 bits

Select the timer mode or pulse-width measurement mode to specify control of the count operation.

- Count mode setting: PWCSL:S/C bit

Select single measurement, continuous measurement, reload operation, or one-shot operation.

Table 13.6-1 lists the operation modes selected using the operation mode bits.

**Table 13.6-1 Operation mode selection**

Operation mode			S/C	MOD2	MOD1	MOD0
Timer	One-shot timer		0	0	0	0
	Reload timer		1	0	0	0 / 1
	Setting prohibited		0	0	0	1
Pulse-width measurement	Rising edge or falling edge to falling edge or rising edge: All edge-to-edge measurement	Single measurement: Buffer invalid	0	0	1	0
		Continuous measurement: Buffer valid	1	0	1	0
	Division count: Divide by 4 to 256	Single measurement: Buffer invalid	0	0	1	1
		Continuous measurement: Buffer valid	1	0	1	1
	Rising edge to rising edge: Rising edge to rising edge period measurement	Single measurement: Buffer invalid	0	1	0	0
		Continuous measurement: Buffer valid	1	1	0	0
	Rising edge to falling edge: H pulse-width measurement	Single measurement: Buffer invalid	0	1	0	1
		Continuous measurement: Buffer valid	1	1	0	1
	Falling edge to rising edge: L pulse-width measurement	Single measurement: Buffer invalid	0	1	1	0
		Continuous measurement: Buffer valid	1	1	1	0
	Falling edge to falling edge: Falling edge to falling edge period measurement	Single measurement: Buffer invalid	0	1	1	1
		Continuous measurement: Buffer valid	1	1	1	1

After reset, the one-shot timer is selected as an initial value.

Note:

Before the timer starts, always selects the operation mode.

## MB90820B Series

### 13.6.2 Starting and Stopping the Timer and Pulse-width Measurement and Clearing the Timer

To start, restart, and forcibly stop the timer and pulse-width measurement, use the PWCSH0/PWCSH1:STRT and PWCSH0/PWCSH1:STOP.

The 16-bit up-count timer is cleared to "0000<sub>H</sub>" at reset, when the measurement start edge is detected, and the count is started in the pulse-width measurement mode.

#### ■ Starting and Stopping Timer and Pulse Width Measurement

Writing "0" to the PWCSH0/PWCSH1:STRT bit starts or restarts the operation, and writing "0" to the PWCSH0/PWCSH1:STOP bit stops the operation. However, unless the value is written to these two bits are different, none of the bits executes operations. If an instruction (byte or word instruction) other than the bit manipulation instruction is being used, a value is written to the following bit combinations only.

**Table 13.6-2 Pulse-width measurement operation (single measurement mode, H width measurement mode)**

Function	STRT	STOP
Starts and restarts the timer or pulse-width measurement	0	1
Stops the timer or pulse-width measurement	1	0

If a bit manipulation instruction (clear bit instruction) is being used, the hardware automatically writes the above combination of values. The user need not know which value is to be written.

#### ● Operation after start

Timer mode: The count operation is started immediately.

Pulse-width measurement mode: Measurement is started after the measurement start edge is input. After the measurement start edge is detected, the 16-bit up-count timer is cleared to "0000<sub>H</sub>" and the count is started.

#### ● Restarting the timer

While the timer operation continues after the timer is started in the timer mode or pulse-width measurement mode, restarting the timer (writing "0" to the PWCSH0/PWCSH1:STRT bit) is called timer restart. The operations to be executed during restart are dependent on the following modes:

One-shot mode: The operation is not affected.

Reload timer mode: Reload is executed and the operation is continued. If the timer is restarted when an overflow occurs, the overflow flag (PWCSH0/PWCSH1:OVIR) is set and the POUT bit is reversed.

Pulse-width measurement mode: In the measurement start edge wait state, the operation is not affected. During measurement, the count stops and the timer state returns to the "measurement start edge wait" state. When the timer is restarted on termination of measurement, the measurement termination flag (PWCSH0/PWCSH1:EDIR) is set and the measurement results are transferred to PWC in continuous measurement mode.



● Stopping the timer

In one-shot timer mode or single measurement mode, measurement is automatically discontinued when the timer overflows or at the end of a count. The user need not know if the timer has stopped. However, in other modes, the timer must be stopped. This is also true when the timer is to be stopped before the timer automatically stops.

● Checking operation state

The previously described STRT and STOP bits function as bits that indicate the operation state of the timer during a read operation. Table 13.6-3 lists the functions of operation state indication bits.

**Table 13.6-3 Functions of operation state indication bits**

STRT	STOP	Operation state
0	0	Timer is stopping (except measurement start edge wait state). The bits indicate that the timer has not started or a measurement has terminated.
1	1	Measurement start edge wait state or timer count operation

During a read operation, both the STRT bit and the STOP bit have the same value. However, during a read operation using the read modify write instruction the values of the bits are always "11<sub>B</sub>". Do not use this instruction to read the values of the bits.

■ Clearing the Timer

In the following cases, the 16-bit up-count timer is cleared to "0000<sub>H</sub>":

- During reset
- When a count has started after the count start edge is detected in the pulse-width measurement mode

## MB90820B Series

### 13.6.3 Timer Mode Operation

---

**The timer mode includes the one-shot operation mode and reload operation mode.**

---

#### ■ One-shot Operation Mode

When the timer is started in this mode, a counter is incremented at each count clock. The timer automatically stops when an overflow occurs from "FFFF<sub>H</sub>" to "0000<sub>H</sub>".

If PWC0/PWC1 is set before the timer has started, the count is started from this set value. After overflow, the set value is deleted and the current count value remains in PWC0/PWC1.

PWCSH0/PWCSH1:POUT is reversed if an overflow occurs.

#### ■ Reload Operation Mode

When the timer is started in this mode, the reload value in PWC0/PWC1 is set in the timer and the counter is incremented at each count clock. If an overflow occurs when the timer counts from "FFFF<sub>H</sub>" to "0000<sub>H</sub>", the reload value in PWC0/PWC1 is set in the timer again, the PWCSH0/PWCSH1:POUT bit is reversed, and the count operation is repeated. The timer does not stop until a value is written to the PWCSH0/PWCSH1:STOP bit to stop the timer or it is reset. The port bit will output to pin PWO0/PWO1 if pulse output mode is specified.

The reload value (set in PWC0/PWC1 before the timer is started) is stored during a count. When the timer is started or restarted and an overflow occurs, the reload value is always set in the timer. If the value that is set during a count is to be changed, a new reload value becomes valid when the next overflow occurs or the timer is restarted.

#### ■ Timer Value and Reload Value

In one-shot operation mode, direct access to PWC register accesses the up-count timer. When a value is written to PWC0/PWC1, the value is directly written to the timer. When PWC0/PWC1 is read during a count operation, the current timer value is read. If the value is set in PWC before the timer is started, the timer starts a count from the specified value.

In reload operation mode, the up-count timer cannot be accessed and PWC0/PWC1 functions as a reload register (stores the reload value). When the timer is started or restarted and an overflow occurs, the value written to PWC is always set in the timer. When PWC0/PWC1 is read, the stored reload value is read.

The PWC value and timer value are undefined if the timer is set in one-shot mode after the operation is discontinued in reload mode. Therefore, always set the values before the timer is used.

The PWC value is undefined if the timer is set in reload mode after the operation is forcibly discontinued in one-shot mode. Therefore, always set the value before the timer is used.

#### ■ Interrupt Request Generation

During operation in timer mode, an overflow enables the generation of an interrupt request. If the increment of a timer count causes an overflow, the overflow flag is set, an overflow interrupt request is enabled, and an interrupt request is generated.

## ■ Timer Period

If the timer is started in one-shot mode after "0000<sub>H</sub>" is set in PWC0/PWC1, a timer overflow occurs and the count is discontinued if the count exceeds "65536". The following formula is used to calculate the time from start to stop of the timer.

$$T_1 = (65536 - n_1) \times t \quad \left\{ \begin{array}{l} T_1 \cdots \cdots \text{Time from start to stop of timer } (\mu\text{s}) \\ n_1 \cdots \cdots \text{Timer value set in PWC when the timer is started} \\ t \cdots \cdots \text{Count clock period } (\mu\text{s}) \end{array} \right.$$

If the timer is started after "0000<sub>H</sub>" is set in PWC0/PWC1, a timer overflow occurs every time the count exceeds "65536". The following formula is used to calculate the reload period and the PWO pin output pulse period.

$$T_R = (65536 - N_R) \times t \quad \left\{ \begin{array}{l} T_R \cdots \cdots \text{Reload period (overflow period) } (\mu\text{s}) \\ T_{\text{POUT}} \cdots \cdots \text{PWO0/PWO1 pin output pulse period } (\mu\text{s}) \\ N_R \cdots \cdots \text{Reload value stored in PWC0/PWC1 } (\mu\text{s}) \\ t \cdots \cdots \text{Time from start to stop of timer } (\mu\text{s}) \end{array} \right.$$

## ■ Count Clock Period and Maximum Period

In timer mode, when "0000<sub>H</sub>" is set in PWC0/PWC1, the maximum period results.

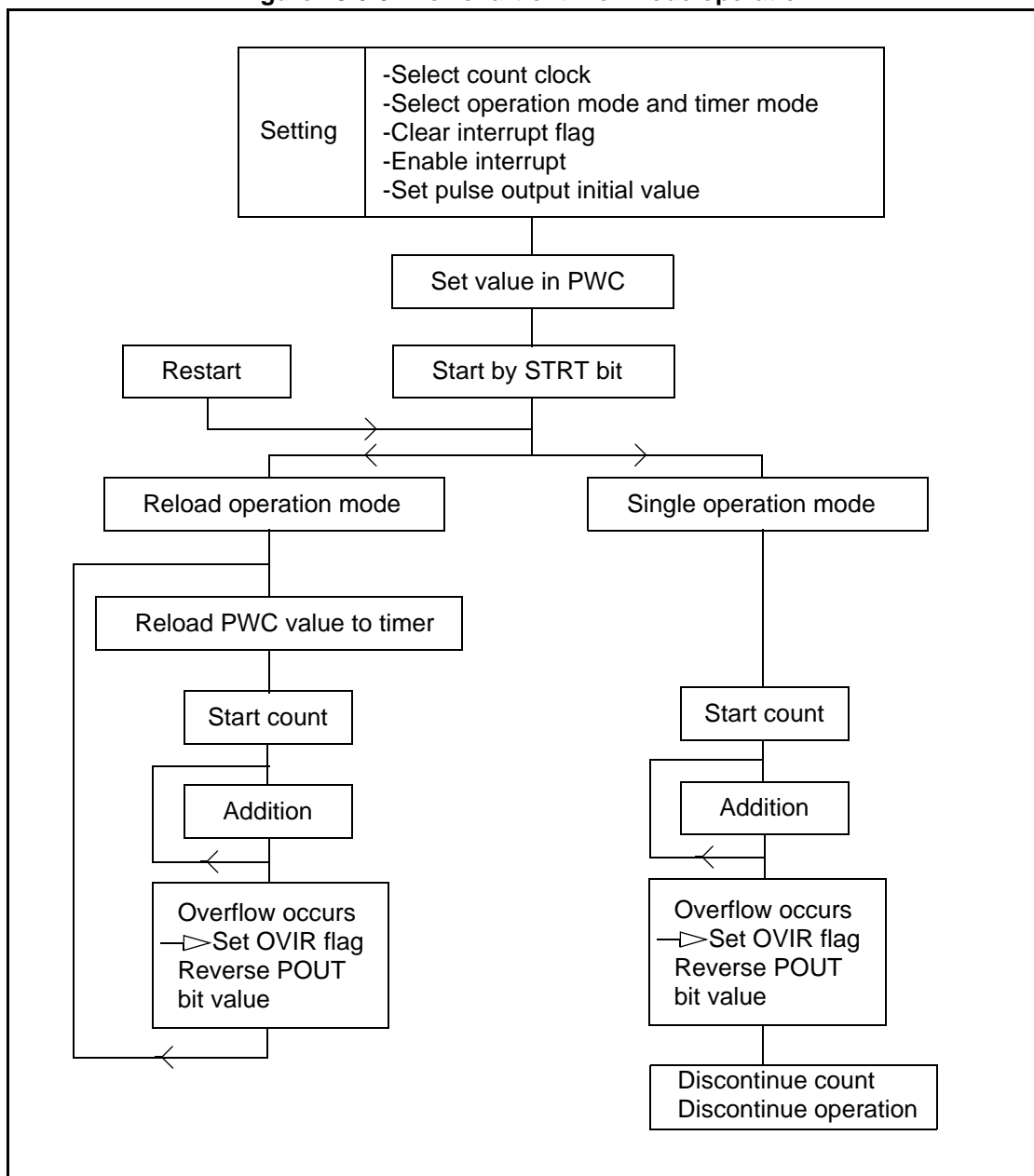
Table 13.6-4 lists the count clock period and maximum timer period corresponding to the machine clock (indicated by  $\phi$  in the table) at 24 MHz.

**Table 13.6-4 Count clock period and maximum period**

Count clock selection	When CKS1, 0=00 <sub>B</sub> ( $\phi/4$ )	When CKS1, 0=01 <sub>B</sub> ( $\phi/16$ )	When CKS1, 0=10 <sub>B</sub> ( $\phi/32$ )
Count clock period	0.17 $\mu\text{s}$	0.67 $\mu\text{s}$	1.33 $\mu\text{s}$
Maximum timer period	10.92 ms	43.69 ms	87.38 ms

## ■ Flowchart of Timer Mode Operation

Figure 13.6-5 Flowchart of timer mode operation



## 13.6.4 Pulse Width Measurement Mode Operation

---

The signal for pulse-width measurement is input from the PWI pin.

The pulse-width measurement mode includes the single measurement mode in which the count is performed only once and continuous measurement mode in which the pulse width is continuously measured.

---

### ■ Single Measurement Mode and Continuous Measurement Mode

The differences between the single measurement mode and continuous measurement mode are as follows:

- Single measurement mode

When the first count end edge is input, the timer discontinues the count, the count end flag (EDIR) of PWCSH0/PWCSH1 register is set, and the subsequent measurement is not performed. However, if a timer restart is also specified, the timer state changes to measurement start edge wait state.

- Continuous measurement mode

**[H/L pulse-width measurement mode]**

When the count end edge is input, the count end flag (EDIR) of PWCSH0/PWCSH1 is set, the timer count result is transferred to PWC0/PWC1, and the timer may continue incrementing the count in a free-run state. When the next count start edge is input, the timer is cleared to "0000<sub>H</sub>" and the pulse-width count is started.

---

Note:

When the count end edge is input and the timer enters a free-run state, the timer may overflow and the OVIR flag may be set. In the H/L pulse-width measurement mode, do not use the OVIR flag to measure the pulse-width time.

---

**[All edge-to-edge pulse-width measurement mode, division period measurement mode, rising edge-to-rising edge period measurement mode, and falling edge-to-falling edge period measurement mode]**

When the count end edge (count start edge) is input, the count end flag (EDIR) of PWCSH0/PWCSH1 register is set, the timer count result is transferred to PWC0/PWC1, the timer is cleared to "0000<sub>H</sub>", and the count is restarted.

### ■ Measurement Result Data

Handling of the measurement result, timer value, and PWC0/PWC1 function varies with the single measurement mode and continuous measurement mode as follows:

- Single measurement mode

When PWC0/PWC1 is read during timer operation, the current timer value is read.

When PWC0/PWC1 is read after termination of measurement, the measurement results are read.

## MB90820B Series

### ● Continuous measurement mode

At termination of measurement, the timer measurement results are transferred to PWC0/PWC1.

When PWC is read, the previous measurement results are read. While measurement is in progress, the previous measurement results are stored in PWC0/PWC1. During measurement, the timer value cannot be read.

In continuous measurement mode, unless the previous measurement results are read before completion of the next measurement, a new measurement result overwrites the existing value. The error flag bit (ERR) of PWCSH0/PWCSH1 register is set. When PWC0/PWC1 is read, the error flag bit (ERR) is cleared automatically.

### ■ Minimum Input Pulse Width

The pulse must be input to the pulse-width count input pin (PWI0/PWI1) longer than the following minimum input pulse width.

Pulse width: 2 machine cycles (83.3 ns or more for the machine clock at 24 MHz)

However, the input pulse that is shorter than the above specification may also be recognized as a valid pulse.

### ■ Calculating Pulse Width/Period

The pulse width or pulse period of the measurement object is calculated based on the count result read from PWC0/PWC1 at the end of a count as follows.

$$T_W = n \times t / \text{Div} (\mu\text{s}) \quad \left\{ \begin{array}{l} T_W \cdots \cdots \text{Measured pulse width or pulse period } (\mu\text{s}) \\ n \cdots \cdots \text{Measurement result contained in PWC0/PWC1} \\ t \cdots \cdots \text{Count clock period } (\mu\text{s}) \end{array} \right.$$

Div ..... Division ratio set in the division ratio register (DIV0/DIV1)  
(a value of 1 is used in a mode other than the division count mode)

### ■ Pulse Width/Period Measurement Range

The range of the pulse width/period that can be measured depends on the count clock and division ratio of an input divider.

Table 13.6-5 lists the measurement range for the machine cycle (indicated by  $\phi$ ) at 24 MHz.

**Table 13.6-5 Pulse width measurement range**

Division ratio	DIV1, DIV1	CKS1, 0=00 <sub>B</sub> ( $\phi/4$ )	CKS1, 0=01 <sub>B</sub> ( $\phi/16$ )	CKS1, 0=10 <sub>B</sub> ( $\phi/32$ )
No division	-	83.3 ns to 10.92 ms [0.17 $\mu$ s]	83.3 ns to 43.7 ms [0.67 $\mu$ s]	83.3 ns to 87.38 ms [1.33 $\mu$ s]
Divide-by 4	00 <sub>B</sub>	83.3 ns to 2.73 ms [41.7 ns]	83.3 ns to 10.92 ms [0.17 $\mu$ s]	083.3 ns to 21.85 ms [333 ns]
Divide-by 16	01 <sub>B</sub>	83.3 ns to 682.7 $\mu$ s [10.4 ns]	83.3 ns to 2.73 ms [41.7 ns]	83.3 ns to 5.46 ms [83.3 ns]
Divide-by 64	10 <sub>B</sub>	83.3 ns to 170.7 $\mu$ s [2.60 ns]	83.3 ns to 682.7 $\mu$ s [10.4 ns]	83.3 ns to 1.37 ms [20.83 ns]
Divide-by 256	11 <sub>B</sub>	83.3 ns to 42.7 $\mu$ s [0.65 ns]	83.3 ns to 170.7 $\mu$ s [2.60 ns]	83.3 ns to 0.34 ms [5.21 ns]

Note : The number in [ ] indicates the resolution per bit.

## ■ Interrupt Request Generation

In the pulse-width measurement mode, the following two interrupt requests can be generated:

### ● Timer overflow interrupt request

If an overflow occurs during a count, the overflow flag is set. When the overflow interrupt request is enabled, an interrupt request is generated.

### ● Measurement termination interrupt request

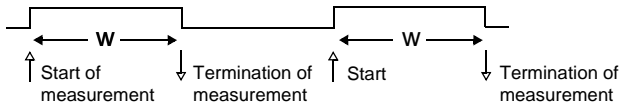
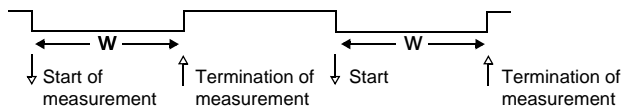
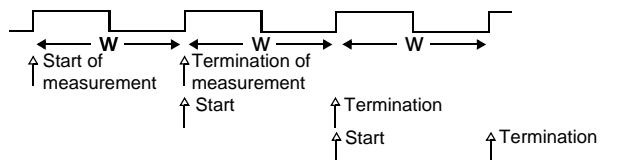
When the measurement termination edge is detected, the count end flag (EDIR) of PWCSH0/PWCSH1 register is set. If the measurement termination interrupt is enabled, an interrupt request is generated.

The measurement termination interrupt request flag bit (EDIR) is automatically cleared when PWC0/PWC1 is read.

## ■ Measurement Mode and Measurement Operation

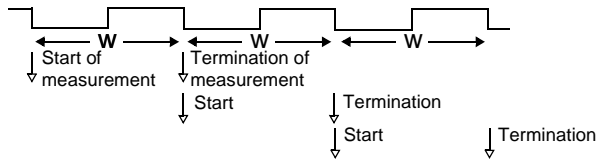
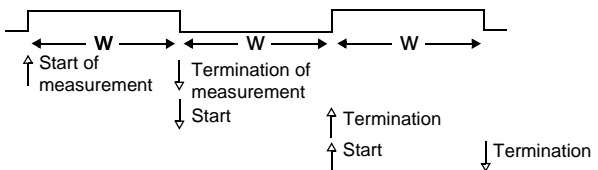
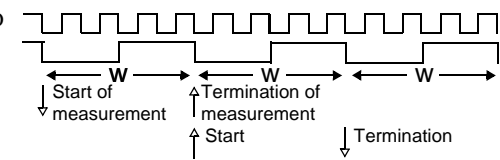
Table 13.6-6 lists measurement mode operations.

**Table 13.6-6 Measurement mode operation (1 / 2)**

Measurement mode	MOD2	MOD1	MOD0	Measurement operation
H pulse-width measurement	1	0	1	 <p>The H period width is measured.</p> <p>Start of measurement: When the rising edge is detected Termination of measurement: When the falling edge is detected</p>
L pulse-width measurement	1	1	0	 <p>The L period width is measured.</p> <p>Start of measurement: When the falling edge is detected End of measurement: When the rising edge is detected</p>
Rising edge-to-rising edge period measurement	1	0	0	 <p>The rising edge-to-rising edge time is measured.</p> <p>Start of measurement: When the rising edge is detected Termination of measurement: When the rising edge is detected</p>

# MB90820B Series

Table 13.6-6 Measurement mode operation (2 / 2)

Measurement mode	MOD2	MOD1	MOD0	Measurement operation
Falling edge-to-falling edge period measurement	1	1	1	 <p>The falling edge-to-falling edge time is measured.</p> <p>Start of measurement: When the falling edge is detected Termination of measurement: When the falling edge is detected</p>
All edge pulse-width measurement	0	1	0	 <p>The width between continuous input edges is measured.</p> <p>Start of measurement: When the edge is detected Termination of measurement: When the edge is detected</p>
Division measurement	0	1	1	 <p>(Divided by 4 in the above example.) The input pulse is divided by the division ratio set in the division ratio register (DIV0/DIV1), and the measurement period is obtained as a result.</p> <p>Start of measurement: The falling edge is detected after the operation is started. Termination of measurement: One period of division signal ends.</p>

W: Pulse width being measured

In all modes, the timer does not start count during the period from the start of measurement to input of measurement start edge. After the measurement start edge is input, the timer is cleared to "0000<sub>H</sub>", and the count is incremented at each count clock until the measurement termination edge is input.

When the measurement termination edge is input, the following operations are executed:

- (1) The count end flag (EDIR) of PWCSH0/PWCSH1 register is set.
- (2) The timer stops count operation (except if the timer is restarted at the same time the measurement end edge is input or continuous measurement mode of the H/L pulse-width measurement is used).

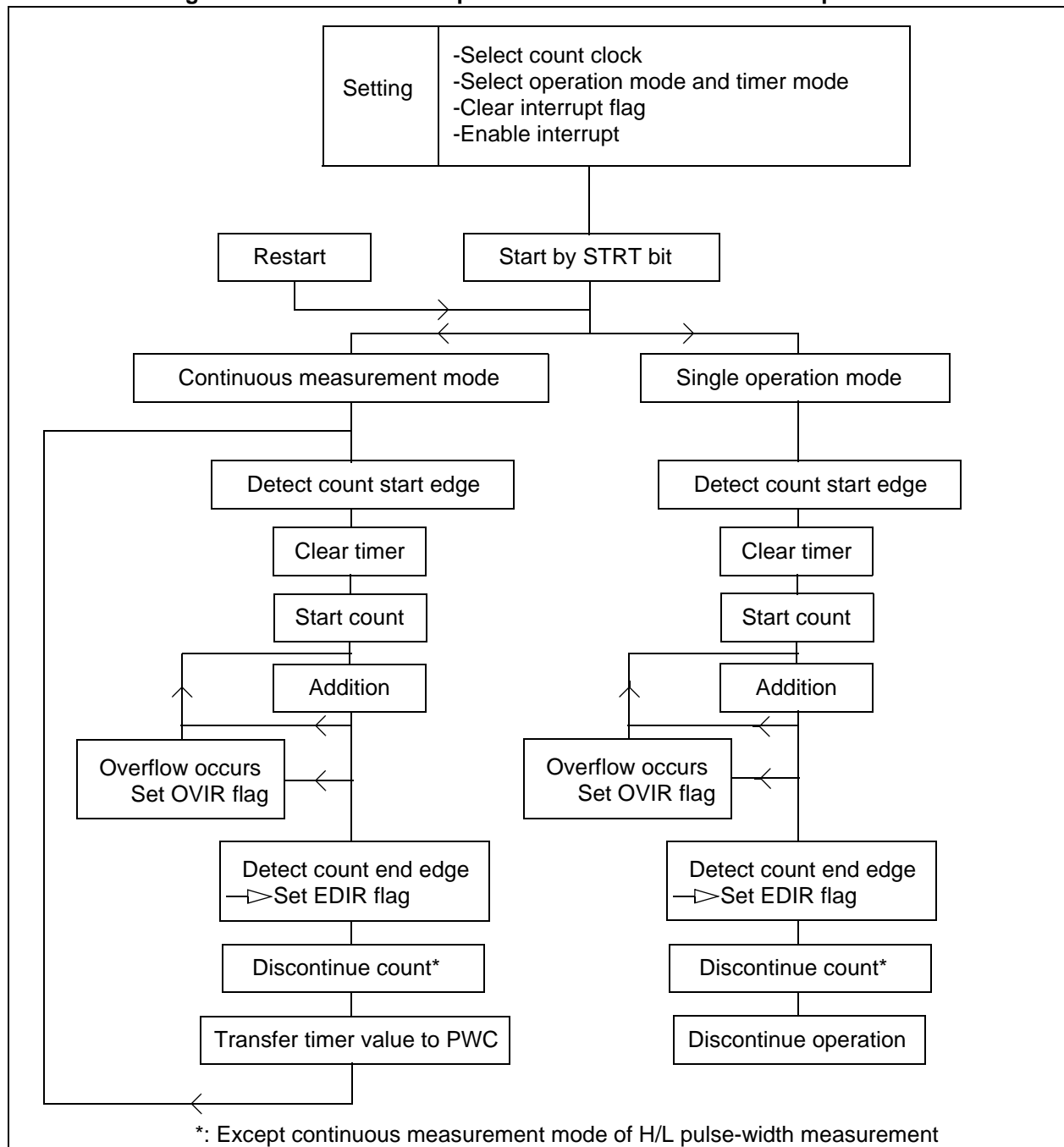


- (3) Continuous measurement mode: The timer value (measurement result) is transferred to PWC0/PWC1.
- (4) Single measurement mode: Measurement is terminated (except if the timer is restarted at the same time the measurement end edge is input).

If all edge-to-edge pulse-width measurement, period measurement, falling edge-to-falling edge period measurement, or rising edge-to-rising edge period measurement is done in continuous measurement mode, the termination edge becomes the next measurement start edge.

## ■ Flowchart of Pulse Width Measurement Operation

Figure 13.6-6 Flowchart of pulse-width measurement mode operation



## 13.7 Usage Notes on the PWC Timer

---

Notes on using the PWC timer are given below.

---

### ■ Usage Notes on the PWC Timer

#### ● Notes about using a program for setting

- Changing the following PWCS0/PWCS1 register bit values is prohibited during timer operation. The bit values are changed only before the timer is started or after the operation is discontinued.

[bit7, bit6] CKS1 and CKS0: Count clock selection bits

[bit3] S/C: Measurement mode (single or continuous) selection bit

[bit2 to bit0] MOD2, MOD1, and MOD0: Operation mode and measurement edge selection bits

Note that the value of pulse output level indication bit (POUT: bit 8) remains unchanged even if the bit is written during timer operation.

- Changing the DIV0/DIV1 value is prohibited during timer operation. Change the DIV0/DIV1 register value before the timer is started or after the operation has stopped.
- Setting the clock selection bits (CKS1 and CKS0) of PWC control status register (PWCSL0/PWCSL1) to "11<sub>B</sub>" is prohibited.
- The PWC0/PWC1 and timer values are determined when the timer is set in the one-shot mode or after the operation is terminated in reload timer mode. Therefore, always set the values after the timer is used.
- The PWC0/PWC1 value is undefined if the timer is set in reload timer mode after the operation is discontinued in the one-shot mode. Therefore, always set the value before the timer is used.
- To change the mode from pulse-width measurement mode to timer mode, always set the value in PWC0/PWC1 before the timer has started.
- When division period measurement mode is used in pulse-width measurement mode, the input pulse is divided. Note that the pulse width calculated from the count result becomes a mean value.
- During continuous measurement in pulse-width measurement mode, the division circuit for an internal count clock is not cleared, and the number of edges smaller than the count clock is added to the count result.

● Notes about using a program for status checking

- In timer mode, the value of the measurement termination interrupt request flag bit (EDIR) of PWCSH0/PWCSH1 register is insignificant. Therefore, always set "0" in the measurement end interrupt enable bit (EDIE) of PWCSH0/PWCSH1 register.
- The STRT and STOP bits of upper byte in PWC control status register (PWCSH0/PWCSH1) are dependent on whether they are read or written (see the details of registers). Read modify write instruction always reads the bits as "11<sub>B</sub>". So bit manipulation instruction cannot be used to read the operation state.  
However, a bit manipulation instruction (bit clear instruction) can be used to start or stop the timer by writing the STRT or STOP bit.
- In the pulse-width measurement mode, the measurement start edge causes the timer to be cleared, and the previous timer data is insignificant.

● Notes about pulse input to the pulse width measurement input pin

- Minimum pulse width is divide-by 2 of machine cycle (83.33 ns or more for the machine cycle at 24 MHz)
- Maximum input frequency is divide-by 4 of machine cycle (4 MHz or less for the machine clock at 24 MHz)  
If a pulse width smaller than the above or a frequency larger than the above is input, the timer operation is not guaranteed. A noise violating the above constraint and appearing in the input signal must be reduced.

● Notes about restart the timer during operation

- When an overflow occurs in reload timer mode, the timer is restarted but the overflow flag (OVIR) is set and the POUT bit is reversed (that is, the same operation as the normal overflow is executed).
- When the measurement termination edge is detected in one-shot pulse-width measurement mode, the timer is restarted and enters measurement start edge wait state, but the measurement termination flag (EDIR) is also set.
- When the measurement termination edge is detected in continuous pulse-width measurement mode, the timer is restarted and enters the measurement start edge wait state, the count termination flag (EDIR) is set, and the measurement results are transferred to PWC0/ PWC1.
- To restart the timer during operation, note the flag bit (OVIR, EDIR) operations to generate interrupts and exercise other controls.

● Notes about interrupts

- When the OVIR bit of the PWC control status register (PWCSH0/PWCSH1) is set to "1" and an interrupt request is enabled (PWCSH0/PWCSH1:OVIE = 1), control cannot be returned from interrupt processing. Always clear the OVIR bit.
- When the EDIR bit of the PWC control status register (PWCSH0/PWCSH1) is set to "1" and an interrupt request is enabled (PWCSH0/PWCSH1:EDIE = 1), control cannot be returned from interrupt processing. Always clear the OVIR bit.
- Since the PWC timer shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.

Also, when EI<sup>2</sup>OS is used by the PWC timer, shared resource interrupts must be disabled.

# **CHAPTER 14**

---

## ***16-BIT PPG TIMER***

**This chapter describes the activation and operation of the 16-bit PPG Timer.**

- 14.1 Overview of 16-bit PPG Timer
- 14.2 Block Diagram of 16-bit PPG Timer
- 14.3 16-bit PPG Timer Pins
- 14.4 16-bit PPG Timer Registers
- 14.5 16-bit PPG Timer Interrupts
- 14.6 Operation of 16-bit PPG Timer
- 14.7 Usage Notes on the 16-bit PPG Timer

## 14.1 Overview of 16-bit PPG Timer

---

**The 16-bit PPG timer consists of a 16-bit down counter, prescaler, 16-bit period setting register, 16-bit duty setting register, 16-bit control register, and PPG output pin.**

---

### ■ 16-bit PPG Timer (x 3)

The 16-bit PPG timer consists of a 16-bit down counter, prescaler, 16-bit period setting register, 16-bit duty setting register, 16-bit control register, and PPG output pin. This module can be used to output pulses synchronized by software trigger or GATE signal from multi-functional timer, refer to "CHAPTER 15 MULTI-FUNCTIONAL TIMER".

- 8 types of counter operation clock ( $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ ) can be selected ( $\phi$  is the machine clock).
- An interrupt is generated when there is a trigger, an counter borrow, or when PPG rising (normal polarity) / PPG falling (inverted polarity).
- PPG output operation

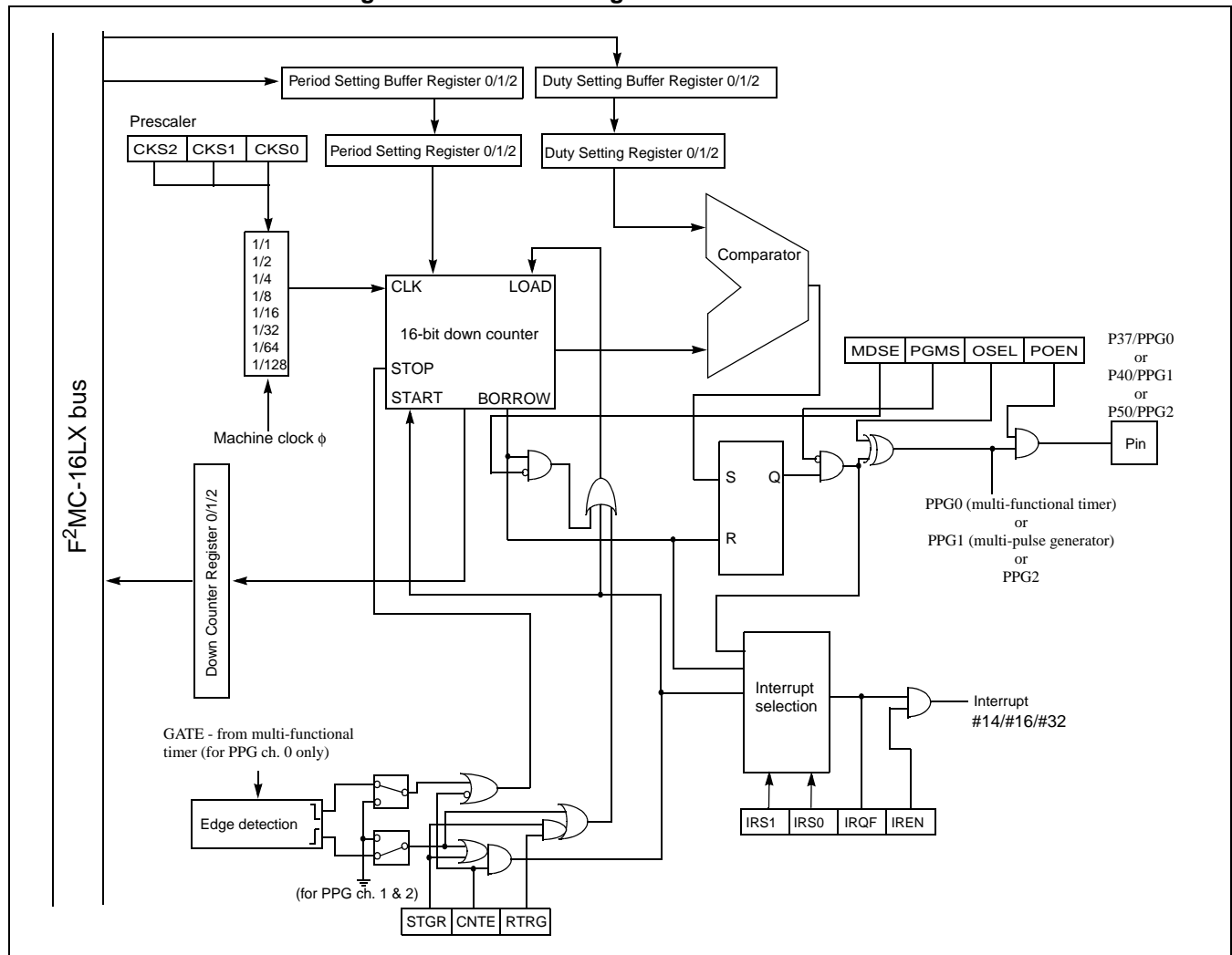
The 16-bit PPG timer can output pulse waveforms with variable period and duty ratio. Also, it can be used as D/A converter in conjunction with an external circuit.

**MB90820B Series****14.2 Block Diagram of 16-bit PPG Timer**

This section shows the block diagram of 16-bit PPG timer.

■ Block Diagram of 16-bit PPG Timer

Figure 14.2-1 Block diagram of 16-bit PPG Timer



## 14.3 16-bit PPG Timer Pins

This section describes the pins of the 16-bit PPG timer and provides a pin block diagram.

### 16-bit PPG Timer Pins

The pins of the 16-bit PPG timer are shared with the general-purpose I/O ports. Table 14.3-1 lists the functions of the pins, I/O format, and settings required to use the 16-bit PPG timer.

Table 14.3-1 16-bit PPG timer pins

Pin name	Pin function	I/O format	Pull-up option	Standby control	Settings required for pins
P37/PPG0	Port 3 input-output / PPG0 output	CMOS output / CMOS input	Selectable	Available	Setting for the PPG timer 0 output (PNCTL0:POEN=1)
P40/PPG1	Port 4 input-output / PPG1 output	CMOS output / CMOS hysteresis input	Not provided		Setting for PPG timer 1 output enable (PNCTL1:POEN=1)
P50/PPG2	Port 5 input-output / PPG2 output				Setting for PPG timer 2 output enable (PNCTL2:POEN=1)

### Block Diagram of the 16-bit PPG Timer Pins

Figure 14.3-1 Block diagram of the 16-bit PPG timer pins (PPG1, PPG2)

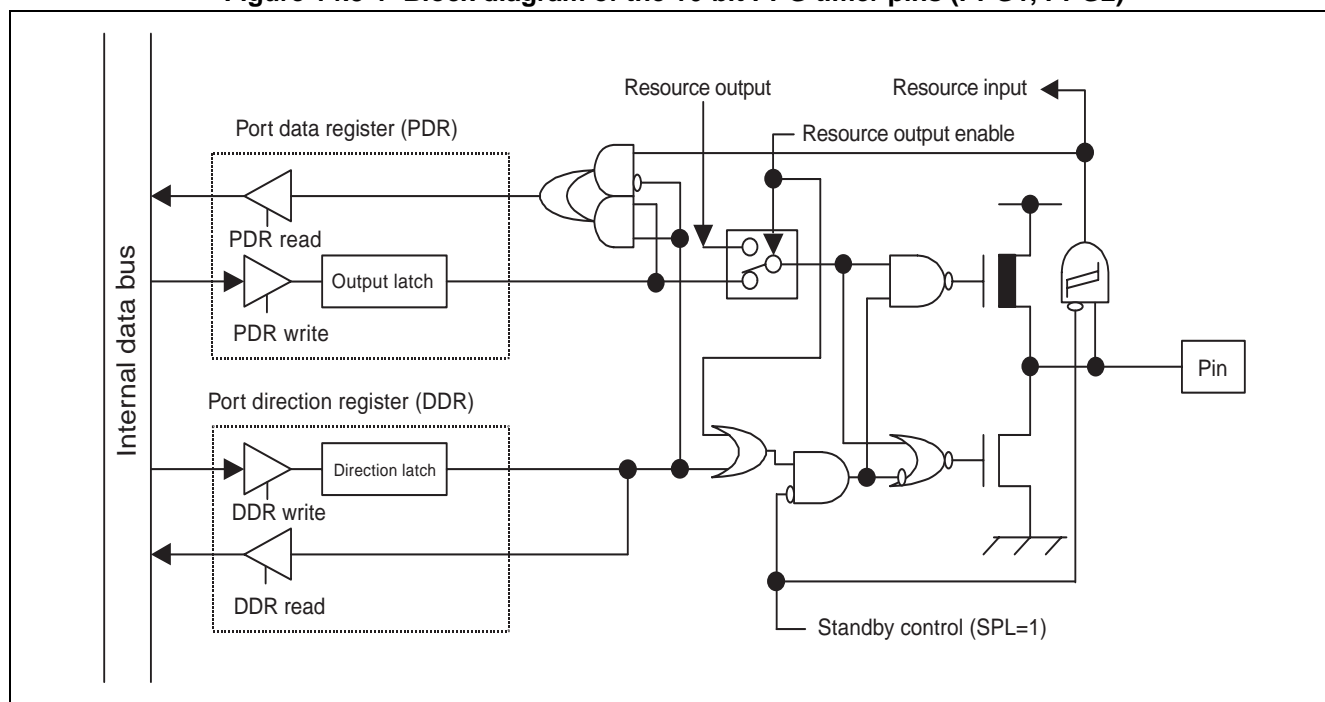
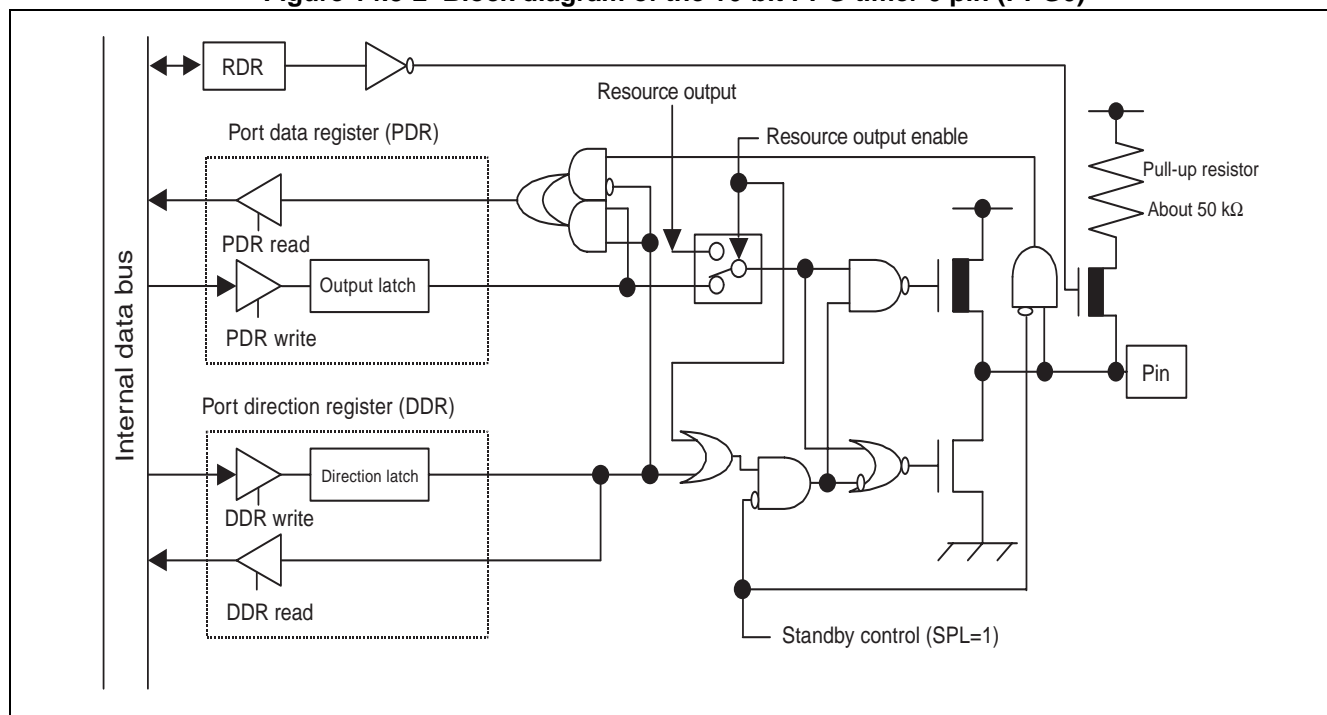


Figure 14.3-2 Block diagram of the 16-bit PPG timer 0 pin (PPG0)



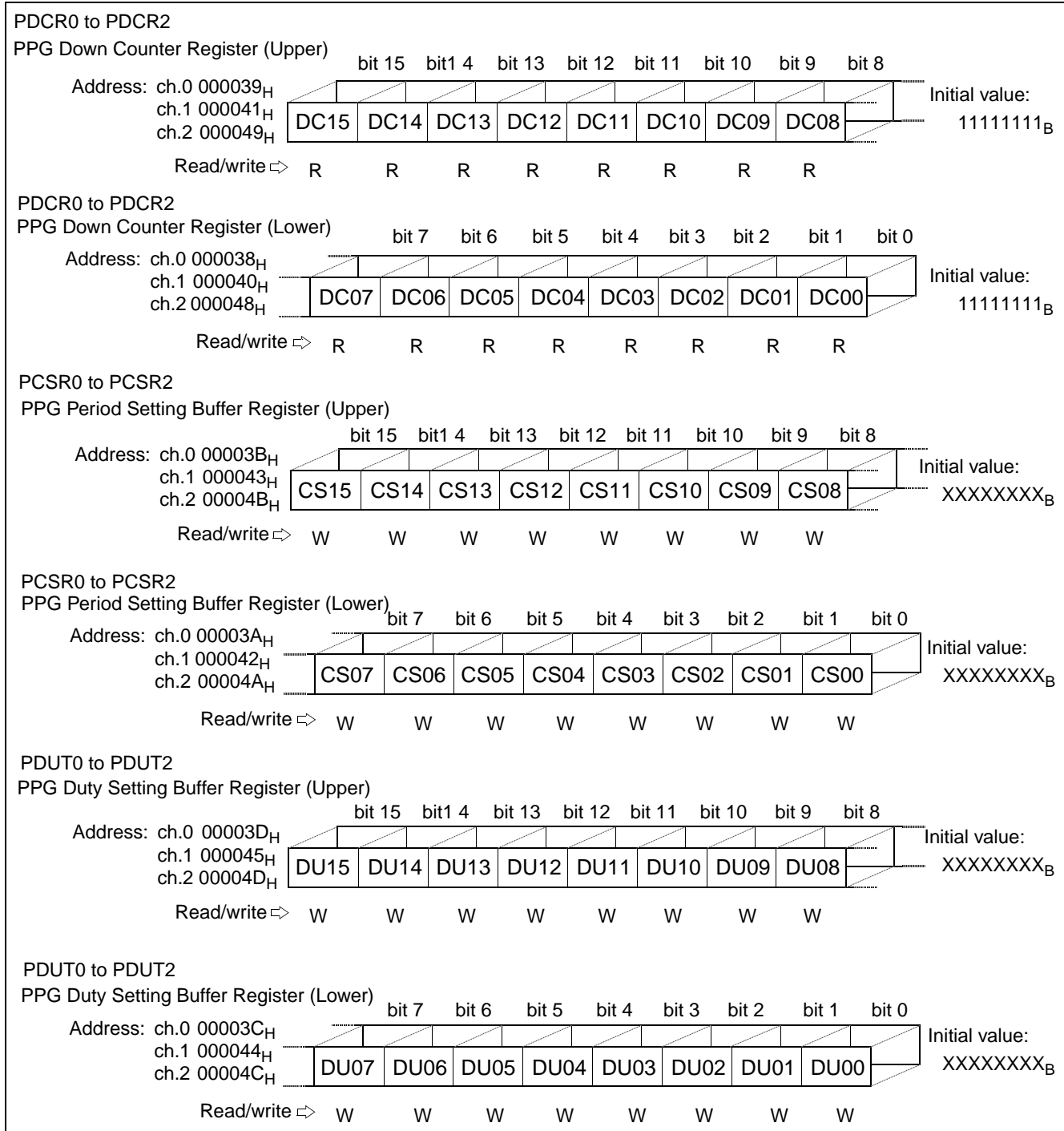


## 14.4 16-bit PPG Timer Registers

This section shows the register of the 16-bit PPG timer.

### 16-bit PPG Timer Registers

Figure 14.4-1 Registers of 16-bit PPG timer



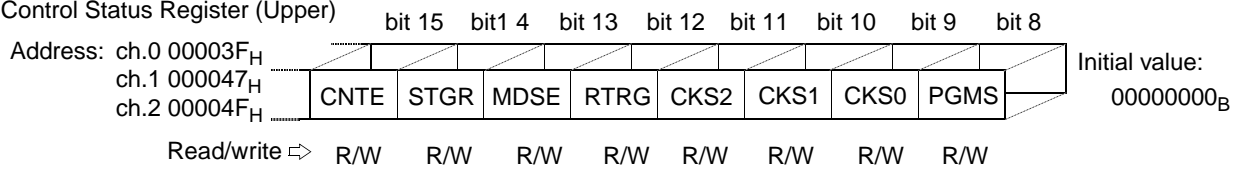
(continued)

**MB90820B Series**

(continued)

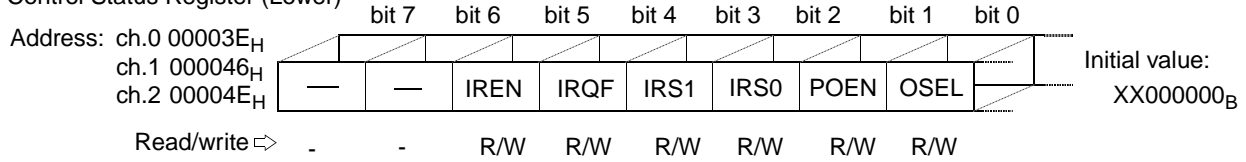
PCNTH0 to PCNTH2

PPG Control Status Register (Upper)



PCNTL0 to PCNTL2

PPG Control Status Register (Lower)

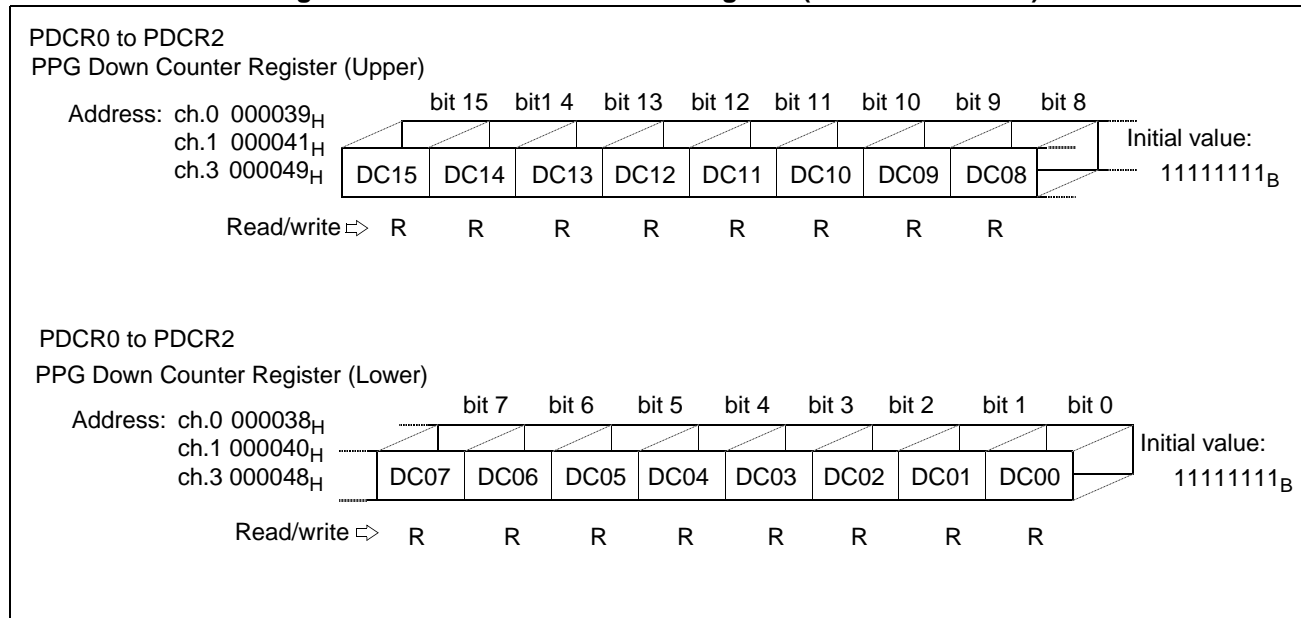


## 14.4.1 PPG Down Counter Register (PDCR0 to PDCR2)

PPG down counter registers (PDCR0 to PDCR2) are 16-bit registers, which are used to read the count value of the 16-bit PPG down counter.

### ■ PPG Down Counter Register (PDCR0 to PDCR2)

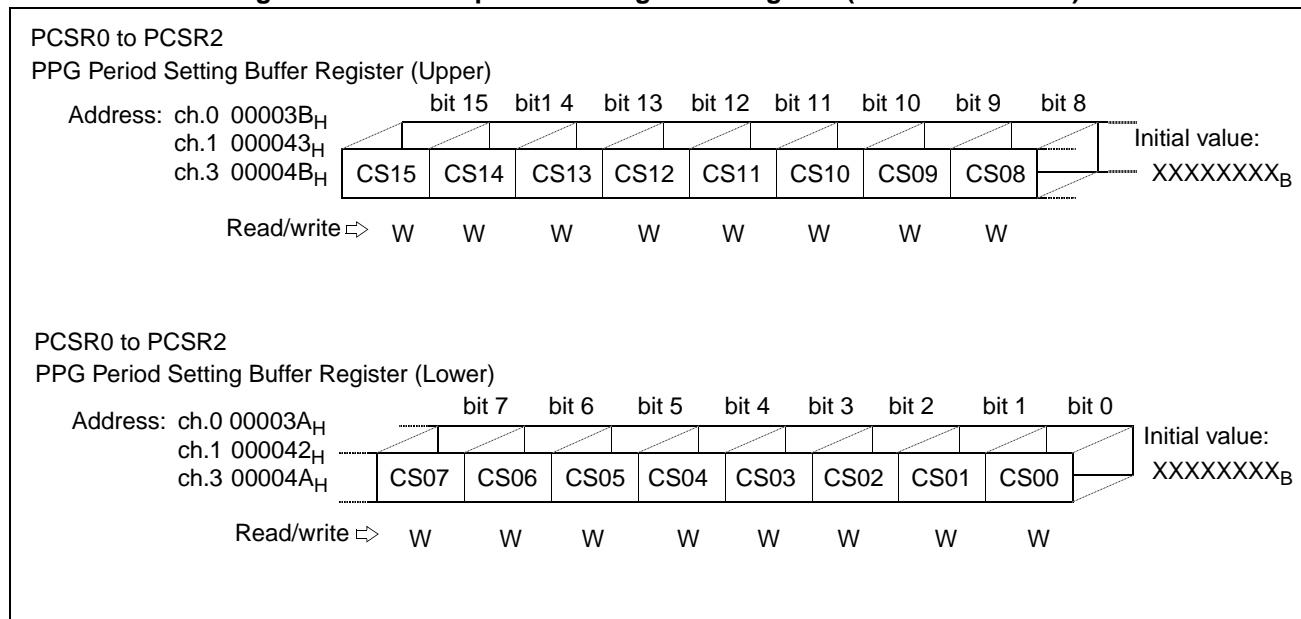
Figure 14.4-2 PPG down counter register (PDCR0 to PDCR2)



These are 16-bit registers that are used to store the values of the 16-bit down counter. The initial value of them are all 1. Word access instruction to these register are recommended. These registers are read only.

**MB90820B Series****14.4.2 PPG Period Setting Buffer Register (PCSR0 to PCSR2)**

PPG period setting buffer register is used to set the period of the output pulses generated by PPG.

**Figure 14.4-3 PPG period setting buffer register (PCSR0 to PCSR2)**

These are 16-bit registers that are used to set the period of the output pulses generated by PPG. The initial value of them are undetermined, so that these registers must be written before starting an operation. Word access instruction to these registers are recommended. These registers are write-only.

Data transfer from PPG period setting buffer register to period setting register will be at counter borrow, trigger, or retrigger, if enabled.

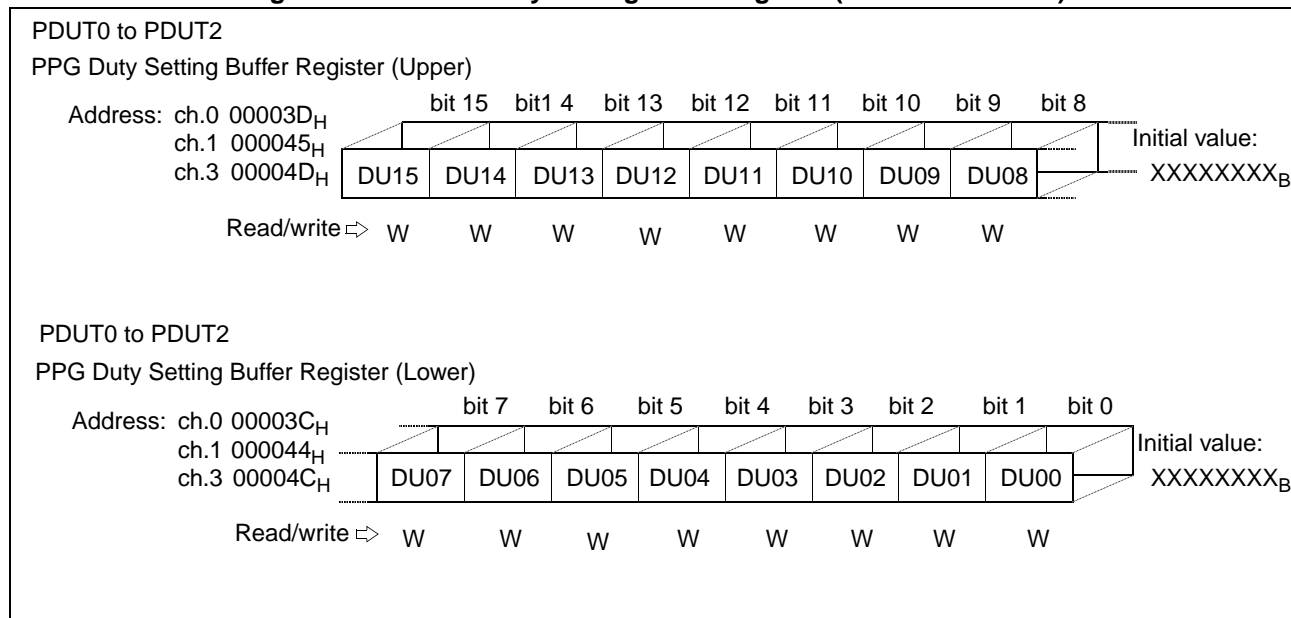
**Note :**

In case of updating PPG period setting buffer register, duty setting buffer register must be written after writing to PPG period setting buffer register. Only updating PPG period setting buffer register is prohibited.

### 14.4.3 PPG Duty Setting Buffer Register (PDUT0 to PDUT2)

PPG duty setting buffer register is used to control the duty ratio of the output pulses generated by PPG.

Figure 14.4-4 PPG duty setting buffer register (PDUT0 to PDUT2)



These are 16-bit registers that are used to control the duty ratio of the output pulses generated by PPG. The initial value of them are undetermined, so that these registers must be set a value before starting an operation. Word access instruction to these registers are recommended. These registers are write-only.

Data transfer from PPG duty setting buffer register to duty setting register is at counter borrow, trigger, or retrigger if enabled.

Setting the same value in both the PPG period setting register and duty setting register outputs all "H"s for normal polarity and all "L"s for inverted polarity.

The output of the PPG is undefined if PCSR < PDUT.

Note :

PPG duty setting buffer register can be written in the case of not updating PPG period setting buffer register.

**MB90820B Series****14.4.4 PPG Control Status Register (PCNTL0 to PCNTL2,  
PCNTH0 to PCNTH2)**

---

PPG control status register is used to set operating conditions for 16-bit PPG timer enable or disable operation, software trigger, retrigger control interrupt, and output polarity and check the status

---

**■ PPG Control Status Register, Upper Byte (PCNTH0 to PCNTH2)**

Figure 14.4-5 PPG control register, upper byte (PCNTH0 to PCNTH2)

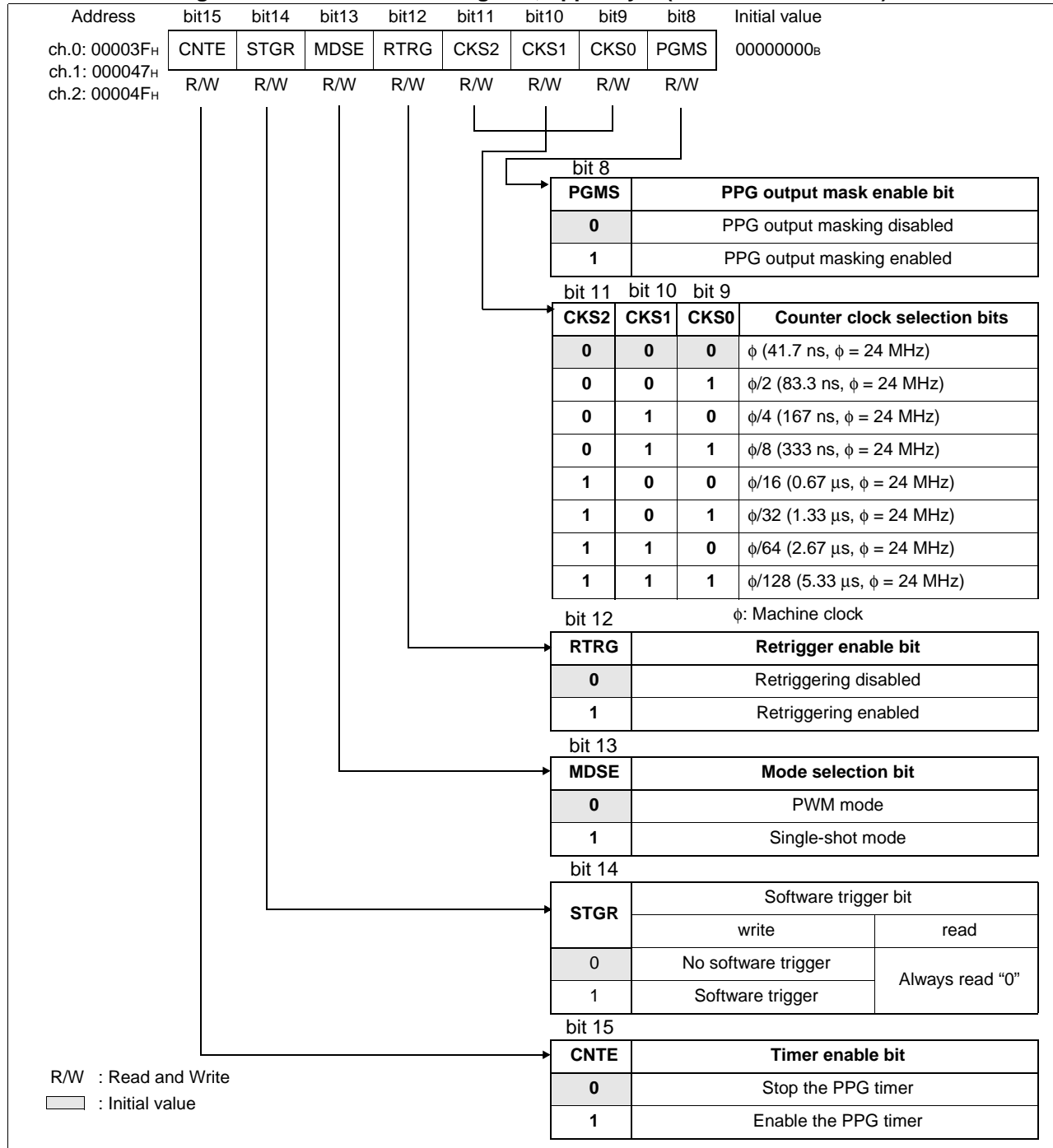


Table 14.4-1 PPG control status register, upper byte (PCNTH0 to PCNTH2) bit

Bit name		Function
bit15	CNTE: Timer enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable the PPG timer operation. Writing "1" will enable the PPG operation and wait for trigger to start PPG operation. Writing "0" will stop the operation.</li> </ul>
bit14	STGR: Software trigger bit	<ul style="list-style-type: none"> <li>This bit is the software trigger bit for PPG. Writing "1" to this bit triggers the PPG by software.</li> <li>This bit is always read as "0".</li> </ul>
bit13	MDSE: Mode selection bit	<p>When this bit is "0", PPG operates in PWM mode. When this bit is "1", PPG operates in single-shot mode.</p>
bit12	RTRG: Retrigger enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable retriggering function of PPG during operation. When this bit is "0", retriggering function is disabled. When this bit is "1", retriggering function is enabled.</li> </ul>
bit11 to bit9	CKS2, CKS1, CKS0: Counter clock selection bits	<ul style="list-style-type: none"> <li>These bits are used to select the operation clock for 16-bit PPG timer.</li> </ul>
bit8	PGMS: PPG output mask enable bit	<ul style="list-style-type: none"> <li>This bit is used to mask the PPG output to specific level regardless of the mode setting (PCNTH:MDSE), period setting (PCSR), or duty setting (PDUT). Write "0" will disable PPG output masking function. Writing "1" to this bit masks the PPG output to always "L" when polarity setting is "Normal" (PCNTL:OSEL=0). Writing "1" to this bit masks the PPG output to always "H" when polarity setting is "Inverted" (PCNTL:OSEL=1).</li> </ul> <p><b>Note:</b> By setting PPG period setting buffer register (PCSR) and PPG duty setting buffer register (PDUT) with same value, all "H" in normal polarity or all "L" in inverted polarity can be outputted when this bit is "1".</p>



■ PPG Control Status Register, Lower Byte (PCNTL1 to PCNTL3)

Figure 14.4-6 PPG control register (PCNTL1 to PCNTL3)

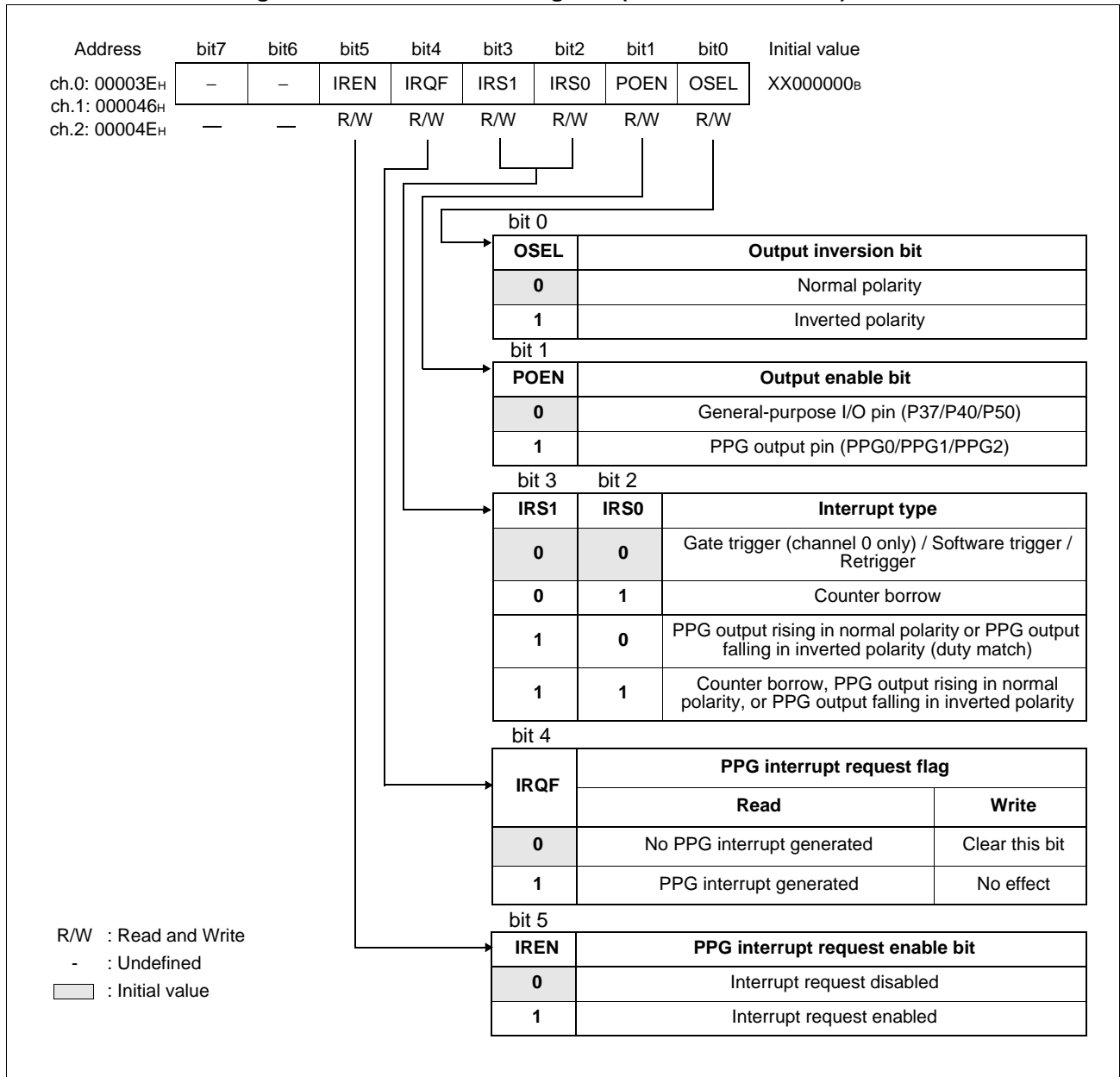


Table 14.4-2 PPG control status register (PCNTL1 to PCNTL3)

Bit name		Function
bit7, bit6	Undefined bits	These read value are undefined. Writing to these bits has no effect on the operation.
bit5	IREN: PPG interrupt request enable bit	<ul style="list-style-type: none"> <li>This bit enables or disables PPG interrupt request to the CPU.</li> <li>When this bit and the interrupt flag (IRQF) bit are "1", PPG outputs an interrupt request.</li> </ul>
bit4	IRQF: PPG interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is set to 1 when PPG interrupt occurs. Writing "0" will clear this bit. Writing 1 has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> <li>This bit is also cleared when EI<sup>2</sup>OS is activated.</li> </ul>
bit3, bit2	IRS1, IRS0: Interrupt selection bits	<ul style="list-style-type: none"> <li>These bits are used to select interrupt operation of the PPG timer.</li> </ul>
bit1	POEN: Output enable bit	<ul style="list-style-type: none"> <li>This bit enables or disables output from the PPG output pin. When this bit is "0", the pin functions as a general-purpose port. When this bit is "1", the pin functions as a PPG timer output pin.</li> </ul>
bit0	OSEL: Output inversion bit	<ul style="list-style-type: none"> <li>This bit selects the polarity of PPG output pin. When this bit is "0", normal polarity is selected. PPG outputs "L" when 16-bit down count vlaue is greater than PDUT, and outputs "H" when smaller than or equals to PDUT. When this bit is "1", the PPG output is inverted.</li> </ul>

## 14.5 16-bit PPG Timer Interrupts

The 16-bit PPG timer is enabled to generate an interrupt request when trigger or counter borrow, PPG rising in normal polarity, or PPG falling in inverted polarity depending on PCNTL : IRS1 and IRS0 setting. It is also coordinated with the extended intelligent I/O service (EI<sup>2</sup>OS).

### ■ 16-bit PPG Timer Interrupts

Table 14.5-1 lists the interrupt control bits and interrupt causes of the 16-bit PPG timer.

**Table 14.5-1 Interrupt control bits and interrupt causes of the 16-bit PPG timer**

	16-bit PPG timer 0	16-bit PPG timer 1	16-bit PPG timer 2
Interrupt flag bit	PCNTL0:IRQF	PCNTL1:IRQF	PCNTL2:IRQF
Interrupt request enable bit	PCNTL0:IREN	PCNTL1:IREN	PCNTL2:IREN
Interrupt type selection bits	PCNTL0:IRS1, IRS0	PCNTL1:IRS1, IRS0	PCNTL2:IRS1, IRS0
Interrupt cause	PCNTL0:IRS1, IRS0=00 gate trigger/software trigger/retrigger of 16-bit down counter 0	PCNTL1:IRS1, IRS0=00 software trigger/retrigger of 16-bit down counter 1	PCNTL2:IRS1, IRS0=00 software trigger/retrigger of 16-bit down counter 2
	PCNTL0:IRS1, IRS0=01 counter borrow of 16-bit down counter 0	PCNTL1:IRS1, IRS0=01 counter borrow of 16-bit down counter 1	PCNTL2:IRS1, IRS0=01 counter borrow of 16-bit down counter 2
	PCNTL0:IRS1, IRS0=10 PPG0 output rising in normal polarity or PPG0 output falling in inverted polarity	PCNTL1:IRS1, IRS0=10 PPG1 output rising in normal polarity or PPG1 output falling in inverted polarity	PCNTL2:IRS1, IRS0=10 PPG2 output rising in normal polarity or PPG2 output falling in inverted polarity
	PCNTL0:IRS1, IRS0=11 Counter borrow of 16-bit down counter 0, PPG0 output rising in normal polarity, or PPG0 output falling in inverted polarity	PCNTL0:IRS1, IRS0=11 Counter borrow of 16-bit down counter 1, PPG1 output rising in normal polarity, or PPG1 output falling in inverted polarity	PCNTL0:IRS1, IRS0=11 Counter borrow of 16-bit down counter 2, PPG2 output rising in normal polarity, or PPG2 output falling in inverted polarity

In the 16-bit PPG timer, the IRQF bit of the PPG control status register (PCNTL) is set to "1" and an interrupt request is enabled (PCNTL: IREN=1), the interrupt request is outputted to the interrupt controller.

**MB90820B Series****■ 16-bit PPG Timer Interrupts and EI<sup>2</sup>OS**

Table 14.5-2 lists the 16-bit PPG timer interrupts and EI<sup>2</sup>OS.

**Table 14.5-2 16-bit PPG timer interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
16-bit PPG timer 0 <sup>*1</sup>	#14 (0E <sub>H</sub> )	ICR01	0000B1 <sub>H</sub>	FFFFC4 <sub>H</sub>	FFFFC5 <sub>H</sub>	FFFFC6 <sub>H</sub>	O
16-bit PPG timer 1 <sup>*2</sup>	#16 (10 <sub>H</sub> )	ICR02	0000B2 <sub>H</sub>	FFFFBC <sub>H</sub>	FFFFBD <sub>H</sub>	FFFFBE <sub>H</sub>	
16-bit PPG timer 2 <sup>*3</sup>	#32 (20 <sub>H</sub> )	ICR10	0000BA <sub>H</sub>	FFFF7C <sub>H</sub>	FFFF7D <sub>H</sub>	FFFF7E <sub>H</sub>	

\*1: The same interrupt control register as that for 16-bit PPG timer 0 is assigned to PWC timer 0.

\*2: The same interrupt control register as that for 16-bit PPG timer 1 is assigned to 16-bit output compare channel 1 match.

\*3: The same interrupt control register as that for 16-bit PPG timer 2 is assigned to 16-bit free-run timer zero detection.

**■ EI<sup>2</sup>OS Function of the 16-bit PPG Timer**

Since the 16-bit PPG timer has a circuit that coordinates with EI<sup>2</sup>OS, the counter can start EI<sup>2</sup>OS when PPG interrupt occurs.

However, EI<sup>2</sup>OS is available only when other peripheral functions sharing the interrupt control register (ICR) do not use interrupts. For example, when 16-bit PPG timer 0 uses EI<sup>2</sup>OS, the output compare channel 0 match must be disabled.

## 14.6 Operation of 16-bit PPG Timer

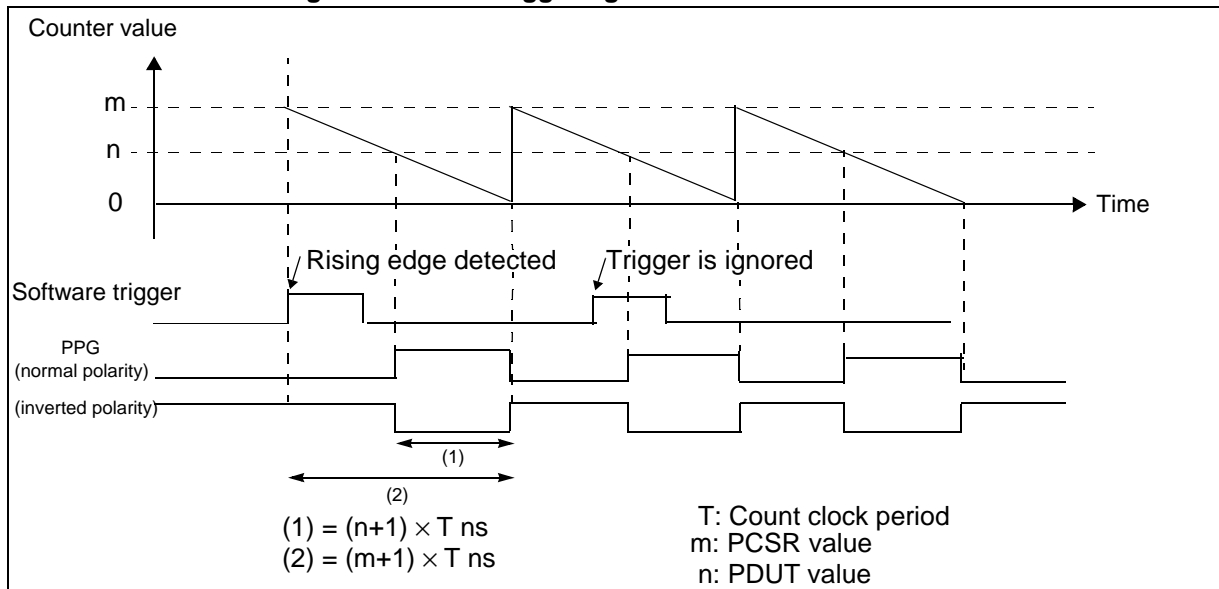
The 16-bit PPG timer operates in either PWM mode or single-shot mode. And Retriggering can be enabled.

### ■ PWM Mode (PCNTL: MDSE = 0)

For PWM operation, the 16-bit down counter will be loaded with PCSR value, starts counting after a valid trigger is detected. And once the 16-bit down counter reached zero, it is reloaded with PCSR value and repeat counting again. PPG output is toggled when 16-bit down counter is reloaded. The period of the output pulses can be controlled by setting PCSR register, and the duty ratio controlled by setting PDUT.

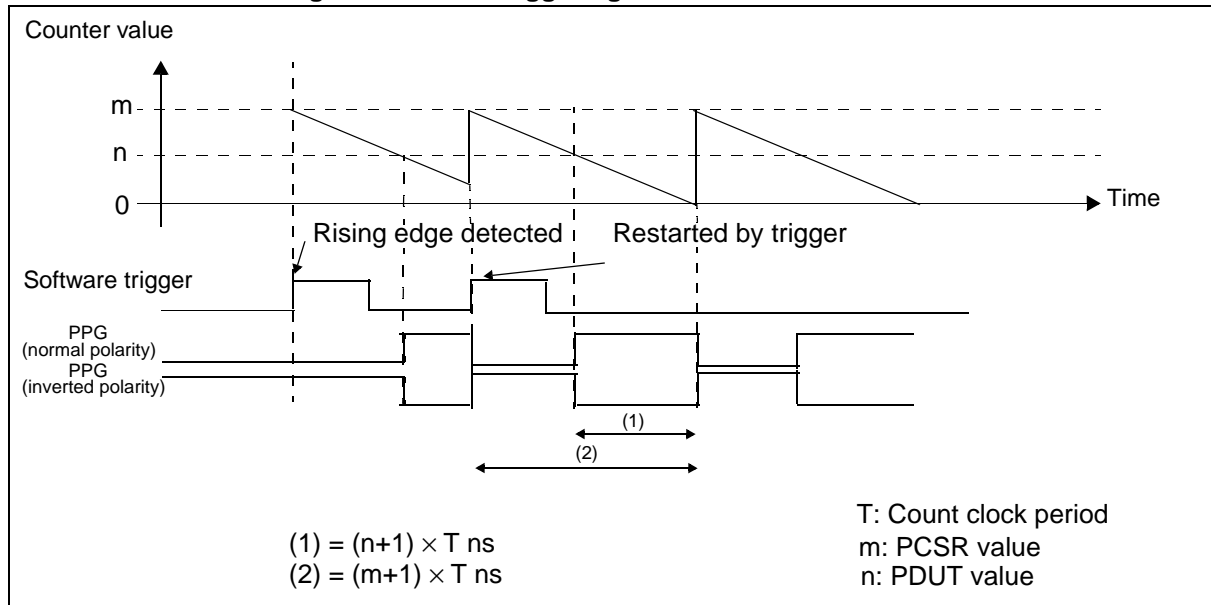
a) Retriggering is disabled (PCNTH: RTRG = 0)

Figure 14.6-1 Retriggering is disabled in PWM mode



b) Retriggerring is enabled (PCNTH: RTRG = 1)

Figure 14.6-2 Retriggerring is enabled in PWM mode

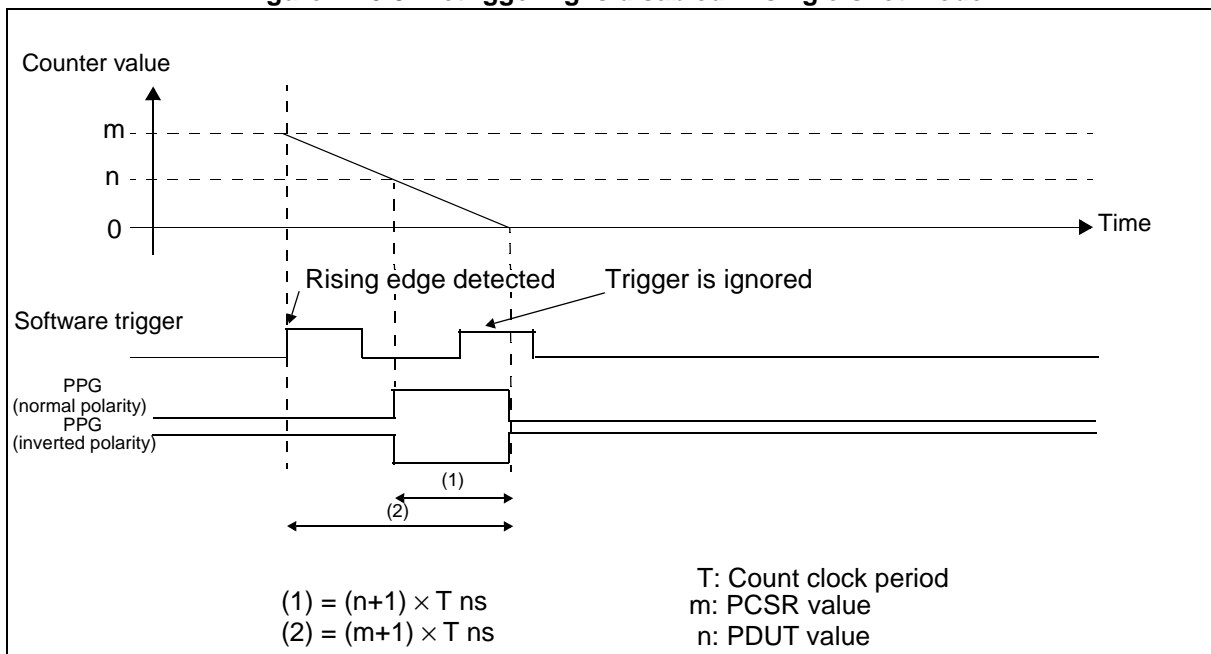


### ■ Single-shot Mode (PCNTL: MDSE = 1)

For single-shot operation, a single pulse of specified width can be output by a valid trigger. When retriggering is enabled, the counter is reloaded if an edge is detected during operation.

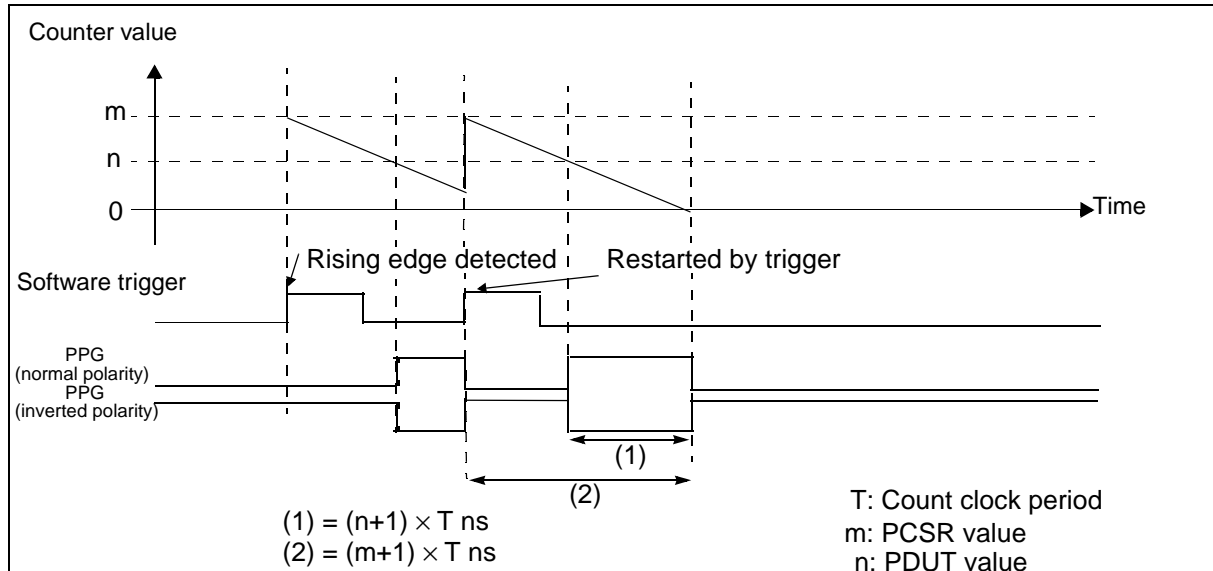
a) Retriggerring is disabled (PCNTH: RTRG = 0)

Figure 14.6-3 Retriggerring is disabled in single-shot mode



b) Retriggering is enabled (PCNTH: RTRG = 1)

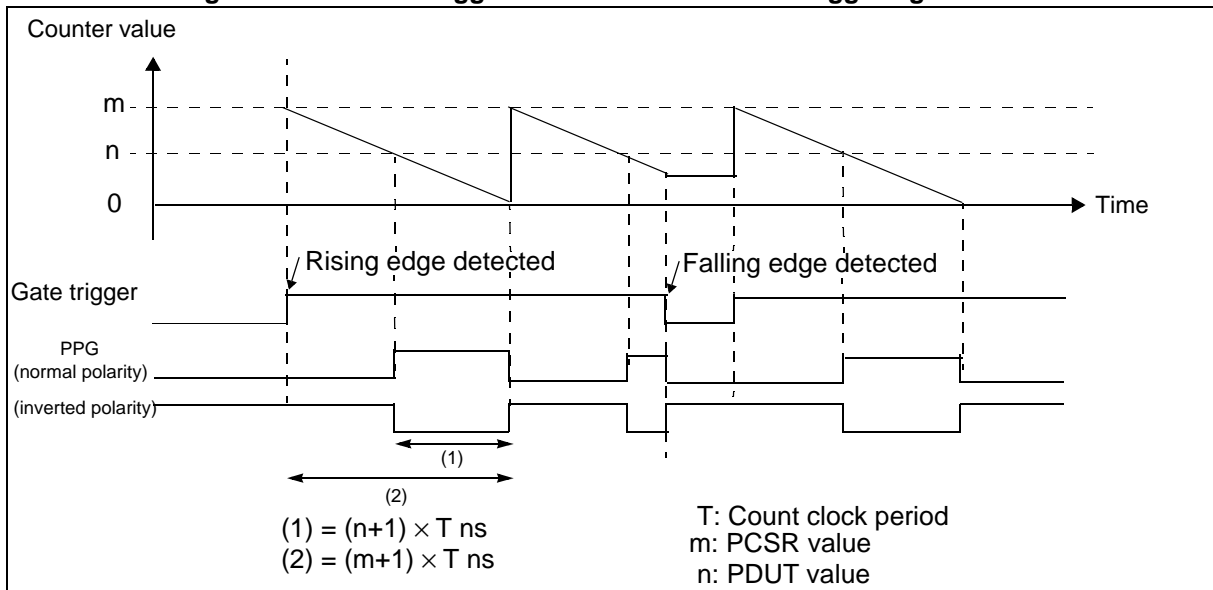
Figure 14.6-4 Retriggering is enabled in single-shot mode



#### ■ Gate Trigger (PPG channel 0 only)

When gate trigger is used, PPG starts operation when rising edge of gate trigger is detected and stops when falling is detected. In next rising edge, PPG restarts operation.

Figure 14.6-5 Gate trigger in PWM mode when retriggering is enable

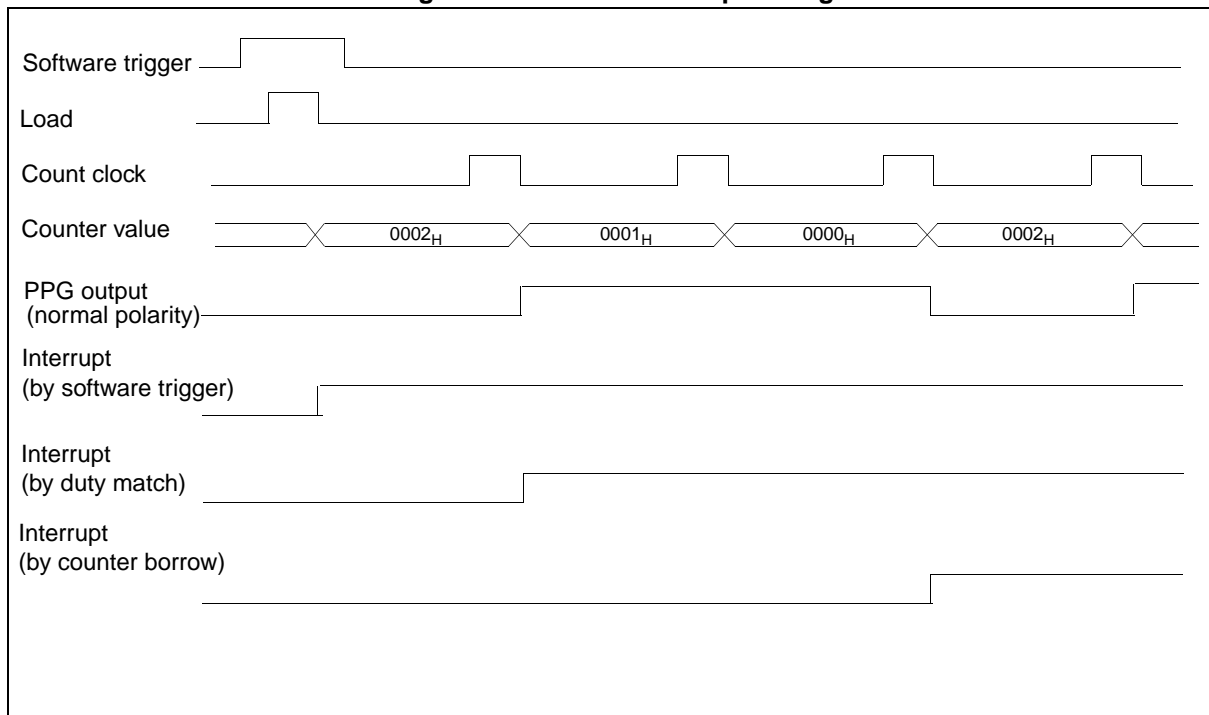


## ■ PPG Interrupts

There are four types of interrupts sharing one interrupt flag (PCNTL:IRQF) selected by interrupt type selection bits (PCNTL:IRS1 and IRS0).

- Gate trigger (for PPG channel 0 only), software trigger, or retrigger
- Counter borrow
- Duty match occurs when PPG output rising in normal polarity or PPG output falling in inverted polarity
- Counter borrow or duty match

**Figure 14.6-6 PPG interrupt timing**





## **14.7 Usage Notes on the 16-bit PPG Timer**

---

**Notes on using the 16-bit PPG timer are given below.**

---

### **■ Usage Notes on the 16-bit PPG Timer**

#### ● Notes on using a program for setting

- When write a value to the PPG period setting buffer register (PCSR), PPG duty setting buffer register (PDUT) must be written after writing to PCSR. Only updating PCSR is prohibited. Be sure to use a word transfer instruction (MOVW A, dir, etc.) to access PCSR and PDUT.
- Always set the value of PPG duty setting buffer register (PDUT) not greater than PPG period setting buffer register (PCSR), otherwise the output of PPG is undefined.
- Change the CKS2, CKS1 and CKS0 bits of the PPG control status register (PCNTH) when the PPG is stopped (PCNTH: CNTE=0).

#### ● Notes about interrupts

- When the IRQF bit of the PPG control status register (PCNTL) is set to "1" and an interrupt request is enabled (PCNTL: IREN = 1), control cannot be returned from interrupt processing. Always clear the IRQF bit.
- Since the 16-bit PPG timer shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.

Also, when EI<sup>2</sup>OS is used by the 16-bit PPG timer, shared resource interrupts must be disabled.

# CHAPTER 15

---

## ***MULTI-FUNCTIONAL TIMER***

**This chapter describes the functions and operation of the multi-functional timer.**

- 15.1 Overview of Multi-functional Timer
- 15.2 Block Diagram of Multi-functional Timer
- 15.3 Multi-functional Timer Pins
- 15.4 Registers of Multi-functional Timer
- 15.5 Multi-functional Timer Interrupts
- 15.6 Operation of Multi-functional Timer
- 15.7 Usage Notes on the Multi-functional Timer

## 15.1 Overview of Multi-functional Timer

---

The multi-functional timer consists of a 16-bit free-run timer, six 16-bit output compare, four 16-bit input capture, 1 channel of 16-bit PPG timer, and a waveform generator. By using this waveform generator, 12 independent waveform can be outputted through 16-bit free-run timer. Furthermore, input pulse width measurement and external clock cycle measurement can be done.

---

### ■ 16-bit Free-run Timer (x 1)

- The 16-bit free-run timer consists of a 16-bit up/up-down counter, timer control status register, 16-bit compare clear register (with buffer register) and a prescaler.
- 8 types of counter operation clock ( $\phi$ ,  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ ,  $\phi/32$ ,  $\phi/64$ ,  $\phi/128$ ) can be selected ( $\phi$  is the machine clock).
- Compare clear interrupt is generated when there is a compare match with compare clear register and 16-bit free-run timer. Zero detection interrupt is generated while 16-bit free-run timer is detected as zero in count value.
- The compare clear register has a selectable buffer register, into which data is written for transfer to the compare clear register. When 16-bit free-run the timer is stopped, transfer occurs immediately when the data is written to the buffer. When the timer is in operation, data transfer from the buffer occurs when the timer value is detected to be zero.
- Reset, software clear, compare match with compare clear register in up-count mode will reset the counter value to "0000<sub>H</sub>".
- The output value of this counter can be used as the count clock of the output compares and input captures in multi-functional timer.

### ■ 16-bit Output Compare (x 6)

- The 16-bit output compare consists of six 16-bit output compare registers (with selectable buffer register), compare output latch, and compare control registers. An interrupt is generated and output level is inverted when the value of 16-bit free-run timer and output compare register are matched.
- 6 output compare registers can be operated independently.

Output pins and interrupt flag are corresponding to each output compare register.

- 2 output compare registers can be paired to control the output pins.

Inverts output pins by using 2 output compare registers together.

- Setting the initial value for each output pin is possible.
- An interrupt is generated when output compare register is matched with 16-bit free-run timer.

### ■ 16-bit Input Capture (x 4)

Input capture consists of 4 independent external input pins, the corresponding input capture register, and input capture control register. By detecting any edge of the input signal from the external pin, the value of the 16-bit free-run timer can be stored in the capture register and an interrupt is generated simultaneously.

- 3 types of trigger edge (rising edge, falling edge, and both edges) of the external input signal can be selected and there is indication bit to show the trigger edge is rising or falling.
- 4 input captures can be operated independently.
- An interrupt is generated by detecting a valid edge from external input.
- Channel 0 and 1 share interrupt #33.
- Channel 2 and 3 share interrupt #35.

#### ■ 16-bit PPG Timer (x 1)

The 16-bit PPG timer 0 is used to provide a PPG signal for waveform generator. The detail of 16-bit PPG timer 0 is described in "CHAPTER 14 16-BIT PPG TIMER".

#### ■ Waveform Generator

The waveform generator consists of three 16-bit timer registers, three timer control registers, and one 16-bit waveform control register.

With waveform generator, it is possible to generate realtime output, 16-bit PPG waveform output, non-overlap 3-phase waveform output for inverter control and DC chopper waveform output.

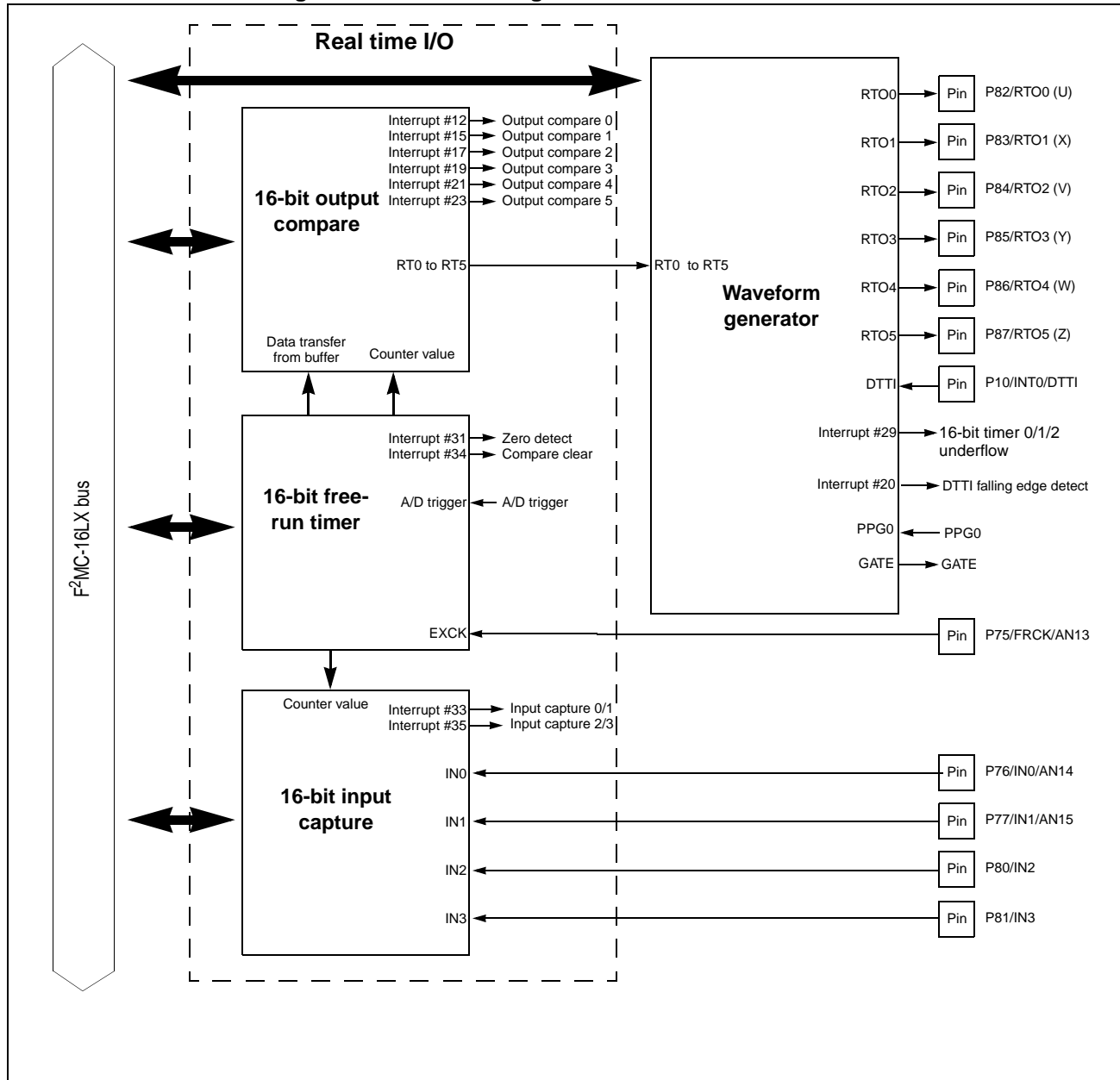
- It is possible to generate a non-overlap waveform output based on dead-time of 16-bit timer. (Dead-time timer function)
- It is possible to generate a non-overlap waveform output when realtime output is operated in 2-channel mode. (Dead-time timer function)
- By detecting realtime output compare match, GATE signal of the PPG timer operation will be generated to start or stop PPG timer operation. (GATE function)
- When a match is detected by realtime output compare, the 16-bit timer is activated. The PPG timer can be started or stopped easily by generating a GATE signal for PPG operation. (GATE function)
- Forced stop control using DTTI pin input

## 15.2 Block Diagram of Multi-functional Timer

The block diagram of the multi-functional timer will be described in the following sections.

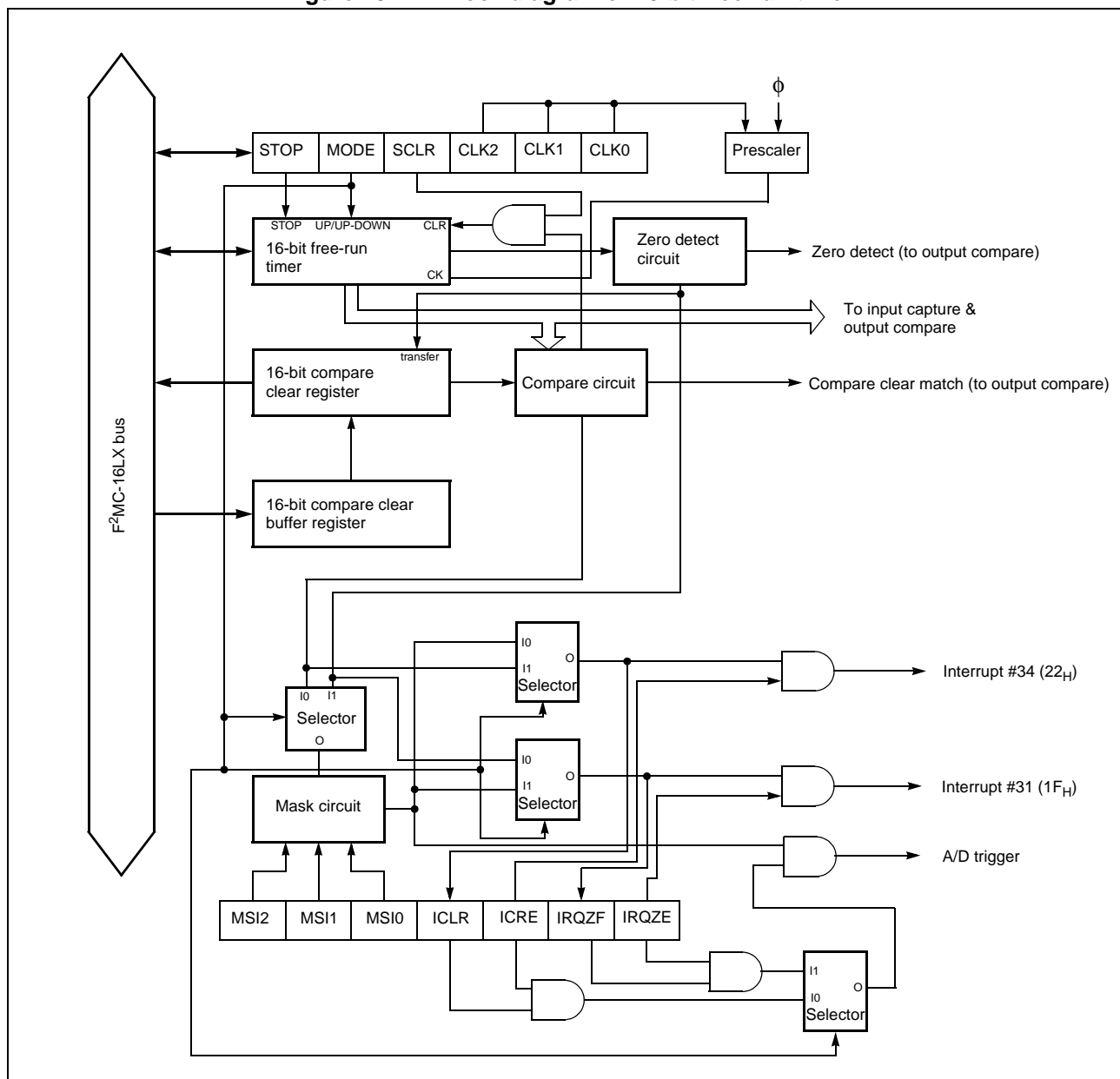
### ■ Block Diagram of Multi-functional Timer

Figure 15.2-1 Block diagram of multi-functional timer



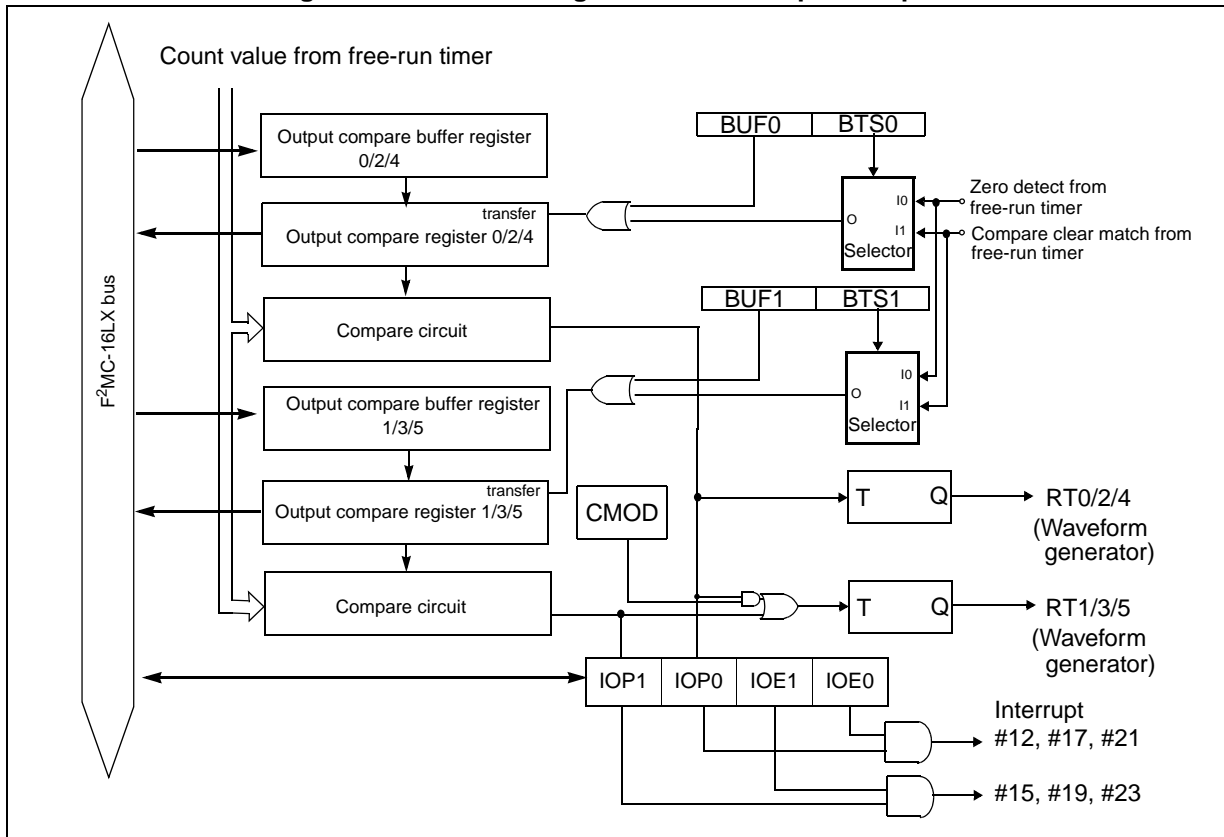
## ■ Block Diagram of 16-bit Free-run Timer

Figure 15.2-2 Block diagram of 16-bit free-run timer



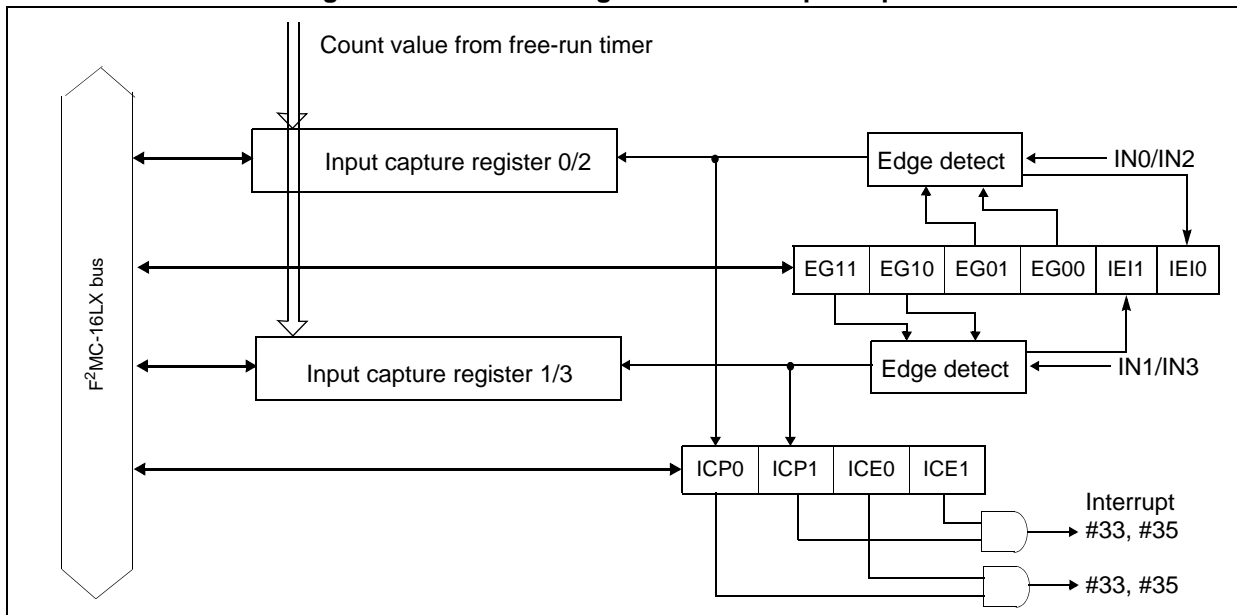
■ **Block Diagram of 16-bit Output Compare**

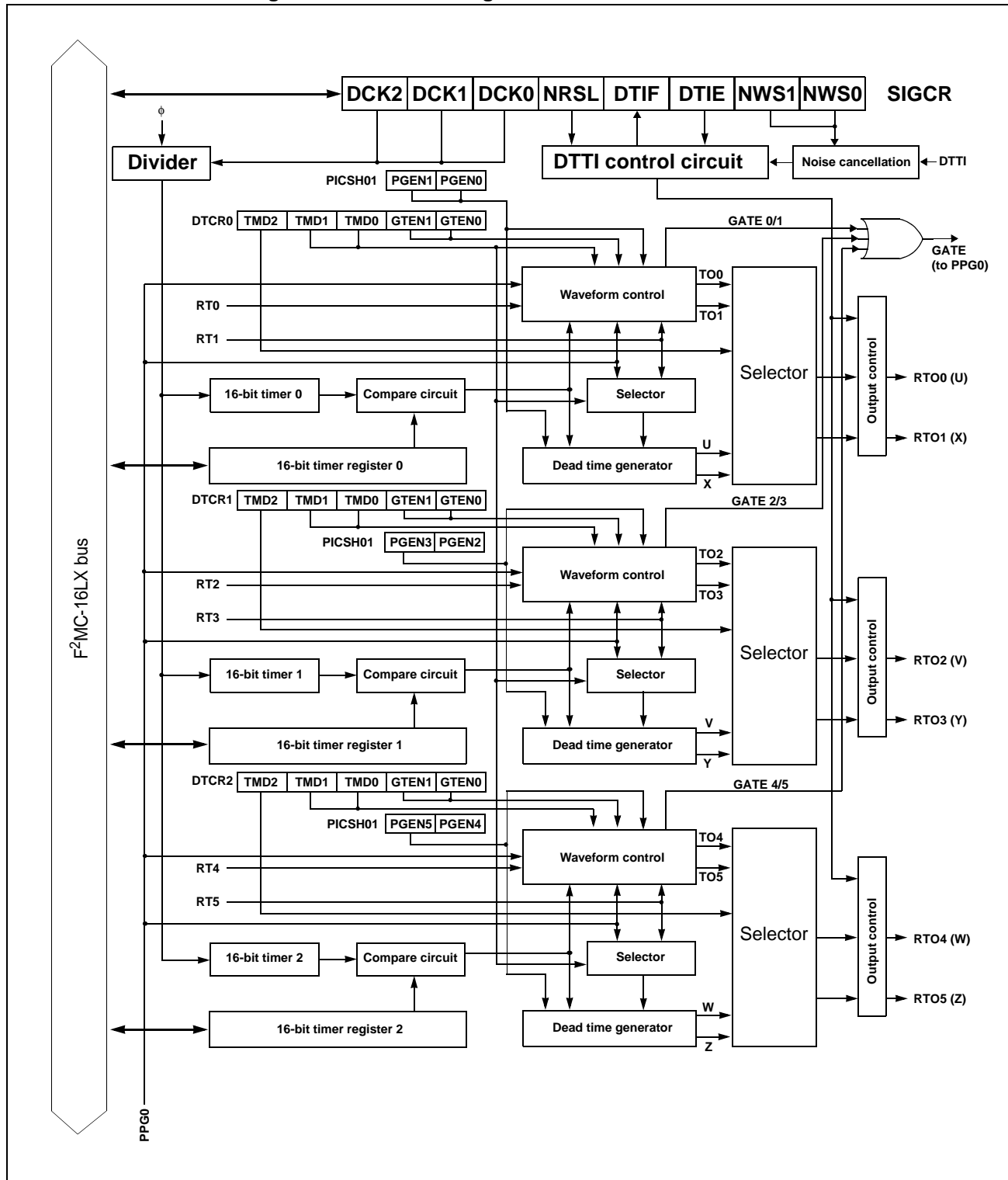
**Figure 15.2-3 Block diagram of 16-bit output compare**



■ **Block Diagram of 16-bit Input Capture**

**Figure 15.2-4 Block diagram of 16-bit input capture**



**MB90820B Series****■ Block Diagram of Waveform Generator****Figure 15.2-5 Block Diagram of Waveform Generator**



## 15.3 Multi-functional Timer Pins

This section describes the pins of the multi-functional timer and provides a pin block diagram.

### ■ Multi-functional Timer Pins

Table 15.3-1 Multi-functional timer pins

Pin Name	Pin function	I/O format	Pull-up option	Standby control	Setting required for pins
P10/INT0/ DTTI	Port 1 input-output/external interrupt input/ DTTI	CMOS output/ CMOS hysteresis input	Selectable	Provided	Set the pin as an input port (DDR1:bit 8 = 0)
P75/FRCK/ AN13	Port 7 input-output/external clock		Not provided		Set the pin as an input port (DDR7:bit 13 = 0)
P76/IN0/ AN14	Port 7 input-output/input capture 0				Set the pin as an input port (DDR7:bit 14 = 0)
P77/IN1/ AN15	Port 7 input-output/input capture 1				Set the pin as an input port (DDR7:bit 15 = 0)
P80/IN2	Port 8 input-output/input capture 2				Set the pin as an input port (DDR8:bit 0 = 0)
P81/IN3	Port 8 input-output/input capture 3				Set the pin as an input port (DDR8:bit 1 = 0)
P82/RTO0 (U)	Port 8 input-output/RTO0				Set RTO0 output (OCS1:OTE0 = 1)
P83/RTO1 (X)	Port 8 input-output/RTO1				Set RTO1 output (OCS1:OTE1 = 1)
P84/RTO2 (V)	Port 8 input-output/RTO2				Set RTO2 output (OCS3:OTE0 = 1)
P85/RTO3 (Y)	Port 8 input-output/RTO3				Set RTO3 output (OCS3:OTE1 = 1)
P86/RTO4 (W)	Port 8 input-output/RTO4				Set RTO4 output (OCS5:OTE0 = 1)
P87/RTO5 (Z)	Port 8 input-output/RTO5				Set RTO5 output (OCS5:OTE1 = 1)

# MB90820B Series

## ■ Block Diagram of Multi-functional Timer Pins

Figure 15.3-1 Block diagram of P10/INT0/DTTI

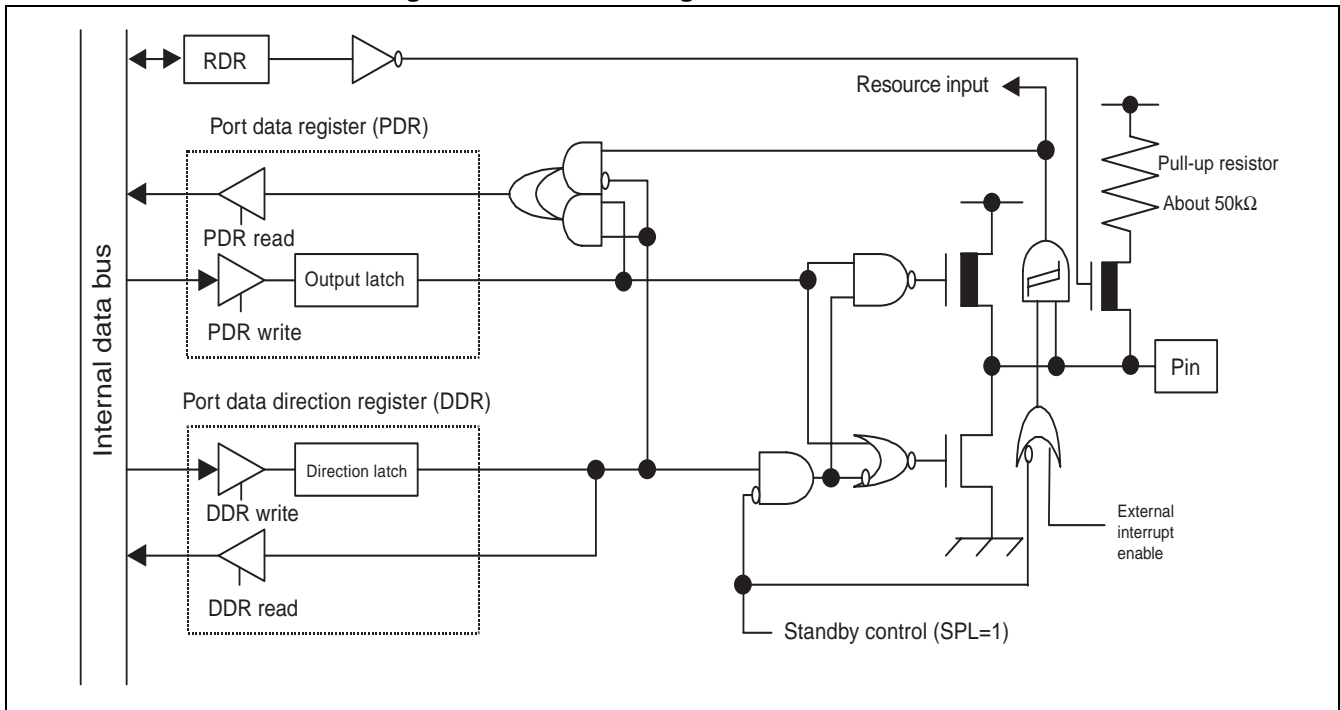
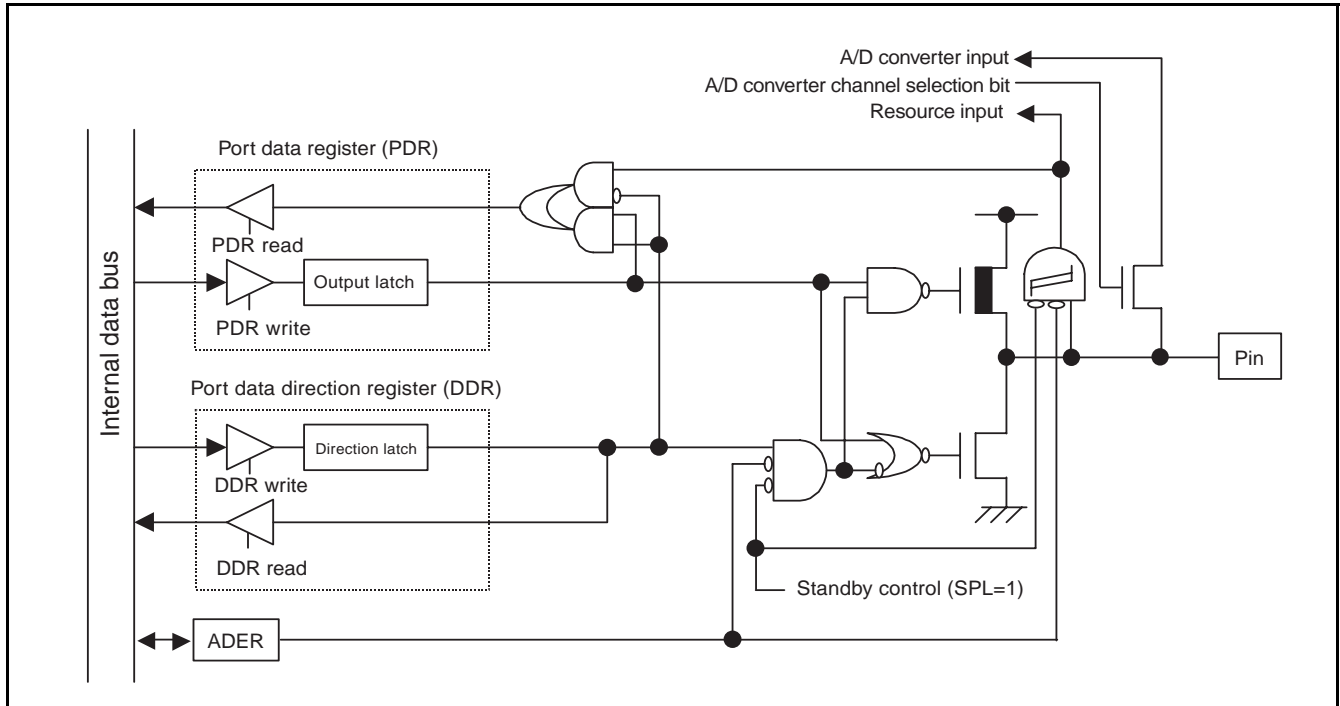
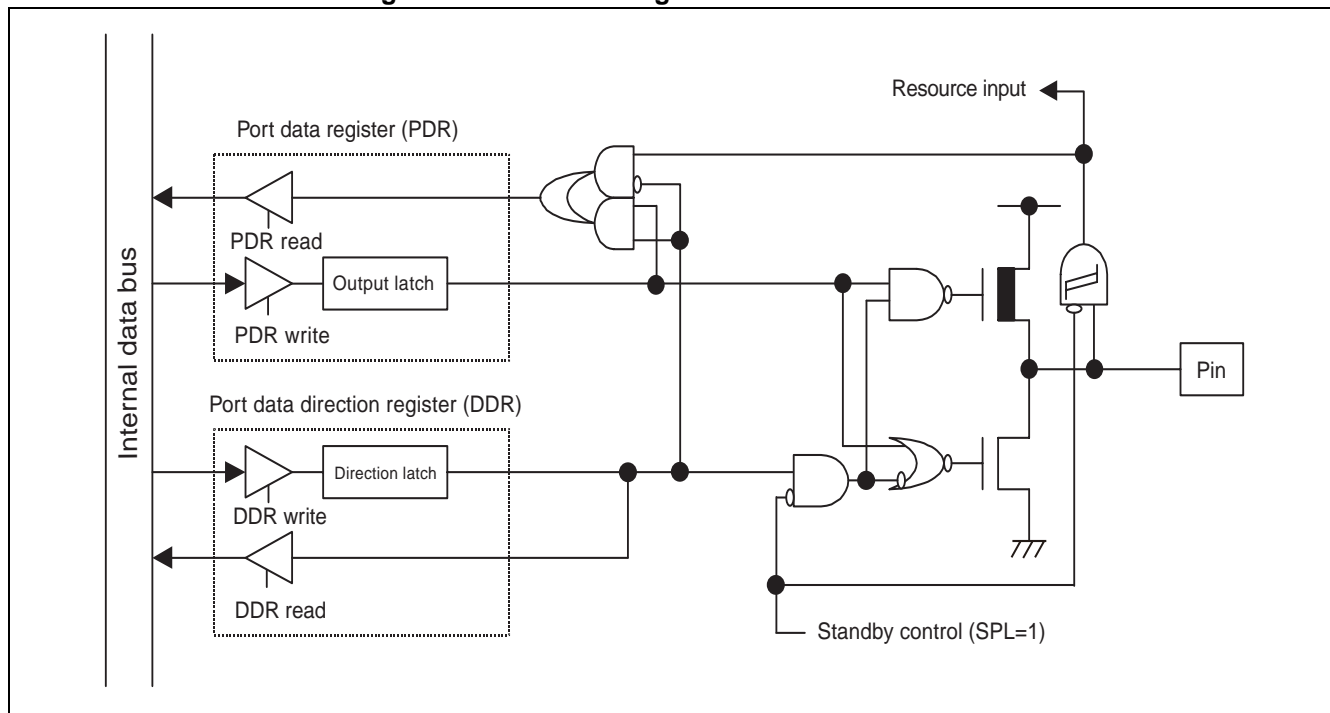


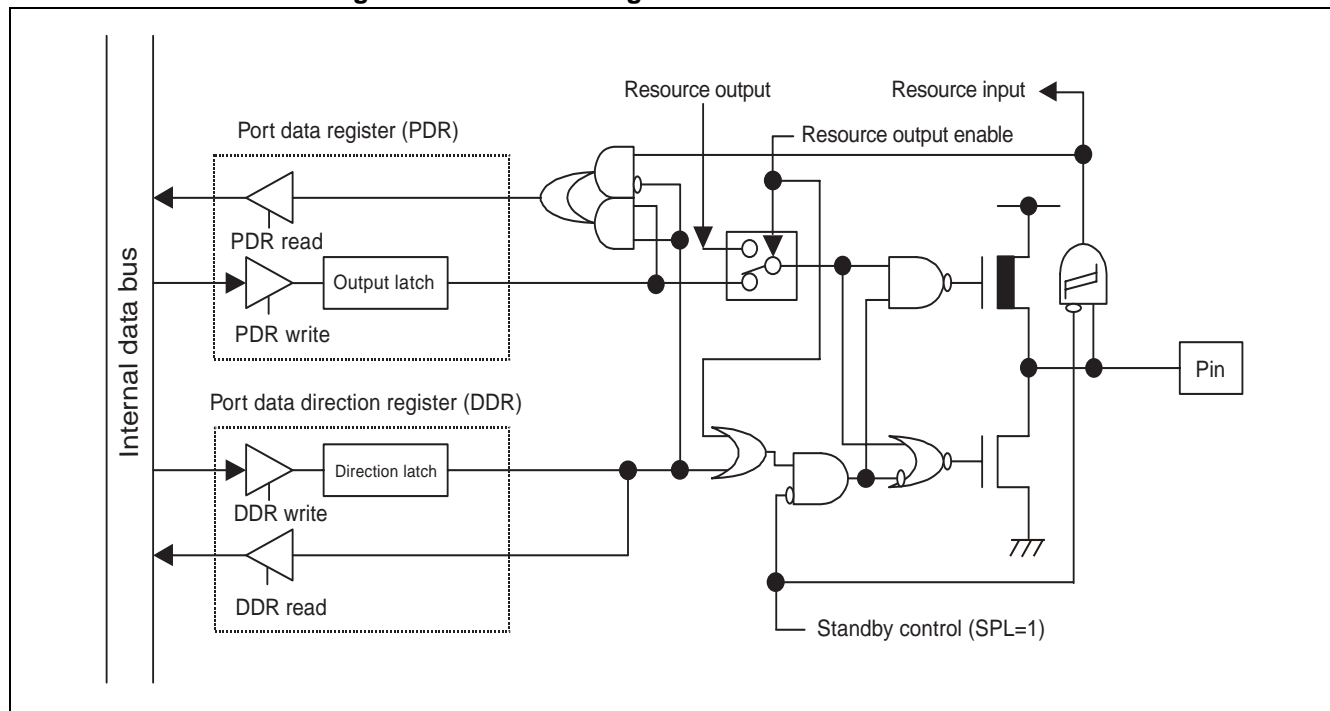
Figure 15.3-2 Block diagram of P75/FRCK/AN13 to P77/IN1/AN15



**Figure 15.3-3 Block diagram of P80/IN2 to P81/IN3**



**Figure 15.3-4 Block diagram of P82/RT00 to P87/RT05**



**MB90820B Series****15.4 Registers of Multi-functional Timer**

This section describes registers of multi-functional timer.

■ 16-bit Free-run Timer Registers

Figure 15.4-1 Registers of 16-bit free-run timer

**Compare Clear Buffer Register / Compare Clear Register (Upper)**

bit	15	14	13	12	11	10	9	8	
Address: 00005B <sub>H</sub>	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	CPCLRB/CPCLR
Read/write ⇒	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value ⇒	1	1	1	1	1	1	1	1	

**Compare Clear Buffer Register / Compare Clear Register (Lower)**

bit	7	6	5	4	3	2	1	0	
Address: 00005A <sub>H</sub>	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	CPCLRB/CPCLR
Read/write ⇒	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value ⇒	1	1	1	1	1	1	1	1	

**Timer Data Register (Upper)**

bit	15	14	13	12	11	10	9	8	
Address: 00005D <sub>H</sub>	T15	T14	T13	T12	T11	T10	T09	T08	TCDT
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)**

bit	7	6	5	4	3	2	1	0	
Address: 00005C <sub>H</sub>	T07	T06	T05	T04	T03	T02	T01	T00	TCDT
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 Control Status Register (Upper)**

bit	15	14	13	12	11	10	9	8	
Address: 00005F <sub>H</sub>	ECKE	IRQZF	IRQZE	MSI2	MSI1	MSI0	ICLR	ICRE	TCCSH
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	

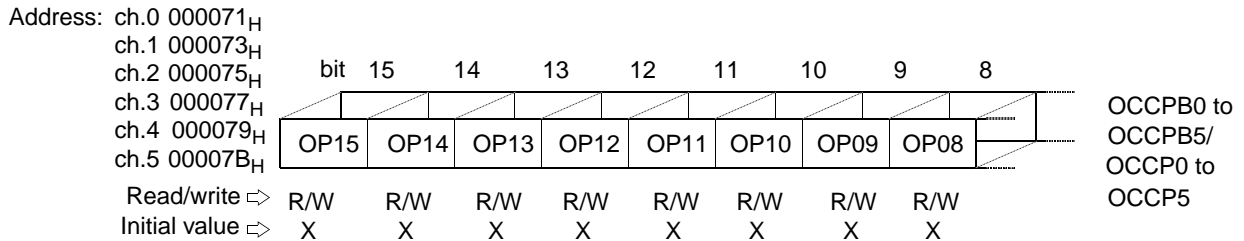
**Timer Control Status Register (Lower)**

bit	7	6	5	4	3	2	1	0	
Address: 00005E <sub>H</sub>	—	BFE	STOP	MODE	SCLR	CLK2	CLK1	CLK0	TCCSL
Read/write ⇒	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value ⇒	X	0	1	0	0	0	0	0	

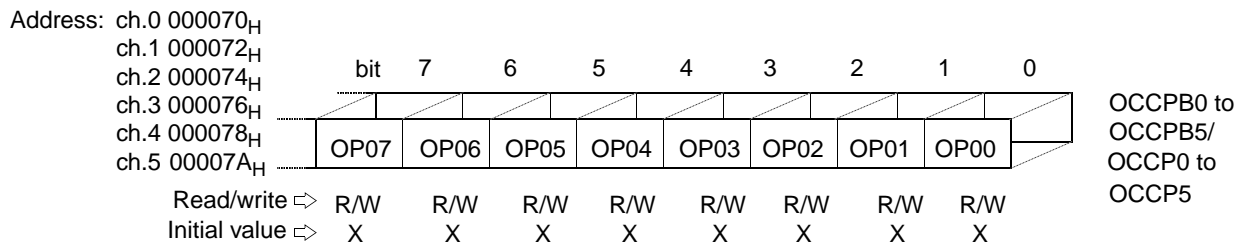
## ■ 16-bit Output Compare Registers

Figure 15.4-2 Registers of 16-bit output compare

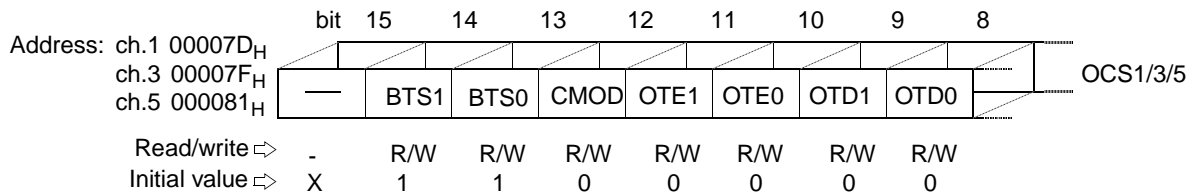
### Output Compare Buffer Register / Output Compare Register (Upper)



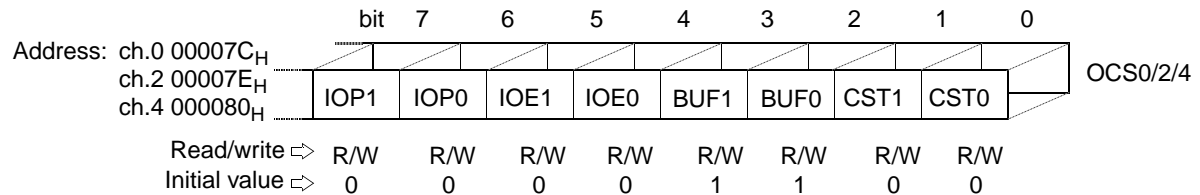
### Output Compare Buffer Register / Output Compare Register (Lower)



### Compare Control Register (Upper)



### Compare Control Register (Lower)



# MB90820B Series

## Input Capture Registers

Figure 15.4-3 Registers of 16-bit input capture

### Input Capture Data Register (Upper)

Address: ch.0 000061 <sub>H</sub>	bit 15	14	13	12	11	10	9	8	
ch.1 000063 <sub>H</sub>									
ch.2 000065 <sub>H</sub>									
ch.3 000067 <sub>H</sub>									
	CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08	IPCP0 to IPCP3
Read/write ⇒	R	R	R	R	R	R	R	R	
Initial value ⇒	X	X	X	X	X	X	X	X	

### Input Capture Data Register (Lower)

Address: ch.0 000060 <sub>H</sub>	bit 7	6	5	4	3	2	1	0	
ch.1 000062 <sub>H</sub>									
ch.2 000064 <sub>H</sub>									
ch.3 000066 <sub>H</sub>									
	CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00	IPCP0 to IPCP3
Read/write ⇒	R	R	R	R	R	R	R	R	
Initial value ⇒	X	X	X	X	X	X	X	X	

### Input Capture Control Status Register (2/3) (Upper)

bit	15	14	13	12	11	10	9	8	
Address: 00006B <sub>H</sub>	—	—	—	—	—	—	IEI3	IEI2	ICSH23
Read/write ⇒	—	—	—	—	—	—	R	R/W	
Initial value ⇒	X	X	X	X	X	X	0	0	

### Input Capture Control Status Register (2/3) (Lower)

bit	7	6	5	4	3	2	1	0	
Address: 00006A <sub>H</sub>	ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20	ICSL23
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	

### PPG output control/ Input Capture Control Status Register (0/1) (Upper)

bit	15	14	13	12	11	10	9	8	
Address: 000069 <sub>H</sub>	PGEN5	PGEN4	PGEN3	PGEN2	PGEN1	PGEN0	IEI1	IEI0	PICSH01
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	

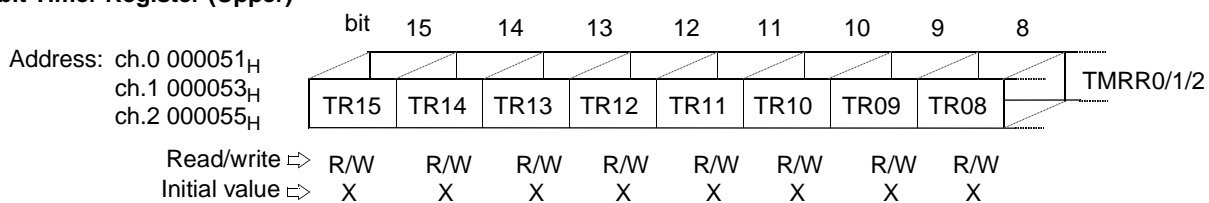
### Input Capture Control Register (0/1) (Lower)

bit	7	6	5	4	3	2	1	0	
Address: 000068 <sub>H</sub>	ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	PICSL01
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	

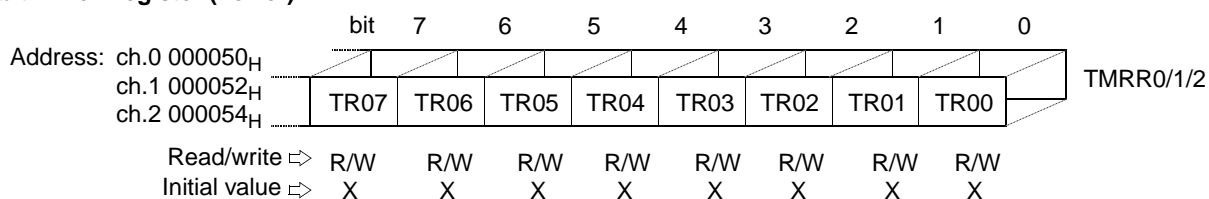
## ■ Waveform Generator Registers

Figure 15.4-4 Registers of waveform generator

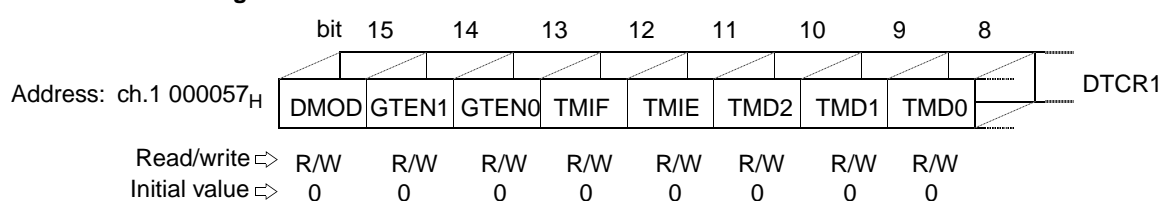
### 16-bit Timer Register (Upper)



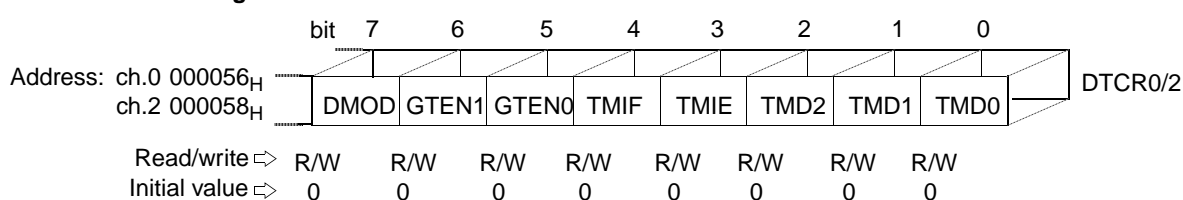
### 16-bit Timer Register (Lower)



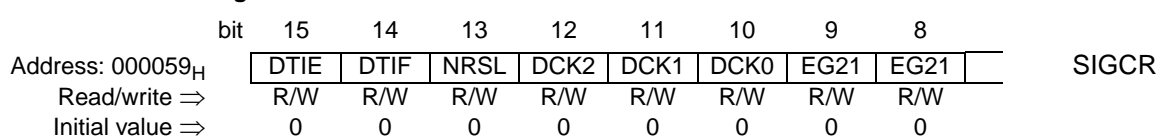
### 16-bit Timer Control Register



### 16-bit Timer Control Register



### Waveform Control Register



**MB90820B Series****15.4.1 Compare Clear Buffer Register (CPCLRB) and Compare Clear Register (CPCLR)**

Compare clear buffer register (CPCLRB) is a 16-bit buffer register of compare clear register (CPCLR). Both CPCLRB and CPCLR registers are located in the same address.

**■ Compare Clear Buffer Register (CPCLRB)****Figure 15.4-5 Compare clear buffer register (CPCLRB)**

<b>Compare Clear Buffer Register (Upper)</b>									
	bit	15	14	13	12	11	10	9	8
Address: 00005B <sub>H</sub>		CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08
Read/write ⇒		W	W	W	W	W	W	W	W
Initial value ⇒		1	1	1	1	1	1	1	1
<b>Compare Clear Buffer Register (Lower)</b>									
	bit	7	6	5	4	3	2	1	0
Address: 00005A <sub>H</sub>		CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00
Read/write ⇒		W	W	W	W	W	W	W	W
Initial value ⇒		1	1	1	1	1	1	1	1

Compare clear buffer register is the buffer register for compare clear register. When buffer function is disabled (TCCSL:BFE=0) or when free-run timer is stopped, value in compare clear buffer register is transferred to compare clear register immediately. When buffer function is enabled, value is transferred to the register when the count value of 16-bit free-run timer is detected as zero.

Word access instruction to this register is recommended.

**■ Compare Clear Register (CPCLR)****Figure 15.4-6 Compare clear register (CPCLR)**

<b>Compare Clear Register (Upper)</b>									
	bit	15	14	13	12	11	10	9	8
Address: 00005B <sub>H</sub>		CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08
Read/write ⇒		R	R	R	R	R	R	R	R
Initial value ⇒		1	1	1	1	1	1	1	1
<b>Compare Clear Register (Lower)</b>									
	bit	7	6	5	4	3	2	1	0
Address: 00005A <sub>H</sub>		CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00
Read/write ⇒		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 up-count mode, when this register is matched with the count value of 16-bit free-run timer, timer will be reset to "0000<sub>H</sub>". In up-down count mode, when this register is matched with the count value of the 16-bit free-run timer, the timer changes from up-count to down-count or changes from down-count to up-count at zero detect.

Word access instruction to this register is recommended.



## 15.4.2 Timer Data Register (TCDT)

The timer data register (TCDT) is used to read the count value of 16-bit free-run timer.

### ■ Timer Data Register (TCDT)

Figure 15.4-7 Timer data register

<b>Timer Data Register (Upper)</b>									
	bit	15	14	13	12	11	10	9	8
Address: 00005D <sub>H</sub>		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
<b>Timer Data Register (Lower)</b>									
	bit	7	6	5	4	3	2	1	0
Address: 00005C <sub>H</sub>		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

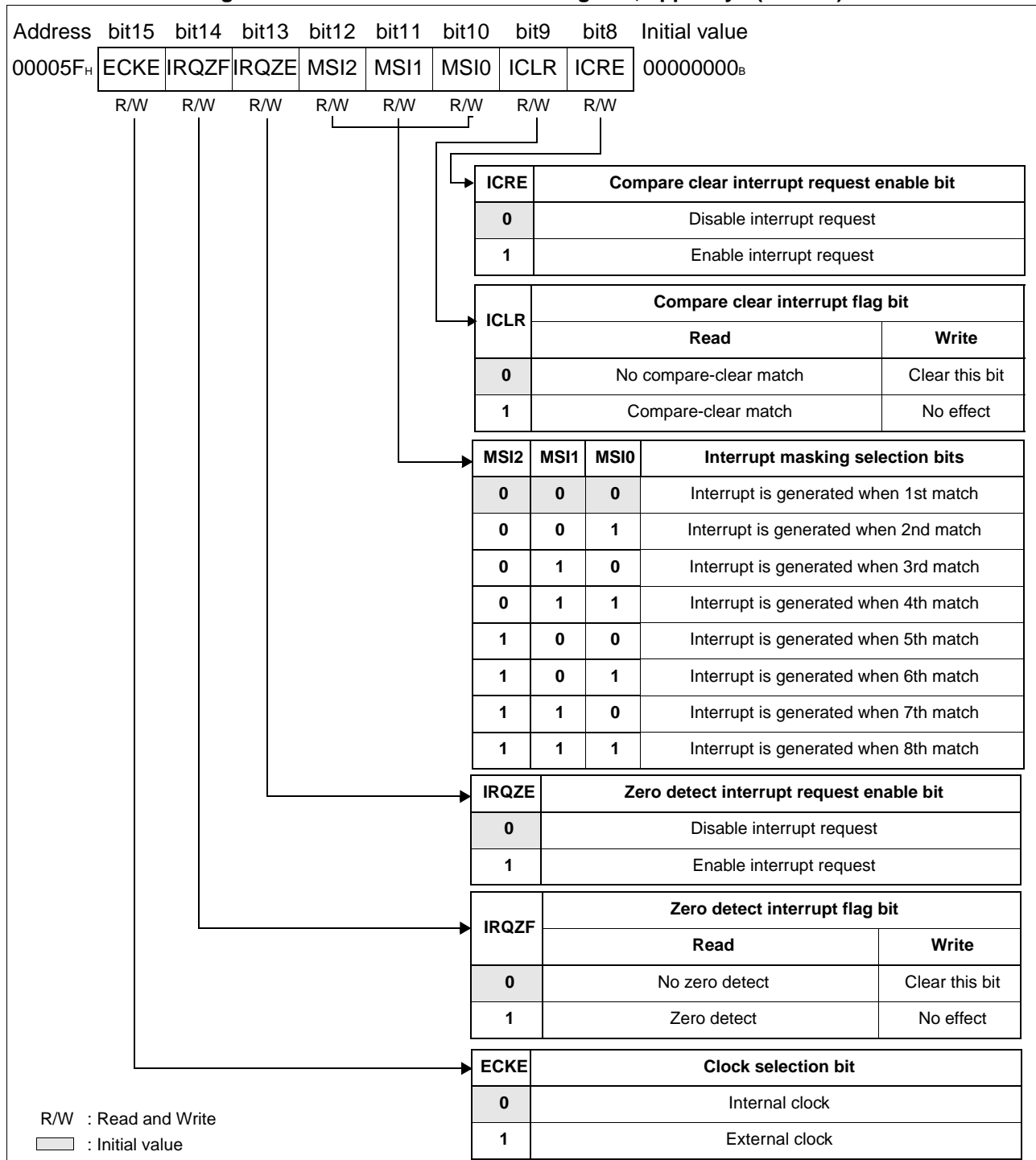
The timer data register is used to read the count value of the 16-bit free-run timer. The counter value is cleared to “0000<sub>H</sub>” upon a reset. The timer value can be set by writing a value to this register. However, ensure that the value is written while the operation is stopped (STOP = 1). Word access instruction to the timer data register is recommended.

The 16-bit free-run timer is initialized upon the following factors:

- Reset
- Clear bit (SCLR) of timer control status register
- A match between compare clear register and the timer counter value in up-count mode (TCCSL:MODE=0)

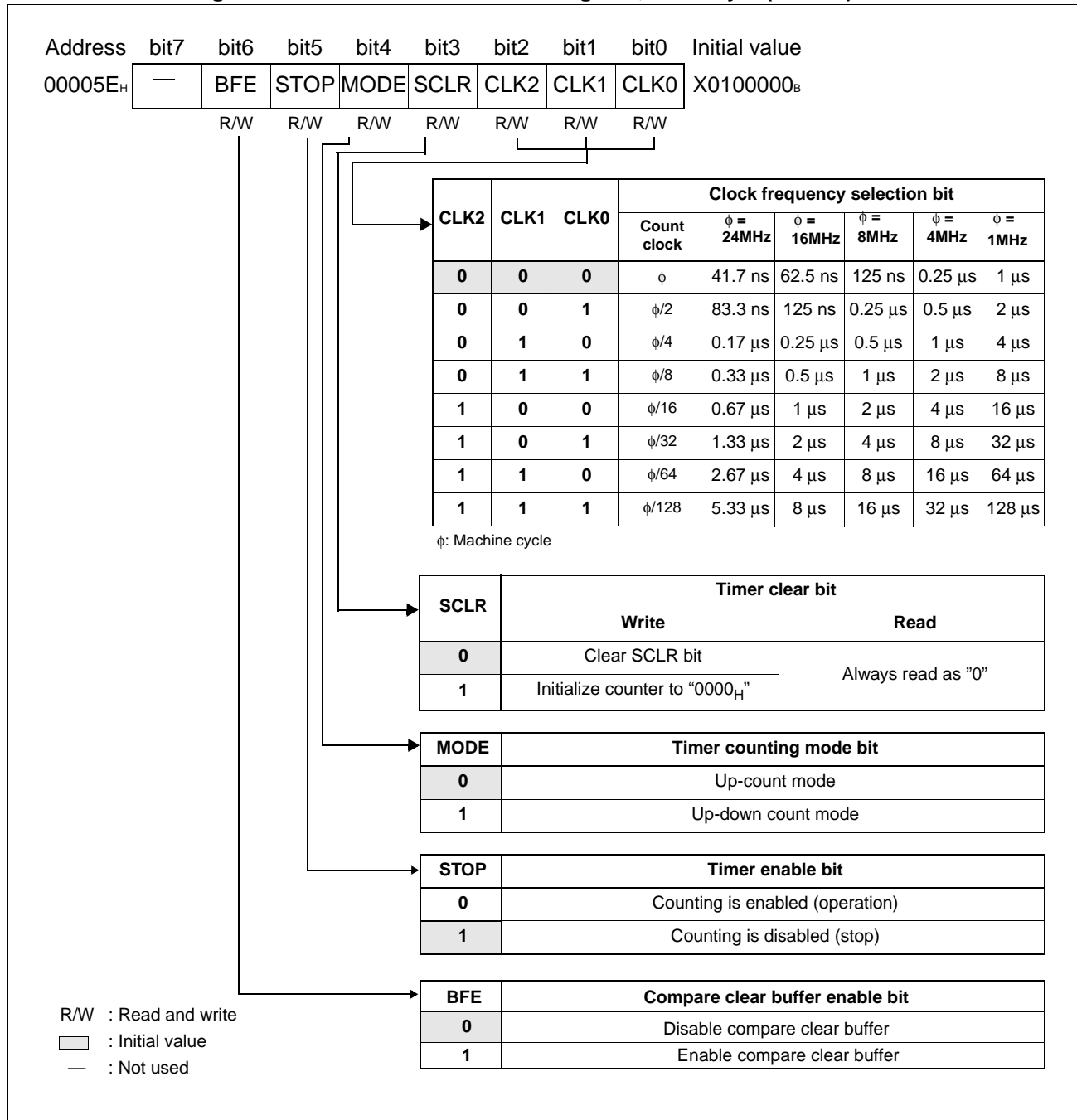
**MB90820B Series****15.4.3 Timer Control Status Register (TCCSH, TCCSL)**

The timer control status register (TCCS) is a 16-bit register and is used to control the operation of 16-bit free-run timer.

**■ Timer Control Status Register, Upper Byte (TCCSH)****Figure 15.4-8 Timer control status register, upper byte(TCCSH)**

**Table 15.4-1 Timer control status register, upper byte (TCCSH)**

Bit name		Function
bit15	ECKE: Clock selection bit	<ul style="list-style-type: none"> <li>This bit is used to select internal or external clock as count clock for 16-bit free-run timer.</li> <li>Writing "0" selects internal clock. The clock frequency selection bits (CLK2 to CLK0) should also be set to select the count clock frequency.</li> <li>Writing "1" selects external clock. External clock is input from pin "P75/FRCK/AN13", so DDR1:7 should be set as "0" to enable external clock input.</li> </ul> <p><b>Note:</b> The count clock is changed immediately after this bit is set. So change this bit while the output compare and input capture units are stopped.</p>
bit 14	IRQZF: Zero detect interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is an interrupt flag for zero detect.</li> <li>When the count value of 16-bit free-run timer is 0000<sub>H</sub>, this bit is set to "1".</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>- In software clear, (writing TCCSL:SCLR "1") will not set this bit.</li> <li>- In up-down count mode (MODE=1) and the interrupt is generated by setting interrupt mask selection bit is selected (MSI2 to MSI0 not equals 000<sub>B</sub>), this bit will only be set to "1". When the interrupt is not generated, this bit does not set "1".</li> <li>- In up-count mode (MODE=0), this bit is set at every zero detect disregarding the value of MSI2 to MSI0.</li> </ul>
bit 13	IRQZE: Zero detect interrupt request enable bit	<ul style="list-style-type: none"> <li>This is the interrupt request enable bit for the zero detect.</li> <li>When this bit is "1" and the interrupt flag bit (bit 14: IRQZF) is set to "1", an interrupt request will be generated to CPU.</li> </ul>
bit12 to bit10	MSI2 to MSI0: Interrupt mask selection bits	<ul style="list-style-type: none"> <li>These bits are used to set the number of times for masking the compare clear interrupt in up-count mode (MODE=0) or zero detect interrupt in up-down count mode (MODE=1).</li> <li>No interrupt cause is masked when MSI2 to MSI0 equals zero.</li> </ul> <p><b>Note:</b> To mask the interrupt cause twice and perform interrupt processing at the third times, MSI2 to MSI0 should be set as 010<sub>B</sub>.</p>
bit9	ICLR: Compare clear interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is an interrupt flag for compare clear.</li> <li>When the compare clear value and 16-bit free-run timer value are matched, this bit is set to "1".</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>- In up-count mode (MODE=0) and the interrupt is generated by setting interrupt mask selection bit is selected (MSI2 to MSI0 not equals 000<sub>B</sub>), this bit will only be set to "1". When the interrupt is not generated, this bit does not set "1".</li> <li>- In up-down count mode (MODE=1), this bit is set at every compare clear disregarding the value of MSI2 to MSI0.</li> </ul>
bit8	ICRE: Compare clear interrupt request enable bit	<ul style="list-style-type: none"> <li>This is the interrupt request enable bit for the compare clear.</li> <li>When this bit is "1" and the compare clear interrupt flag bit (bit 9: ICLR) is set to "1", an interrupt request will be generated to CPU.</li> </ul>

**MB90820B Series****■ Timer Control Status Register, Lower Byte (TCCSL)****Figure 15.4-9 Timer control status register, lower byte (TCCSL)**

**Table 15.4-2 Timer control status register, lower byte (TCCSL) (1 / 2)**

Bit name		Function
bit7	Unused bit	<ul style="list-style-type: none"> <li>The read value is undefined.</li> <li>Writing to this bit has no effect on the operation.</li> </ul>
bit6	BFE: Compare clear buffer enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable compare clear buffer.</li> <li>Writing "0" disables compare clear buffer. Directly write in compare clear register is possible.</li> <li>Writing "1" enables compare clear buffer. Data written in compare clear buffer register will be held and transfer to compare clear register when the count value of 16-bit free-run timer is detected as zero.</li> </ul>
bit5	STOP: Timer enable bit	<ul style="list-style-type: none"> <li>This bit is used to stop/start the counting of the 16-bit free-run timer.</li> <li>Writing "0" starts the counting of the 16-bit free-run timer.</li> <li>Writing "1" stops the counting of the 16-bit free-run timer.</li> </ul> <p><b>Note:</b> When the 16-bit free-run timer is stopped, the output compare operation will also be stopped.</p>
bit4	MODE: Timer counting mode bit	<ul style="list-style-type: none"> <li>This bit is used to select the count mode of the 16-bit free-run timer.</li> <li>Writing "0" selects up-count mode. Timer counts up until counter value matches with compare clear register and resets to "0000<sub>H</sub>" and then counts up again.</li> <li>Writing "1" selects up-down count mode. In up-down count mode, whenever zero in the counter data register is detected, the timer count mode will always be reseted to up-counting. The timer will reverse its count mode whenever the counter value matches with compare clear register.</li> <li>This bit can be written at any time whether the timer is operating or stopped. The value written to this bit is buffered during the timer is operating and the count mode will be changed when timer value is "0000<sub>H</sub>".</li> </ul> <p><b>Note:</b> Becasue the timer will reverse its count mode when compare-match is detected in up-down count mode (MODE = 1), it should be careful to set the compare clear register and timer data register when the timer is being counted down.</p>

Table 15.4-2 Timer control status register, lower byte (TCCSL) (2 / 2)

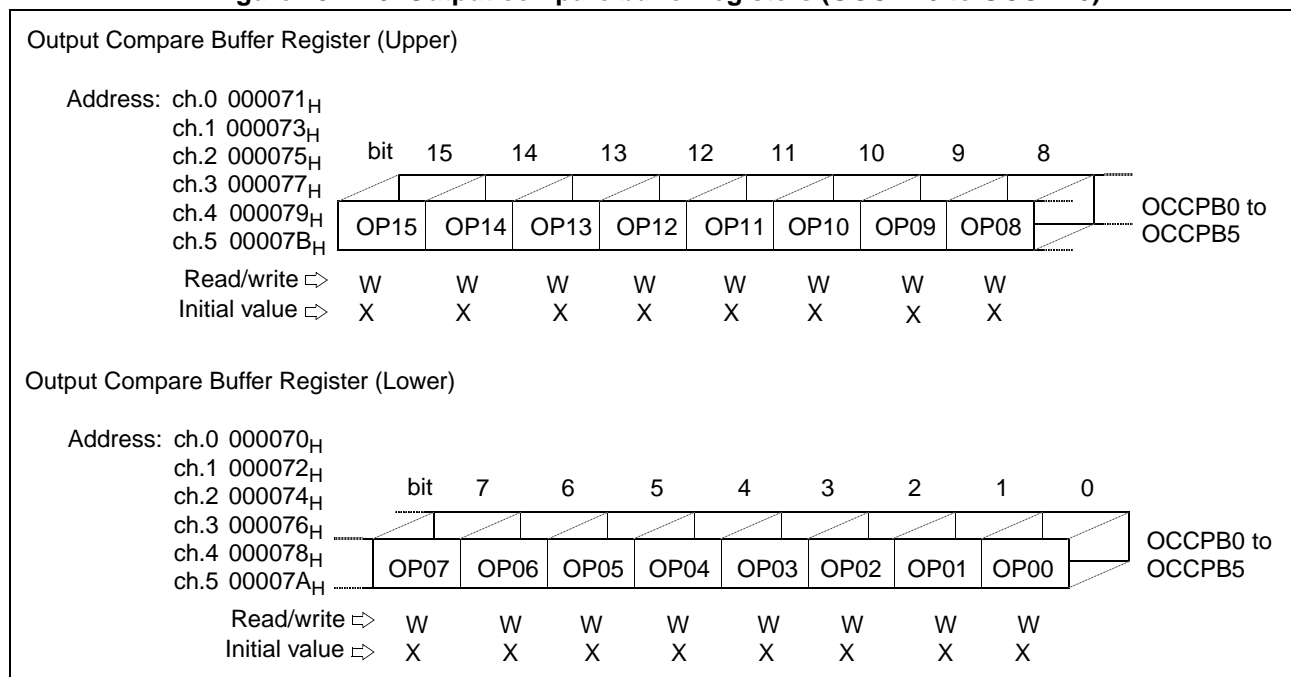
Bit name		Function
bit3	SCLR: Timer clear bit	<ul style="list-style-type: none"> <li>This bit is used to initialize the 16-bit free-run timer to "0000<sub>H</sub>".</li> <li>Writing "0" will clear the SCLR bit if it is "1".</li> <li>Writing "1" initializes 16-bit free-run timer to "0000<sub>H</sub>" at the next count clock.</li> <li>Read value is always "0".</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>- This bit cannot be used to initialize the timer when timer stops (STOP=1). Writing "0000<sub>H</sub>" to timer data register (TCDT) can initialize the timer.</li> <li>- Writing "1" will not generate zero detect interrupt.</li> <li>- This bit will be cleared by hardware after the timer is initialized to "0000". If "0" is written to the bit before timer initialization, the bit is cleared and the timer did not initialize.</li> </ul>
bit2 to bit0	CLK2 to CLK0: Clock frequency selection bits	<ul style="list-style-type: none"> <li>This bit is used to select count clock frequency for the 16-bit free-run timer.</li> <li>The count clock is changed immediately after these bits are set. So change them while the output compare and input capture units are stopped.</li> </ul>

### 15.4.4 Output Compare Buffer Registers (OCCPB0 to OCCPB5)/ Output Compare Registers (OCCP0 to OCCP5)

Output compare buffer register (OCCPB) is a 16-bit buffer register of output compare register (OCCP). Both OCCPB and OCCP registers are located in the same address.

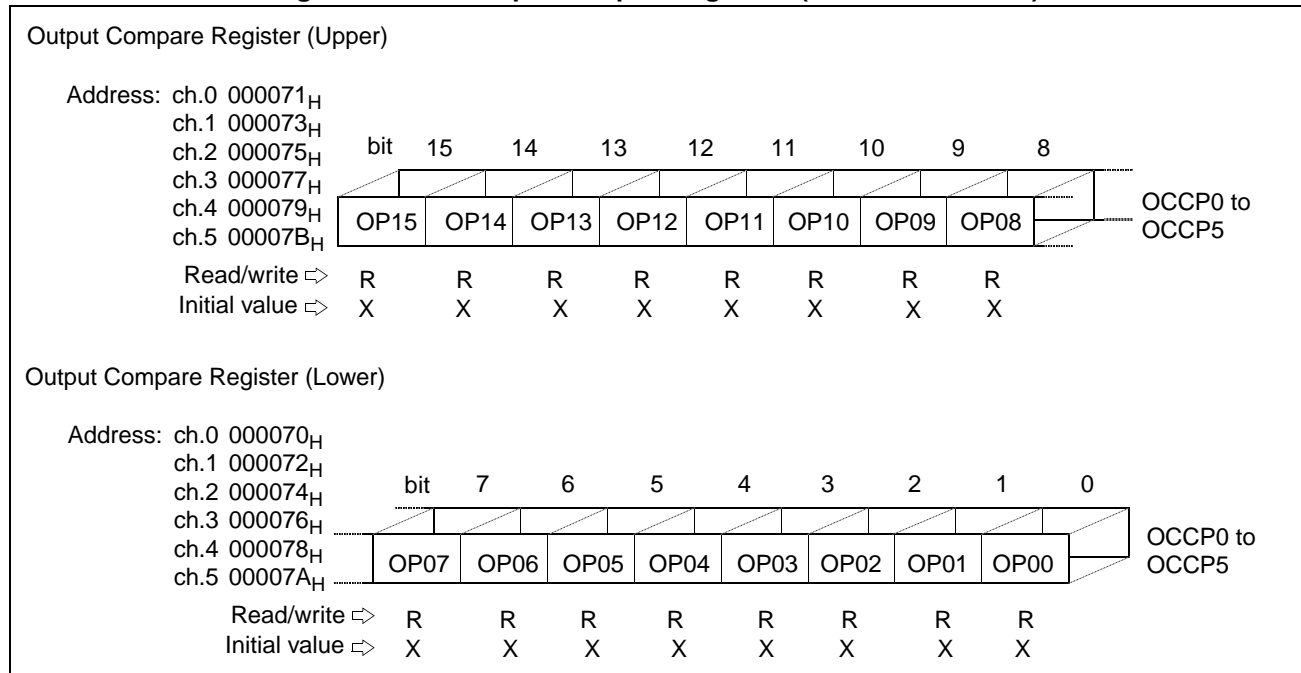
#### ■ Output Compare Buffer Registers (OCCPB0 to OCCPB5)

Figure 15.4-10 Output compare buffer registers (OCCPB0 to OCCPB5)



Output compare buffer register is the buffer register of output compare register (OCCP). When buffer function is disabled (OCS0/2/4:BUF0/1=1) or when free-run timer is stopped, value in output compare buffer register is transferred to output compare register immediately. When buffer function is enabled (OCS0/2/4:BUF0/1=0), value is transferred at compare clear match or zero detection depending on transfer selection bit (BTs) in compare control register (OCS1/3/5).

Word access to instruction this register is recommended.

**MB90820B Series****■ Output Compare Registers (OCCP0 to OCCP5)****Figure 15.4-11 Output compare registers (OCCP0 to OCCP5)**

The output compare register is a 16-bit register which is used to compare the count value of 16-bit free-run timer. The initial value of the output compare register is undetermined, so output compare buffer register (OCCPB) must be set with a value before enabling the timer operation.

When the value of the output compare register matches the count value of 16-bit free-run timer, a compare signal is generated to set the output compare interrupt flag bit (OCS0/OCS2/OCS4:IOP0/IOP1). If output level is set (OCS1/OCS3/OCS5:OTD0/OTD1), the output level waveform generator RT0 to RT5 corresponding to the output compare register (OCCP0 to OCCP5) can be reversed.

Word access instruction to this register is recommended.



### 15.4.5 Compare Control Registers (OCS0 to OCS5)

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 for RTO0 to RTO5.

#### ■ Compare Control Register, Upper Byte (OCS1/3/5)

Figure 15.4-12 Compare control register, upper byte (OCS1/3/5)

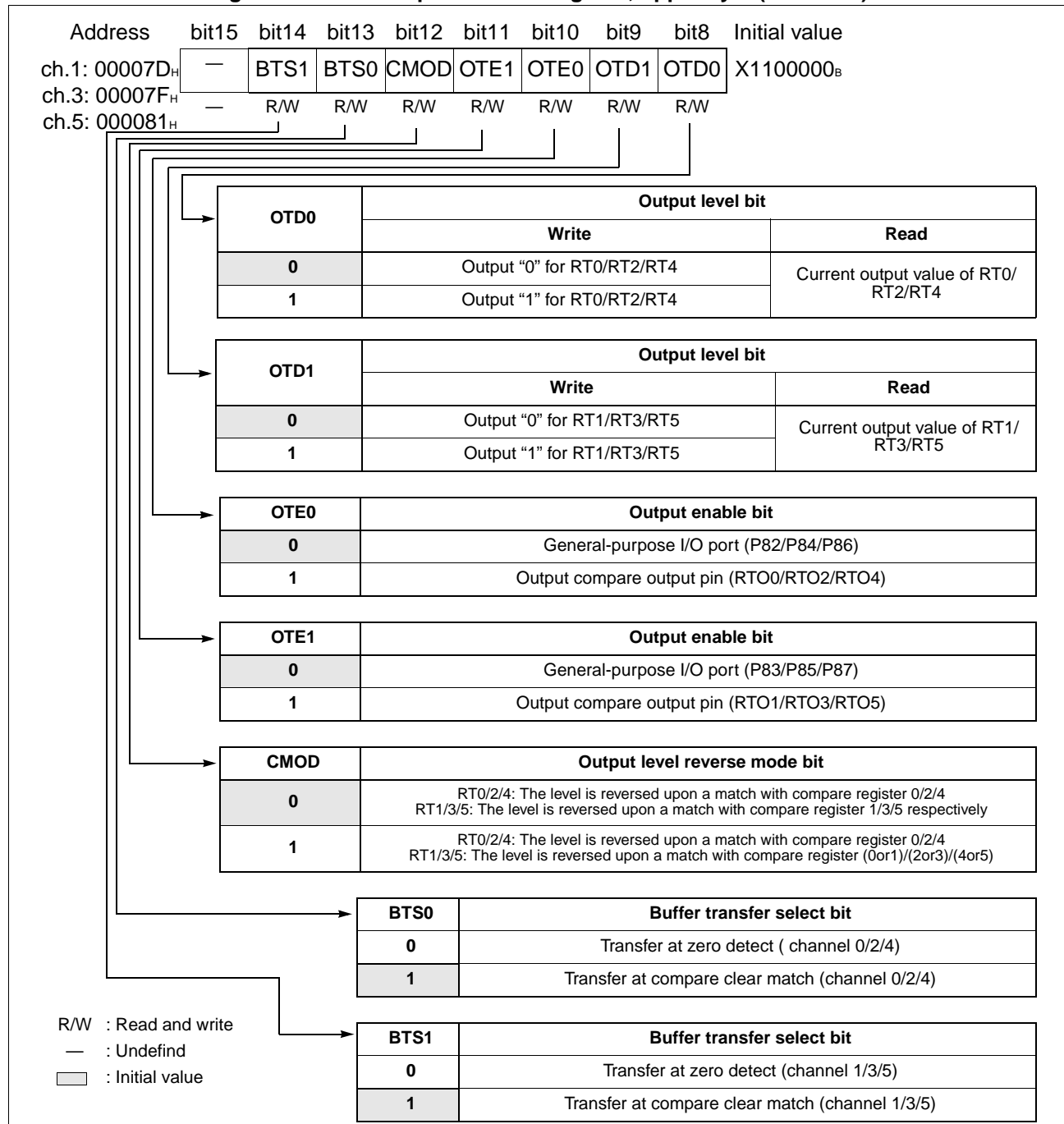


Table 15.4-3 Compare control register, upper byte (OCS1/3/5) (1 / 2)

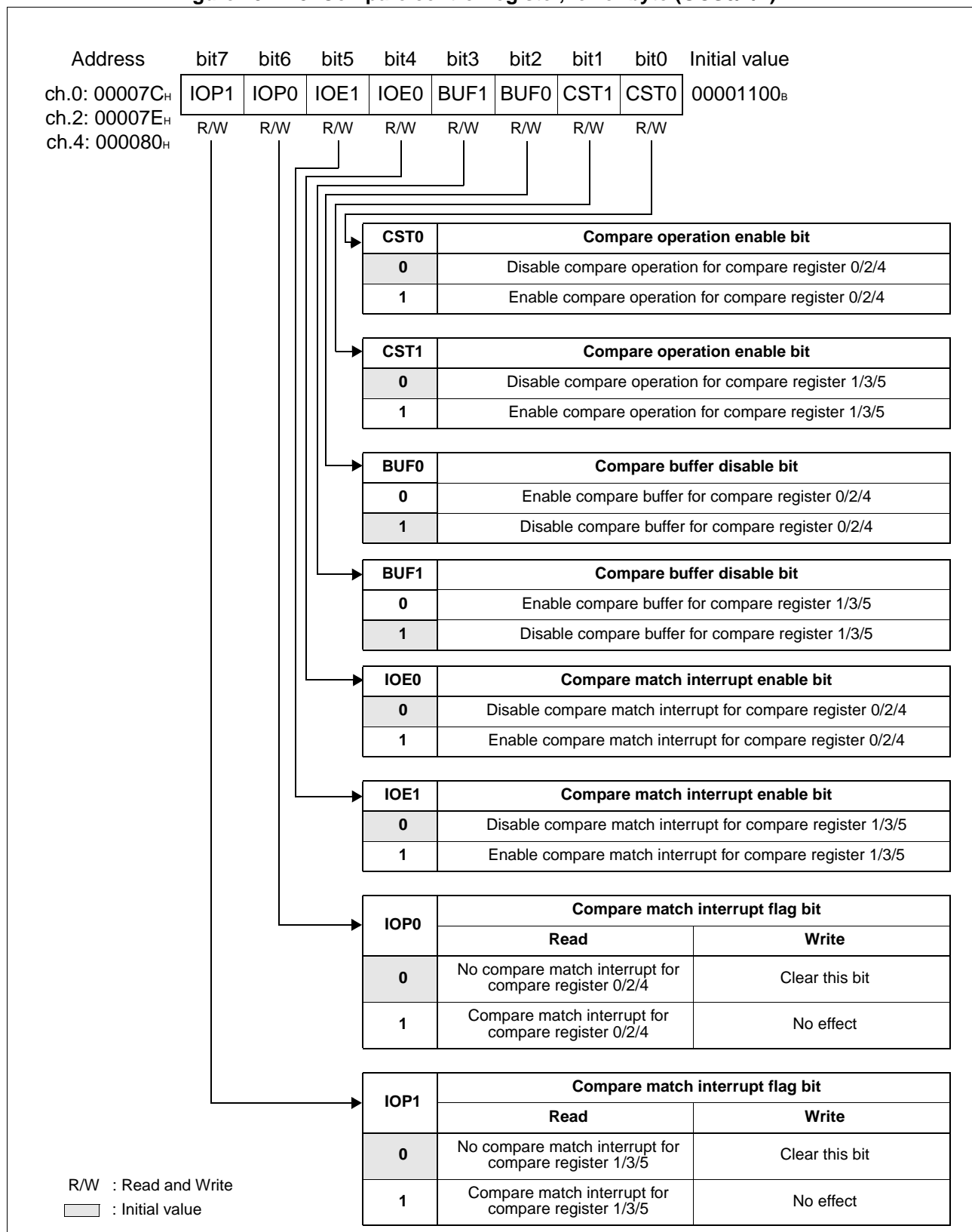
Bit name		Function
bit15	Unfind bit	<ul style="list-style-type: none"> <li>The read value is undefined.</li> <li>Writing to this bit has no effect on the operation.</li> </ul>
bit14	BTS1: Buffer transfer select bit	<ul style="list-style-type: none"> <li>This bit is used to select when data transfer from output compare buffer register (OCCPB1/3/5) to output compare register (OCCP1/3/5).</li> <li>When BTS1=0, data transfer is occurred when count value of 16-bit free-run timer is detected as zero.</li> <li>When BTS1=1, data transfer is occurred when compare clear match is occurred in 16-bit free-run timer.</li> </ul>
bit13	BTS0: Buffer transfer select bit	<ul style="list-style-type: none"> <li>This bit is used to select when data transfer from output compare buffer register (OCCPB0/2/4) to output compare register (OCCP0/2/4).</li> <li>When BTS0=0, data transfer is occurred when count value of 16-bit free-run timer is detected as zero.</li> <li>When BTS0=1, data transfer is occurred when compare clear match is occurred in 16-bit free-run timer.</li> </ul>
bit12	CMOD: Output level reverse mode bit	<ul style="list-style-type: none"> <li>CMOD is used to switch the pin output level reverse mode upon a match while pin output is enabled (OTE1 = 1 or OTE0 = 1).</li> <li>When CMOD = 0, the output level of the pin is reversed upon a match with corresponding compare register. <ul style="list-style-type: none"> <li>RT0/2/4: The level is reversed upon a match between the 16-bit free-run timer and compare register 0/2/4.</li> <li>RT1/3/5: The level is reversed upon a match between the 16-bit free-run timer and compare register 1/3/5.</li> </ul> </li> <li>When CMOD = 1, the output level of the pin RT0/2/4 corresponding to compare register is reversed as same as when CMOD = 0. However, the output level of the pin (RT1/3/5) corresponding to compare register 1/3/5 is reversed when a match is detected in compare register 0/2/4 or 1/3/5. If compare registers 0/2/4 and 1/3/5 have the same value, the same operation as when only one compare register is used. <ul style="list-style-type: none"> <li>RT0/2/4: The level is reversed upon a match between the 16-bit free-run timer and compare register 0/2/4.</li> <li>RT1/3/5: The level is reversed upon a match between the 16-bit free-run timer and compare register (0 or 1)/(2 or 3)/(4 or 5).</li> </ul> </li> </ul>
bit11	OTE1: Output enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable waveform generator output (RTO1/3/5 to P83/P85/P87) to the port.</li> <li>The initial value for this bit is "0".</li> </ul> <p><b>Note:</b> If waveform generator is disabled (DTCR:TMD2 to TMD0=000<sub>B</sub>), RTO1/3/5 outputs the same value in output compare RT1/3/5.</p>
bit10	OTE0: Output enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable waveform generator output (RTO0/2/4 to P82/P84/P86) to the port.</li> <li>The initial value for this bit is "0".</li> </ul> <p><b>Note:</b> If waveform generator is disabled (DTCR:TMD2 to TMD0=000<sub>B</sub>), RTO0/2/4 outputs the same value in output compare RT0/2/4.</p>

**Table 15.4-3 Compare control register, upper byte (OCS1/3/5) (2 / 2)**

Bit name		Function
bit9	OTD1: Output level bit	<ul style="list-style-type: none"> <li>• This bit is used to change the pin output level for output compare 1/3/5 (RT1/3/5).</li> <li>• The initial value of the compare pin output is "0".</li> <li>• Ensure that the compare operation is stopped before a value is written. When reading this bit, this bit indicates the output compare value in RT1/3/5.</li> </ul>
bit8	OTD0: Output level bit	<ul style="list-style-type: none"> <li>• This bit is used to change the pin output level for output compare 0/2/4 (RT0/2/4).</li> <li>• The initial value of the compare pin output is "0".</li> <li>• Ensure that the compare operation is stopped before a value is written. When reading this bit, this bit indicates the output compare value in RT0/2/4.</li> </ul>

# Compare Control Register, Lower Byte (OCS0/2/4)

Figure 15.4-13 Compare control register, lower byte (OCS0/2/4)

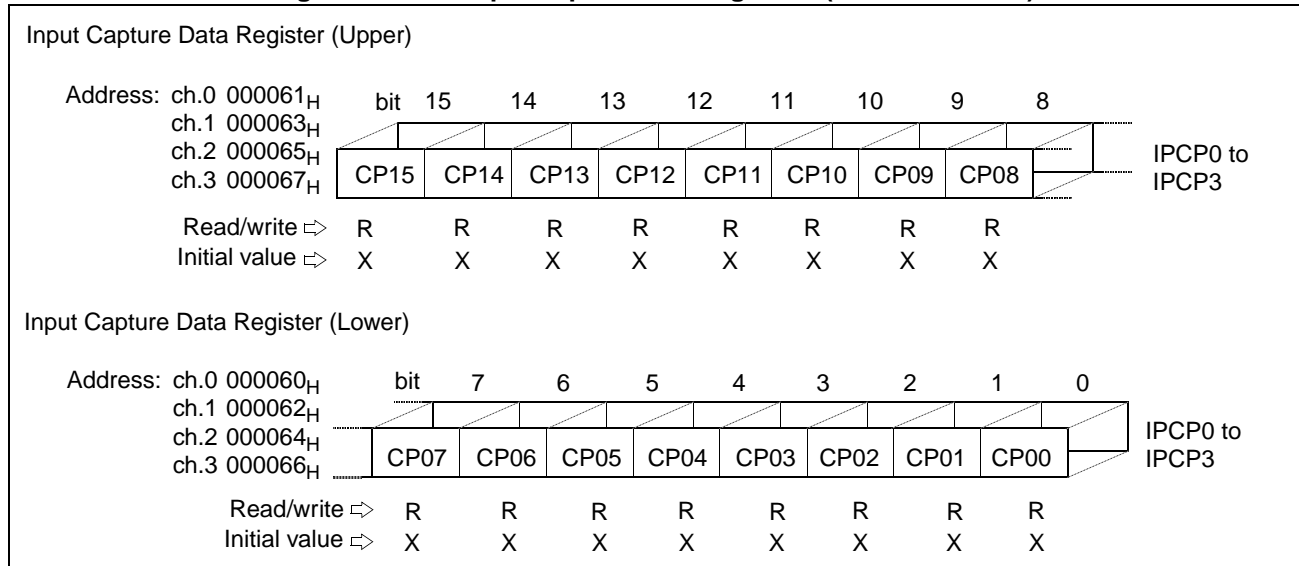


**Table 15.4-4 Compare control register, lower byte (OCS0/2/4)**

Bit name		Function
bit7	IOP1: Compare match interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is an interrupt flag when compare register 1/3/5 is matched with the value of 16-bit free-run timer.</li> <li>"1" is set to this bit when the compare register value matches the 16-bit free-run timer value.</li> <li>While the compare match interrupt enable bits (IOE1) is enabled, an output compare interrupt occurs when the IOP1 bit is set.</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul>
bit6	IOP0: Compare match interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is an interrupt flag when compare register 0/2/4 is matched with the value of 16-bit free-run timer.</li> <li>"1" is set to this bit when the compare register value matches the 16-bit free-run timer value.</li> <li>While the compare match interrupt enable bits (IOE0) is enabled, an output compare interrupt occurs when the IOP0 bit is set.</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul>
bit5	IOE1: Compare match interrupt enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable output compare interrupt for compare register 1/3/5.</li> <li>While the "1" is written to this bit, an output compare interrupt occurs when an compare match interrupt flag bit (IOP1) is set.</li> </ul>
bit4	IOE0: Compare match interrupt enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable output compare interrupt for compare register 0/2/4.</li> <li>While the "1" is written to this bit, an output compare interrupt occurs when an compare match interrupt flag bit (IOP0) is set.</li> </ul>
bit3	BUF1: Compare buffer disable bit	<ul style="list-style-type: none"> <li>This bit is used to disable buffer function for output compare register 1/3/5.</li> <li>Writing "0" will enable the buffer function.</li> </ul>
bit2	BUF0: Compare buffer disable bit	<ul style="list-style-type: none"> <li>This bit is used to disable buffer function for output compare register 0/2/4.</li> <li>Writing "0" will enable the buffer function.</li> </ul>
bit1	CST1: Compare operation enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable the compare operation between 16-bit free-run timer and compare register 1/3/5.</li> <li>Ensure that a value is written into the compare register and timer data register before the compare operation is enabled.</li> </ul> <p><b>Note:</b> Since output compare is synchronized with the 16-bit free-run timer clock, stopping the 16-bit free-run timer stops compare operation.</p>
bit0	CST0: Compare operation enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable the compare operation between 16-bit free-run timer and compare register 0/2/4.</li> <li>Ensure that a value is written into the compare register and timer data register before the compare operation is enabled.</li> </ul> <p><b>Note:</b> Since output compare is synchronized with the 16-bit free-run timer clock, stopping the 16-bit free-run timer stops zero detect and compare operation.</p>

**MB90820B Series****15.4.6 Input Capture Register (IPCP0 to IPCP3)**

Input capture registers are used to hold the count value of 16-bit timer when a valid edge of the input waveform is detected.

**■ Input Capture Register (IPCP0 to IPCP3)****Figure 15.4-14 Input capture data registers (IPCP0 to IPCP3)**

This register is used to store the value of the 16-bit timer when a valid edge of the corresponding external pin input waveform is detected. (Word access instruction to this register is recommended. No data can be written to this register.)

15.4.7 Input Capture Control Status Registers (ICS23, PICS01)

Input capture control status registers (ICS23, PICS01) are used to control edge selection, interrupt request enable, and interrupt request flag and to indicate valid edge detected for input capture 0 to 3.

■ Input Capture Control Status Register, Upper Byte (ICSH23)

Figure 15.4-15 Input capture control status register, upper byte (ICSH23)

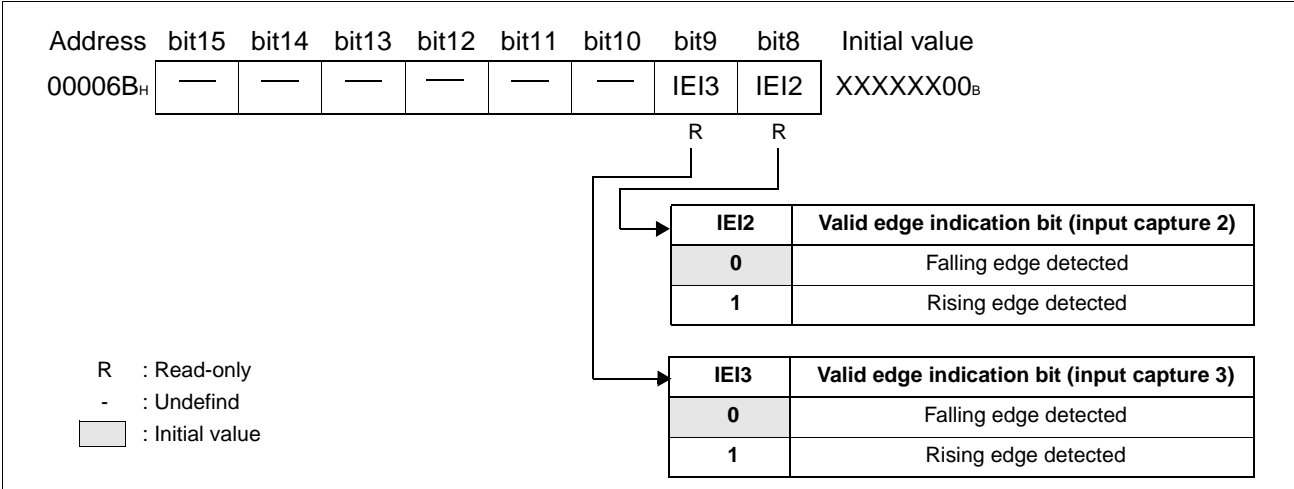


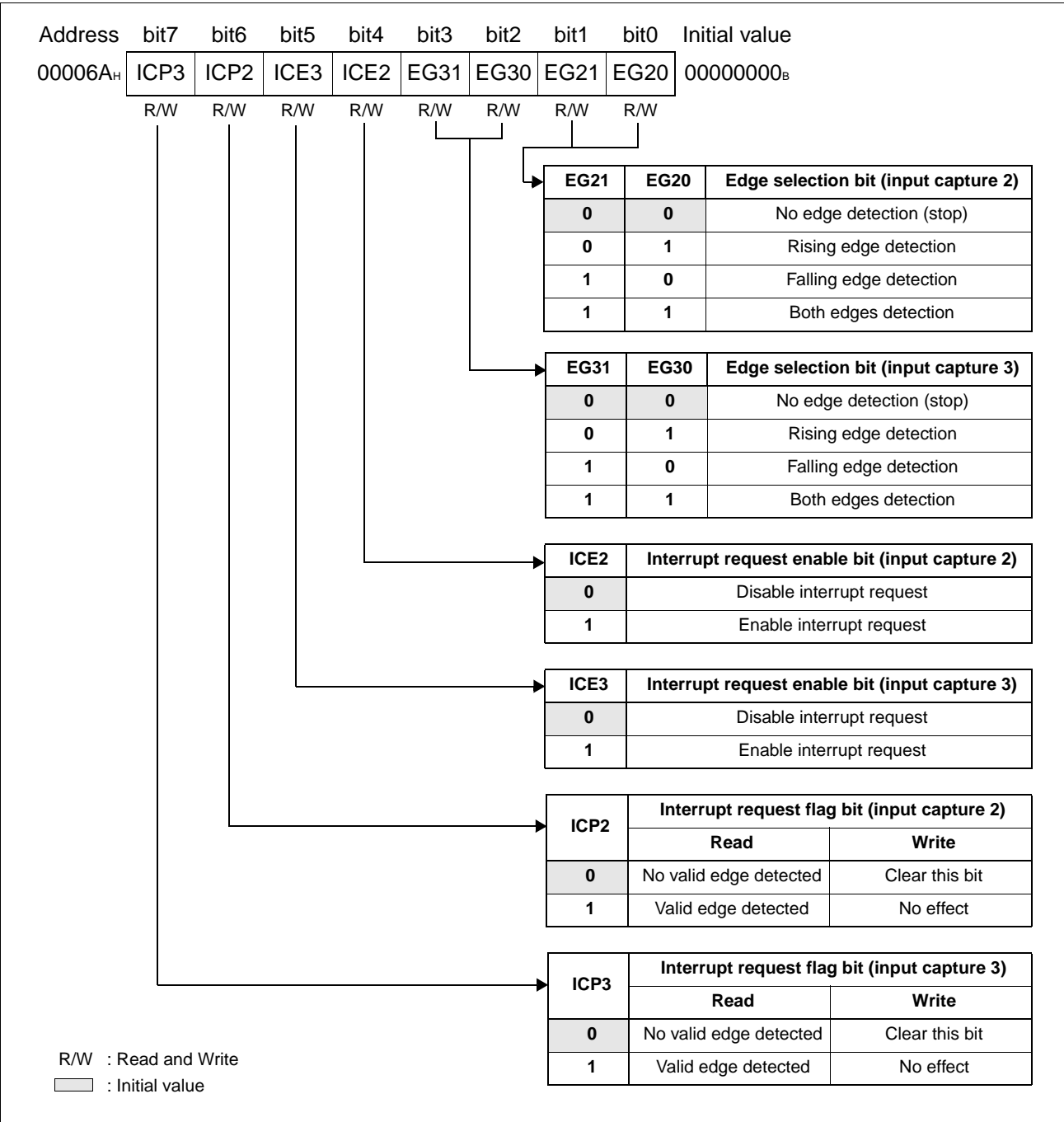
Table 15.4-5 Input capture control status register, upper byte (ICSH23)

Bit name		Function
bit15 to bit10	Undefind bits	<ul style="list-style-type: none"> <li>The read value is undefined.</li> <li>Writing to these bits have no effect on the operation.</li> </ul>
bit9	IEI3: Valid edge indication bit (input capture 3)	<ul style="list-style-type: none"> <li>This bit is an valid edge indication bit for capture register 3, to indicate a rising or falling edge is detected.</li> <li>"0" is written to this bit when falling edge is detected.</li> <li>"1" is written to this bit when rising edge is detected.</li> <li>This bit is read-only.</li> </ul> <p><b>Note:</b> The read value is meaningless when EG31, EG30 = 00<sub>B</sub>.</p>
bit8	IEI2: Valid edge indication bit (input capture 2)	<ul style="list-style-type: none"> <li>This bit is an valid edge indication bit for capture register 2, to indicate a rising or falling edge is detected.</li> <li>"0" is written to this bit when falling edge is detected.</li> <li>"1" is written to this bit when rising edge is detected.</li> <li>This bit is read-only.</li> </ul> <p><b>Note:</b> The read value is meaningless when EG21, EG20 = 00<sub>B</sub>.</p>



■ Input Capture Control Status Register, Lower Byte (ICSL23)

Figure 15.4-16 Input capture control status register, lower byte (ICSL23)

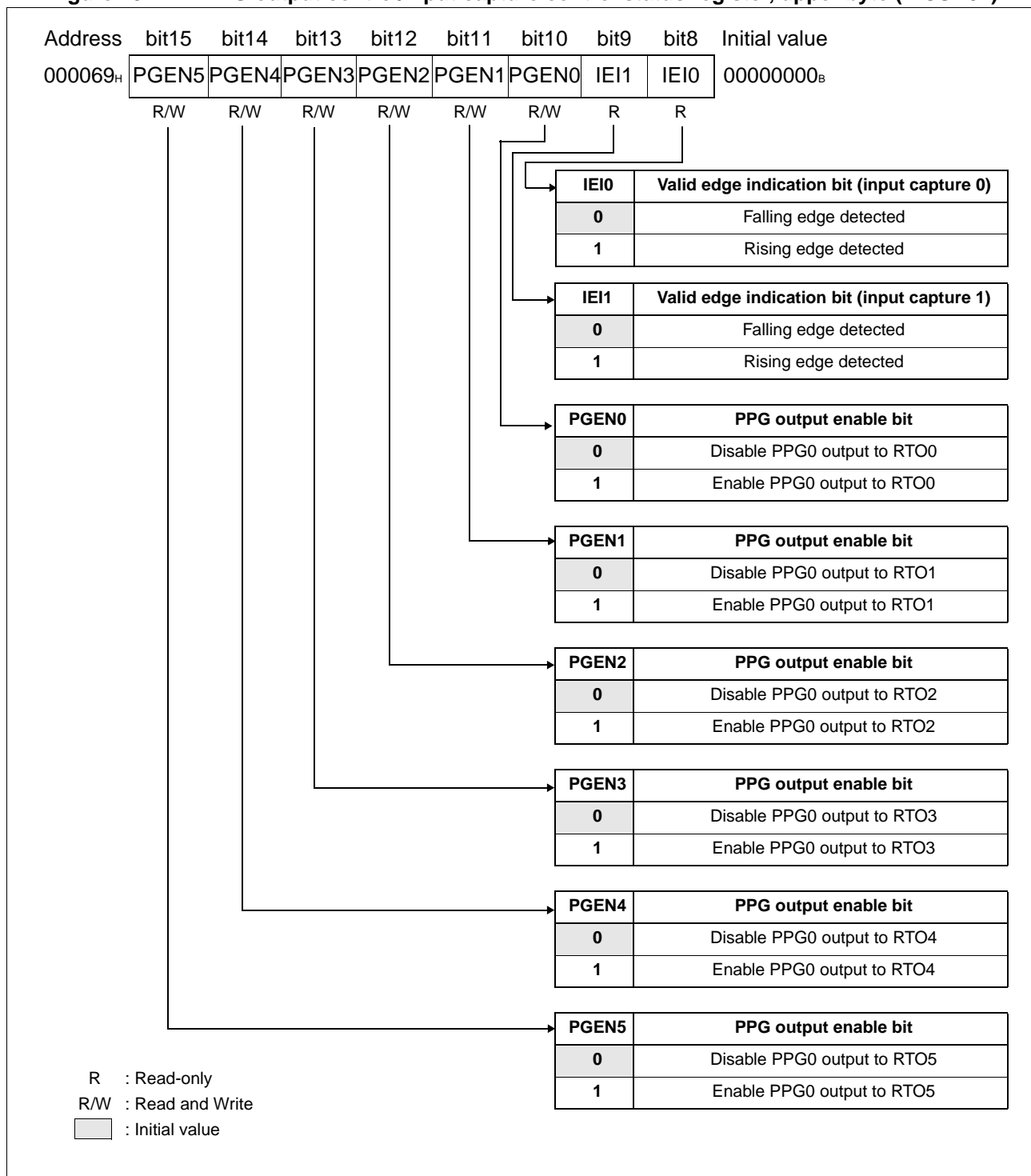


**Table 15.4-6 Input capture control status register, lower byte (ICSL23)**

Bit name		Function
bit7	ICP3: Interrupt request flag bit (Input capture 3)	<ul style="list-style-type: none"> <li>• This bit is used as interrupt request flag for input capture 3.</li> <li>• "1" is set to this bit upon detection of a valid edge in an external input pin.</li> <li>• While the interrupt enable bit (ICE3) is set, an interrupt can be generated upon detection of a valid edge.</li> <li>• Writing "0" will clear this bit.</li> <li>• Writing "1" has no effect.</li> <li>• In read-modify-write operation, "1" is always read.</li> </ul>
bit6	ICP2: Interrupt request flag bit (Input capture 2)	<ul style="list-style-type: none"> <li>• This bit is used as interrupt request flag for input capture 2.</li> <li>• "1" is set to this bit upon detection of a valid edge in an external input pin.</li> <li>• While the interrupt enable bit (ICE2) is set, an interrupt can be generated upon detection of a valid edge.</li> <li>• Writing "0" will clear this bit.</li> <li>• Writing "1" has no effect.</li> <li>• In read-modify-write operation, "1" is always read.</li> </ul>
bit5	ICE3: Interrupt request enable bit (Input capture 3)	<ul style="list-style-type: none"> <li>• This bit is used to enable input capture interrupt request for input capture 3.</li> <li>• While "1" is written to this bit, an input capture interrupt is generated when the interrupt flag (ICP3) is set.</li> </ul>
bit4	ICE2: Interrupt request enable bit (Input capture 2)	<ul style="list-style-type: none"> <li>• This bit is used to enable input capture interrupt request for input capture 2.</li> <li>• While "1" is written to this bit, an input capture interrupt is generated when the interrupt flag (ICP2) is set.</li> </ul>
bit3, bit2	EG31, EG30: Edge selection bits (Input capture 3)	<ul style="list-style-type: none"> <li>• These bits are used to specify the valid edge polarity of an external input for input capture 3.</li> <li>• These bits are also used to enable input capture operation.</li> </ul>
bit1, bit0	EG21, EG20: Edge selection bits (Input capture 2)	<ul style="list-style-type: none"> <li>• These bits are used to specify the valid edge polarity of an external input for input capture 2.</li> <li>• These bits are also used to enable input capture operation.</li> </ul>

■ **PPG Output Control / Input Capture Control Status Register, Upper Byte (PICSH01)**

**Figure 15.4-17 PPG output control/input capture control status register, upper byte (PICSH01)**

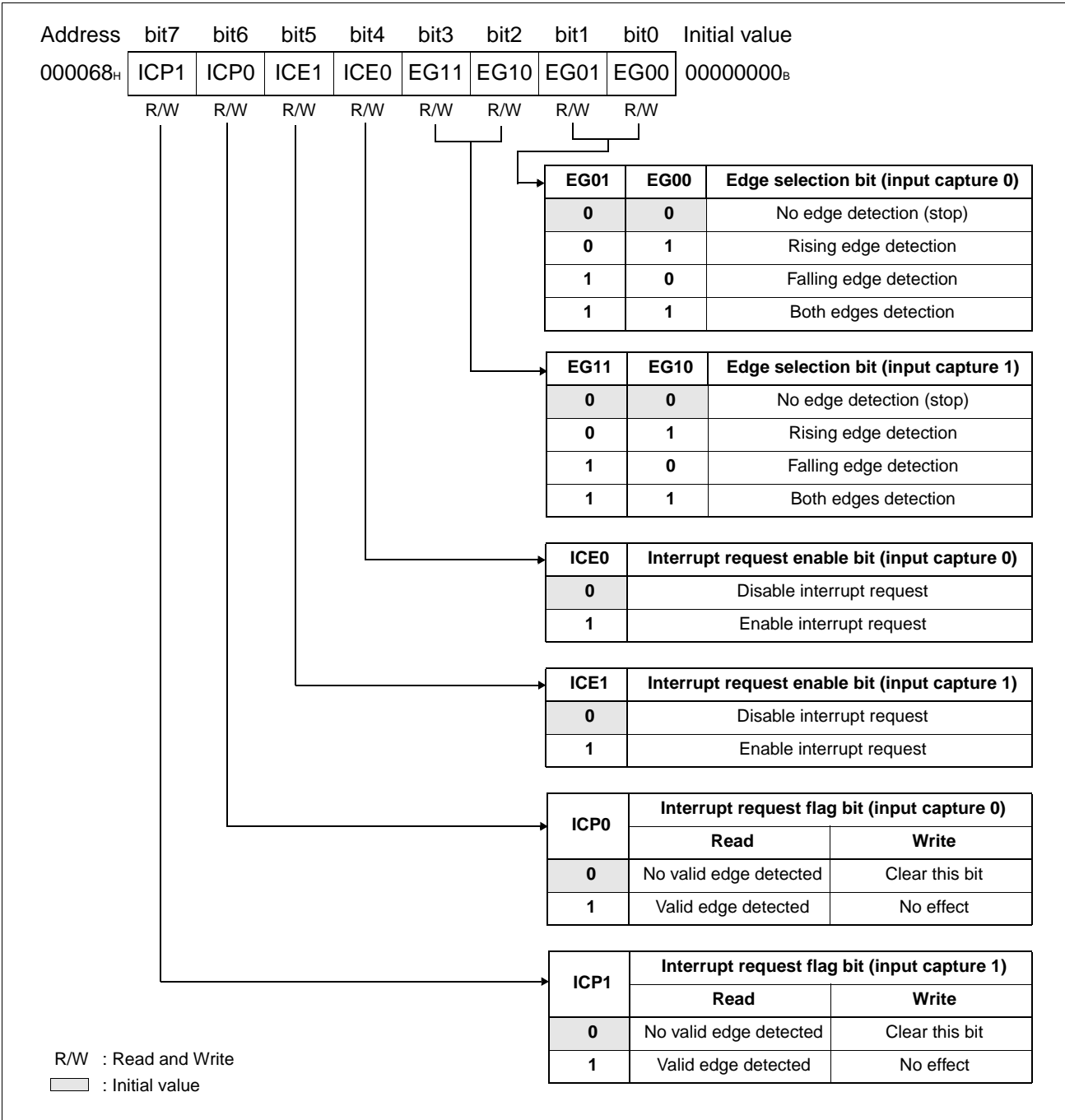


**Table 15.4-7 PPG output control/input capture control status register, upper byte (PICSH01)**

Bit name		Function
bit15 to bit10	PGEN5 to PGEN0: PPG output enable bits	<ul style="list-style-type: none"> <li>These bits are used to select PPG0 output to RTO0/1/2/3/4/5.</li> </ul>
bit9	IEI1: Valid edge indication bit (input capture 1)	<ul style="list-style-type: none"> <li>This bit is an valid edge indication bit for capture register 1, to indicate a rising or falling edge is detected.</li> <li>"0" is written to this bit when falling edge is detected.</li> <li>"1" is written to this bit when rising edge is detected.</li> <li>This bit is read-only.</li> </ul> <p><b>Note:</b> The read value is meaningless when EG11, EG10 = 00<sub>B</sub>.</p>
bit8	IEI0: Valid edge indication bit (input capture 0)	<ul style="list-style-type: none"> <li>This bit is an value edge indication bit for capture register 0, to indicate a rising or falling edge is detected.</li> <li>"0" is written to this bit when falling edge is detected.</li> <li>"1" is written to this bit when rising edge is detected.</li> <li>This bit is read-only.</li> </ul> <p><b>Note:</b> The read value is meaningless when EG01, EG00 = 00<sub>B</sub>.</p>

■ Input Capture Control Status Register, Lower Byte (PICSL01)

Figure 15.4-18 Input capture control status register, lower byte (PICSL01)



**Table 15.4-8 Input capture control status register, lower byte (PICSL01)**

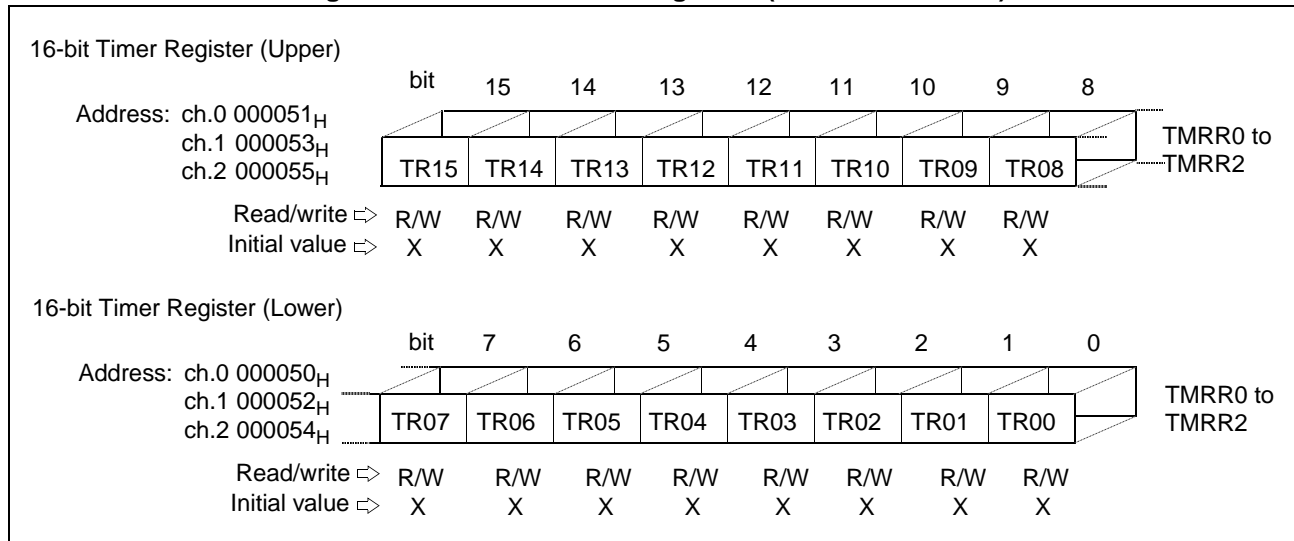
Bit name		Function
bit7	ICP1: Interrupt request flag bit (Input capture 1)	<ul style="list-style-type: none"> <li>• This bit is used as interrupt request flag for input capture 1.</li> <li>• "1" is set to this bit upon detection of a valid edge for an external input pin.</li> <li>• While the interrupt enable bit (ICE1) is set, an interrupt can be generated upon detection of a valid edge.</li> <li>• Writing "0" will clear this bit.</li> <li>• Writing "1" has no effect.</li> <li>• In read-modify-write operation, "1" is always read.</li> </ul>
bit6	ICP0: Interrupt request flag bit (Input capture 0)	<ul style="list-style-type: none"> <li>• This bit is used as interrupt request flag for input capture 0.</li> <li>• "1" is set to this bit upon detection of a valid edge for an external input pin.</li> <li>• While the interrupt enable bit (ICE0) is set, an interrupt can be generated upon detection of a valid edge.</li> <li>• Writing "0" will clear this bit.</li> <li>• Writing "1" has no effect.</li> <li>• In read-modify-write operation, "1" is always read.</li> </ul>
bit5	ICE1: Interrupt request enable bit (Input capture 1)	<ul style="list-style-type: none"> <li>• This bit is used to enable input capture interrupt request for input capture 1.</li> <li>• While "1" is written to this bit, an input capture1 interrupt is generated when the interrupt flag (ICP1) is set.</li> </ul>
bit4	ICE0: Interrupt request enable bit (Input capture 0)	<ul style="list-style-type: none"> <li>• This bit is used to enable input capture interrupt request for input capture 0.</li> <li>• While "1" is written to this bit, an input capture0 interrupt is generated when the interrupt flag (ICP0) is set.</li> </ul>
bit3, bit2	EG11, EG10: Edge selection bits (Input capture 1)	<ul style="list-style-type: none"> <li>• These bits are used to specify the valid edge polarity of an external input for input capture 1.</li> <li>• These bits are also used to enable input capture1 operation.</li> </ul>
bit1, bit0	EG01, EG00: Edge selection bits (Input capture 0)	<ul style="list-style-type: none"> <li>• These bits are used to specify the valid edge polarity of an external input for input capture 0.</li> <li>• These bits are also used to enable input capture0 operation.</li> </ul>

## 15.4.8 16-bit Timer Register (TMRR0 to TMRR2)

16-bit timer registers hold the compare value of 16-bit timers.

### ■ 16-bit Timer Registers (TMRR0 to TMRR2)

Figure 15.4-19 16-bit timer registers (TMRR0 to TMRR2)



These registers are used to store the comparison value of 16-bit timers. The value in these registers will be reloaded when the 16-bit timer is started to operate. Therefore, if the value is re-written into these registers during timer operation, this value will be valid at the next timer initiation/operation.

In dead-time timer mode, these registers are used to set the non-overlap time.

- Non-overlap time = (set value + 1) x selected clock.

#### Notes:

- The value of "0000<sub>H</sub>" cannot be set.
- The maximum offset of non-overlap time is counter value of set value-1.

In timer mode, these registers are used to set the GATE time for PPG timer 0 operation.

- GATE time = (set value + 1) x selected clock.

#### Notes:

- The value of "0000<sub>H</sub>" cannot be set and maximum offset is counter value of set value-1.
- The maximum offset of GATE time is counter value of set value-1.

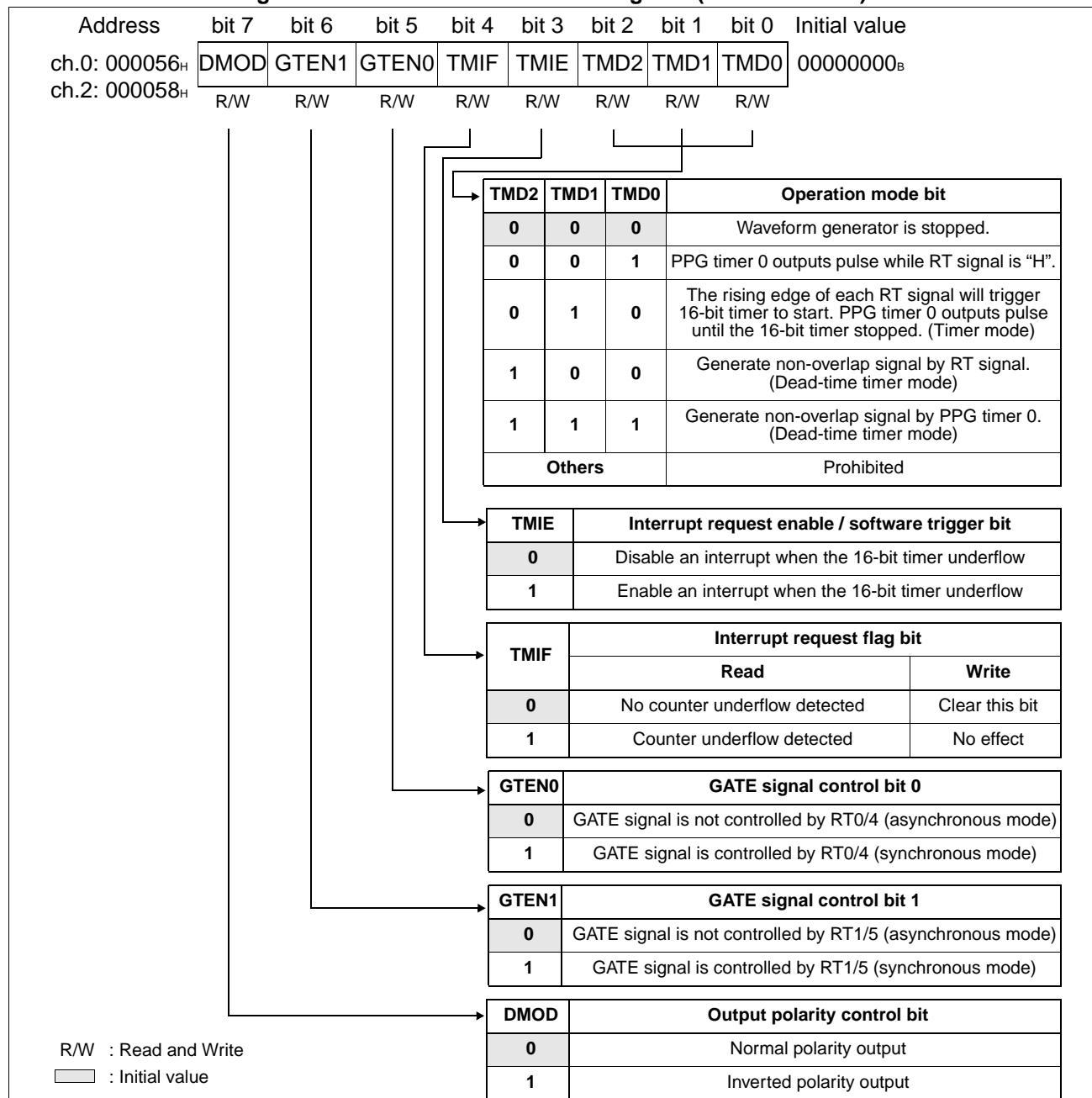
## MB90820B Series

## 15.4.9 16-bit Timer Control Register (DTCR0 to DTCR2)

16-bit timer control registers (DTCR0 to DTCR2) are used to control the operation mode, interrupt request enable, interrupt request flag, GATE signal enable, and output level polarity for the waveform generator.

## ■ 16-bit Timer Control Register (DTCR0/DTCR2)

Figure 15.4-20 16-bit timer control register (DTCR0/DTCR2)





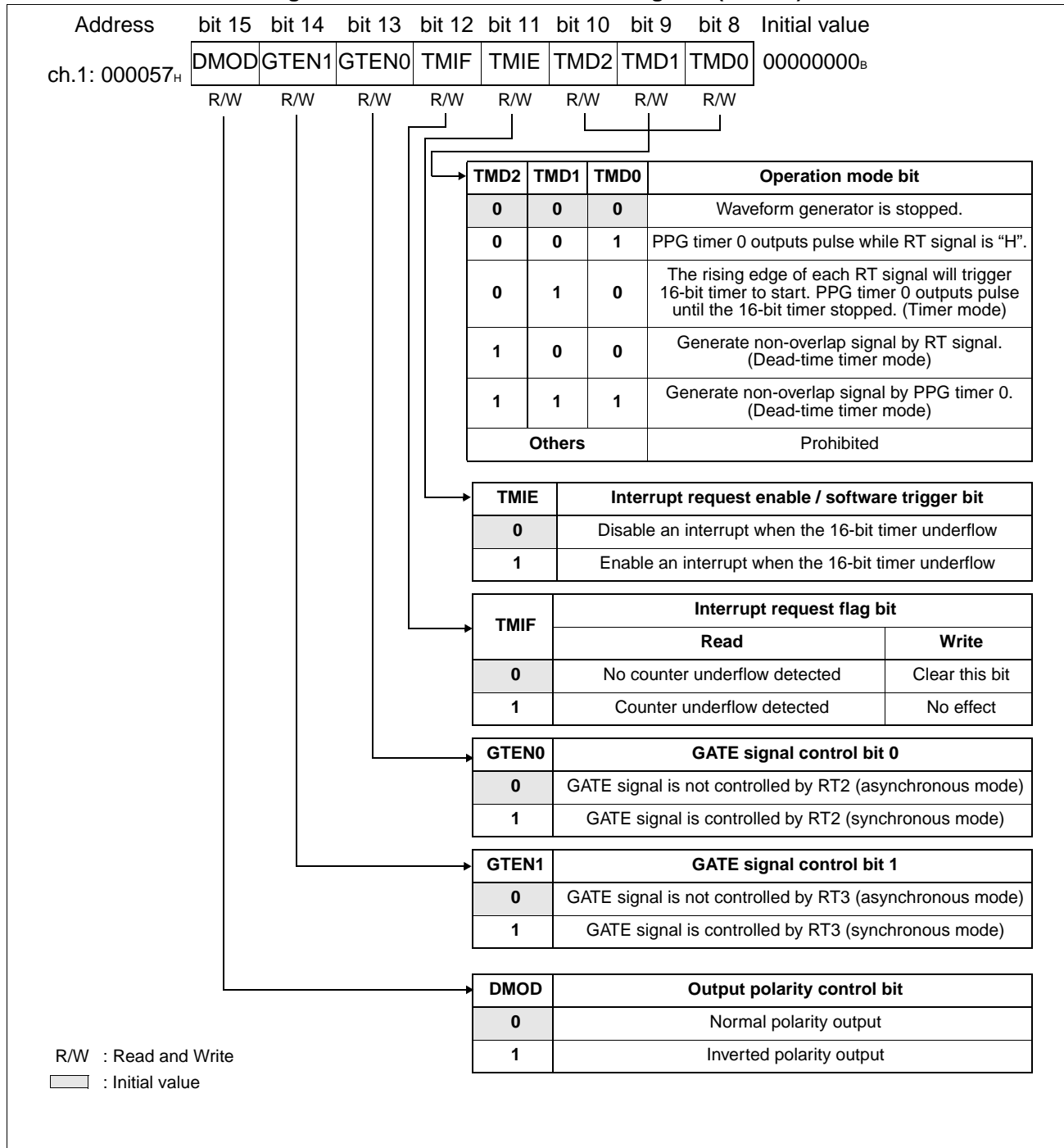
**Table 15.4-9 16-bit timer control registers (DTCR0/DTCR2)**

Bit name		Function
bit7	DMOD: Output polarity control bit	<ul style="list-style-type: none"> <li>This bit is used to set the output polarity of U/V/W in dead-time timer mode.</li> <li>By setting this bit, the output polarity of U/V/W is inverted.</li> </ul> <p><b>Note:</b> This bit is meaningless when dead-time timer mode is not selected (bit 2: TMD2 = 0).</p>
bit6	GTEN1: GATE signal control bit 1	<ul style="list-style-type: none"> <li>This bit is used to control the GATE signal output for PPG timer 0 by RT1/5.</li> </ul>
bit5	GTEN0: GATE signal control bit 0	<ul style="list-style-type: none"> <li>This bit is used to control the GATE signal output of PPG timer 0 by RT0/4.</li> </ul>
bit4	TMIF: Interrupt request flag bit	<ul style="list-style-type: none"> <li>This bit is used as an interrupt request flag for 16-bit timers.</li> <li>This bit will be set to "1" when 16-bit timer 0/2 is underflow.</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>- This bit functions only in mode TMD2 to TMD0=000<sub>B</sub> or 001<sub>B</sub>. In other modes, this bit is always "0".</li> <li>- If both software clear (writing "0") and hardware set (16-bit timer 0/2 underflow) occurs simultaneously, software clear takes the higher priority to clear this bit.</li> </ul>
bit3	TMIE: Interrupt request enable / software trigger bit	<ul style="list-style-type: none"> <li>This bit is used as the software trigger bit and interrupt enable bit for the 16-bit timer 0/2.</li> <li>When TMD2 to TMD0=000<sub>B</sub> or 001<sub>B</sub>, this bit is used as software trigger for 16-bit timer. Setting this bit from "0" to "1" triggers the 16-bit timer to reload and starts down-counting.</li> <li>When this bit is "1" and the interrupt request flag bit (bit 4: TMIF) is "1", an interrupt request is sent to CPU.</li> </ul> <p><b>Note:</b> To retrigger the 16-bit timer, be sure to write "0" before write "1" to this bit.</p>
bit2 to bit0	TMD2 to TMD0: Operation mode bits	<ul style="list-style-type: none"> <li>These bits are used to select the operation mode of the waveform generator.</li> <li>When TMD2 to TMD0=000<sub>B</sub>, output compare RT0/4 and RT1/5 output to RTO0/4 and RTO1/5 respectively. And 16-bit timer can be used as reload timer.</li> <li>When TMD2 to TMD0=001<sub>B</sub>, output compare RT0/4 and RT1/5 output to RTO0/4 and RTO1/5 respectively if PPG0 output is disabled (PICSH01:PGEN0/4=0, PGEN1/5=0). And 16-bit timer can be used as reload timer.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>- To operate the waveform generator in dead-time timer mode, be sure to select 2-channel mode for RT1/5 (OCS1/5:CMOD=1)</li> <li>- When TMD2 to TMD0=111<sub>B</sub> is selected, RTO0/4 and RTO1/5 output are independent of setting in PICSH01:PGEN0/4, PGEN1/5.</li> </ul>

## MB90820B Series

## ■ 16-bit Timer Control Register (DTCR1)

Figure 15.4-21 16-bit timer control register (DTCR1)

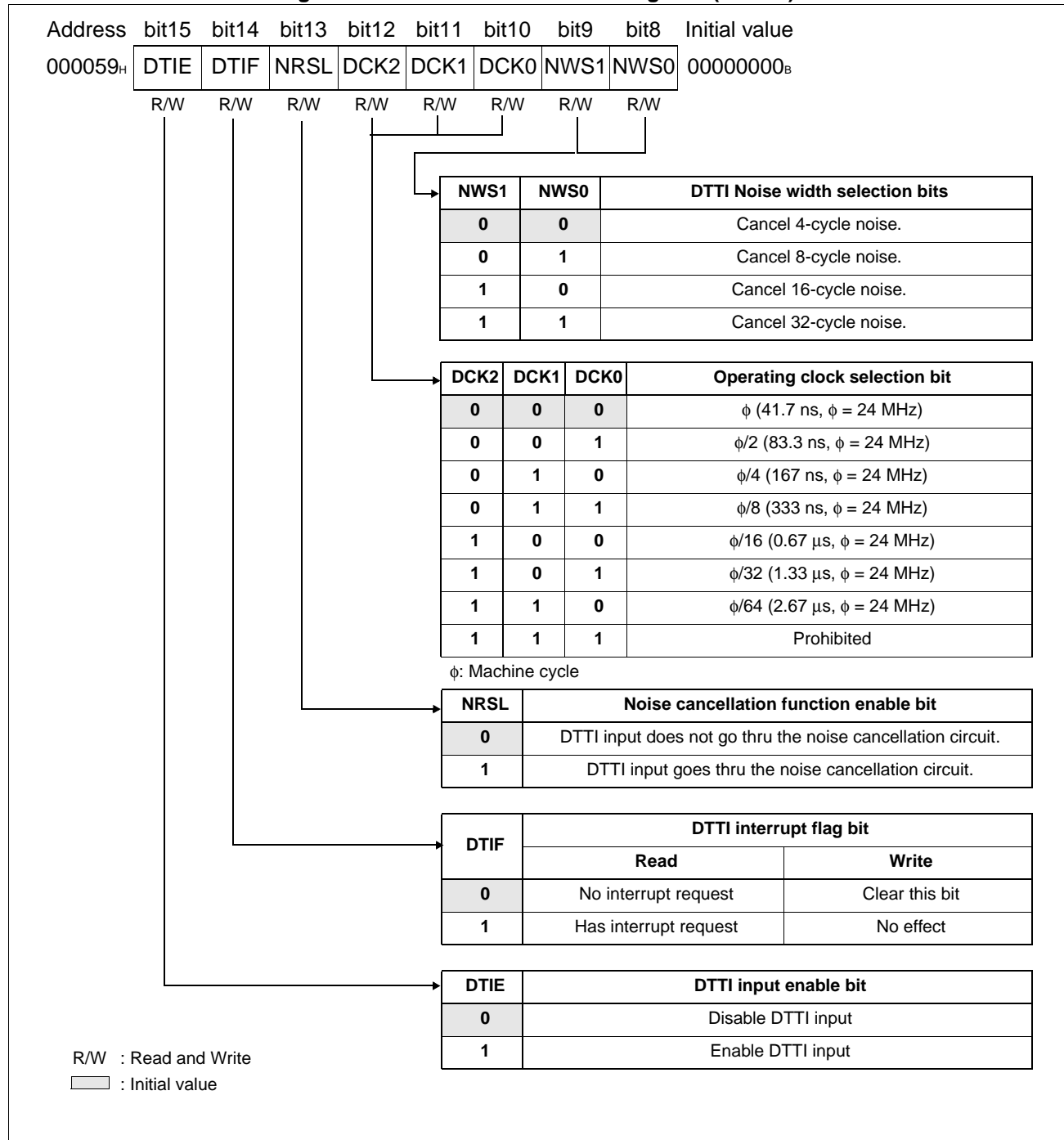


**Table 15.4-10 16-bit timer control registers (DTCR1)**

Bit name		Function
bit15	DMOD: Output polarity control bit	<ul style="list-style-type: none"> <li>This bit is used to set the output polarity of U/V/W in dead-time timer mode.</li> <li>By setting this bit, the output polarity of U/V/W is inverted.</li> </ul> <b>Note:</b> This bit is meaningless when dead-time timer mode is not selected (bit10: TMD2 = 0).
bit14	GTEN1: GATE signal control bit 1	<ul style="list-style-type: none"> <li>This bit is used to control the GATE signal output for PPG timer 0 by RT3.</li> </ul>
bit13	GTEN0: GATE signal control bit 0	<ul style="list-style-type: none"> <li>This bit is used to control the GATE signal output of PPG timer 0 by RT2.</li> </ul>
bit12	TMIF: Interrupt request flag bit	<ul style="list-style-type: none"> <li>This bit is used as an interrupt request flag for 16-bit timers.</li> <li>This bit will be set to "1" when 16-bit timer 1 is underflow.</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <b>Notes:</b> <ul style="list-style-type: none"> <li>- This bit functions only in mode TMD2 to TMD0=000<sub>B</sub> or 001<sub>B</sub>. In other modes, this bit is always "0".</li> <li>- If both software clear (writing "0") and hardware set (16-bit timer 1 underflow) occurs simultaneously, software clear takes the higher priority to clear this bit.</li> </ul>
bit11	TMIE: Interrupt request enable / software trigger bit	<ul style="list-style-type: none"> <li>This bit is used as the software trigger bit and interrupt enable bit for the 16-bit timer.</li> <li>When TMD2 to TMD0=000<sub>B</sub> or 001<sub>B</sub>, this bit is used as software trigger for 16-bit timer. Setting this bit from "0" to "1" triggers the 16-bit timer to reload and starts down-counting.</li> <li>When this bit is "1" and the interrupt request flag bit (bit12: TMIF) is "1", an interrupt request is sent to CPU.</li> </ul> <b>Note:</b> To retrigger the 16-bit timer, be sure to write "0" before write "1" to this bit.
bit10 to bit8	TMD2 to TMD0: Operation mode bits	<ul style="list-style-type: none"> <li>These bits are used to select the operation mode of the waveform generator.</li> <li>When TMD2 to TMD0=000<sub>B</sub>, output compare RT2 and RT3 output to RTO2 and RTO3 respectively. And 16-bit timer can be used as reload timer.</li> <li>When TMD2 to TMD0=001<sub>B</sub>, output compare RT2 and RT3 output to RTO2 and RTO3 respectively if PPG0 output is disabled (PICSH01:PGEN2=0, PGEN3=0). And 16-bit timer can be used as reload timer.</li> </ul> <b>Notes:</b> <ul style="list-style-type: none"> <li>- To operate the waveform generator in dead-time timer mode, be sure to select 2-channel mode for RT3 (OCS3:CMOD=1)</li> <li>- When TMD2 to TMD0=111<sub>B</sub> is selected, RTO2 and RTO3 outputs are independent of setting in PICSH01:PGEN2, 3.</li> </ul>

**MB90820B Series****15.4.10 Waveform Control Register (SIGCR)**

Waveform control register is used to control how the operating clock frequencies, noise cancellation function enable, DTTI input enable, and DTTI interrupt.

**■ Waveform Control Register (SIGCR)****Figure 15.4-22 Waveform control register (SIGCR)**

**Table 15.4-11 Waveform control register (SIGCR)**

Bit name		Function
bit15	DTIE: DTTI input enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable the DTTI pin to control the output level of RTO0 to 5 pins.</li> </ul>
bit14	DTIF: DTTI interrupt flag bit	<ul style="list-style-type: none"> <li>This bit is an interrupt flag for DTTI.</li> <li>When DTTI input is enabled (DTIE=1) and L level of DTTI is detected, this bit will be set, and interrupt request will send to CPU.</li> <li>Writing "0" will clear this bit.</li> <li>Writing "1" has no effect.</li> <li>In read-modify-write operation, "1" is always read.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>If noise cancellation function is enabled (NRSL=1), this bit will be set to "1" when noise pulse width is passed.</li> <li>If both software clear (writing "0") and hardware set (L level of DTTI is detected) occurs simultaneously, software clear takes the higher priority to clear this bit.</li> </ul>
bit13	NRSL: Noise cancellation function enable bit	<ul style="list-style-type: none"> <li>This bit is used to enable the noise cancellation function.</li> <li>Noise cancellation circuit will receive DTTI input signal when the L level is held until the counter overflows. The counter is n-bit counter which is operated by the L level input. The value of n can be 2, 3, 4 and 5 which depends on the setting of NWS1 and NWS0 bits.</li> </ul> <p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>To cancel the noise pulse width, it takes approximately <math>2^n</math> machine cycles.</li> <li>When the noise cancellation circuit is selected, the input will become invalid in a mode such as STOP mode in which the internal clock is stopped.</li> </ul>
bit12 to bit10	DCK2 to DCK0: Operating clock selection bits	<ul style="list-style-type: none"> <li>These bits are used to select the operating clock for the 16-bit timer.</li> </ul>
bit9, bit8	NWS1 and NWS0: DTTI Noise width selection bits	<ul style="list-style-type: none"> <li>These bits are used to select the noise pulse width to be removed for DTTI pin.</li> </ul>

## MB90820B Series

## 15.5 Multi-functional Timer Interrupts

The multi-functional timer is enabled to generate interrupts in 16-bit free-run timer, 16-bit output compare, 16-bit input capture, and waveform generator.

### ■ 16-bit Free-run Timer Interrupts

Table 15.5-1 lists the interrupt control bits and interrupt causes of the 16-bit free-run timer.

**Table 15.5-1 Interrupt control bits and interrupt causes of the 16-bit free-run timer**

	16-bit free-run timer	
	Compare Clear	Zero Detect
Interrupt request flag bit	TCCSH:ICLR	TCCSH:IRQZF
Interrupt request enable bit	TCCSH:ICRE	TCCSH:IRQZE
Interrupt cause	16-bit free-run timer value matches with compare clear register (CPCLR)	16-bit free-run timer value equals zero

In the 16-bit free-run timer, the ICLR bit of the timer control status register (TCCSH) is set to "1" when timer value matches compare clear register (CPCLR). If an interrupt request is enabled (TCCSH:ICRE = 1) in this operation, the interrupt request is output to the interrupt controller.

The IRQZF bit of the timer control status register (TCCSH) is set to "1" when timer value equals 0000<sub>H</sub>. If an interrupt request is enabled (TCCSH:IRQZE = 1) in this operation, the interrupt request is output to the interrupt controller.

### ■ 16-bit Free-run Timer Interrupts and EI<sup>2</sup>OS

Table 15.5-2 lists the 16-bit free-run timer interrupts and EI<sup>2</sup>OS.

**Table 15.5-2 16-bit free-run timer interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
Compare clear <sup>*1</sup>	#34 (22 <sub>H</sub> )	ICR11	0000BB <sub>H</sub>	FFFF74 <sub>H</sub>	FFFF75 <sub>H</sub>	FFFF76 <sub>H</sub>	Δ
Zero detect <sup>*2</sup>	#31 (1F <sub>H</sub> )	ICR10	0000BA <sub>H</sub>	FFFF80 <sub>H</sub>	FFFF81 <sub>H</sub>	FFFF82 <sub>H</sub>	

<sup>\*1</sup>: The same interrupt control register as that for 16-bit free-run timer compare clear is assigned to 16-bit input capture channels 0/1.

<sup>\*2</sup>: The same interrupt control register as that for 16-bit free-run timer zero detect is assigned to 16-bit PPG timer 2.

## ■ 16-bit Output Compare Interrupts

Table 15.5-3 lists the interrupt control bits and interrupt causes of the 16-bit output compare.

**Table 15.5-3 Interrupt control bits and interrupt causes of the 16-bit output compare 0 to 5**

	16-bit output compare 0/1	16-bit output compare 2/3	16-bit output compare 4/5
Interrupt request flag bit	OCS0:IOP0/IOP1	OCS2:IOP0/IOP1	OCS4:IOP0/IOP1
Interrupt request enable bit	OCS0:IOE0/IOE1	OCS2:IOE0/IOE1	OCS4:IOE0/IOE1
Interrupt cause	16-bit free-run timer value matches with output compare register (OCCP0/OCCP1)	16-bit free-run timer value matches with output compare register (OCCP2/OCCP3)	16-bit free-run timer value matches with output compare register (OCCP4/OCCP5)

The IOP0/IOP1 bit of the compare control register, lower byte (OCS0/OCS2/OCS4) is set to "1" when 16-bit free-run timer value matches output compare register (OCCP0 to OCCP5). If an interrupt request is enabled (OCS0/OCS2/OCS4:IOE0/IOE1 = 1) in this operation, the interrupt request is output to the interrupt controller.

## ■ 16-bit Output Compare Interrupts and EI<sup>2</sup>OS

Table 15.5-4 lists the 16-bit output compare interrupts and EI<sup>2</sup>OS

**Table 15.5-4 16-bit output compare interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
Output compare 0 match <sup>*1</sup>	#12 (0C <sub>H</sub> )	ICR00	0000B0 <sub>H</sub>	FFFFCC <sub>H</sub>	FFFFCD <sub>H</sub>	FFFFCE <sub>H</sub>	O
Output compare 1 match <sup>*2</sup>	#15 (0F <sub>H</sub> )	ICR02	0000B2 <sub>H</sub>	FFFFC0 <sub>H</sub>	FFFFC1 <sub>H</sub>	FFFFC2 <sub>H</sub>	
Output compare 2 match <sup>*3</sup>	#17 (11 <sub>H</sub> )	ICR03	0000B3 <sub>H</sub>	FFFFB8 <sub>H</sub>	FFFFB9 <sub>H</sub>	FFFFBA <sub>H</sub>	
Output compare 3 match <sup>*4</sup>	#19 (13 <sub>H</sub> )	ICR04	0000B4 <sub>H</sub>	FFFFB0 <sub>H</sub>	FFFFB1 <sub>H</sub>	FFFFB2 <sub>H</sub>	
Output compare 4 match <sup>*5</sup>	#21 (15 <sub>H</sub> )	ICR05	0000B5 <sub>H</sub>	FFFAA8 <sub>H</sub>	FFFAA9 <sub>H</sub>	FFFAAA <sub>H</sub>	
Output compare 5 match <sup>*6</sup>	#23 (17 <sub>H</sub> )	ICR06	0000B6 <sub>H</sub>	FFFA0 <sub>H</sub>	FFFA1 <sub>H</sub>	FFFA2 <sub>H</sub>	

\*1: The same interrupt control register as that for 16-bit output compare 0 is assigned to A/D conversion termination.

\*2: The same interrupt control register as that for 16-bit output compare 1 is assigned to 16-bit PPG timer 1.

\*3: The same interrupt control register as that for 16-bit output compare 2 is assigned to 16-bit reload timer 1 underflow.

\*4: The same interrupt control register as that for 16-bit output compare 3 is assigned to DTP/external interrupt channels 0/1 detection / DTTL.

\*5: The same interrupt control register as that for 16-bit output compare 4 is assigned to DTP/external interrupt channels 2/3 detection / DTTL.

\*6: The same interrupt control register as that for 16-bit output compare 5 is assigned to PWC timer 1.

## MB90820B Series

### ■ 16-bit Input Capture Interrupts

Table 15.5-5 lists the interrupt control bits and interrupt causes of the 16-bit input capture.

**Table 15.5-5 Interrupt control bits and interrupt causes of the 16-bit input capture 0 to 3**

	16-bit input capture 0/1	16-bit input capture 2/3
Interrupt request flag bit	PICSL01:ICP0/ICP1	ICSL23:ICP2/ICP3
Interrupt request enable bit	PICSL01:ICE0/ICE1	ICSL23:ICE2/ICE3
Interrupt cause	Valid edge is detected in IN0/IN1 pins	Valid edge is detected in IN2/IN3 pins

In the 16-bit input capture, the ICP0/ICP1/ICP2/ICP3 bit of the input capture control status register (PICSL01/ICSL23) is set to "1" when valid edge is detected in IN0/IN1/IN2/IN3 pins. If an interrupt request is enabled (PICSL01/ICSL23:ICE0/ICE1 = 1) in this operation, the interrupt request is output to the interrupt controller.

### ■ 16-bit Input Capture Interrupts and EI<sup>2</sup>OS

Table 15.5-6 lists the 16-bit input capture interrupts and EI<sup>2</sup>OS.

**Table 15.5-6 16-bit input capture interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
Input capture 0/1 <sup>*1</sup>	#33 (21 <sub>H</sub> )	ICR11	0000BB <sub>H</sub>	FFFF78 <sub>H</sub>	FFFF79 <sub>H</sub>	FFFF7A <sub>H</sub>	O
Input capture 2/3 <sup>*2</sup>	#35 (23 <sub>H</sub> )	ICR12	0000BC <sub>H</sub>	FFFF70 <sub>H</sub>	FFFF71 <sub>H</sub>	FFFF72 <sub>H</sub>	

\*1: The same interrupt control register as that for 16-bit input capture 0/1 is assigned to 16-bit free-run timer compare clear.

\*2: The same interrupt control register as that for 16-bit input capture 2/3 is assigned to time-base timer.

### ■ Waveform Generator Interrupts

Table 15.5-7 lists the interrupt control bits and interrupt causes of the waveform generator.

**Table 15.5-7 Interrupt control bits and interrupt causes of the waveform generator**

	Waveform generator	
	16-bit timer 0/1/2	DTTI
Interrupt request flag bit	DTCR0/DTCR1/DTCR2:TMIF	SIGCR:DTIF
Interrupt request enable bit	DTCR0/DTCR1/DTCR2:TMIE	--
Interrupt cause	16-bit timer 0/1/2 underflow	L level is detected in DTTI

In the waveform generator, the TMIF bit of the 16-bit timer control register (DTCR0/DTCR1/DTCR2) is set to "1" when 16-bit timer underflow and DTCR0/DTCR1/DTCR2:TMD2 to TMD0=000<sub>B</sub> or 001<sub>B</sub>. If an interrupt request is enabled (DTCR0/DTCR1/DTCR2:TMIE = 1) in this operation, the interrupt request is output to the interrupt controller.



## ■ Waveform Generator Interrupts and EI<sup>2</sup>OS

Table 15.5-8 lists the waveform generator interrupts and EI<sup>2</sup>OS.

**Table 15.5-8 Waveform generator interrupts and EI<sup>2</sup>OS**

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
16-bit timer 0/1/2 underflow *1	#29 (1D <sub>H</sub> )	ICR09	0000B9 <sub>H</sub>	FFFF88 <sub>H</sub>	FFFF89 <sub>H</sub>	FFFF8A <sub>H</sub>	Δ
DTTI *2	#20 (14 <sub>H</sub> )	ICR04	0000B4 <sub>H</sub>	FFFFAC <sub>H</sub>	FFFFAD <sub>H</sub>	FFFFAE <sub>H</sub>	

\*1: The same interrupt control register as that for 16-bit timer 0/1/2 underflow is assigned to 16-bit reload timer 0 underflow.

\*2: The same interrupt control register as that for DTTI is assigned to DTP/external interrupt channels 0/1 detection and 16-bit output compare 3.

## ■ EI<sup>2</sup>OS Function of the Multi-functional Timer

Since the multi-functional timer has a circuit that coordinates with EI<sup>2</sup>OS, when the interrupt is generated, it can start EI<sup>2</sup>OS.

However, EI<sup>2</sup>OS is available only when other peripheral functions sharing the interrupt control register (ICR) do not use interrupts. For example, when 16-bit free-run timer compare clear uses EI<sup>2</sup>OS, interrupts of 16-bit input capture channels 0/1 must be disabled.

## MB90820B Series

### 15.6 Operation of Multi-functional Timer

---

This section describes the operation of the multi-functional timer.

---

#### ■ Operation of Multi-functional Timer

- 16-bit free-run timer

The 16-bit free-run timer starts counting up from value set in timer data register (TCDT) after a reset has been completed. The counter value is used as the reference time for 16-bit output compare and 16-bit input capture.

- 16-bit output compare

The 16-bit output compare is used to compare the value set in the specified output compare register with the value of the 16-bit free-run timer. If a match is detected, the interrupt flag is set and the output level is inverted.

- 16-bit input capture

The 16-bit input capture is used to detect a specified valid edge. If a valid edge is detected, the interrupt flag is set, and the value of 16-bit free-run timer is fetched and stored into the input capture data register.

- Waveform generator

Waveform generator can produce various waveform such as dead-time, by using the realtime outputs (RTO0 to RTO5), 16-bit PPG timer 0, and 16-bit timers.

## 15.6.1 Operation of 16-bit Free-run Timer

The 16-bit free-run timer starts counting up from counter value specified in timer data register (TCDT) after a reset has been completed. The counter value is used as the reference time for 16-bit output compare and 16-bit input capture.

### ■ Timer Clear

The counter value of 16-bit free-run timer is cleared in the following conditions:

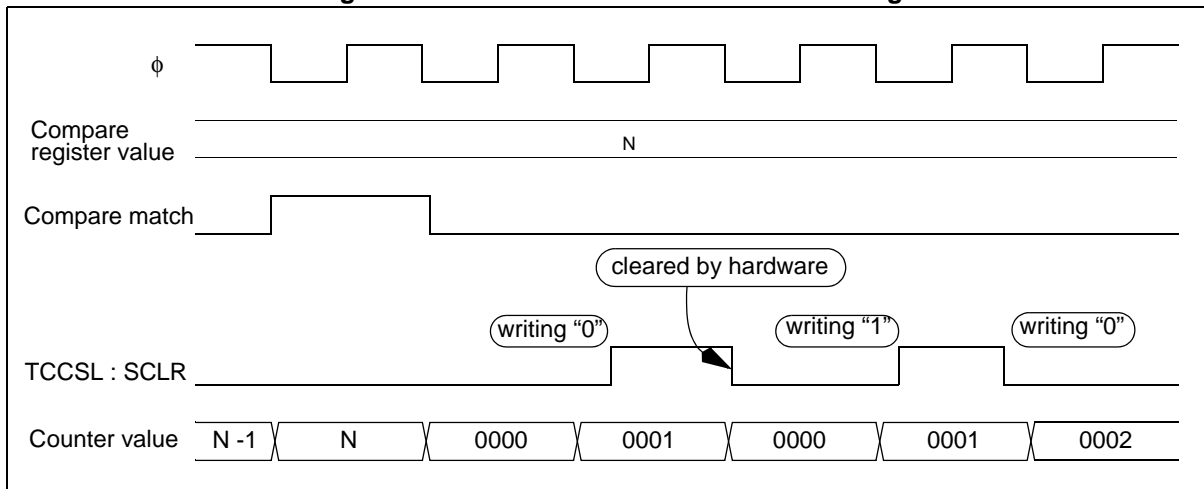
- When a match with compare clear register is detected in up-count mode (TCCSL:MODE=0)
- When "1" is written to the SCLR bit of the TCCSL register during operation, the timer will be cleared at the valid edge of count clock.

**Note :** If writing "0" to the SCLR bit before a valid edge of count clock, the SCLR bit is cleared and the timer would not be cleared to "0000<sub>H</sub>"

- When "0000<sub>H</sub>" is written to the TCDT register during stop.
- Reset

By a reset, the counter is immediately cleared. By a software clear or a match with compare clear register, the counter is cleared in synchronization with the count timing.

Figure 15.6-1 16-bit free-run timer clear timing



### ■ Timer Mode

Two count modes can be selected in 16-bit free-run timer:

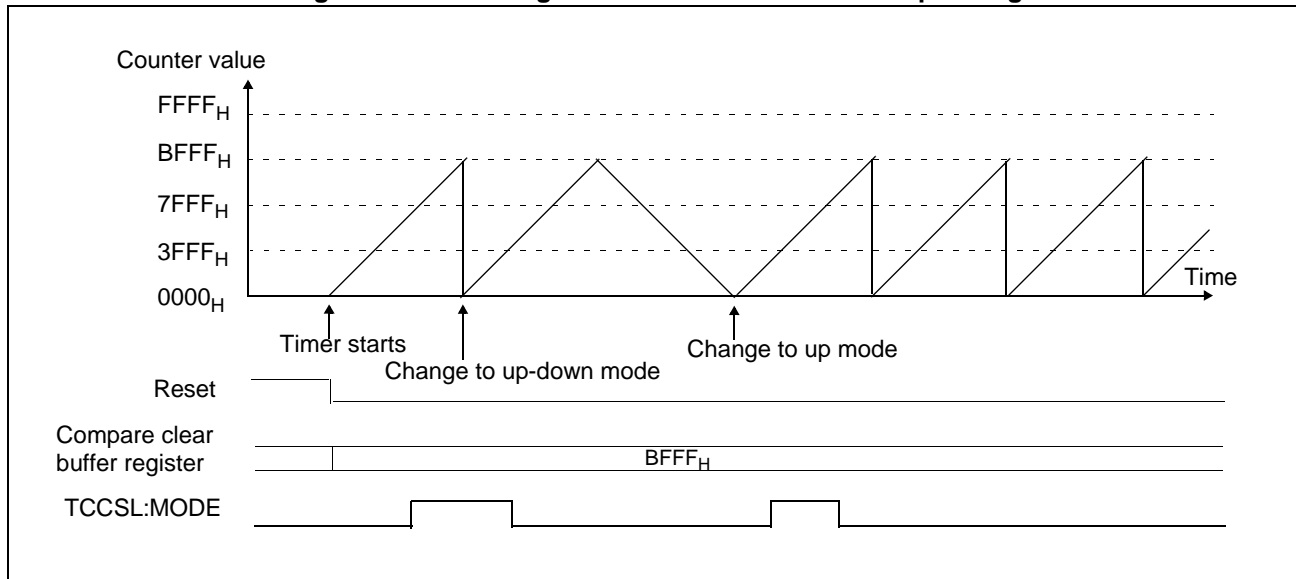
- up-count mode (TCCSL:MODE=0)
- up-down count mode (TCCSL:MODE=1)

In up-count mode, counter starts counting from pre-set timer data register (TCDT), counts up until counter value matches value of compare clear register (CPCLR), then counter is cleared to 0000<sub>H</sub>, and counts up again.

In up-down count mode, counter starts counting from pre-set timer data register (TCDT), counts up until counter value matches value of compare clear register (CPCLR), then counter changes from up-count to down-count, counts down until counter value reaches "0000<sub>H</sub>" and then counts up again.

There is a buffer in mode bit, TCCSL:MODE, it can be written at any time no matter the timer is operating or stopped. While the timer is operating, value written to this bit is buffered and the count mode will be changed when timer value is "0000<sub>H</sub>".

Figure 15.6-2 Change timer mode while timer is operating



### ■ Compare Clear Buffer

There is a selected buffer function on compare clear register (CPCLR). In buffer enable (TCCSL:BFE=1), data written in compare clear buffer register (CPCLRB) will transfer to CPCLR at zero detection of the 16-bit free-run timer. In buffer disable (TCCSL:BFE=0), CPCLRB is transparent, data can directly be written into CPCLR.

Figure 15.6-3 Operation in up-count mode with compare clear buffer is disabled (TCCSL:BFE=0)

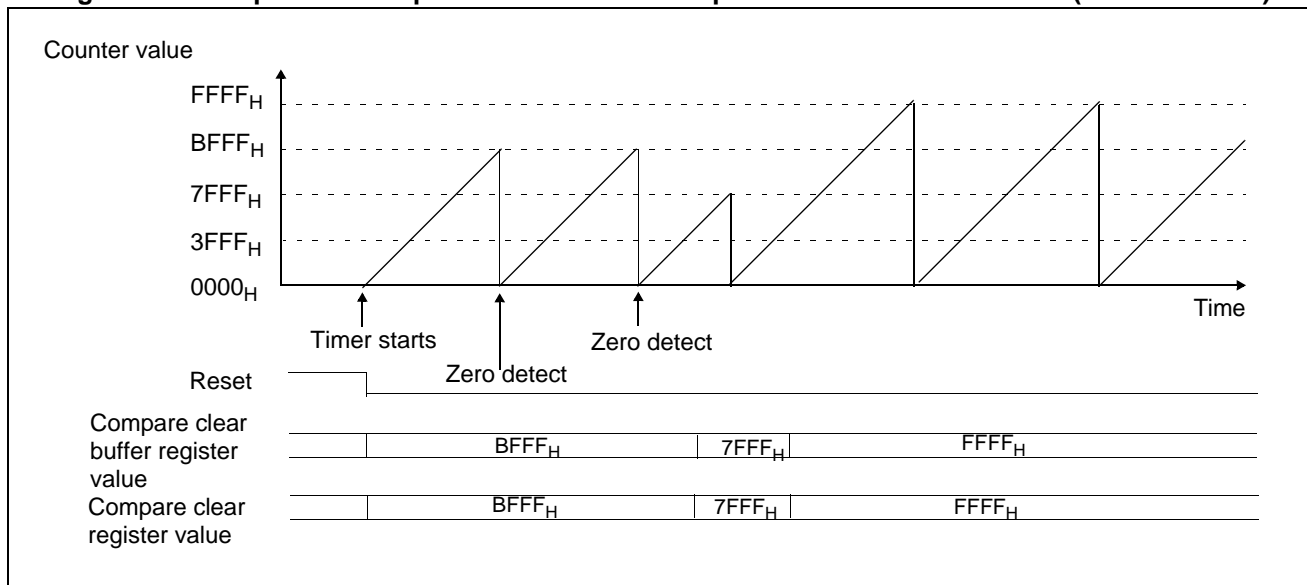


Figure 15.6-4 Operation in up-count mode with compare clear buffer is enabled (TCCSL:BFE=1)

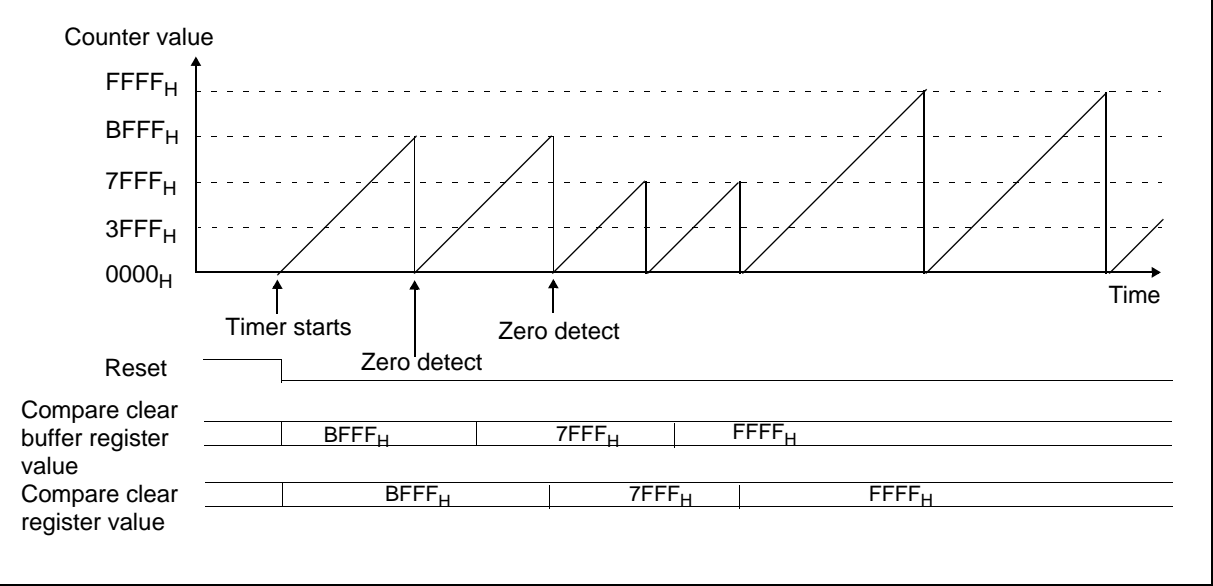
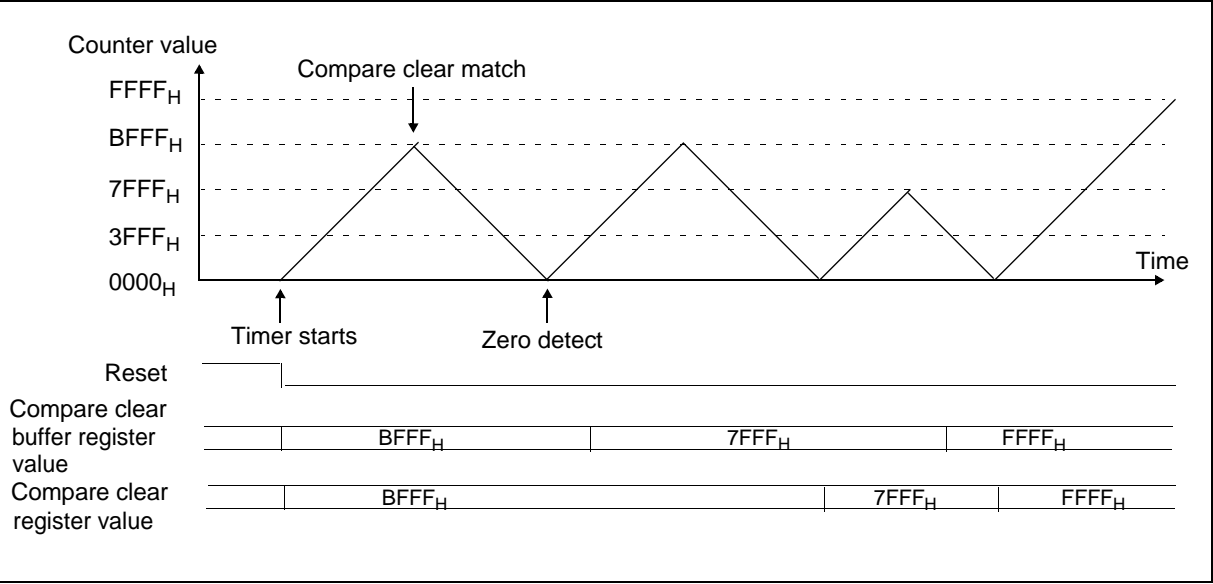


Figure 15.6-5 Operation in up-down count mode with compare clear buffer enabled (TCCSL:BFE=1)



## ■ Timer Interrupts

Two interrupts can be generated from 16-bit free-run timer:

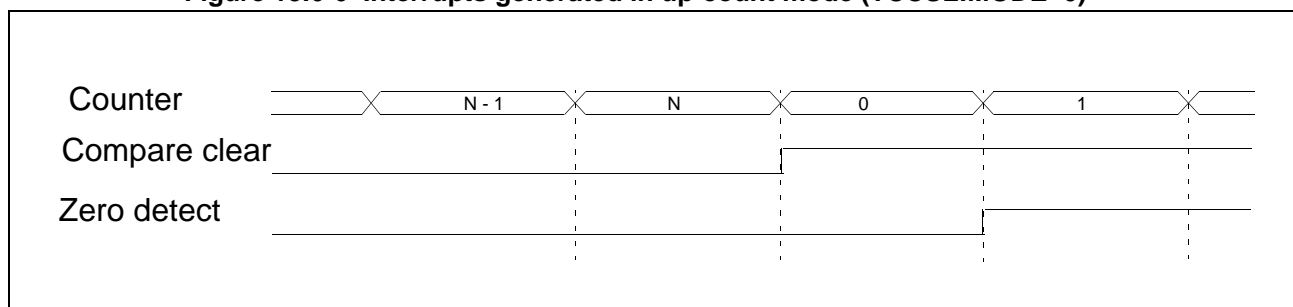
- Compare clear interrupt
- Zero detect interrupt

Compare clear interrupt is generated when the timer value matches compare clear register (CPCLR). Zero detect interrupt is generated when the timer value reaches "0000<sub>H</sub>".

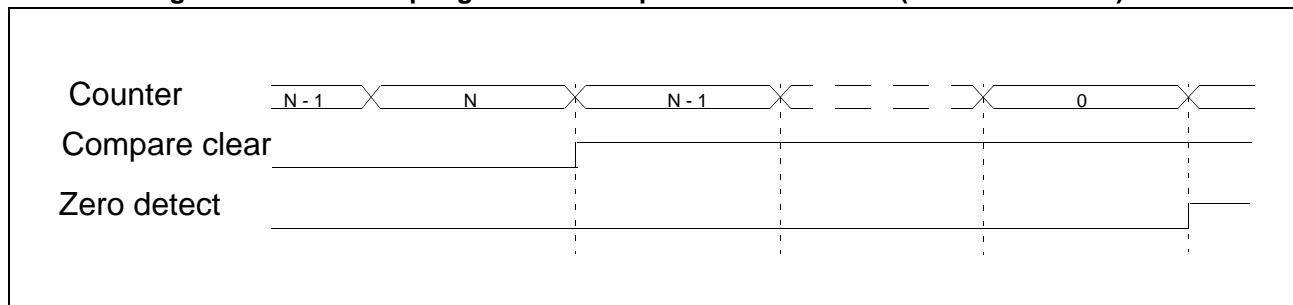
Note:

Software clear (TCCSL:SCLR=1) will not generate zero detect interrupt.

**Figure 15.6-6 Interrupts generated in up-count mode (TCCSL:MODE=0)**



**Figure 15.6-7 Interrupts generated in up-down count mode (TCCSL:MODE=1)**



## ■ Interrupt Mask Function

The interrupt request can be masked by setting TCCSH:MSI2 to MSI0. MSI2 to MSI0 configure a 3-bit reload down counter, which reloads when its count value reaches "000<sub>B</sub>". Count value can also be loaded by writing directly to MSI2 to MSI0. The mask count equals the value set in MSI2 to MSI0 and there is no interrupt source will be masked when MSI2 to MSI0 equals "000<sub>B</sub>".

The interrupt source depends on the count mode (TCCSL:MODE). In up-count mode, only compare clear interrupt can be masked, zero detect interrupt is generated in every zero detection. In up-down count mode, only zero detect interrupt can be masked, compare clear interrupt is generated in every compare clear.

Note:

Software clear (TCCSL:SCLR=1) will not generate zero detection.

**Figure 15.6-8 Compare clear interrupt masked in up-count mode**

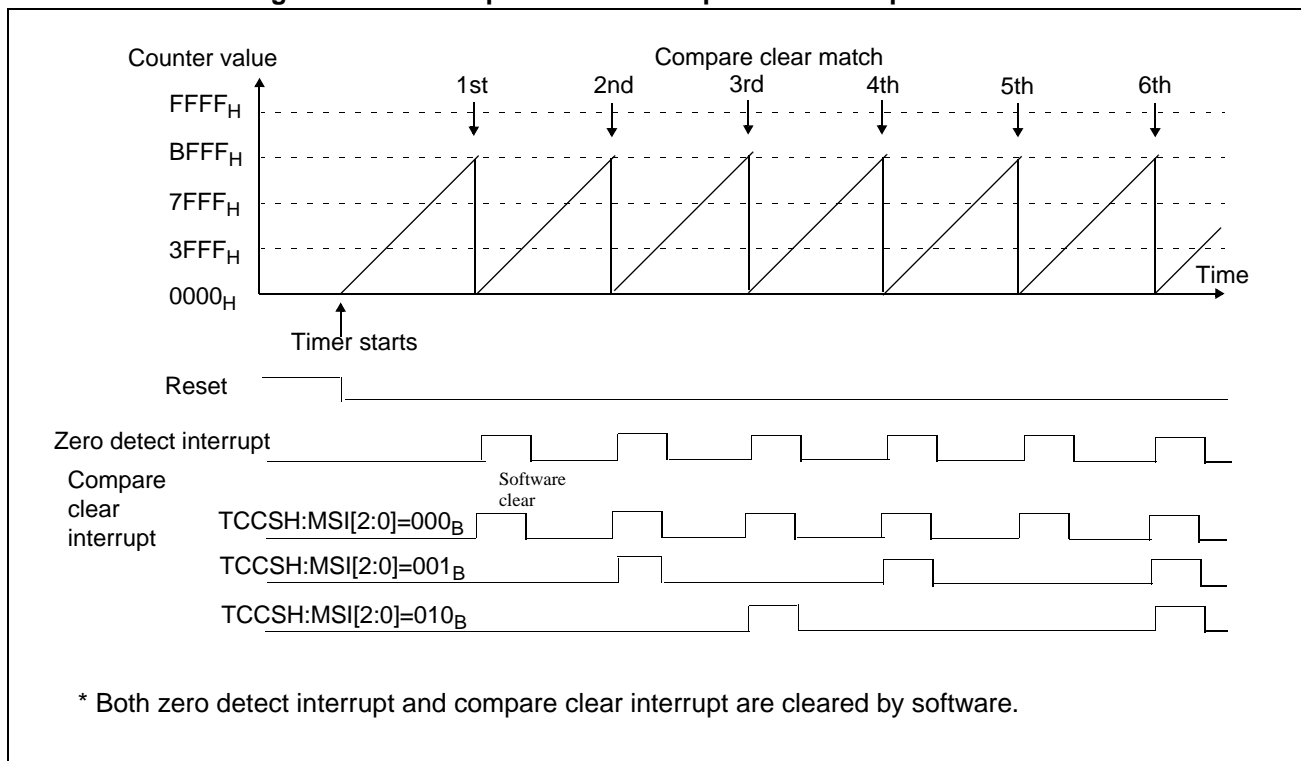
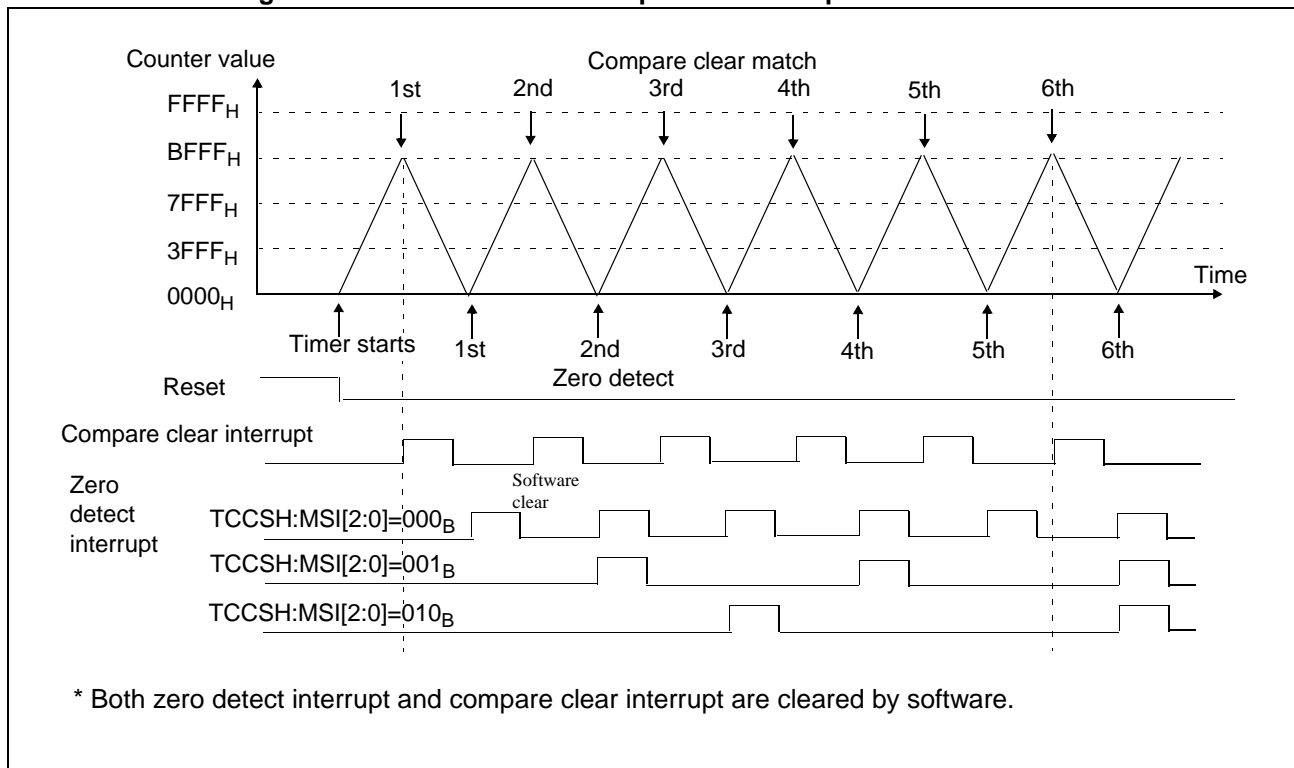


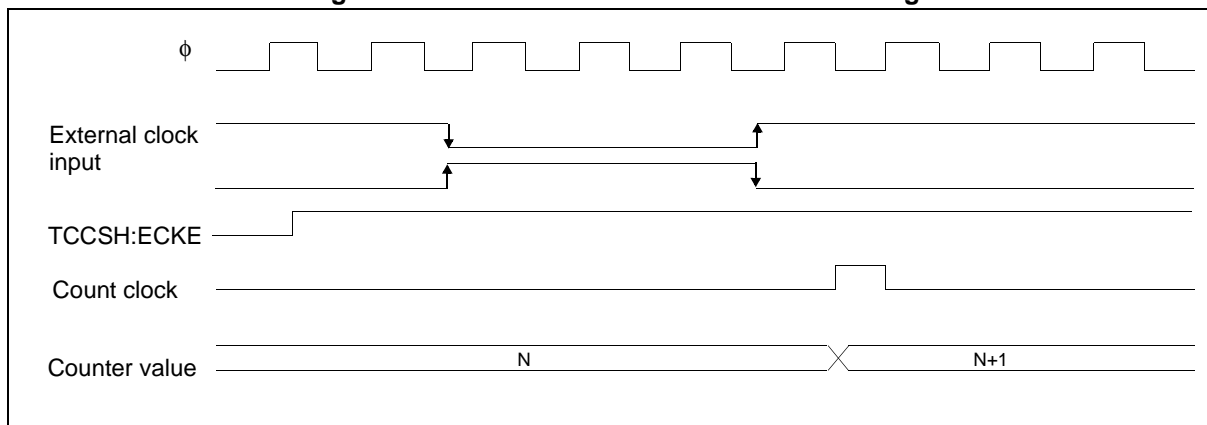
Figure 15.6-9 Zero detect interrupt masked in up-down count mode



## External Count Clock Selected

The 16-bit free-run timer is incremented based on the input clock (internal or external clock). When external clock is selected, the 16-bit free-run timer counts up at a rising edge when the initial value of external input is “1” or at a falling edge when initial value of external clock input is “0” after external clock mode is selected (TCCSH:ECKE=1).

Figure 15.6-10 16-bit free-run timer count timing





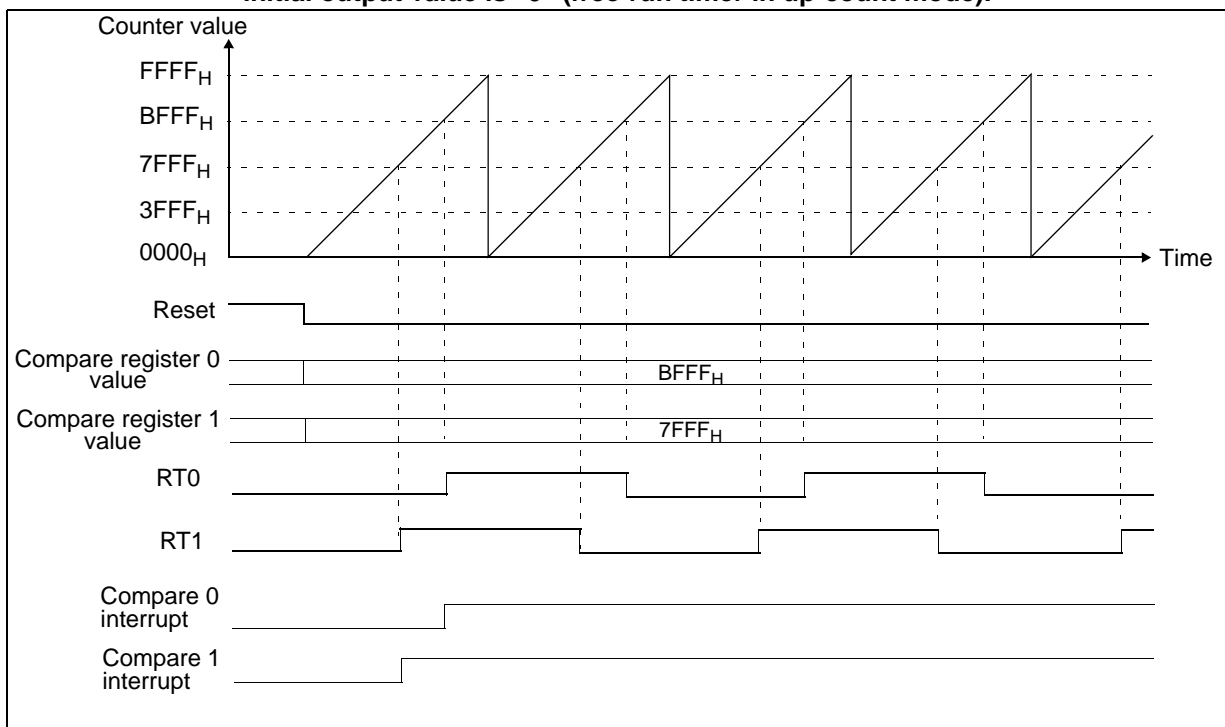
## 15.6.2 Operation of 16-bit Output Compare

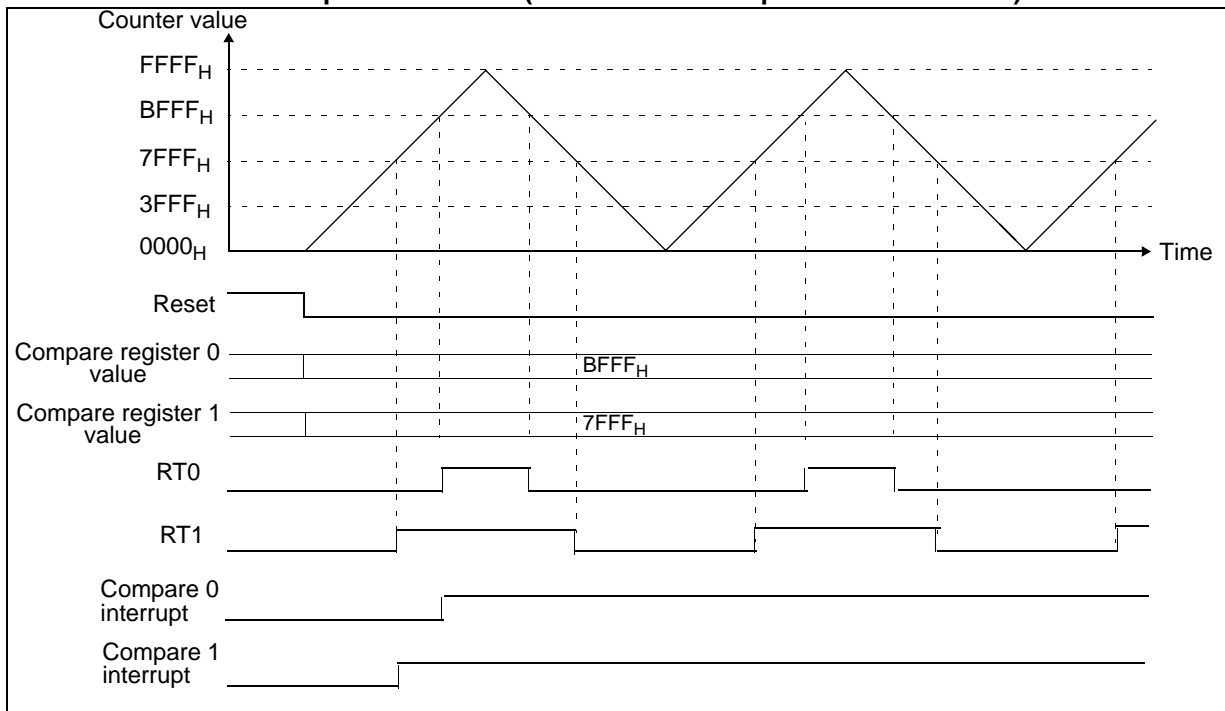
The output compare unit is used to compare the value set in the specified compare register with the value of the 16-bit free-run timer. If a match is detected, the interrupt flag is set and the output level is inverted.

### ■ 16-bit Output Compare Operation

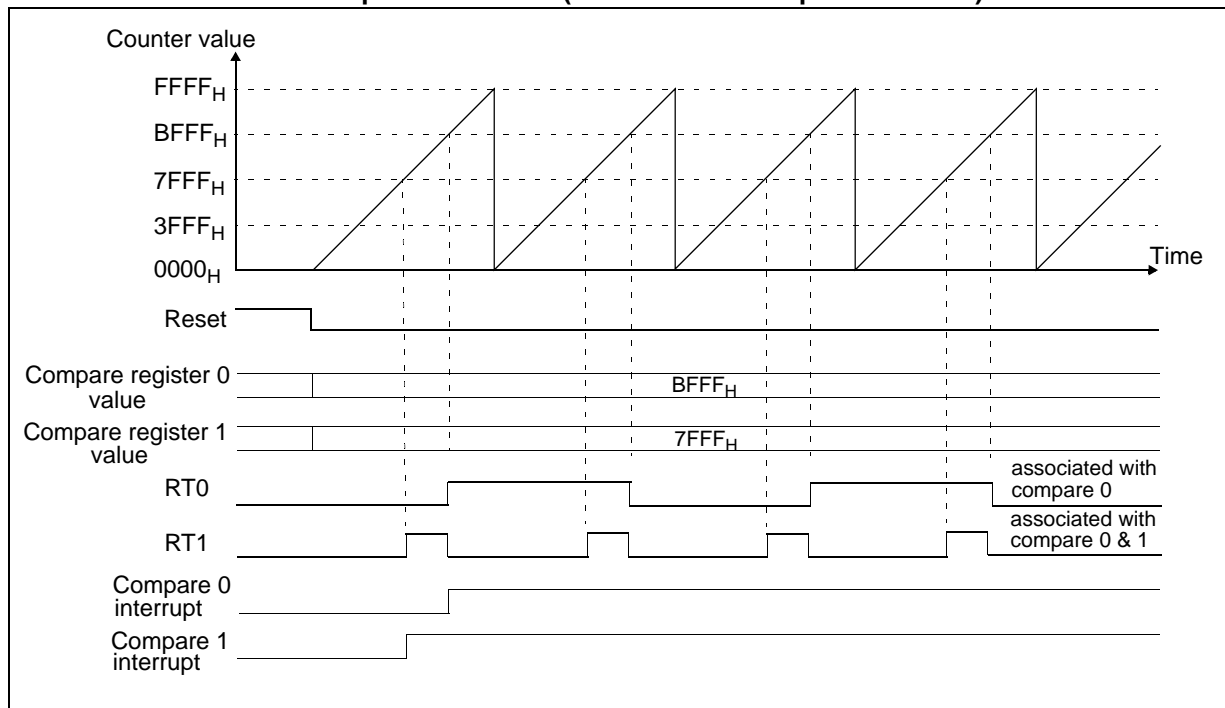
a) Compare operation can be performed for individual channel (OCS1/3/5:CMOD = 0)

**Figure 15.6-11 Sample output waveform when compare registers 0 and 1 are used individually when the initial output value is "0" (free-run timer in up-count mode).**

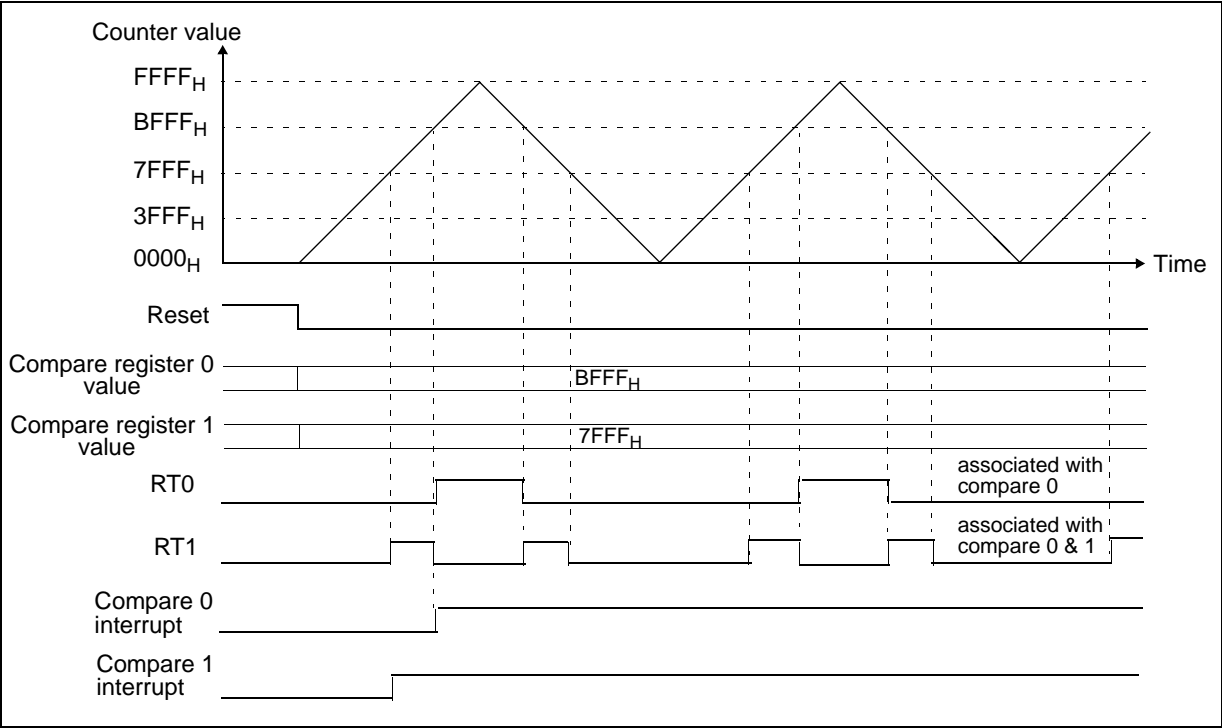


**MB90820B Series****Figure 15.6-12 Sample output waveform when compare registers 0 and 1 are used individually when the initial output value is "0" (free-run timer in up-down count mode).**

b) Output level can be changed by using a pair of compare registers (OCS1/3/5:CMOD = 1)

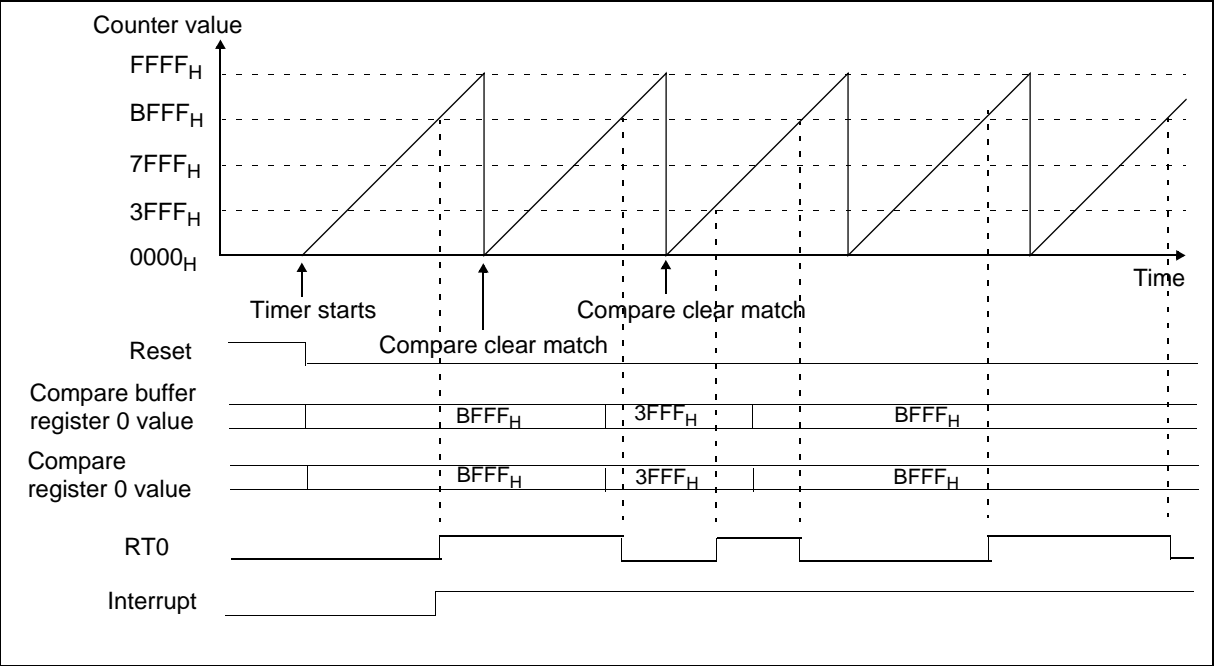
**Figure 15.6-13 Sample output waveform when compare registers 0 and 1 are used in a pair when the initial output value is "0" (free-run timer in up-count mode).**

**Figure 15.6-14 Sample output waveform when compare register 0 and 1 are used in a pair when the initial output value is "0" (free-run timer in up-down count mode).**



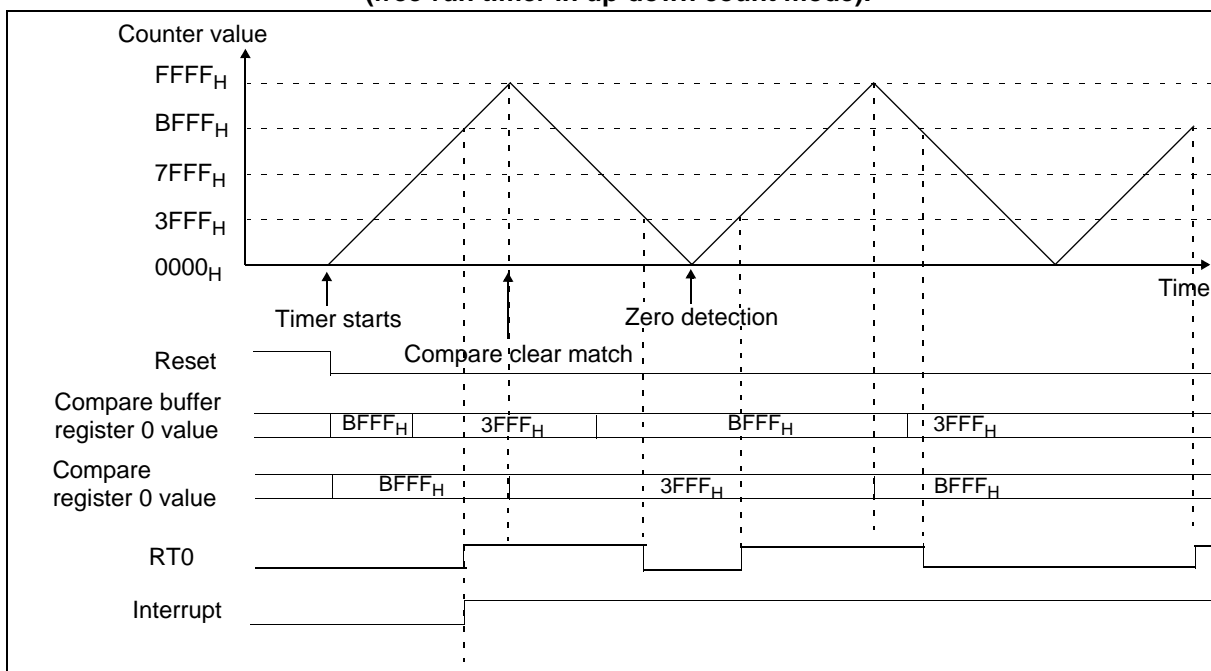
c) Output level when compare buffer is disabled

**Figure 15.6-15 Sample output waveform when compare buffer is disabled (free-run timer in up-count mode).**



d) Output level when compare buffer is selected at compare clear match

**Figure 15.6-16 Sample output waveform when compare buffer is enable (free-run timer in up-down count mode).**



### ■ 16-bit Output Compare Timing

When the free-run timer matches the value set in the compare register, the output compare unit generates a compare match signal to invert the output and generates an interrupt. When a compare match occurs, the output is inverted in synchronization with the count timing of the counter.

Note:

When the compare register is updated, comparison with the counter value is not performed.

**Figure 15.6-17 Compare operation upon update of compare registers**

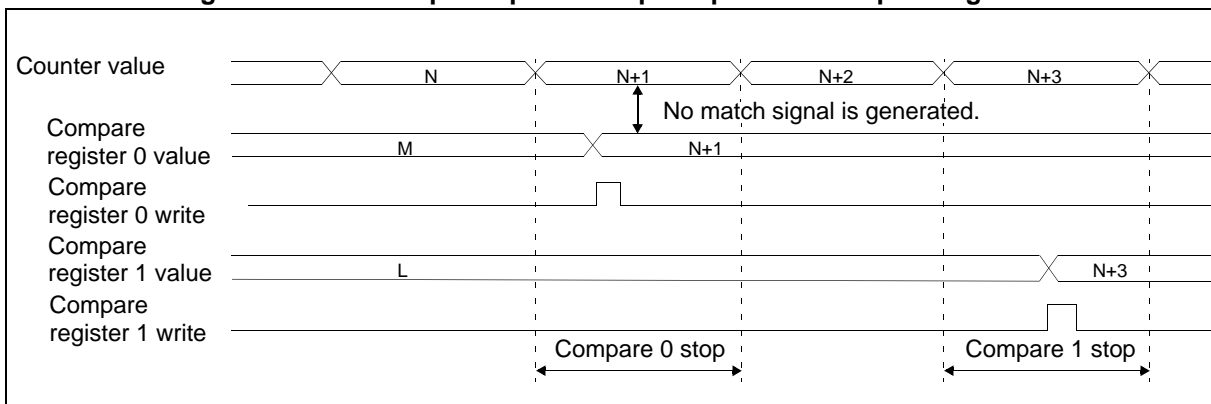


Figure 15.6-18 Compare interrupt timing

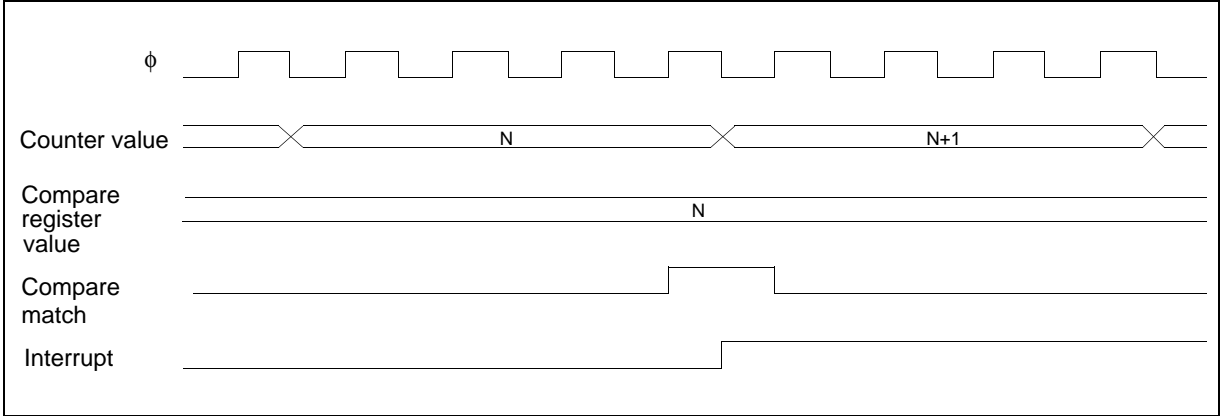
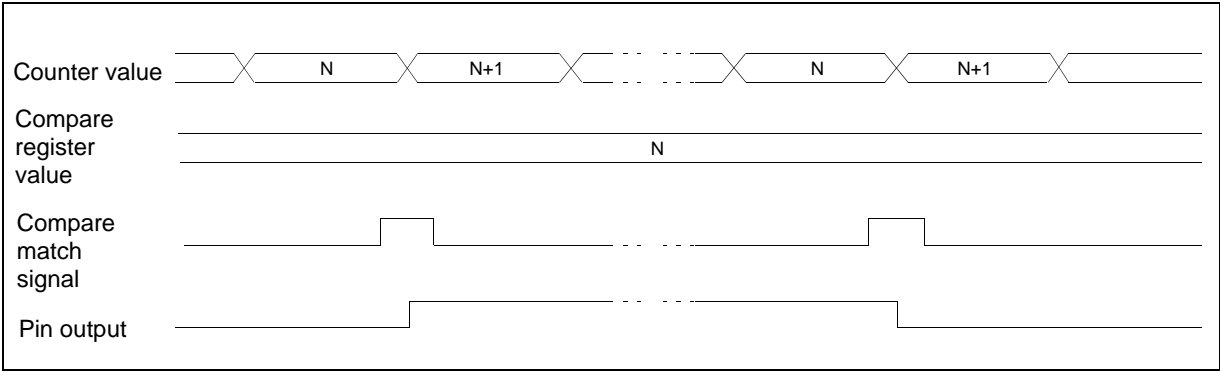
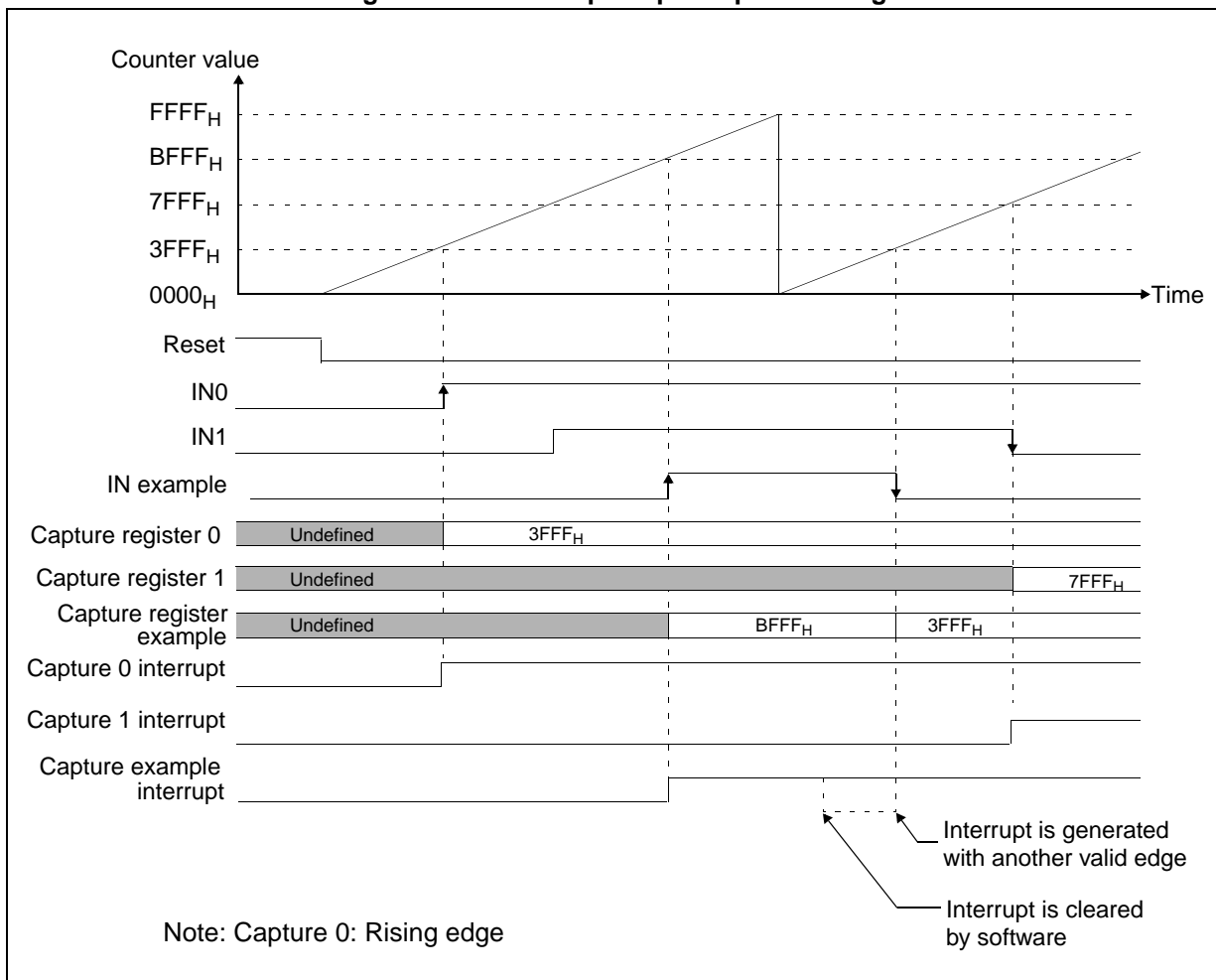


Figure 15.6-19 Output pin change timing



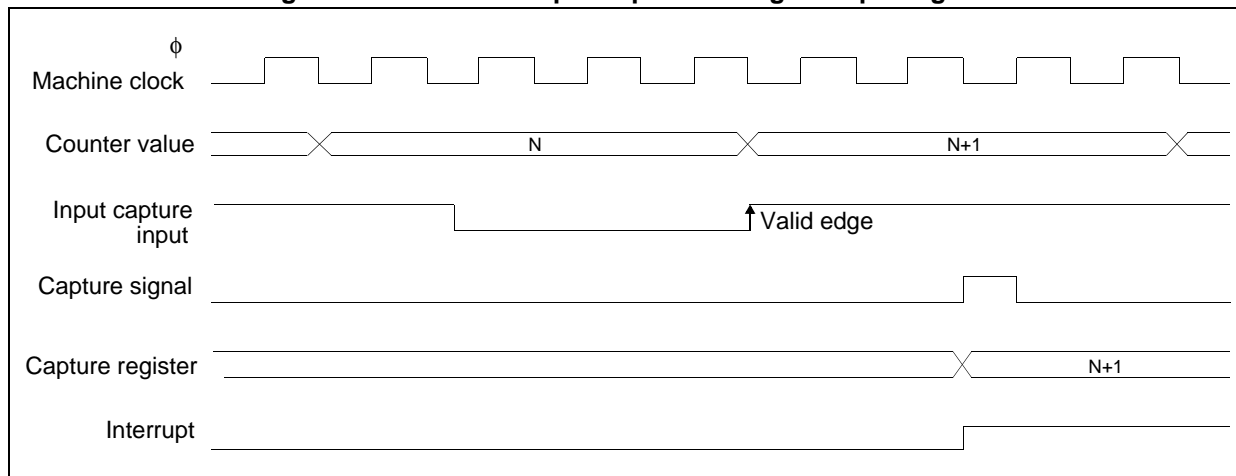
**MB90820B Series****15.6.3 Operation of 16-bit Input Capture**

The input capture unit is used to detect a specified valid edge. If a valid edge is detected, the interrupt flag is set and the value of 16-bit free-run timer is loaded into the capture register.

**■ 16-bit Input Capture Operation****Figure 15.6-20 Sample input capture timing**

## ■ 16-bit Input Capture Input Timing

**Figure 15.6-21 16-bit input capture timing for input signals**



**MB90820B Series****15.6.4 Operation of Waveform Generator**

Waveform generator can produce various waveform such as dead-time, by using the realtime outputs (RT0 to RT5), 16-bit PPG timer 0, and 16-bit timers 0/1/2.

**■ Output Condition of RTO0 to RTO5 and GATE****Table 15.6-1 Output condition of RTO0 to RTO5, GATE and register bit setting**

TMD2	TMD1	TMD0	GTENx	PGENx	RTOx *2	GATE
0	0	0	X	X	Realtime output, RTx	Always "0"
0	0	1	X	0	Realtime output, RTx	OR(RTx & GTENx)
0	0	1	0	1	PPG0 output pulse when RTx is high	Always "0"
0	0	1	1	1	Gate triggered PPG0 output pulse when RTx is high	OR(RTx)
0	1	0	X	0	Output "H" from rising edge of RTx to 16-bit timer 0 underflow (x=0, 1)	OR(RTOx & GTENx)
					Output "H" from rising edge of RTx to 16-bit timer 1 underflow (x=2, 3)	
					Output "H" from rising edge of RTx to 16-bit timer 2 underflow (x=4, 5)	
0	1	0	0	1	PPG0 output pulse from rising edge of RTx to 16-bit timer 0 underflow (x=0, 1)	Always "0"
					PPG0 output pulse from rising edge of RTx to 16-bit timer 1 underflow (x=2, 3)	
					PPG0 output pulse from rising edge of RTx to 16-bit timer 2 underflow (x=4, 5)	
0	1	0	1	1	Gate triggered PPG0 output pulse from rising edge of RTx to 16-bit timer 0 underflow (x=0, 1)	OR(output "H" from RTx/y/z rising edge to timer 0/1/2 underflow) x=0, 1 y=2, 3 z=4, 5
					Gate triggered PPG0 output pulse from rising edge of RTx to 16-bit timer 1 underflow (x=2, 3)	
					Gate triggered PPG0 output pulse from rising edge of RTx to 16-bit timer 2 underflow (x=4, 5)	
1	0	0	X	X	Generate non-overlap signal by RT1 (x=0, 1) *1	Always "0"
					Generate non-overlap signal by RT3 (x=2, 3) *1	
					Generate non-overlap signal by RT5 (x=4, 5) *1	
1	1	1	0	X	Generate non-overlap signal by PPG0	Always "0"
1	1	1	1	X	Generate non-overlap signal by gate triggered PPG0	OR(RTx)
Others					Always "0"	Always "0"

\*1 In order to generate non-overlap signal, be sure to select 2-channel mode for RT1/3/5 (OCS1/3/5:CMOD=1)

\*2 RTO0/1 is controlled by DTCR0:TMD2 to TMD0, RTO2/3 is controlled by DTCR1:TMD2 to TMD0 and RTO4/5 is controlled by DTCR2:TMD2 to TMD0.



### ■ PPG0 Output Control

PPG0 output to RTO0 to RTO5 can be enabled by PGEN0 to PGEN5 in PPG output control/input capture control status register (PICS01).

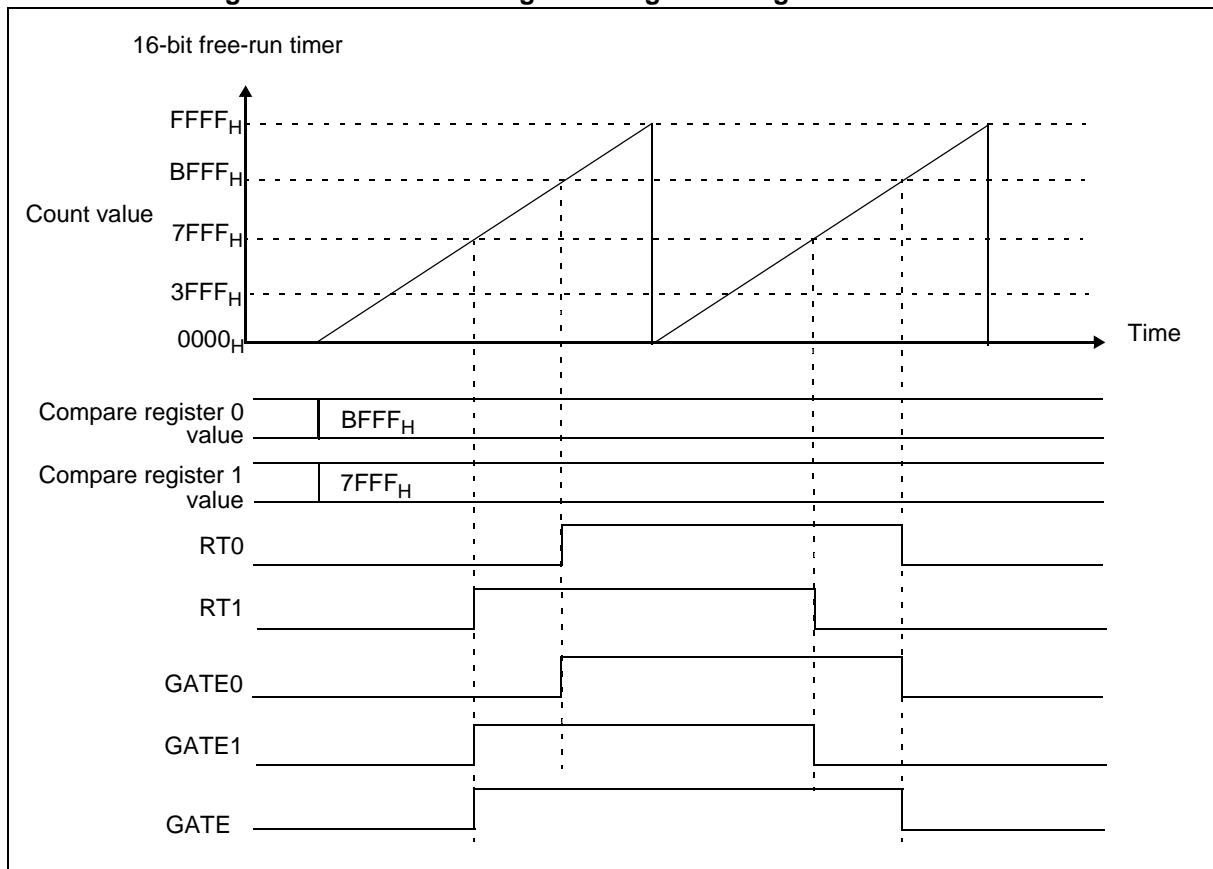
### ■ Gate Triggered PPG0 Output

In waveform generator, a GATE signal can be generated by using realtime outputs RT0 to RT5 or cope with 16-bit timers 0/1/2 to trigger PPG0 counting. When 16-bit timer is used, two real-time outputs (RT0/2/4 and RT1/3/5) is operated with one 16-bit timer 0/1/2 to generate six individual gate signal. And these six gate signals are logically OR to generate a GATE signal to trigger PPG0 counting.

If PGEN0 to PGEN5 signals are also used, six different waveforms can be output to RTO0 to RTO5 by using one PPG0 only.

### ■ Generating GATE Signal During Each RTx is at "H" Level When GTENx is Active (DTCR0/1/2:TMD2 to TMD0=001<sub>B</sub> or 111<sub>B</sub>)

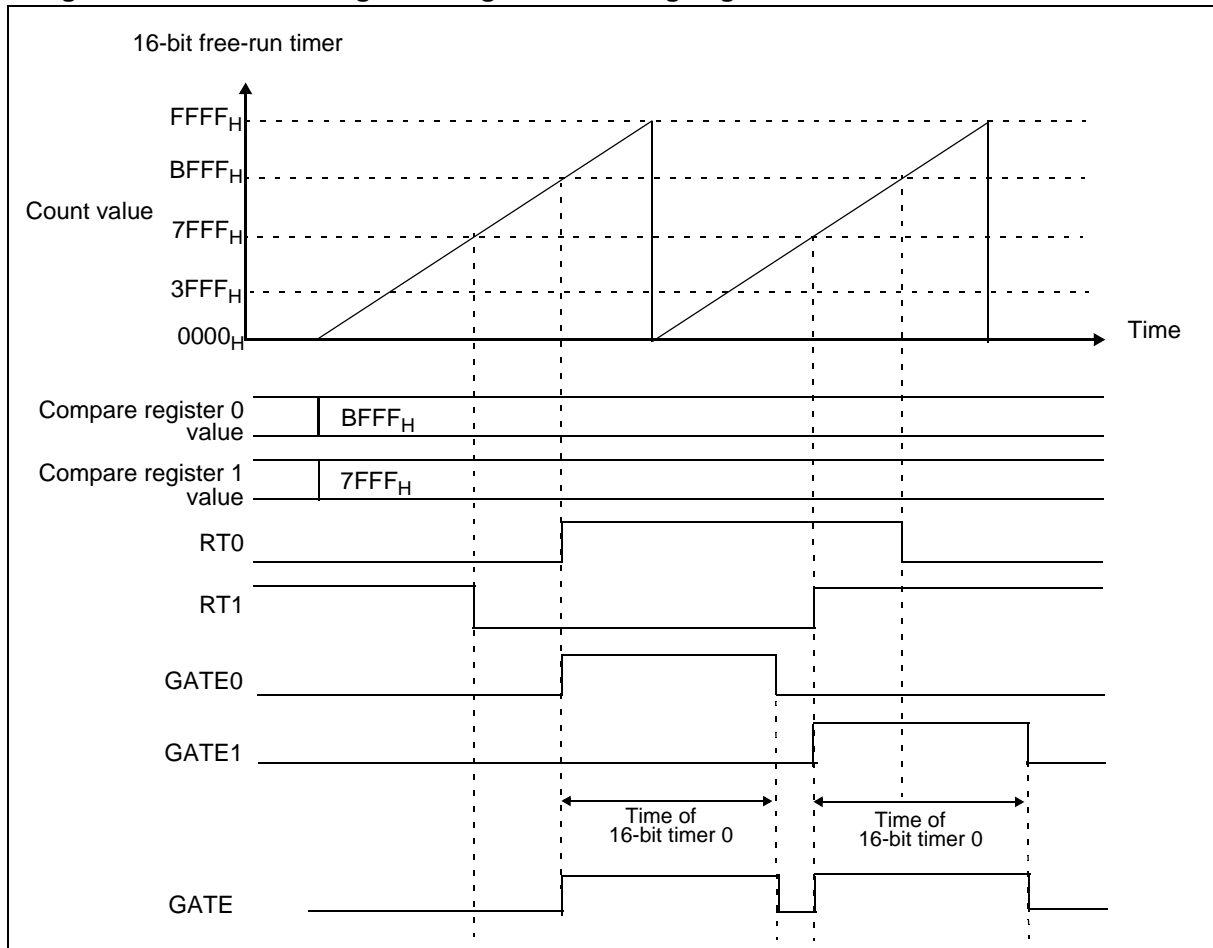
Figure 15.6-22 Generating GATE signal during RTx is at "H" level



**MB90820B Series**

■ **Generating GATE Signal from Rising Edge of Each RTx until 16-bit Timer 0/1/2 Underflow When GTENx is Active (DTCR0/1/2:TMD2 to TMD0=010<sub>B</sub>)**

**Figure 15.6-23 Generating GATE signal from rising edge of RTx until 16-bit timer underflow**



**Note:**

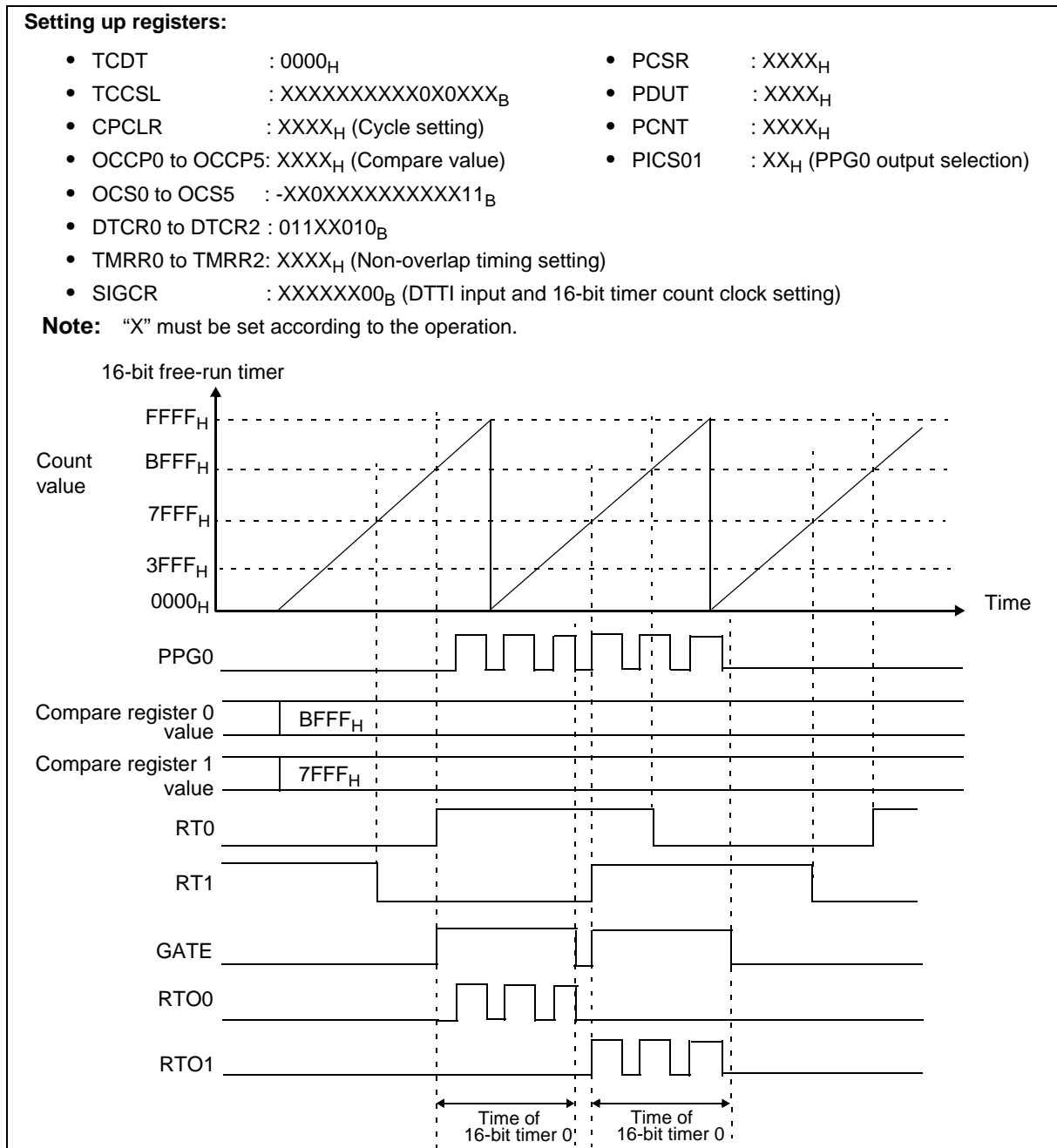
Each 16-bit timer is used for two RTs. i.e. 16-bit timer 0 is used for RT0 and RT1; 16-bit timer 1 is used for RT2 and RT3; 16-bit timer 2 is used for RT4 and RT5. Therefore, do not use an RT and attempt to start the corresponding timer that is already operating. Doing so may cause that the outputting GATE signal will be extended and malfunction will be occurred.

### 15.6.4.1 Operation in Timer Mode

With RT0 to RT5 rising edge, the 16-bit timer is reloaded and starts down-counting, and the PPG timer 0 keeps outputting to RTO0 to RTO5 until the 16-bit timer is underflow.

#### ■ PPG0 Output Pulse from Rising Edge of RT to 16-bit Timer Underflow (DTCR0/1/2:TMD2 to TMD0=010<sub>B</sub>)

Figure 15.6-24 Waveform generated when TMD2 to TMD0=010<sub>B</sub>



---

**Note:**

Each 16-bit timer is used for two RTs. i.e. 16-bit timer 0 is used for RT0 and RT1; 16-bit timer 1 is used for RT2 and RT3; 16-bit timer 2 is used for RT4 and RT5. Therefore, do not use an RT and attempt to start PPG0 which is under operation. Doing so may cause that the outputting GATE signal will be extended and malfunction will be occurred.

---

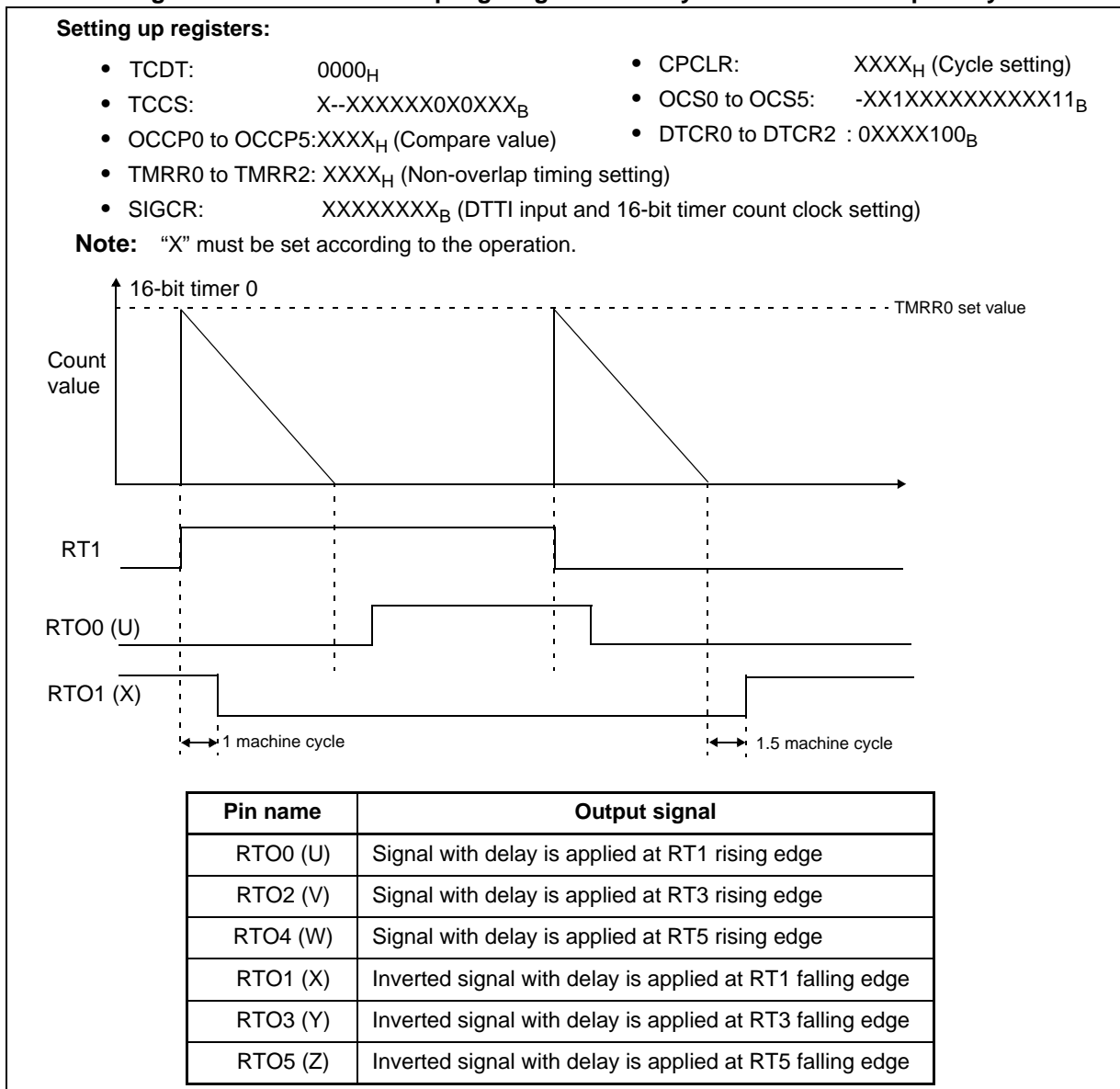
### 15.6.4.2 Operation in Dead-time Timer Mode

The dead-time generator will input the realtime output (RT1/3/5), input PPG timer 0 pulse, and output non-overlap signals (inverted signals) to external pins (RTO0 to RTO5).

#### ■ Making Non-overlap Signals by Using RT1/3/5 in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=100<sub>B</sub>)

When selecting non-overlap signal for an active level “0” (normal polarity) in DTCR0/1/2:DMOD, a delay corresponding to the non-overlap time set in the TMRR0/1/2 register (16-bit timer register) is applied. The delay is applied at a rising edge of RT1/3/5 or its falling edge. If RT1/3/5 pulse width is smaller than the set non-overlap time, the 16-bit timer will restart down-counting from TMRR0/1/2 value at the next RT’s edge.

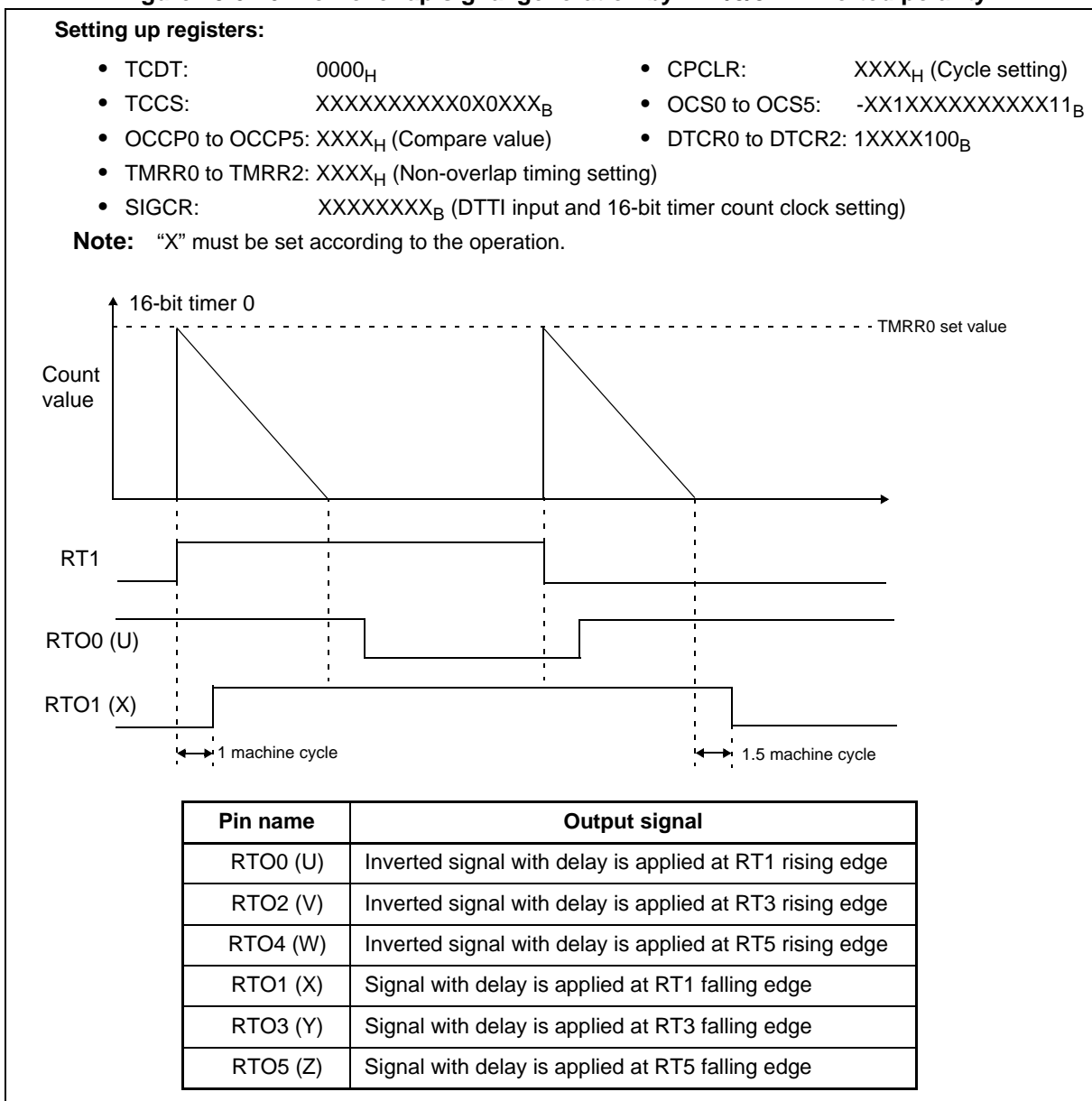
Figure 15.6-25 Non-overlap signal generation by RT1/3/5 in normal polarity



# ■ Making Non-overlap Signals by using RT1/3/5 in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=100<sub>B</sub>)

When selecting non-overlap signal for an active level “1” (inverted polarity) in DTCR0/1/2:DMOD, a delay corresponding to the non-overlap time set in the TMRR0/1/2 register (16-bit timer register) is applied. The delay is applied at a rising edge of RT1/3/5 or its falling edge. If RT1/3/5 pulse width is smaller than the set non-overlap time, the 16-bit timer will restart down-counting from TMRR0/1/2 value at the next RT’s edge.

**Figure 15.6-26 Non-overlap signal generation by RT1/3/5 in inverted polarity**



## ■ Making Non-overlap Signals by Using PPG in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=111<sub>B</sub>)

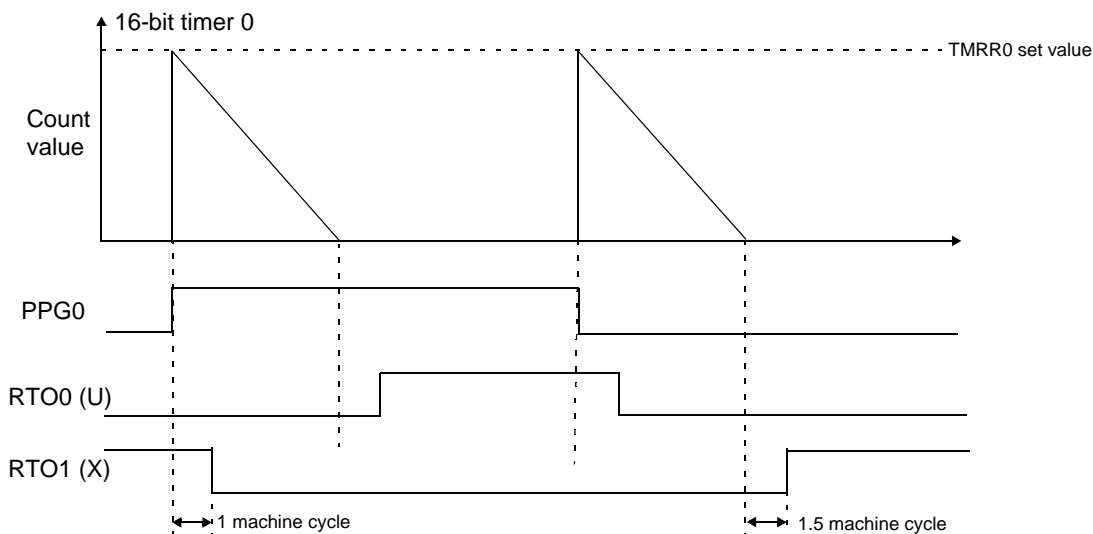
When selecting non-overlap signal for an active level “0” (normal polarity) in DTCR0/1/2:DMOD, a delay corresponding to the non-overlap time set in the TMRR0/1/2 register (16-bit timer register) is applied. The delay is applied at a rising edge of PPG timer 0 pulse signal or its inverted signal. If PPG timer pulse width is smaller than the set non-overlap time, the 16-bit timer will restart down-counting from TMMR0/1/2 value at the next edge of PPG0 pulse.

**Figure 15.6-27 Non-overlap signal generation by PPG timer0 in normal polarity**

### Setting up registers:

- TCDT: 0000<sub>H</sub>
- TCCS: XXXXXXXXXXXX0X0XX<sub>B</sub>
- CPCLR: XXX<sub>H</sub> (Cycle setting)
- OCCP0 to OCCP5: XXX<sub>H</sub> (Compare value)
- OCS0 to OCS5: -XX1XXXXXXXXXX11<sub>B</sub>
- DTCR0 to DTCR2: 0XXXX111<sub>B</sub>
- TMRR0 to TMRR2: XXX<sub>H</sub> (Non-overlap timing setting)
- SIGCR: XXXXXXX<sub>B</sub> (DTTI input and 16-bit timer count clock setting)
- PCSR : XXX<sub>H</sub>
- PDUT : XXX<sub>H</sub>
- PCNT : XXX<sub>H</sub>

**Note:** “X” must be set according to the operation.



Pin name	Output signal
RTO0 (U)	Signal with delay is applied at PPG0 rising edge
RTO2 (V)	Signal with delay is applied at PPG0 rising edge
RTO4 (W)	Signal with delay is applied at PPG0 rising edge
RTO1 (X)	Inverted signal with delay is applied at PPG0 falling edge
RTO3 (Y)	Inverted signal with delay is applied at PPG0 falling edge
RTO5 (Z)	Inverted signal with delay is applied at PPG0 falling edge

## MB90820B Series

### ■ Making Non-overlap Signals by Using PPG in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=111<sub>B</sub>)

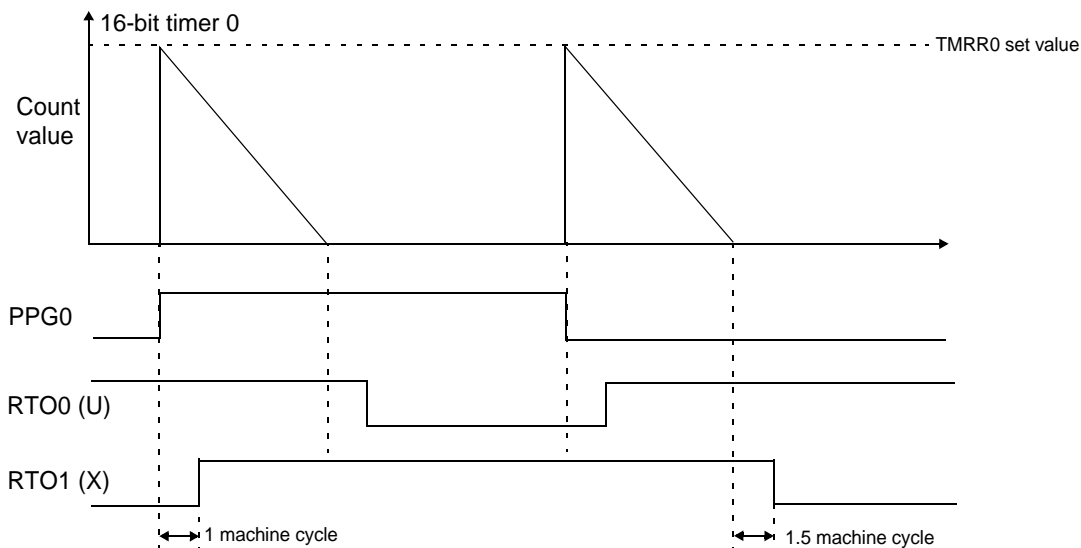
When selecting non-overlap signal for an active level “1” (inverted polarity) in DTCR0/1/2:DMOD, a delay corresponding to the non-overlap time set in the TMRR0/1/2 register (16-bit timer register) is applied. The delay is applied at a rising edge of PPG timer 0 pulse signal or its inverted signal. If PPG timer 0 pulse width is smaller than the set non-overlap time, the 16-bit timer will restart down-counting from TMMR0/1/2 value at the next edge of PPG0 pulse.

**Figure 15.6-28 Non-overlap signal generation by PPG timer0 in inverted polarity**

**Setting up registers:**

- TCDT: 0000<sub>H</sub>
- TCCS: XXXXXXXXXXXX0XXXX<sub>B</sub>
- CPCLR: XXXX<sub>H</sub> (Cycle setting)
- OCCP0 to OCCP5: XXXX<sub>H</sub> (Compare value)
- OCS0 to OCS5: -XX1XXXXXXXXXX11<sub>B</sub>
- DTCR0 to DTCR2: 1XXXX111<sub>B</sub>
- TMRR0 to TMRR2: XXXX<sub>H</sub> (Non-overlap timing setting)
- SIGCR: XXXXXXXX<sub>B</sub> (DTTI input and 16-bit timer count clock setting)
- PCSR : XXXX<sub>H</sub>
- PDUT : XXXX<sub>H</sub>
- PCNT : XXXX<sub>H</sub>

**Note:** “X” must be set according to the operation.



Pin name	Output signal
RTO0 (U)	Inverted signal with delay is applied at PPG0 rising edge
RTO2 (V)	Inverted signal with delay is applied at PPG0 rising edge
RTO4 (W)	Inverted signal with delay is applied at PPG0 rising edge
RTO1 (X)	Signal with delay is applied at PPG0 falling edge
RTO3 (Y)	Signal with delay is applied at PPG0 falling edge
RTO5 (Z)	Signal with delay is applied at PPG0 falling edge



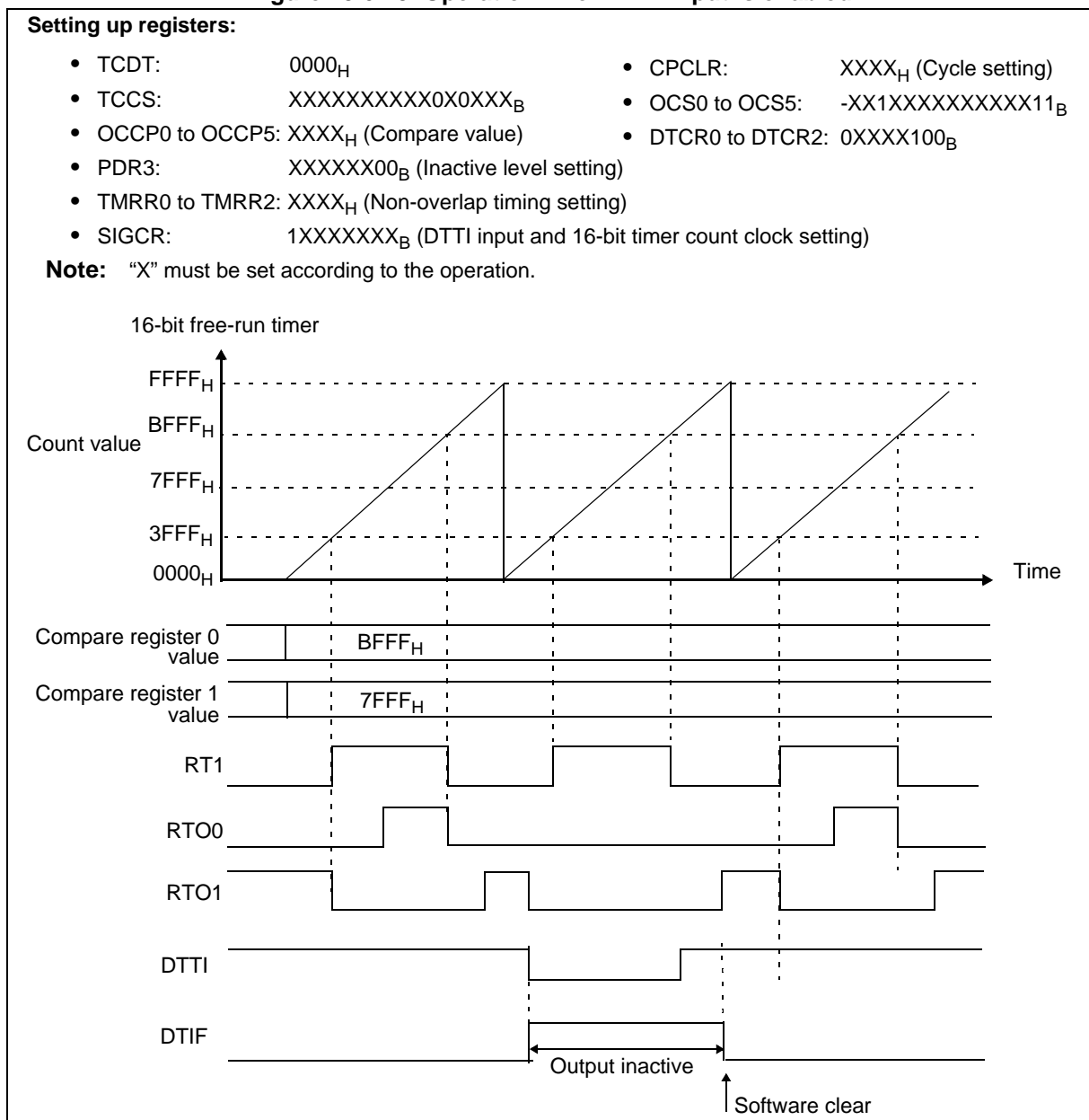
### 15.6.4.3 Operation of DTTI Pin Control

By setting “1” to waveform control register, SIGCR: bit 15 (DTIE), the output of RTO0 to RTO5 can be controlled by the DTTI pin. When “L” level in DTTI pin is detected, the output of RTO0 to RTO5 will be fixed to an inactive level until the interrupt flag, SIGCR: bit 14 (DTIF) is cleared. The inactive level of RTO0 to RTO5 can be set by PDR8 in port 8 by software.

#### ■ DTTI pin Input Operation

Even when the “L” level of DTTI pin input is detected, the timer will keep running for the waveform generator operation, but no waveform will be outputted to external pins P82/RTO0 to P87/RTO5.

Figure 15.6-29 Operation when DTTI input is enabled



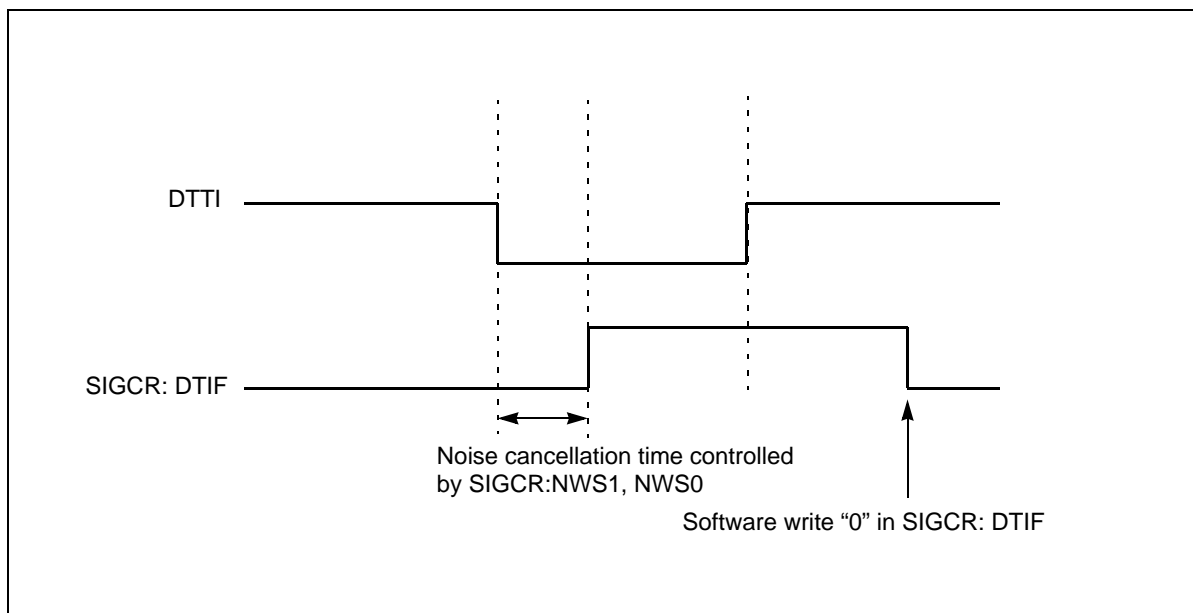
## ■ DTTI Pin Noise Cancellation Function

By setting bit 13 (NRSL) of the waveform control register (SIGCR) to “1”, the noise cancellation function for DTTI pin input is enabled. When noise cancellation function is enabled, the time for fixing an output pin (RTO0 to RTO5) to inactive level is delayed for about 4, 8, 16 or 32 machine cycles (selected by SIGCR:NWS1, NWS0). Since the noise cancellation circuit uses a peripheral function, input is invalidated even if the DTTI input is enabled in a mode such as STOP mode in which the oscillation stops.

## ■ DTTI Interrupt

When L level of DTTI is detected, DTTI interrupt flag (SIGCR:DTIF) is set to “1” after noise cancellation time is passed and an interrupt request is sent to interrupt controller.

**Figure 15.6-30 DTTI interrupt timing**



### Notes:

- If SIGCR:NWS1, NWS0 is changed within noise cancellation time, the larger value of NWS1, NWS0 will take effect.
- SIGCR:DTIF can only be cleared by software.

## 15.7 Usage Notes on the Multi-functional Timer

---

Notes on using the multi-functional timer are given below.

---

### ■ Usage Notes on the 16-bit Free-run Timer

#### ● Notes about using a program for setting

- After reset, the timer value is 0000<sub>H</sub>, zero detect interrupt flag will be set to “1” in next count clock after timer enable (TCCSL:STOP=0).
- Since the timer mode bit (TCCSL:MODE) has a buffer, changing timer mode will take effect in next count cycle. Zero detect interrupt is always generated when timer mode is changed from up-count to up-down count mode.
- Software clear (TCCSL:SCLR=1) will initialize the timer but not generate zero detect interrupt.

#### ● Notes about interrupts

- When the IRQZF bit of the timer control status register (TCCSH) is set to 1 and an interrupt request is enabled (TCCSH:IRQZE=1), control cannot be returned from interrupt processing. Always clear the IRQZF bit.
- When the ICLR bit of the timer control status register (TCCSH) is set to 1 and an interrupt request is enabled (TCCSH:ICRE=1), control cannot be returned from interrupt processing. Always clear the ICLR bit.
- Since the 16-bit free-run timer shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.

Also, when EI<sup>2</sup>OS is used by the 16-bit free-run timer, shared resource interrupts must be disabled.

### ■ Usage Notes on the 16-bit Output Compare

#### ● Notes about interrupts

- When the IOP bit of the compare control register (OCS0/2/4) is set to 1 and an interrupt request is enabled (OCS0/2/4:IOE=1), control cannot be returned from interrupt processing. Always clear the IOP bit.
- Since the 16-bit output compare shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.

Also, when EI<sup>2</sup>OS is used by the 16-bit output compare, shared resource interrupts must be disabled.

### ■ Usage Notes on the 16-bit Input Capture

#### ● Notes about interrupts

- When the ICP bit of the input capture control status register (PICSL01/ICSL23) is set to 1 and an interrupt request is enabled (PICSL01/ICSL23:ICE=1), control cannot be returned from interrupt processing. Always clear the ICP bit.
- If input capture pins (IN) level is toggled after ICP bit is set but before interrupt routine is processed, the valid edge indication bit (ICSH23:IEI3, IEI2 or PICSH01:IEI1, IEI0) will show the latest edge detected.
- Since the 16-bit input capture shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.

Also, when EI<sup>2</sup>OS is used by the 16-bit input capture, shared resource interrupts must be disabled.

## ■ Usage Notes on the Waveform Generator

### ● Notes on using a program for setting

- Change the TMD2, TMD1 and TMD0 bits of the 16-bit timer control register (DTCR0/1/2) when the waveform generator is under operation (DTCR0/1/2: TMD2 to TMD0=001<sub>B</sub>, 010<sub>B</sub>, 100<sub>B</sub> or 111<sub>B</sub>), always be sure no trigger source and 16-bit timer are not counting. Otherwise unexpected waveform in RTO will be occurred due to prescheduled output by previous trigger. But RTO output becomes normal once after timer is underflow or retriggered by new trigger source in new mode setting. Trigger source is H level of RT when DTCR0/1/2: TMD2 to TMD0=001<sub>B</sub>, rising edge of RT when TMD2 to TMD0=010<sub>B</sub>, rising/falling edge of RT when TMD2 to TMD0=100<sub>B</sub> or rising/falling edge of PPG0 when TMD2 to TMD0=111<sub>B</sub>.  
For example, changing TMD2 to TMD0 from 100<sub>B</sub> to 111<sub>B</sub>, you can set in following procedures  
1) set TMRR0/1/2 to a very small value like 0001<sub>H</sub>  
2) set RT1/3/5 to output “L”/”H” and wait until timer 0/1/2 underflow  
3) change mode bits TMD2, TMD1 and TMD0 and corresponding setting  
4) corrected output waveform will appear in RTO pins one machine cycle later
- Writing a value in 16-bit timer register (TMRR0/1/2) during timer counting, new value will be valid at the next timer trigger. And always be sure to use a word transfer instruction (MOVW A, dir, etc.) to access timer register.
- Change the DCK2, DCK1 and DCK0 bits of the waveform control register (SIGCR) when the timer is not counting.
- Change the NWS1 and NWS0 bits of waveform control register (SIGCR) when the noise cancellation function is disabled (SIGCR: NRSL=0).

### ● Notes about interrupts

- When the TMIF bit of the 16-bit timer control register (DTCR0/1/2) is set to 1 and an interrupt request is enabled (DTCR0/1/2:TMIE=1), control cannot be returned from interrupt processing. Always clear the TMIF bit.
- When the DTIF bit of the waveform control register (SIGCR) is set to 1, control cannot be returned from interrupt processing. Always clear the DTIF bit.
- Since the waveform generator shares an interrupt vector with other resource, interrupt causes must be checked carefully by the interrupt processing routine when interrupts are used.  
Also, when EI<sup>2</sup>OS is used by the waveform generator, shared resource interrupts must be disabled.



# **CHAPTER 16**

---

## ***DELAYED INTERRUPT GENERATOR MODULE***

**This chapter describes the functions and operation of the delayed interrupt generator module.**

- 16.1 Overview of the Delayed Interrupt Generator Module
- 16.2 Delayed Interrupt Generator Module Register
- 16.3 Operation of the Delayed Interrupt Generator Module
- 16.4 Usage Notes on the Delayed Interrupt Generator Module

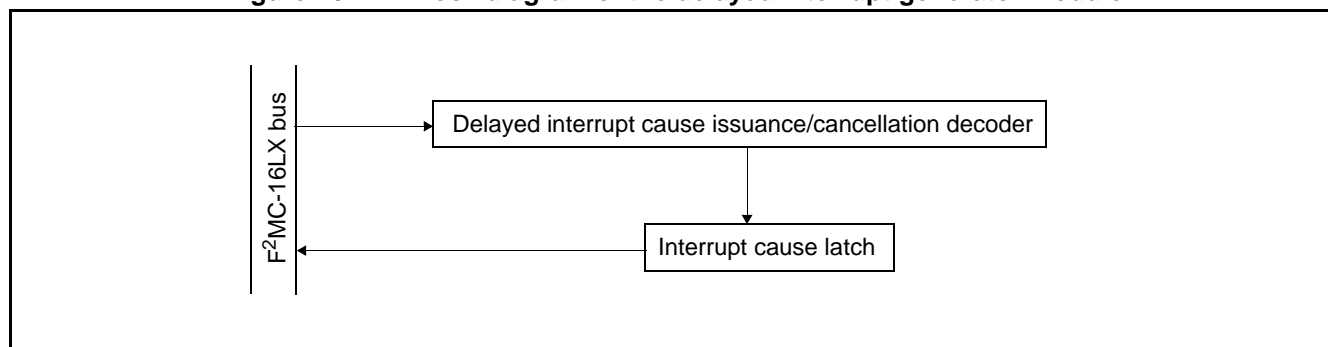
## 16.1 Overview of the Delayed Interrupt Generator Module

The delayed interrupt generator module generates interrupts for task switching. By using this module, software can issue and cancel interrupt requests for the F<sup>2</sup>MC-16LX CPU.

### ■ Block Diagram of the Delayed Interrupt Generator Module

Figure 16.1-1 shows the block diagram of the delayed interrupt generator module.

Figure 16.1-1 Block diagram of the delayed interrupt generator module



MB90820B Series

16.2 Delayed Interrupt Generator Module Register

This section lists the delayed interrupt generator module register.

■ Delayed Interrupt Generator Module Register (DIRR)

Figure 16.2-1 Delayed interrupt generator module register (DIRR)

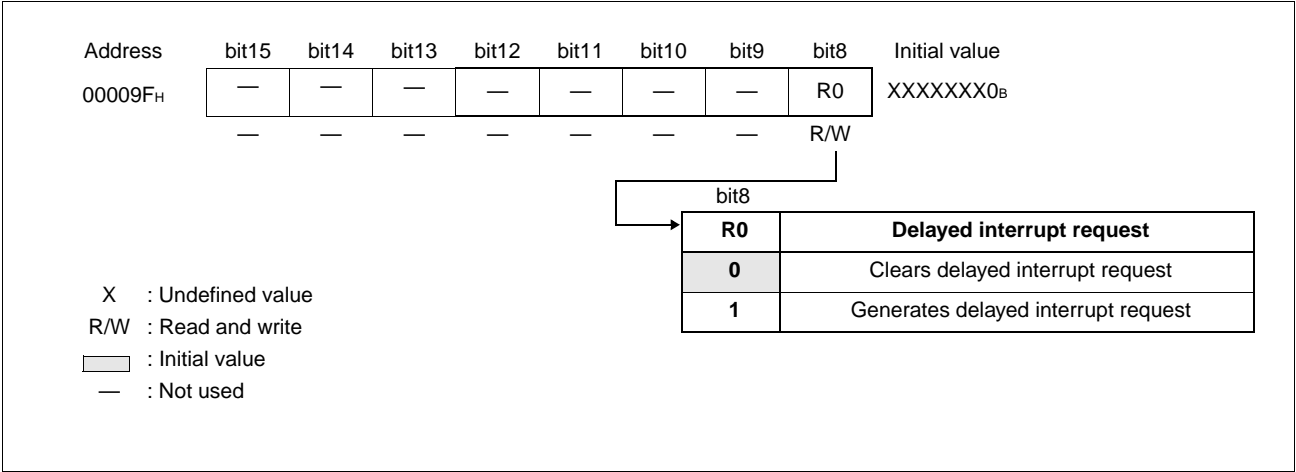


Table 16.2-1 Function of delayed interrupt request output/cancellation register (DIRR)

Bit name		Function
bit15 to bit9	Reserved bits	<ul style="list-style-type: none"><li>Both "0" and "1" may be written to the reserved bits, writing to these bits has no effect on the operation.</li></ul>
bit8	R0: Delayed interrupt request bit	<ul style="list-style-type: none"><li>This bit is used to controls the generation or clearing of a delayed interrupt request.</li><li>Writing "0" to this bit clears the delayed interrupt request.</li><li>Writing "1" to this bit generates a delayed interrupt request.</li><li>The register is cleared at reset.</li><li>Both "0" and "1" may be written to the reserved bit area. However, the set bit and clear bit instructions should be used to access this register to prepare for future expansion.</li></ul>



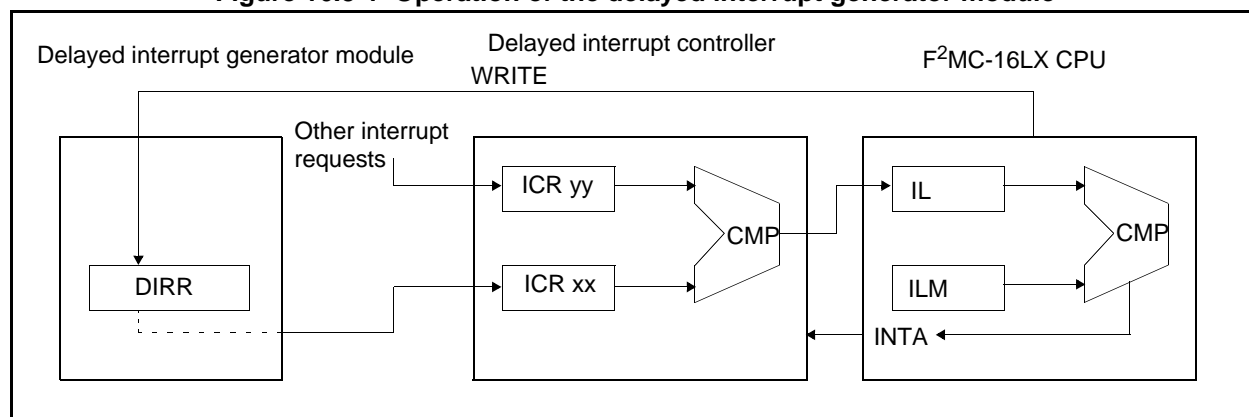
## 16.3 Operation of the Delayed Interrupt Generator Module

When software causes the CPU to write 1 to the R0 bit of DIRR, the request latch in the delayed interrupt generator module is set and an interrupt request is generated to the interrupt controller.

### ■ Operation of the Delayed Interrupt Generator Module

When software causes the CPU to write 1 to the R0 bit of DIRR, the request latch in the delayed interrupt generator module is set and an interrupt request is generated to the interrupt controller. If the priority of other interrupt requests is lower than that of this interrupt or no other interrupt request is generated, the interrupt controller generates an interrupt request to the F<sup>2</sup>MC-16LX CPU. The F<sup>2</sup>MC-16LX CPU compares the ILM bit of the internal CCR register and the interrupt request. When the request level is higher than that of the ILM bit, the CPU starts the delayed interrupt processing microprogram immediately after execution of the current instruction ends. As a result, the interrupt processing routine for this interrupt is executed. This interrupt request is cleared and task switching is done by writing 0 to the R0 bit of user program in the interrupt processing routine. Figure 16.3-1 Operation of the delayed interrupt generator module shows the operation of the delayed interrupt generator module.

**Figure 16.3-1 Operation of the delayed interrupt generator module**



**MB90820B Series****16.4 Usage Notes on the Delayed Interrupt Generator Module**

---

Notes on using the delayed interrupt generator module are given below.

---

**■ Usage Notes on the Delayed Interrupt Request Latch**

This latch is set by writing "1" to the R0 bit of DIRR and cleared by writing "0" to the same bit. Note that interrupt processing is restarted at the moment control returns from interrupt processing unless software is created to clear the cause in the interrupt processing routine.



# **CHAPTER 17**

---

# ***DTP/EXTERNAL INTERRUPT CIRCUIT***

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

- 17.1 Overview of the DTP/External Interrupt Circuit
- 17.2 Block Diagram of the DTP/External Interrupt Circuit
- 17.3 DTP/External Interrupt Circuit Pins
- 17.4 DTP/External Interrupt Circuit Registers
- 17.5 Operation of the DTP/External Interrupt Circuit
- 17.6 Usage Notes on the DTP/External Interrupt Circuit

## 17.1 Overview of the DTP/External Interrupt Circuit

The data transfer peripheral (DTP)/external interrupt circuit is located between external peripherals and the F<sup>2</sup>MC-16LX CPU. It receives interrupt requests and data transfer requests from external peripherals and passes them to the CPU to generate interrupt requests or activate the extended intelligent I/O service (EI<sup>2</sup>OS).

### ■ DTP/External Interrupt Functions

The DTP/external interrupt circuit detects the interrupt request which external peripherals generate. The interrupt request is output to the CPU using the same procedure it uses for peripheral function interrupts and generates interrupts or activates the extended intelligent I/O service (EI<sup>2</sup>OS).

If the extended intelligent I/O service (EI<sup>2</sup>OS) is disabled by the interrupt control register (ICR:ISE=0) when an interrupt request is accepted by the CPU, the circuit executes its external interrupt function and branches to an interrupt processing. If EI<sup>2</sup>OS is enabled (ICR: ISE=1), the circuit executes its DTP function, which performs automatic data transfer using EI<sup>2</sup>OS and branches to an interrupt processing routine after the data transfer has been performed a specified number of times.

Table 17.1-1 provides an overview of the DTP/external interrupt circuit.

**Table 17.1-1 Overview of the DTP/external interrupt circuit**

	External interrupt function	DTP function
Input pins	Eight (P10/INT0/DTTI to P16/INT6, P51/INT7)	
Interrupt cause	By using the request level setting register (ELVR), the level or edge to be detected can be selected for each pin	
	Input of H level, L level, rising edge or falling edge	Input of H level or L level
Interrupt number	#20 (14 <sub>H</sub> ), #22 (16 <sub>H</sub> ), #25 (19 <sub>H</sub> ), #26 (1A <sub>H</sub> ), #27 (1B <sub>H</sub> ), #28 (1C <sub>H</sub> )	
Interrupt control	The output of interrupt requests is enabled and disabled using the DTP/external interrupt enable register (ENIR)	
Interrupt flag	Interrupt causes are stored in the DTP/external interrupt cause register (EIRR)	
Processing selection	EI <sup>2</sup> OS is disabled (ICR: ISE = 0)	EI <sup>2</sup> OS is enabled (ICR: ISE = 1)
Processing	The circuit branches to an external interrupt processing	The circuit performs automatic data transfer using EI <sup>2</sup> OS for a specified number of times and then branches to an interrupt processing

ICR: Interrupt control register

# ■ Interrupt of the DTP/External Interrupt Circuit and EI<sup>2</sup>OS

Table 17.1-2 Interrupt of the DTP/external interrupt circuit and EI<sup>2</sup>OS

Channel	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
INT0/INT1	#20 (14 <sub>H</sub> )	ICR04	0000B4 <sub>H</sub>	FFFFAC <sub>H</sub>	FFFFAD <sub>H</sub>	FFFFAE <sub>H</sub>	O
INT2/INT3	#22 (16 <sub>H</sub> )	ICR05	0000B5 <sub>H</sub>	FFFA4 <sub>H</sub>	FFFA5 <sub>H</sub>	FFFA6 <sub>H</sub>	
INT4	#25 (19 <sub>H</sub> )	ICR07	0000B7 <sub>H</sub>	FFF98 <sub>H</sub>	FFF99 <sub>H</sub>	FFF9A <sub>H</sub>	
INT5	#26 (1A <sub>H</sub> )			FFF94 <sub>H</sub>	FFF95 <sub>H</sub>	FFF96 <sub>H</sub>	
INT6	#27 (1B <sub>H</sub> )	ICR08	0000B8 <sub>H</sub>	FFF90 <sub>H</sub>	FFF91 <sub>H</sub>	FFF92 <sub>H</sub>	
INT7	#28 (1C <sub>H</sub> )			FFF8C <sub>H</sub>	FFF8D <sub>H</sub>	FFF8E <sub>H</sub>	

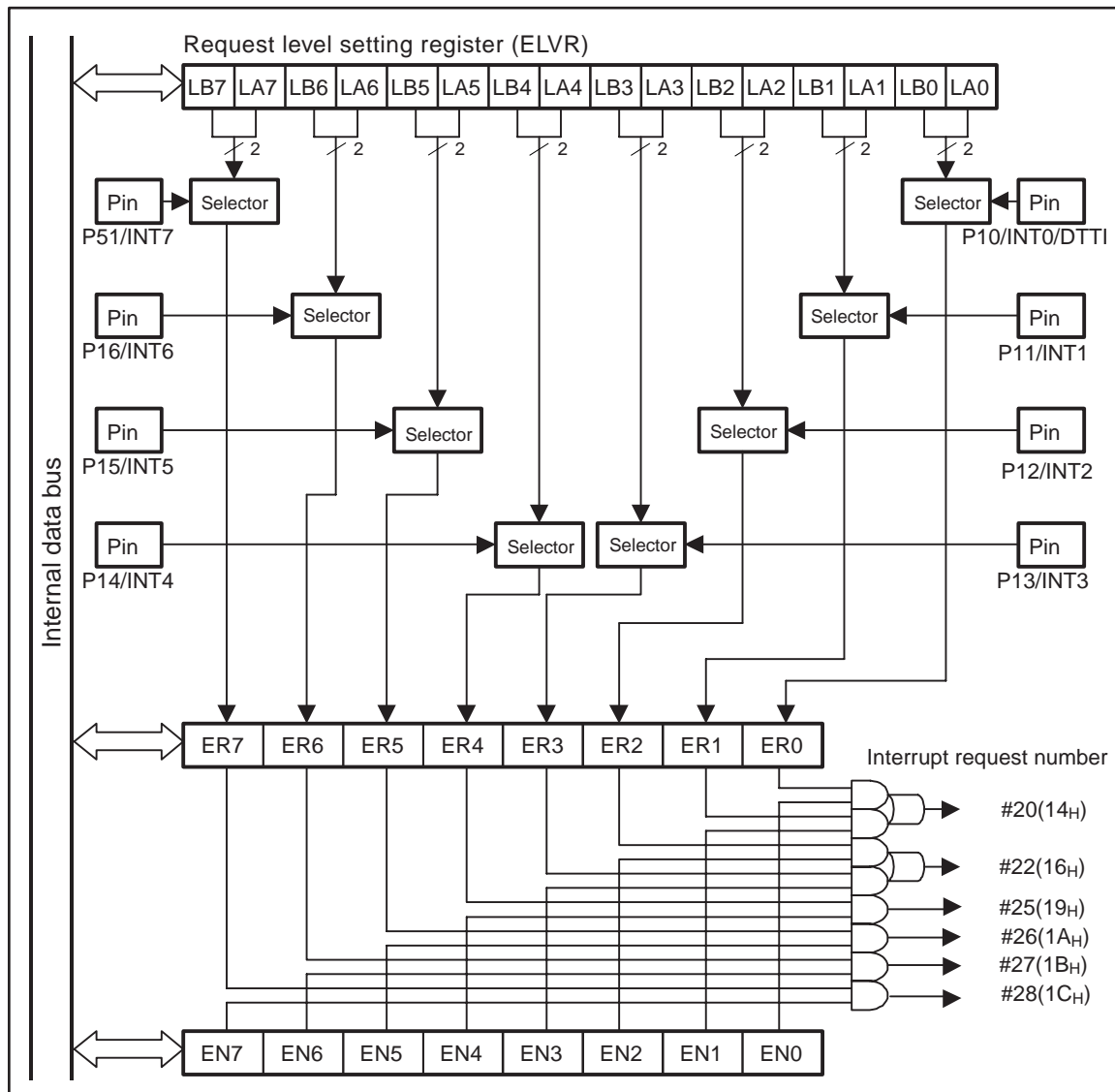
O: Can be used and interrupt request flag is cleared by EI<sup>2</sup>OS interrupt clear signal.

## 17.2 Block Diagram of the DTP/External Interrupt Circuit

The DTP/external interrupt circuit consists of four blocks, and the block diagram is shown in Figure 17.2-1.

### ■ Block Diagram of the DTP/External Interrupt Circuit

Figure 17.2-1 Block diagram of the DTP/external interrupt circuit



#### ● DTP/external interrupt input detection circuit

Upon detecting the level or edge selected for each pin by the interrupt request level setting register (ELVR), this circuit sets "1" to the IR bit of the DTP/external interrupt cause register (EIRR) that corresponds to the pin.

- Request level setting register (ELVR)

This register selects the effective level or edge for each pin.

- DTP/external interrupt cause register (EIRR)

This register stores DTP/external interrupt causes. It contains DTP/external interrupt request flag bit for each pin. The bit is set to "1" if a valid signal is input to the corresponding pin.

- DTP/external interrupt enable register (ENIR)

This register enables and disables DTP/external interrupt request of external peripherals.



## 17.3 DTP/External Interrupt Circuit Pins

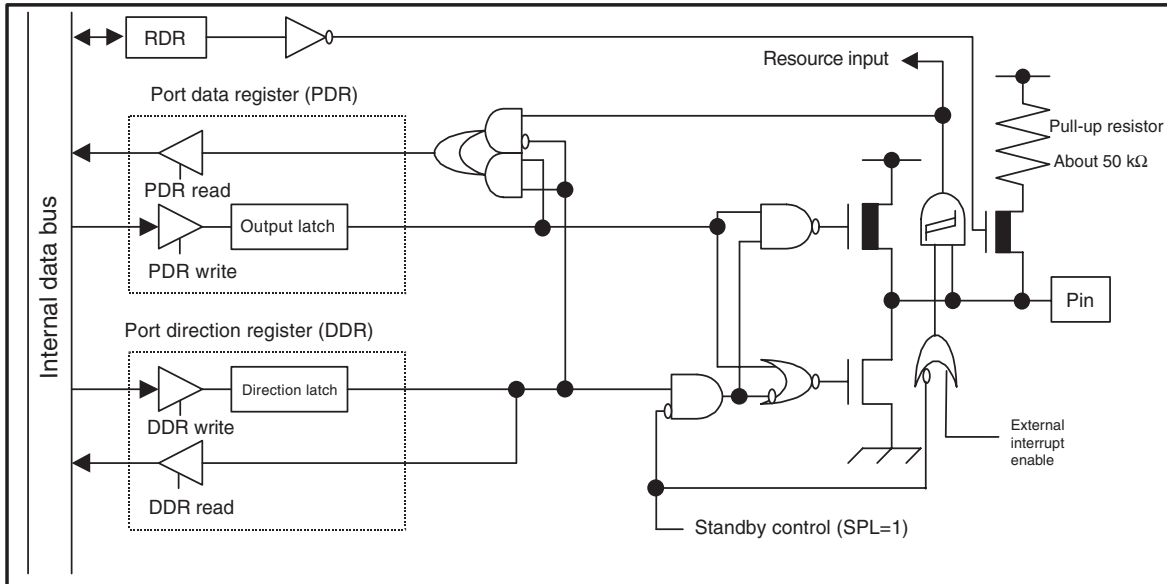
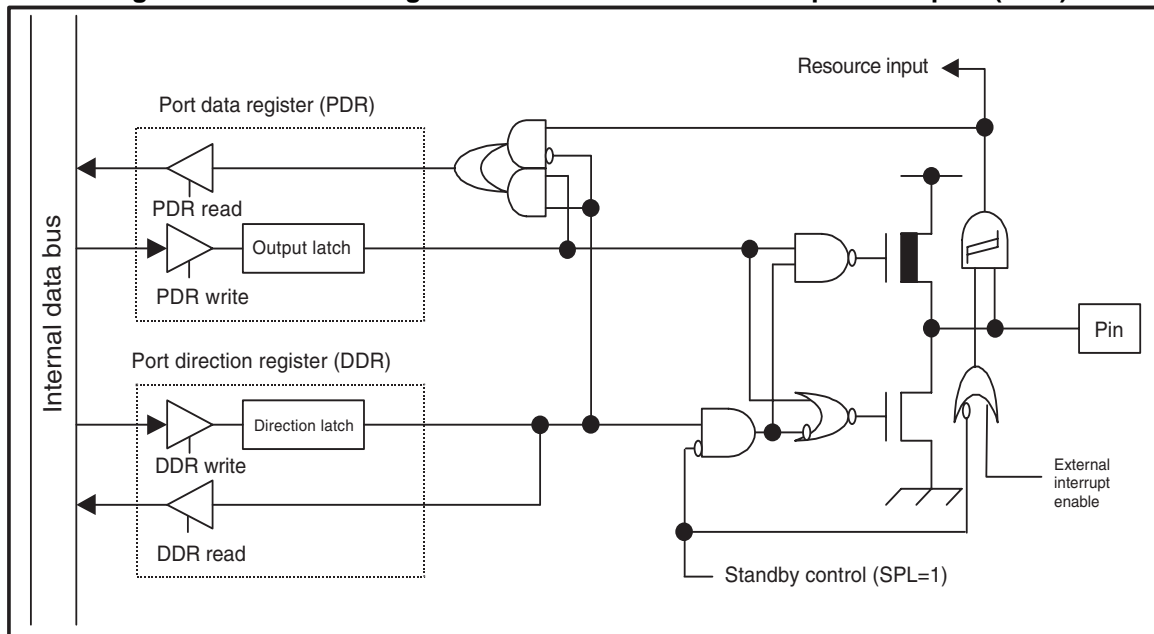
This section describes the DTP/external interrupt circuit pins and provides a pin block diagram.

### ■ DTP/External Interrupt Circuit Pins

The DTP/external interrupt circuit pins are also used as general-purpose I/O ports. Table 17.3-1 lists the pin functions, I/O formats, and settings required to use the DTP/external interrupt circuit.

**Table 17.3-1 DTP/external interrupt circuit pins**

Pin name	Pin function	I/O format	Pull-up operation	Standby control	Setting required to use pins
P10/ INT0/ DTTI	Port 1 input-output/external interrupt input/resource input-output	CMOS output / CMOS hysteresis input	Selectable	Provided	Set the pin as an input port (DDR1: bit8 = 0)
P11/INT1					Set the pin as an input port (DDR1: bit9 = 0)
P12/INT2					Set the pin as an input port (DDR1: bit10 = 0)
P13/INT3					Set the pin as an input port (DDR1: bit11 = 0)
P14/INT4					Set the pin as an input port (DDR1: bit12 = 0)
P15/INT5					Set the pin as an input port (DDR1: bit13 = 0)
P16/INT6					Set the pin as an input port (DDR1: bit14 = 0)
P51/INT7	Port 5 input-output/external interrupt input		Not provided		Set the pin as an input port (DDR5: bit9 = 0)

**MB90820B Series****■ Block Diagram of the DTP/External Interrupt Circuit Pins****Figure 17.3-1 Block diagram of the DTP/external interrupt circuit pins (INT0 to INT6)****Figure 17.3-2 Block diagram of the DTP/external interrupt circuit pins (INT7)**

## 17.4 DTP/External Interrupt Circuit Registers

This section describes DTP/external interrupt circuit registers.

Figure 17.4-1 DTP/external interrupt circuit registers

### DTP / Interrupt Cause Register

bit	15	14	13	12	11	10	9	8	
Address: 000031 <sub>H</sub>	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	EIRR
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	

### DTP / Interrupt Enable Register

bit	7	6	5	4	3	2	1	0	
Address: 000030 <sub>H</sub>	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	ENIR
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	

### Request Level Setting Register (Upper)

bit	15	14	13	12	11	10	9	8	
Address: 000033 <sub>H</sub>	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	ELVRH
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	

### Request Level Setting Register (Lower)

bit	7	6	5	4	3	2	1	0	
Address: 000032 <sub>H</sub>	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	ELVRL
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	

MB90820B Series

17.4.1 DTP/external interrupt cause register (EIRR)

The DTP/external interrupt cause register (EIRR) stores and clears interrupt causes.

■ DTP/External Interrupt Cause Register (EIRR)

Figure 17.4-2 DTP/external interrupt cause register (EIRR)

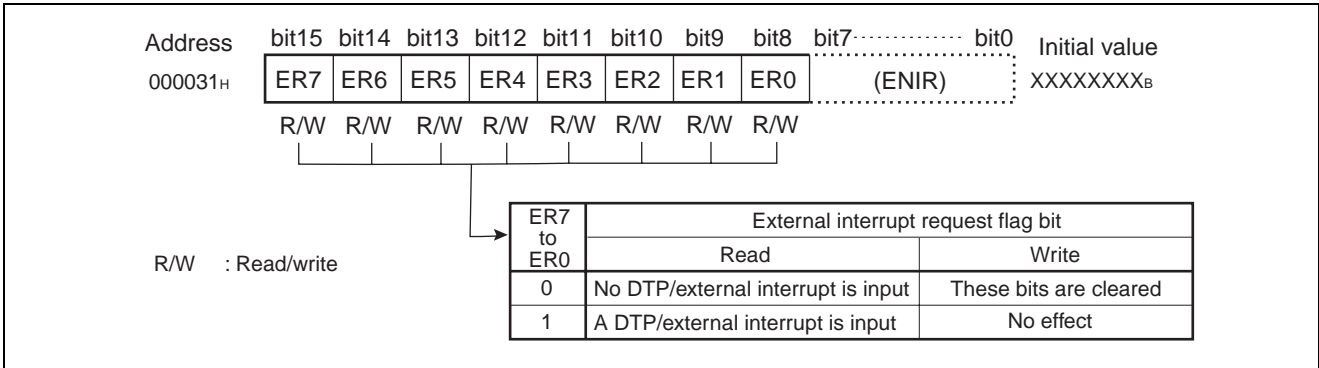


Table 17.4-1 Function description of each bit in the DTP/interrupt cause register (EIRR)

Bit name		Function
bit15 to bit8	ER7 to ER0: DTP/External interrupt request flag bits	<div><ul style="list-style-type: none"><li>Each of these bits is set to "1" if a signal with the edge or level selected by bits LB7, LA7 to LB0, LA0 of the request level setting register (ELVR) is input to the DTP/external external interrupt pin (stores an interrupt cause).</li><li>If these bits and corresponding bits EN7 to EN0 of the DTP/external interrupt enable register (ENIR) are "1", an interrupt request is output to the CPU.</li><li>Writing "0" to these bits clears the bit. Writing "1" to these bits does not change the bit value and has no effect on other bits.</li></ul><div>Note: If more than one external interrupt request output is enabled (ENIR: EN7 to EN0 = 1), clear only the bit that caused the CPU to accept an interrupt (bits ER7 to ER0 set to "1"). Do not clear the other bits without a reason.</div><div>Reference: When the extended intelligent I/O service (EI<sup>2</sup>OS) is activated, the corresponding external interrupt request flag bit is automatically cleared when the transfer of one data ends.</div></div>

## 17.4.2 DTP/external interrupt enable register (ENIR)

The DTP/external interrupt enable register (ENIR) enables and disables the output of DTP/external interrupt requests of external peripherals to the CPU.

### ■ DTP/External Interrupt Interrupt Enable Register (ENIR)

Figure 17.4-3 DTP/external interrupt enable register (ENIR)

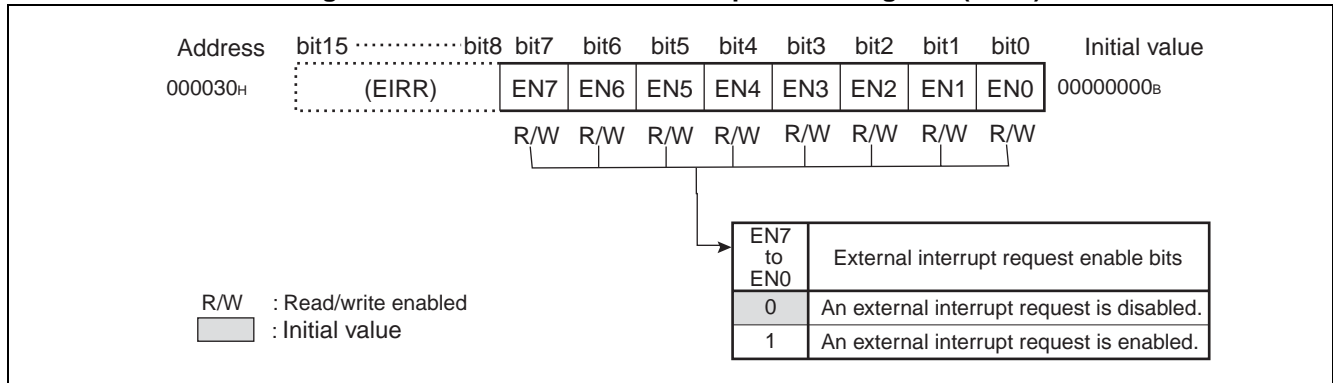


Table 17.4-2 Function description of each bit in the DTP/external interrupt enable register (ENIR)

Bit name		Function
bit7 to bit0	EN7 to EN0: DTP/External interrupt request enable bits	<p>Each of these bits enables and disables the output of interrupt requests to the CPU. If these bits and corresponding bits ER7 to ER0 of the DTP/external interrupt cause register (EIRR) are "1", an interrupt request is output to the CPU.</p> <p><b>Reference:</b></p> <ul style="list-style-type: none"><li>- To use a DTP/external interrupt pin, write "0" to the corresponding bit of the port direction register to set the pin as an input port.</li><li>- The states of the DTP/external interrupt pins can be read directly using the port data register regardless of the states of external interrupt request enable bits.</li><li>- Bits ER7 to ER0 of the DTP/external interrupt cause register (EIRR) are set to "1" if an interrupt cause is detected regardless of the values of external interrupt request enable bits.</li></ul>

Notes:

- The value of a DTP/external interrupt request flag bits (EIRR:ER) is only valid if the corresponding DTP/external interrupt request enable bit (ENIR:EN) is set to "1". When DTP/external interrupts are disabled (ENIR:EN=0), it is possible for the DTP/external interrupt cause bit to be set regardless of whether a DTP/external interrupt cause is present.
- Always clear the corresponding DTP/external interrupt request flag bits (EIRR:ER) immediately before enabling DTP/external interrupts (ENIR:EN=1).

**Table 17.4-3 Correspondence among the DTP/external interrupt pin, interrupt request flag bit, and interrupt enable bit**

DTP/external interrupt pin	Interrupt number	DTP/external interrupt request flag bit	DTP/external interrupt request enable bit
P51/INT7	#28 (1C <sub>H</sub> )	ER7	EN7
P16/INT6	#27 (1B <sub>H</sub> )	ER6	EN6
P15/INT5	#26 (1A <sub>H</sub> )	ER5	EN5
P14/INT4	#25 (19 <sub>H</sub> )	ER4	EN4
P13/INT3	#22 (16 <sub>H</sub> )	ER3	EN3
P12/INT2		ER2	EN2
P11/INT1	#20 (14 <sub>H</sub> )	ER1	EN1
P10/INT0/DTTI		ER0	EN0

### 17.4.3 Request Level Setting Register (ELVR)

The request level setting register (ELVR) selects the level or edge of the signal input to each DTP/external interrupt pin that is to be detected as a DTP/external interrupt cause.

#### ■ Request Level Setting Register (ELVR)

Figure 17.4-4 Request level setting register (ELVR)

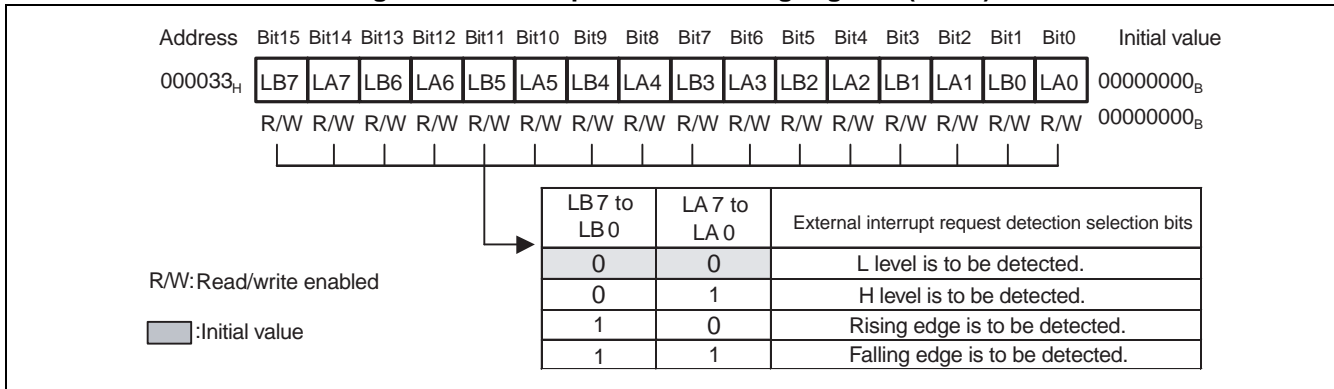


Table 17.4-4 Function description of each bit in the request level setting register (ELVR)

Bit name		Function
bit15 to bit0	LB7 to LB0, LA7 to LA0: Request detection selection bits	<ul style="list-style-type: none"><li>Each of these bits selects the level or edge of the signal input to the DTP/external interrupt pin to be detected as a DTP/external interrupt cause.</li><li>The external interrupt is selected from two types of level or edge, and the EI<sup>2</sup>OS is selected from two types of level.</li></ul> <p><b>Reference:</b></p> <p>If the selected detection signal is input to a DTP/external interrupt pin, the DTP/external interrupt request flag bit is set to "1" regardless of the settings of the DTP/external interrupt enable register (ENIR:EN=0).</p>

Note:

Always clear the corresponding DTP/external interrupt request flag bits (EIRR:ER) immediately before enabling DTP/external interrupts (ENIR:EN=1).

**Table 17.4-5 Correspondence between request level setting register (ELVR) and each channel**

DTP/external interrupt pin	Interrupt number	Bit name
P51/INT7	#28 (1C <sub>H</sub> )	LB7, LA7
P16/INT6	#27 (1B <sub>H</sub> )	LB6, LA6
P15/INT5	#26 (1A <sub>H</sub> )	LB5, LA5
P14/INT4	#25 (19 <sub>H</sub> )	LB4, LA4
P13/INT3	#22 (16 <sub>H</sub> )	LB3, LA3
P12/INT2		LB2, LA2
P11/INT1	#20 (14 <sub>H</sub> )	LB1, LA1
P10/INT0/DTTI		LB0, LA0



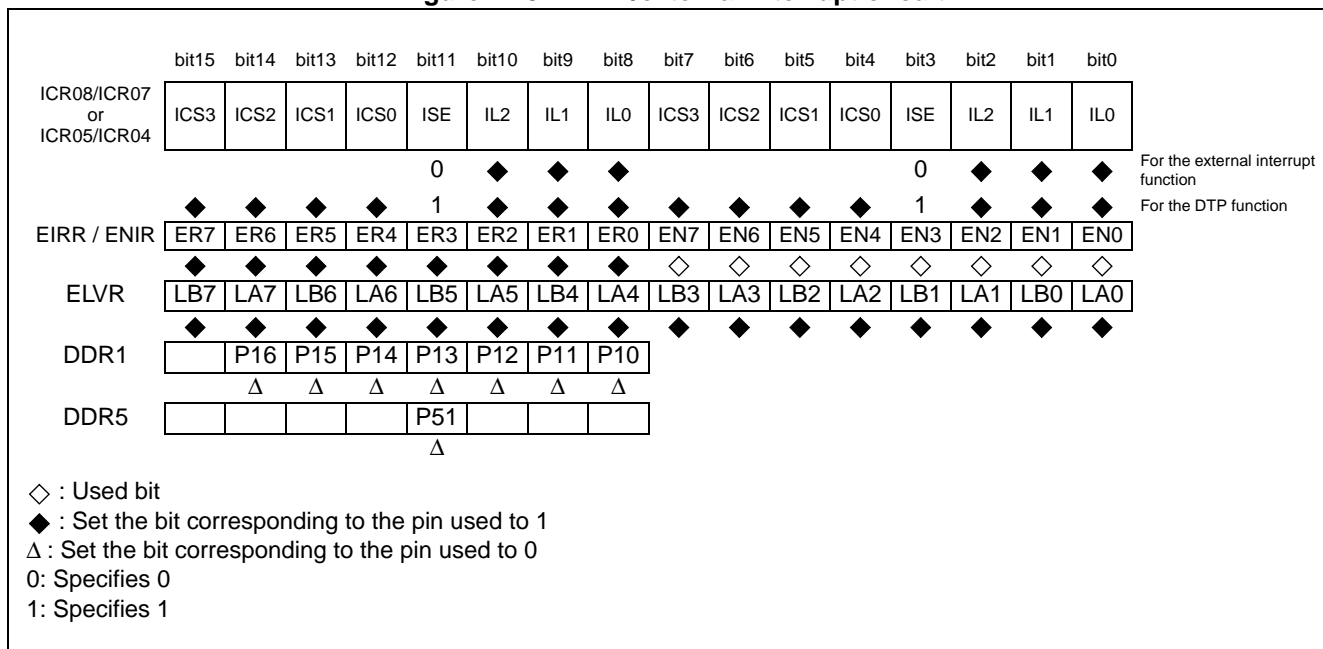
## 17.5 Operation of the DTP/External Interrupt Circuit

The DTP/external interrupt circuit provides the external interrupt function and the DTP function. This section describes the settings required for each function and the operation of the circuit.

### ■ Setting the DTP/External Interrupt Circuit

Figure 17.5-1 shows the settings required to operate the DTP/external interrupt circuit.

Figure 17.5-1 DTP/external interrupt circuit



### ● Setting procedure

Set the DTP/external interrupt circuit registers with the following procedure:

1. Set the general-purpose I/O ports that share pins with external interrupt inputs to input port.
2. Set the target bit of the DTP/interrupt enable register (ENIR) to disable interrupts.
3. Set the target bit of the request level setting register (ELVR).
4. Clear the target bit of the DTP/interrupt cause register (EIRR).
5. Set the target bit of the DTP/interrupt enable register (ENIR) to enable interrupts.

The procedure for setting the DTP/external interrupt circuit registers must start with disabling the output of external interrupt requests (ENIR:EN7 to EN0 = 0). Before the output of DTP/external interrupt requests can be enabled (ENIR:EN7 to EN0 = 1), the corresponding DTP/external interrupt request flag bits must be cleared (ENIR:EN7 to EN0 = 0).

This is in order to avoid interrupt requests from being generated accidentally while the registers are being set.

● Switching between the external interrupt function and the DTP function

Switching between the external interrupt function and the DTP function is accomplished by the ISE bit of the corresponding interrupt control register (ICR). If the ISE bit is "1", the extended intelligent I/O service (EI<sup>2</sup>OS) is enabled and the circuit executes its DTP function. If it is "0", EI<sup>2</sup>OS is disabled and the circuit executes the its external interrupt function.

Note :

If multiple interrupt requests are assigned to a single ICR register, the interrupt level (IL2 to IL0) is common to all of the interrupt requests. As a rule, when one interrupt request uses EI<sup>2</sup>OS, the other interrupt requests cannot use it.

■ Operation of the DTP/External Interrupt Circuit

Table 17.5-1 shows the control bits and interrupt causes of the DTP/external interrupt circuit.

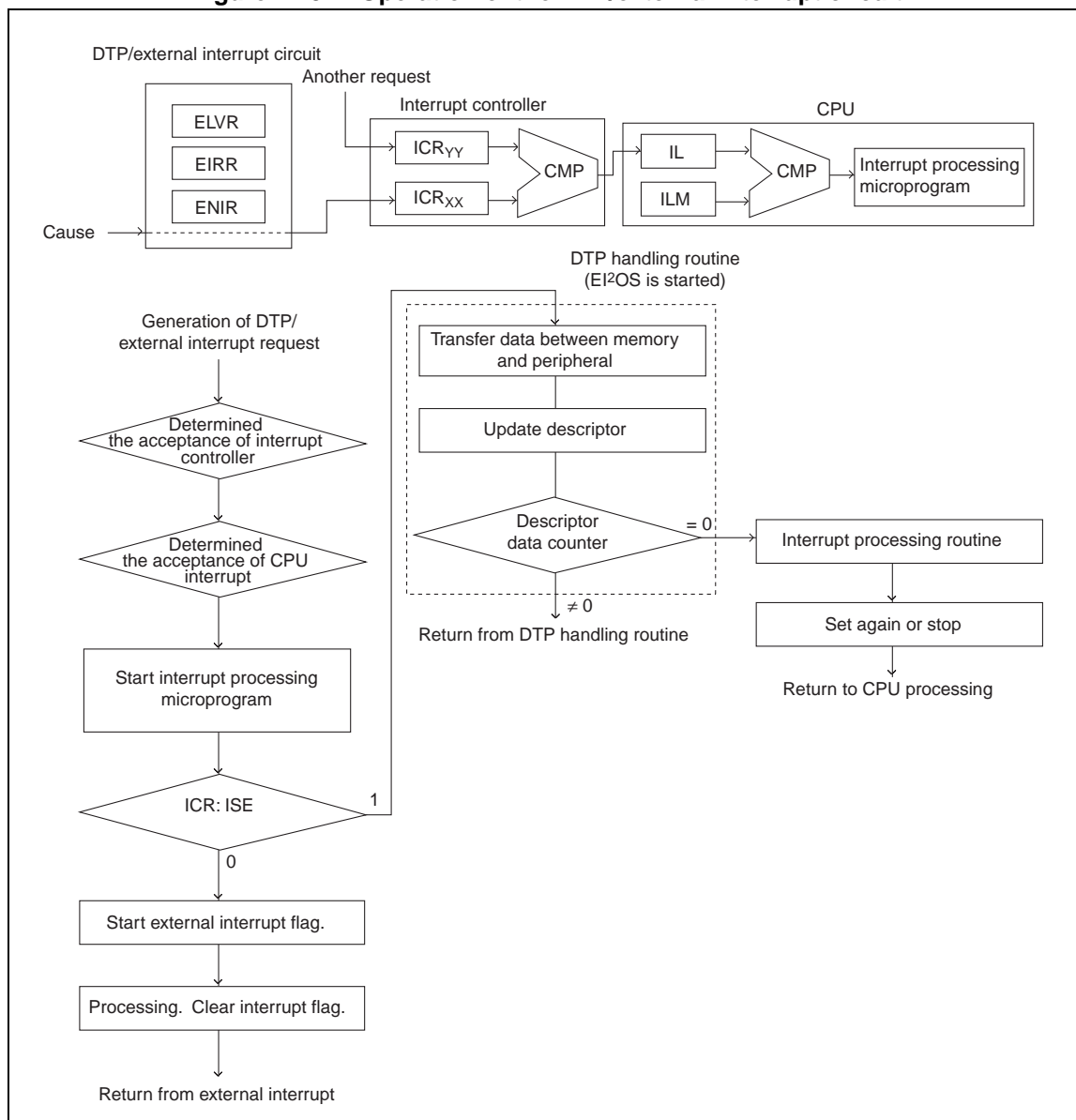
**Table 17.5-1 Control bit and interrupt cause of the DTP/external interrupt circuit**

	DTP/external interrupt circuit
Interrupt request flag bit	EIRR: ER7 to ER0
Interrupt request enable bit	ENIR: EN7 to EN0
Interrupt cause	Input of an effective edge or level to pin INT7 to INT0

If the interrupt request of the DTP/external interrupt is outputted to the interrupt microcontroller and if the ISE bit of the ICR is "0", the interrupt processing microprogram is executed. If it is "1", the extended intelligent I/O service (EI<sup>2</sup>OS) microprogram is executed.

Figure 17.5-2 shows the operation of the DTP/external interrupt circuit.

**Figure 17.5-2 Operation of the DTP/external interrupt circuit**



## MB90820B Series

### 17.5.1 External Interrupt Function

---

The DTP/external interrupt circuit has an external interrupt function that generates an interrupt request when a selected signal (edge or level) is input to a DTP/external interrupt pin.

---

#### ■ External Interrupt Function

- When the signal (edge or level) specified in the request level setting register (ELVR) is detected on a DTP/external interrupt pin, the corresponding interrupt request flag bit in the DTP/external interrupt cause register (EIRR: ER7 to ER0) is set to "1".
  - If the interrupt request enable bit in the DTP/external interrupt enable register is set to enable (ENIR: EN7 to EN0 = 1) when the corresponding interrupt request flag bit is set to "1", an interrupt request is issued to the interrupt controller.
  - The interrupt controller checks whether the interrupt has a higher priority than any other interrupt request and, if so, generates an interrupt request.
  - CPU compares the interrupt level mask register (PS:ILM) and the interrupt request level (ICR:IL) for the processor status (PS), and if the interrupt request level is higher than ILM, and if the interrupt request enable bit is set to enabled (PS:CCR:I=1), then the interrupt processing is carried out after the current instruction has been finished to branch to the interrupt processing.
  - The interrupt handler must set the corresponding DTP/external interrupt request flag bit to "0" to clear the DTP/external interrupt request.
- 

#### Notes :

- An ER bit is set to "1" if a DTP/external interrupt start cause is generated, regardless of the state of the corresponding EN bit.
  - When the interrupt processing is activated, the ER bit that caused the routine to be activated must be cleared. If the ER bit is kept at "1", control cannot return from the interrupt. Only clear the flag bit that caused the interrupt; do not clear the other bits without reason.
-

## **17.5.2 DTP Function**

---

**The DTP/external interrupt circuit has a DTP function that detects a signal supplied to a DTP/external interrupt pin from an external peripheral and activates the extended intelligent I/O service.**

---

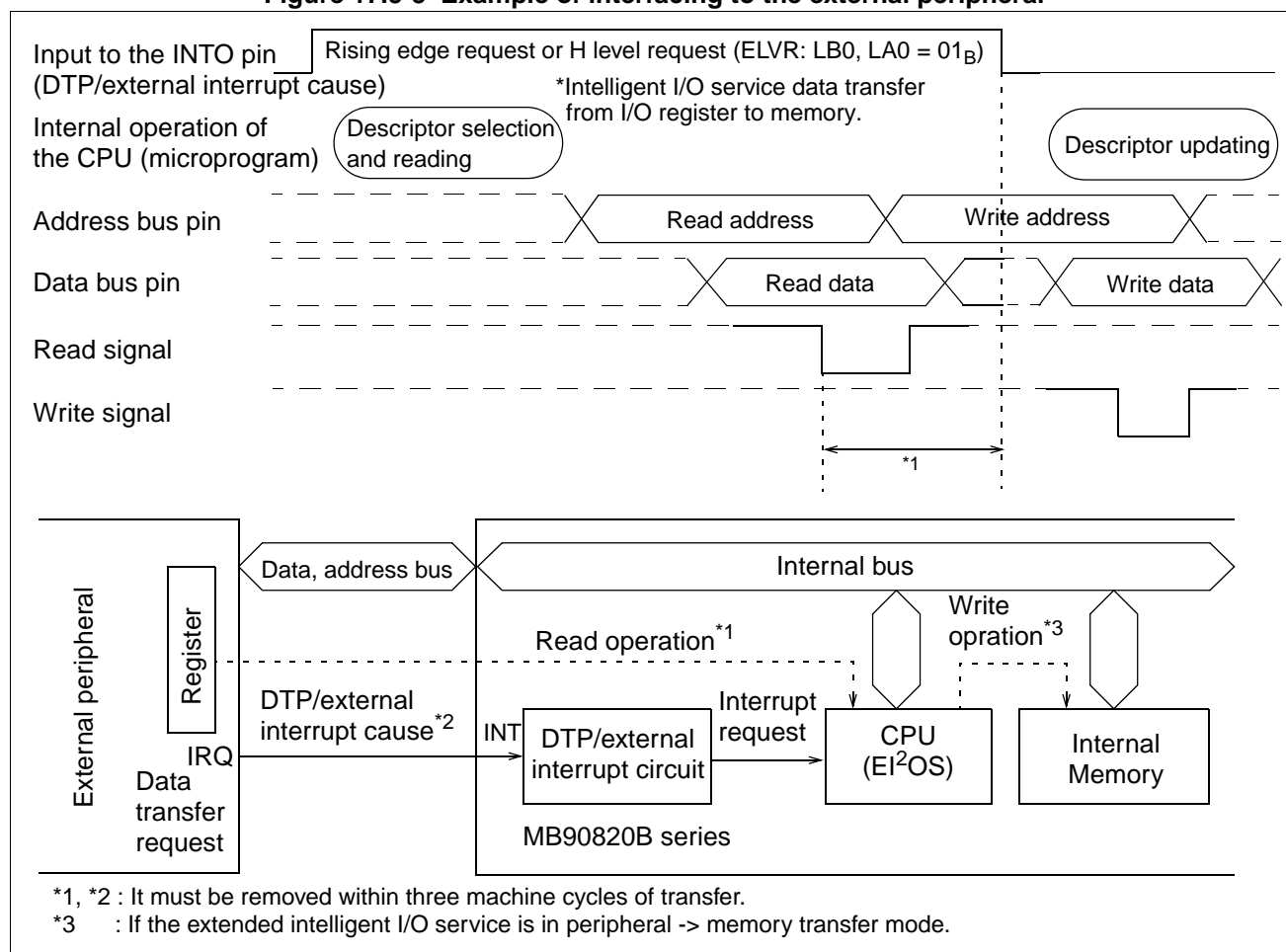
### **■ Operation of the DTP Function**

The DTP function detects a data transfer request signal from an external peripheral to automatically transfer data between memory and the peripheral.

The extended intelligent I/O service (EI<sup>2</sup>OS) is activated by the external interrupt function using level detection. The operation of the DTP function is the same as that of the external interrupt function up to the point that the CPU accepts an interrupt request. If the operation of EI<sup>2</sup>OS is enabled (ICR:ISE = 1), EI<sup>2</sup>OS is activated to start data transfer when an interrupt request is accepted. When the transfer of one data unit ends, the descriptor is updated and the interrupt request flag bit is cleared to wait for the next request from the pin. When the entire transfer using EI<sup>2</sup>OS is completed, control is transferred to the interrupt processing routine.

The external peripheral must remove only the level of the data transfer request signal (DTP external interrupt cause) within three machine cycles of the first transfer.

Figure 17.5-3 Example of interfacing to the external peripheral



## 17.6 Usage Notes on the DTP/External Interrupt Circuit

Notes on using the DTP/external interrupt is given below.

### ■ Usage Notes on the DTP/External Interrupt Circuit

#### ● Conditions for external peripherals using the DTP function

To support the DTP function, external peripherals must be able to clear data transfer requests automatically in response to transfer operations. If a transfer request is within three machine cycles of the start for transfer, the DTP/external interrupt circuit interprets the request as another transfer request.

#### ● Input polarities of external interrupts

- If the request level setting register (ELVR) is set so that an edge is detected, the pulse width must be at least three machine cycles for the edge to be detected.
- If the register is set for level detection, and the level to be detected as an interrupt cause is input, cause F/F in the DTP/external interrupt cause register (EIRR) is set to "1" to store the cause, as shown in Figure 17.6-1. Even if the cause is retained and the DTP/external interrupt request is removed, the request to the interrupt controller remains active provided the output of interrupt requests is enabled (ENIR:EN=1). Thus, to cancel the request to the interrupt controller, clear the external interrupt request flag bit (ENIR:ER) and cause F/F, as shown in Figure 17.6-2.

Figure 17.6-1 Clearing the cause retention circuit when a level is specified

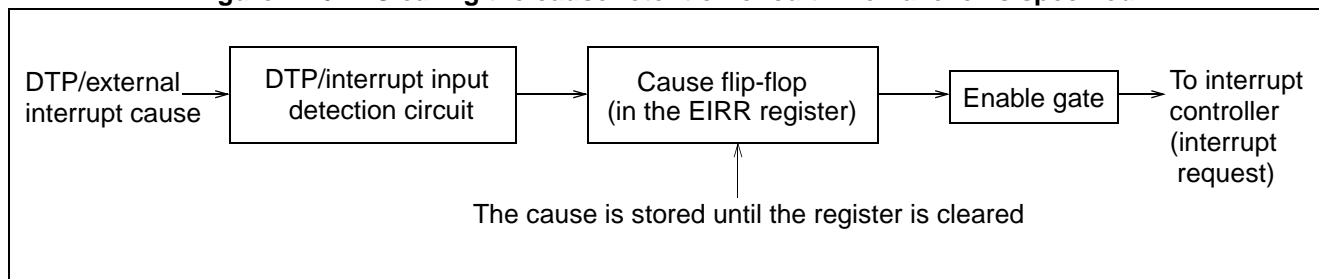
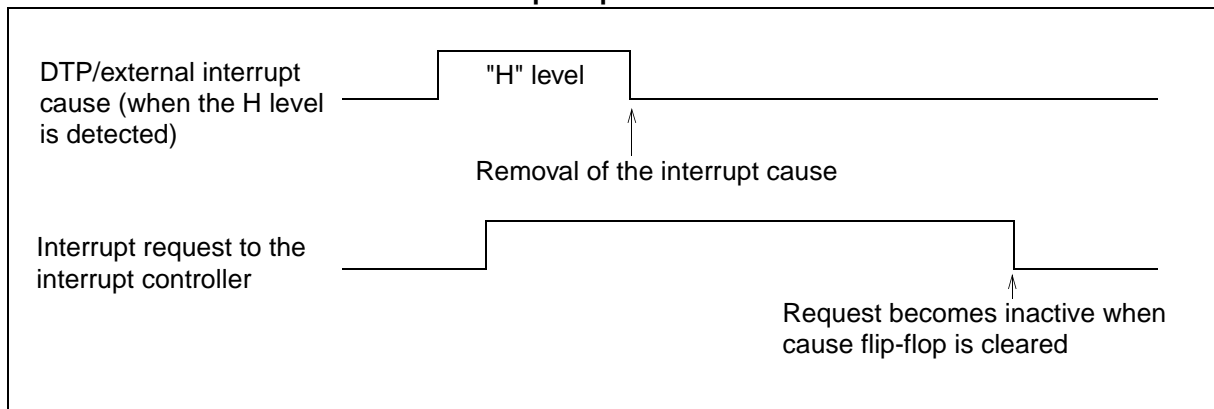


Figure 17.6-2 DTP/external interrupt cause and interrupt request when the output of interrupt requests is enabled



**● Notes about interrupts**

When the external interrupt function is used, control cannot return from the interrupt processing if the DTP/external interrupt request flag bit is "1" (EIRR:ER) and the output of DTP/external interrupt requests is enabled (ENIR:EN=1). In the interrupt processing routine, the DTP/external external interrupt request flag bit must be set to "0". (EIRR:ER). For level detection in the request level setting register, the DTP/external interrupt request flag bit is set again as soon as it is cleared (EIRR:ER=0) if the level assumed as an interrupt cause continues to be input. Either disable the output of DTP/external interrupt requests (ENIR:EN=0) or remove the interrupt cause, if required.





# **CHAPTER 18**

---

## ***8/10-Bit A/D Converter***

**This chapter explains the function and operation of the 8/10-bit A/D converter.**

- 18.1 Overview of 8/10-Bit A/D Converter
- 18.2 Block Diagram of 8/10-Bit A/D Converter
- 18.3 Configuration of 8/10-Bit A/D Converter
- 18.4 Interrupt of 8/10-Bit A/D Converter
- 18.5 Operation of 8/10-Bit A/D Converter
- 18.6 Precautions for Using the 8/10-Bit A/D Converter

## **18.1 Overview of 8/10-Bit A/D Converter**

---

The 8/10-bit A/D converter converts analog input voltages to 8-bit or 10-bit digital values by means of the RC sequential compare conversion.

- The input signal is selected from up to 16 channels analog input pins.
  - The activation trigger can be selected from software trigger, internal timer output, or external trigger.
- 

### **■ Features of 8/10-bit A/D converter**

The 8/10-bit A/D converter converts analog input voltages to 8-bit or 10-bit digital values (A/D conversion).

The 8/10-bit A/D converter has the following functions:

- The A/D conversion time is  $1.9\mu\text{s} \times \text{minimum per channel including the sampling time}$ .
- The sampling time is  $0.5\mu\text{s} \times \text{minimum per channel}$ .
- The conversion method is RC sequential compare conversion with sample and hold circuit.
- The resolution can be set to 8-bit or 10-bit.
- Up to 16 channels can be used for the analog input pin.
- Interrupt requests can be generated by storing the A/D conversion result in the A/D data register.
- The interrupt request can start EI<sup>2</sup>OS.
- The activation can be selected from software or internal timer output (16-bit reload timer 1, 16-bit free-run timer zero detection, or compare clear).

\*: When operating with 24-MHz machine clock frequency and  $AV_{CC} \geq 4.5\text{V}$

**■ Conversion Mode of 8/10-bit A/D Converter**

The 8/10-Bit A/D converter has the following conversion modes.

**Table 18.1-1 Conversion Mode of 8/10-Bit A/D Converter**

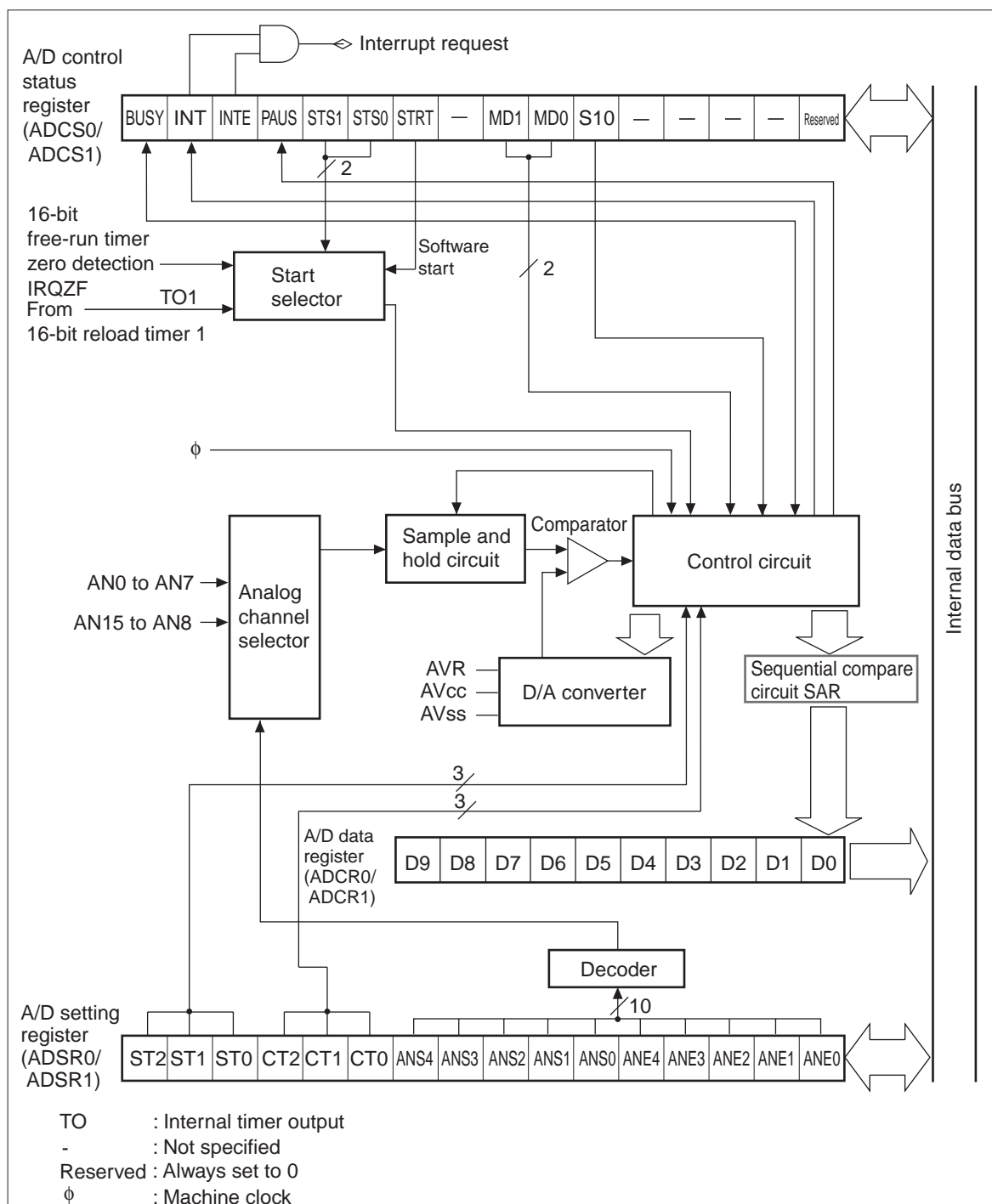
Conversion mode	Description
Single mode	A/D conversion is done from the start channel to the end channel sequential and stopped after the conversion of the end channel.
Continuous mode	A/D conversion is done from the start channel to the end channel sequential and returned to the start channel after the conversion of the end channel. Then A/D conversion is repeated.
Stop mode	A/D conversion is done for each one channel and stopped after the conversion of the channel. A/D conversion is returned to the start channel after the conversion of the end channel. Then the process of conversion and stop is repeated.

## 18.2 Block Diagram of 8/10-Bit A/D Converter

The 8/10-bit A/D converter is configured with the following blocks.

### ■ Block diagram of 8/10-bit A/D converter

Figure 18.2-1 Block Diagram of 8/10-Bit A/D Converter



## MB90820B Series

### ● Pins detail in the block diagram

Table 17.2-1 shows the actual pin name and the interrupt request number of the 8/10-bit A/D converter.

**Table 18.2-1 Pin Name and Interrupt Request Number in Block Diagram**

Pin name / interrupt request number in the block diagram		Actual pin name / interrupt request number
TO1	Internal timer output	16-bit reload timer 1 output
IRQZF	Internal timer output	16-bit free-run timer zero detection or compare clear
AN0 to AN7	Analog input pins ch.0 to ch.7	P60/AN0 to P67/AN7
AN8 to AN15	Analog input pins ch.8 to ch.15	P70/AN8 to P77/AN15
AVR	Vref input pin	AVR
AV <sub>CC</sub>	V <sub>CC</sub> input pin	AV <sub>CC</sub>
AV <sub>SS</sub>	V <sub>SS</sub> input pin	AV <sub>SS</sub>
Interrupt request output	Interrupt request output	#29 (1D <sub>H</sub> )

### ● A/D control status register (ADCS)

The A/D control status register activates the A/D conversion with a software, selects the activation trigger of A/D conversion, selects the conversion mode, enables/disables the interrupt request, confirms/clears the interrupt request flag, suspends the A/D conversion operation, confirms the ongoing conversion status, and selects the resolution.

### ● Sequential compare circuit (SAR)

The sequential compare circuit performs the sequential comparison for each one bit and stores the result. The A/D conversion result in this circuit is cleared when the next A/D conversion starts.

### ● A/D data register (ADCR)

The A/D conversion result is stored in the sequential compare circuit for each bit, and stored in this A/D data register when the A/D conversion finishes with the defined conversion results. The A/D conversion results can be read by this register.

### ● A/D setting register (ADSR)

The A/D setting register sets the start channel and the end channel of A/D conversion, the compare time of A/D conversion, and the sampling time of A/D conversion.

### ● Activation selector

The activation selector selects the activation trigger of A/D conversion. The activation trigger can be set from the internal timer output or the external pin input.

● **Decoder**

The decoder selects the analog input pins for A/D conversion from the A/D conversion start channel bit (ADSR: ANS3 to ANS0) and the A/D conversion end channel bit (ADSR: ANE3 to ANE0) selected by the A/D setting register.

● **Analog channel selector**

The analog channel selector selects the pins for A/D conversion from the 16 channel analog input pins according to the signal from decoder.

● **Sample and hold circuit**

The sample and hold circuit holds the input voltage selected by the analog channel selector. Hold of the input voltage just after starting A/D conversion enables the conversion without an effect of input voltage variation.

● **D/A converter**

The D/A converter generates the reference voltage to compare it to the input voltage held in the sample and hold circuit.

● **Comparator**

The comparator compares the input voltage held in the sample and hold circuit with the output voltage from D/A converter to determine the large/small.

● **Control circuit**

The control circuit defines the A/D conversion value with the large/small signal from the comparator, and stores the conversion result data in the A/D data register after the definition of the conversion result. An interrupt occurs if the interrupt request is enabled.

## MB90820B Series

### 18.3 Configuration of 8/10-Bit A/D Converter

This section shows pins, registers, and interrupt factors of the A/D converter.

#### ■ Pins of 8/10-bit A/D converter

Pins of the 8/10-bit A/D converter are also used as general purpose I/O ports. Table 17.3-1 shows the pin functions and the setting for 8/10-bit A/D converter.

**Table 18.3-1 8/10-Bit A/D Converter Pins**

Channel	Pin name	Pin function	Setting for 8/10-bit A/D converter
Channel 0	P60/AN0	General purpose I/O port / Analog input	Analog signal input enabled (ADER0: set the bits corresponding to ADE7 to ADE0 to "1")
Channel 1	P61/AN1		
Channel 2	P62/AN2		
Channel 3	P63/AN3		
Channel 4	P64/AN4		
Channel 5	P65/AN5		
Channel 6	P66/AN6		
Channel 7	P67/AN7		
Channel 8	P70/DA0/AN8	General purpose I/O port / Analog input / D/A converter output	Analog signal input enabled (ADER1: set the bits corresponding to ADE15 to ADE8 to "1")
Channel 9	P71/DA1/AN9		
Channel 10	P72/SIN1/AN10	General purpose I/O port / Analog input / UART1 I/O	
Channel 11	P73/SOT1/AN11		
Channel 12	P74/SCK1/AN12		
Channel 13	P75/FRCK/AN13	General purpose I/O port / Analog input / Free-run timer clock input	
Channel 14	P76/IN0/AN14	General purpose I/O port / Analog input / External interrupt input	
Channel 15	P77/IN1/AN15		



■ Registers of the 8/10-bit A/D converter and their initial value

**Figure 18.3-1 Registers of the 8/10-Bit A/D Converter and Their Initial Value**

A/D control status registers high order ADCS1								
15	14	13	12	11	10	9	8	Initial value
BUSY	INT	INTE	PAUS	STS1	STS0	STRT	—	0000000XB
R/W	R/W	R/W	R/W	R/W	R/W	W	—	
A/D control status registers low order ADCS0								
7	6	5	4	3	2	1	0	Initial value
MD1	MD0	S10	—	—	—	—	Reserved	000XXXX0B
R/W	R/W	R/W	—	—	—	—	R/W	
Data registers high order ADCR1								
15	14	13	12	11	10	9	8	Initial value
—	—	—	—	—	—	D9	D8	XXXXXXXXXB
—	—	—	—	—	—	R	R	
Data registers low order ADCR0								
7	6	5	4	3	2	1	0	Initial value
D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXXXB
R	R	R	R	R	R	R	R	
A/D setting registers high order ADSR1								
15	14	13	12	11	10	9	8	Initial value
ST2	ST1	ST0	CT2	CT1	CT0	Reserved	ANS3	00000000B
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
A/D setting registers low order ADSR0								
7	6	5	4	3	2	1	0	Initial value
ANS2	ANS1	ANS0	Reserved	ANE3	ANE2	ANE1	ANE0	00000000B
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Read and write  
 R : Read only  
 W : Write only  
 — : Undefined bit  
 X : Undefined

## MB90820B Series

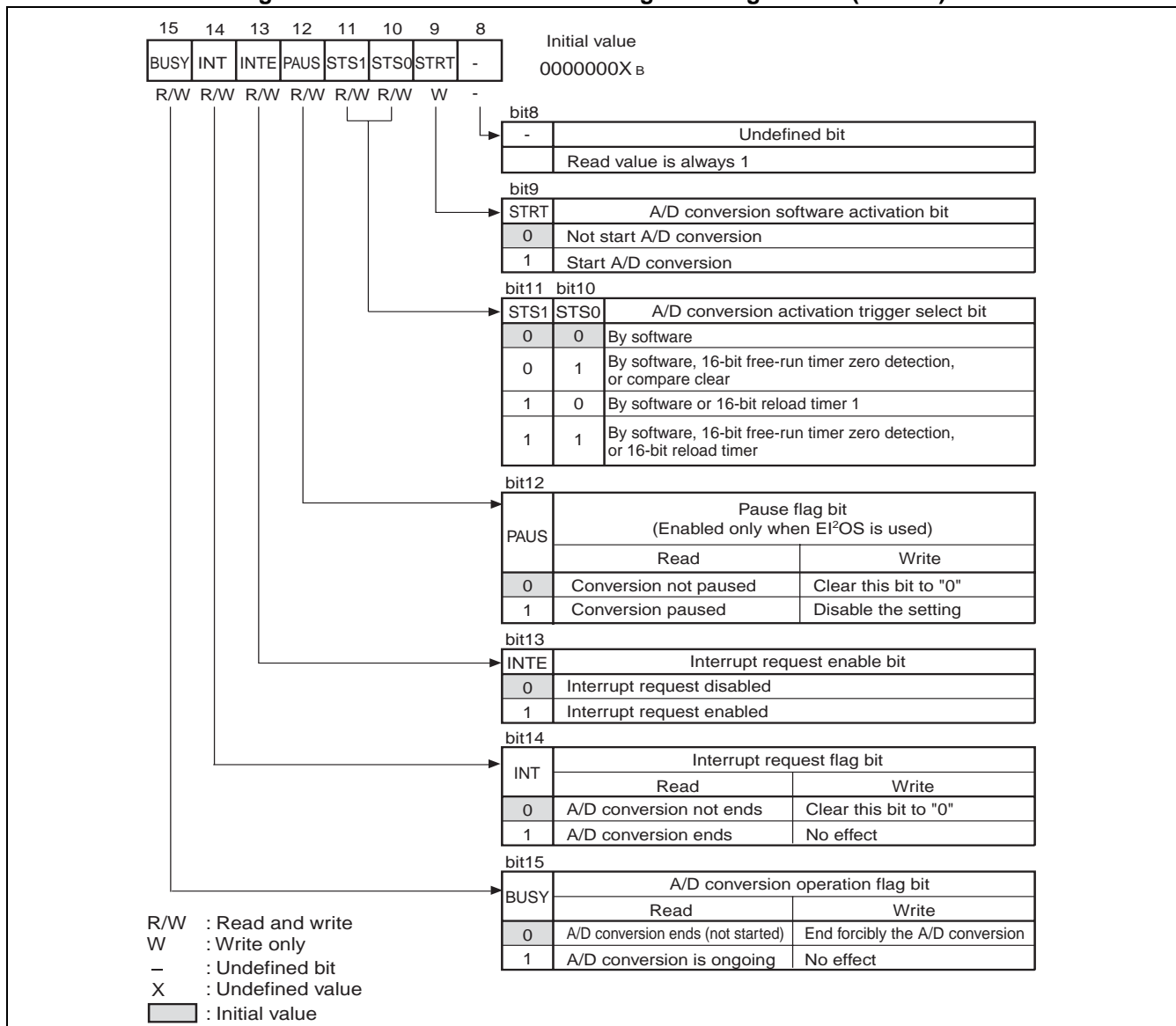
### 18.3.1 A/D Control Status Registers High Order (ADCS1)

The following functions can be set by the A/D control status registers high order (ADCS1).

- Activation of the A/D conversion with software
- Selection of the activation trigger for the A/D conversion function
- Enabling and disabling the interrupt request by storing the A/D conversion results in the A/D data register
- Confirmation and clearing the interrupt request by storing the A/D conversion results in the A/D data register
- Suspend of the A/D conversion and confirmation of ongoing conversion status

#### ■ A/D control status registers high order (ADCS1)

Figure 18.3-2 A/D Control Status Registers High Order (ADCS1)



**Table 18.3-2 Function of A/D Control Status Register High Order (ADCS1) (1 / 3)**

Bit name		Function
bit 15	BUSY:A/D conversion operation flag bit	<p>The 8/10-bit A/D conversion stops forcibly. When read, this bit indicates an operation or a stop of the 8/10-bit A/D converter. <b>Set to "0"</b>: 8/10-bit A/D converter stops forcibly. <b>Set to "1"</b>: No effect <b>When read</b>: "1" is read when the 8/10-bit A/D converter is ongoing, "0" when stopped. "1" is read in "stop status" in the stop mode.</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>"1" is read from this bit when an RMW instruction is used.</li> <li>In the single mode, this bit is cleared when A/D conversion ends.</li> <li>In the continuous or stop mode, this bit is not cleared until writing "0" to this bit to stop the A/D conversion.</li> <li>Do not perform the forced stop (BUSY = 0) and the starting of the A/D conversion concurrently (using software (STRT = 1), or timer).</li> </ul>
bit 14	INT:Interrupt request flag bit	<p>This bit indicates that an interrupt request is generated.</p> <ul style="list-style-type: none"> <li>After the A/D conversion, when the converted data is stored in A/D data register (ADCR), INT bit is set to "1".</li> <li>With the interrupt request enabled (INTE=1), if the interrupt request flag bit is set (INT=1), an interrupt request is generated.</li> <li>This bit is cleared when "0" is written. This bit is automatically cleared after EI<sup>2</sup>OS data transmission of A/D conversion result.</li> </ul> <p><b>Set to "0"</b>: Cleared <b>Set to "1"</b>: No effect</p> <p>Note:</p> <ul style="list-style-type: none"> <li>"1" is read from this bit when an RMW instruction is used.</li> </ul>
bit 13	INTE:Interrupt request enable bit	<p>This bit sets enabling/disabling of interrupts.</p> <ul style="list-style-type: none"> <li>With the interrupt request enabled (INTE=1), if the interrupt request flag bit is set (INT=1), an interrupt request is generated.</li> </ul> <p>Note:</p> <p>When using EI<sup>2</sup>OS to transmit the A/D converted result, set this bit to "1".</p>

Table 18.3-2 Function of A/D Control Status Register High Order (ADCS1) (2 / 3)

Bit name		Function
bit 12	PAUS: Pause flag bit	<p>PAUS bit indicates that the A/D conversion data protection function is set off. PAUS bit is valid only when the interrupt request output is set enabled (ADCS: INTE=1).</p> <p><b>A/D conversion data protection function is set off:</b> set to "1"</p> <p><b>When set to "0":</b> Cleared to "0"</p> <p><b>When set to "1":</b> Set to "1"</p> <ul style="list-style-type: none"> <li>When the interrupt request output is enabled (ADCS: INTE=1) and A/D conversion is performed, after one A/D conversion the interrupt request flag bit (ADCS: INT) is set and the interrupt request is generated simultaneously. If the next A/D conversion finishes with the interrupt request flag bit (ADCS:INT) not cleared, the A/D conversion pauses to prevent previous data from being overwritten (A/D conversion data protection function). PAUS bit is set to "1" when the A/D conversion pauses.</li> <li>When interrupt request flag bit (ADCS:INT) is cleared, 8/10-bit A/D converter returns from the pause state and resumes the A/D conversion.</li> <li>Interrupt request flag bit (ADCS:INT) is cleared by writing "0". When EI<sup>2</sup>OS is used to transmit the A/D converted result from A/D register, the interrupt request flag bit (ADCS:INT) is cleared by EI<sup>2</sup>OS.</li> </ul> <p>Notes:</p> <ul style="list-style-type: none"> <li>See "18.5.5 A/D Converted Data Protection Function1" for the A/D conversion data protection function.</li> <li>PAUS bit is not cleared automatically even after returning from the pause state. Write "0" to clear PAUS bit.</li> </ul>

**Table 18.3-2 Function of A/D Control Status Register High Order (ADCS1) (3 / 3)**

Bit name		Function
bit 11, bit 10	STS1, STS0: A/D conversion activation trigger select bits	<p>This bit selects the activation trigger of 8/10-bit A/D converter.</p> <ul style="list-style-type: none"> <li>• 00<sub>B</sub>: By software</li> <li>• 01<sub>B</sub>: By 16-bit free-run timer zero detection or compare clear / software</li> <li>• 10<sub>B</sub>: By 16-bit reload timer / software</li> <li>• 11<sub>B</sub>: 16 bit free-run timer zero detection or compare clear / 16-bit reload timer / software</li> </ul> <p>Notes:</p> <ul style="list-style-type: none"> <li>• With the 16-bit free-run timer zero detection or compare clear selected (01<sub>B</sub> or 11<sub>B</sub>), when the 16-bit free-run timer zero detection, A/D conversion starts.</li> <li>• With the 16-bit reload timer selected (10<sub>B</sub> or 11<sub>B</sub>), when the 16-bit reload timer 1 is "1", A/D conversion starts.</li> </ul> <p>Notes:</p> <ul style="list-style-type: none"> <li>• When multiple activation triggers are set, (other than that both STS1 and STS0 are "00<sub>B</sub>"), 8/10-bit A/D converter starts with the activation trigger generated first.</li> <li>• Change of the activation trigger setting should be performed when the peripheral functions which generate an activation trigger stops (trigger is inactive)</li> </ul>
bit 9	STRT: A/D conversion software activation bit	<p>This bit activates the 8/10-bit A/D converter by software.</p> <p><b>Set to "1":</b> 8/10-bit A/D converter is activated.</p> <ul style="list-style-type: none"> <li>• When the A/D conversion pauses in stop mode, the A/D conversion is resumed by writing "1" to STRT bit.</li> </ul> <p><b>Set to "0":</b> No effect</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>• "0" is read from this bit when an RMW instruction is used.</li> <li>• Not the written value, bit "1" is read from this bit when any instructions other than RMW are used.</li> <li>• Forced end of the 8/10-bit A/D converter (BUSY=0) and software activation (STRT=1) must not be performed simultaneously.</li> </ul>
bit 8	Undefined bit	<ul style="list-style-type: none"> <li>• Read: Always "1" is read.</li> <li>• Write: No effect</li> </ul>

## MB90820B Series

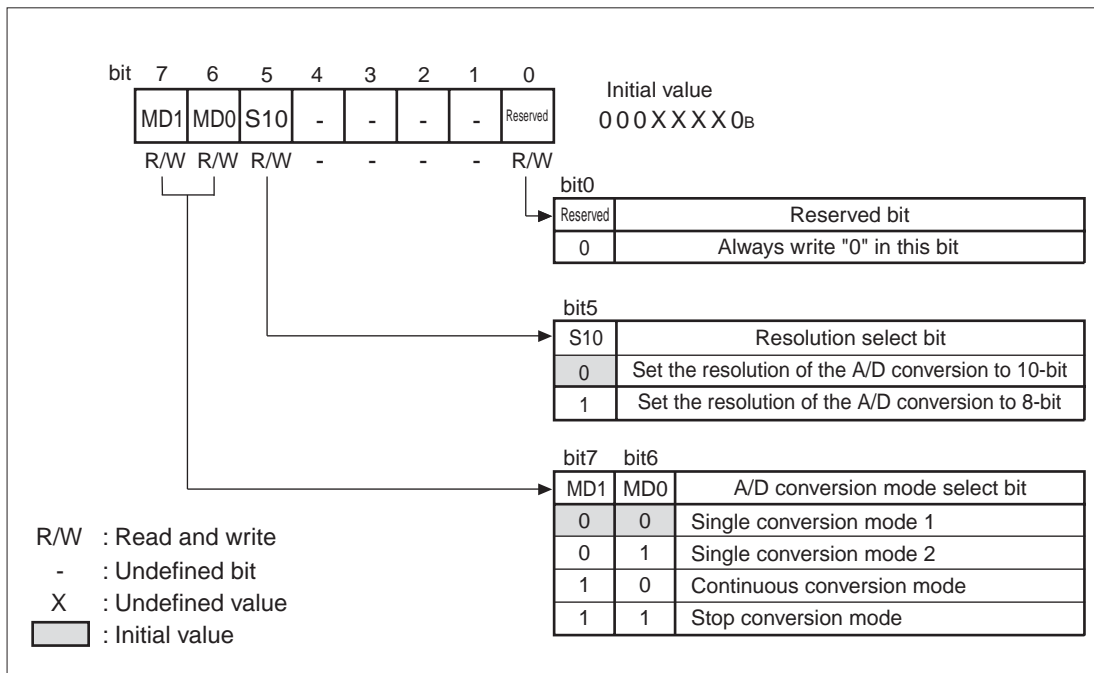
### 18.3.2 A/D Control Status Register Low Order (ADCS0)

The following functions can be set by the A/D control status registers low order (ADCS0).

- Select of the A/D conversion mode
- Select of start channel and end channel of the A/D conversion

#### ■ A/D control status registers low order (ADCS0)

Figure 18.3-3 A/D Control Status Register Low Order (ADCS0)



**Table 18.3-3 Function of A/D Control Status Register Low Order (ADCS0)**

Bit name		Function
bit 7, bit 6	MD1, MD0: A/D conversion mode selection bits	<p>These bits set the A/D conversion mode. For detailed usage of each mode, see 17.5 8/10-Bit A/D Converter Operation.</p> <p><b>Single conversion mode 1 and single conversion mode 2:</b></p> <ul style="list-style-type: none"> <li>A/D conversion is continuously performed for analog inputs from the start channel (ADSR0/1: ANS3 to ANS0) to the end channel (ADSR0/1: ANE3 to ANE0).</li> <li>After the A/D conversion of the end channel, the A/D conversion stops.</li> <li>For the difference between the single conversion 1 and the single conversion 2, see 17.5 8/10-Bit A/D Converter Operation.</li> </ul> <p><b>Continuous conversion mode:</b></p> <ul style="list-style-type: none"> <li>A/D conversion is continuously performed for analog inputs from the start channel (ADSR0/1: ANS3 to ANS0) to the end channel (ADSR0/1: ANE3 to ANE0).</li> <li>After the A/D conversion of the end channel, the A/D conversion continues repeatedly from analog input of the start channel.</li> </ul> <p><b>Stop conversion mode:</b></p> <ul style="list-style-type: none"> <li>A/D conversion is performed from the start channel (ADSR0/1: ANS3 to ANS0). After the A/D conversion for one channel, the A/D conversion stops. The next A/D conversion starts when the activation trigger is input during the A/D conversion stop state.</li> <li>After the A/D conversion of the end channel, the A/D conversion stops. When the activation trigger is input during the A/D conversion stop state, the A/D conversion returns to the start channel and continues.</li> </ul> <p>Note:</p> <ul style="list-style-type: none"> <li>Change of the conversion mode should be performed when the conversion is in the stop state before start.</li> </ul>
bit 5	S10: Resolution select bit	<p>This bit selects the resolution of 8/10-bit A/D converter.</p> <p><b>Set to "0":</b> The resolution of A/D conversion is set to 10 bits of A/D conversion data bit D9 to D0.</p> <p><b>Set to "1":</b> The resolution of A/D conversion is set to 8 bits of A/D conversion data bit D7 to D0.</p> <p>Note:</p> <p>Any change of S10 bit should be done in stopped state of A/D conversion, before the conversion starts. If S10 bit is changed after the A/D conversion starts, the converted result stored in the A/D conversion data bits (D9 to D0) become void.</p>

## MB90820B Series

### 18.3.3 A/D Data Register (ADCR0/ADCR1)

The A/D data register (ADCR0, ADCR1) is used to store digital values generated as a result of conversion. ADCR0 stores lower 8 bits, and ADCR1 stores most significant 2 bits of the conversion result. These registers' values are rewritten every time conversion ends. Normally, the last converted value is stored in these registers' bits.

#### ■ A/D data register (ADCR0/ADCR1)

Figure 18.3-4 A/D Data Register (ADCR0/ADCR1)

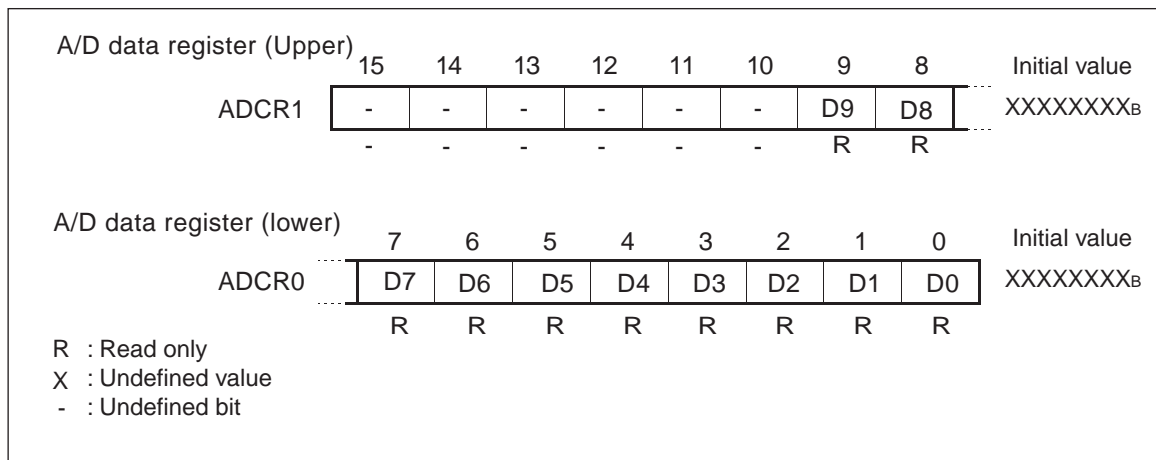


Table 18.3-4 Function of A/D Data Register (ADCR0/ADCR1)

Bit name		Function
bit 15 to bit 10	Undefined bits	Reading these bits always reads "1".
bit 9 to bit 0	D9 to D0: A/D conversion data bits	<p>These bits store the A/D conversion result.</p> <p><b>When the 10-bit mode is established (S10 bit of ADCS0 is "0"):</b> Converted data is stored in 10 bits of D9 to D0.</p> <p><b>When the 8-bit mode is established (S10 bit of ADCS0 is "1"):</b> Converted data is stored in 8 bits of D7 to D0. In this case, reading D9 to D8 always returns "1".</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>Do not write in these registers.</li> <li>To read the converted-data in the A/D conversion data bits (D9 to D0), use a word instruction (MOVW).</li> </ul>



## 18.3.4 A/D Setting Register (ADSR0/ADSR1)

The A/D setting register (ADSR0/ADSR1) is used for:

- A/D conversion time (sampling time and compare time) setting
- Sampling channels (start channel and end channel) setting
- Current sampling channel indication

### ■ A/D setting register (ADSR0/ADSR1)

Figure 18.3-5 A/D Setting Register (ADSR0/ADSR1)

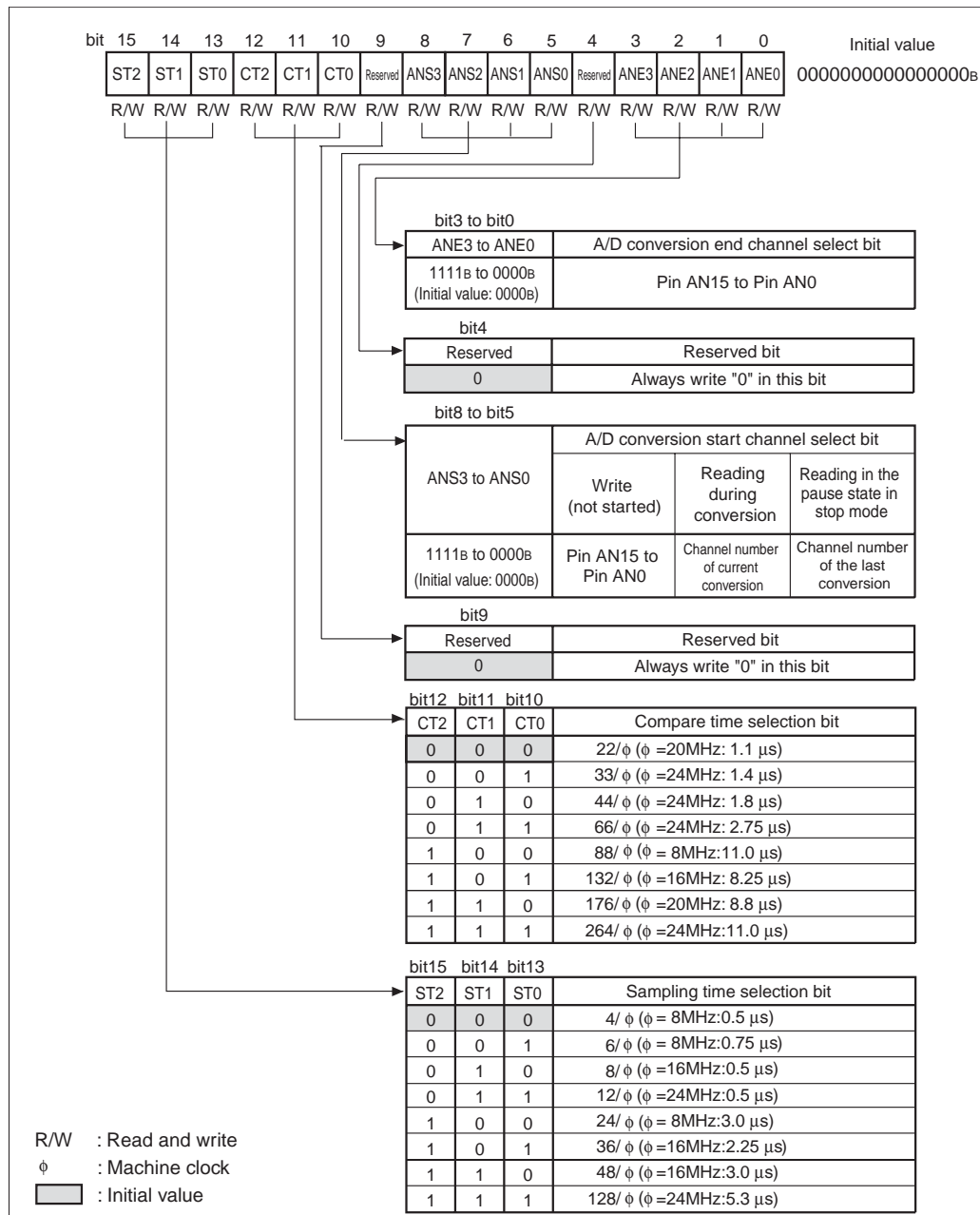


Table 18.3-5 Function of A/D Setting register (ADSR0/ADSR1) (1 / 2)

Bit name		Function
bit 15 to bit 13	ST2, ST1, ST0: Sampling time selection bits	<p>These bits set the sampling time of the A/D conversion.</p> <ul style="list-style-type: none"> <li>These bits set the time from start of the A/D conversion to sampling and holding the input analog voltage in the sample and hold circuit.</li> <li>See the table 17.3-6 for the setting of these bits.</li> </ul>
bit 12 to bit 10	CT2, CT1, CT0: Compare time selection bits	<p>These bits set the compare time of the A/D conversion.</p> <ul style="list-style-type: none"> <li>Set the time from start of the A/D conversion to storing the analog input into the data bits (D9 to D0).</li> <li>See the table 17.3-7 for the setting of these bits.</li> </ul>
bit 8 to bit 5	ANS3 to ANS0: A/D conversion start channel selection bits	<p>These bits set the start channel of the A/D conversion. When the A/D conversion is ongoing, read values of these bits indicate the channel number of the current conversion. When the A/D conversion stops or after the A/D conversion ends, these bits indicate the channel number of the last A/D conversion.</p> <p>Even if these bits are set to any values, read values indicate not the set value, but the channel number of the last A/D conversion until the A/D conversion starts. When these bits are reset, return to "0000<sub>B</sub>".</p> <p><b>Start channel &lt; end channel:</b> A/D conversion starts from the channel set in the A/D conversion start channel selection bits (ANS3 to ANS0), and ends at the channel set in the A/D conversion end channel selection bits (ANE3 to ANE 0).</p> <p><b>Start channel = end channel:</b> The A/D conversion is performed for only one channel set in the A/D conversion start channel selection bits (ANS3 to ANS0=NE3 to ANE 0).</p> <p><b>In the continuous conversion mode or the stop mode:</b> After the A/D conversion of the channel set in the A/D conversion end channel selection bits (ANE3 to ANE 0), the A/D conversion returns to the channel set in the A/D conversion start channel selection bits (ANS3 to ANS0).</p> <p><b>Reading these bits (in other than the stop mode):</b> The channel number (15 to 0) of the current A/D conversion is read.</p> <p><b>Reading these bits (in the stop mode):</b> The last channel number just before the stop is read.</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>The number of the start channel must not be larger than the number of the end channel.</li> <li>Do not set the A/D conversion start channel selection bits (ANS3 to ANS 0) during the A/D conversion.</li> <li>Writing in these bits should be performed with Word access. If with Byte writing or bit control, the A/D conversion may start from an unintended channel.</li> </ul>

**Table 18.3-5 Function of A/D Setting register (ADSR0/ADSR1) (2 / 2)**

Bit name		Function
bit 3 to bit 0	ANE3 to ANE0: A/D conversion end channel selection bits	<p>These bits set the end channel of the A/D conversion.</p> <p><b>Start channel &lt; end channel:</b> A/D conversion starts from the channel set in the A/D conversion start channel selection bits (ANS3 to ANS0), and ends at the channel set in the A/D conversion end channel selection bits (ANE3 to ANE 0).</p> <p><b>Start channel = end channel:</b> The A/D conversion is performed for only one channel set in the A/D conversion start channel selection bits (ANS3 to ANS0=ANE3 to ANE 0).</p> <p><b>In the continuous conversion mode or the stop mode:</b> After the A/D conversion of the channel set in the A/D conversion end channel selection bits (ANE3 to ANE 0), the A/D conversion returns to the channel set in the A/D conversion start channel selection bits (ANS3 to ANS0).</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>• The number of the start channel must not be larger than the number of the end channel.</li> <li>• Do not set the A/D conversion end channel selection bits (ANE3 to ANE0) during the A/D conversion.</li> <li>• After setting of the A/D conversion start channel selection bits (ANS3, ANS2, ANS1, ANS0), the sampling time selection bits (ST2, ST1, ST0), the compare time selection bits (CT2, CT1, CT0), and the A/D conversion end channel selection bits (ANE3, ANE2, ANE1, ANE0) should not be set using the read modify write instructions. Because reading of ANS3, ANS2, ANS1 and ANS0 are the last conversion channel until the A/D conversion starts, if ST2, ST1, ST0, and CT2, CT1, CT0, and ANE3, ANE2, ANE1, ANE0 are set using the read modify instructions after setting of ANS3, ANS2, ANS1, and ANS0, the values of ANS3, ANS2, ANS1, and ANS0 may be overwritten.</li> </ul>

## MB90820B Series

### ■ Sampling time setting (ST2 to ST0)

Table 18.3-6 Reference between Bits ST2 to ST0 and Sampling Time

ST2	ST1	ST0	Sampling time setting	Setting example ( $\phi$ : internal operating frequency)
0	0	0	4 machine cycles	$\phi = 8\text{MHz}$ : $0.5\mu\text{s}$
0	0	1	6 machine cycles	$\phi = 8\text{MHz}$ : $0.75\mu\text{s}$
0	1	0	8 machine cycles	$\phi = 16\text{MHz}$ : $0.5\mu\text{s}$
0	1	1	12 machine cycles	$\phi = 24\text{MHz}$ : $0.5\mu\text{s}$
1	0	0	24 machine cycles	$\phi = 8\text{MHz}$ : $3\mu\text{s}$
1	0	1	36 machine cycles	$\phi = 16\text{MHz}$ : $2.25\mu\text{s}$
1	1	0	48 machine cycles	$\phi = 16\text{MHz}$ : $3.0\mu\text{s}$
1	1	1	128 machine cycles	$\phi = 24\text{MHz}$ : $5.3\mu\text{s}$

The sampling time needs to be set according to the driving impedance  $R_{\text{ext}}$  for the analog input pin. When the following condition is not met, the A/D conversion precision is not assured:

- When  $R_{\text{ext}}$  is 1.5 k $\Omega$  or less:
  - $4.5\text{V} \leq AV_{\text{CC}} < 5.5\text{V}$ : Set the sampling time so as to be  $0.5\mu\text{s}$  or more.
  - $4.0\text{V} \leq AV_{\text{CC}} < 4.5\text{V}$ : Set the sampling time so as to be  $1.2\mu\text{s}$  or more.
- When  $R_{\text{ext}}$  is more than 1.5 k $\Omega$ : Set the sampling time  $T_{\text{samp}}$  to the value obtained using the following expression or more:

Flash memories

- $4.5\text{V} \leq AV_{\text{CC}} < 5.5\text{V}$ :  $T_{\text{samp}} = (2\text{k}\Omega + R_{\text{ext}}) \times 16\text{pF} \times 7$
- $4.0\text{V} \leq AV_{\text{CC}} < 4.5\text{V}$ :  $T_{\text{samp}} = (8.2\text{k}\Omega + R_{\text{ext}}) \times 16\text{pF} \times 7$

Mask ROMs

- $4.5\text{V} \leq AV_{\text{CC}} < 5.5\text{V}$ :  $T_{\text{samp}} = (2\text{k}\Omega + R_{\text{ext}}) \times 14.4\text{pF} \times 7$
- $4.0\text{V} \leq AV_{\text{CC}} < 4.5\text{V}$ :  $T_{\text{samp}} = (8.2\text{k}\Omega + R_{\text{ext}}) \times 14.4\text{pF} \times 7$

■ Compare time setting (CT2 to CT0)

**Table 18.3-7 Reference between Bits CT2 to CT0 and Compare Time**

CT2	CT1	CT0	Compare time setting	Setting example ( $\phi$ : internal operating frequency)
0	0	0	22 machine cycles	$\phi = 20\text{MHz}$ : $1.1\mu\text{s}$
0	0	1	33 machine cycles	$\phi = 24\text{MHz}$ : $1.4\mu\text{s}$
0	1	0	44 machine cycles	$\phi = 24\text{MHz}$ : $1.8\mu\text{s}$
0	1	1	66 machine cycles	$\phi = 24\text{MHz}$ : $2.75\mu\text{s}$
1	0	0	88 machine cycles	$\phi = 8\text{MHz}$ : $11.0\mu\text{s}$
1	0	1	132 machine cycles	$\phi = 16\text{MHz}$ : $8.25\mu\text{s}$
1	1	0	176 machine cycles	$\phi = 20\text{MHz}$ : $8.8\mu\text{s}$
1	1	1	264 machine cycles	$\phi = 24\text{MHz}$ : $11.0\mu\text{s}$

The compare time needs to be set according to the analog power  $AV_{CC}$ . When the following condition is not met, the A/D conversion precision is not assured:

- $4.5\text{V} \leq AV_{CC} < 5.5\text{V}$ : Set the compare time so as to be  $1.00\mu\text{s}$  or more.
- $4.0\text{V} \leq AV_{CC} < 4.5\text{V}$ : Set the compare time so as to be  $2.00\mu\text{s}$  or more.

## MB90820B Series

### 18.3.5 Analog Input Enable Resister (ADER0/ADER1)

This register enables or disables the analog input pins used in the 8/10-bit A/D converter.

#### ■ Analog input enable registers

Figure 18.3-6 Analog Input Enable Register (ADER0/ADER1)

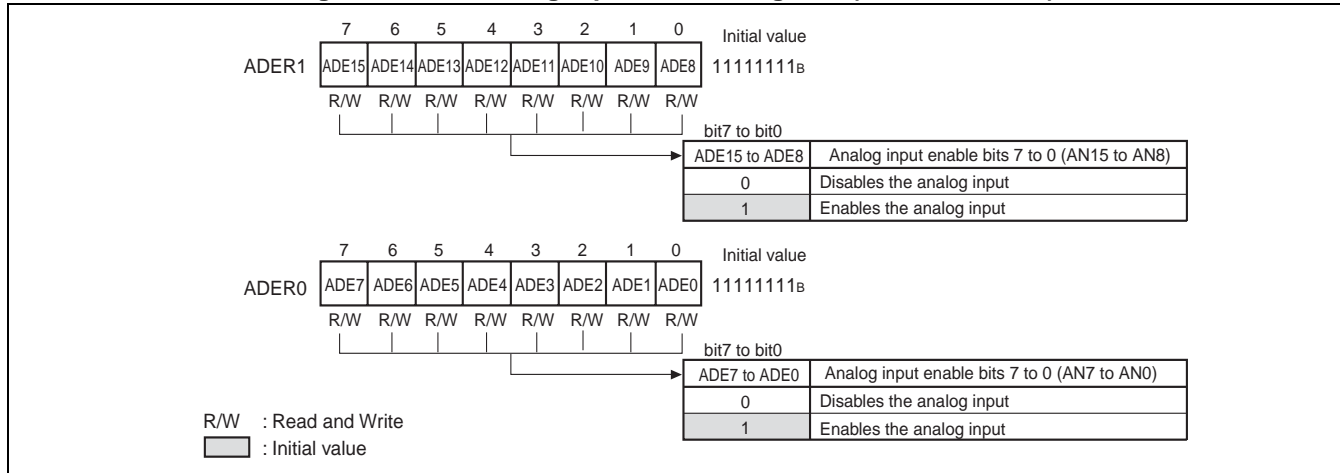


Table 18.3-8 Function of Port 6 Analog Input Enable Register (ADER1)

Bit name		Function
bit7 to bit0	ADE15 to ADE8: Analog input enable bits 7 to 0	These bits enable or disable the analog input of the A/D conversion analog input pins AN15 to AN8 on port 5. <b>When set to "0":</b> Disables the analog input. <b>When set to "1":</b> Enables the analog input.

Table 18.3-9 Function of Port 6 Analog Input Enable Register (ADER0)

Bit name		Function
bit7 to bit0	ADE7 to ADE0: Analog input enable bits 7 to 0	These bits enable or disable the analog input of the A/D conversion analog input pins AN7 to AN0 on port 6. <b>When set to "0":</b> Disables the analog input. <b>When set to "1":</b> Enables the analog input.

#### Notes:

- When used as an analog input for the A/D converter, the bits of corresponding analog input enable register (ADER0/ADER1) should be set to "1" to establish the analog input.
- For analog signal input, do not set the analog input pin so as to ADE<sub>x</sub>=0. Always set ADE<sub>x</sub> to "1".
- Every analog input pin is used as both general purpose I/O port and peripheral function I/O. Pins set to ADE<sub>x</sub>= 1 are forcibly established as analog input pins regardless of the port direction registers (DDR6/DDR7) and I/O settings of the peripheral functions. In this case, the pins cannot be used as others.

## **18.4 Interrupt of 8/10-Bit A/D Converter**

---

**For the 8/10-bit A/D converter, after the A/D conversion ends and the converted result is stored in the A/D data register (ADCR), an interrupt request is generated. In this case, the extended intelligent I/O service (EI<sup>2</sup>OS) can be used.**

---

### **■ Interrupt of A/D converter**

After the A/D conversion of the analog input voltage ends and the A/D converted result is stored in the A/D data register (ADCR), the interrupt request flag bit (ADCS: INT) of the A/D control status register is set to "1". If the interrupt request flag bit is set (ADCS: INT=1) with the interrupt request output enabled (ADCS: INTE=1), an interrupt request is generated.

### **■ Interrupt of 8/10-bit A/D converter and EI<sup>2</sup>OS**

---

#### **Reference:**

For the interrupt number, interrupt control register, and interrupt vector address, see Chapter 6 Interrupt.

---

### **■ EI<sup>2</sup>OS of 8/10-bit A/D converter**

For 8/10-bit A/D converter, the A/D converted result can be transmitted from the A/D data register (ADCR) to the memory. For the usage of EI<sup>2</sup>OS function, see "18.5.4 Conversion Using EI<sup>2</sup>OS" and "18.5.5 A/D Converted Data Protection Function".

## MB90820B Series

### 18.5 Operation of 8/10-Bit A/D Converter

A/D conversion of the 8/10-bit A/D converter includes the following conversion modes. These mode settings are established by setting of the A/D conversion mode selection bits (ADCS: MD1, MD0) of the A/D control status register.

- Single conversion mode
- Continuous conversion mode
- Stop conversion mode

#### ■ Single Conversion Mode (ADCS: MD1, MD0=00<sub>B</sub> or 01<sub>B</sub>)

- In this mode, when an activation trigger is input, analog inputs from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) are sequentially A/D converted.
- After the A/D conversion of the end channel, the A/D conversion stops.

#### Notes:

- In the single conversion mode 1 (ADCS: MD1, MD0=00<sub>B</sub>), do not input an activation trigger during the A/D conversion and the pause state\*. If do so, the 8/10-bit A/D converter may restart. In the single conversion mode 2 (ADCS: MD1, MD0=01<sub>B</sub>), the 8/10-bit A/D converter does not restarts even if an activation trigger is input during the A/D conversion and the pause state\*.
  - In both single conversion mode 1 and single conversion mode 2, a restart shall be performed according to "18.5.1 Single Conversion Mode".
- \*: The pause state means that the A/D conversion protection function sets off. For the detail, see "18.5.5 A/D Converted Data Protection Function".

#### ■ Continuous conversion mode (ADCS: MD1, MD0=10<sub>B</sub>)

- In this mode, when an activation trigger is input, analog inputs from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) are sequentially A/D converted.
- After the A/D conversion of the end channel, the A/D conversion returns to the start channel and repeats the conversion.

#### ■ Stop conversion mode (ADCS: MD1, MD0=11<sub>B</sub>)

- In this mode, when an activation trigger is input, A/D conversion of the start channel (ADSR: ANS3 to ANS0) starts. After the A/D conversion for one channel, the conversion operation stops. This is called as "Stop state". When an activation trigger is input in the stop state, the A/D conversion of the next channel starts.
- After the A/D conversion of the end channel, the A/D conversion stops. When an activation trigger is input in the stop state, the A/D conversion returns to the start channel and repeats the conversion.



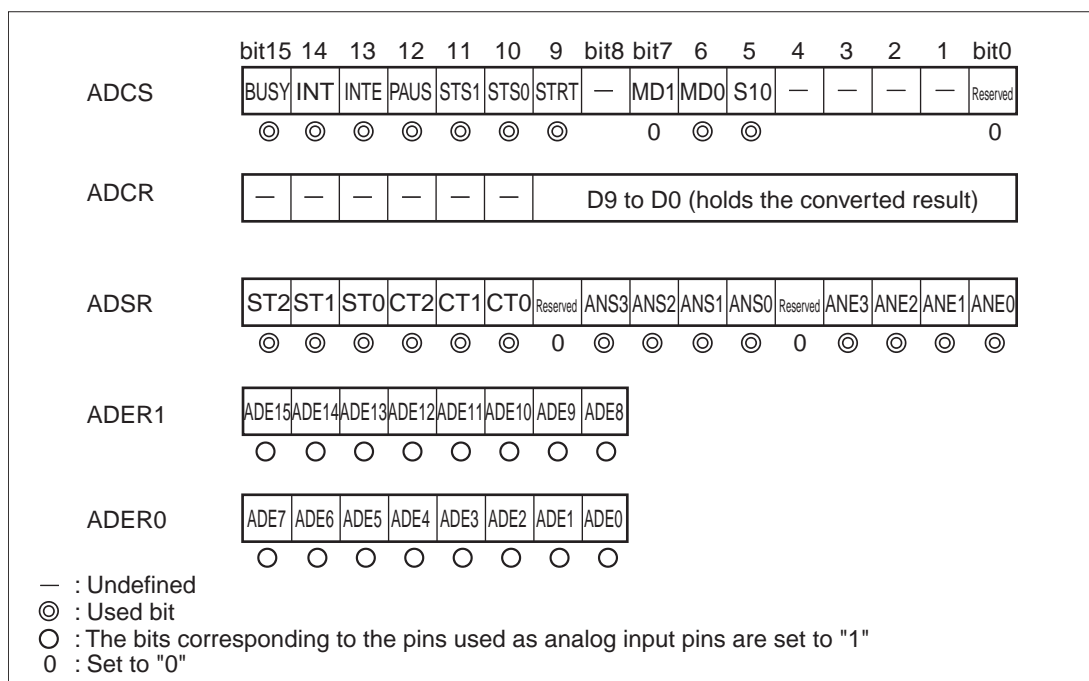
## 18.5.1 Single Conversion Mode

In this mode, the A/D conversion is performed from the start channel to the end channel sequentially. After the A/D conversion of the end channel, the A/D conversion stops.

### ■ Setting of Single Conversion Mode

To establish the single conversion mode of the 8/10-bit A/D converter, the setting as shown in Figure 18.5-1 is needed.

Figure 18.5-1 Setting of Single Conversion Mode



## MB90820B Series

### ■ Operation and Usage of the Single Conversion Mode

- When an activation trigger is input, the A/D conversion is performed sequentially from the channel set by the A/D conversion start channel selection bits (ANS3 to ANS0) to the channel set by the A/D conversion end channel selection bits (ANE3 to ANE0).
- After the A/D conversion of the channel set by the A/D conversion end channel selection bits (ANE3 to ANE0), the A/D conversion stops.
- To forcibly end the A/D conversion, write "0" in the A/D conversion ongoing operation flag bit (ADCS: BUSY).

#### [If the start channel and the end channel are same:]

- If the start channel and the end channel are set to same (ADSR: ANS3 to ANS0=ADSR: ANE3 to ANE0), the A/D conversion is performed once only for the start channel (= end channel), then the A/D conversion ends.

#### [Conversion sequence in the single conversion mode]

Examples of the conversion sequence in the single conversion mode are shown in Table 18.5-1 .

**Table 18.5-1 Conversion Sequence in Single Conversion Mode**

Start channel	End channel	Conversion sequence in the single conversion mode
Pin AN0 (ADSR: ANS=0000 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN0 -> AN1 -> AN2 -> AN3 -> End
Pin AN3 (ADSR: ANS=0011 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN3 -> End

#### [Restart]

To restart the A/D conversion during the A/D conversion or in the stop state, forcibly end the conversion and restart with the following method:

- 1) Clear the A/D conversion ongoing operation flag bit (ADCS:BUSY).
- 2) Clear the interrupt request flag bit (ADCS:INT).
- 3) Set the A/D conversion software activation bit (ADCS:STRT).

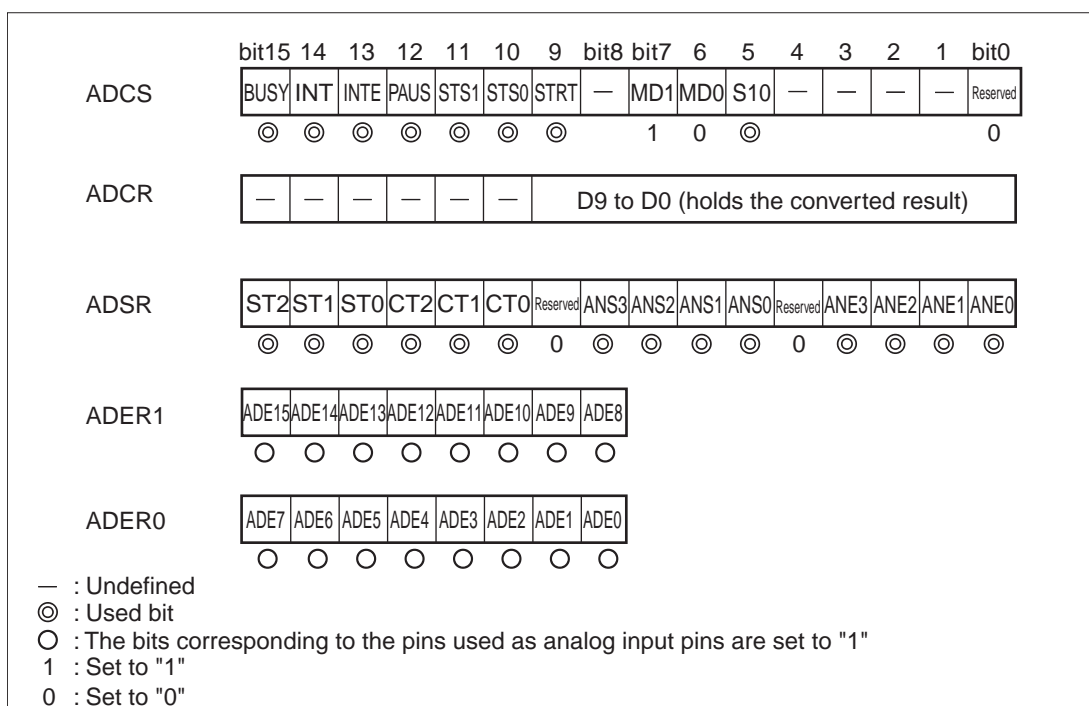
## 18.5.2 Continuous Conversion Mode

In this mode, the A/D conversion is performed from the start channel to the end channel sequentially. After the A/D conversion of the end channel, the A/D conversion returns to the start channel and repeats the conversion.

### ■ Setting of Continuous Conversion Mode

To establish the continuous conversion mode of the 8/10-bit A/D converter, the setting as shown in Figure 17.5-4 is needed.

Figure 18.5-2 Setting of Continuous Conversion Mode



### ■ Operation and Usage of the Continuous Conversion Mode

- When an activation trigger is input, the A/D conversion is performed sequentially from the channel set by the A/D conversion start channel selection bits (ANS3 to ANS0) to the channel set by the A/D conversion end channel selection bits (ANE3 to ANE0).
- After the A/D conversion of the channel set by the A/D conversion end channel selection bits (ANE3 to ANE0), the A/D conversion returns to the channel set by the A/D conversion start channel selection bits (ANS3 to ANS0) and repeats the conversion.
- To forcibly end the A/D conversion, write "0" in the A/D conversion ongoing operation flag bit (ADCS: BUSY).

[If the start channel and the end channel are same:]

- If the start channel and the end channel are set to same (ADSR: ANS3 to ANS0=ADSR: ANE3 to ANE0), the A/D conversion is performed once only for the start channel (= end channel) repeatedly.

**[Conversion sequence in the continuous conversion mode]**

Examples of the conversion sequence in the continuous conversion mode are shown in Table 18.5-2 .

**Table 18.5-2 Conversion Sequence in Continuous Conversion Mode**

Start channel	End channel	Conversion sequence in the continuous conversion mode
Pin AN0 (ADSR: ANS=0000 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN0 -> AN1 -> AN2 -> AN3 -> AN0 -> Repeat
Pin AN3 (ADSR: ANS=0011 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN3 -> AN3 -> Repeat

**[Restart]**

To restart the A/D conversion during the A/D conversion or in the stop state, forcibly end the conversion and restart with the following method:

- 1) Clear the A/D conversion ongoing operation flag bit (ADCS:BUSY).
- 2) Clear the interrupt request flag bit (ADCS:INT).
- 3) Set the A/D conversion software activation bit (ADCS:STRT).

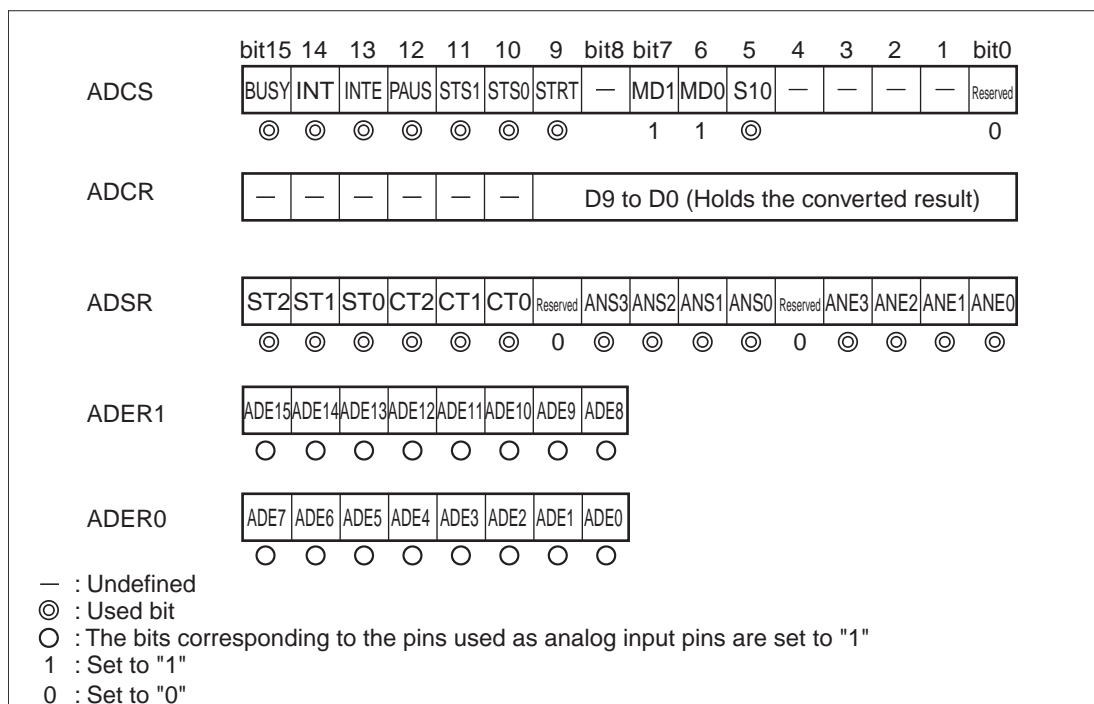
### 18.5.3 Stop Conversion Mode

In this mode, the A/D conversion starts and stops every one channel. When an activation trigger is input in the stop state after the end channel, the A/D conversion returns to the start channel and repeats the conversion.

#### ■ Setting of Stop Conversion Mode

To establish the stop conversion mode of the 8/10-bit A/D converter, the setting as shown in Figure 18.5-3 is needed.

Figure 18.5-3 Setting of Stop Conversion Mode



#### ■ Operation and Usage of the Stop Conversion Mode

- When an activation trigger is input, the A/D conversion is performed from the channel set by the A/D conversion start channel selection bits (ANS3 to ANS0). After the A/D conversion of one channel, the A/D conversion stops. When an activation trigger is input in the stop state, the A/D conversion of the next channel starts.
- After the A/D conversion of the channel set by the A/D conversion end channel selection bits (ANE3 to ANE0), the A/D conversion stops. When an activation trigger is input in the stop state, the A/D conversion returns to the channel set by the A/D conversion start channel selection bits (ANS3 to ANS0) and repeats the conversion.
- To forcibly end the A/D conversion, write "0" in the A/D conversion ongoing operation flag bit (ADCS: BUSY).

**[If the start channel and the end channel are same:]**

If the start channel and the end channel are set to same (ADSR: ANS3 to ANS0=ADSR: ANE3 to ANE0), the A/D conversion is performed only for the start channel (= end channel) repeatedly.

**[Conversion sequence in the stop conversion mode]**

Examples of the conversion sequence in the stop conversion mode are shown in Table 18.5-3 .

**Table 18.5-3 Conversion Sequence in Stop Conversion Mode**

Start channel	End channel	Conversion sequence in the single conversion mode
Pin AN0 (ADSR: ANS=0000 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN0 -> stop/start -> AN1 -> stop/start -> AN2 -> stop/start -> AN3 -> stop/start -> AN0 -> repeat
Pin AN3 (ADSR: ANS=0011 <sub>B</sub> )	Pin AN3 (ADSR: ANE=0011 <sub>B</sub> )	AN3 -> stop/start -> AN3 -> stop/start -> repeat

**[Restart]**

To restart the A/D conversion during the A/D conversion or in the stop state, forcibly end the conversion and restart with the following method:

- 1) Clear the A/D conversion ongoing operation flag bit (ADCS:BUSY).
- 2) Clear the interrupt request flag bit (ADCS:INT).
- 3) Set the A/D conversion software activation bit (ADCS:STRT).

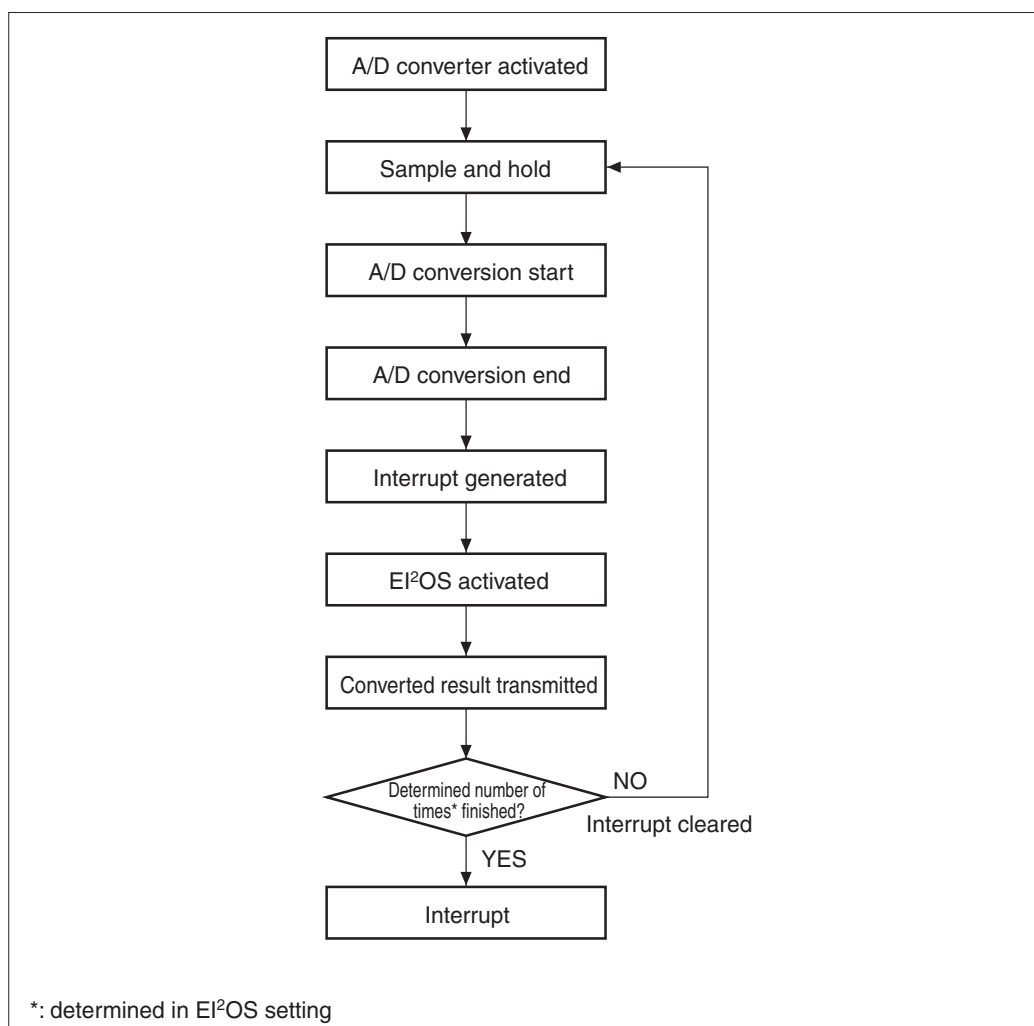
## 18.5.4 Conversion Using EI<sup>2</sup>OS

The 8/10-bit A/D converter can transmit the A/D converted result to the memory using EI<sup>2</sup>OS function.

### ■ Conversion Using EI<sup>2</sup>OS

The flow of conversion using EI<sup>2</sup>OS is shown in Figure 18.5-4 .

Figure 18.5-4 Flow of Conversion Using EI<sup>2</sup>OS



[Restriction on using EI<sup>2</sup>OS]

In the following cases, do not use EI<sup>2</sup>OS to transmit the A/D converted result data. Use the CPU data reading instruction to the A/D converted result data.

- 1) In the case that the A/D conversion is performed in the single conversion mode for sequential three or more channels.
- 2) In the case that the A/D conversion is performed in the continuous conversion mode.
- 3) In the case that the A/D conversion is performed in the stop conversion mode with the external trigger activation or 16-bit reload timer activation for sequential three or more channels (ADSR:ANE-ADSR:ANS $\geq$ 2).

[Software activation from the stop state in the stop conversion mode]

In the case that the A/D conversion is performed in the stop mode with the software activation for sequential three or more channels (ADSR:ANE-ADSR:ANS $\geq$ 2), the activation from the stop state should be the following steps.

- 1) Wait for the A/D conversion time passing from start of the A/D conversion.
- 2) Read the interrupt request flag bit (ADCS:INT). When the bit is "0", go to step 3). If "1", go to 2).
- 3) Write "1" in ADCS:STRT bit to software-activate the A/D conversion.



## 18.5.5 A/D Converted Data Protection Function

When the A/D conversion is performed with the interrupt request output enabled, the data protection function sets off.

### ■ The A/D converted data protection function of the 8/10-bit A/D converter

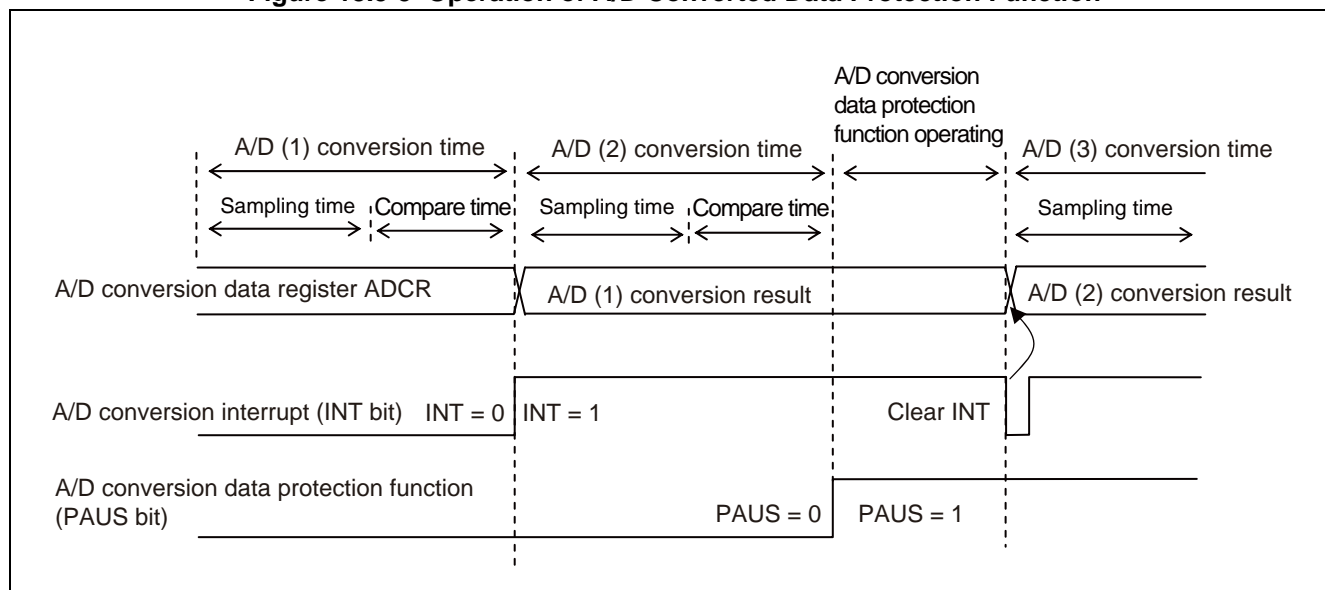
The A/D converted data protection function is designed for preventing from failing the A/D converted data. The 8/10-bit A/D converter includes one A/D data register (ADCR1/ADCR0) for storing the converted data and one sequential compare circuit for storing the ongoing A/D converted data.

During the A/D conversion, the 8/10-bit A/D converter stores the converted data in the sequential compare circuit for every one bit. After the A/D conversion, the A/D converted result is stored in the A/D data register.

The 8/10-bit A/D converter operates as the followings depending on use or no use of the A/D converted data protection function.

- To disable the data protection function, set the interrupt request enable bit (ADCS:INTE) to "0". In this case, during the sequential A/D conversion, the 8/10-bit A/D converter stores the converted result each time the A/D conversion ends. (Always the latest converted data is stored.)
- To enable the data protection function, set the interrupt request enable bit (ADCS:INTE) to "1". In this case, when the A/D conversion is performed sequentially, the interrupt request flag bit (ADCS:INT) is set to "1" after the first conversion ends. Then the next conversion is performed. If the conversion ends with INT=1, the 8/10-bit A/D converter pauses just before the converted result is transmitted from the sequential compare circuit to the A/D data register, therefore this prevents the converted data from overwritten. At this point, the pause flag bit (ADCS:PAUS) of the A/D control status register is set to "1". In this pause state, when the interrupt request flag bit (ADCS:INT) is cleared to zero, the data stored in the sequential compare circuit is transmitted to the A/D data register. (See Figure 18.5-5.)

Figure 18.5-5 Operation of A/D Converted Data Protection Function



## ● A/D converted data protection function for CPU reading

- After the analog input is A/D converted, when the A/D converted result is stored in the A/D data register (ADCR), the interrupt request flag (ADCS: INT) of the A/D control status register is set to "1".
- At the time the next A/D conversion ends, if the interrupt request flag bit (ADCS: INT) remains the value set at the end of the last A/D conversion with the interrupt request enabled (ADCS: INTE=1), the A/D conversion pauses just before the A/D data register is overwritten by the new data in order to protect the data.
- Because the interrupt request of the A/D control status register is enabled (ADCS: INTE=1), the interrupt request is generated when INT bit is set. When the INT bit is cleared, the pause state of the A/D conversion is released.
- During the sequential A/D conversion, the 8/10-bit A/D converter starts the next A/D conversion. In this case, the pause flag bit (ADCS: PAUS) is not cleared to zero automatically. To clear this, write zero in the bit.

---

### Notes:

- If the interrupt request output is set to disabled (ADCS:INTE=0) in the pause state, the A/D conversion may start and overwrite the A/D data register.
  - When multiple A/D conversions are performed sequentially, the data stored in the A/D data register must be read before the interrupt request flag bit (ADCS: INT) is cleared. If the interrupt request flag (ADCS: INT) is cleared before the data stored in the A/D data register is read in the pause state, the converted data stored first is overwritten by the next converted data.
- 

## ● A/D converted data protection function for transmitting the A/D converted result using EI<sup>2</sup>OS

After the A/D conversion, during transmitting the A/D converted result from the A/D data register to the memory using EI<sup>2</sup>OS, if the next A/D conversion ends, the A/D conversion pauses to protect data just before the A/D register is overwritten by the new data. When the A/D conversion stops, the pause flag bit of the A/D control status register (ADCS: PAUS) is set to "1".

After the A/D converted result is transmitted to the memory using EI<sup>2</sup>OS, the pause state of the A/D conversion is released. In the case of the sequential A/D conversion, the A/D conversion resumes. At this point, the pause flag bit (ADCS: PAUS) is not cleared to zero automatically. To clear this, write zero in the bit.

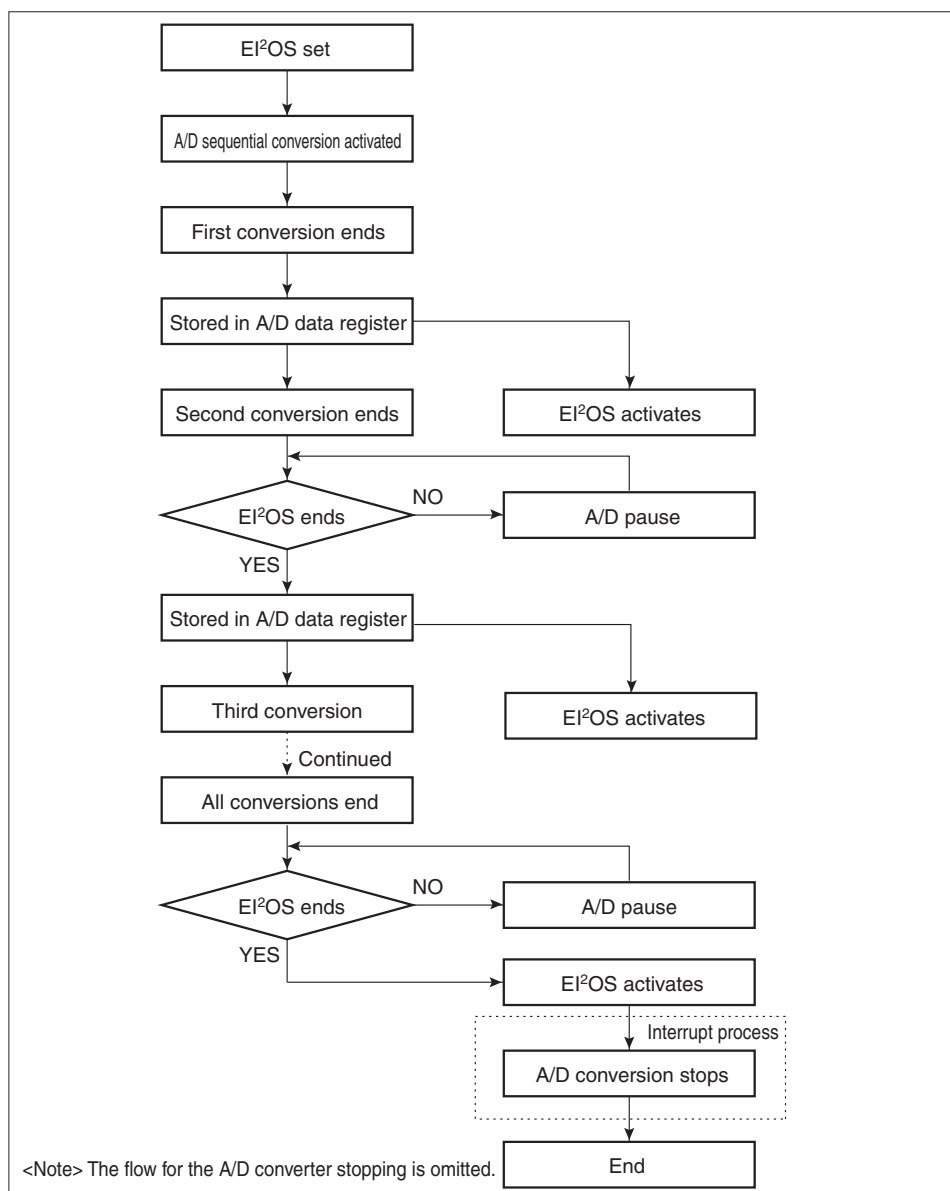
---

**Notes:**

- In case that the A/D converted result is transmitted to the memory using EI<sup>2</sup>OS, the interrupt request flag bit should not be cleared (ADCS: INT=0) by CPU. The data of the A/D data register in transmission may be overwritten.
  - In case that the A/D converted result is transmitted to the memory using EI<sup>2</sup>OS, the interrupt request output should not be disabled. If the interrupt request output is disabled (ADCS: INTE=0) in pause state, the A/D conversion starts and the data of the A/D data register in transmission may be overwritten.
  - In case that the A/D converted result is transmitted to the memory using EI<sup>2</sup>OS, do not restart. Restarting in the pause state of the A/D conversion may destruct the converted result.
- 

● Process flow of the A/D converted data protection function using EI<sup>2</sup>OS

The process flow diagram of the A/D converted data protection function using EI<sup>2</sup>OS is shown in Figure 18.5-6 .

Figure 18.5-6 Process Flow of A/D Converted Data Protection Function Using EI<sup>2</sup>OS

## **18.6 Precautions for Using the 8/10-Bit A/D Converter**

---

**Precautions for using the 8/10-bit A/D converter are as the followings:**

---

### **■ Precautions for using the 8/10-bit A/D converter**

- **Analog input pins**
  - An analog input pin is also used as the general I/O port of port 6 and 7. To use this pin as the analog input pin, set the analog input enable register (ADER0/ADER1) to switch the pin to the analog input pin.
  - To use this pin as the analog input pin, write "1" in the bit of the analog input enable register (ADER0/ADER1) corresponding to the used pin and set the analog input enabled.
  - If a middle level signal is input to the pin remaining as the general purpose I/O port, the input leak current flows to the gate. Always set the analog input enabled before using this pin as the analog input pin.
- **Precautions for activation with an internal timer or an external trigger.**
  - To set the A/D activation trigger selection bit (ADCS: STS1, STS0) so as to activate the 8/10-bit A/D converter using an internal timer output or an external trigger, set the level of the timer output or the external trigger to inactive ("H" for an external trigger). If the input level of the activation trigger is set active, the operation may start simultaneously with setting the A/D activation trigger selection bit (ADCS: STS1, STS0) of the A/D control status register.
- **Order for turning on the 8/10-bit A/D converter and applying an analog input**
  - The digital power source ( $V_{CC}$ ) must be turned on before turning on the 8/10-bit A/D converter and applying an analog input (AN0 to AN15).
  - The digital power source must be turned off after turning off the 8/10-bit A/D converter and applying an analog input.
  - AVR must be turned on/off so that  $AV_{CC}$  is not exceeded. (Turning on/off the analog power source and the digital power source simultaneously is not a problem.)
- **Power voltage of the 8/10-bit A/D converter**
  - To prevent a latchup, power source of the 8/10-bit A/D converter ( $AV_{CC}$ ) must not exceed voltage of the digital power source ( $V_{CC}$ ).

# ***CHAPTER 19***

---

## ***D/A CONVERTER***

**This chapter explains the functions and operation of the digital/analog (D/A) converter.**

- 19.1 Overview of D/A Converter
- 19.2 Block Diagram of D/A Converter
- 19.3 D/A Converter Pins
- 19.4 D/A Converter Registers

## 19.1 Overview of D/A Converter

The digital/analog (D/A) converter converts an 8-bit digital input into an analog output by using R-2R method. The D/A converter has two channels. Output control can be individually executed for each channel by using its D/A control register .

### ■ Function and Operation of the D/A Converter

This circuit is used to generate an analog output from an 8-bit digital input. By setting the enable bit in the D/A control register (DACR) to "1", it will enable the corresponding D/A output channel. Hence, setting this bit to "0" will disable that channel.

If D/A output is disabled, the analog switch inserted to the output of each D/A converter channel in series is turned off. In the D/A converter, the bit is cleared to "0" and the direct-current path is shut off. The above is also true in the stop mode.

The output voltage of the D/A converter ranges from 0 V to  $255/256 \times AV_{CC}$ . To change the output voltage range, adjust the  $AV_{CC}$  voltage externally.

The D/A converter output does not have the internal buffer amplifier. The analog switch (= 100  $\Omega$ ) is inserted to the output in series. To apply load to the output externally, estimate a sufficient stabilizing time.

Table 19.1-1 lists the theoretical values of output voltage for the D/A converter.

**Table 19.1-1 Theoretical values of output voltage for the D/A converter**

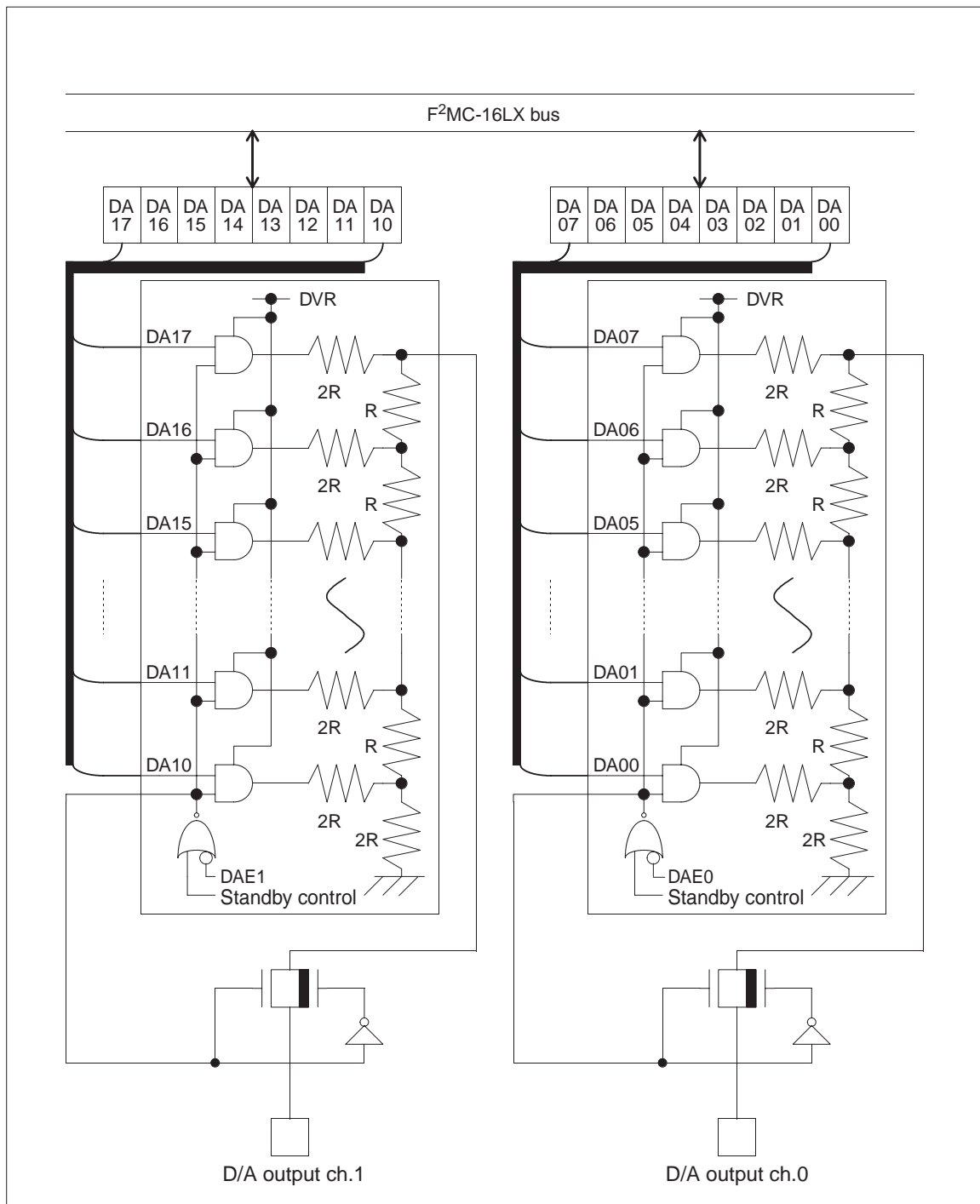
Value written to DA07 to DA00 and DA17 to DA10	Theoretical value of output voltage
00 <sub>H</sub>	$0/256 \times AV_{CC}$ (=0 V)
01 <sub>H</sub>	$1/256 \times AV_{CC}$
02 <sub>H</sub>	$2/256 \times AV_{CC}$
:	:
FD <sub>H</sub>	$253/256 \times AV_{CC}$
FE <sub>H</sub>	$254/256 \times AV_{CC}$
FF <sub>H</sub>	$255/256 \times AV_{CC}$

**MB90820B Series****19.2 Block Diagram of D/A Converter**

This section shows the block diagram of D/A converter

**■ D/A Converter Block Diagram**

Figure 19.2-1 Block diagram of D/A converter





19.3 D/A Converter Pins

This section describes the pins of the D/A converter and provides a pin block diagram.

D/A Converter Pins

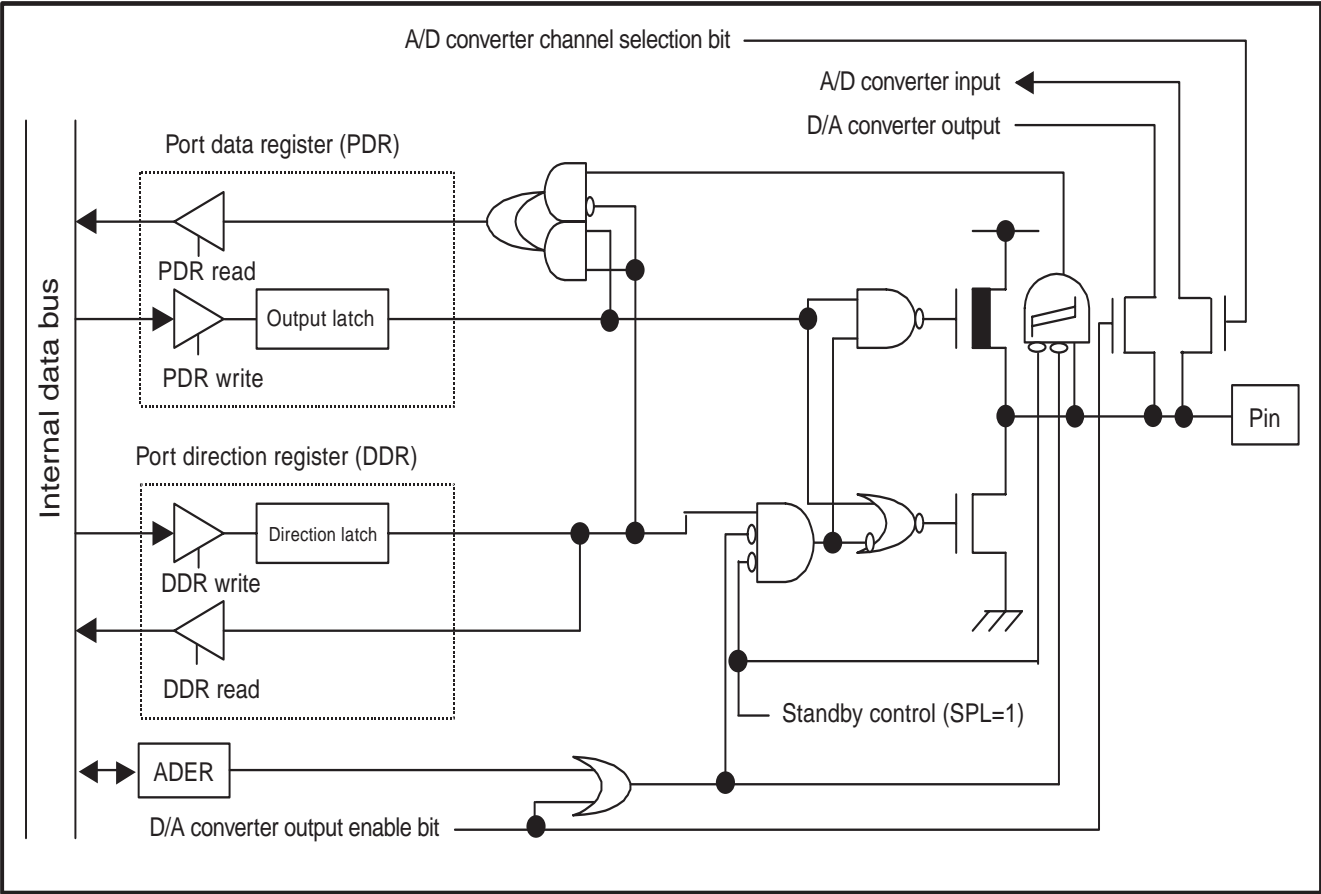
The pins of the D/A converter are shared with the general-purpose I/O ports. Table 19.3-1 lists the functions of the pins, I/O format, and settings required to use the D/A converter.

Table 19.3-1 D/A converter pins

Pin name	Pin function	I/O format	Pull-up option	Standby	Settings required for pins
P70/DA0/AN8	Port 7 input-output / Analog D/A converter pins	Analog/CMOS output / CMOS hysteresis input	Not provided	Provided	DACR0:DAE0=1
P71/DA1/AN9					DACR1:DAE1=1

Block Diagram of the D/A Converter Pins

Figure 19.3-1 Block diagram of the D/A converter pins.



**MB90820B Series****19.4 D/A Converter Registers**

The D/A converter has the following two types of registers :

- D/A converter registers (DAT0 and DAT1)
- D/A control registers (DACR0 and DACR1)

### ■ D/A Converter Registers

**Figure 19.4-1 D/A converter registers**

<b>D/A data register 1</b>										
	bit	15	14	13	12	11	10	9	8	
Address: 0000CD <sub>H</sub>		DA17	DA16	DA15	DA14	DA13	DA12	DA11	DA10	DAT1
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	

<b>D/A data register 0</b>										
	bit	7	6	5	4	3	2	1	0	
Address: 0000CC <sub>H</sub>		DA07	DA06	DA05	DA04	DA03	DA02	DA01	DA00	DAT0
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	

<b>D/A control register 1</b>										
	bit	15	14	13	12	11	10	9	8	
Address: 0000CF <sub>H</sub>		—	—	—	—	—	—	—	DAE1	DACR1
Read/write ⇒		—	—	—	—	—	—	—	R/W	
Initial value ⇒		X	X	X	X	X	X	X	0	

<b>D/A control register 0</b>										
	bit	7	6	5	4	3	2	1	0	
Address: 0000CE <sub>H</sub>		—	—	—	—	—	—	—	DAE0	DACR0
Read/write ⇒		—	—	—	—	—	—	—	R/W	
Initial value ⇒		X	X	X	X	X	X	X	0	

### 19.4.1 D/A Converter Register 1 (DAT1)

The D/A converter register 1 (DAT1) is used to set the digital input data for channel 1, which will be converted into an analog output.

■ D/A Converter Register 1 (DAT1)

Figure 19.4-2 D/A converter register 1 (DAT1)

D/A converter register 1								
Bit	15	14	13	12	11	10	9	8
Address: 0000CD <sub>H</sub>	DA17	DA16	DA15	DA14	DA13	DA12	DA11	DA10
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
DAT1								

R/W: Read and write  
X: Unknown

The D/A converter register 1 (DAT1) is used to set the digital input data for channel 1, which will be converted into an analog output.

## MB90820B Series

### 19.4.2 D/A Converter Register 0 (DAT0)

The D/A converter register 0 (DAT0) is used to set the digital input data for channel 0, which will be converted into an analog output.

#### ■ D/A Converter Register 0 (DAT0)

Figure 19.4-3 D/A converter register 0 (DAT0)

D/A converter register 0									
	Bit	7	6	5	4	3	2	1	0
Address: 0000CC <sub>H</sub>		DA07	DA06	DA05	DA04	DA03	DA02	DA01	DA00
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

R/W: Read and write  
X: Unknown

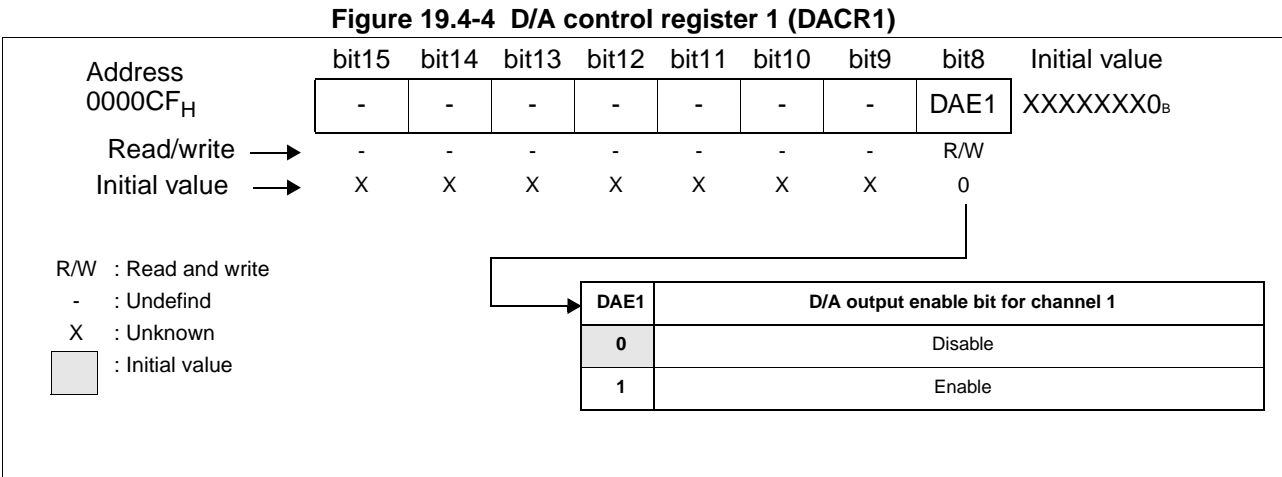
DAT0

The D/A converter register 0 (DAT0) is used to set the digital input data for channel 0, which will be converted into an analog output.

### 19.4.3 D/A Control Register 1 (DACR1)

The D/A control register 1 is used to keep the output enable signal for channel 1.

#### ■ D/A Control Register 1 (DACR1)



**Table 19.4-1 D/A control register 1 (DACR1)**

Bit name		Function
bit15 to bit 9	Undefined bits	<ul style="list-style-type: none"><li>The read value is undefined.</li><li>Writing to these bits have no effect on the operation.</li></ul>
bit 8	DAE1 : D/A output enable bit for channel 1	<ul style="list-style-type: none"><li>This bit is used to determine if D/A converter output is enabled or disabled. DAE1 is responsible for channel 1.</li><li>Writing "1" to this bit enables D/A output; similarly "0" disables D/A output.</li><li>This bit is initialized to "0" at reset.</li><li>This bit can be read and written.</li></ul>

## MB90820B Series

### 19.4.4 D/A Control Register 0 (DACR0)

The D/A control register 0 is used to keep the output enable signal for channel 0.

#### ■ D/A Control Register 0 (DACR0)

Figure 19.4-5 D/A control register 0 (DACR0)

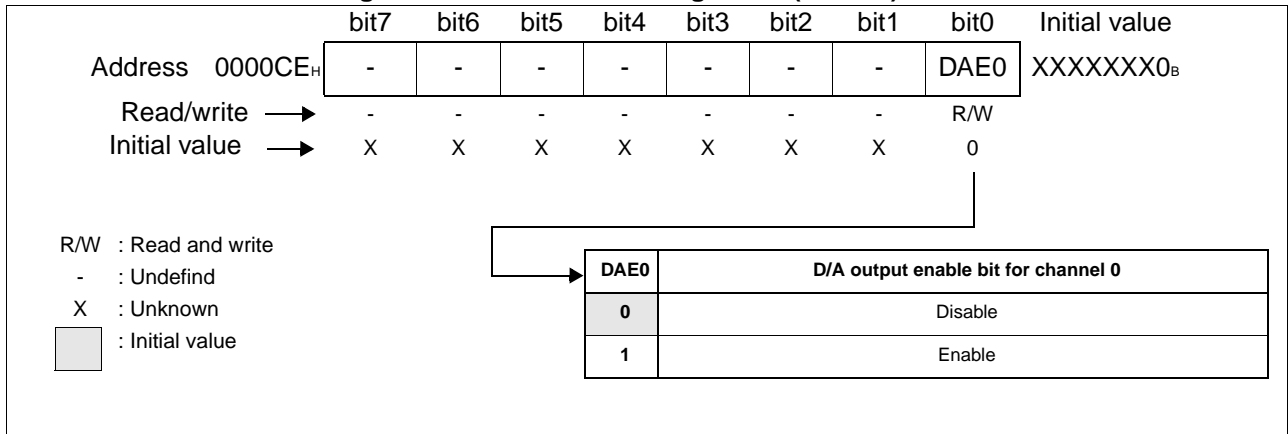


Table 19.4-2 D/A control register 0 (DACR0)

Bit name		Function
bit 7 to bit 1	Undefined bits	<ul style="list-style-type: none"> <li>The read value is undefined.</li> <li>Writing to these bits have no effect on the operation.</li> </ul>
bit 0	DAE0 : D/A output enable bit for channel 0	<ul style="list-style-type: none"> <li>This bit is used to determine if D/A converter output is enabled or disabled. DAE0 is responsible for channel 0.</li> <li>Writing "1" to this bit enables D/A output; similarly "0" disables D/A output.</li> <li>This bit is initialized to "0" at reset.</li> <li>This bit can be read and written.</li> </ul>



# **CHAPTER 20**

---

# **UART**

**This chapter explains the functions and operation of UART.**

- 20.1 Overview of UART
- 20.2 Block Diagram of UART
- 20.3 UART Pins
- 20.4 UART Registers
- 20.5 UART Interrupts
- 20.6 UART Baud Rates
- 20.7 Operation of UART
- 20.8 Usage Notes on UART



## 20.1 Overview of UART

UART is a general-purpose serial data communication interface for performing synchronous or asynchronous (start-stop synchronization) communication with external devices. The UART has a clock synchronous and bidirectional communication function (normal mode), additionally the master-slave communication function (multiprocessor mode) is only available for the master system.

### ■ UART Functions (x 2)

#### ● UART functions

UART is a general-purpose serial data communication interface for transmitting serial data to and receiving data from another CPU and external peripheral devices. It has the functions listed in Table 20.1-1.

**Table 20.1-1 UART functions**

	Function
Data buffer	Full-duplex, double buffering
Transfer mode	<ul style="list-style-type: none"> <li>• Clock synchronous</li> <li>• Clock asynchronous (start-stop synchronization)</li> </ul>
Baud rate	<ul style="list-style-type: none"> <li>• A dedicated baud rate generator is provided. Eight settings can be selected</li> <li>• An external clock can be input</li> <li>• Internal clock (From 16-bit reload timer 0 and 1, the clock of 16-bit reload timer 0 is supplied to UART0, and the clock of 16 bit reload timer 1 is supplied to UART1. )</li> </ul>
Data length	<ul style="list-style-type: none"> <li>• 7 bits (in asynchronous normal mode only)</li> <li>• 8 bits</li> </ul>
Signal mode	<ul style="list-style-type: none"> <li>• Non-return to zero (NRZ)</li> </ul>
Reception error detection	<ul style="list-style-type: none"> <li>• Framing error</li> <li>• Overrun error</li> <li>• Parity error (cannot be detected in multiprocessor mode)</li> </ul>
Interrupt request	<ul style="list-style-type: none"> <li>• Reception interrupt (reception completion and reception error detection)</li> <li>• Transmission interrupt (transmission completion)</li> <li>• Extended intelligent I/O service (EI<sup>2</sup>OS) is available for both transmission and reception interrupts</li> </ul>
Master-slave communication function (multiprocessor mode)	One-to-n communication (one master to n slaves) can be performed (this function is supported only for the master system)

Note :

During clock synchronous transfer, start and stop bits are not added so only data is transferred in UART.

**Table 20.1-2 ART operation mode**

Operation mode		Data length		Synchronization mode	Stop bit length
		When parity is enabled	When parity is disabled		
0	Normal mode	7 or 8 bits		Asynchronous	1 or 2 bits *2
1	Multiprocessor	8+1*1 bits	—	Asynchronous	
2	Normal mode	8 bits	—	Synchronous	None

— : Setting not possible.

\*1 : "+1" indicates the address/data selection bit (A/D) for communication control.

\*2 : During reception, only one stop bit can be detected.

## ■ UART Interrupt and EI<sup>2</sup>OS

**Table 20.1-3 UART interrupt and EI<sup>2</sup>OS**

Interrupt cause	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Upper	Bank	
UART1 reception interrupt	#37(25 <sub>H</sub> )	ICR13	0000BD <sub>H</sub>	FFFF68 <sub>H</sub>	FFFF69 <sub>H</sub>	FFFF6A <sub>H</sub>	⊙
UART1 transmission interrupt	#38(26 <sub>H</sub> )	ICR13	0000BD <sub>H</sub>	FFFF64 <sub>H</sub>	FFFF65 <sub>H</sub>	FFFF66 <sub>H</sub>	Δ
UART0 reception interrupt	#39(27 <sub>H</sub> )	ICR14	0000BE <sub>H</sub>	FFFF60 <sub>H</sub>	FFFF61 <sub>H</sub>	FFFF62 <sub>H</sub>	⊙
UART0 transmission interrupt	#40(28 <sub>H</sub> )	ICR14	0000BE <sub>H</sub>	FFFF5C <sub>H</sub>	FFFF5D <sub>H</sub>	FFFF5E <sub>H</sub>	Δ

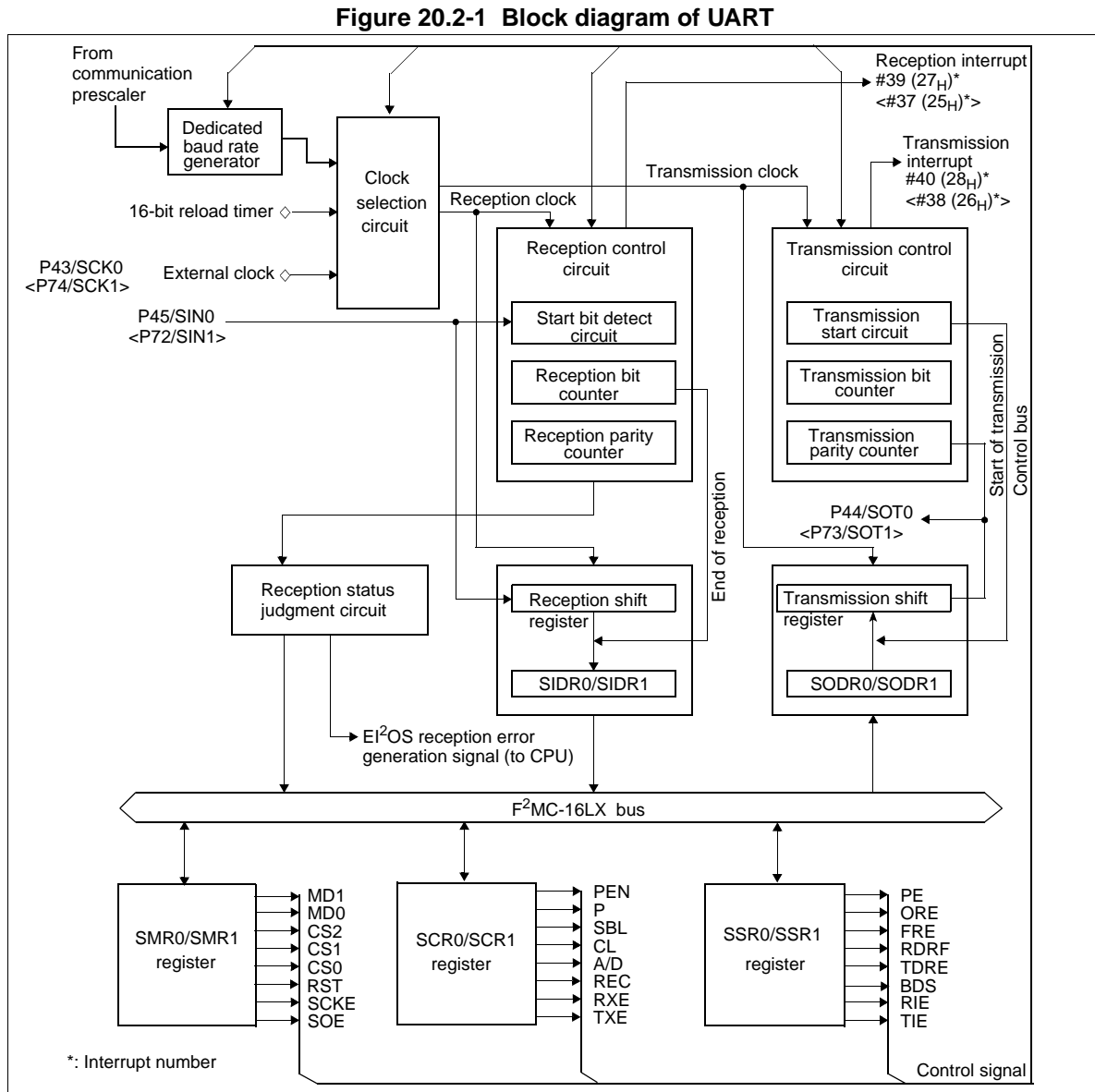
⊙ : Provided with a function that detects a UART reception error and stops EI<sup>2</sup>OS

Δ : Usable when ICR13 and ICR14 or interrupt causes that share an interrupt vector are not used

## 20.2 Block Diagram of UART

The block diagram of the UART is shown in Figure 20.2-1.

### ■ Block Diagram of UART



#### ● Clock selector

The clock selector selects the dedicated baud rate generator, external input clock, or internal timer output clock (clock supplied from the 16-bit reload timer) as the transmitting and receiving clocks.

**● Reception control circuit**

The reception control circuit consists of a received bit counter, start bit detection circuit, and received parity counter. The received bit counter counts receive datas. When reception of one data item for the specified data length is complete, the received bit counter generates a reception interrupt request. The start bit detection circuit detects start bits from the serial input signal. When the circuit detects a start bit, it writes data in the SIDR0/SIDR1 register by shifting per bit at the specified transfer rate. The received parity counter calculates the parity bit of the receive data.

**● Transmission control circuit**

The transmission control circuit consists of a transmission bit counter, transmission start circuit, and transmission parity counter. The transmission bit counter counts transmission datas. When transmission of one data item for the specified data length is complete, the transmission bit counter generates a transmission interrupt request. The transmission start circuit starts transmission when data is written to SIDR0/SIDR1. The transmission parity counter generates a parity bit for data to be transmitted when parity is enabled.

**● Reception shift register**

The reception shift register fetches receive data input from the SIN pin, shifting the data bit by bit. When reception is complete, the reception shift register transfers receive data to the SIDR0/ SIDR1 register.

**● Transmission shift register**

The transmission shift register transfers data written to the SODR0/SODR1 register to itself and outputs the data to the SOT pin, shifting the data bit by bit.

**● Mode control register (SMR0/SMR1)**

This register performs the following operations:

- Selecting a UART operation mode
- Selecting a clock input source
- Setting up the dedicated baud rate generator
- Selecting a clock rate (clock division value) when using the dedicated baud rate generator
- Specifying whether to enable or disable serial data output to the corresponding pin
- Specifying whether to enable or disable clock output to the corresponding pin

**● Control register (SCR0/SCR1)**

This register performs the following operations:

- Specifying whether to provide parity bits
- Selecting parity bits
- Specifying a stop bit length
- Specifying a data length
- Selecting a frame data format in mode 1
- Clearing an error flag
- Specifying whether to enable or disable transmission
- Specifying whether to enable or disable reception

- Serial status register (SSR0/SSR1)

This register checks the transmission and reception status and error status, and enables and disables transmission and reception interrupt requests.

- Serial input data register (SIDR0/SIDR1)

This register retains receive data. Serial input data is converted and stored in this register.

- Serial output data register (SODR0/SODR1)

This register sets transmission data. Data written to this register is converted to serial data and output.

**MB90820B Series****20.3 UART Pins**

This section describes the UART pins and provides a pin block diagram.

**■ UART Pins**

The UART pins also serve as general-purpose I/O ports. Table 20.3-1 lists the pin functions, I/O formats and settings required to use UART.

**Table 20.3-1 UART pins**

Pin name	Pin function	I/O format	Pull-up	Standby control	Setting required to use pin
P45/SIN0	Port 4 I/O or serial data input	CMOS output and CMOS hysteresis input	Not provided	Provided	Set as an input port (DDR4: bit 5 = 0)
P44/SOT0	Port 4 I/O or serial data output				Set to output enable mode (SMR0: SOE = 1)
P43/SCK0	Port 4 I/O or serial clock input/output				Set as an input port when a serial clock is input (DDR4: bit 3 = 0 )
					Set to output enable mode when a serial clock is output (SMR0: SCKE = 1)
P72/SIN1	Port 7 I/O or serial data input	CMOS output and CMOS hysteresis input	Not provided	Provided	Set as an input port (DDR7: bit 10 = 0)
P73/SOT1	Port 7 I/O or serial data output				Set to output enable mode (SMR1: SOE = 1)
P74/SCK1	Port 7 I/O or serial clock input/output				Set as an input port when a serial clock is input (DDR7: bit 12 = 0)
					Set to output enable mode when a serial clock is output (SMR1:SCKE = 1)

## ■ Block Diagram of UART Pins

Figure 20.3-1 Block diagram of UART serial data input pin(P45)

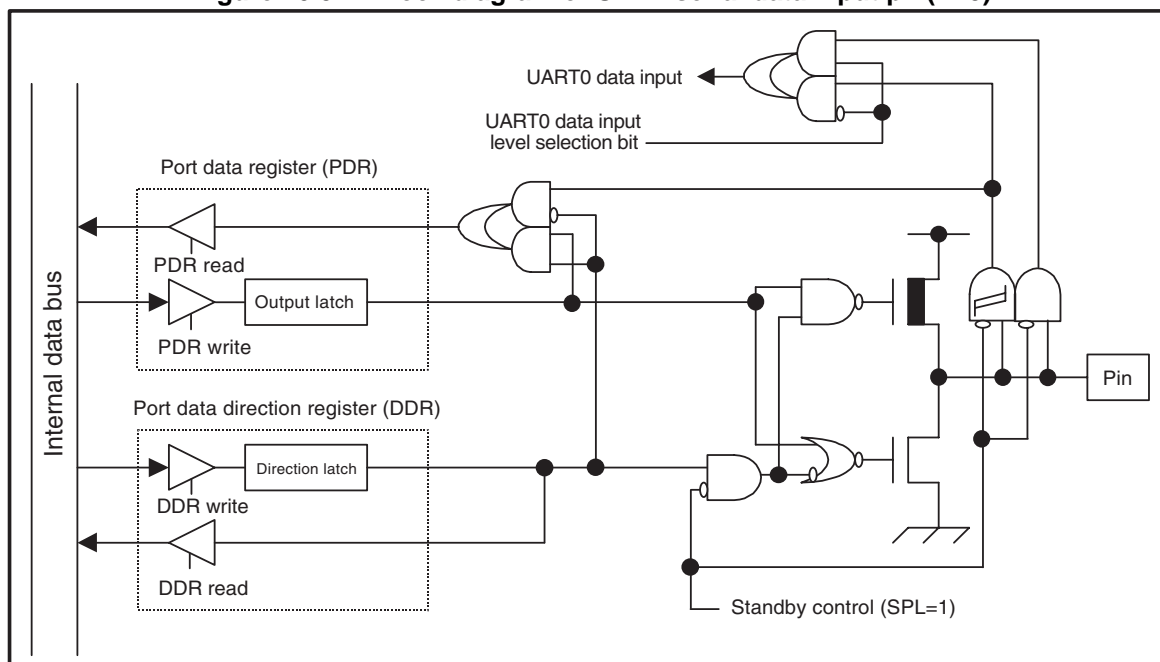


Figure 20.3-2 Block diagram of UART serial clock input/output pin(P43) & serial data output pin(P44)

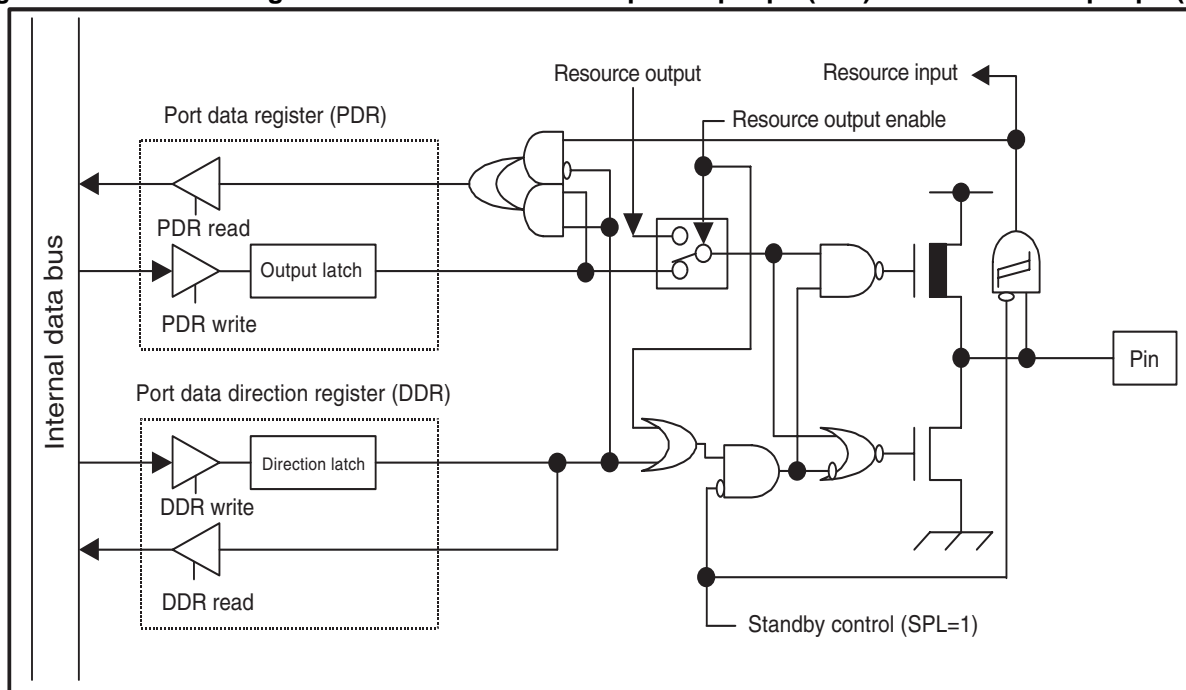


Figure 20.3-3 Block diagram of UART serial data input pin(P72))

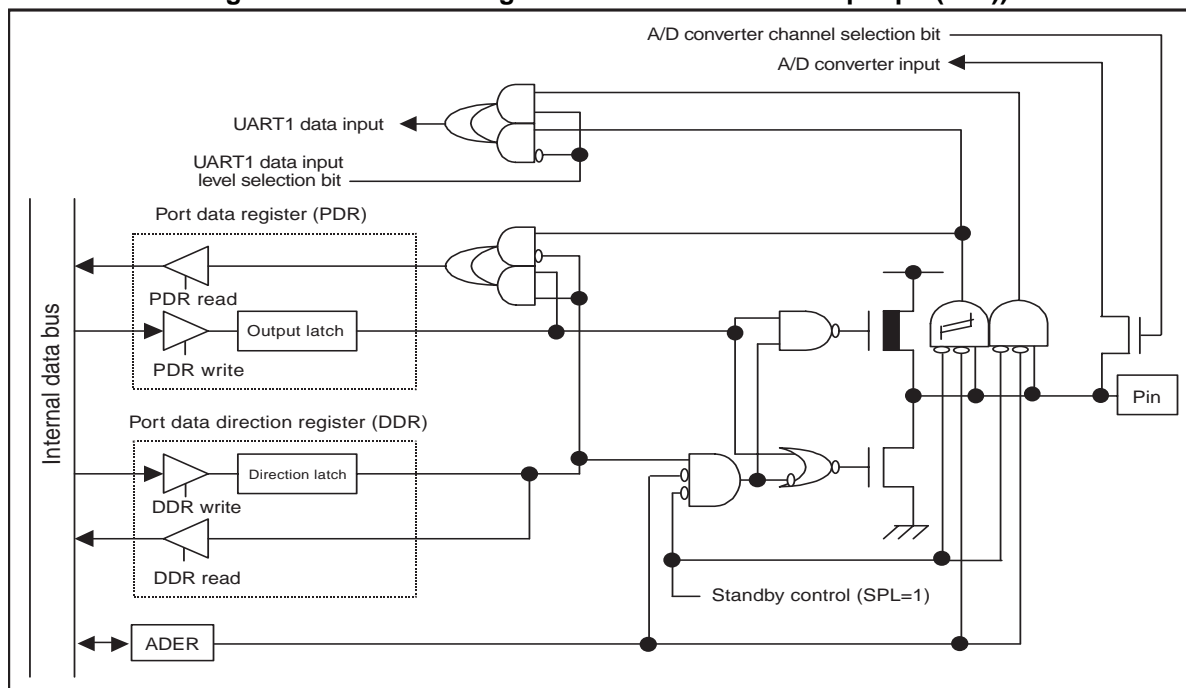
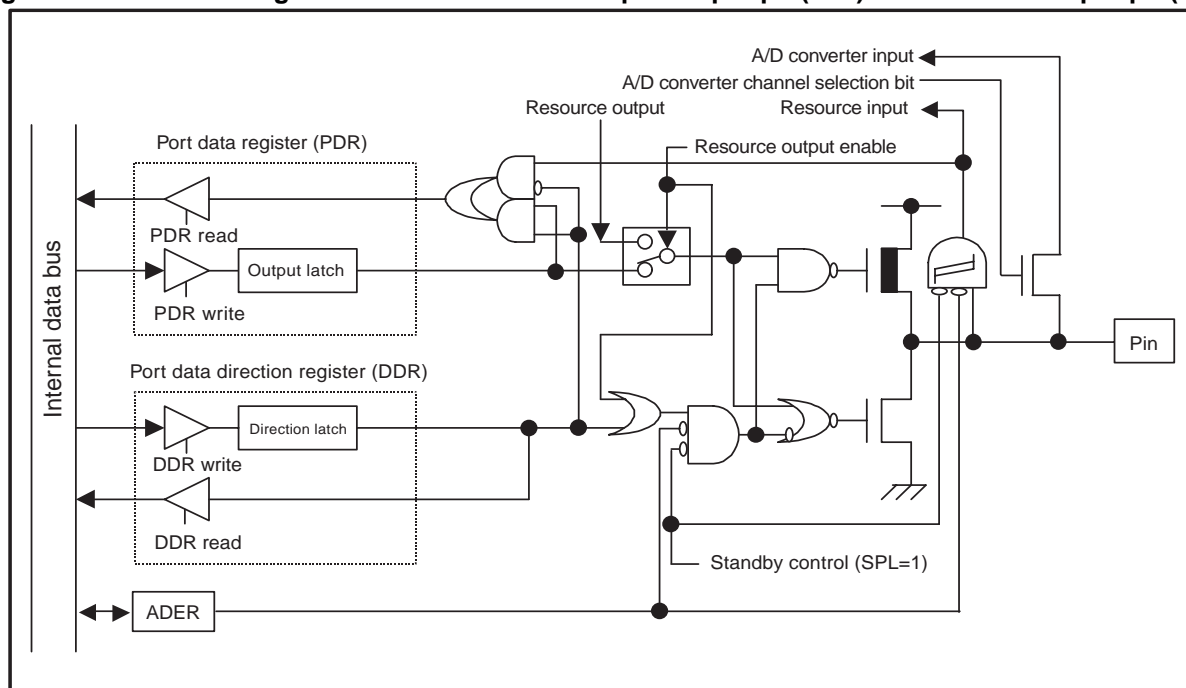


Figure 20.3-4 Block diagram of UART serial data input/output pin(P74) & serial data output pin(P73)



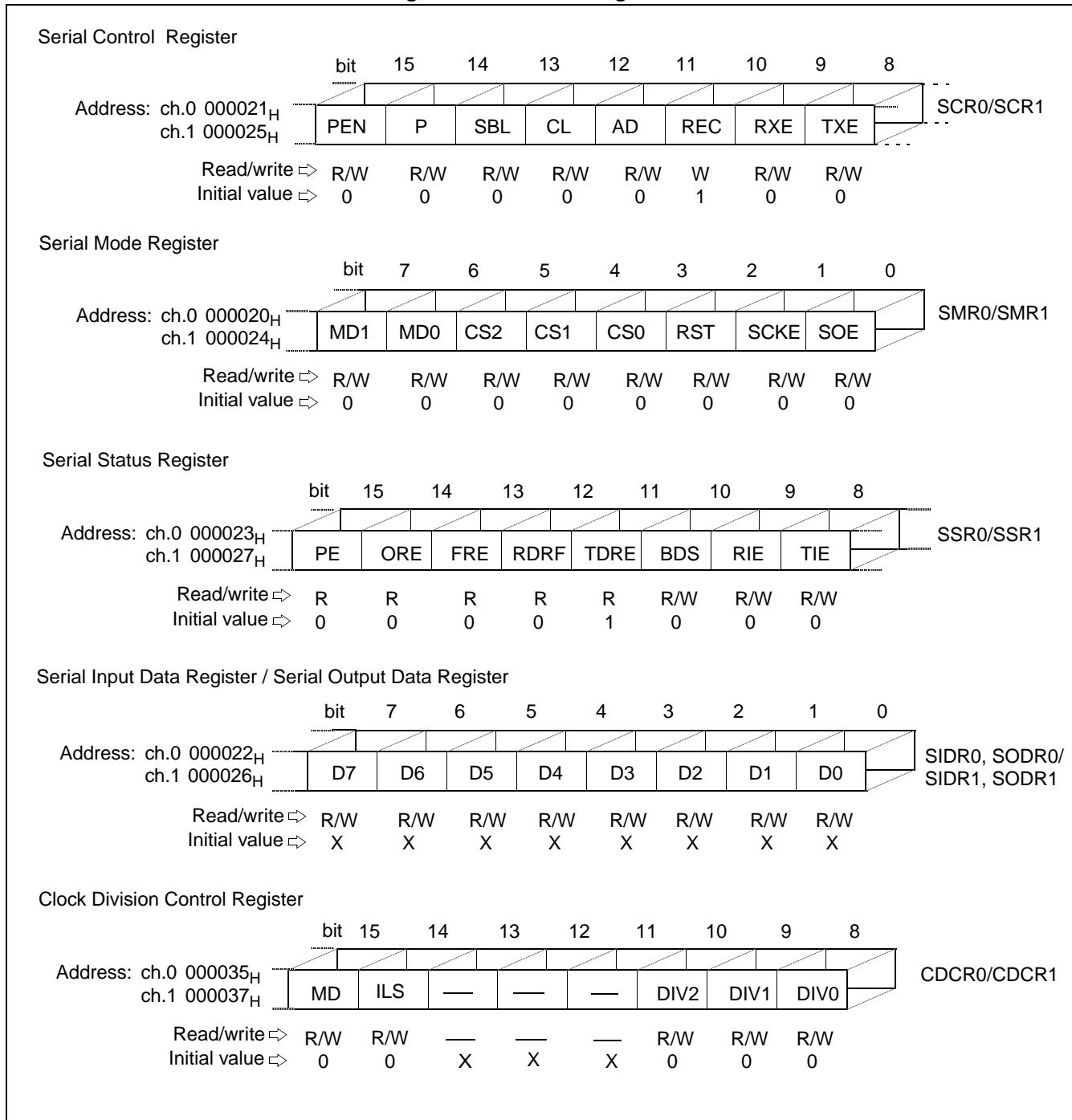


## 20.4 UART Registers

The following figure shows the UART registers.

### ■ UART Registers

Figure 20.4-1 UART registers



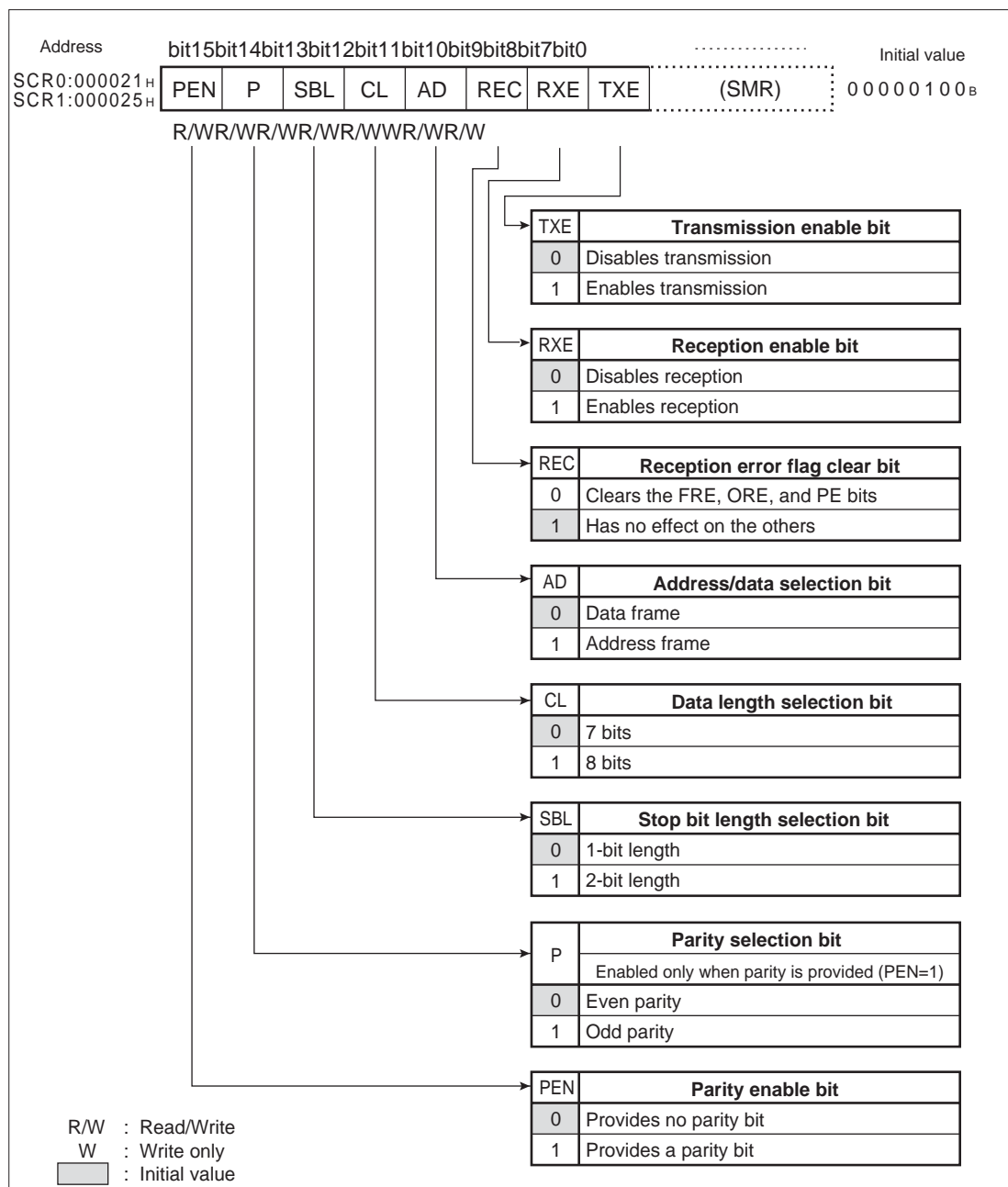
## MB90820B Series

### 20.4.1 Serial Control Register (SCR0/SCR1)

This register specifies parity bits, selects the stop bit and data lengths, selects a frame data format in mode 1, clears the reception error flag, and specifies whether to enable or disable transmission and reception.

#### Serial Control Register (SCR0/SCR1)

Figure 20.4-2 Serial control register (SCR0/SCR1)



**Table 20.4-1 Function of Serial control register (SCR0/SCR1)**

Bit name		Function
bit15	PEN: Parity enable bit	<ul style="list-style-type: none"> <li>This bit selects whether to add a parity bit during transmission or to detect it during reception.</li> </ul> <b>Note :</b> No parity can be used in operation modes 1 and 2. Therefore, fix this bit to 0.
bit14	P: Parity selection bit	<ul style="list-style-type: none"> <li>When parity is provided (PEN = 1), this bit selects an even or odd parity.</li> </ul>
bit13	SBL: Stop bit length selection bit	<ul style="list-style-type: none"> <li>This bit selects the length of the stop bits or the frame end mark bit of send data in asynchronous transfer mode.</li> </ul> <b>Note :</b> During reception, only the first bit of the stop bits is detected.
bit12	CL: Data length selection bit	<ul style="list-style-type: none"> <li>This bit specifies the length of send and receive data.</li> </ul> <b>Note :</b> Seven bits can be selected in operation mode 0 (asynchronous) only. Be sure to select eight bits (CL = 1) in operation mode 1 (multiprocessor mode) and operation mode 2 (synchronous).
bit11	A/D: Address/data selection bit	<ul style="list-style-type: none"> <li>Specify the data format of a frame to be sent or received in multiprocessor mode (mode 1).</li> <li>Select usual data frame when this bit is 0, and select address data frame when the bit is 1.</li> </ul>
bit10	REC: Reception error flag clear bit	<ul style="list-style-type: none"> <li>This bit clears the FRE, ORE and PE flags of the serial status register (SSR0/SSR1) to 0.</li> <li>Write 0 to this bit to clear the FRE, ORE and PE flags. Writing 1 to this bit has no effect on the others.</li> </ul> <b>Note :</b> If UART is active and a reception interrupt is enabled, clear the REC bit to 0 only when the FRE, DRE or PE flag indicates 1.
bit9	RXE: Reception enable bit	<ul style="list-style-type: none"> <li>This bit controls to enable or disable UART reception.</li> <li>When this bit is 0, reception is disabled. When it is 1, reception is enabled.</li> </ul> <b>Note :</b> If this bit is cleared during reception, reception can only be disabled until the reception of current frame is completed and the reception data is stored in the serial input data register (SIDR0/SIDR1).
bit8	TXE: Transmission enable bit	<ul style="list-style-type: none"> <li>This bit controls to enable or disable UART transmission.</li> <li>When this bit is 0, transmission is disabled. When the bit is 1, transmission is enabled.</li> </ul> <b>Note :</b> If this bit is cleared during transmission, transmission can only be disabled until all data in the serial output data register (SODR0/SODR1) has been transmitted.

## MB90820B Series

### 20.4.2 Serial Mode Register (SMR0/SMR1)

This register selects an operation mode and baud rate clock and specifies whether to enable or disable output of serial data and clocks to the corresponding pin.

#### Serial Mode Register (SMR0/SMR1)

Figure 20.4-3 Serial mode register (SMR0/SMR1)

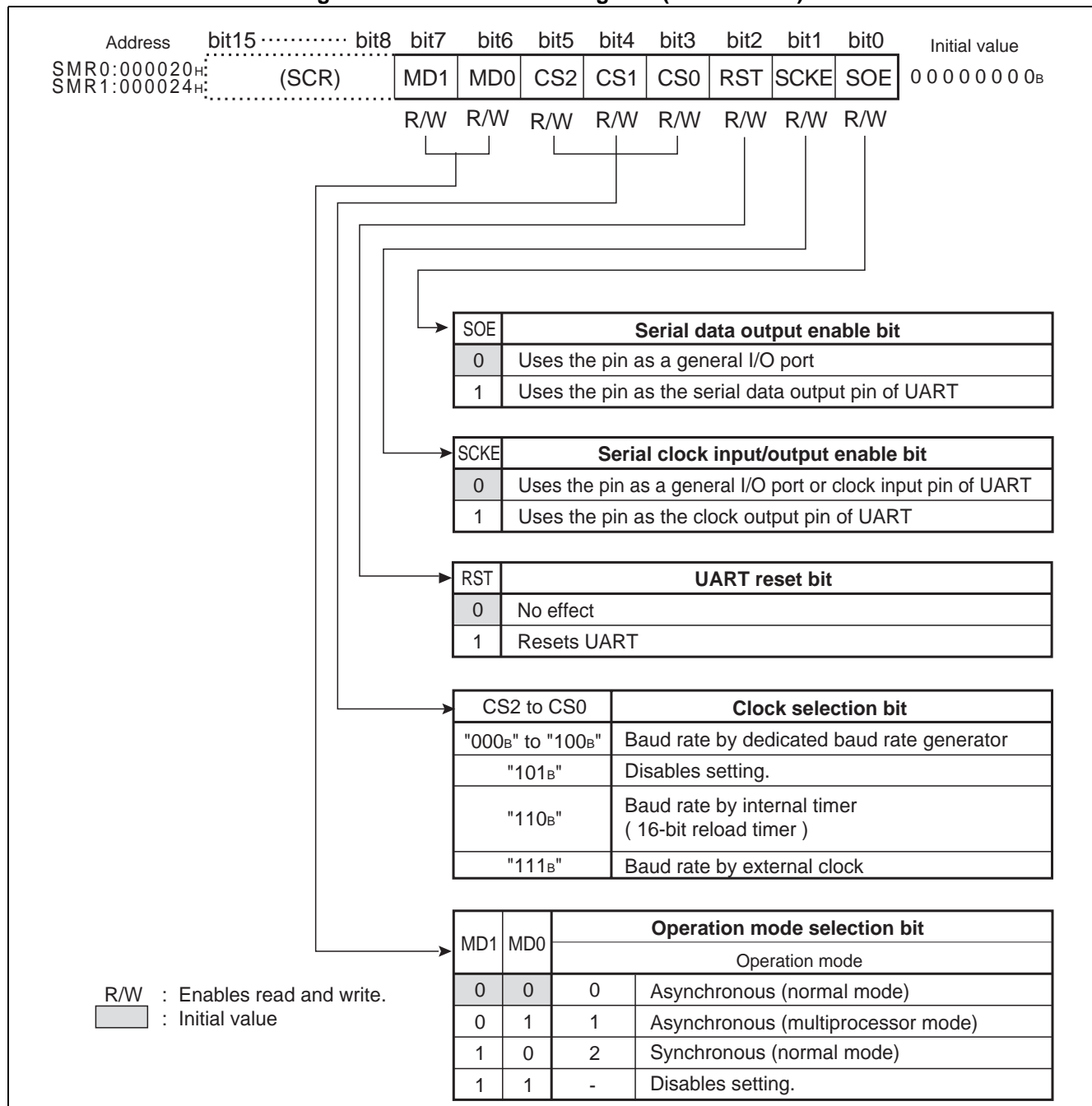


Table 20.4-2 Function of Serial mode register (SMR0/SMR1) (1 / 2)

Bit name		Function
bit7, bit6	MD1, MD0: Operation mode selection bits	<ul style="list-style-type: none"> <li>These bits select an operation mode.</li> </ul> <p><b>Note :</b> Operation mode 1 (multiprocessor mode) can be used only from the master system during master-slave communication. UART cannot be used from the slave system because it has no address/data detection function during reception.</p>
bit5 to bit3	CS2 to CS0: Clock input source selection bits	<p>These bits select the baud rate clock source. If selecting the dedicated baud rate generator, also set the baud rate.</p> <ul style="list-style-type: none"> <li>When selecting the dedicated baud rate generator, specify one of the six available baud rates, and set synchronous or asynchronous transfer mode. A total of eight baud rate settings are available if the baud rate is generated from an internal or external timer.</li> <li>The available clock input sources are an external clock (SCK0/SCK1 pins), the 16-bit reload timer0, and the dedicated baud rate generator.</li> </ul> <p><b>Note :</b> The following settings are prohibited if using the dedicated baud rate generator in synchronous transfer mode:</p> <ol style="list-style-type: none"> <li>1) CS2 to CS0 = 000<sub>B</sub></li> <li>2) CS2 to CS0 = 001<sub>B</sub> to DIV2 to DIV0 = 000<sub>B</sub></li> </ol>
bit2	RST: UART reset bit	<p>This bit resets the UART and registers CDCR0/CDCR1, SSR0/SSR1, SCR0/SCR1.</p> <ul style="list-style-type: none"> <li>Writing 0 to this bit has no effect.</li> <li>Writing 1 to this bit resets the UART and registers CDCR0/CDCR1, SSR0/SSR1, SCR0/SCR1. It will auto-clear after the reset operation.</li> <li>Always read as 0.</li> </ul> <p><b>Note :</b> After writing 1 to this bit, it is necessary to initialize the setting of UART and registers CDCR0/CDCR1, SSR0/SSR1, SCR0/SCR1 again.</p>

Table 20.4-2 Function of Serial mode register (SMR0/SMR1) (2 / 2)

Bit name		Function
bit1	SCKE: Serial clock input/ output enable bit	<ul style="list-style-type: none"> <li>This bit controls the serial clock input-output ports of the SCK0/SCK1 pin.</li> <li>When this bit is 0, the P43/SCK0 and P74/SCK1 pins operate as general input-output ports (P43 and P74) or serial clock input pins. When this bit is 1, the pins operate as serial clock output pins.</li> </ul> <p><b>Note :</b></p> <ul style="list-style-type: none"> <li>When using the P43/SCK0 and P74/SCK1 pins as serial clock input (SCKE = 0) pins, set the P43 and P74 as input ports. Also, select external clocks (SMR0/SMR1: CS2 to CS0 = 111<sub>B</sub>) using the clock selection bits.</li> <li>When using the pins as serial clock output (SCKE = 1) pins, select bits other than external timer (other than SMR0/SMR1: CS2 to CS0 = 111<sub>B</sub>).</li> </ul> <p><b>[Reference]</b> When the SCK0/SCK1 pin is assigned to serial clock output (SCKE = 1) pins, it functions as the serial clock output pin regardless of the status for the general input-output ports.</p>
bit0	SOE: Serial data output enable bit	<ul style="list-style-type: none"> <li>This bit enables or disables the output of serial data.</li> <li>When this bit is 0, the P44/SOT0 and P73/SOT1 pins operate as general input-output ports (P44 and P73). When this bit is 1, the P44/SOT0 and P73/SOT1 pins operate as serial data output pins (SOT0/SOT1).</li> </ul> <p><b>[Reference]</b> When serial data output pin is set (SOE=1), the P44/SOT0 and P73/SOT1 pins function as SOT0/SOT1 pins regardless of the status of general input-output ports (P44 and P73).</p>

20.4.3 Serial Status Register (SSR0/SSR1)

This register checks the transmission and reception status and error status, and enables and disables the transmission and reception interrupts.

Serial Status Register (SSR0/SSR1)

Figure 20.4-4 Serial status register (SSR0/SSR1)

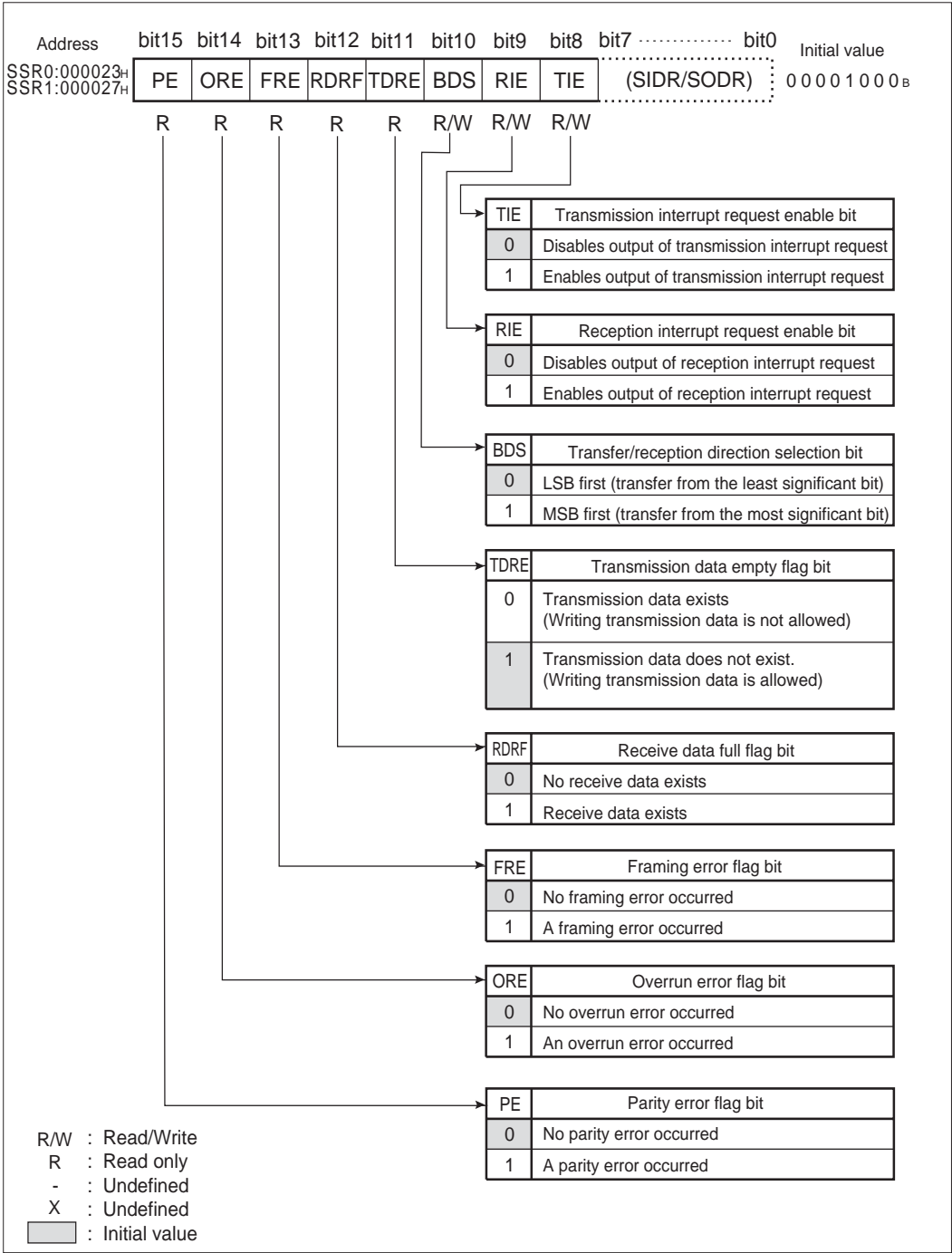


Table 20.4-3 Functions of each bit in serial status register (SSR0/SSR1)

Bit name		Function
bit15	PE: Parity error flag bit	<p>This flag indicates whether the parity error is performed during reception.</p> <ul style="list-style-type: none"> <li>This bit is set to 1 when a parity error occurs during reception and is cleared to 0 when 0 is written to the REC bit of the serial control register (SCR0/SCR1).</li> <li>A reception interrupt request is output when this bit and the RIE bit are 1.</li> <li>Data in serial input data register (SIDR0/SIDR1) is invalid when this bit is set.</li> </ul>
bit14	ORE: Overrun error flag bit	<p>This flag indicates whether the overrun error is generated during reception.</p> <ul style="list-style-type: none"> <li>This bit is set to 1 when an overrun error occurs during reception and is cleared to 0 when 0 is written to the REC bit of the serial control register (SCR0/SCR1).</li> <li>A reception interrupt request is output when this bit and the RIE bit are 1.</li> <li>Data in the serial input data register (SIDR0/SIDR1) is invalid when this bit is set.</li> </ul>
bit13	FRE: Framing error flag bit	<p>This flag indicates whether the framing error occurs during reception.</p> <ul style="list-style-type: none"> <li>This bit is set to 1 when a framing error occurs during reception and is cleared to 0 when 0 is written to the REC bit of the serial control register (SCR0/SCR1).</li> <li>A reception interrupt request is output when this bit and the RIE bit are 1.</li> <li>Data in the serial input data register (SIDR0/SIDR1) is invalid when this bit is set.</li> </ul>
bit12	RDRF: Receive data full flag bit	<ul style="list-style-type: none"> <li>This flag indicates the status of the serial input data register (SIDR0/SIDR1).</li> <li>This bit is set to 1 when receive data is loaded into SIDR0/SIDR1 and is cleared to 0 when SIDR0/SIDR1 is read.</li> <li>A reception interrupt request is output when the data is loaded to the SIDR and the RIE bit are 1.</li> </ul>
bit11	TDRE: Transmission data empty flag bit	<ul style="list-style-type: none"> <li>This flag indicates the status of serial output data register (SODR0/SODR1).</li> <li>This bit is cleared to 0 when transmission data is written to SODR0/SODR1 and is set to 1 when data is loaded into the transmission shift register and transmission starts.</li> <li>A transmission interrupt request is output when the data set in the SODR is transferred and the TIE bit are 1.</li> </ul> <p><b>Note :</b> This bit is set to 1 (SODR0/SODR1 empty) as its initial value.</p>
bit10	BDS: Transfer direction selection bit	<ul style="list-style-type: none"> <li>This bit selects whether to transfer serial data from the least significant bit (LSB first, BDS = 0) or the most significant bit (MSB first, BDS = 1).</li> </ul> <p><b>Note :</b> The high-order and low-order sides of serial data are interchanged with each other during reading from or writing to the serial data register. If this bit is set to another value after the data is written to the SODR register, the data becomes invalid.</p>
bit9	RIE: Reception interrupt request enable bit	<ul style="list-style-type: none"> <li>This bit enables or disables input of a request for reception interrupt to the CPU.</li> <li>A reception interrupt request is output when this bit and the RDRF are 1 and one or more error flag bits (PE, ORE and FRE) are 1.</li> </ul>
bit8	TIE: Transmission interrupt request enable bit	<ul style="list-style-type: none"> <li>This bit enables or disables output of a request for transmission interrupt to the CPU.</li> <li>A transmission interrupt request is output when this bit and the TDRE bit are 1.</li> </ul>



## 20.4.4 Serial Input Data Register (SIDR0/SIDR1) and Serial Output Data Register (SODR0/SODR1)

The serial input data register (SIDR0/SIDR1) is a serial data reception register. The serial output data register (SODR0/SODR1) is a serial data transmission register. Both SIDR0/SIDR1 and SODR0/SODR1 registers are located in the same address.

### Serial Input Data Register (SIDR0/SIDR1)

Figure 20.4-5 shows the bit configuration of serial input data register.

Figure 20.4-5 Serial input data register (SIDR0/SIDR1)

Serial input data register								Initial value	
Address :	bit 7	6	5	4	3	2	1	0	
SIDR0 000022 <sub>H</sub>	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX <sub>B</sub>
SIDR1 000026 <sub>H</sub>									
Read/write ⇒	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	
Initial value ⇒	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

SIDR0/SIDR1 is a register that contains receive data. The serial data signal transmitted to the SIN0/SIN1 pin is converted in the shift register and stored there. When the data length is 7 bits, the uppermost bit (D7) contains invalid data. When receive data is stored in this register, the receive data full flag bit (SSR0/SSR1: RDRF) is set to 1. If a reception interrupt request is enabled at this point (SSR0/SSR1: RIE=1), a reception interrupt occurs.

Read SIDR0/SIDR1 when the RDRF bit of the serial status register (SSR0/SSR1) is 1. The RDRF bit is cleared automatically to 0 when SIDR0/SIDR1 is read.

Data in SIDR0/SIDR1 is invalid when a reception error occurs (SSR0/SSR1: PE, ORE or FRE = 1).

### Serial Output Data Register (SODR0/SODR1)

Figure 20.4-6 shows the bit configuration of the serial output data register.

Figure 20.4-6 Serial output data register (SODR0/SODR1)

Serial output data register								Initial value	
Address : bit 7	6	5	4	3	2	1	0		
SODR0 000022 <sub>H</sub>	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX <sub>B</sub>
SODR1 000026 <sub>H</sub>									
Read/write ⇒	(W)	(W)	(W)	(W)	(W)	(W)	(W)	(W)	
Initial value ⇒	(X)	(X)	(X)	(X)	(X)	(X)	(X)	(X)	

This register is a data buffer register for the serial data transmission.

When data to be transmitted is written to this register in transmission enable state, it is transferred to the transmission shift register, then converted to serial data, and transmitted from the serial data output (SOT0/SOT1 pin). When the data length is 7 bits, the uppermost bit (D7) contains invalid data.

When transmission data is written to this register, the transmission data empty flag bit (SSR0/SSR1:TDRE) is cleared to "0". When transfer to the transmission shift register is complete, the bit is set to "1". When the TDRE bit is "1", the next piece of transmission data can be written. If output transmission interrupt requests have been enabled, a transmission interrupt is generated. Write the next piece of transmission data when a transmission interrupt is generated or the TDRE bit is "1".

---

**Note :**

SODR0/SODR1 is a write-only register and SIDR0/SIDR1 is a read-only register. These registers are located in the same address, so the read value is different from the write value. Therefore, instructions that perform a read-modify-write (RMW) operation, such as the INC/DEC instruction, cannot be used.

---

## 20.4.5 Communication Prescaler Control Register (CDCR)

This register controls the division ratio of machine clocks.

### ■ Communication Prescaler Control Register (CDCR)

The operation clocks of UART can be obtained by dividing machine clocks. UART is designed to obtain certain baud rates for various machine clock using this communication prescaler. Output from the communication prescaler is used for the operation clocks of I/O extended serial interfaces.

Figure 20.4-7 Communication prescaler control register (CDCR)

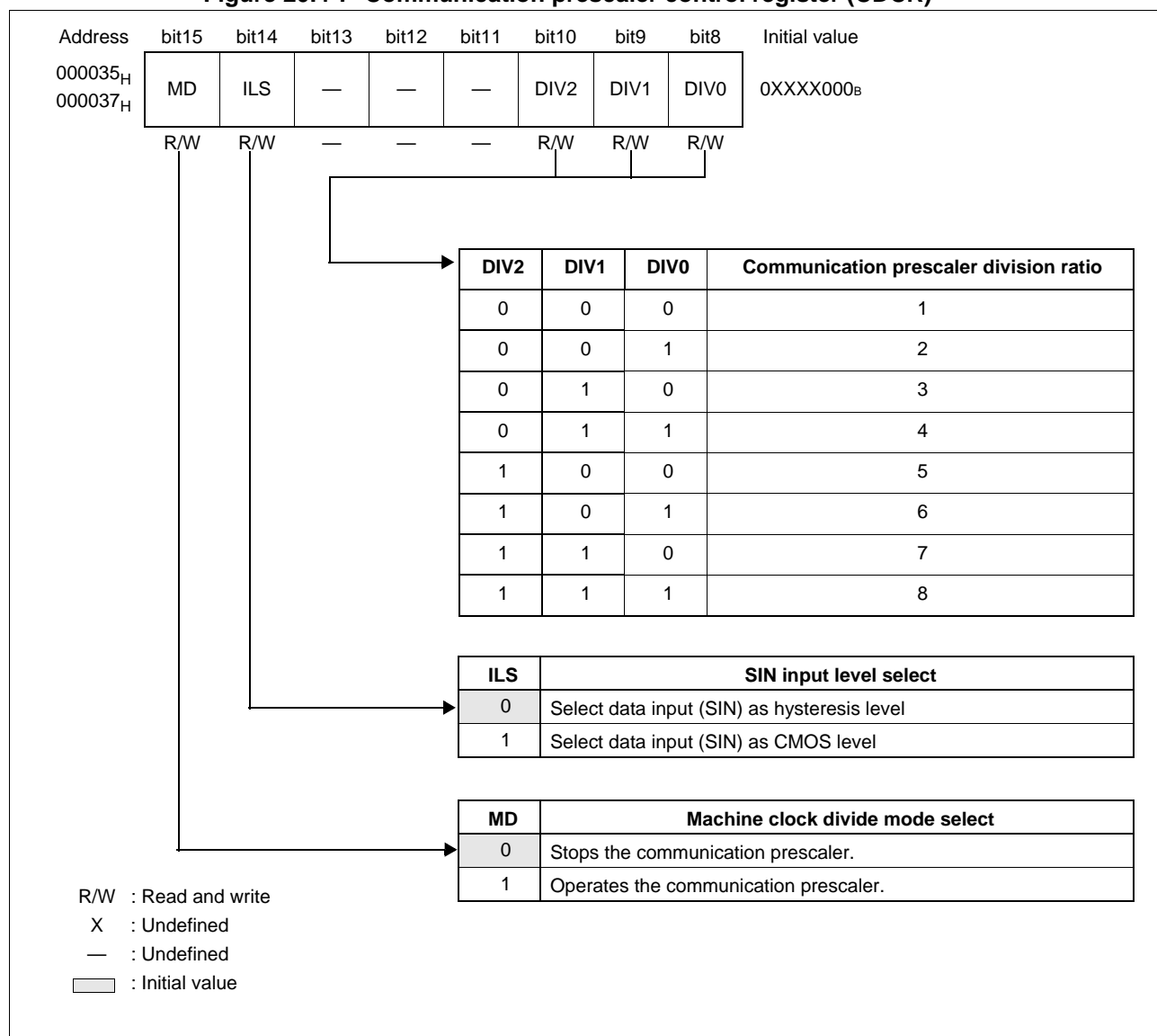


Table 20.4-4 Communication prescaler control register (CDCR)

Bit name		Function
bit15	MD: Machine clock divide mode select bit	<ul style="list-style-type: none"> <li>• This bit enables or stops the operation of the communication prescaler.</li> <li>• When "0" is set, the communication prescaler stops.</li> <li>• When "1" is set, the communication prescaler operates.</li> </ul>
bit 14	ILS: SIN input level select bit	<ul style="list-style-type: none"> <li>• This bit selects input level of UART data input pin (SIN)</li> <li>• Write "0" selects hysteresis input level.</li> <li>• Write "1" selects CMOS input level.</li> </ul>
bit13, bit12	Reserved bits	<ul style="list-style-type: none"> <li>• Always read as "0".</li> </ul>
bit10 to bit8	DIV2 to DIV0: Communication prescaler division ratio bits	<p>These bits set the divide ratio for the machine clock. The divide ratio can only be set when the MD bit is set to "1".</p> <ul style="list-style-type: none"> <li>• Once the divide ratio has been altered, 2 cycle wait is generated before starting the communication.</li> </ul> <p><b>Note:</b> The following settings are prohibited if using the dedicated baud rate generator in synchronous transfer mode:</p> <ol style="list-style-type: none"> <li>1) CS2 to CS0 = 000<sub>B</sub></li> <li>2) CS2 to CS0 = 001<sub>B</sub> and DIV2 to DIV0 = 000<sub>B</sub></li> </ol>

## 20.5 UART Interrupts

UART uses both reception and transmission interrupts. An interrupt request can be generated for either of the receive data is set in the serial input data register (SIDR0/SIDR1), or a reception error occurs and transmission data is transferred from serial output data register (SODR0/SODR1) to the transmission shift register.

The extended intelligent I/O service (EI<sup>2</sup>OS) is available for these interrupts.

### ■ UART Interrupts

Table 20.5-1 lists the interrupt control bits and causes of UART.

**Table 20.5-1 Interrupt control bits and interrupt causes of UART**

Reception/ transmission	Interrupt request flag bit	Operation mode			Interrupt cause	Interrupt output enable bit	When interrupt request flag is cleared
		0	1	2			
Reception	RDRF	O	O	O	Loading receive data into buffers (SIDR0/SIDR1)	SSR0/SSR1:RIE	Receive data is read
	ORE	O	O	O	Overrun error		“0” is written to the reception error flag clear bit (SSR0/SSR1: REC)
	FRE	O	O	X	Framing error		
	PE	O	X	X	Parity error		
Transmission	TDRE	O	O	O	Empty transmission buffer (SODR0/SODR1)	SSR0/SSR1:TIE	Transmission data is written

O : Used

X : Not used

#### ● Reception interrupt

The interrupt request is generated if one of the following events occurs in reception mode, the corresponding flag bit of the serial status register is set to 1:

- Data reception is complete (SSR0/SSR1: RDRF)
- Overrun error (SSR0/SSR1: ORE)
- Framing error (SSR0/SSR1: FRE)
- Parity error (SSR0/SSR1: PE)

When at least one of the flag bits is 1 and the reception interrupts are enabled (SSR0/SSR1: RIE = 1), a reception interrupt request is output to the interrupt controller.

When the serial input data register (SIDR0/SIDR1) is read, the receive data full flag (SSR0/SSR1: RDRF) is automatically cleared to “0”. When “0” is written to the REC bit of the serial control register (SCR0/SCR1), all the reception error flags (SSR0/SSR1: PE, ORE and FRE) are cleared to “0”.

● Transmission interrupt

When transmission data is transferred from the serial output data register (SODR0/SODR1) to the transfer shift register, the TDRE bit of the serial status register (SSR0/SSR1) is set to 1. When the transmission interrupts have been enabled (SSR0/SSR1: TIE = 1), a transmission interrupt request is output to the interrupt controller.

## ■ UART Interrupts and EI<sup>2</sup>OS

Table 20.5-2 UART interrupts and EI<sup>2</sup>OS

Interrupt cause	Interrupt number	Interrupt control register		Vector table address			EI <sup>2</sup> OS
		Register name	Address	Lower	Middle	Upper	
UART1 reception interrupt	#37(25 <sub>H</sub> )	ICR13	0000BD <sub>H</sub>	FFFF68 <sub>H</sub>	FFFF69 <sub>H</sub>	FFFF6A <sub>H</sub>	⊙
UART1 transmission interrupt	#38(26 <sub>H</sub> )	ICR13	0000BD <sub>H</sub>	FFFF64 <sub>H</sub>	FFFF65 <sub>H</sub>	FFFF66 <sub>H</sub>	Δ
UART0 reception interrupt	#39(27 <sub>H</sub> )	ICR14	0000BE <sub>H</sub>	FFFF60 <sub>H</sub>	FFFF61 <sub>H</sub>	FFFF62 <sub>H</sub>	⊙
UART0 transmission interrupt	#40(28 <sub>H</sub> )	ICR14	0000BE <sub>H</sub>	FFFF5C <sub>H</sub>	FFFF5D <sub>H</sub>	FFFF5E <sub>H</sub>	Δ

⊙ : Provided with a function that detects a UART reception error and stops EI<sup>2</sup>OS

Δ : Usable when interrupt causes that share the ICR13 and ICR14 or the interrupt vectors are not used

## ■ UART EI<sup>2</sup>OS Functions

UART has a circuit for operating EI<sup>2</sup>OS, which can be started up for either reception or transmission interrupts.

● For reception

EI<sup>2</sup>OS can be used regardless of the status of other resources.

● For transmission

UART shares the interrupt control registers (ICR13 and ICR14) with the UART reception interrupts. Therefore, EI<sup>2</sup>OS can be started up only when no UART reception interrupts are used.

## 20.5.1 Reception Interrupt Request Generation and Flag Set Timing

The following are the reception interrupt causes: completion of reception (SSR0/SSR1: RDRF) and occurrence of a reception error (SSR0/SSR1: PE, ORE, or FRE).

### ■ Reception Interrupt Request Generation and Flag Set Timing

Receive data is stored in serial input data register (SIDR0/SIDR1) if a stop bit is detected (in operation mode 0 or 1) or the last bit (D7) of data is detected (in operation mode 2) during reception. At that time, if a reception error is detected, the error flags (SSR0/SSR1: PE, ORE and FRE) are set, then the receive data full flag (SSR0/SSR1: RDRF) is set to 1. If one of the error flags is 1 in each mode, the SIDR0/SIDR1 register contains invalid data.

#### ● Operation mode 0 (asynchronous, normal mode)

The RDRF bit is set to 1 when a stop bit is detected. If a reception error is detected, the error flags (PE, ORE and FRE) are set to 1.

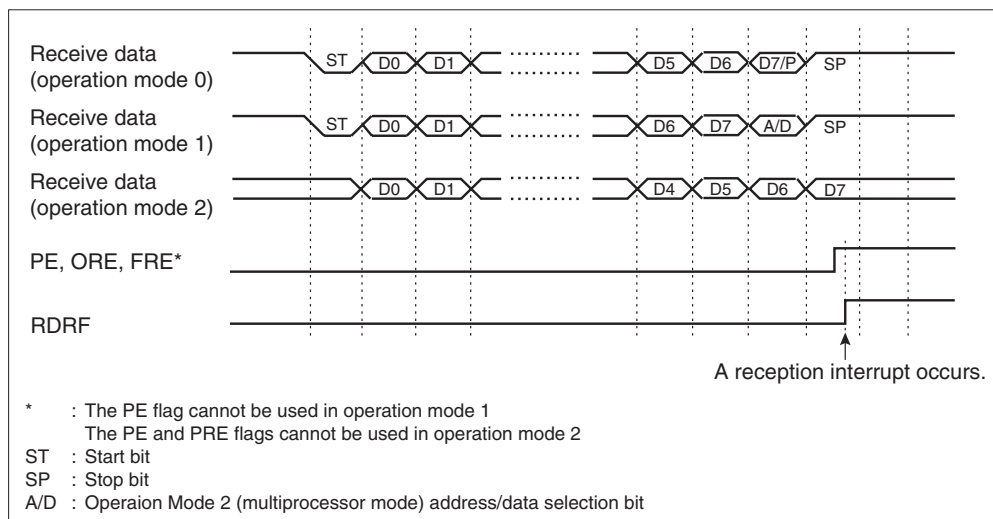
#### ● Operation mode 1 (asynchronous, multiprocessor mode)

The RDRF bit is set to 1 when a stop bit is detected. If a reception error is detected, the error flags (ORE and FRE) are set to 1. Parity errors cannot be detected.

#### ● Operation mode 2 (synchronous, normal mode)

The RDRF bit is set to 1 when the last bit of receive data (D7) is detected. If a reception error is detected, the error flag (ORE) is set. Parity and framing errors cannot be detected. Figure 20.5-1 below shows the reception operation and flag set timing.

Figure 20.5-1 Reception operation and flag set timing



- Reception interrupt request generation timing

When the RDRF, PE, ORE or FRE flag is set to 1 in the reception interrupt enable state (SSR0/SSR1:RIE = 1), reception interrupt requests (#37 and #39) are generated.



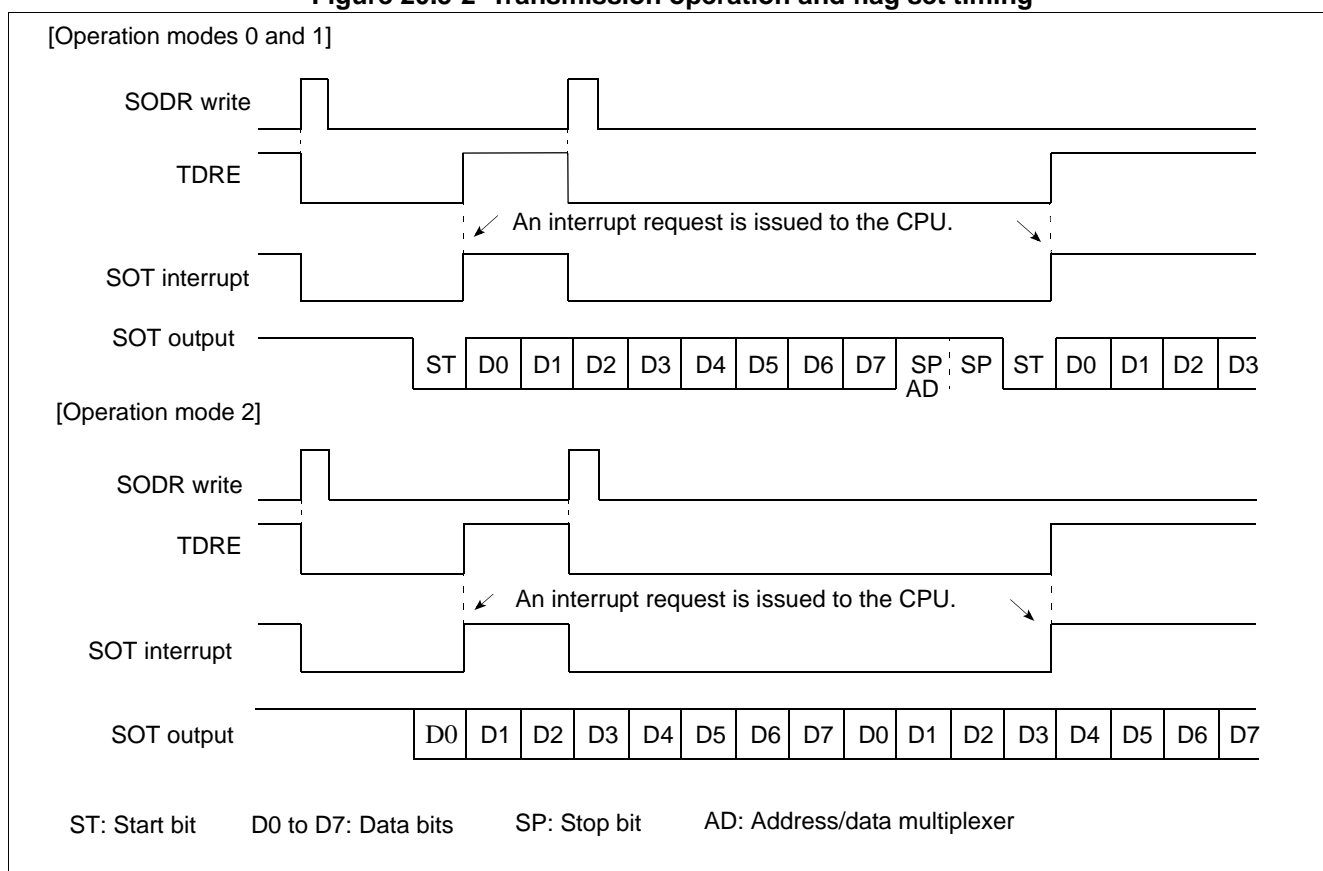
## 20.5.2 Transmission Interrupt Request Generation and Flag Set Timing

A transmission interrupt request is generated when the next piece of data is ready to be written to the serial output data register (SODR0/SODR1).

### ■ Transmission Interrupt Request Generation and Flag Set Timing

The transmission data empty flag bit (SSR0/SSR1: TDRE) is set to 1 when data written to the serial output data register (SODR0/SODR1) is transferred to the transmission shift register, and the next piece of data is ready to be written. TDRE is cleared to 0 when transmission data is written to SODR0/SODR1. Figure 20.5-2 shows the transmission operation and flag set timing.

Figure 20.5-2 Transmission operation and flag set timing



### ● Transmission interrupt request generation timing

If the TDRE flag is set to 1 when a transmission interrupt is enabled (SSR0/SSR1: TIE = 1), transmission interrupt requests (#38 and #40) are generated.

---

Note :

A transmission completion interrupt is generated immediately after the transmission interrupts are enabled (TIE = 1) because the TDRE bit is set to 1 as its initial value. TDRE bit is a read-only bit that can be cleared only by writing new data to the serial output data register (SODR0/SODR1). Carefully specify the transmission interrupt enable timing.

---

## **20.6 UART Baud Rates**

---

**One of the following can be selected as the UART transmitting/receiving clocks:**

- **Dedicated baud rate generator**
  - **Internal clock (16-bit reload timer)**
  - **External clock (clock of SCK pin input)**
- 

### **■ UART Baud Rate Selection**

The baud rate selection circuit is designed as shown below. One of the following three types for baud rates can be selected:

#### ● Baud rates determined using the dedicated baud rate generator

UART has an internal dedicated baud rate generator. One of eight baud rates can be selected using the mode control register (SMR0/SMR1).

An asynchronous or synchronous baud rate is selected using the machine clock frequency and setting the CS2 to CS0 bits of the mode control register (SMR0/SMR1) .

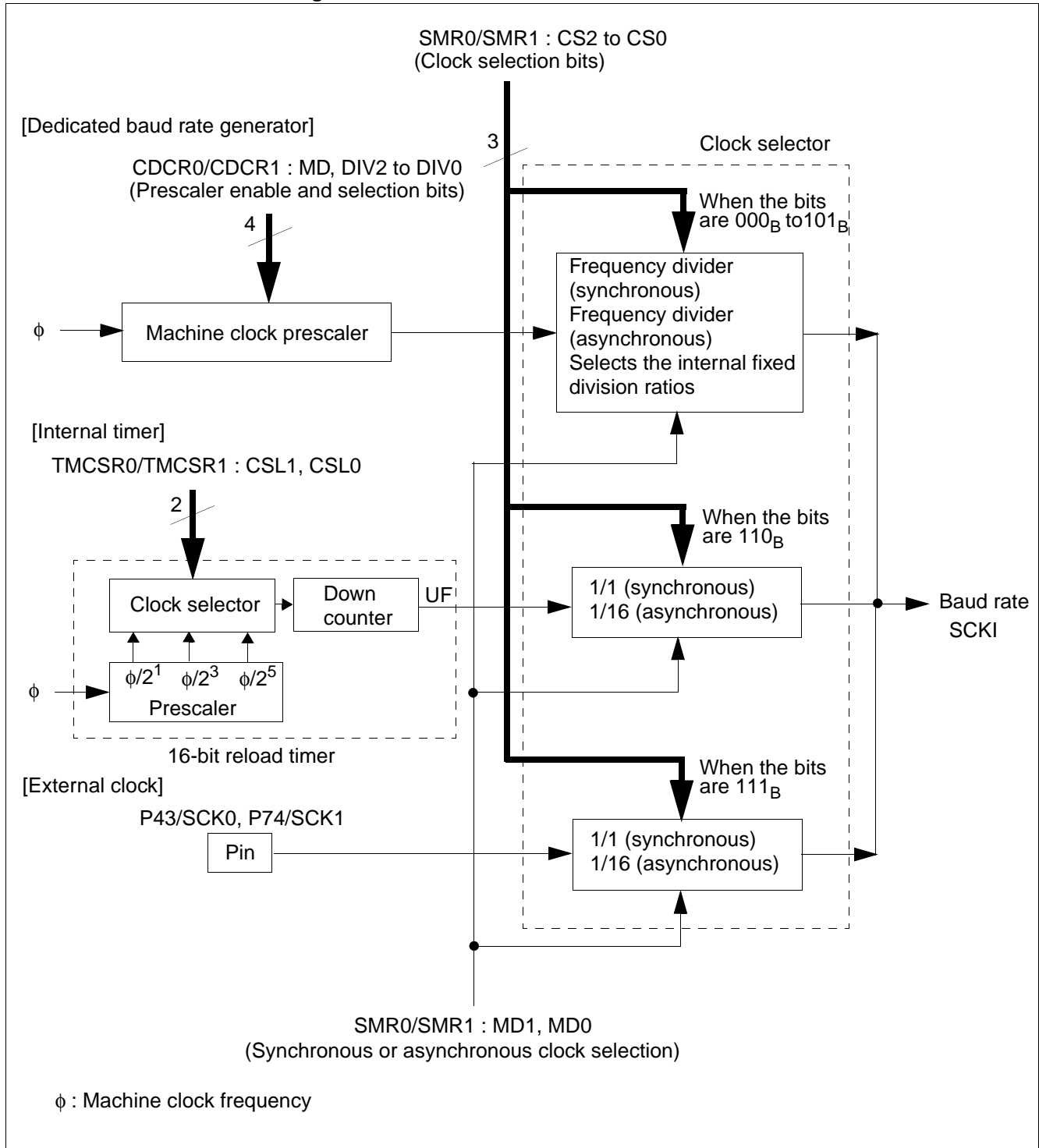
#### ● Baud rates determined using the internal timer

The internal clock supplied from 16-bit reload timer is used as it is (synchronous) or by dividing it by 16 (asynchronous) for the baud rate. Any baud rate can be set by setting the reload value.

#### ● Baud rates determined using the external clock

The clock input from the UART clock input pins (P43/SCK0 and P74/SCK1) is used as it is (synchronous) or by dividing it by 16 (asynchronous) for the baud rate. Any baud rate can be set externally.

Figure 20.6-1 UART baud rate selection circuit



## 20.6.1 Baud Rates Determined Using the Dedicated Baud Rate Generator

This section describes the baud rates that can be set when the clock from the dedicated baud rate generator is selected as the UART Transmission/reception clock.

### ■ Baud Rates Determined Using the Dedicated Baud Rate Generator

When the transmission/reception clock is generated using the dedicated baud rate generator, the machine clock is divided with the machine clock prescaler. The divided machine clock is then divided by the transfer clock division ratio selected with the clock selector again. The machine clock division ratios are common to the asynchronous and synchronous baud rates, but different values set internally are selected as the transfer clock division ratio for the asynchronous and synchronous baud rates.

The actual transfer rate can be calculated using the following formulas:

asynchronous baud rate =  $\phi \times (\text{prescaler division ratio}) \times (\text{asynchronous transfer clock division ratio})$

synchronous baud rate =  $\phi \times (\text{prescaler division ratio}) \times (\text{synchronous transfer clock division ratio})$

$\phi$  : Machine clock frequency

- Division ratios for the communication prescaler (common to asynchronous and synchronous baud rates)

Each machine clock division ratio is selected using the DIV2 to DIV0 bits of the CDCR register as listed in Table 20.6-1.

**Table 20.6-1 Output frequency of communication prescaler**

MD	DIV2	DIV1	DIV0	Division ratio
0	–	–	–	Stops
1	0	0	0	1
1	0	0	1	2
1	0	1	0	3
1	0	1	1	4
1	1	0	0	5
1	1	0	1	6
1	1	1	0	7
1	1	1	1	8

● Synchronous baud rate division ratios

A division ratio for synchronous baud rates is selected using the CS2 to CS0 bits of the serial mode control register (SMR0/SMR1) as listed in Table 20.6-2

**Table 20.6-2 Selection of synchronous baud rate division ratios**

CS2	CS1	CS0	CLK synchronization	Calculation formula
0	0	0	2 MHz	$(\phi \div \text{div}) / 1$
0	0	1	1 MHz	$(\phi \div \text{div}) / 2$
0	1	0	500 kHz	$(\phi \div \text{div}) / 4$
0	1	1	250 kHz	$(\phi \div \text{div}) / 8$
1	0	0	125 kHz	$(\phi \div \text{div}) / 16$
1	0	1	62.5 kHz	$(\phi \div \text{div}) / 32$

Note that the calculation is supposing that  $\phi$  (machine clock) = 16 MHz and div (machine clock division ratio) = 8. The maximum baud rate is 1/8 machine clock.

● Asynchronous baud rate division ratios

A division ratio for asynchronous baud rates is selected using the CS2 to CS0 bits of the serial mode control register (SMR0/SMR1) as listed in Table 20.6-3

**Table 20.6-3 Selection of asynchronous baud rate division ratios**

CS2	CS1	CS0	Asynchronous (start-stop synchronization)	Calculation formula
0	0	0	76923 Hz	$(\phi \div \text{div}) / (8 \times 13 \times 2)$
0	0	1	38461 Hz	$(\phi \div \text{div}) / (8 \times 13 \times 4)$
0	1	0	19230 Hz	$(\phi \div \text{div}) / (8 \times 13 \times 8)$
0	1	1	9615 Hz	$(\phi \div \text{div}) / (8 \times 13 \times 16)$
1	0	0	500 kHz	$(\phi \div \text{div}) / (8 \times 2 \times 2)$
1	0	1	250 kHz	$(\phi \div \text{div}) / (8 \times 2 \times 4)$

Note that the calculation is supposing that  $\phi$  (machine clock) = 16 MHz, div (machine clock division ratio) = 1.

● Internal timer

When CS2 to CS0 are set to 110<sub>B</sub> and the internal timer is selected, the formulas for calculating baud rates (when using the reload timer) are as follows:

Asynchronous (start-stop synchronization):  $(\phi \div N) / (16 \times 2 \times (n + 1))$

CLK synchronization:  $(\phi \div N) / (2 \times (n + 1))$

N: Division ratio for the prescaler of 16-bit reload timer

n: Reload value of the 16-bit reload timer

---

Note :

In mode 2 (CLK synchronization mode), SCK0/SCK1 is up to three clocks later than SCKI. A logically attainable transfer rate is 1/3 of the system clock frequency. However, 1/4 of the system clock frequency is recommended as taken from the actual specifications.

---

● External clock

When CS2 to CS0 are set to 111<sub>B</sub> and the external timer is selected, note the following:

If the external clock frequency is specified as  $f$ , the following baud rates are assumed:

Asynchronous (start-stop synchronization):  $f/16$

CLK synchronization:  $f$

Note that the maximum external clock frequency  $f$  is 2 MHz.

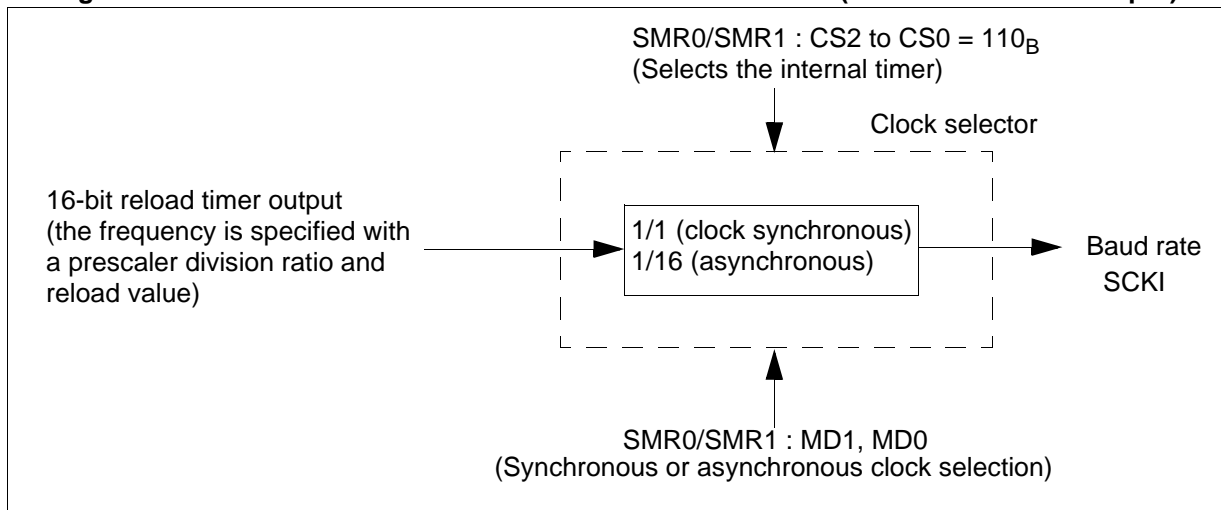
## 20.6.2 Baud Rates Determined Using the Internal Timer (16-bit Reload Timer)

This section describes the settings used when the internal clock supplied from 16-bit reload timer is selected as the UART transmission/reception clock. It also shows the baud rate calculation formulas.

### ■ Baud Rates Determined Using the Internal Timer (16-bit Reload Timer)

Writing 110<sub>B</sub> to the CS2 to CS0 bits of the serial mode control register (SMR0/SMR1) selects the baud rate determined using the internal timer. Any baud rate can be set by specifying a prescaler division ratio and reload value for 16-bit reload timer. Figure 20.6-2 shows the baud rate selection circuit for the internal timer.

Figure 20.6-2 Baud rate selection circuit for the internal timer (16-bit reload timer output)



#### ● Baud rate calculation formulas

$$\text{Asynchronous baud rate} = \frac{\phi}{X(n+1) \times 2 \times 16} \text{ bps}$$

$$\text{Synchronous baud rate} = \frac{\phi}{X(n+1) \times 2} \text{ bps}$$

$\phi$ : Machine clock frequency

X: Division ratio for the prescaler of 16-bit reload timer ( $2^1, 2^3, 2^5$ )

n: Reload value for 16-bit reload timer (0 to 65535)



- Examples of setting baud rate and reload register values (machine clock: 7.3728 MHz)

**Table 20.6-4 Baud rates and reload register values**

Baud rate (bps)	Reload value			
	Clock asynchronous (start-stop synchronization)		Clock synchronous	
	X=2 <sup>1</sup> (machine clock divided by 2)	X=2 <sup>3</sup> (machine clock divided by 8)	X=2 <sup>1</sup> (machine clock divided by 2)	X=2 <sup>3</sup> (machine clock divided by 8)
38400	2	–	47	11
19200	5	–	95	23
9600	11	2	191	47
4800	23	5	383	95
2400	47	11	767	191
1200	95	23	1535	383
600	191	47	9071	767
300	383	95	6143	1535

X : Division ratio for the prescaler of 16-bit reload timer

– : Setting not allowed

---

Note :

The following settings are prohibited in clock synchronous mode.

N=1, n=0

---

## MB90820B Series

### 20.6.3 Baud Rates Determined Using the External Clock

This section describes the settings used when the external clock is selected as the UART transmission and reception clock. It also shows the baud rate calculation formulas.

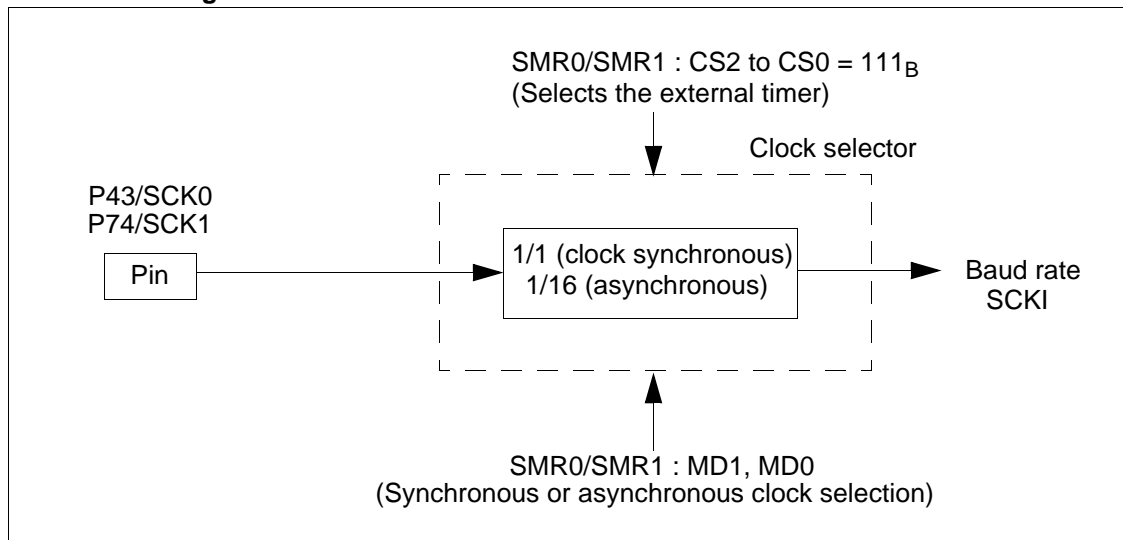
#### ■ Baud Rates Determined Using the External Clock

The following three settings are required to select the baud rate determined by using the external clock input:

- Write  $111_B$  to the CS2 to CS0 bits of the serial mode control register (SMR0/SMR1) to select the baud rate determined by using the external clock input.
- Set the P43/SCK0 and P74/SCK1 pins as input ports (DDR4: bit 3 = 0 and DDR7: bit 12 = 0).
- Write 0 to the SCKE bit of the serial mode control register (SMR0/SMR1) to set the pin as an external clock input pin.

As shown in Figure 19.6.3-1, a baud rate is selected using the external clock input from the SCK1 pin. To change the baud rate, the external input clock cycle must be changed because the internal division ratio is fixed.

**Figure 20.6-3 Baud rate selection circuit for the external clock**



#### ● Baud rate calculation formulas

Asynchronous baud rate =  $f/16$

Synchronous baud rate =  $f$

$f$ : External clock frequency (up to 2 MHz)

## 20.7 Operation of UART

UART operates in operation modes 0 and 2 for bidirectional serial communication and in operation mode 1 for master-slave communication.

### ■ Operation of UART

#### ● Operation modes

There are three UART operation modes: modes 0 to 2. As listed in Table 19.7-1, an operation mode can be selected according to the inter-CPU connection method and data communication mode.

**Table 20.7-1 UART operation mode**

Operation mode		Data length		Synchronization mode	Stop bit
		When parity is disabled	When parity is enabled		
0	Normal mode	7 or 8 bits		Asynchronous	1 or 2 bits *2
1	Multiprocessor mode	8+1*1 bits	—	Asynchronous	
2	Normal mode	8 bits	—	Synchronous	None

— : Setting not possible.

\*1 : "+1" indicates the address/data selection bit (A/D) for communication control.

\*2 : During reception, only one stop bit can be detected.

Note :

Operation mode 1 of UART is used only from the master system during master-slave connection.

#### ● Inter-CPU connection method

One-to-one connection (normal mode) and master-slave connection (multiprocessor mode) can be selected. For either connection method, the data length, whether to enable parity, and the synchronization method must be common to all CPUs. Select an operation mode as follows:

- In the one-to-one connection method, operation mode 0 or 2 must be used in the two CPUs. Select operation mode 0 for asynchronous mode and operation mode 2 for clock synchronous mode.
- Select operation mode 1 for the master-slave connection method and use it from the master system. Select "When parity is disabled" for this connection method.

#### ● Synchronization method

Asynchronous mode (start-stop synchronization) or clock synchronous mode can be selected in different operation modes.

- Signal mode

UART can treat data only in NRZ (Non-return to Zero) format.

- Operation enable bit

UART controls both transmission and reception using the operation enable bit for TXE (transmission) and that for RXE (reception). If each of the operations is disabled, stop it as follows:

- If reception operation is disabled during reception (data is input to the reception shift register), finish frame reception and store the received data in the serial input data register (SIDR0/SIDR1). Then stop the reception operation.
- If the transmission operation is disabled during transmission (data is output from the transmission shift register), wait until there is no data in the serial output data register (SODR0/SODR1) before stopping the transmission operation.

## 20.7.1 Operation in Asynchronous Mode (Operation Modes 0 and 1)

**When UART is used in operation mode 0 (normal mode) or operation mode 1 (multiprocessor mode), the asynchronous mode is selected.**

### ■ Operation in Asynchronous Mode

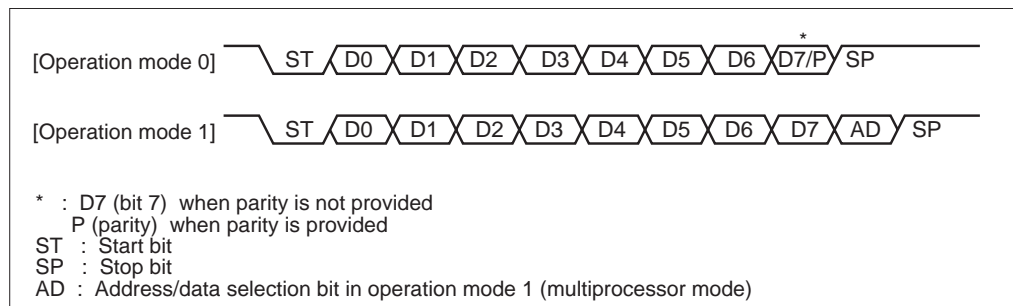
#### ● Transmission/reception data format

Transmission/reception data begins with the start bit (L level) and ends with the stop bit (H level). The data of the specified data bit length is transferred in LSB first mode.

- The data length can be set to 7 or 8 bits in normal mode for operation mode 0.
- In operation mode 1, the length of data is fixed to 8 bits with an address/data (A/D) selection bit added instead of parity.

Figure 20.7-1 shows the transmission/reception data format in asynchronous mode.

**Figure 20.7-1 Transmission/reception data format (operation modes 0 and 1)**



#### ● Transmission operation

Transmission data is written to the serial output data register (SODR0/SODR1) when the transmission data empty flag bit (SSR0/SSR1: TDRE) is 1. This data is transmitted if the transmission operation is enabled (SCR0/SCR1: TXE = 1).

The TDRE flag is again set to 1 when the transmission data is transferred to the transmission shift register and its transmission starts. Then, the next piece of transmission data gets ready to be set. At this point, a transmission interrupt request is output requesting that the next piece of transmission data be set in the SODR0/SODR1 register if that request is enabled (SSR0/SSR1: TIE = 1). The TDRE flag is cleared to 0 when the transmission data is written to SODR0/SODR1.

#### ● Reception operation

Reception operation is performed every time it is enabled (SCR0/SCR1: RXE = 1). When a start bit is detected, a frame of data is received according to the data format specified by the control register (SCR0/SCR1). After the frame has been received, the error flag is set if an error occurs, then the receive data full flag bit (SSR0/SSR1: RDRF) is set to 1. At this point, a reception interrupt request is output if it is enabled (SSR0/SSR1: TIE = 1).

Check each flag of the input data register (SIDR0/SIDR1). If the reception is normal, read the input data register (SIDR0/SIDR1). If an error is found, proceed to error handling. The RDRF flag is cleared to 0 every time receive data is read from SIDR0/SIDR1.

## ● Stop bit

For transmission, 1 or 2 bits can be selected. During reception however, the first bit is the only one that is always checked.

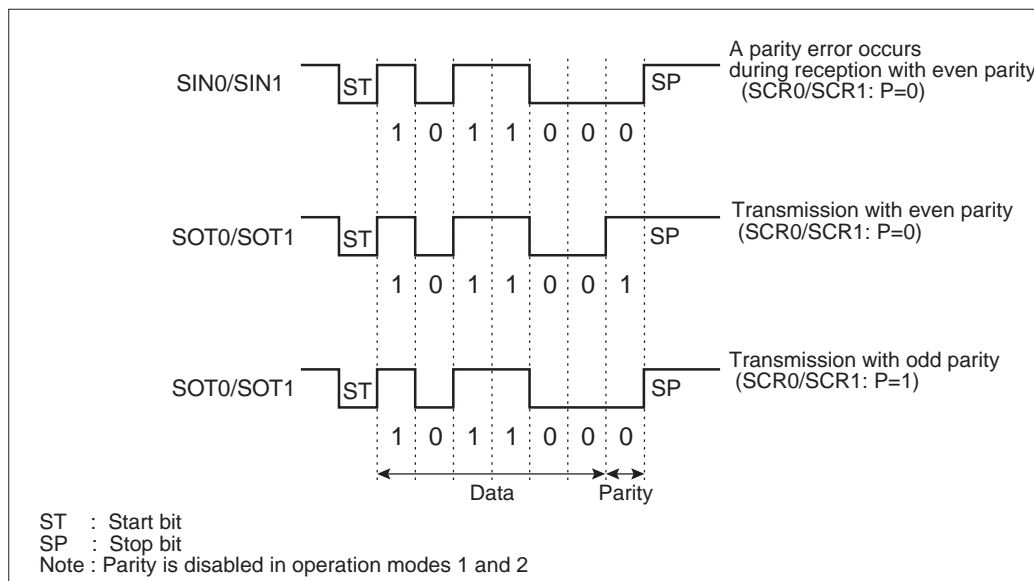
## ● Error detection

- In operation mode 0, parity, overrun and framing errors can be detected.
- In operation mode 1, overrun and framing errors can be detected but parity errors cannot be detected.

## ● Parity bit

Parity can only be used in operation mode 0 (asynchronous, normal mode). Whether to provide parity can be specified using the PEN bit of the control register (SCR0/SCR1). Even or odd parity can also be specified using the P bit of the control register (SCR0/SCR1). In operation mode 1 bit (asynchronous, multiprocessor mode) and operation mode 2 (synchronous, normal mode), parity cannot be used. Figure 20.7-2 shows both transmission and receive datas when parity bit is enabled.

**Figure 20.7-2 Transmission and reception datas when parity bit is enabled**



## 20.7.2 Operation in Clock Synchronous Mode (Operation Mode 2)

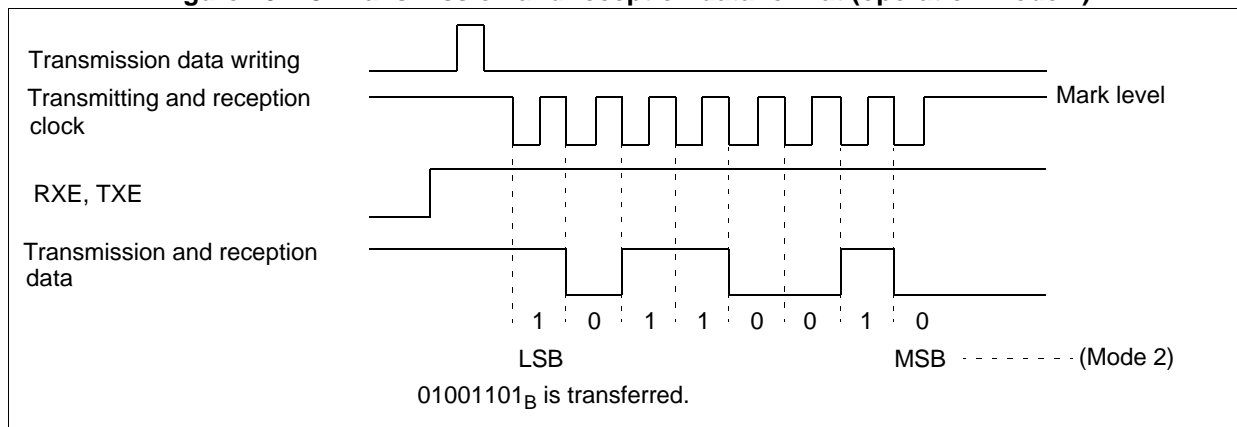
The clock synchronous method is used for UART operation mode 2.

### ■ Operation in Clock Synchronous Mode (Operation Mode 2)

#### ● Transmission and reception data format

In clock synchronous mode, 8-bit data is transmitted and received using the LSB first method, in which start and stop bits are not added. Figure 20.7-3 shows the transmission and reception data format in clock synchronous mode.

Figure 20.7-3 Transmission and reception data format (operation mode 2)



#### ● Clock supply

In clock synchronous mode (I/O extended serial), as many clocks as the number of transmission and reception bits must be supplied.

- When the internal clock (dedicated baud rate generator or internal timer) is selected, the data receiving synchronous clocks is generated automatically if data is transmitted.
- When the external clock is selected, confirm that the transmission side UART output data register (SODR0/SODR1) contains data (SSR0/SSR1: TDRE = 0). Then, clocks for just 1 byte must be supplied from outside.

The mark level (H) must be retained before transmission starts and after it is complete.

#### ● Error detection

Only overrun errors can be detected; parity and framing errors cannot be detected.

#### ● Initialization

The following shows the set values of each control register using the synchronous mode:

[Serial mode control register (SMR0/SMR1)]

MD1, MD0:10<sub>B</sub>

CS2, CS1, CS0:Specify clock input using the clock selector.

SCKE:1 for dedicated baud rate generator or internal timer  
0 for clock output and external clock (clock input)  
SOE:1 for transmission; 0 for reception only

[Serial control register (SCR0/SCR1)]

PEN:0

P, SBL, ADThese bits make no sense.

CL1 (8-bit data)

REC:0 (the error flag is cleared for initialization.)

RXE, TXE:At least one of the two bits is set to 1.

[Serial status register (SSR0/SSR1)]

RIE:1 when using interrupts; 0 when using no interrupts.

TIE:1 when using interrupts; 0 when using no interrupts.

## ● Starting communication

Write data to the serial output data register (SODR0/SODR1) to start communication. Temporary data must be written to SODR0/SODR1 to start communication for reception.

## ● Ending communication

The RDRF flag of the serial status register (SSR0/SSR1) is set to 1 when transmission or reception of a data frame is completed. During reception, check the overrun error flag bit (SSR0/SSR1: ORE) to see if communication is performing normally.



20.7.3 Bidirectional Communication Function (Normal Mode)

In operation mode 0 or 2, normal serial bidirectional communication (one-to-one connection) is available. Select operation mode 0 for asynchronous communication and operation mode 2 for clock synchronous communication.

■ Bidirectional Communication Function

The settings shown in Figure 20.7-4 are required to operate UART in normal mode (operation mode 0 or 2).

Figure 20.7-4 Settings for UART operation mode 0 or 2

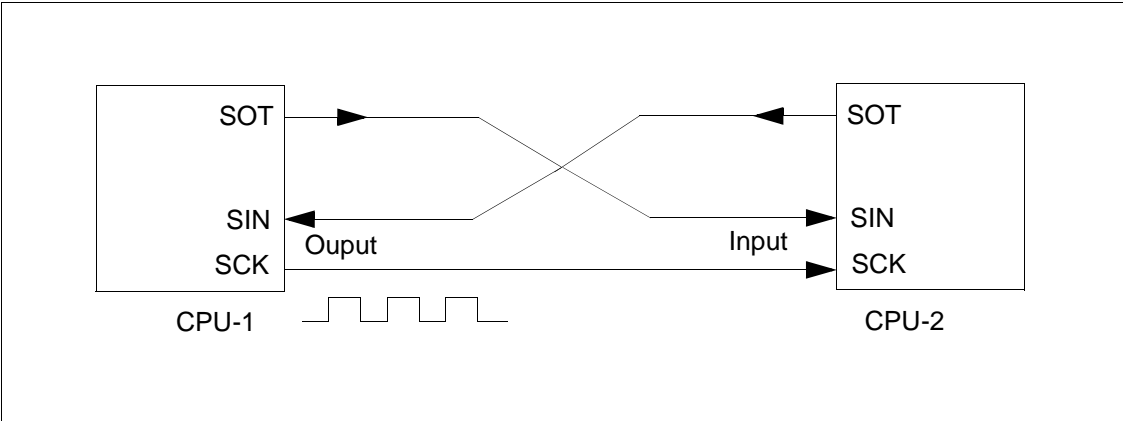
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR0/SCR1, SMR0/SMR1	PEN	P	SBL	CL	AD	REC	RXE	TXE	MD1	MD0	CS2	CS1	CS0	RST	SCKE	SOE
Mode 0 ⇒	⊙	⊙	⊙	⊙	X	0	⊙	⊙	0	0	⊙	⊙	⊙	X	⊙	⊙
Mode 2 ⇒	0	X	X	1	X	0	⊙	⊙	1	0	⊙	⊙	⊙	X	⊙	⊙
SSR0/SSR1, SIDR0/SIDR1, SODR0/SODR1	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set conversion data (during writing). Retain receive data (during reading).							
Mode 0 ⇒	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙								
Mode 2 ⇒	X	⊙	X	⊙	⊙	⊙	⊙	⊙								
DDR4 (UART0)									<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div> <div>△</div>							
DDR6 (UART1)									<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div> <div>△</div>							

⊙ : Bit used  
X : Bit not used  
1 : Set 1  
0 : Set 0  
△

● Inter-CPU connection

Figure 20.7-5 shows interconnect two CPU's.

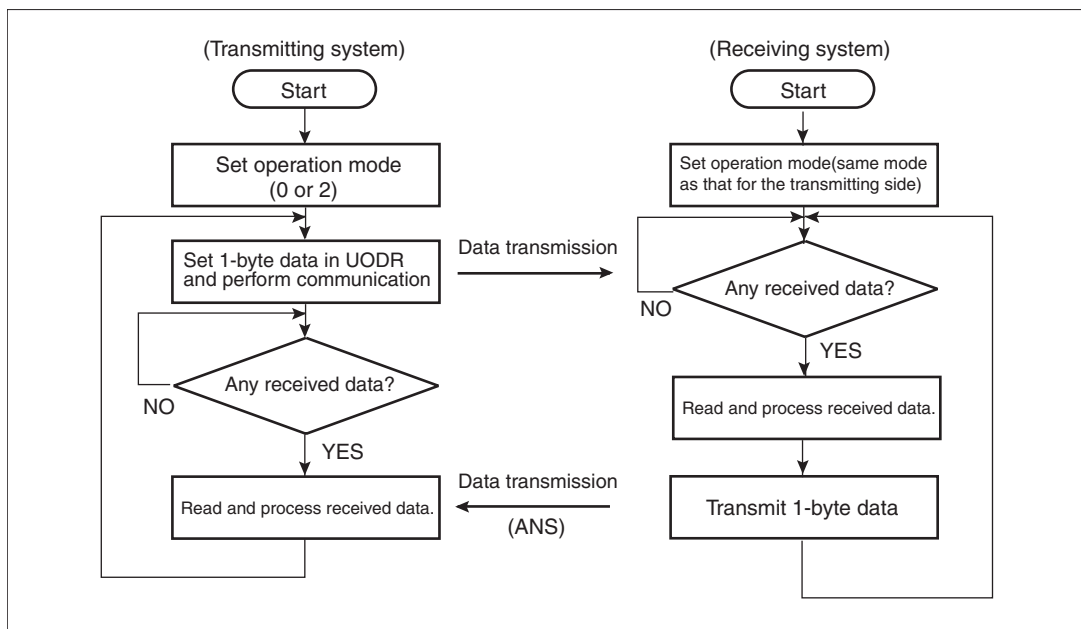
Figure 20.7-5 Connection example of UART bidirectional communication



● Communication procedure

Communication starts from the transmitting system at an optional timing when transmission data has been prepared. An ANS is returned periodically (byte by byte in this example) when the receiving system receives transmission data. Figure 20.7-6 shows an example of a bidirectional communication flowchart.

**Figure 20.7-6 Example of bidirectional communication flowchart**



### 20.7.4 Master-slave Communication Function (Multiprocessor Mode)

With UART, communication with multiple CPUs connected in master-slave mode is available in operation mode 1. However, UART can be used only from the master system.

#### ■ Master-slave Communication Function

The settings shown in Figure 20.7-7 are required to operate UART in multiprocessor mode (operation mode 1).

Figure 20.7-7 Settings for UART operation mode 1

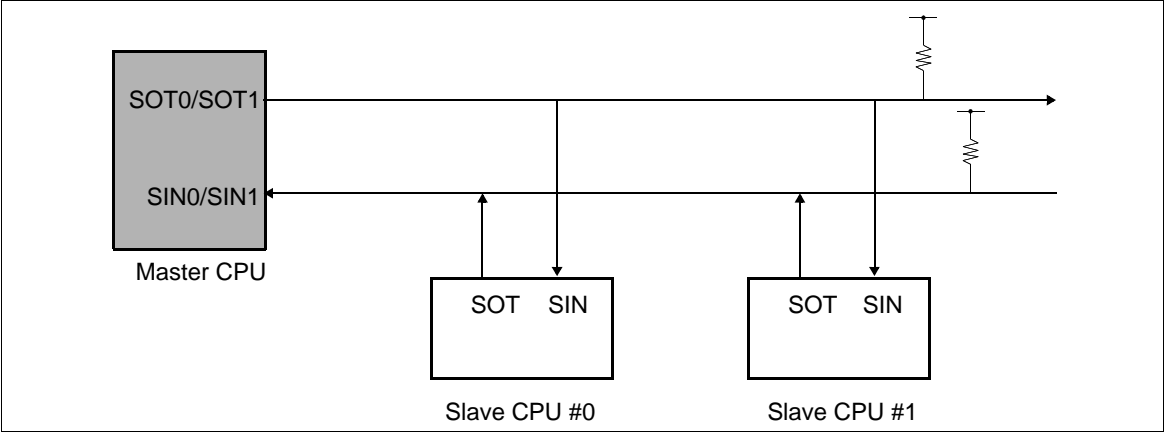
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR0/SCR1, SMR0/SMR1	PEN	P	SBL	CL	AD	REC	RXE	TXE	MD1	MD0	CS2	CS1	CS0	RST	SCKE	SOE
	0	X	⊙	1	⊙	0	⊙	⊙	0	0	⊙	⊙	⊙	X	⊙	⊙
SSR0/SSR1, SIDR0/SIDR1, SODR0/SOD1	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set transmission data (during writing). Retain receive data (during reading).							
	X	⊙	⊙	⊙	⊙	⊙	⊙	⊙								
DDR4 (UART0)									<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>							
DDR6 (UART1)									<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>							

⊙ : Bit used  
X : Bit not used  
1 : Set 1  
0 : Set 0  
△ : Set 0 to use an input pin

#### ● Inter-CPU connection

As shown in Figure 20.7-8 , a communication system consists of one master CPU and multiple slave CPUs connected to two communication lines. UART can be used only from the master CPU.

Figure 20.7-8 Connection example of UART master-slave communication



● Function selection

Select the operation mode and communication mode for master-slave communication as shown in Table 20.7-2.

**Table 20.7-2 Selection of the master-slave communication function**

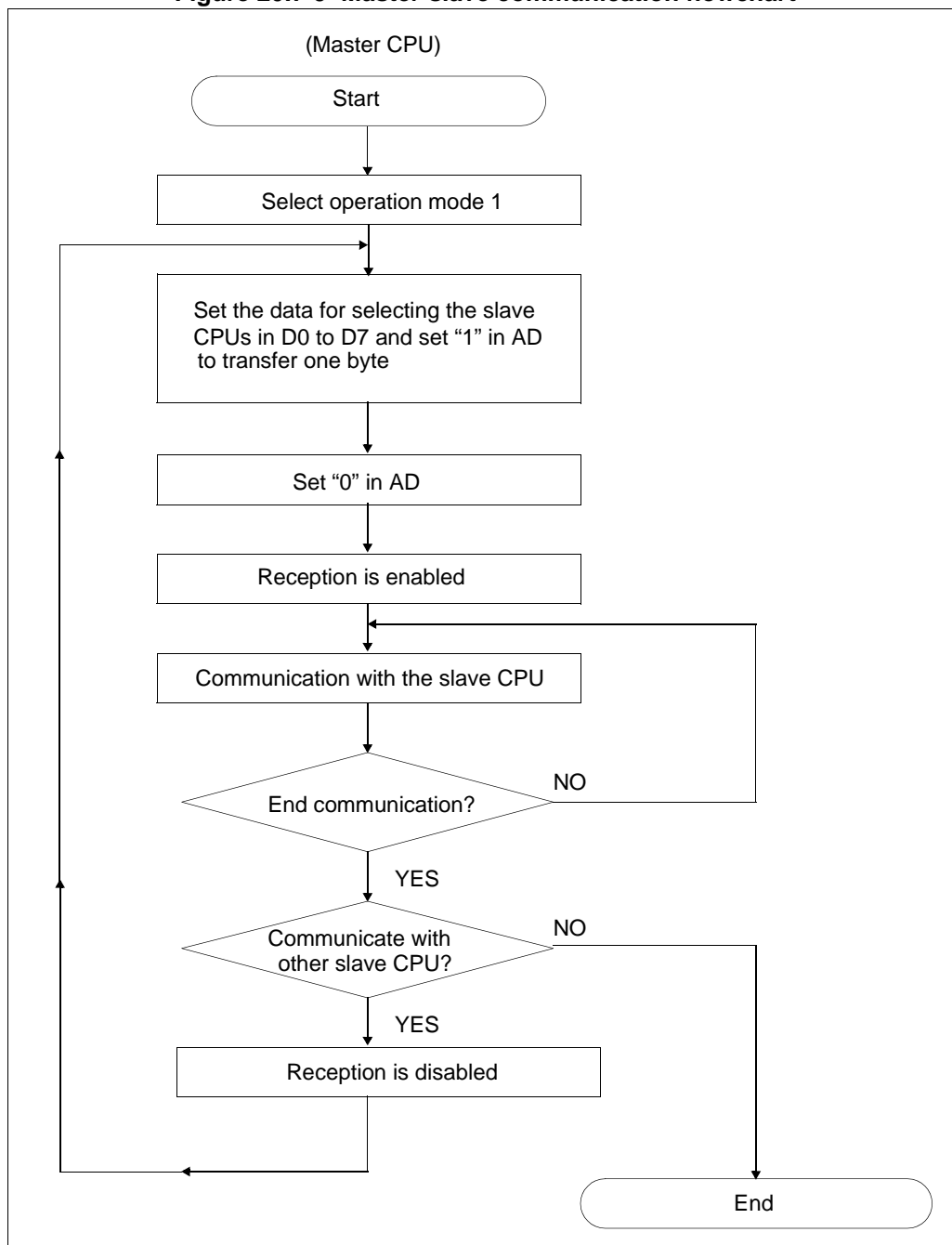
	Operation mode		Data	Parity	Synchroniza tion method	Stop bit
	Master CPU	Slave CPU				
Address transmission and reception	Operaiton Mode 1	—	AD =1 + 8-bit address	None	Asynchronous	1 or 2 bits
Data transmission and reception			AD =0 + 8-bit data			

● Communication procedure

When the master CPU transmits address data, communication starts. The AD bit in the address data is set to 1, and the communication destination slave CPU is selected. Each slave CPU checks the address data using a program. When the address data indicates the address assigned to a slave CPU, the slave CPU communicates with the master CPU (ordinary data).

Figure 20.7-9 shows a flowchart of master-slave communication (multiprocessor mode).

Figure 20.7-9 Master-slave communication flowchart



## 20.8 Usage Notes on UART

---

Notes on using UART are given below.

---

### ■ Notes on Using UART

#### ● Enabling operations

In UART, the serial control register (SCR0/SCR1) has both TXE (transmission) and RXE (reception) operation enable bits. Both transmission and reception operations must be enabled before the transfer starts because they have been disabled as the default value (initial value). The transfer can also be canceled by disabling its operation as required.

#### ● Operation mode setting

Set the operation mode while the system is not operating after the operation enable bit is set to disabled. If the mode is set during transmission or reception, the transmission or reception data is not guaranteed.

#### ● Synchronous mode

UART clock synchronous mode (operation mode 2) uses clock control (I/O extended serial) mode, in which start and stop bits are not added to the data.

#### ● Transmission interrupt enabling timing

The default (initial value) of the transmission data empty flag bit (SSR0/SSR1: TRE) is 1 (no transmission data and transmission data write enable state). A transmission interrupt request is generated as soon as the transmission interrupt requests are enabled (SSR0/SSR1: TIE = 1). Be sure to set the TIE flag to 1 after setting the transmission data.



# ***CHAPTER 21***

---

# ***ROM CORRECTION FUNCTION***

**This chapter describes the functions and operation of the ROM correction function.**

- 21.1 Overview of the ROM Correction Function
- 21.2 Block Diagram of ROM Correction Function
- 21.3 ROM Correction Function Registers
- 21.4 Operation of the ROM Correction Function
- 21.5 Example of Using ROM Correction Function



## 21.1 Overview of the ROM Correction Function

---

An instruction code to be read by the CPU is forcibly replaced with an INT9 instruction code (01<sub>H</sub>) when the corresponding address is equal to the value set in a program address detect register. A program patch application function can be implemented by processing with the INT #9 interrupt routine.

---

### ■ Program Address Detection Registers (x 2)

There are two program address detection registers (PADR0/PADR1), each is provided with an interrupt enable bit and interrupt flag.

### ■ ROM Correction Interrupts

When the interrupt enable bit is "1", the value set in the program address detection register is compared with the address. If the value matches the address, "1" is set in the interrupt flag bit and the instruction code to be read to the CPU is forcibly replaced with an INT9 instruction code.

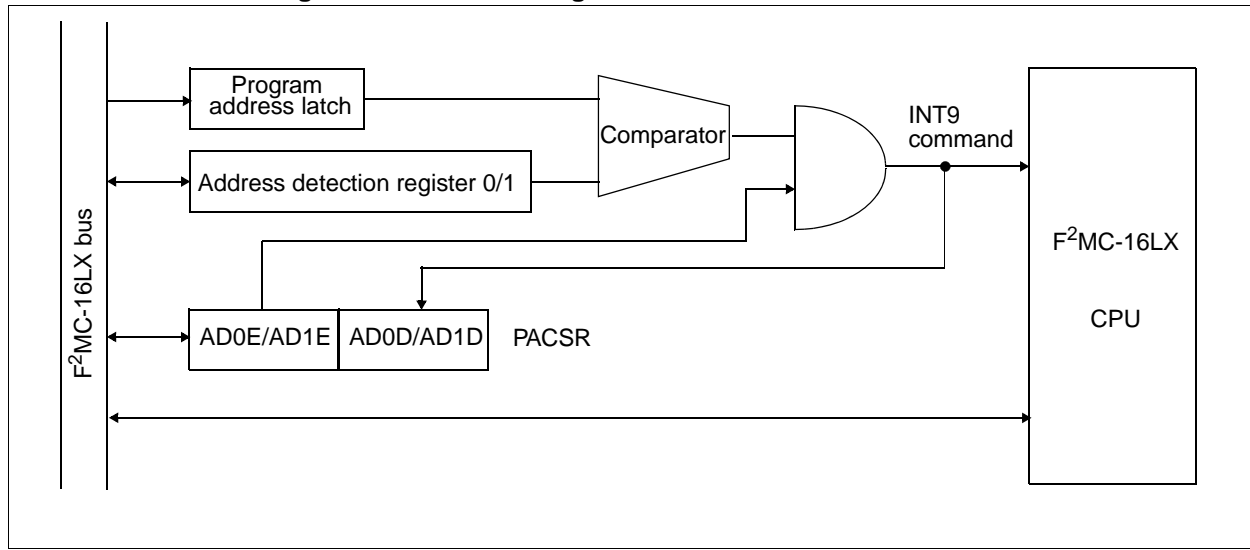
The interrupt flag bit is cleared to "0" by writing "0" to it using an instruction.

**MB90820B Series****21.2 Block Diagram of ROM Correction Function**

The block diagram of ROM correction function is shown as below.

■ Block Diagram of ROM Correction Function

Figure 21.2-1 Block diagram of ROM correction function



### 21.3 ROM Correction Function Registers

The section lists the ROM correction function registers.

#### ■ ROM Correction Function Registers

Figure 21.3-1 Registers of ROM Correction Function

Program Address Detection Register 0/1										
		Upper Byte		Middle Byte		Lower Byte				
Address :	1FF2 <sub>H</sub> /1FF1 <sub>H</sub> /1FF0 <sub>H</sub>	PADRH0		PADRM0		PADRL0				
Address :	1FF5 <sub>H</sub> /1FF4 <sub>H</sub> /1FF3 <sub>H</sub>	PADRH1		PADRM1		PADRL1				
Read/write ⇒		(R/W)		(R/W)		(R/W)				
Initial value ⇒		(XXXXXXXX <sub>B</sub> )		(XXXXXXXX <sub>B</sub> )		(XXXXXXXX <sub>B</sub> )				
Program Address Detection Control Status Register										
Address:	bit	7	6	5	4	3	2	1	0	Initial value
PACSR	009E <sub>H</sub>	—	—	—	—	AD1E	AD1D	AD0E	AD0D	XXXX0000 <sub>B</sub>
Read/write ⇒		(—)	(—)	(—)	(—)	(R/W)	(R/W)	(R/W)	(R/W)	

**MB90820B Series****21.3.1 Program Address Detection Register (PADR0/PADR1)**

The program address detection register (PADR0/PADR1) is a 24-bit register and used to store the address to be compared with internal address bus.

**■ Program Address Detection Register 0/1 (PADR0/PADR1)****Figure 21.3-2 Program address detection register**

Program Address Detection Register 0/1			
	Upper Byte	Middle Byte	Lower Byte
Address : 1FF2 <sub>H</sub> /1FF1 <sub>H</sub> /1FF0 <sub>H</sub>	PADRH0	PADRM0	PADRL0
Address : 1FF5 <sub>H</sub> /1FF4 <sub>H</sub> /1FF3 <sub>H</sub>	PADRH1	PADRM1	PADRL1
Read/write ⇒	(R/W)	(R/W)	(R/W)
Initial value ⇒	(XXXXXXXX <sub>B</sub> )	(XXXXXXXX <sub>B</sub> )	(XXXXXXXX <sub>B</sub> )

The value written to this register is compared with a target address. If the value matches the address, and the corresponding interrupt enable bit of the PACSR register is "1", the corresponding interrupt bit is set to "1" to request the CPU to generate an INT9 instruction. If the corresponding interrupt enable bit is "0", no operation is performed.

Table 21.3-1 lists the correspondence between the program address detection register and PACSR.

**Table 21.3-1 Correspondence between program address detection register and PACSR**

Program address detection register	Interrupt enable bit	Interrupt bit
PADR0	AD0E	AD0D
PADR1	AD1E	AD1D

## 21.3.2 Program Address Detection Control Status Register (PACSR)

The program address detection control status register (PACSR) is an 8-bit register and used to control the operation of ROM correction function.

### ■ Program Address Detection Control Status Register (PACSR)

Figure 21.3-3 Program address detection control status register

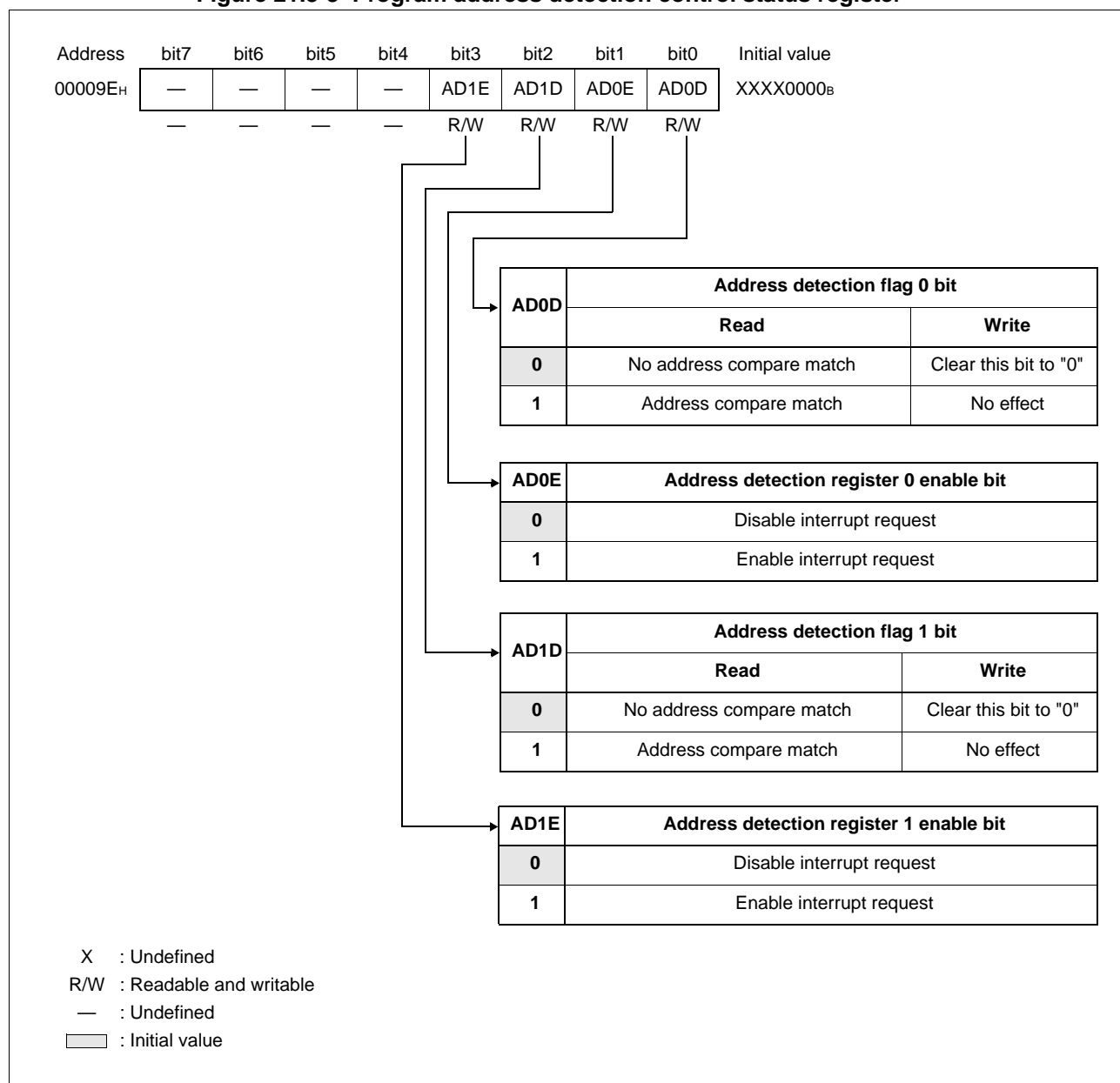


Table 21.3-2 Program address detection control status register

Bit name		Function
bit7 to bit4	Reserved bits	<ul style="list-style-type: none"> <li>Always write "0" to these bits.</li> </ul>
bit3	AD1E: Address detection register 1 enable bit	<ul style="list-style-type: none"> <li>PADR1 operation enable bit.</li> <li>When this bit is "1", the value set in the PADR1 register is compared with the address. If the two values are equal, an INT9 instruction is generated and the AD1D bit is set to "1".</li> </ul>
bit2	AD1D: Address detection flag 1 bit	<ul style="list-style-type: none"> <li>PADR1 address match detection bit.</li> <li>This bit is set to "1" to indicate that the value set in the PADR1 register matches the address. It is cleared to "0" by writing "0" to it. It is left unchanged by writing "1" to it.</li> </ul>
bit1	AD0E: Address detection register 0 enable bit	<ul style="list-style-type: none"> <li>PADR0 operation enable bit.</li> <li>When this bit is "1", the value set in the PADR0 register is compared to the address. If the two values are equal, an INT9 instruction is generated and the AD0D bit is set to "1".</li> </ul>
bit0	AD0D: Address detection flag 0 bit	<ul style="list-style-type: none"> <li>PADR0 address match detection bit.</li> <li>This bit is set to "1" to indicate that the value set in the PADR0 register is equal to the address. It is cleared to "0" by writing "0" to it. It is left unchanged by writing "1" to it.</li> </ul>

## 21.4 Operation of the ROM Correction Function

---

If the program counter specifies the same address as that in program address detection register (PADR), the INT9 instruction is executed. The ROM correction function can be done by processing the INT9 instruction routine.

---

### ■ Operation of the ROM Correction Function

An instruction code to be read by the CPU is forcibly replaced with an INT9 instruction code (01<sub>H</sub>) when the corresponding address is equal to the value set in an address detection register. Therefore, the CPU executes the INT9 instruction when executing the set instruction.

A program patch application function can be implemented by processing with the INT #9 interrupt routine.

There are two address detection registers, of which each is provided with an interrupt enable bit and interrupt flag. When the address is equal to the value set in the address detection register, and the interrupt enable bit is "1", assume the following: the interrupt flag is set to "1", and the instruction code to be read by the CPU is forcibly replaced with the INT9 instruction code. The interrupt flag is cleared to "0" by writing "0" to it using an instruction.

---

#### Notes:

- The address match detection function fails if an address later than the first byte of the instruction is set in the address detection register. The value in the set address is replaced with "01<sub>H</sub>", so a wrong instruction is executed or an invalid address is accessed. Before changing the value set in the address detection register, set the interrupt enable bit to "0". If data is written while the interrupt enable bit is "1", the address may be wrongly detected during writing, causing a malfunction.
  - The program address detection register exists at 001FF0<sub>H</sub> to 001FF5<sub>H</sub> that overlaps with the RAM area of MB90F828B, preventing RAM access when using this feature on MB90F828B.
-

**MB90820B Series****21.5 Example of Using ROM Correction Function**

This section contains example of using the address match detection function.

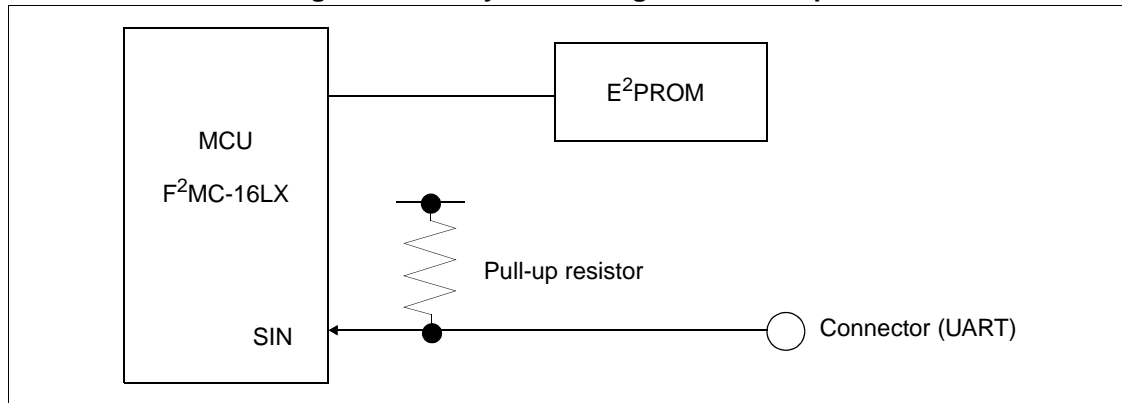
**■ System Configuration****Figure 21.5-1 System configuration example****■ E²PROM Memory Map**

Table 20.5-1 lists the E²PROM memory map.

**Table 21.5-1 E²PROM memory map**

Address	Meaning
0000 <sub>H</sub>	Number of bytes for patch program No. 0 (0 for no program error)
0001 <sub>H</sub>	Bit 7 to bit 0 of program address No. 0
0002 <sub>H</sub>	Bit 15 to bit 8 of program address No. 0
0003 <sub>H</sub>	Bit 24 to bit 16 of program address No. 0
0004 <sub>H</sub>	Number of bytes for patch program No. 1 (0 for no program error)
0005 <sub>H</sub>	Bit 7 to bit 0 of program address No. 1
0006 <sub>H</sub>	Bit 15 to bit 8 of program address No. 1
0007 <sub>H</sub>	Bit 24 to bit 16 of program address No. 1
to 0010 <sub>H</sub> <sup>+</sup> Number of bytes for patch program No. 0	Original of patch program No. 0



## ■ Initial State

The contents of E<sup>2</sup>PROM are all 0's.

## ■ If a Program Error Occurs

The original of a patch program and its address is transferred to the MCU via the connector (UART). The MCU writes the information to E<sup>2</sup>PROM.

## ■ Reset Sequence

After the reset sequence is completed, the MCU reads the value of E<sup>2</sup>PROM. If the number of bytes for the patch program is not 0, the MCU reads the original patch program and writes it to RAM. Then, the MCU sets the program address to PADR0 or PADR1 and enables the program to run. The first address of the program written to RAM is saved in RAM as specified for each address detection register.

## ■ INT9 Interrupt

During execution of an interrupt routine, control checks the interrupt flag for an address in which an interrupt was enabled and branches to the corresponding program. The information stacked by the interrupt is deleted. The interrupt flag is also cleared.

Figure 21.5-2 System configuration example

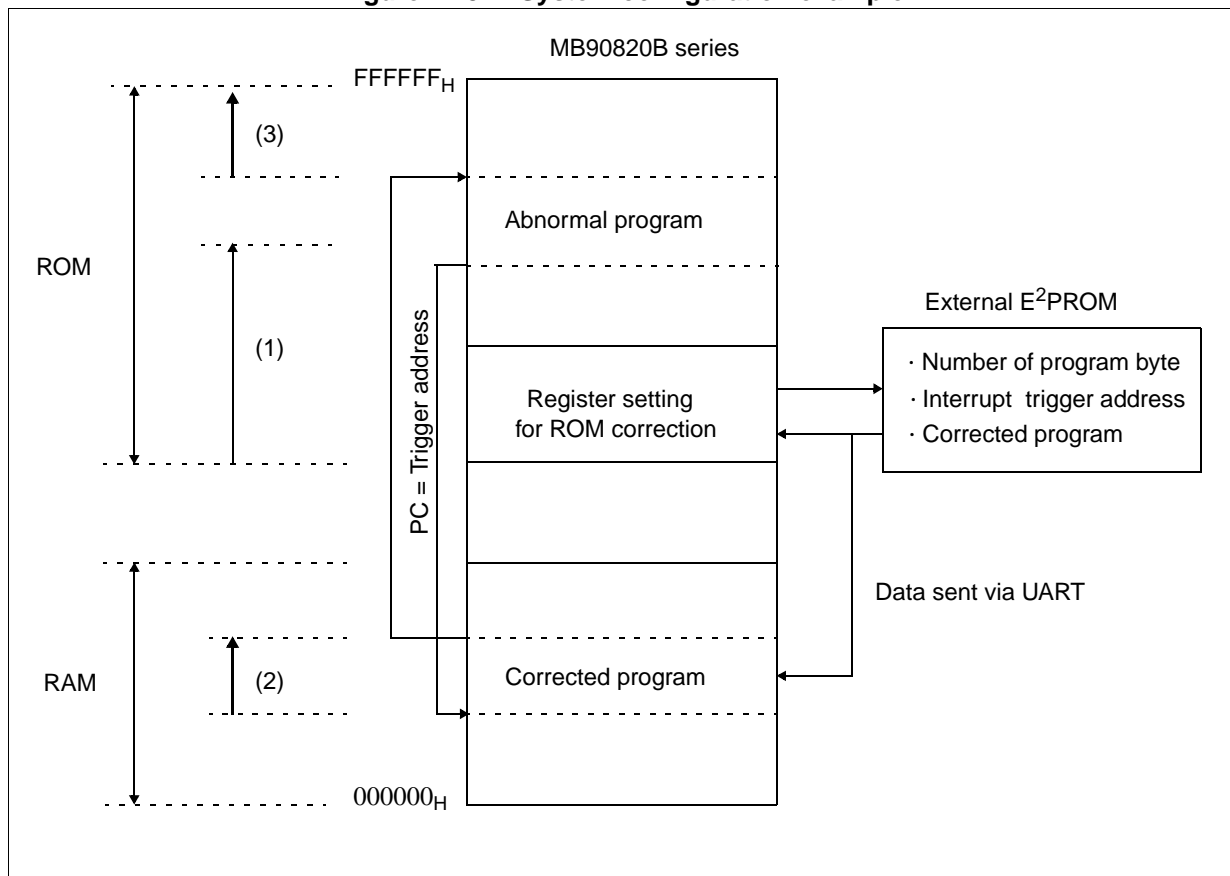
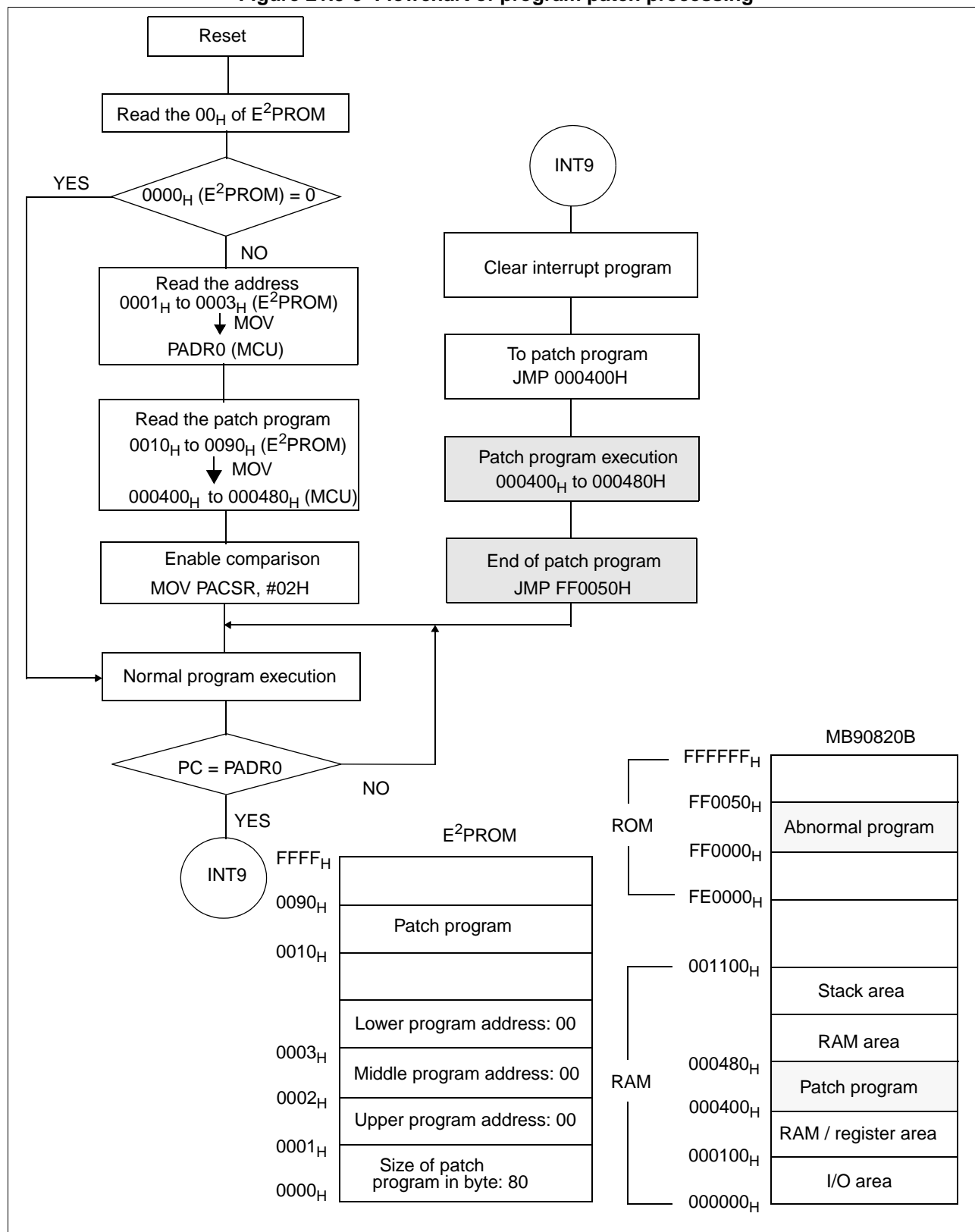


Figure 21.5-3 Flowchart of program patch processing





# **CHAPTER 22**

---

## ***ROM MIRRORING FUNCTION SELECTION MODULE***

**This chapter explains the function and operation of the MB90820B series ROM mirroring function selection module.**

- 22.1 Overview of the ROM Mirroring Function Selection Module
- 22.2 ROM Mirroring Function Selection Register (ROMM)

## 22.1 Overview of the ROM Mirroring Function Selection Module

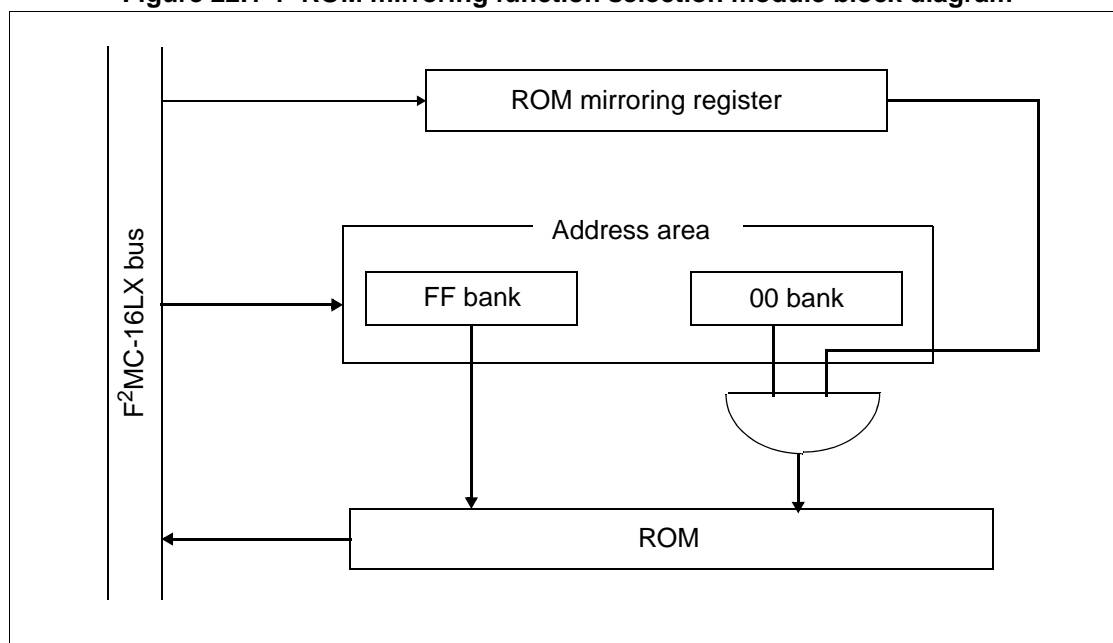
The ROM mirroring function selection module can access bank FF located in ROM from bank 00 by setting the register.

### ■ ROM Mirroring Function Selection Module Register

ROM Mirror Function Selection Register									Initial value	
Address:	bit	15	14	13	12	11	10	9	8	
ROMM	0006F <sub>H</sub>	—	—	—	—	—	—	—	MI	XXXXXXXX <sub>1B</sub>
	Read/write ⇒	(—)	(—)	(—)	(—)	(—)	(—)	(—)	(W)	

### ■ ROM Mirroring Function Selection Module Block Diagram

Figure 22.1-1 ROM mirroring function selection module block diagram



MB90820B Series

22.2 ROM Mirroring Function Selection Register (ROMM)

The ROM mirroring function selection register (ROMM) is used to enable mirroring function.

■ ROM Mirroring Function Selection Register (ROMM)

Figure 22.2-1 ROM mirroring function selection register (ROMM)

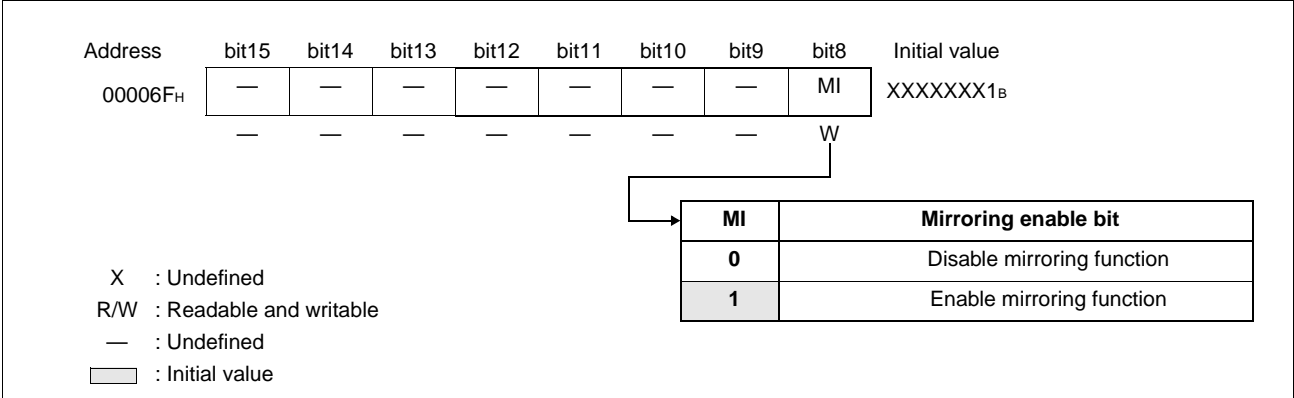


Table 22.2-1 Function of ROM mirroring function selection register (ROMM)

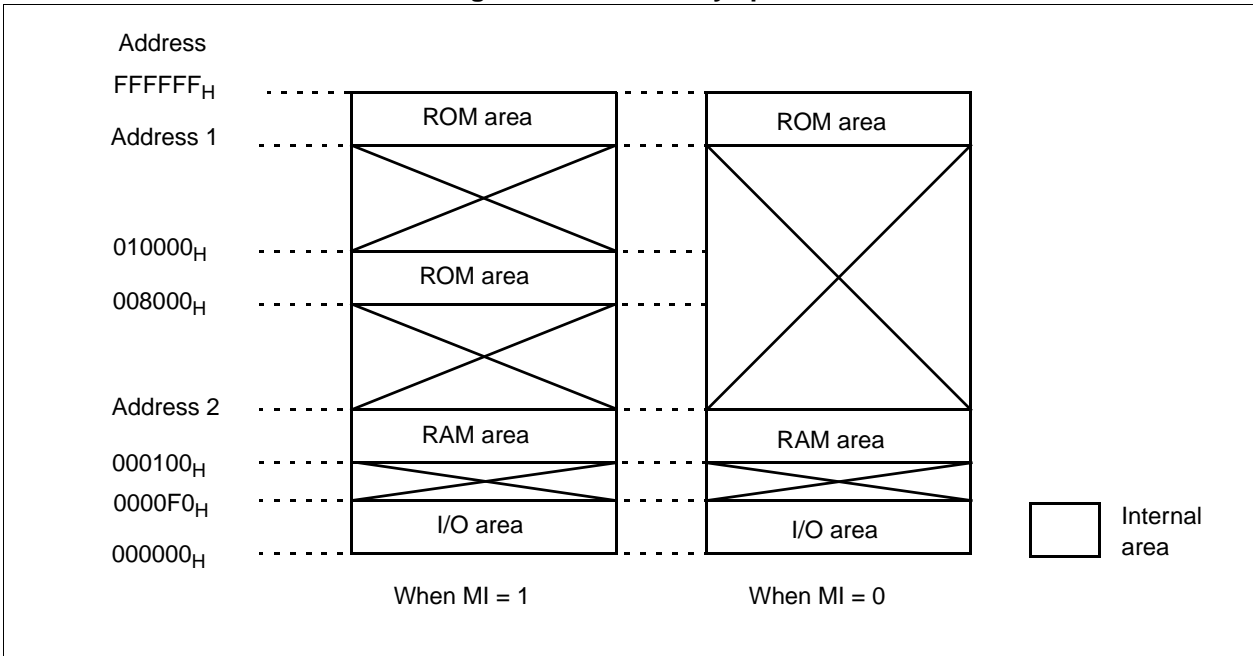
Bit name		Function
bit15 to bit9	Unfind bits	<ul style="list-style-type: none"><li>The read value is undefined.</li><li>Always write "0" to these bits.</li></ul>
bit8	MI: Mirroring enable bit	<ul style="list-style-type: none"><li>When "1" has been written to this bit, the ROM data in bank FF can be read from bank 00.</li><li>When "0" has been written to this bit, the function is disabled in bank 00. This bit is write only.</li></ul>

Note :

Bank 00 accesses FF8000<sub>H</sub> to FFFFFFF<sub>H</sub> from 008000<sub>H</sub> to 00FFFF<sub>H</sub>. Therefore, FFF000<sub>H</sub> to FF7FFF<sub>H</sub> cannot be accessed even by selecting the ROM mirroring function.

	MB90822B	MB90823B	MB90F822B	MB90F823B	MB90F828B	MB90V820B
Address 1	FF0000 <sub>H</sub>	FE0000 <sub>H</sub>	FF0000 <sub>H</sub>	FE0000 <sub>H</sub>	FE0000 <sub>H</sub>	FE0000 <sub>H</sub>
Address 2	0010FF <sub>H</sub>	0010FF <sub>H</sub>	0010FF <sub>H</sub>	0010FF <sub>H</sub>	0020FF <sub>H</sub>	0040FF <sub>H</sub>

Figure 22.2-2 Memory space



# **CHAPTER 23**

---

## ***512K / 1024K BIT FLASH MEMORY***

**The following explains the functions and operations of the 512K / 1024K bit flash memory.**

- 23.1 Overview of the 512K / 1024K Bit Flash Memory
- 23.2 512K / 1024K Bit Flash Memory Sector Configuration
- 23.3 Flash Memory Control Status Register (FMCS)
- 23.4 Method of Starting the Automatic Algorithm in Flash Memory
- 23.5 Verifying Automatic Algorithm Execution Status
- 23.6 Detailed Explanation on the Flash Memory Write/Delete
- 23.7 Flash Security Feature



## 23.1 Overview of the 512K / 1024K Bit Flash Memory

Three methods of data writing/deleting to the 512K/1024K bit flash memory are provided:

Parallel writer (MODEL1890A made by Minato Electronics)

Serial dedicated writer (AF-200 made by Yokogawa Digital Computer)

Write/delete operation by program execution

This chapter provides an explanation for the above item "3. Write/delete operation by program execution".

### ■ Overview of the 512K/1024K Bit Flash Memory

The 512K bit flash memory is allocated in the FF bank on the CPU memory map while 1024K bit flash memory is allocated in FE and FF bank. The function of the flash memory interface circuit enables the read/access or program access from the CPU to the flash memory, same as the mask ROM. The write/delete operation to the flash memory can be executed through the flash memory interface circuit by executing an instruction issued from the CPU. Therefore, the flash memory mounted can be rewritten under the control of the internal CPU, so that the program or data can be upgraded or updated more efficiently. However, no selector operation such as the enable sector protect can be used.

### ■ Characteristics of the 512K / 1024K Bit Flash Memory

- 512K Bit: 64K words × 8 bits/32K words × 16 bits (16K+8K+8K+32K) sector configuration
- 1024K Bit: 128K words × 8 bits/64K words × 16 bits (64K+16K+8K+8K+32K) sector configuration
- Automatic program algorithm (same as the Embedded Algorithm : MBM29F400TA)
- Installation of the deletion temporary stop/delete restart function
- Write/delete completion detected by the data polling or toggle bit
- Write/delete completion detected by the CPU interrupt
- Compatibility with the JEDEC standard-type command
- Each sector deletion can be executed (Sectors can be freely combined)
- Number of write/delete operations 10,000 times guaranteed
- Flash security feature

"Embedded Algorithm" is the trademark of Advanced Micro Device

### ■ Procedure for Writing/Deleting the Data to the Flash Memory

The write/delete operation of the flash memory cannot be executed simultaneously. In executing the data write/delete operation in the flash memory, only the write operation can be executed, by copying a program on the flash memory to RAM and executing the program.

### ■ Register on the Flash Memory

Figure 23.1-1 Flash memory control status register (FMCS)

bit	7	6	5	4	3	2	1	0	Initial value
Address:0000AE <sub>H</sub>	INTE	RDYINT	WE	RDY	Reserved	Reserved	Reserved	Reserved	000x0000 <sub>B</sub>
Read/write	(R/W)	(R/W)	(R/W)	(R)	—	—	—	—	

**MB90820B Series****23.2 512K / 1024K Bit Flash Memory Sector Configuration**

Figure 23.2-1 and figure 23.2-2 shows the sector configuration in the 512K bit flash memory. The address indicated in figure 23.2-1 and figure 23.2-2 is classified into the upper address and lower address of each sector.

**■ Sector Configuration**

When accessing the 512Kbit flash memory from the CPU, four sector addresses, SA0 to SA3, are allocated in the FF bank register.

**Figure 23.2-1 512K Bit Flash Memory Sector Configuration**

Flash memory	CPU address	*Writer address
SA3 (16 Kbytes)	FFFFFF <sub>H</sub>	7FFFF <sub>H</sub>
	FFC000 <sub>H</sub>	7C000 <sub>H</sub>
SA2 (8 Kbytes)	FFBFFF <sub>H</sub>	7BFFF <sub>H</sub>
	FFA000 <sub>H</sub>	7A000 <sub>H</sub>
SA1 (8 Kbytes)	FF9FFF <sub>H</sub>	79FFF <sub>H</sub>
	FF8000 <sub>H</sub>	78000 <sub>H</sub>
SA0 (32 Kbytes)	FF7FFF <sub>H</sub>	77FFF <sub>H</sub>
	FF0000 <sub>H</sub>	70000 <sub>H</sub>

When accessing the 1024Kbit flash memory from the CPU, five sector addresses, SA0 to SA4, are allocated in the FE and FF bank register.

**Figure 23.2-2 1024K Bit Flash Memory Sector Configuration**

Flash memory	CPU address	*Writer address
SA4 (16 Kbytes)	FFFFFF <sub>H</sub>	7FFFF <sub>H</sub>
	FFC000 <sub>H</sub>	7C000 <sub>H</sub>
SA3 (8 Kbytes)	FFBFFF <sub>H</sub>	7BFFF <sub>H</sub>
	FFA000 <sub>H</sub>	7A000 <sub>H</sub>
SA2 (8 Kbytes)	FF9FFF <sub>H</sub>	79FFF <sub>H</sub>
	FF8000 <sub>H</sub>	78000 <sub>H</sub>
SA1 (32 Kbytes)	FF7FFF <sub>H</sub>	77FFF <sub>H</sub>
	FF0000 <sub>H</sub>	70000 <sub>H</sub>
SA0 (64 Kbytes)	FEFFFF <sub>H</sub>	6FFFF <sub>H</sub>
	FE0000 <sub>H</sub>	60000 <sub>H</sub>

\* Writer address

The writer address is equivalent to the CPU address when writing the data to the flash memory using the parallel writer. If the write/delete operation is executed using the general-purpose writer, the write/delete operation is executed using this address.

## 23.3 Flash Memory Control Status Register (FMCS)

Figure 22.3-1 shows the function of the flash memory control status register (FMCS).

### Flash Memory Control Status Register (FMCS)

Figure 23.3-1 Flash memory control status register (FMCS)

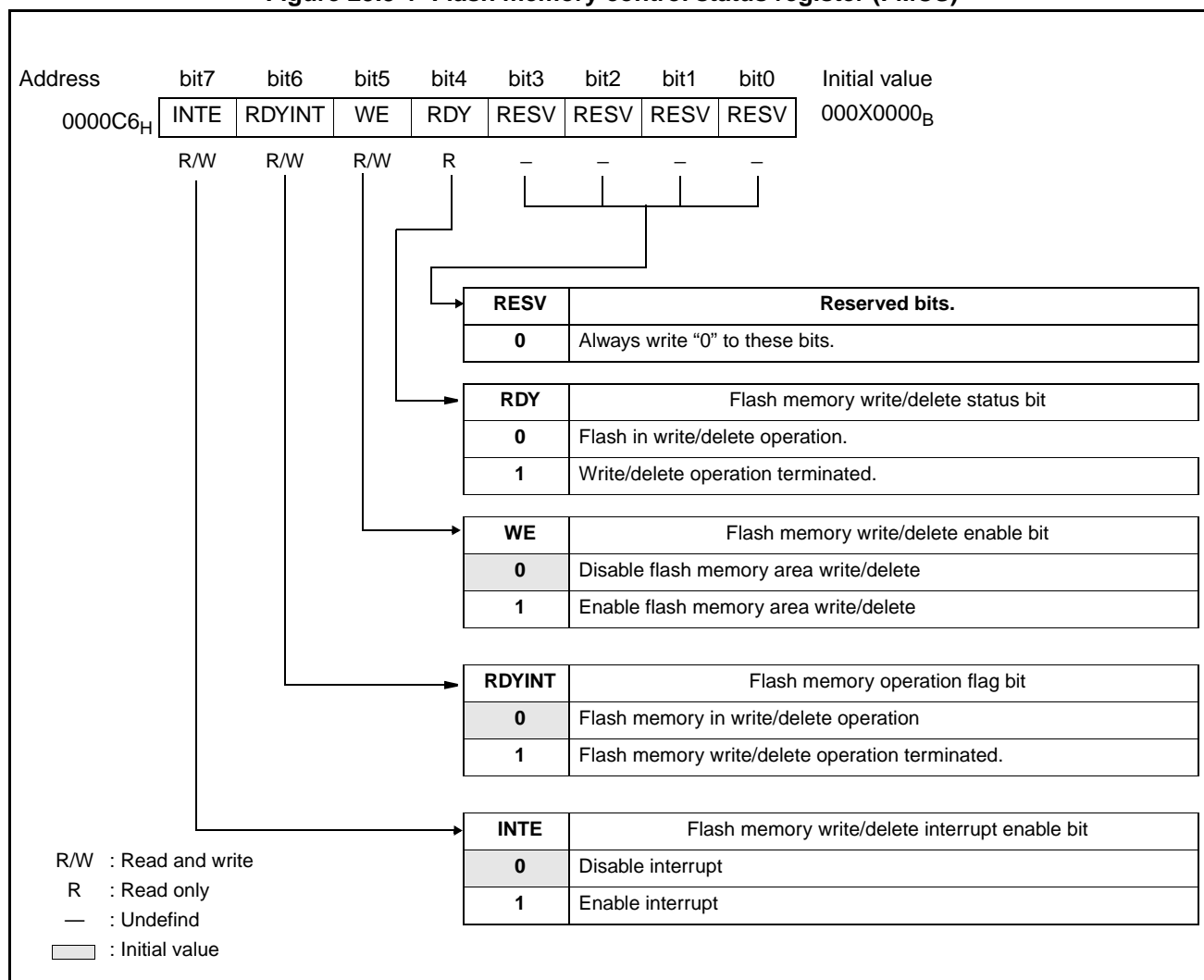
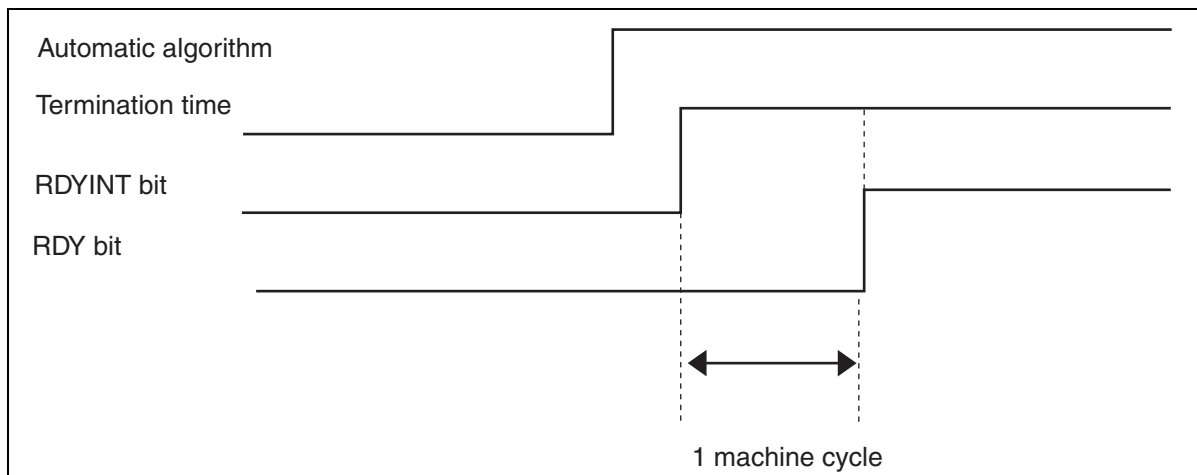


Table 23.3-1 Function of flash memory control status register (FMCS)

Bit name		Function
bit7	INTE: Flash memory interrupt/delete Interrupt Enable bit	<p>This bit generates an interrupt to the CPU when the write/delete operation to the flash memory is terminated.</p> <p>When the INTE bit is "1" and the RDYINT bit is "1", an interrupt is generated and sent to the CPU. If the INTE bit is "0", no interrupt is generated:</p> <p>0: Interrupt disabled when the write/delete operation is terminated. 1: Interrupt enabled when the write/delete operation is terminated.</p>
bit6	RDYINT: Flash memory operation flag bit	<p>This bit indicates the flash memory operating status.</p> <p>After the write/delete to the flash memory is terminated, this bit is set to "1". While this bit is "0" after the end of write/delete operation to the flash memory, the flash memory cannot be written or deleted. After the write/delete operation is terminated and this bit is set to "1", the flash memory can be written or deleted. This bit is cleared to "0" by writing "0" and the writing of "1" is ignored. At the termination time of the automatic algorithm in the flash memory (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), this bit is set to "1". While using the read modify write (RMW) instruction, "1" can be read at any time.</p> <p>0: During the write/delete operation 1: Write/delete operation terminated (An interrupt request is generated)</p>
bit5	WE: Flash memory write/delete enable bit	<p>This bit is the write enable bit for the flash memory area.</p> <p>When this bit is "1", the write instruction after issuing command sequence to FF bank (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory") is equivalent to writing to the flash memory area. When this bit is "0", no write/delete signal is generated. This bit is used when the flash memory write/delete command is started.</p> <p>0: Flash memory write/delete disabled 1: Flash memory write/delete enabled</p>
bit4	RDY: Flash memory write/delete status bit	<p>This bit is the flash memory write/delete permission bit.</p> <p>While this bit is "0", the write/delete cannot be executed to the flash memory. Even in this state, however, suspend commands such as the read/reset command and the sector deletion temporary stop can be accepted.</p> <p>0: During the write/delete operation 1: Write/delete operation terminated (Next data write/delete operation permitted)</p>
bit3 to bit0	RESV: Reserved bits	Always write "0" to these bits.

Note :

The RDYINT and RDY bits do not change at the same time. Create a program to determine the termination of write/delete operation using either of bits.



**MB90820B Series****23.4 Method of Starting the Automatic Algorithm in Flash Memory**

There are four types of commands for starting the automatic algorithm in the flash memory, i.e., the read/reset command, write command, and chip deletion command. In addition, the sector deletion command can be temporarily stopped and restarted.

**■ Command Sequence Table**

Table 23.4-1 lists the commands to be used for writing/deleting the data to the flash memory. All the data is written to the command register in units of bytes, though it should be accessed and written in units of words.

In this case, the data in the upper bytes written in units of word is ignored.

**Table 23.4-1 Command Sequence Table**

Command sequence	Bus write access	1st bus write cycle		2nd bus write cycle		3rd bus write cycle		4th bus write cycle		5th bus write cycle		6th bus write cycle	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Read/reset*	1	YYYYXX <sub>H</sub>	XXF0 <sub>H</sub>	-	-	-	-	-	-	-	-	-	-
Read/reset*	4	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	YYYYAA <sub>H</sub>	XXF0 <sub>H</sub>	RA	RD	-	-	-	-
Write	4	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	YYYYAA <sub>H</sub>	XXA0 <sub>H</sub>	PA	PD	-	-	-	-
Chip erase	6	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	YYYYAA <sub>H</sub>	XX80 <sub>H</sub>	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	YYYYAA <sub>H</sub>	XX10 <sub>H</sub>
Sector erase	6	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	YYYYAA <sub>H</sub>	XX80 <sub>H</sub>	YYYYAA <sub>H</sub>	XXAA <sub>H</sub>	YYY554 <sub>H</sub>	XX55 <sub>H</sub>	SA	XX30 <sub>H</sub>
Sector erase suspend	Entering address YYYYXXH and data "XXB0H" suspends during sector erasing.												
Sector erase resume	Entering address YYYYXXH and data "XX30H" resumes suspended sector erasing.												

RA: Read address

PA: Write address

SA: Sector address (specify an arbitray address in sector)

RD: Read data

PD: Write data

YYY: Upper 12 bits of an arbitrary address in the flash memory area

\*: Both of the two types of read/reset command can reset the flash memory to read mode

**Note :**

Addresses in the table are the values in the CPU memory map. All addresses and data are hexadecimal values. However, "X" is an arbitrary value.

## 23.5 Verifying Automatic Algorithm Execution Status

The flash memory contains the hardware for posting the internal flash memory operating status or the flash memory operation completion, because the automatic algorithm executes the sequence of data writing/deleting procedures. This automatic algorithm can verify the internal flash memory operating status, depending on the following hardware sequence.

### ■ Hardware Sequence Flag

The hardware sequence flag consists of the four flag bits, DQ7, DQ6, DQ5, and DQ3. These flag bits have the data polling flag (DQ7) function, toggle bit flag (DQ6) function, time limit exceeded flag (DQ5) function, and sector deletion timer flag (DQ3) function, respectively. These functions can verify whether the write/chip sector deletion is terminated or whether the deletion code write is valid.

The hardware sequence flag can be referred by accessing/reading the address of the target sector in the flash memory, after setting the command sequence (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"). Table 23.5-1 indicates the hardware sequence flag bit allocation.

**Table 23.5-1 Hardware Sequence Flag Bit Allocation**

Bit no.	7	6	5	4	3	2	1	0
Hardware sequence flag	DQ7	DQ6	DQ5	-	DQ3	-	-	-

It can be determined whether automatic write/chip sector deletion is being performed, depending on the end of write processing, by checking the hardware sequence flag or the RDY bit in the flash memory control status register (FMCS). After the write/delete operation is terminated, the flash memory is returned to the read/reset status. To actually create a program, it should be verified whether automatic write/delete operation is terminated, depending on any flag, and the next operation such as the data reading should be executed. Also, it can be verified whether the second sector deletion code write command or later commands are valid, depending on the hardware sequence flag. The following explains the hardware sequence flags. Table 23.5-2 lists the hardware sequence flag functions.

Table 23.5-2 Hardware Sequence Flag Function List

State		DQ7	DQ6	DQ5	DQ3
Status transition during normal operation	Write operation --> Write completion (when the write address is specified)	$\overline{DQ7}$ --> DATA:7	Toggle --> DATA:6	0 --> DATA:5	0 --> DATA:3
	Chip/sector deletion operation --> Deletion completion	0 --> 1	Toggle --> Stop	0 --> 1	1
	Sector deletion wait --> Deletion start	0	Toggle	0	0 --> 1
	Deletion processing --> Sector deletion temporary stop (sector being deleted)	0 --> 1	Toggle --> 1	0	1 --> 0
	Sector deletion temporary stop --> Deletion restart (sector being deleted)	1 --> 0	1 --> Toggle	0	0 --> 1
	While the sector deletion is being temporarily stopped --> (sector not being deleted)	DATA:7	DATA:6	DATA:5	DATA:3
Abnormal operation	Write operation	$\overline{DQ7}$	Toggle	1	0
	Chip/sector deletion operation	0	Toggle	1	1



## **23.5.1 Data Polling Flag (DQ7)**

---

The data polling flag (DQ7) indicates whether the automatic algorithm is being executed or has been terminated, using the data polling function. Table 23.5-3 shows the data polling flag status transition.

---

### **■ When the Write Operation is Executed.**

When the read/access is executed during automatic write algorithm execution, the flash memory outputs the reverse data of bit 7 in the last-written data, irrespective of the specified address. When the read/access is executed at the end of the automatic write algorithm, the flash memory outputs the data of bit 7 in the specified read address.

### **■ When the Chip/Sector Deletion Operation is Executed.**

When the read/access is executed during the chip/sector deletion algorithm execution, the flash memory outputs "0" from the sector being deleted, or irrespective of the specified address during the chip deletion. Similarly, the flash memory outputs "1" at the end of chip/sector deletion algorithm.

### **■ When the Sector Deletion Temporary Stop is Executed.**

When the read/access is executed while executing the sector deletion temporary stop, the flash memory outputs "1" if the specified address is the sector being deleted. However, the flash memory outputs the data of bit 7 (DATA:7) for the specified read address, if the specified address is not the sector being deleted. By referring this together with the toggle bit flag (DQ6), it can be determined whether the current sector is in the temporary stop state or which sector is being deleted.

---

#### **Note :**

When the automatic algorithm is started, the read/access to the specified address is ignored. As for the data reading, the end of data polling flag (DQ7) is posted, and then other data bit can be output. Therefore, the data read operation after the end of the automatic algorithm should be executed next to the read/access after verifying the end of data polling flag.

---

Table 23.5-3 shows the data polling flag status transition.

**Table 23.5-3 Data Polling Flag Status Transition**

- Status transition during normal operation

Operating status	Write operation --> Completion	Chip/sector deletion -->Completion	Sector deletion wait-->Start	Sector deletion -->Deletion temporary stop (Sector being deleted)	Sector deletion temporary stop -->Restart (Sector being deleted)	During the sector deletion temporary stop (Sector not being deleted)
DQ7	$\overline{\text{DQ7}}$ -->DATA:7	0-->1	0	0-->1	1-->0	DATA:7

- Status transition during abnormal operation

Operating status	Write operation	Chip/sector deletion operation
DQ7	$\overline{\text{DQ7}}$	0

## **23.5.2 Toggle Bit Flag (DQ6)**

---

The toggle bit flag (DQ6) specifies whether the automatic algorithm is being executed or has been terminated, using the toggle bit function, the same as the data polling flag.

Table 23.5-4 shows the toggle bit flag status transition.

---

### **■ When the Write Operation or Chip/Sector Deletion Operation is Executed.**

When the continuous read/access is executed during the automatic write algorithm or chip/sector deletion algorithm execution, the flash memory outputs the toggle status, in which "1" and "0" are alternately output for each read operation, irrespective of the specified address. If the continuous read/access is executed at the end of the automatic write algorithm or chip/sector deletion algorithm, the flash memory stops the toggle operation in bit 6 and outputs the data of bit 6 (DATA:6) in the specified read address.

### **■ When the Sector Deletion Temporary Stop is Executed.**

When the read/access is executed while executing the sector deletion temporary stop, the flash memory outputs "1" if the specified address belongs to the sector being deleted. The flash memory outputs the data of bit 6 (DATA:6) in the specified read address unless the specified address belongs to the sector being deleted.

---

#### **Reference:**

When executing the write operation, the toggle bit executes the toggle operation for about 2  $\mu$ s, then terminates it without rewriting the data, if the sector to be written is write-protected.

When executing the deletion operation, the toggle bit executes the toggle operation for about 100  $\mu$ s, then returns to the read/reset status without rewriting the data, if all the selected sectors are protected from rewriting.

---

Table 23.5-4 shows the toggle bit flag status transition.

**Table 23.5-4 Toggle Bit Flag Status Transition**

- Status transition during normal operation

Operating status	Write operation --> Completion	Chip/sector deletion -->Completion	Sector deletion wait-->Start	Sector deletion -->Deletion temporary stop (Sector being deleted)	Sector deletion temporary stop -->Restart (Sector being deleted)	During the sector deletion temporary stop (Sector not being deleted)
DQ6	Toggle -->DATA:6	Toggle-->Stop	Toggle	Toggle-->1	1-->Toggle	DATA:6

- Status transition during abnormal operation

Operating status	Write operation	Chip/sector deletion operation
DQ6	Toggle	Toggle

### 23.5.3 Time limit Exceeded Flag (DQ5)

The time limit exceeded flag (DQ5) indicates that the automatic algorithm execution time has exceeded the time defined within the flash memory (i.e., internal pulse count).

Table 23.5-5 shows the transition of the time limit exceeded flag status.

#### ■ When the Write Operation or Chip/Sector Deletion Operation is Executed.

When the read/access is executed after starting the write or chip/sector deletion automatic algorithm, this flag is set to "0" if the execution time is within the defined time (required for writing/deletion), or it outputs "1" if this execution time exceeds the defined time. This is not related to the state in which the automatic algorithm is being executed or has been terminated, so it can be determined whether the write/delete has succeeded or failed. Thus, when this flag is set to "1", it indicates that the write operation has failed if the automatic algorithm is being performed by the data polling function or toggle bit function.

For example, a fail occurs if an attempt is made to write "1" to the flash memory with "0" written. In this case, the flash memory is locked and the automatic algorithm is not terminated. Therefore, no valid data is set in data polling flag (DQ7). The toggle bit flag (DQ6) does not stop the toggle operation, so the execution time exceeds the time limit. Then, the time limit exceeded flag (DQ5) outputs "1". This event indicates that the flash memory has not been correctly used, but does not indicate that the flash memory is not good. If this event occurs, the reset command should be executed.

Table 23.5-5 shows the time limit exceeded flag status transition.

**Table 23.5-5 Transition of the Time Limit Exceeded Flag Status**

- Status transition during normal operation

Operating status	Write operation --> Completion	Chip/sector deletion -->Completion	Sector deletion wait-->Start	Sector deletion -->Deletion temporary stop (Sector being deleted)	Sector deletion temporary stop -->Restart (Sector being deleted)	During the sector deletion temporary stop (Sector not being deleted)
DQ5	0-->DATA:5	0-->1	0	0	0	DATA:5

- Status transition during abnormal operation

Operating status	Write operation	Chip/sector deletion operation
DQ5	1	1

**MB90820B Series****23.5.4 Sector Deletion Timer Flag (DQ3)**

After starting the sector deletion command, the sector deletion timer flag (DQ3) indicates whether it is "during the sector deletion waiting period". Table 23.5-6 shows the sector deletion timer flag status transition.

**■ When the Sector Deletion Operation is Executed.**

When the read/access is executed after starting the sector deletion command, the flash memory outputs "0" if this flag indicates "during the sector deletion waiting period", irrespective of the address specified by the address signal from the sector having issued the command. However, the flash memory outputs "1" if this flag exceeds the defined sector deletion waiting period.

When the deletion algorithm is being executed by the data polling function or toggle bit function, the internally-controlled deletion operation is started if this flag is "1". The succeeding commands except the sector deletion temporary stop command are ignored until deletion is terminated.

If this flag is "0", the flash memory accepts the additional sector deletion command. To verify this event, it is recommended that this flag status be checked before writing the succeeding sector deletion command. If this flag is "1" the additional sector deletion command that is temporary stopped may not be accepted.

**■ When the Sector Deletion Temporary Stop Operation is Executed.**

When the read/access is executed while executing the sector deletion temporary stop, the flash memory outputs "1" if the specified address belongs to the sector being deleted. However, unless the specified address belongs to the sector being deleted, the flash memory outputs the data of bit 3 (DATA:3) for the specified read address.

Table 23.5-6 shows the sector deletion timer flag status transition.

**Table 23.5-6 Sector Deletion Timer Flag Status Transition**

- Status transition during normal operation

Operating status	Write operation --> Completion	Chip/sector deletion --> Completion	Sector deletion wait-->Start	Sector deletion -->Deletion temporary stop (Sector being deleted)	Sector deletion temporary stop -->Restart (Sector being deleted)	During the sector deletion temporary stop (Sector not being deleted)
DQ3	0-->DATA:3	1	0-->1	1-->0	0-->1	DATA:3

- Status transition during abnormal operation

Operating status	Write operation	Chip/sector deletion operation
DQ3	0	1

## **23.6 Detailed Explanation on the Flash Memory Write/Delete**

---

**This section explains the procedures for issuing the command to start the automatic algorithm, reading/resetting the flash memory, writing the data to the flash memory, deleting the chip, deleting the sector, temporarily stopping the sector deletion, and restarting the sector deletion.**

---

### **■ Detailed Explanation on the Flash Memory Write/Delete**

The read/reset, write, chip deletion, sector deletion, sector deletion temporary stop, or deletion restart operation can be performed by the automatic algorithm which can be started by setting the command sequence (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory") to the flash memory from the CPU. The write cycles to the flash memory from the CPU must be executed continuously. The ending time of the automatic algorithm can be notified by the data polling function and so on. After the normal end of the automatic algorithm, the flash memory returns to the read/reset status.

The following describes the operations of flash memory write/delete.

- Setting the Read/Reset Status
- Writing the Data
- Deleting All Data (Chip Deletion)
- Deleting Arbitrary Data (Sector Deletion)
- Temporarily Stopping the Sector Deletion
- Restarting the Sector Deletion

## MB90820B Series

### 23.6.1 Setting the Read/Reset Status

---

**This section explains the procedure of issuing the read/reset command and setting the flash memory to the read/reset status.**

---

#### ■ Setting the Read/Reset Status

When the flash memory is set to the read/reset status, the read/reset command can be executed by continuously sending the read/reset command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU.

There are two types of read/reset command sequences, one is that the bus operation is executed once and the other is that the bus operations are executed three times. However, these command sequences have no essential difference.

The read/reset status is the initial status of the flash memory. When the power supply is turned on, or when the command is normally terminated, the flash memory is always set to the read/reset status. The read/reset status means the status of the flash memory that is waiting for another command to be input.

In the read/reset status, data can be read from the flash memory by executing a usual read/access command. The data can be program-accessed from CPU same as the mask ROM. This command is not required for usual data reading. This command should be mainly used for initializing the automatic algorithm if the command has not been normally terminated for any reason.



## **23.6.2 Writing the Data**

---

**This section explains the procedure of issuing the write command and writing the data to the flash memory. Figure 23.6-1 shows an example of procedure for writing data to the flash memory.**

---

### **■ Writing the Data**

The automatic algorithm for writing the data to the flash memory can be performed by continuously sending the write command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU. When the data write operation to the target address in the 4th cycle has been terminated, the automatic algorithm is started for automatic writing.

### **■ How to Specify the Address**

Only even addresses can be specified as the write address in the write data cycle. If an odd address is specified, data cannot be correctly written. That is, it is necessary to write the data in units of words to even addresses.

Data can be written to the flash memory, and any address sequence may be specified, even if data has been written across the sector boundary. However, only one-word data can be written by executing the write command once.

### **■ Notes on Writing the Data**

By writing the data, data "0" cannot be returned to data "1". If data "1" is written to data "0", the data polling algorithm (DQ7) or toggle operation (DQ6) is not terminated and the flash memory element is determined to be bad. Then, the time limit exceeded flag (DQ5) error may be determined by the excess of the write defined time, or data "1" may be apparently written but is not actually done. However, if data is read from the flash memory in the read/reset status, data remains "0". Only the deletion operation enables data "0" to be changed to data "1".

All commands are ignored while automatic writing is being performed. Note that the data at the address for writing is not assured if the hardware is reset during automatic writing.

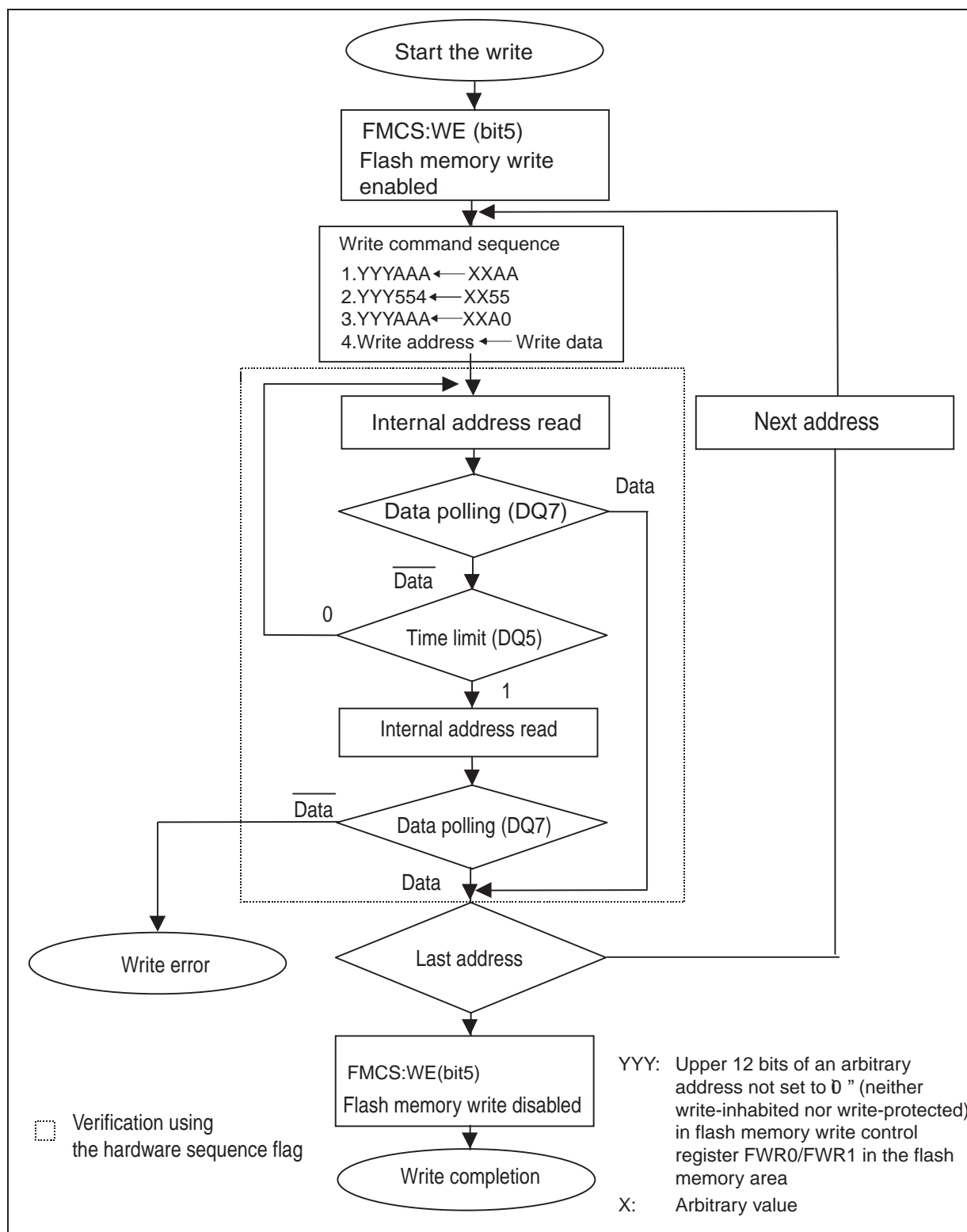
### **■ Procedure of Writing the Data to the Flash Memory**

Figure 23.6-1 shows an example of procedure for writing the data to the flash memory. Using the hardware sequence flag (see Section "23.5 Verifying Automatic Algorithm Execution Status"), the status of the automatic algorithm within the flash memory can be determined. Here, the data polling flag (DQ7) is used for verifying the end of writing.

The data reading for checking the flag is started from the last-written address. The data polling flag (DQ7) is changed at the same time when the time limit exceeded flag (DQ5) is changed, so the data polling flag (DQ7) must be rechecked even if the time limit exceeded flag (DQ5) is "1".

Similarly, the toggle bit flag (DQ6) stops the toggle operation at the same time when the time limit exceeded flag (DQ5) is changed to "1", so the toggle bit flag (DQ6) must be rechecked.

Figure 23.6-1 Example of Procedure for Writing the Data to the Flash Memory



### **23.6.3 Deleting All Data (Chip Deletion)**

---

**This section explains the procedure of issuing the chip deletion command and deleting all the data in the flash memory.**

---

#### **■ Deleting the Data (Chip Deletion)**

All the data can be deleted from the flash memory by continuously sending the chip deletion command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU.

The chip deletion command is executed by executing the bus operation six times. When the 6th-cycle write operation has been completed, the chip deletion operation is started. The user need not write the data to the flash memory before chip deletion operation. During the automatic deletion algorithm execution, the flash memory writes data "0" to all the cells and verifies them before they are automatically deleted.

## MB90820B Series

### 23.6.4 Deleting Arbitrary Data (Sector Deletion)

---

**This section explains the procedure of issuing the sector deletion command and deleting any sector from the flash memory (sector deletion). This command enables each sector to be deleted, and two or more sectors to be specified at the same time.**

---

#### ■ Deleting Arbitrary Data (Sector Deletion)

Any sector can be deleted from the flash memory by continuously sending the sector deletion command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU.

Method of Specifying a Sector

The sector deletion command is executed by executing the bus operation six times. The sector deletion wait of 50  $\mu$ s is started by writing the sector deletion code (30<sub>H</sub>) to any accessible even address in the target sector in the 6th-cycle bus operation. To delete two or more sectors, the deletion code (30<sub>H</sub>) should be written to the address in the target sector just after the above operation.

#### ■ Notes on Specifying Two or More Sectors

The deletion operation is started after the last sector deletion code is written and the sector deletion wait period of 50  $\mu$ s is terminated. Thus, the next deletion sector address and deletion code (i.e. in the 6th cycle of the command sequence) must be input each within 50  $\mu$ s to delete two or more sectors simultaneously, which may not be accepted later than 50  $\mu$ s. It can be checked whether the succeeding sector deletion code write operation is valid, using the sector deletion timer flag (DQ3). In this case, the address from which the sector deletion timer flag (DQ3) is read must indicate the sector to be deleted.

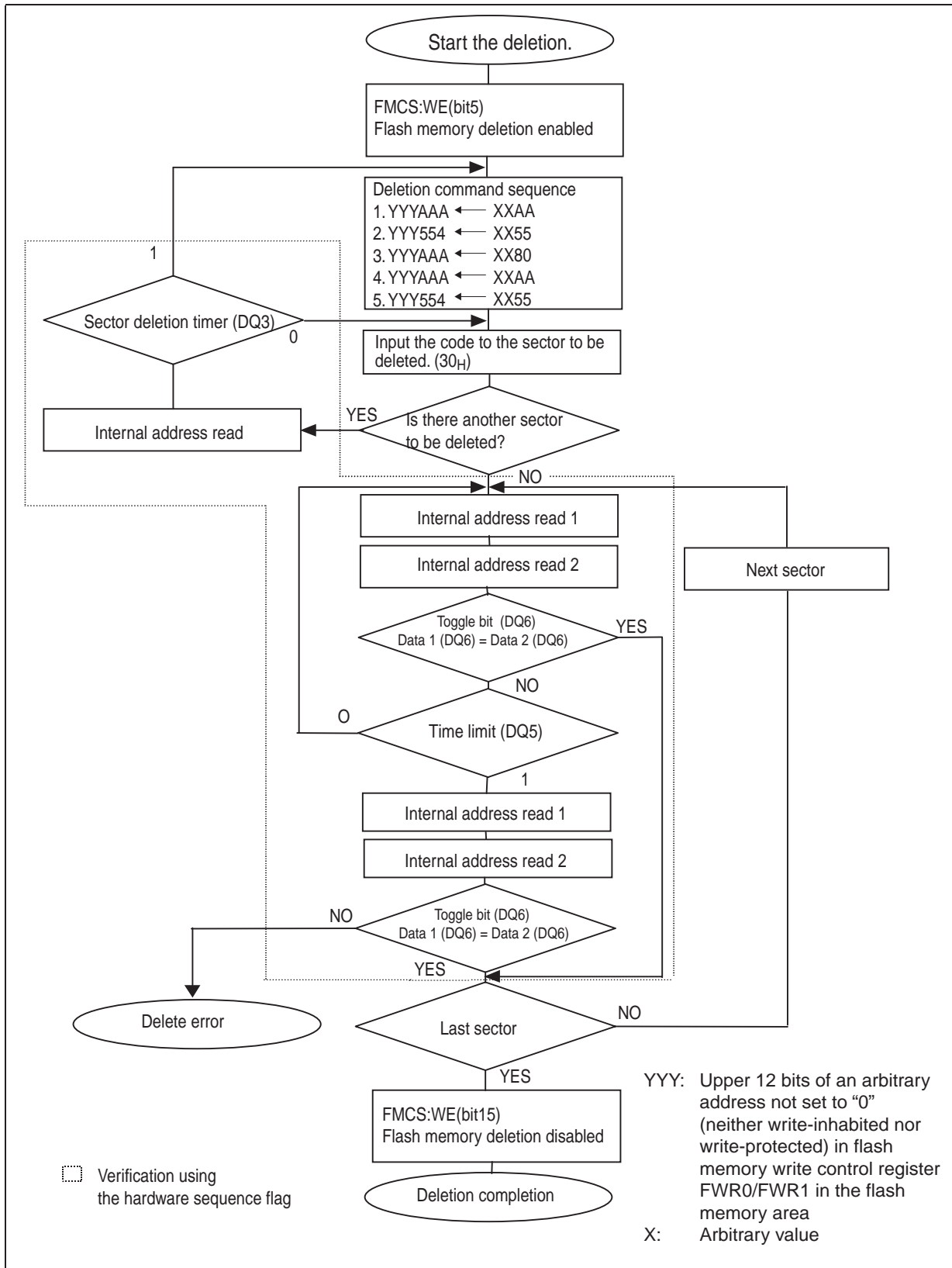
#### ■ Procedure of Deleting a Sector

Using the hardware sequence flag (see Section "23.5 Verifying Automatic Algorithm Execution Status"), the status of the automatic algorithm within the flash memory can be determined. Figure 23.6-2 shows an example of procedure for deleting the sector from the flash memory. Here, the toggle bit flag (DQ6) is used for verifying the end of deletion. Note that data to be used for checking the flag is read from the sector to be deleted.

The toggle bit flag (DQ6) stops the toggle operation at the same time when the time limit exceeded flag (DQ5) is changed to "1", so the toggle bit flag (DQ6) must be rechecked even if the time limit exceeded flag (DQ5) is "1".

Similarly, the data polling flag (DQ7) is changed at the same time when the time limit exceeded flag (DQ5) is changed, so the data polling flag (DQ7) must be rechecked.

**Figure 23.6-2 Example of Procedure for Deleting the Sector from the Flash Memory**



## MB90820B Series

### 23.6.5 Temporarily Stopping the Sector Deletion

---

**This section explains the procedure of issuing the sector deletion temporary stop command and temporarily stopping the deletion of a sector from the flash memory. This command can read the data from the sector not being deleted.**

---

#### ■ Temporarily Stopping the Sector Deletion

The sector deletion from the flash memory can be temporarily stopped by continuously sending the sector deletion temporary stop command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU.

The sector deletion temporary stop command stops the sector deletion operation temporarily, and enables the data reading from the sector not being deleted. In this case, only the read operation can be executed, but the write operation cannot be done. This command is valid only during the sector deletion operation including the deletion waiting time, but it is ignored during the chip deletion operation or the write operation.

This command is executed by writing the deletion temporary stop code (B0<sub>H</sub>). In this case, the address must indicate any address within the flash memory. During the deletion temporary stop operation, the reissued deletion temporary stop command is ignored.

If the sector deletion temporary stop command is input during the sector deletion waiting period, the sector deletion wait is immediately terminated to stop the deletion operation, and the flash memory enters the deletion stop status. If the sector deletion temporary stop command is input during the sector deletion operation after the sector deletion waiting period, the flash memory enters the deletion temporary stop status after a lapse of up to 15  $\mu$ s.

#### ■ Note

Before issuing a suspend command, wait for at least 20ms after issuing the sector erase command or sector erase resume command. The suspend command should not be issued too many times.

## **23.6.6 Restarting the Sector Deletion**

---

**This section explains the procedure of issuing the sector deletion restart command and restarting the operation of deleting a sector from the flash memory, which has been temporarily stopped.**

---

### **■ Restarting the Sector Deletion**

The sector deletion operation which has been temporarily stopped can be restarted by continuously sending the sector deletion restart command, listed in the command sequence table (see Section "23.4 Method of Starting the Automatic Algorithm in Flash Memory"), to the target sector in the flash memory from the CPU.

The sector deletion restart command is used to restart the sector deletion operation when the flash memory is in the sector deletion temporary stop status caused by the sector deletion temporary stop command. This command is executed by writing the deletion restart code (30<sub>H</sub>). In this case, the address must indicate any address within the flash memory area.

However, the sector deletion restart command issued during the sector deletion operation is ignored.

## MB90820B Series

### 23.7 Flash Security Feature

---

**The flash security controller provides possibilities to protect the content of the flash memory from being read from external pins.**

---

- One predefined address of the flash memory is assigned to the flash security controller (MB90F822B: FF0001<sub>H</sub>; MB90F823B: FE0001<sub>H</sub>). If the protection code of “01<sub>H</sub>” is written in this address, access to the flash memory is restricted. Once the flash memory is protected, performing the chip erase operation only can unlock the function otherwise read/write access to the flash memory from any external pins is not generally possible.
- This function is suitable for applications requiring security of self-containing program and data stored in the flash memory. If the target application requires any part of the program to locate outside the microcontroller, the flash security controller can not offer the intended features. For this reason, the external vector fetch mode should not be used when the protection code is set.
- Programming of the flash microcontroller by standard parallel programmer may require unique set-up. For example, with the programmer from Minato Electronics, the device checking should be turned off. Writing the protection code is generally recommended to take place at the end of the flash programming. This is to avoid unnecessary protection during the programming.
- In order to re-program the once protected flash memory, the chip erase operation should be preformed. For further information, please contact Fujitsu Microelectronics.





# ***APPENDIX***

---

APPENDIX A I/O Map

APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection  
for Serial Writing

APPENDIX C Instructions

## APPENDIX A I/O Map

Table A-1 lists the addresses assigned to the registers for peripheral functions in the MB90820B series.

### ■ I/O Map

Table A-1 I/O map (1 / 6)

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
000000 <sub>H</sub>	PDR0	Port 0 data register	R/W	R/W	Port 0	XXXXXXXX <sub>B</sub>
000001 <sub>H</sub>	PDR1	Port 1 data register	R/W	R/W	Port 1	XXXXXXXX <sub>B</sub>
000002 <sub>H</sub>	PDR2	Port 2 data register	R/W	R/W	Port 2	XXXXXXXX <sub>B</sub>
000003 <sub>H</sub>	PDR3	Port 3 data register	R/W	R/W	Port 3	XXXXXXXX <sub>B</sub>
000004 <sub>H</sub>	PDR4	Port 4 data register	R/W	R/W	Port 4	XXXXXXXX <sub>B</sub>
000005 <sub>H</sub>	PDR5	Port 5 data register	R/W	R/W	Port 5	XXXXXXXX <sub>B</sub>
000006 <sub>H</sub>	PDR6	Port 6 data register	R/W	R/W	Port 6	XXXXXXXX <sub>B</sub>
000007 <sub>H</sub>	PDR7	Port 7 data register	R/W	R/W	Port 7	XXXXXXXX <sub>B</sub>
000008 <sub>H</sub>	PDR8	Port 8 data register	R/W	R/W	Port 8	XXXXXXXX <sub>B</sub>
000009 <sub>H</sub> to 00000F <sub>H</sub>	Prohibited area					
000010 <sub>H</sub>	DDR0	Port 0 data direction register	R/W	R/W	Port 0	00000000 <sub>B</sub>
000011 <sub>H</sub>	DDR1	Port 1 data direction register	R/W	R/W	Port 1	00000000 <sub>B</sub>
000012 <sub>H</sub>	DDR2	Port 2 data direction register	R/W	R/W	Port 2	00000000 <sub>B</sub>
000013 <sub>H</sub>	DDR3	Port 3 data direction register	R/W	R/W	Port 3	00000000 <sub>B</sub>
000014 <sub>H</sub>	DDR4	Port 4 data direction register	R/W	R/W	Port 4	00000000 <sub>B</sub>
000015 <sub>H</sub>	DDR5	Port 5 data direction register	R/W	R/W	Port 5	XXXXXXXX00 <sub>B</sub>
000016 <sub>H</sub>	DDR6	Port 6 data direction register	R/W	R/W	Port 6	00000000 <sub>B</sub>
000017 <sub>H</sub>	DDR7	Port 7 data direction register	R/W	R/W	Port 7	00000000 <sub>B</sub>
000018 <sub>H</sub>	DDR8	Port 8 data direction register	R/W	R/W	Port 8	00000000 <sub>B</sub>
000019 <sub>H</sub> to 00001F <sub>H</sub>	Prohibited area					
000020 <sub>H</sub>	SMR0	Serial mode register 0	R/W	R/W	UART0	00000000 <sub>B</sub>
000021 <sub>H</sub>	SCR0	Serial control register 0	R/W	R/W		00000100 <sub>B</sub>
000022 <sub>H</sub>	SIDR0 / SODR0	Serial Input data register 0 / Serial Output data register 0	R/W	R/W		XXXXXXXX <sub>B</sub>
000023 <sub>H</sub>	SSR0	Serial status register 0	R/W	R/W		00001000 <sub>B</sub>
000024 <sub>H</sub>	SMR1	Serial mode register 1	R/W	R/W	UART1	00000000 <sub>B</sub>
000025 <sub>H</sub>	SCR1	Serial control register 1	R/W	R/W		00000100 <sub>B</sub>
000026 <sub>H</sub>	SIDR1 / SODR1	Serial Input data register 1 / Serial Output data register 1	R/W	R/W		XXXXXXXX <sub>B</sub>
000027 <sub>H</sub>	SSR1	Status register 1	R/W	R/W		00001000 <sub>B</sub>

Table A-1 I/O map (2 / 6)

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
000028 <sub>H</sub>	PWCSL1	PWC control status register 1	R/W	R/W	PWC timer 1	00000000 <sub>B</sub>
000029 <sub>H</sub>	PWCSH1		R/W	R/W		00000000 <sub>B</sub>
00002A <sub>H</sub>	PWC1	PWC data buffer register 1	-	R/W		XXXXXXXX <sub>B</sub>
00002B <sub>H</sub>						XXXXXXXX <sub>B</sub>
00002C <sub>H</sub>	DIV1	Divide ratio control register 1	R/W	R/W		XXXXXXXX00 <sub>B</sub>
00002D <sub>H</sub> , 00002E <sub>H</sub>	Prohibited area					
00002F <sub>H</sub>	PCKCR	PLL clock control register	W	W	PLL	XXXX0000 <sub>B</sub>
000030 <sub>H</sub>	ENIR	DTP / Interrupt enable register	R/W	R/W	DTP/external interrupt	00000000 <sub>B</sub>
000031 <sub>H</sub>	EIRR	DTP / Interrupt cause register	R/W	R/W		XXXXXXXX <sub>B</sub>
000032 <sub>H</sub>	ELVRL	Request level setting register (lower byte)	R/W	R/W		00000000 <sub>B</sub>
000033 <sub>H</sub>	ELVRH	Request level setting register (higher byte)	R/W	R/W		00000000 <sub>B</sub>
000034 <sub>H</sub>	Prohibited area					
000035 <sub>H</sub>	CDCR0	Clock division control register 0	R/W	R/W	Communication prescaler 0	00XXX000 <sub>B</sub>
000036 <sub>H</sub>	Prohibited area					
000037 <sub>H</sub>	CDCR1	Clock division control register 1	R/W	R/W	Communication prescaler 1	00XXX000 <sub>B</sub>
000038 <sub>H</sub>	PDCR0	PPG0 down counter register 0	-	R	16-bit PPG timer 0	1111111 <sub>B</sub>
000039 <sub>H</sub>						1111111 <sub>B</sub>
00003A <sub>H</sub>	PCSR0	PPG0 period setting register 0	-	W		XXXXXXXX <sub>B</sub>
00003B <sub>H</sub>						XXXXXXXX <sub>B</sub>
00003C <sub>H</sub>	PDUT0	PPG0 duty setting register 0	-	W		XXXXXXXX <sub>B</sub>
00003D <sub>H</sub>						XXXXXXXX <sub>B</sub>
00003E <sub>H</sub>	PCNTL0	PPG0 control status register 0	R/W	R/W		XX000000 <sub>B</sub>
00003F <sub>H</sub>	PCNTH0		R/W	R/W		00000000 <sub>B</sub>
000040 <sub>H</sub>	PDCR1	PPG1 down counter register 1	-	R	16-bit PPG timer 1	1111111 <sub>B</sub>
000041 <sub>H</sub>						1111111 <sub>B</sub>
000042 <sub>H</sub>	PCSR1	PPG1 period setting register 1	-	W		XXXXXXXX <sub>B</sub>
000043 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000044 <sub>H</sub>	PDUT1	PPG1 duty setting register 1	-	W		XXXXXXXX <sub>B</sub>
000045 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000046 <sub>H</sub>	PCNTL1	PPG1 control status register 1	R/W	R/W		XX000000 <sub>B</sub>
000047 <sub>H</sub>	PCNTH1		R/W	R/W		00000000 <sub>B</sub>
000048 <sub>H</sub>	PDCR2	PPG2 down counter register 2	-	R	16-bit PPG timer 2	1111111 <sub>B</sub>
000049 <sub>H</sub>						1111111 <sub>B</sub>
00004A <sub>H</sub>	PCSR2	PPG2 period setting register 2	-	W		XXXXXXXX <sub>B</sub>
00004B <sub>H</sub>						XXXXXXXX <sub>B</sub>
00004C <sub>H</sub>	PDUT2	PPG2 duty setting register 2	-	W		XXXXXXXX <sub>B</sub>
00004D <sub>H</sub>						XXXXXXXX <sub>B</sub>
00004E <sub>H</sub>	PCNTL2	PPG2 control status register 2	R/W	R/W		XX000000 <sub>B</sub>
00004F <sub>H</sub>	PCNTH2		R/W	R/W		00000000 <sub>B</sub>

**Table A-1 I/O map (3 / 6)**

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
000050 <sub>H</sub>	TMRR0	16-bit timer register 0	-	R/W	Waveform generator	XXXXXXXX <sub>B</sub>
000051 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000052 <sub>H</sub>	TMRR1	16-bit timer register 1	-	R/W		XXXXXXXX <sub>B</sub>
000053 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000054 <sub>H</sub>	TMRR2	16-bit timer register 2	-	R/W		XXXXXXXX <sub>B</sub>
000055 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000056 <sub>H</sub>	DTCR0	16-bit timer control register 0	R/W	R/W		00000000 <sub>B</sub>
000057 <sub>H</sub>	DTCR1	16-bit timer control register 1	R/W	R/W		00000000 <sub>B</sub>
000058 <sub>H</sub>	DTCR2	16-bit timer control register 2	R/W	R/W		00000000 <sub>B</sub>
000059 <sub>H</sub>	SIGCR	Waveform control register	R/W	R/W		00000000 <sub>B</sub>
00005A <sub>H</sub>	CPCLRB / CPCLR	Compare clear buffer register / Compare clear register (lower)	-	R/W	16-bit free-run timer	11111111 <sub>B</sub>
00005B <sub>H</sub>						11111111 <sub>B</sub>
00005C <sub>H</sub>	TCDT	Timer data register (lower)	-	R/W		00000000 <sub>B</sub>
00005D <sub>H</sub>						00000000 <sub>B</sub>
00005E <sub>H</sub>	TCCSL	Timer control status register (lower)	R/W	R/W	16-bit free-run timer	00000000 <sub>B</sub>
00005F <sub>H</sub>	TCCSH	Timer control status register (upper)	R/W	R/W		X0000000 <sub>B</sub>
000060 <sub>H</sub>	IPCP0	Input capture data register 0	-	R	16-bit input capture (0 to 3)	XXXXXXXX <sub>B</sub>
000061 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000062 <sub>H</sub>	IPCP1	Input capture data register 1	-	R		XXXXXXXX <sub>B</sub>
000063 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000064 <sub>H</sub>	IPCP2	Input capture data register 2	-	R		XXXXXXXX <sub>B</sub>
000065 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000066 <sub>H</sub>	IPCP3	Input capture data register 3	-	R		XXXXXXXX <sub>B</sub>
000067 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000068 <sub>H</sub>	PICSL01	Input capture control status register 01 (lower)	R/W	R/W		00000000 <sub>B</sub>
000069 <sub>H</sub>	PICSH01	PPG output control / Input capture control status register 01 (upper)	R/W	R/W		00000000 <sub>B</sub>
00006A <sub>H</sub>	ICSL23	Input capture control status register 23 (lower)	R/W	R/W		00000000 <sub>B</sub>
00006B <sub>H</sub>	ICSH23	Input capture control status register 23 (upper)	R	R		XXXXXX00 <sub>B</sub>
00006C <sub>H</sub> to 00006E <sub>H</sub>	Prohibited area					
00006F <sub>H</sub>	ROMM	ROM mirroring function selection register	W	W	ROM mirroring function	XXXXXXX1 <sub>B</sub>

Table A-1 I/O map (4 / 6)

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
000070 <sub>H</sub>	OCCPB0 /OCCP0	Output compare buffer register / Output compare register 0	-	R/W	Output compare (0 to 5)	XXXXXXXX <sub>B</sub>
000071 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000072 <sub>H</sub>	OCCPB1 /OCCP1	Output compare buffer register / Output compare register 1	-	R/W		XXXXXXXX <sub>B</sub>
000073 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000074 <sub>H</sub>	OCCPB2 /OCCP2	Output compare buffer register / Output compare register 2	-	R/W		XXXXXXXX <sub>B</sub>
000075 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000076 <sub>H</sub>	OCCPB3 /OCCP3	Output compare buffer register / Output compare register 3	-	R/W		XXXXXXXX <sub>B</sub>
000077 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000078 <sub>H</sub>	OCCPB4 /OCCP4	Output compare buffer register / Output compare register 4	-	R/W		XXXXXXXX <sub>B</sub>
000079 <sub>H</sub>						XXXXXXXX <sub>B</sub>
00007A <sub>H</sub>	OCCPB5 /OCCP5	Output compare buffer register / Output compare register 5	-	R/W		XXXXXXXX <sub>B</sub>
00007B <sub>H</sub>						XXXXXXXX <sub>B</sub>
00007C <sub>H</sub>	OCS0	Compare control register 0	R/W	R/W		00000000 <sub>B</sub>
00007D <sub>H</sub>	OCS1	Compare control register 1	R/W	R/W		X0000000 <sub>B</sub>
00007E <sub>H</sub>	OCS2	Compare control register 2	R/W	R/W		00000000 <sub>B</sub>
00007F <sub>H</sub>	OCS3	Compare control register 3	R/W	R/W		X0000000 <sub>B</sub>
000080 <sub>H</sub>	OCS4	Compare control register 4	R/W	R/W		00000000 <sub>B</sub>
000081 <sub>H</sub>	OCS5	Compare control register 5	R/W	R/W		X0000000 <sub>B</sub>
000082 <sub>H</sub>	TMCSRL0	Timer control status register 0 (lower)	R/W	R/W	16-bit reload timer 0	00000000 <sub>B</sub>
000083 <sub>H</sub>	TMCSRH0	Timer control status register 0 (upper)	R/W	R/W		XXXX0000 <sub>B</sub>
000084 <sub>H</sub>	TMR0 / TMRD0	16 bit timer register 0 / 16-bit reload register 0	-	R/W		XXXXXXXX <sub>B</sub>
000085 <sub>H</sub>						XXXXXXXX <sub>B</sub>
000086 <sub>H</sub>	TMCSRL1	Timer control status register 1 (lower)	R/W	R/W	16-bit reload timer 1	00000000 <sub>B</sub>
000087 <sub>H</sub>	TMCSRH1	Timer control status register 1 (upper)	R/W	R/W		XXXX0000 <sub>B</sub>
000088 <sub>H</sub>	TMR1 / TMRD1	16 bit timer register 1 / 16-bit reload register 1	-	R/W		XXXXXXXX <sub>B</sub>
000089 <sub>H</sub>						XXXXXXXX <sub>B</sub>
00008A <sub>H</sub>	CSVCR	Clock supervisor control register *	R, R/W	-	Clock supervisor	00011100 <sub>B</sub>
00008B <sub>H</sub>	Prohibited area					
00008C <sub>H</sub>	RDR0	Port 0 pull-up resistor setting register	R/W	R/W	Port 0	00000000 <sub>B</sub>
00008D <sub>H</sub>	RDR1	Port 1 pull-up resistor setting register	R/W	R/W	Port 1	00000000 <sub>B</sub>
00008E <sub>H</sub>	RDR2	Port 2 pull-up resistor setting register	R/W	R/W	Port 2	00000000 <sub>B</sub>
00008F <sub>H</sub>	RDR3	Port 3 pull-up resistor setting register	R/W	R/W	Port 3	00000000 <sub>B</sub>
000090 <sub>H</sub> to 00009D <sub>H</sub>	Prohibited area					
00009E <sub>H</sub>	PACSR	Program address detect control status register	R/W	R/W	Address match detection	00000000 <sub>B</sub>
00009F <sub>H</sub>	DIRR	Delayed interrupt reset cause / clear register	R/W	R/W	Delayed interrupt input generation	XXXXXXXX0 <sub>B</sub>
0000A0 <sub>H</sub>	LPMCR	Low-power consumption mode control register	R/W	R/W	Low-power consumption mode	00011000 <sub>B</sub>
0000A1 <sub>H</sub>	CKSCR	Clock selection register	R/W	R/W		11111100 <sub>B</sub>
0000A2 <sub>H</sub> to 0000A7 <sub>H</sub>	Prohibited area					
0000A8 <sub>H</sub>	WDTC	Watchdog timer control register	R/W	R/W	Watchdog timer	XXXXX111 <sub>B</sub>

**Table A-1 I/O map (5 / 6)**

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
0000A9 <sub>H</sub>	TBTC	Time-base timer control register	R/W	R/W	Time-base timer	1XX00100 <sub>B</sub>
0000AA <sub>H</sub> to 0000AD <sub>H</sub>	Prohibited area					
0000AE <sub>H</sub>	FMCS	Flash memory control status register	R/W	R/W	Flash memory interface circuit	000X0000 <sub>B</sub>
0000AF <sub>H</sub>	Prohibited area					
0000B0 <sub>H</sub>	ICR00	Interrupt control register 00	R/W	R/W	Interrupt controller	00000111 <sub>B</sub>
0000B1 <sub>H</sub>	ICR01	Interrupt control register 01	R/W	R/W		00000111 <sub>B</sub>
0000B2 <sub>H</sub>	ICR02	Interrupt control register 02	R/W	R/W		00000111 <sub>B</sub>
0000B3 <sub>H</sub>	ICR03	Interrupt control register 03	R/W	R/W		00000111 <sub>B</sub>
0000B4 <sub>H</sub>	ICR04	Interrupt control register 04	R/W	R/W		00000111 <sub>B</sub>
0000B5 <sub>H</sub>	ICR05	Interrupt control register 05	R/W	R/W		00000111 <sub>B</sub>
0000B6 <sub>H</sub>	ICR06	Interrupt control register 06	R/W	R/W		00000111 <sub>B</sub>
0000B7 <sub>H</sub>	ICR07	Interrupt control register 07	R/W	R/W		00000111 <sub>B</sub>
0000B8 <sub>H</sub>	ICR08	Interrupt control register 08	R/W	R/W		00000111 <sub>B</sub>
0000B9 <sub>H</sub>	ICR09	Interrupt control register 09	R/W	R/W		00000111 <sub>B</sub>
0000BA <sub>H</sub>	ICR10	Interrupt control register 10	R/W	R/W		00000111 <sub>B</sub>
0000BB <sub>H</sub>	ICR11	Interrupt control register 11	R/W	R/W		00000111 <sub>B</sub>
0000BC <sub>H</sub>	ICR12	Interrupt control register 12	R/W	R/W		00000111 <sub>B</sub>
0000BD <sub>H</sub>	ICR13	Interrupt control register 13	R/W	R/W		00000111 <sub>B</sub>
0000BE <sub>H</sub>	ICR14	Interrupt control register 14	R/W	R/W		00000111 <sub>B</sub>
0000BF <sub>H</sub>	ICR15	Interrupt control register 15	R/W	R/W		00000111 <sub>B</sub>
0000C0 <sub>H</sub>	PWCSL0	PWC control status register 0	R/W	R/W	PWC timer 0	00000000 <sub>B</sub>
0000C1 <sub>H</sub>	PWCSH0		R/W	R/W		00000000 <sub>B</sub>
0000C2 <sub>H</sub>	PWC0	PWC data buffer register 0	-	R/W		XXXXXXXX <sub>B</sub>
0000C3 <sub>H</sub>						XXXXXXXX <sub>B</sub>
0000C4 <sub>H</sub>	DIV0	Divide ratio control register 0	R/W	R/W		XXXXXXXX00 <sub>B</sub>
0000C5 <sub>H</sub>	ADER0	A/D input enable register 0	R/W	R/W	Port 6, A/D	11111111 <sub>B</sub>
0000C6 <sub>H</sub>	ADCS0	A/D control status register 0	R/W	R/W	8/10-bit A/D converter	000XXXX0 <sub>B</sub>
0000C7 <sub>H</sub>	ADCS1	A/D control status register 1	R/W	R/W		0000000X <sub>B</sub>
0000C8 <sub>H</sub>	ADCR0	A/D data register 0	R	R		00000000 <sub>B</sub>
0000C9 <sub>H</sub>	ADCR1	A/D data register 1	R/W	R/W		XXXXXXXX00 <sub>B</sub>
0000CA <sub>H</sub>	ADSR0	A/D setting register 0	R/W	R/W		00000000 <sub>B</sub>
0000CB <sub>H</sub>	ADSR1	A/D setting register 1	R/W	R/W		00000000 <sub>B</sub>
0000CC <sub>H</sub>	DAT0	D/A data register 0	R/W	R/W	8-bit D/A converter	XXXXXXXX <sub>B</sub>
0000CD <sub>H</sub>	DAT1	D/A data register 1	R/W	R/W		XXXXXXXX <sub>B</sub>
0000CE <sub>H</sub>	DACR0	D/A control register 0	R/W	R/W		XXXXXXXX0 <sub>B</sub>
0000CF <sub>H</sub>	DACR1	D/A control register 1	R/W	R/W		XXXXXXXX0 <sub>B</sub>
0000D0 <sub>H</sub>	ADER1	A/D input enable register 1	R/W	R/W	Port 7, A/D	11111111 <sub>B</sub>
0000D1 <sub>H</sub> to 0000EF <sub>H</sub>	Prohibited area					
0000F0 <sub>H</sub> to 0000 FF <sub>H</sub>	External area					

**Table A-1 I/O map (6 / 6)**

Address	Abbreviation	Register	Byte access	Word access	Resource name	Initial value
001FF0 <sub>H</sub>	PADRL0	Program address detection register 0 (lower byte)	R/W	R/W	Address match detection	XXXXXXXX <sub>B</sub>
001FF1 <sub>H</sub>	PADRM0	Program address detection register 0 (middle byte)	R/W	R/W		XXXXXXXX <sub>B</sub>
001FF2 <sub>H</sub>	PADRH0	Program address detection register 0 (higher byte)	R/W	R/W		XXXXXXXX <sub>B</sub>
001FF3 <sub>H</sub>	PADRL1	Program address detection register 1 (lower byte)	R/W	R/W		XXXXXXXX <sub>B</sub>
001FF4 <sub>H</sub>	PADRM1	Program address detection register 1 (middle byte)	R/W	R/W		XXXXXXXX <sub>B</sub>
001FF5 <sub>H</sub>	PADRH1	Program address detection register 1 (higher byte)	R/W	R/W		XXXXXXXX <sub>B</sub>

\*: MB90F828B only. Prohibited for the other product types.

● Meaning of abbreviations used for reading and writing

R/W: Read and write enabled

R: Read-only

W: Write-only

● Explanation of initial values

0: The bit is initialized to 0.

1: The bit is initialized to 1.

X: The initial value of the bit is undefined.



## **APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing**

---

**This chapter describes examples of F<sup>2</sup>MC-16LX MB90F822B/F823B connections for serial writing.**

---

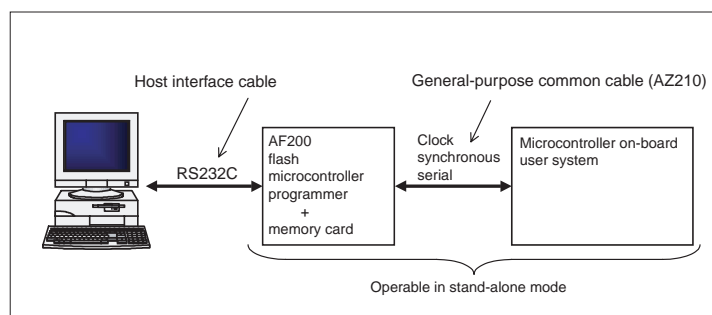
- B.1 Standard Configuration for Serial On-board Writing (Fujitsu Standard)
- B.2 Example of Connection for Serial Writing (When Power Supplied by User)
- B.3 Example of Connection for Serial Writing (When Power Supplied from Writer)
- B.4 Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied by User)
- B.5 Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied from Writer)

**MB90820B Series****B.1 Standard Configuration for Serial On-board Writing (Fujitsu Standard)**

**MB90F822B/F823B supports serial on-board writing (Fujitsu standard) to flash ROM. This describes the specifications for serial on-board writing.**

**■ Standard Configuration for Fujitsu Standard Serial On-board Writing**

The AF220/AF210/AF120/AF110 flash microcontroller programmer of Yokogawa Digital Computer Co., Ltd. is used for Fujitsu standard serial on-board writing.



Note :

Contact Yokogawa Digital Computer Co., Ltd. for the functionality and operation of the flash microcontroller programmer and information on the general-purpose common cable (AZ210) and connectors.

**Table B.1-1 Pins used for Fujitsu standard serial on-board writing (1 / 2)**

Pin	Function	Description															
MD2, MD1, MD0	Mode pins	Used to enable write mode for the flash microcontroller programmer.															
X0, X1	Oscillator pins	In write mode, since the operation clock is one times the CPU clock, the oscillation clock frequency is the internal operation clock. The resonator used for serial rewriting is therefore 1 MHz to 16 MHz.															
P00, P01	Programing activation pins	<table border="1"> <thead> <tr> <th>P00</th><th>P01</th><th>Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Asynchronous mode Machine clock = 16 MHz Baud rate = 19200 bps</td></tr> <tr> <td>0</td><td>1</td><td>Asynchronous mode Machine clock = 20 MHz Baud rate = 19200 bps</td></tr> <tr> <td>1</td><td>0</td><td>Synchronous mode</td></tr> <tr> <td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table>	P00	P01	Function	0	0	Asynchronous mode Machine clock = 16 MHz Baud rate = 19200 bps	0	1	Asynchronous mode Machine clock = 20 MHz Baud rate = 19200 bps	1	0	Synchronous mode	1	1	Reserved
P00	P01	Function															
0	0	Asynchronous mode Machine clock = 16 MHz Baud rate = 19200 bps															
0	1	Asynchronous mode Machine clock = 20 MHz Baud rate = 19200 bps															
1	0	Synchronous mode															
1	1	Reserved															

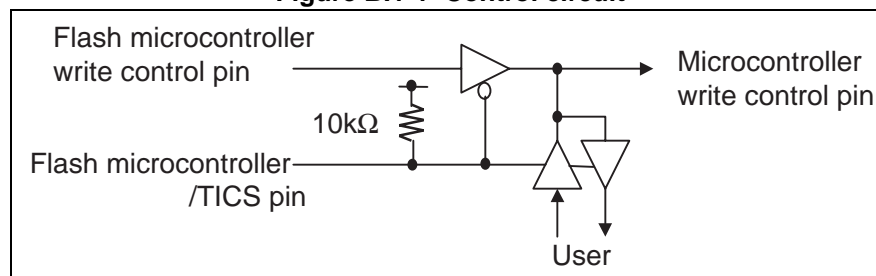
### Table B.1-1 Pins used for Fujitsu standard serial on-board writing (2 / 2)

Pin	Function	Description
RSTX	Reset pin	—
SIN0	Serial data input pin	UART0 is used in CLK synchronous mode.
SOT0	Serial data output pin	
SCK0	Serial clock input pin	
C	C pin	Capacitance pin for power stabilization. Connect a ceramic capacitor of about 0.1 $\mu$ F to the outside.
VCC	Power supply pin	If write voltage (5 V $\pm$ 10%) is supplied from the user system, this pin need not be connected to the flash microcontroller programmer. If the pin is connected to the flash microcontroller programmer, do not connect it to the power of the user system.
VSS	Ground pin	Used also as the ground pin for the flash microcontroller programmer.

Note:

When the P00, P01, SIN0, SOT0, SCK0 pins are also used by the user system, the control circuit shown below is required. (The /TICS signal of the flash microcontroller programmer can separate the user circuit during serial writing. See the connection example shown later.)

**Figure B.1-1 Control circuit**



See the following four serial writing examples in Appendix B.2 to B.5.

- Example of serial write connection when power supplied by user
- Example of serial write connection when power supplied from writer
- Example of minimum connection to flash microcontroller programmer when power supplied by user
- Example of minimum connection to flash microcontroller programmer when power supplied from writer

**Table B.1-2 System configuration of AF200 flash microcontroller programmer  
(Yokogawa Digital Computer Co., Ltd.)**

Model		Function
Main unit	AF220/ AC4P	Ethernet interface built-in model and 100 to 220 V AC power adapter
	AF210/ AC4P	Standard model and 100 to 220 V AC power adapter
	AF120/ AC4P	Single-key Ethernet interface built-in model and 100 to 220 V AC power adapter
	AF110/ AC4P	Single-key model and 100 to 220 V AC power adapter
AZ221		PC/AT RS232C cable for programmer
AZ210		Standard target probe (a) with a 1 m cable
FF201		Fujitsu F <sup>2</sup> MC-16LX flash microcontroller control module
AZ290		Remote controller
/P2		2MB PC card (optional) for flash memory sizes up to 128 KB
/P4		4MB PC card (optional) for flash memory sizes up to 512 KB

Inquiries: Yokogawa Digital Computer Corporation

Telephone number: (81)-42-333-6224

---

**Note:**

Although the flash microcontroller programmer is no longer manufactured, the programmer still can be used in combination with the FF201 control module.

Examples of serial programming connection are given in "Oscillating clock frequency and serial clock input frequency".

---

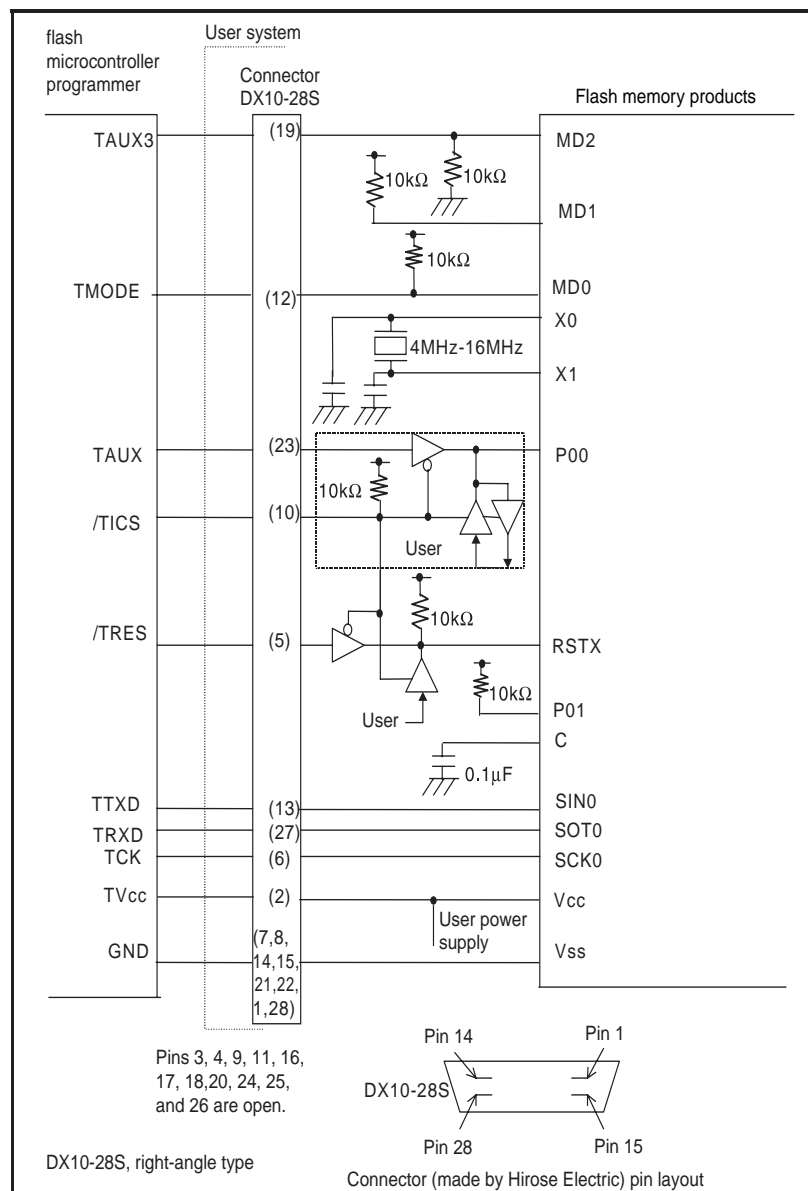
## B.2 Example of Connection for Serial Writing (When Power Supplied by User)

Figure B.2-1 is an example of serial write connection when power is supplied by the user.

MD2=1 and MD0=0 are input from TAUX3 and TMODE respectively in AF200 flash microcontroller programmer. Serial write mode: MD2, MD1, MD0 = 110<sub>B</sub>

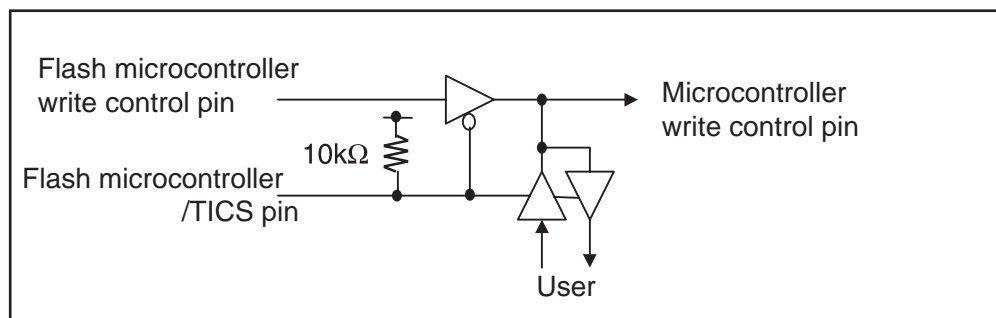
### ■ Example of Connection for Serial Writing (When Power Supplied by User)

Figure B.2-1 Example of connection for serial writing in Single chip mode (when power supplied by user)



**MB90820B Series**APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing

- When the user system also uses pins SIN0, SOT0 and SCK0, the control circuit shown below is necessary, just as it is for P00. (During serial writing, the user circuit can be disconnected by the flash microcontroller programmer /TICS signal.)
- Before connecting the Flash microcontroller, turn off the power supplied by the user.

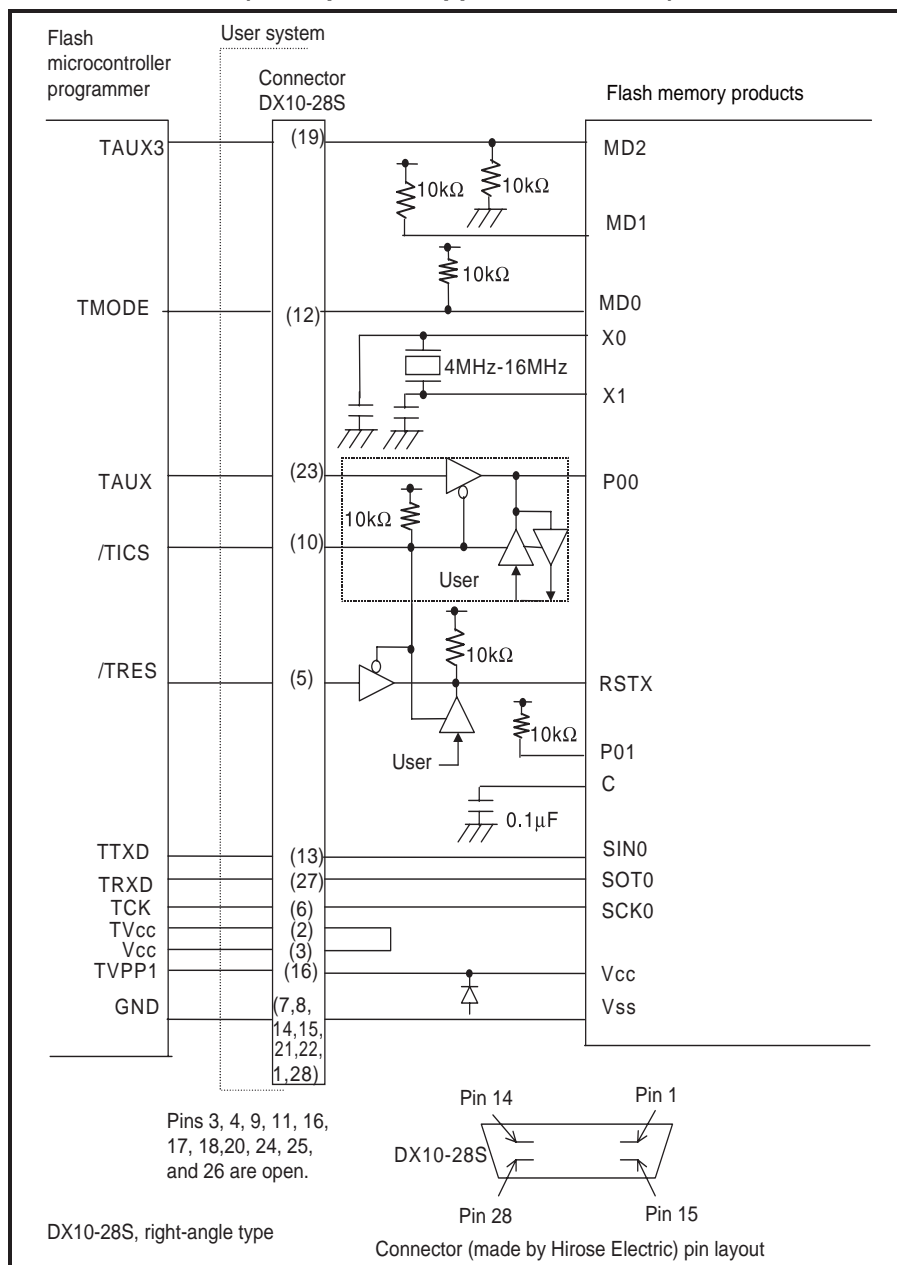
**Figure B.2-2 Control circuit**

### B.3 Example of Connection for Serial Writing (When Power Supplied from Writer)

Figure B.3-1 is an example of serial write connection when power is supplied from the writer. MD2=1 and MD0 are input from TAUX3 and TMODE respectively in flash microcontroller programmer. Serial write mode: MD2, MD1, MD0 = 110<sub>B</sub>

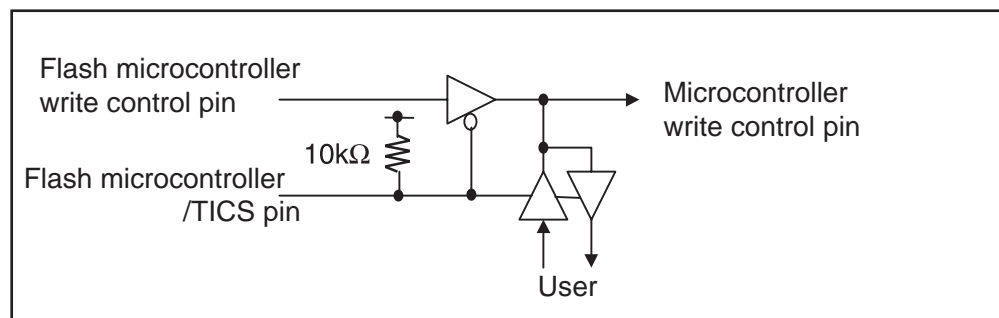
#### ■ Example of Connection for Serial Writing (When Power Supplied from Writer)

Figure B.3-1 Example of connection for serial writing in Single chip mode  
(when power supplied from writer)



**MB90820B Series**APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing

- When the SIN0, SOT0 and SCK0 pins are also used by the user system, the control circuit shown below is necessary, just as it is for P00. (During serial writing, the user circuit can be disconnected by the flash microcontroller /TICS signal.)
- Before connecting the Flash microcontroller programmer, turn off the power supplied by the user.
- When supplying write power from Flash microcontroller, do not create a short with the power supplied by the user.

**Figure B.3-2 Control circuit**



## B.4 Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied by User)

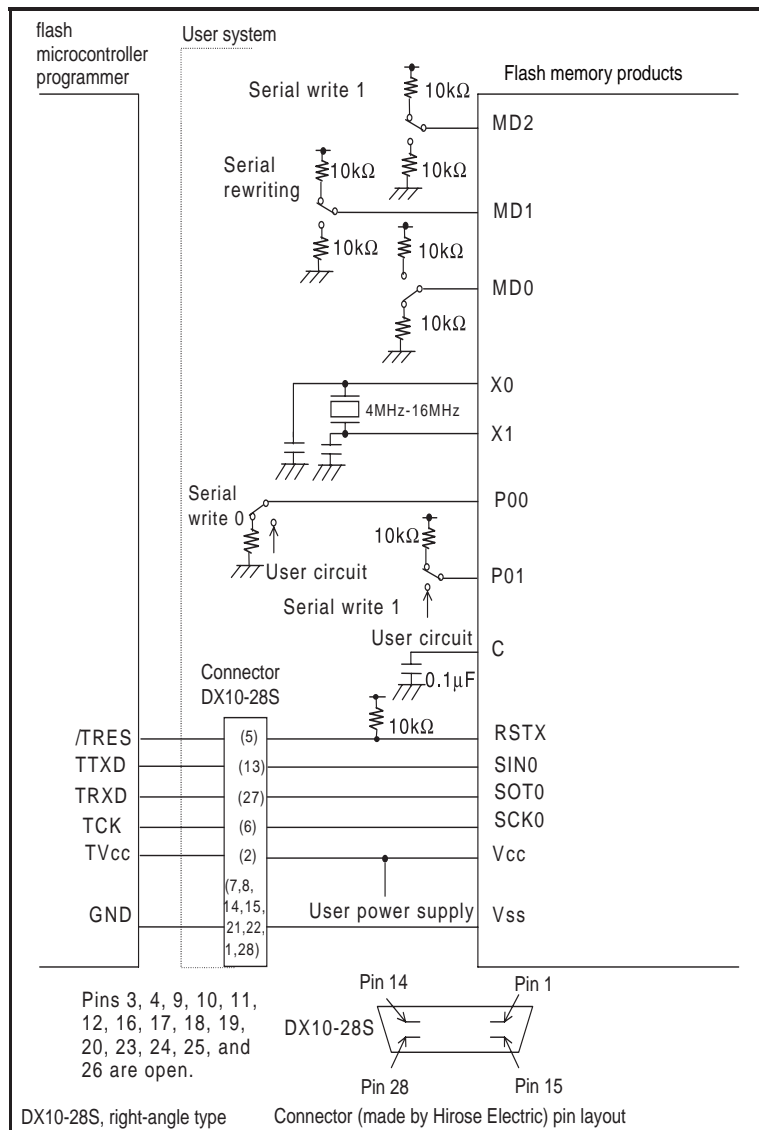
Figure B.4-1 is an example of the minimum connection with the flash microcontroller programmer when power is supplied by the user.

Serial write mode: MD2, MD1, MD0 = 110<sub>B</sub>

### ■ Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied by User)

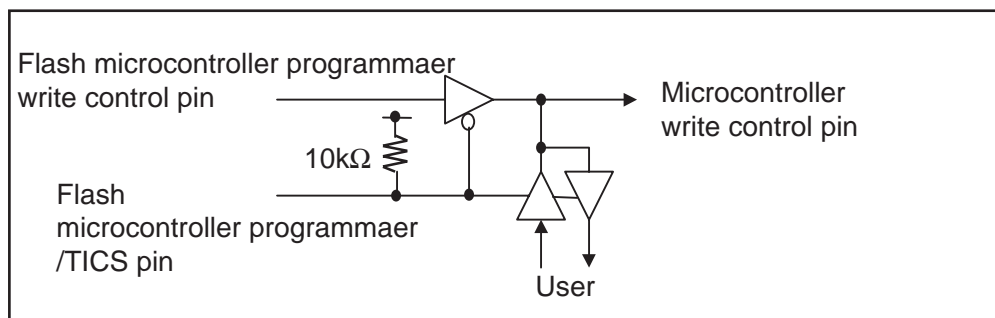
If the pins are set as shown in Figure B.4-1 during writing to flash memory, MD2, MD1, MD0, P00 and flash microcontroller programmer connection is unnecessary.

**Figure B.4-1 Example of minimum connection with flash microcontroller programmer (when power supplied by user)**



**MB90820B Series**APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing

- When the user system also uses the SIN0, SOT0 and SCK0 pins, the control circuit shown below is necessary. (During serial writing, the user circuit can be disconnected by the flash microcontroller programmer /TICS signal.)
- Before connecting the Flash microcontroller programmer, turn off the power supplied by the user.

**Figure B.4-2 Control circuit**

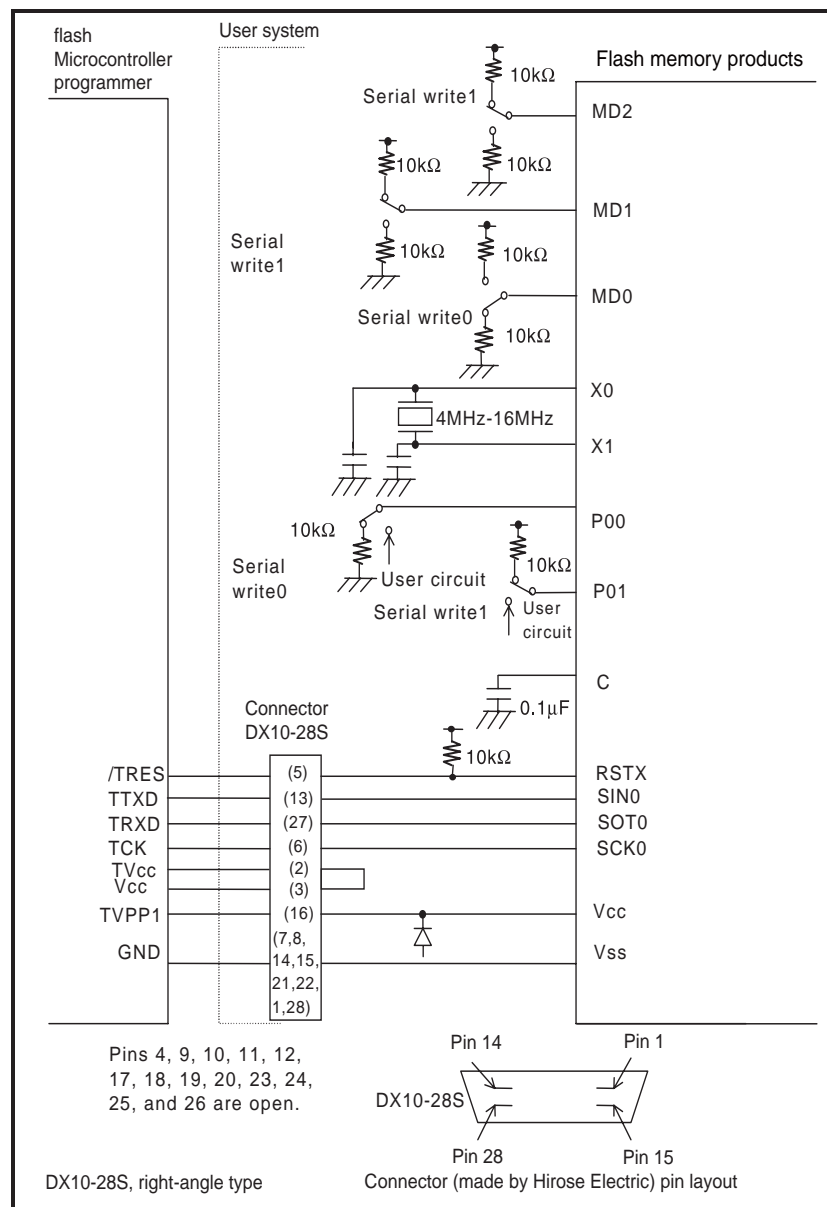
## B.5 Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied from Writer)

Figure B.5-1 is an example of the minimum connection with the flash microcontroller programmer when power is supplied from the writer.

### ■ Example of Minimum Connection with Flash Microcontroller Programmer (When Power Supplied from Writer)

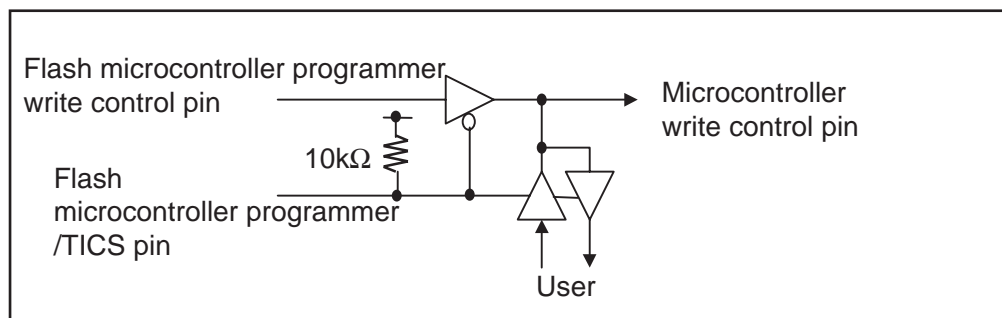
If the pins are set as shown in Figure B.5-1 during writing to flash memory, MD2, MD1, MD0, P00 and flash microcontroller programmer connection is unnecessary.

**Figure B.5-1 Example of minimum connection with flash microcontroller programmer (when power supplied from writer)**



**MB90820B Series**APPENDIX B Example of F<sup>2</sup>MC-16LX MB90F822B/F823B Connection for Serial Writing

- When the user system also uses the SIN0, SOT0, SCK0 pins, the control circuit shown below is necessary. (During serial writing, the user circuit can be disconnected by the flash microcontroller programmer /TICS signal.)
- Before connecting Flash microcontroller programmer, turn off the power supplied by the user.
- When write power is supplied from Flash microcontroller programmer, do not create a short with the power supplied by user.

**Figure B.5-2 Control circuit**

## **APPENDIX C Instructions**

---

**APPENDIX C describes the instructions used by the F<sup>2</sup>MC-16LX.**

---

- C.1 Instruction Types
- C.2 Addressing
- C.3 Direct Addressing
- C.4 Indirect Addressing
- C.5 Execution Cycle Count
- C.6 Effective address field
- C.7 How to Read the Instruction List
- C.8 F<sup>2</sup>MC-16LX Instruction List
- C.9 Instruction Map

## C.1 Instruction Types

---

The F<sup>2</sup>MC-16LX supports 351 types of instructions. Addressing is enabled by using an effective address field of each instruction or using the instruction code itself.

---

### ■ Instruction Types

The F<sup>2</sup>MC-16LX supports the following 351 types of instructions:

- 41 transfer instructions (byte)
- 38 transfer instructions (word or long word)
- 42 addition/subtraction instructions (byte, word, or long word)
- 12 increment/decrement instructions (byte, word, or long word)
- 11 comparison instructions (byte, word, or long word)
- 11 unsigned multiplication/division instructions (word or long word)
- 11 signed multiplication/division instructions (word or long word)
- 39 logic instructions (byte or word)
- 6 logic instructions (long word)
- 6 sign inversion instructions (byte or word)
- 1 normalization instruction (long word)
- 18 shift instructions (byte, word, or long word)
- 50 branch instructions
- 6 accumulator operation instructions (byte or word)
- 28 other control instructions (byte, word, or long word)
- 21 bit operation instructions
- 10 string instructions

## C.2 Addressing

---

With the F<sup>2</sup>MC-16LX, the address format is determined by the instruction effective address field or the instruction code itself (implied). When the address format is determined by the instruction code itself, specify an address in accordance with the instruction code used. Some instructions permit the user to select several types of addressing.

---

### ■ Addressing

The F<sup>2</sup>MC-16LX supports the following 23 types of addressing:

- Immediate (#imm)
- Register direct
- Direct branch address (addr16)
- Physical direct branch address (addr24)
- I/O direct (io)
- Abbreviated direct address (dir)
- Direct address (addr16)
- I/O direct bit address (io:bp)
- Abbreviated direct bit address (dir:bp)
- Direct bit address (addr16:bp)
- Vector address (#vct)
- Register indirect (@RWj j = 0 to 3)
- Register indirect with post increment (@RWj+ j = 0 to 3)
- Register indirect with displacement (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)
- Long register indirect with displacement (@RLi + disp8 i = 0 to 3)
- Program counter indirect with displacement (@PC + disp16)
- Register indirect with base index (@RW0 + RW7, @RW1 + RW7)
- Program counter relative branch address (rel)
- Register list (rlst)
- Accumulator indirect (@A)
- Accumulator indirect branch address (@A)
- Indirectly-specified branch address (@ear)
- Indirectly-specified branch address (@eam)

## ■ Effective Address Field

Table C.2-1 lists the address formats specified by the effective address field.

**Table C.2-1 Effective Address Field**

Code	Representation			Address format	Default bank
00	R0	RW0	RL0	Register direct: Individual parts correspond to the byte, word, and long word types in order from the left.	None
01	R1	RW1	(RL0)		
02	R2	RW2	RL1		
03	R3	RW3	(RL1)		
04	R4	RW4	RL2		
05	R5	RW5	(RL2)		
06	R6	RW6	RL3		
07	R7	RW7	(RL3)		
08	@RW0			Register indirect	DTB
09	@RW1				DTB
0A	@RW2				ADB
0B	@RW3				SPB
0C	@RW0+			Register indirect with post increment	DTB
0D	@RW1+				DTB
0E	@RW2+				ADB
0F	@RW3+				SPB
10	@RW0+disp8			Register indirect with 8-bit displacement	DTB
11	@RW1+disp8				DTB
12	@RW2+disp8				ADB
13	@RW3+disp8				SPB
14	@RW4+disp8			Register indirect with 8-bit displacement	DTB
15	@RW5+disp8				DTB
16	@RW6+disp8				ADB
17	@RW7+disp8				SPB
18	@RW0+disp16			Register indirect with 16-bit displacement	DTB
19	@RW1+disp16				DTB
1A	@RW2+disp16				ADB
1B	@RW3+disp16				SPB
1C	@RW0+RW7			Register indirect with index	DTB
1D	@RW1+RW7			Register indirect with index	DTB
1E	@PC+disp16			PC indirect with 16-bit displacement	PCB
1F	addr16			Direct address	DTB



## C.3 Direct Addressing

An operand value, register, or address is specified explicitly in direct addressing mode.

### ■ Direct Addressing

#### ● Immediate addressing (#imm)

Specify an operand value explicitly (#imm4/ #imm8/ #imm16/ #imm32).

**Figure C.3-1 Example of Immediate Addressing (#imm)**

MOVW A, #01212H (This instruction stores the operand value in A.)											
Before execution	A	<table><tr><td>2</td><td>2</td><td>3</td><td>3</td><td>:</td><td>4</td><td>4</td><td>5</td><td>5</td></tr></table>	2	2	3	3	:	4	4	5	5
2	2	3	3	:	4	4	5	5			
After execution	A	<table><tr><td>4</td><td>4</td><td>5</td><td>5</td><td>:</td><td>1</td><td>2</td><td>1</td><td>2</td></tr></table> (Some instructions transfer AL to AH.)	4	4	5	5	:	1	2	1	2
4	4	5	5	:	1	2	1	2			

#### ● Register direct addressing

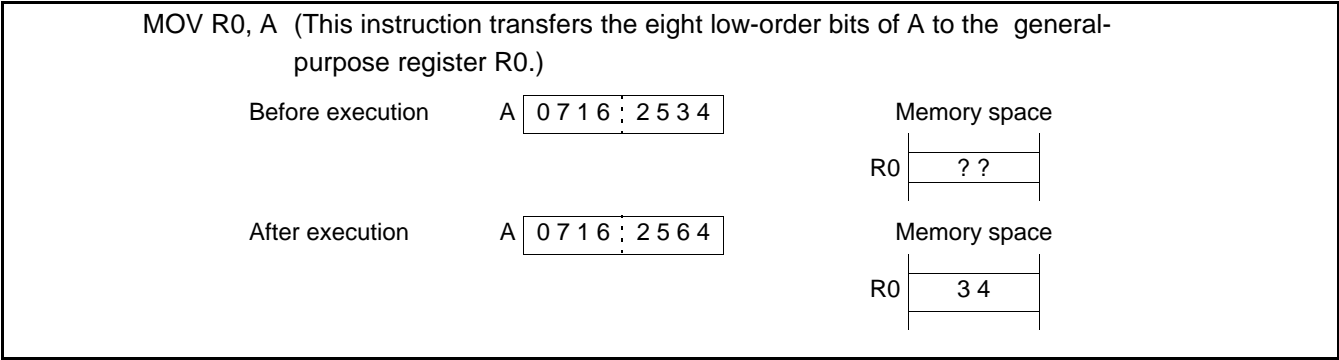
Specify a register explicitly as an operand. Table C.3-1 lists the registers that can be specified. Figure C.3-2 shows an example of register direct addressing.

**Table C.3-1 Direct Addressing Registers**

General-purpose register	Byte	R0, R1, R2, R3, R4, R5, R6, R7
	Word	RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7
	Long word	RL0, RL1, RL2, RL3
Special-purpose register	Accumulator	A, AL
	Pointer	SP *
	Bank	PCB, DTB, USB, SSB, ADB
	Page	DPR
	Control	PS, CCR, RP, ILM

\*: One of the user stack pointer (USP) and system stack pointer (SSP) is selected and used depending on the value of the S flag bit in the condition code register (CCR). For branch instructions, the program counter (PC) is not specified in an instruction operand but is specified implicitly.

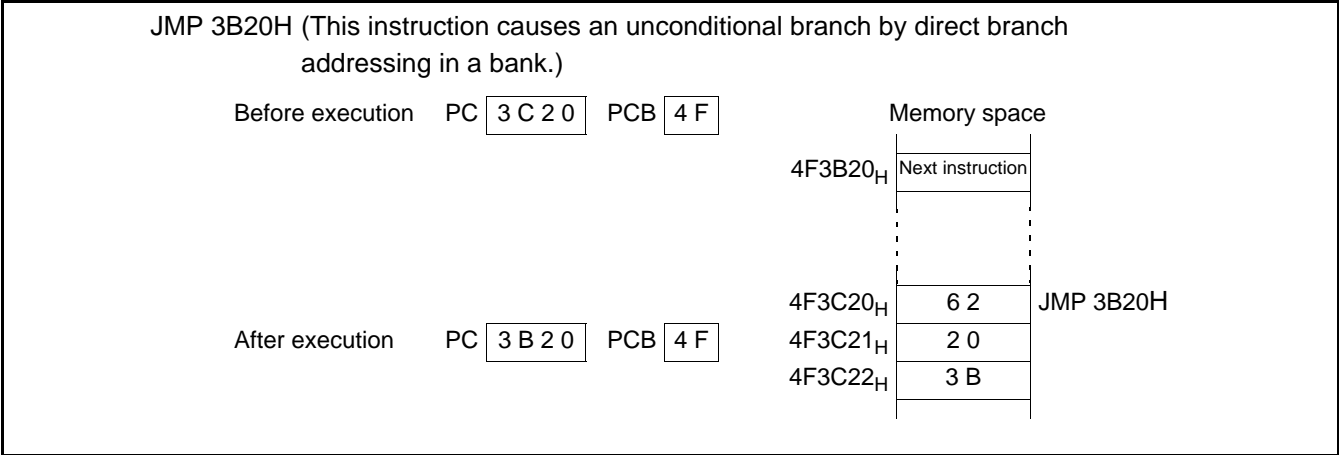
Figure C.3-2 Example of Register Direct Addressing



● Direct branch addressing (addr16)

Specify an offset explicitly for the branch destination address. The size of the offset is 16 bits, which indicates the branch destination in the logical address space. Direct branch addressing is used for an unconditional branch, subroutine call, or software interrupt instruction. Bit23 to bit16 of the address are specified by the program counter bank register (PCB).

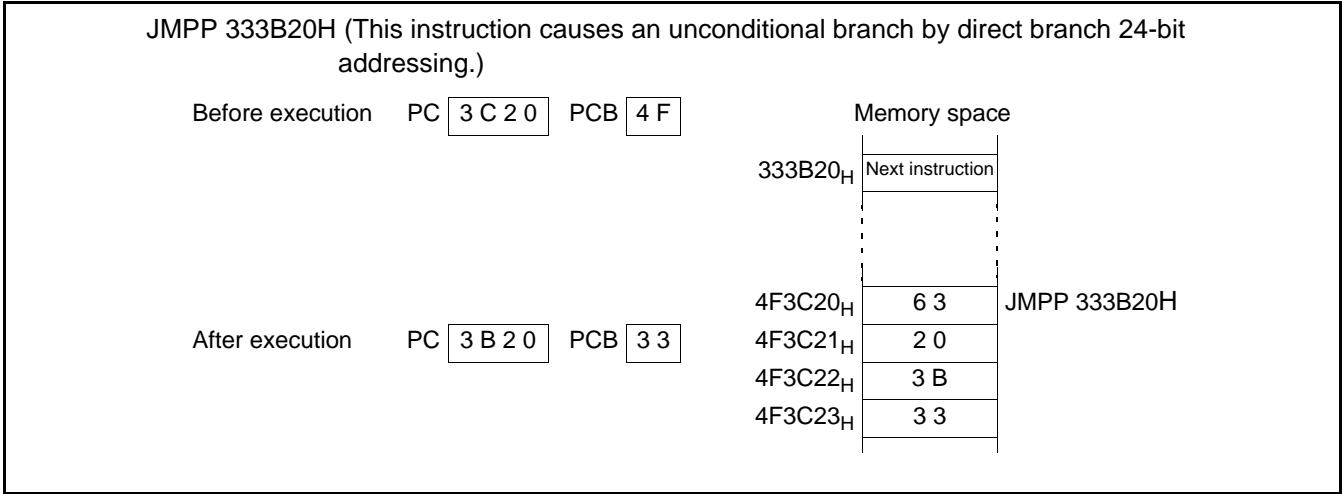
Figure C.3-3 Example of Direct Branch Addressing (addr16)



● Physical direct branch addressing (addr24)

Specify an offset explicitly for the branch destination address. The size of the offset is 24 bits. Physical direct branch addressing is used for unconditional branch, subroutine call, or software interrupt instruction.

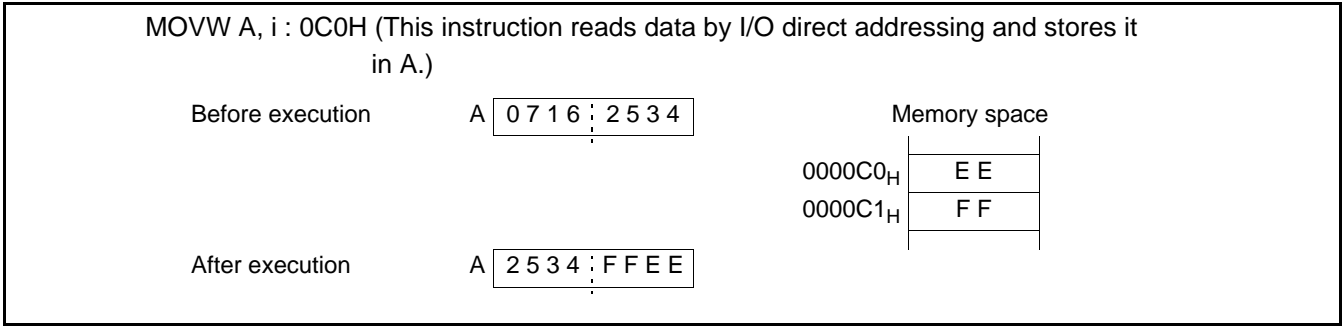
Figure C.3-4 Example of Direct Branch Addressing (addr24)



● I/O direct addressing (io)

Specify an 8-bit offset explicitly for the memory address in an operand. The I/O address space in the physical address space from 000000<sub>H</sub> to 0000FF<sub>H</sub> is accessed regardless of the data bank register (DTB) and direct page register (DPR). A bank select prefix for bank addressing is invalid if specified before an instruction using I/O direct addressing.

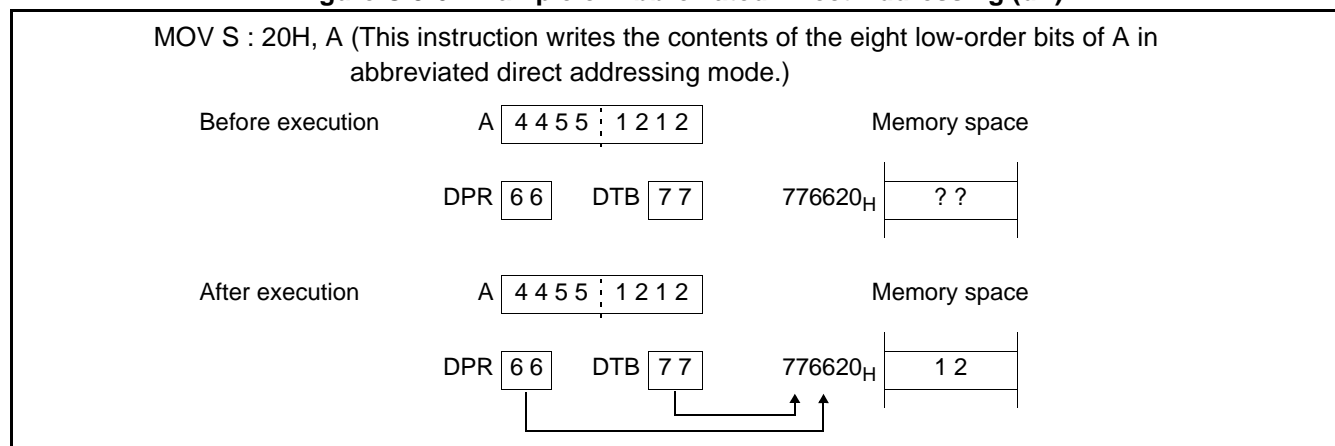
Figure C.3-5 Example of I/O Direct Addressing (io)



## ● Abbreviated direct addressing (dir)

Specify the eight low-order bits of a memory address explicitly in an operand. Address bits 8 to 15 are specified by the direct page register (DPR). Address bits 16 to 23 are specified by the data bank register (DTB).

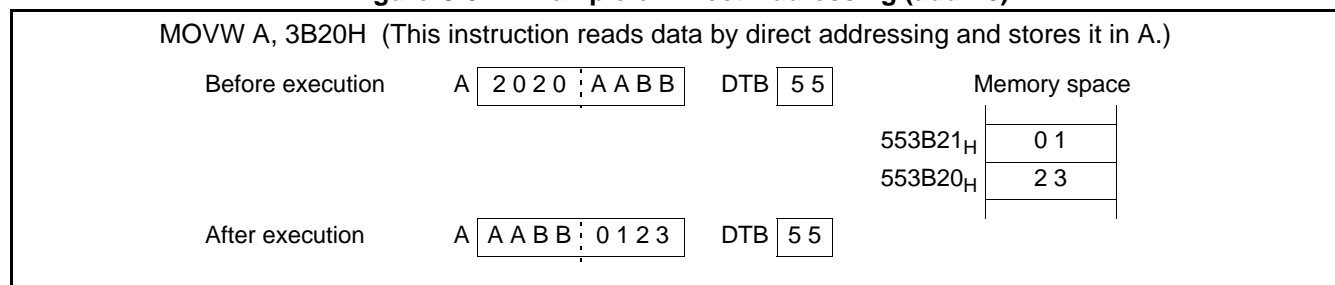
**Figure C.3-6 Example of Abbreviated Direct Addressing (dir)**



## ● Direct addressing (addr16)

Specify the 16 low-order bits of a memory address explicitly in an operand. Address bits 16 to 23 are specified by the data bank register (DTB). A prefix instruction for access space addressing is invalid for this mode of addressing.

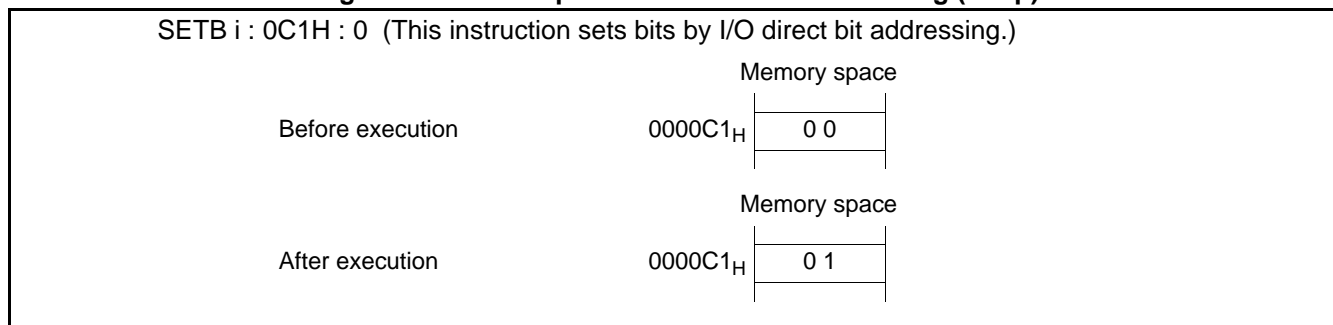
**Figure C.3-7 Example of Direct Addressing (addr16)**



● I/O direct bit addressing (io:bp)

Specify bits in physical addresses 000000<sub>H</sub> to 0000FF<sub>H</sub> explicitly. Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB).

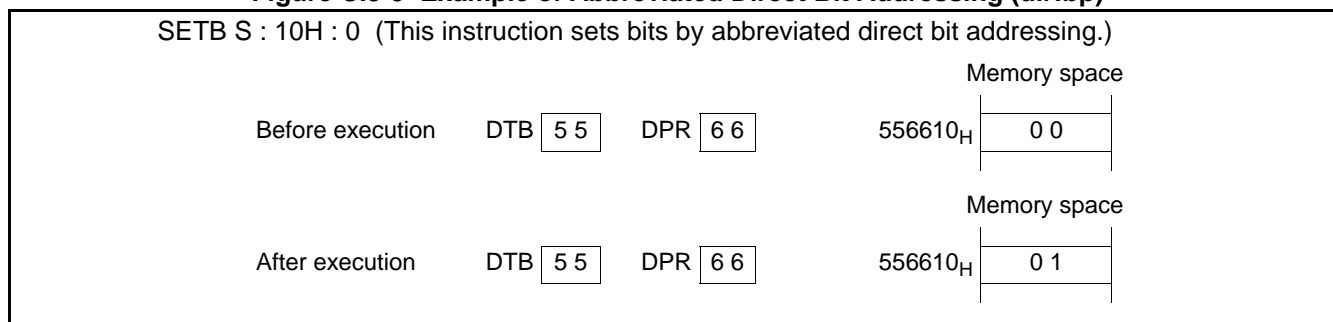
**Figure C.3-8 Example of I/O Direct Bit Addressing (io:bp)**



● Abbreviated direct bit addressing (dir:bp)

Specify the eight low-order bits of a memory address explicitly in an operand. Address bits 8 to 15 are specified by the direct page register (DPR). Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB).

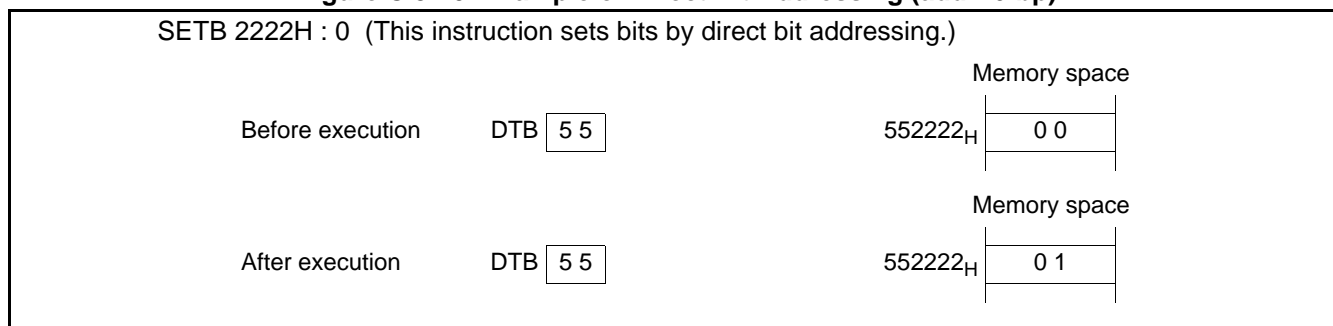
**Figure C.3-9 Example of Abbreviated Direct Bit Addressing (dir:bp)**



● Direct bit addressing (addr16:bp)

Specify arbitrary bits in 64 kilobytes explicitly. Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB).

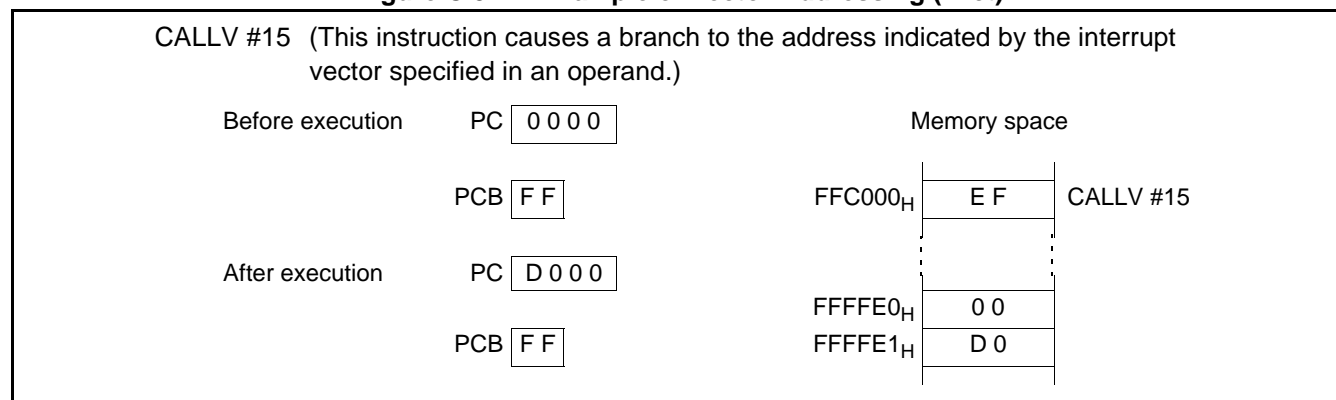
**Figure C.3-10 Example of Direct Bit Addressing (addr16:bp)**



## ● Vector Addressing (#vct)

Specify vector data in an operand to indicate the branch destination address. There are two sizes for vector numbers: 4 bits and 8 bits. Vector addressing is used for a subroutine call or software interrupt instruction.

**Figure C.3-11 Example of Vector Addressing (#vct)**



**Table C.3-2 CALLV Vector List**

Instruction	Vector address L	Vector address H
CALLV #0	XXFFFE <sub>H</sub>	XXFFFF <sub>H</sub>
CALLV #1	XXFFFC <sub>H</sub>	XXFFFD <sub>H</sub>
CALLV #2	XXFFFA <sub>H</sub>	XXFFFB <sub>H</sub>
CALLV #3	XXFFF8 <sub>H</sub>	XXFFF9 <sub>H</sub>
CALLV #4	XXFFF6 <sub>H</sub>	XXFFF7 <sub>H</sub>
CALLV #5	XXFFF4 <sub>H</sub>	XXFFF5 <sub>H</sub>
CALLV #6	XXFFF2 <sub>H</sub>	XXFFF3 <sub>H</sub>
CALLV #7	XXFFF0 <sub>H</sub>	XXFFF1 <sub>H</sub>
CALLV #8	XXFFEE <sub>H</sub>	XXFFEF <sub>H</sub>
CALLV #9	XXFFEC <sub>H</sub>	XXFFED <sub>H</sub>
CALLV #10	XXFFEA <sub>H</sub>	XXFFEB <sub>H</sub>
CALLV #11	XXFFE8 <sub>H</sub>	XXFFE9 <sub>H</sub>
CALLV #12	XXFFE6 <sub>H</sub>	XXFFE7 <sub>H</sub>
CALLV #13	XXFFE4 <sub>H</sub>	XXFFE5 <sub>H</sub>
CALLV #14	XXFFE2 <sub>H</sub>	XXFFE3 <sub>H</sub>
CALLV #15	XXFFE0 <sub>H</sub>	XXFFE1 <sub>H</sub>

Note: A PCB register value is set in XX.

### Note:

When the program counter bank register (PCB) is FF<sub>H</sub>, the vector area overlaps the vector area of INT #vct8 (#0 to #7). Use vector addressing carefully (see Table C.3-2 ).

## C.4 Indirect Addressing

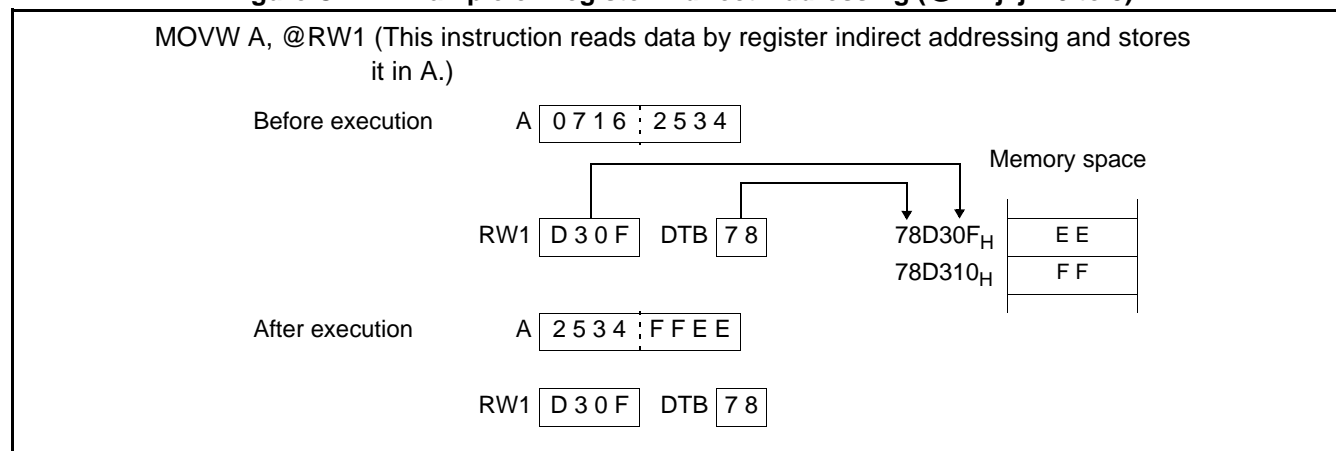
In indirect addressing mode, an address is specified indirectly by the address data of an operand.

### ■ Indirect Addressing

#### ● Register indirect addressing (@RWj j = 0 to 3)

Memory is accessed using the contents of general-purpose register RWj as an address. Address bits 16 to 23 are indicated by the data bank register (DTB) when RW0 or RW1 is used, system stack bank register (SSB) or user stack bank register (USB) when RW3 is used, or additional data bank register (ADB) when RW2 is used.

**Figure C.4-1 Example of Register Indirect Addressing (@RWj j = 0 to 3)**

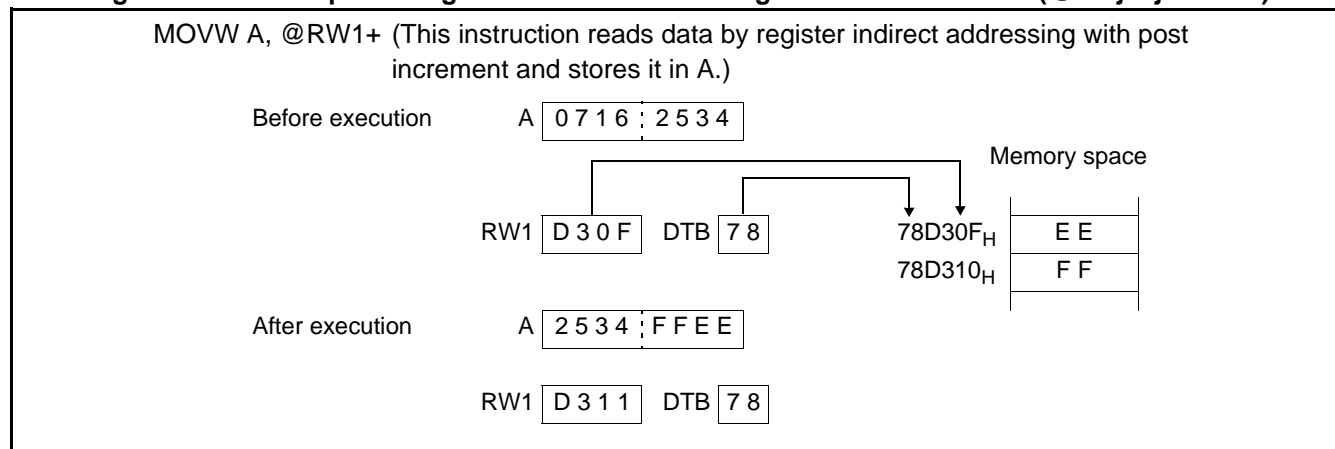


#### ● Register indirect addressing with post increment (@RWj+ j = 0 to 3)

Memory is accessed using the contents of general-purpose register RWj as an address. After operand operation, RWj is incremented by the operand size (1 for a byte, 2 for a word, or 4 for a long word). Address bits 16 to 23 are indicated by the data bank register (DTB) when RW0 or RW1 is used, system stack bank register (SSB) or user stack bank register (USB) when RW3 is used, or additional data bank register (ADB) when RW2 is used.

If the post increment results in the address of the register that specifies the increment, the incremented value is referenced after that. In this case, if the next instruction is a write instruction, priority is given to writing by an instruction and, therefore, the register that would be incremented becomes write data.

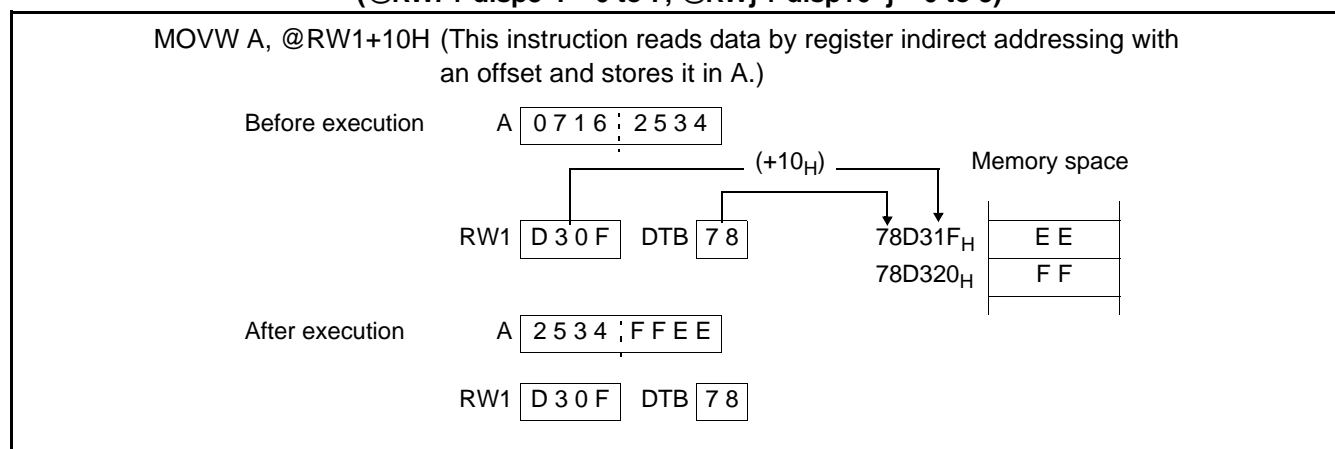
**Figure C.4-2 Example of Register Indirect Addressing with Post Increment (@RWj+ j = 0 to 3)**



● Register indirect addressing with offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)

Memory is accessed using the address obtained by adding an offset to the contents of general-purpose register RWj. Two types of offset, byte and word offsets, are used. They are added as signed numeric values. Address bits 16 to 23 are indicated by the data bank register (DTB) when RW0, RW1, RW4, or RW5 is used, system stack bank register (SSB) or user stack bank register (USB) when RW3 or RW7 is used, or additional data bank register (ADB) when RW2 or RW6 is used.

**Figure C.4-3 Example of Register Indirect Addressing with Offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)**

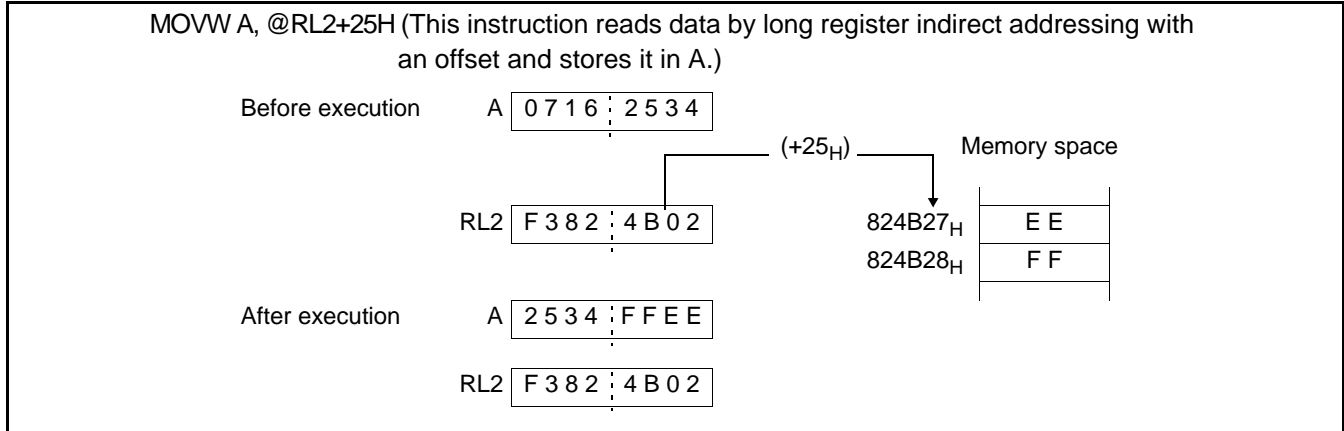




● Long register indirect addressing with offset ( $@RLi + disp8$   $i = 0$  to  $3$ )

Memory is accessed using the address that is the 24 low-order bits obtained by adding an offset to the contents of general-purpose register  $RLi$ . The offset is 8-bits long and is added as a signed numeric value.

**Figure C.4-4 Example of Long Register Indirect Addressing with Offset ( $@RLi + disp8$   $i = 0$  to  $3$ )**

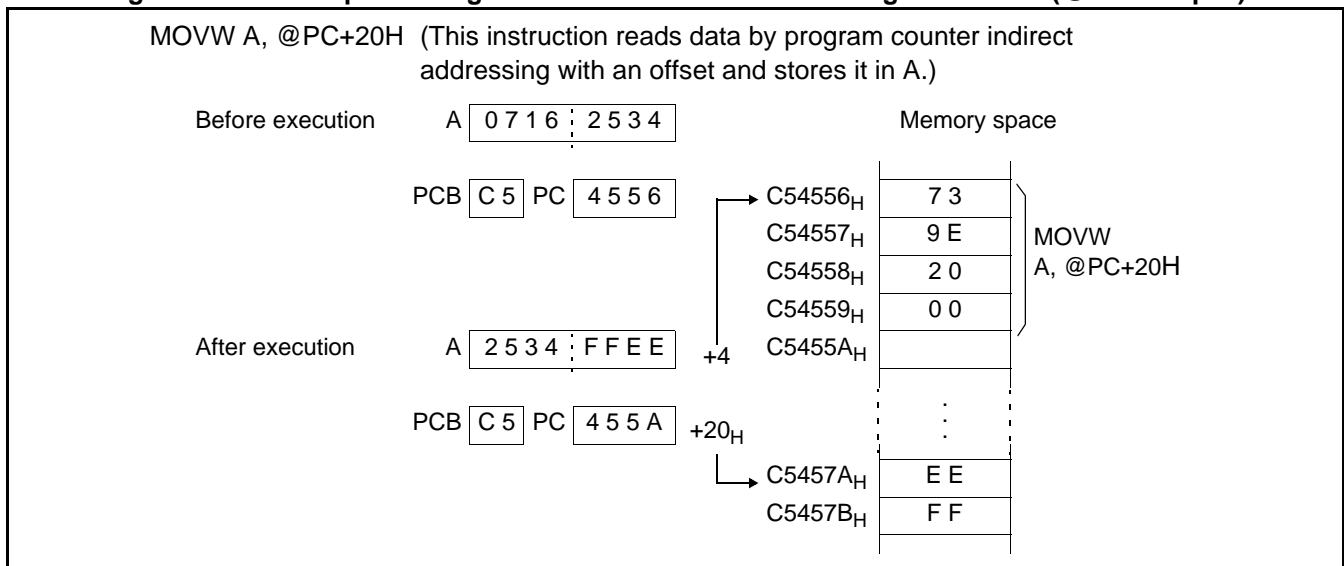


● Program counter indirect addressing with offset ( $@PC + disp16$ )

Memory is accessed using the address indicated by (instruction address + 4 +  $disp16$ ). The offset is one word long. Address bits 16 to 23 are specified by the program counter bank register (PCB). Note that the operand address of each of the following instructions is not deemed to be (next instruction address +  $disp16$ ):

- DBNZ eam, rel
- DWBNZ eam, rel
- CBNE eam, #imm8, rel
- CWBNE eam, #imm16, rel
- MOV eam, #imm8
- MOVW eam, #imm16

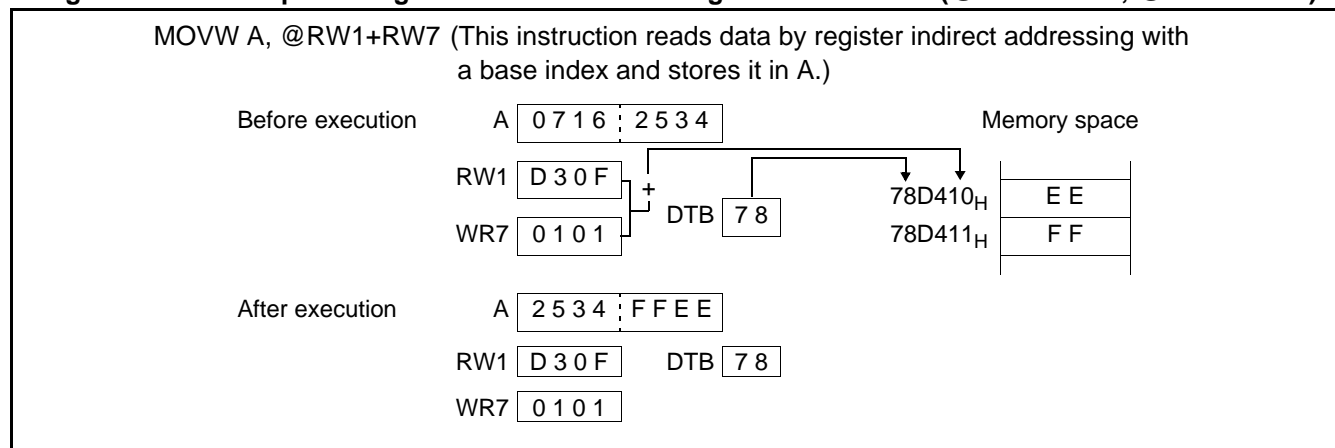
**Figure C.4-5 Example of Program Counter Indirect Addressing with Offset ( $@PC + disp16$ )**



● Register indirect addressing with base index (@RW0 + RW7, @RW1 + RW7)

Memory is accessed using the address determined by adding RW0 or RW1 to the contents of general-purpose register RW7. Address bits 16 to 23 are indicated by the data bank register (DTB).

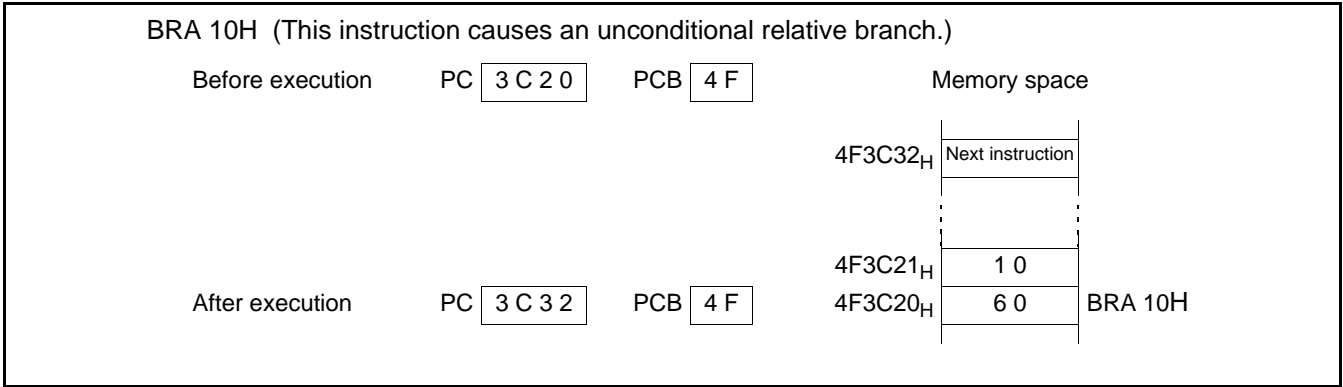
**Figure C.4-6 Example of Register Indirect Addressing with Base Index (@RW0 + RW7, @RW1 + RW7)**



● Program counter relative branch addressing (rel)

The address of the branch destination is a value determined by adding an 8-bit offset to the program counter (PC) value. If the result of addition exceeds 16 bits, bank register incrementing or decrementing is not performed and the excess part is ignored, and therefore the address is contained within a 64-kilobyte bank. This addressing is used for both conditional and unconditional branch instructions. Address bits 16 to 23 are indicated by the program counter bank register (PCB).

Figure C.4-7 Example of Program Counter Relative Branch Addressing (rel)



● Register list (rlst)

Specify a register to be pushed onto or popped from a stack.

Figure C.4-8 Configuration of the Register List

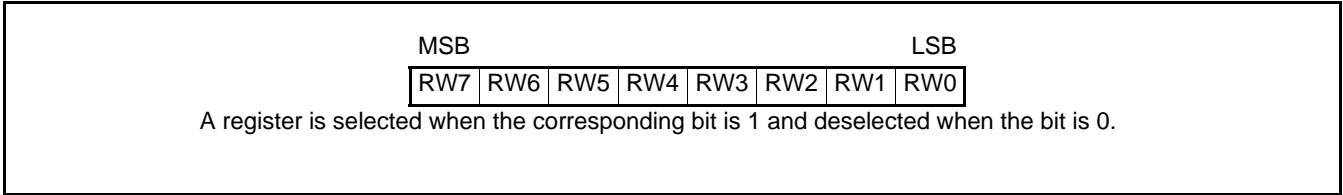
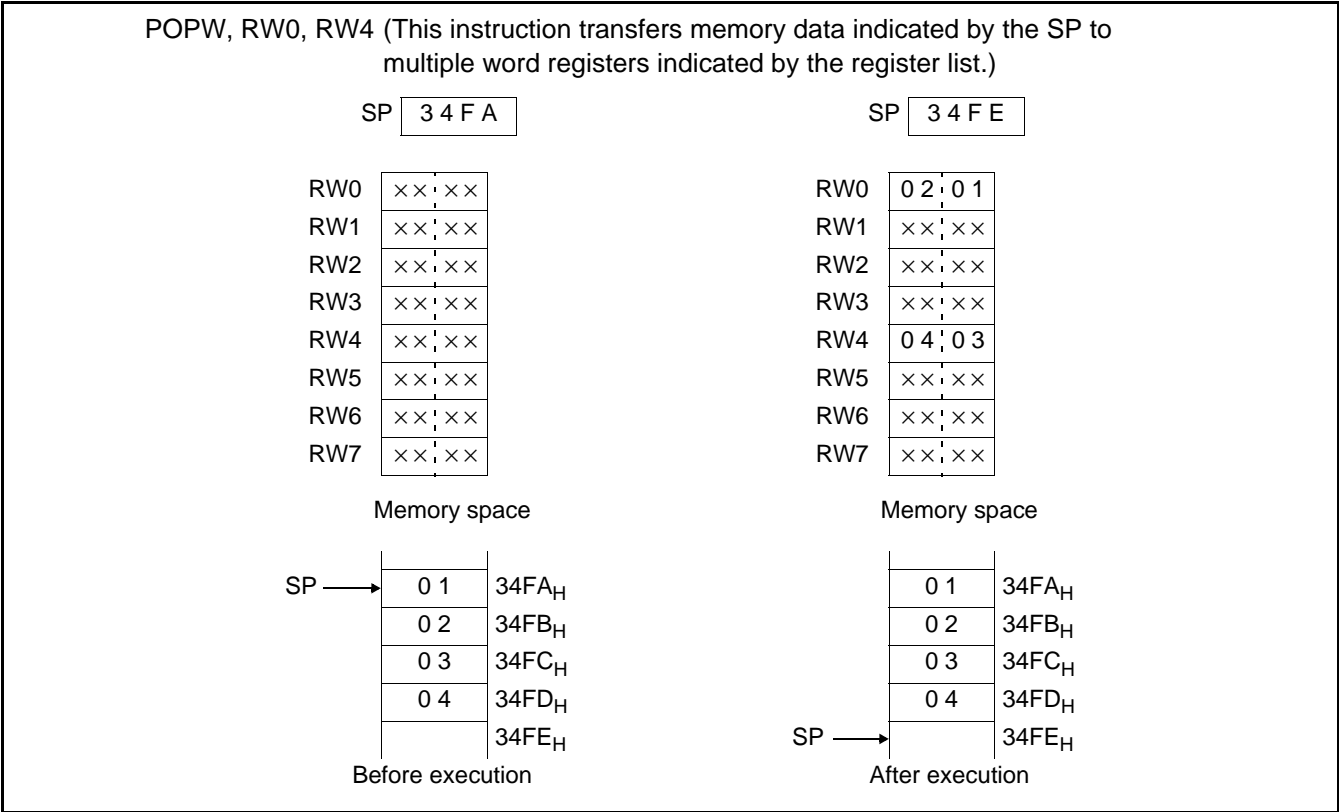


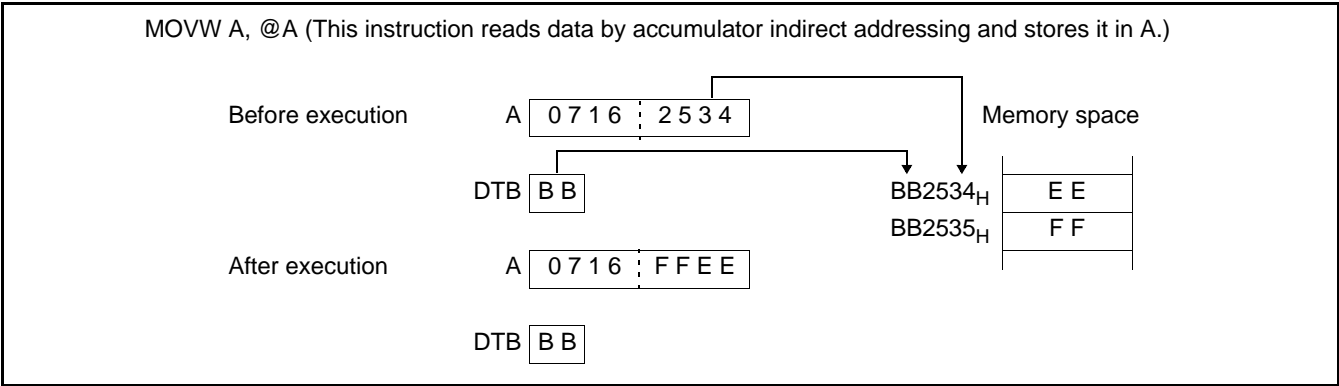
Figure C.4-9 Example of Register List (rlist)



● Accumulator indirect addressing (@A)

Memory is accessed using the address indicated by the contents of the low-order bytes (16 bits) of the accumulator (AL). Address bits 16 to 23 are specified by a mnemonic in the data bank register (DTB).

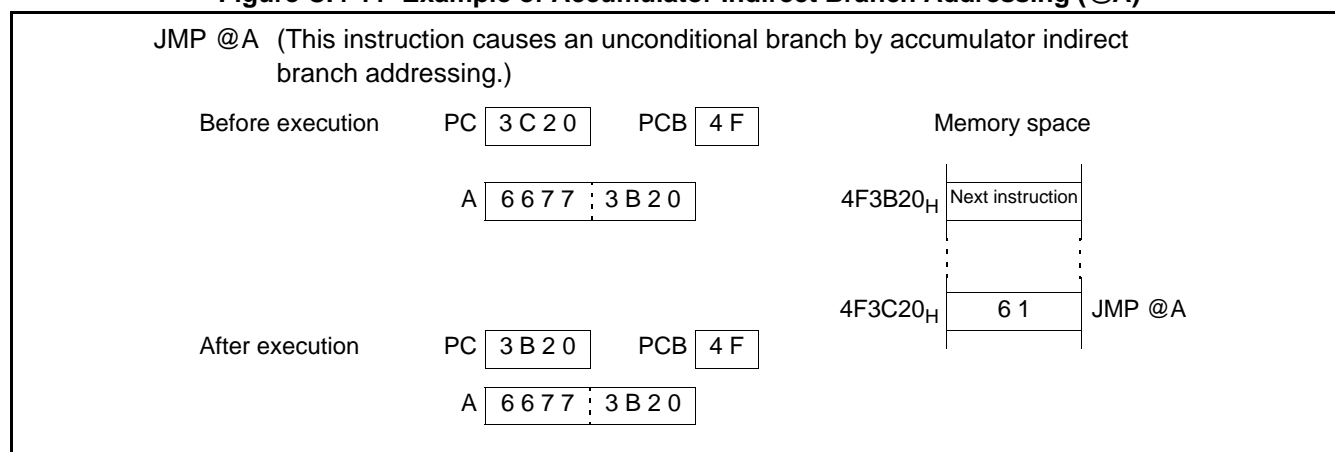
Figure C.4-10 Example of Accumulator Indirect Addressing (@A)



● Accumulator indirect branch addressing (@A)

The address of the branch destination is the content (16 bits) of the low-order bytes (AL) of the accumulator. It indicates the branch destination in the bank address space. Address bits 16 to 23 are specified by the program counter bank register (PCB). For the Jump Context (JCTX) instruction, however, address bits 16 to 23 are specified by the data bank register (DTB). This addressing is used for unconditional branch instructions.

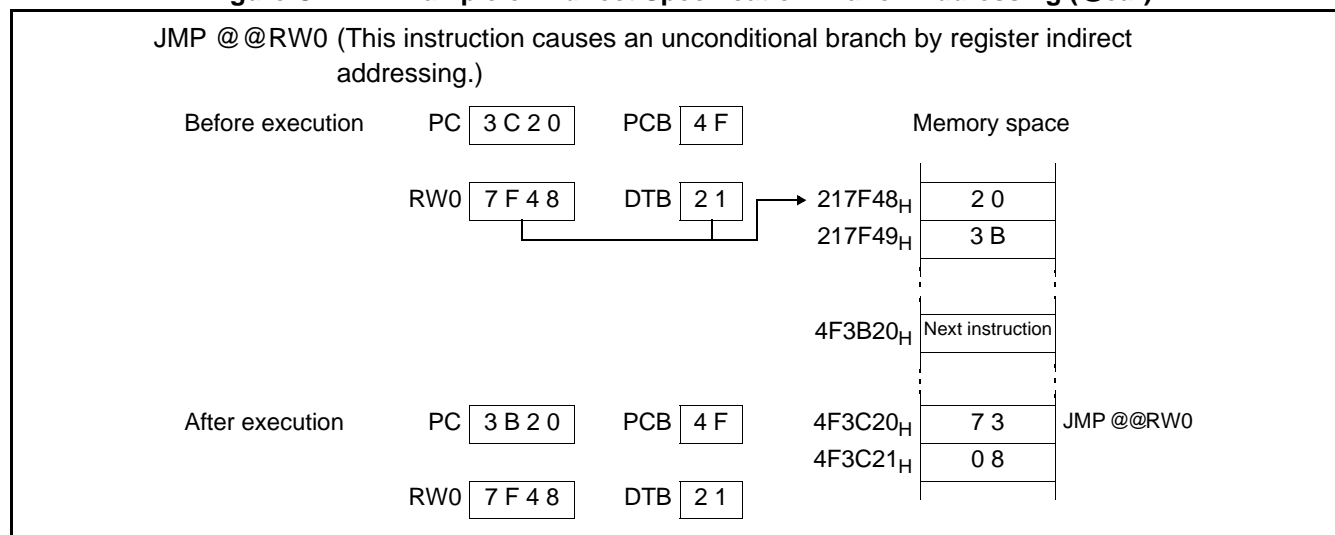
**Figure C.4-11 Example of Accumulator Indirect Branch Addressing (@A)**



● Indirect specification branch addressing (@ear)

The address of the branch destination is the word data at the address indicated by ear.

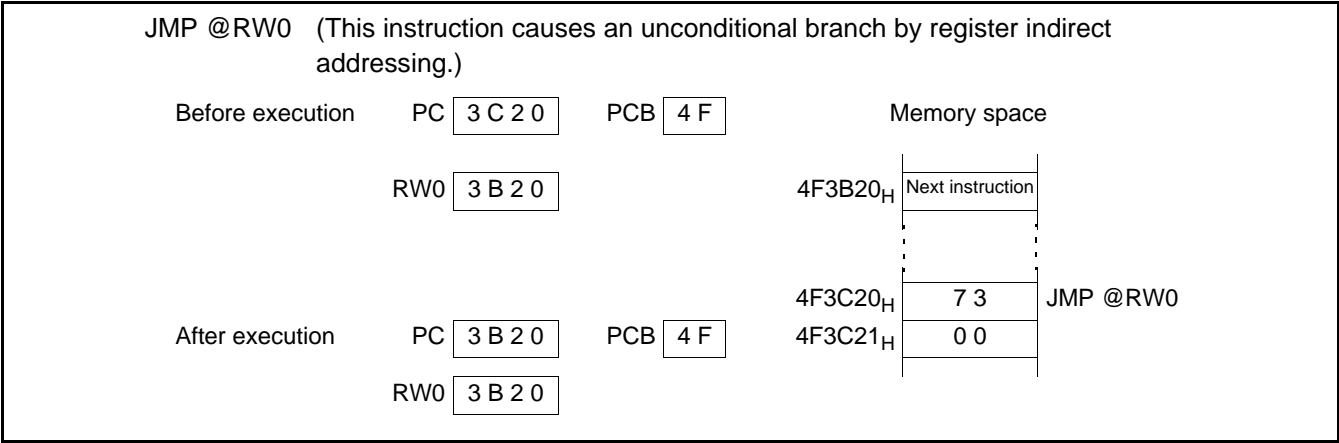
**Figure C.4-12 Example of Indirect Specification Branch Addressing (@ear)**



● Indirect specification branch addressing (@eam)

The address of the branch destination is the word data at the address indicated by eam.

Figure C.4-13 Example of Indirect Specification Branch Addressing (@eam)



## **C.5 Execution Cycle Count**

---

**The number of cycles required for instruction execution (execution cycle count) is obtained by adding the number of cycles required for each instruction, "correction value" determined by the condition, and the number of cycles for instruction fetch.**

---

### **■ Execution Cycle Count**

The number of cycles required for instruction execution (execution cycle count) is obtained by adding the number of cycles required for each instruction, "correction value" determined by the condition, and the number of cycles for instruction fetch. In the mode of fetching an instruction from memory such as internal ROM connected to a 16-bit bus, the program fetches the instruction being executed in word increments. Therefore, intervening in data access increases the execution cycle count.

Similarly, in the mode of fetching an instruction from memory connected to an 8-bit external bus, the program fetches every byte of an instruction being executed. Therefore, intervening in data access increases the execution cycle count. In CPU intermittent operation mode, access to a general-purpose register, internal ROM, internal RAM, internal I/O, or external data bus causes the clock to the CPU to halt for the cycle count specified by the CG0 and CG1 bits of the low power consumption mode control register. Therefore, for the cycle count required for instruction execution in CPU intermittent operation mode, add the "access count x cycle count for the halt" as a correction value to the normal execution count.

## ■ Calculating the Execution Cycle Count

Table C.5-1 lists execution cycle counts and Table C.5-2 and Table C.5-3 summarize correction value data.

**Table C.5-1 Execution Cycle Counts in Each Addressing Mode**

Code	Operand	(a) *	Register access count in each addressing mode
		Execution cycle count in each addressing mode	
00   07	Ri Rwi RLi	See the instruction list.	See the instruction list.
08   0B	@RWj	2	1
0C   0F	@RWj+	4	2
10   17	@RWi+disp8	2	1
18   1B	@RWi+disp16	2	1
1C 1D 1E 1F	@RW0+RW7 @RW1+RW7 @PC+disp16 addr16	4 4 2 1	2 2 0 0

\*: (a) is used for ~ (cycle count) and B (correction value) in "C.8 F2MC-16LX Instruction List".



**Table C.5-2 Cycle Count Correction Values for Counting Execution Cycles**

Operand	(b) byte *		(c) word *		(d) long *	
	Cycle count	Access count	Cycle count	Access count	Cycle count	Access count
Internal register	+0	1	+0	1	+0	2
Internal memory Even address	+0	1	+0	1	+0	2
Internal memory Odd address	+0	1	+2	2	+4	4
External data bus 16-bit even address	+1	1	+1	1	+2	2
External data bus 16-bit odd address	+1	1	+4	2	+8	4
External data bus 8-bits	+1	1	+4	2	+8	4

\*: (b), (c), and (d) are used for ~ (cycle count) and B (correction value) in "C.8 F2MC-16LX Instruction List".

**Note:**

When an external data bus is used, the cycle counts during which an instruction is made to wait by ready input or automatic ready must also be added.

**Table C.5-3 Cycle Count Correction Values for Counting Instruction Fetch Cycles**

Instruction	Byte boundary	Word boundary
Internal memory	-	+2
External data bus 16-bits	-	+3
External data bus 8-bits	+3	-

**Notes:**

- When an external data bus is used, the cycle counts during which an instruction is made to wait by ready input or automatic ready must also be added.
- Actually, instruction execution is not delayed by every instruction fetch. Therefore, use the correction values to calculate the worst case.

## C.6 Effective address field

Table C.6-1 shows the effective address field.

■ Effective Address Field

Table C.6-1 Effective Address Field

Code	Representation			Address format	Byte count of extended address part *
00	R0	RW0	RL0	Register direct: Individual parts correspond to the byte, word, and long word types in order from the left.	-
01	R1	RW1	(RL0)		
02	R2	RW2	RL1		
03	R3	RW3	(RL1)		
04	R4	RW4	RL2		
05	R5	RW5	(RL2)		
06	R6	RW6	RL3		
07	R7	RW7	(RL3)		
08	@RW0			Register indirect	0
09	@RW1				
0A	@RW2				
0B	@RW3				
0C	@RW0+			Register indirect with post increment	0
0D	@RW1+				
0E	@RW2+				
0F	@RW3+				
10	@RW0+disp8			Register indirect with 8-bit displacement	1
11	@RW1+disp8				
12	@RW2+disp8				
13	@RW3+disp8				
14	@RW4+disp8				
15	@RW5+disp8				
16	@RW6+disp8				
17	@RW7+disp8				
18	@RW0+disp16			Register indirect with 16-bit displacement	2
19	@RW1+disp16				
1A	@RW2+disp16				
1B	@RW3+disp16				
1C	@RW0+RW7			Register indirect with index	0
1D	@RW1+RW7			Register indirect with index	0
1E	@PC+disp16			PC indirect with 16-bit displacement	2
1F	addr16			Direct address	2

\*1: Each byte count of the extended address part applies to + in the # (byte count) column in "C.8 F2MC-16LX Instruction List".

## C.7 How to Read the Instruction List

Table C.7-1 describes the items used in "C.8 F2MC-16LX Instruction List", and Table C.7-2 describes the symbols used in the same list.

### ■ Description of Instruction Presentation Items and Symbols

**Table C.7-1 Description of Items in the Instruction List (1/2)**

Item	Description
Mnemonic	Uppercase, symbol: Represented as is in the assembler. Lowercase: Rewritten in the assembler. Number of following lowercase: Indicates bit length in the instruction.
#	Indicates the number of bytes.
~	Indicates the number of cycles. See Table C.2-1 for the alphabetical letters in items.
RG	Indicates the number of times a register access is performed during instruction execution. The number is used to calculate the correction value for CPU intermittent operation.
B	Indicates the correction value used to calculate the actual number of cycles during instruction execution. The actual number of cycles during instruction execution can be determined by adding the value in the ~ column to this value.
Operation	Indicates the instruction operation.
LH	Indicates the special operation for bit15 to bit08 of the accumulator. Z: Transfers 0. X: Transfers after sign extension. -: No transfer
AH	Indicates the special operation for the 16 high-order bits of the accumulator. *: Transfers from AL to AH. -: No transfer Z: Transfers 00 to AH. X: Transfers 00 <sub>H</sub> or FF <sub>H</sub> to AH after AL sign extension.

**Table C.7-1 Description of Items in the Instruction List (1/2)**

Item	Description
I	Each indicates the state of each flag: I (interrupt enable), S (stack), T (sticky bit), N (negative), Z (zero), V (overflow), C (carry). *: Changes upon instruction execution. -: No change S: Set upon instruction execution. R: Reset upon instruction execution.
S	
T	
N	
Z	
V	
C	
RMW	Indicates whether the instruction is a Read Modify Write instruction (reading data from memory by the I instruction and writing the result to memory). *: Read Modify Write instruction -: Not Read Modify Write instruction <b>Note:</b> Cannot be used for an address that has different meanings between read and write operations.

**Table C.7-2 Explanation on Symbols in the Instruction List (1/2)**

Symbol	Explanation
A	The bit length used varies depending on the 32-bit accumulator instruction. Byte: Low-order 8 bits of byte AL Word: 16 bits of word AL Long word: 32 bits of AL and AH
AH	16 high-order bits of A
AL	16 low-order bits of A
SP	Stack pointer (USP or SSP)
PC	Program counter
PCB	program counter bank register
DTB	Data bank register
ADB	Additional data bank register
SSB	System stack bank register
USB	User stack bank register
SPB	Current stack bank register (SSB or USB)
DPR	Direct page register
brg1	DTB, ADB, SSB, USB, DPR, PCB, SPB
brg2	DTB, ADB, SSB, USB, DPR, SPB

**Table C.7-2 Explanation on Symbols in the Instruction List (1/2)**

Symbol	Explanation
Ri	R0, R1, R2, R3, R4, R5, R6, R7
RWi	RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7
RWj	RW0, RW1, RW2, RW3
RLi	RL0, RL1, RL2, RL3
dir	Abbreviated direct addressing
addr16	Direct addressing
addr24	Physical direct addressing
ad24 0-15	Bit0 to bit15 of addr24
ad24 16-23	Bit16 to bit23 of addr24
io	I/O area (000000 <sub>H</sub> to 0000FF <sub>H</sub> )
#imm4	4-bit immediate data
#imm8	8-bit immediate data
#imm16	16-bit immediate data
#imm32	32-bit immediate data
ext (imm8)	16-bit data obtained by sign extension of 8-bit immediate data
disp8	8-bit displacement
disp16	16-bit displacement
bp	Bit offset
vct4	Vector number (0 to 15)
vct8	Vector number (0 to 255)
( ) b	Bit address
rel	PC relative branch
ear	Effective addressing (code 00 to 07)
eam	Effective addressing (code 08 to 1F)
rlst	Register list

C.8 F<sup>2</sup>MC-16LX Instruction List

Table C.8-1 to Table C.8-18 list the instructions used by the F<sup>2</sup>MC-16LX.

■ F<sup>2</sup>MC-16LX Instruction List

Table C.8-1 41 Transfer Instructions (Byte)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOV A,dir	2	3	0	(b)	byte (A) ← (dir)	Z	*	-	-	-	*	*	-	-	-
MOV A,addr16	3	4	0	(b)	byte (A) ← (addr16)	Z	*	-	-	-	*	*	-	-	-
MOV A,Ri	1	2	1	0	byte (A) ← (Ri)	Z	*	-	-	-	*	*	-	-	-
MOV A,ear	2	2	1	0	byte (A) ← (ear)	Z	*	-	-	-	*	*	-	-	-
MOV A,eam	2+	3 + (a)	0	(b)	byte (A) ← (eam)	Z	*	-	-	-	*	*	-	-	-
MOV A,io	2	3	0	(b)	byte (A) ← (io)	Z	*	-	-	-	*	*	-	-	-
MOV A,#imm8	2	2	0	0	byte (A) ← imm8	Z	*	-	-	-	*	*	-	-	-
MOV A,@A	2	3	0	(b)	byte (A) ← ((A))	Z	-	-	-	-	*	*	-	-	-
MOV A,@RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	Z	*	-	-	-	*	*	-	-	-
MOVN A,#imm4	1	1	0	0	byte (A) ← imm4	Z	*	-	-	-	R	*	-	-	-
MOVX A,dir	2	3	0	(b)	byte (A) ← (dir)	X	*	-	-	-	*	*	-	-	-
MOVX A,addr16	3	4	0	(b)	byte (A) ← (addr16)	X	*	-	-	-	*	*	-	-	-
MOVX A,Ri	2	2	1	0	byte (A) ← (Ri)	X	*	-	-	-	*	*	-	-	-
MOVX A,ear	2	2	1	0	byte (A) ← (ear)	X	*	-	-	-	*	*	-	-	-
MOVX A,eam	2+	3 + (a)	0	(b)	byte (A) ← (eam)	X	*	-	-	-	*	*	-	-	-
MOVX A,io	2	3	0	(b)	byte (A) ← (io)	X	*	-	-	-	*	*	-	-	-
MOVX A,#imm8	2	2	0	0	byte (A) ← imm8	X	*	-	-	-	*	*	-	-	-
MOVX A,@A	2	3	0	(b)	byte (A) ← ((A))	X	-	-	-	-	*	*	-	-	-
MOVX A,@RWi+disp8	2	5	1	(b)	byte (A) ← ((RWi)+disp8)	X	*	-	-	-	*	*	-	-	-
MOVX A,@RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	X	*	-	-	-	*	*	-	-	-
MOV dir,A	2	3	0	(b)	byte (dir) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV addr16,A	3	4	0	(b)	byte (addr16) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV Ri,A	1	2	1	0	byte (Ri) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV ear,A	2	2	1	0	byte (ear) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV eam,A	2+	3 + (a)	0	(b)	byte (eam) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV io,A	2	3	0	(b)	byte (io) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV @RLi+disp8,A	3	10	2	(b)	byte ((RLi)+disp8) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV Ri,ear	2	3	2	0	byte (Ri) ← (ear)	-	-	-	-	-	*	*	-	-	-
MOV Ri,eam	2+	4 + (a)	1	(b)	byte (Ri) ← (eam)	-	-	-	-	-	*	*	-	-	-
MOV ear,Ri	2	4	2	0	byte (ear) ← (Ri)	-	-	-	-	-	*	*	-	-	-
MOV eam,Ri	2+	5 + (a)	1	(b)	byte (eam) ← (Ri)	-	-	-	-	-	*	*	-	-	-
MOV Ri,#imm8	2	2	1	0	byte (Ri) ← imm8	-	-	-	-	-	*	*	-	-	-
MOV io,#imm8	3	5	0	(b)	byte (io) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV dir,#imm8	3	5	0	(b)	byte (dir) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV ear,#imm8	3	2	1	0	byte (ear) ← imm8	-	-	-	-	-	*	*	-	-	-
MOV eam,#imm8	3+	4 + (a)	0	(b)	byte (eam) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV @AL,AH	2	3	0	(b)	byte ((A)) ← (AH)	-	-	-	-	-	*	*	-	-	-
XCH A,ear	2	4	2	0	byte (A) ↔ (ear)	Z	-	-	-	-	-	-	-	-	-
XCH A,eam	2+	5 + (a)	0	2 × (b)	byte (A) ↔ (eam)	Z	-	-	-	-	-	-	-	-	-
XCH Ri,ear	2	7	4	0	byte (Ri) ↔ (ear)	-	-	-	-	-	-	-	-	-	-
XCH Ri,eam	2+	9 + (a)	2	2 × (b)	byte (Ri) ↔ (eam)	-	-	-	-	-	-	-	-	-	-

Note:

See Table C.5-1 and Table C.5-2 for information on (a) and (b) in the table.

Table C.8-2 38 Transfer Instructions (Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVW A,dir	2	3	0	(c)	word (A) ← (dir)	-	*	-	-	-	*	*	-	-	-
MOVW A,addr16	3	4	0	(c)	word (A) ← (addr16)	-	*	-	-	-	*	*	-	-	-
MOVW A,SP	1	1	0	0	word (A) ← (SP)	-	*	-	-	-	*	*	-	-	-
MOVW A,RWi	1	2	1	0	word (A) ← (RWi)	-	*	-	-	-	*	*	-	-	-
MOVW A,ear	2	2	1	0	word (A) ← (ear)	-	*	-	-	-	*	*	-	-	-
MOVW A,eam	2+	3 + (a)	0	(c)	word (A) ← (eam)	-	*	-	-	-	*	*	-	-	-
MOVW A,io	2	3	0	(c)	word (A) ← (io)	-	*	-	-	-	*	*	-	-	-
MOVW A,@A	2	3	0	(c)	word (A) ← ((A))	-	-	-	-	-	*	*	-	-	-
MOVW A,#imm16	3	2	0	0	word (A) ← imm16	-	*	-	-	-	*	*	-	-	-
MOVW A,@RWi+disp8	2	5	1	(c)	word (A) ← ((RWi)+disp8)	-	*	-	-	-	*	*	-	-	-
MOVW A,@RLi+disp8	3	10	2	(c)	word (A) ← ((RLi)+disp8)	-	*	-	-	-	*	*	-	-	-
MOVW dir,A	2	3	0	(c)	word (dir) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW addr16,A	3	4	0	(c)	word (addr16) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW SPA	1	1	0	0	word (SP) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW RWi,A	1	2	1	0	word (RWi) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW ear,A	2	2	1	0	word (ear) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW eam,A	2+	3 + (a)	0	(c)	word (eam) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW io,A	2	3	0	(c)	word (io) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW @RWi+disp8,A	2	5	1	(c)	word ((RWi)+disp8) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW @RLi+disp8,A	3	10	2	(c)	word ((RLi)+disp8) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW RWi,ear	2	3	2	0	word (RWi) ← (ear)	-	-	-	-	-	*	*	-	-	-
MOVW RWi,eam	2+	4 + (a)	1	(c)	word (RWi) ← (eam)	-	-	-	-	-	*	*	-	-	-
MOVW ear,RWi	2	4	2	0	word (ear) ← (RWi)	-	-	-	-	-	*	*	-	-	-
MOVW eam,RWi	2+	5 + (a)	1	(c)	word (eam) ← (RWi)	-	-	-	-	-	*	*	-	-	-
MOVW RWi,#imm16	3	2	1	0	word (RWi) ← imm16	-	-	-	-	-	*	*	-	-	-
MOVW io,#imm16	4	5	0	(c)	word (io) ← imm16	-	-	-	-	-	-	-	-	-	-
MOVW ear,#imm16	4	2	1	0	word (ear) ← imm16	-	-	-	-	-	*	*	-	-	-
MOVW eam,#imm16	4+	4 + (a)	0	(c)	word (eam) ← imm16	-	-	-	-	-	-	-	-	-	-
MOVW @AL,AH	2	3	0	(c)	word ((A)) ← (AH)	-	-	-	-	-	*	*	-	-	-
XCHW A,ear	2	4	2	0	word (A) ↔ (ear)	-	-	-	-	-	-	-	-	-	-
XCHW A,eam	2+	5 + (a)	0	2 × (c)	word (A) ↔ (eam)	-	-	-	-	-	-	-	-	-	-
XCHW RWi, ear	2	7	4	0	word (RWi) ↔ (ear)	-	-	-	-	-	-	-	-	-	-
XCHW RWi, eam	2+	9 + (a)	2	2 × (c)	word (RWi) ↔ (eam)	-	-	-	-	-	-	-	-	-	-
MOVL A,ear	2	4	2	0	long (A) ← (ear)	-	-	-	-	-	*	*	-	-	-
MOVL A,eam	2+	5 + (a)	0	(d)	long (A) ← (eam)	-	-	-	-	-	*	*	-	-	-
MOVL A,#imm32	5	3	0	0	long (A) ← imm32	-	-	-	-	-	*	*	-	-	-
MOVL ear,A	2	4	2	0	long (ear) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVL eam,A	2+	5 + (a)	0	(d)	long(eam) ← (A)	-	-	-	-	-	*	*	-	-	-

Note:

See Table C.5-1 and Table C.5-2 for information on (a), (c), and (d) in the table.

Table C.8-3 42 Addition/Subtraction Instructions (Byte, Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ADD A,#imm8	2	2	0	0	byte (A) ← (A) + imm8	Z	-	-	-	-	*	*	*	*	-
ADD A,dir	2	5	0	(b)	byte (A) ← (A) + (dir)	Z	-	-	-	-	*	*	*	*	-
ADD A,ear	2	3	1	0	byte (A) ← (A) + (ear)	Z	-	-	-	-	*	*	*	*	-
ADD A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) + (eam)	Z	-	-	-	-	*	*	*	*	-
ADD ear,A	2	3	2	0	byte (ear) ← (ear) + (A)	-	-	-	-	-	*	*	*	*	-
ADD eam,A	2+	5 + (a)	0	2 × (b)	byte (eam) ← (eam) + (A)	Z	-	-	-	-	*	*	*	*	*
ADDC A	1	2	0	0	byte (A) ← (AH) + (AL) + (C)	Z	-	-	-	-	*	*	*	*	-
ADDC A,ear	2	3	1	0	byte (A) ← (A) + (ear) + (C)	Z	-	-	-	-	*	*	*	*	-
ADDC A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) + (eam) + (C)	Z	-	-	-	-	*	*	*	*	-
ADDC A	1	3	0	0	byte (A) ← (AH) + (AL) + (C) (decimal)	Z	-	-	-	-	*	*	*	*	-
SUB A,#imm8	2	2	0	0	byte (A) ← (A) - imm8	Z	-	-	-	-	*	*	*	*	-
SUB A,dir	2	5	0	(b)	byte (A) ← (A) - (dir)	Z	-	-	-	-	*	*	*	*	-
SUB A,ear	2	3	1	0	byte (A) ← (A) - (ear)	Z	-	-	-	-	*	*	*	*	-
SUB A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) - (eam)	Z	-	-	-	-	*	*	*	*	-
SUB ear,A	2	3	2	0	byte (ear) ← (ear) - (A)	-	-	-	-	-	*	*	*	*	-
SUB eam,A	2+	5 + (a)	0	2 × (b)	byte (eam) ← (eam) - (A)	-	-	-	-	-	*	*	*	*	*
SUBC A	1	2	0	0	byte (A) ← (AH) - (AL) - (C)	Z	-	-	-	-	*	*	*	*	-
SUBC A,ear	2	3	1	0	byte (A) ← (A) - (ear) - (C)	Z	-	-	-	-	*	*	*	*	-
SUBC A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) - (eam) - (C)	Z	-	-	-	-	*	*	*	*	-
SUBC A	1	3	0	0	byte (A) ← (AH) - (AL) - (C) (decimal)	Z	-	-	-	-	*	*	*	*	-
ADDW A	1	2	0	0	word (A) ← (AH) + (AL)	-	-	-	-	-	*	*	*	*	-
ADDW A,ear	2	3	1	0	word (A) ← (A) + (ear)	-	-	-	-	-	*	*	*	*	-
ADDW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) + (eam)	-	-	-	-	-	*	*	*	*	-
ADDW A,#imm16	3	2	0	0	word (A) ← (A) + imm16	-	-	-	-	-	*	*	*	*	-
ADDW ear,A	2	3	2	0	word (ear) ← (ear) + (A)	-	-	-	-	-	*	*	*	*	-
ADDW eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) + (A)	-	-	-	-	-	*	*	*	*	*
ADDCW A,ear	2	3	1	0	word (A) ← (A) + (ear) + (C)	-	-	-	-	-	*	*	*	*	-
ADDCW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) + (eam) + (C)	-	-	-	-	-	*	*	*	*	-
SUBW A	1	2	0	0	word (A) ← (AH) - (AL)	-	-	-	-	-	*	*	*	*	-
SUBW A,ear	2	3	1	0	word (A) ← (A) - (ear)	-	-	-	-	-	*	*	*	*	-
SUBW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) - (eam)	-	-	-	-	-	*	*	*	*	-
SUBW A,#imm16	3	2	0	0	word (A) ← (A) - imm16	-	-	-	-	-	*	*	*	*	-
SUBW ear,A	2	3	2	0	word (ear) ← (ear) - (A)	-	-	-	-	-	*	*	*	*	-
SUBW eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) - (A)	-	-	-	-	-	*	*	*	*	*
SUBCW A,ear	2	3	1	0	word (A) ← (A) - (ear) - (C)	-	-	-	-	-	*	*	*	*	-
SUBCW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) - (eam) - (C)	-	-	-	-	-	*	*	*	*	-
ADDL A,ear	2	6	2	0	long (A) ← (A) + (ear)	-	-	-	-	-	*	*	*	*	-
ADDL A,eam	2+	7+(a)	0	(d)	long (A) ← (A) + (eam)	-	-	-	-	-	*	*	*	*	-
ADDL A,#imm32	5	4	0	0	long (A) ← (A) + imm32	-	-	-	-	-	*	*	*	*	-
SUBL A,ear	2	6	2	0	long (A) ← (A) - (ear)	-	-	-	-	-	*	*	*	*	-
SUBL A,eam	2+	7+(a)	0	(d)	long (A) ← (A) - (eam)	-	-	-	-	-	*	*	*	*	-
SUBL A,#imm32	5	4	0	0	long (A) ← (A) - imm32	-	-	-	-	-	*	*	*	*	-

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (d) in the table.



**Table C.8-4 12 Increment/decrement Instructions (Byte, Word, Long Word)**

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
INC	ear	2	3	2	0	byte (ear) $\leftarrow$ (ear) + 1	-	-	-	-	-	*	*	*	-	-
INC	eam	2+	5+(a)	0	2 $\times$ (b)	byte (eam) $\leftarrow$ (eam) + 1	-	-	-	-	-	*	*	*	-	*
DEC	ear	2	3	2	0	byte (ear) $\leftarrow$ (ear) - 1	-	-	-	-	-	*	*	*	-	-
DEC	eam	2+	5+(a)	0	2 $\times$ (b)	byte (eam) $\leftarrow$ (eam) - 1	-	-	-	-	-	*	*	*	-	*
INCW	ear	2	3	2	0	word (ear) $\leftarrow$ (ear) + 1	-	-	-	-	-	*	*	*	-	-
INCW	eam	2+	5+(a)	0	2 $\times$ (c)	word (eam) $\leftarrow$ (eam) + 1	-	-	-	-	-	*	*	*	-	*
DECW	ear	2	3	2	0	word (ear) $\leftarrow$ (ear) - 1	-	-	-	-	-	*	*	*	-	-
DECW	eam	2+	5+(a)	0	2 $\times$ (c)	word (eam) $\leftarrow$ (eam) - 1	-	-	-	-	-	*	*	*	-	*
INCL	ear	2	7	4	0	long (ear) $\leftarrow$ (ear) + 1	-	-	-	-	-	*	*	*	-	-
INCL	eam	2+	9+(a)	0	2 $\times$ (d)	long (eam) $\leftarrow$ (eam) + 1	-	-	-	-	-	*	*	*	-	*
DECL	ear	2	7	4	0	long (ear) $\leftarrow$ (ear) - 1	-	-	-	-	-	*	*	*	-	-
DECL	eam	2+	9+(a)	0	2 $\times$ (d)	long (eam) $\leftarrow$ (eam) - 1	-	-	-	-	-	*	*	*	-	*

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (d) in the table.

**Table C.8-5 11 Compare Instructions (Byte, Word, Long Word)**

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CMP	A	1	1	0	0	byte (AH) - (AL)	-	-	-	-	-	*	*	*	*	-
CMP	A,ear	2	2	1	0	byte (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMP	A,eam	2+	3+(a)	0	(b)	byte (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMP	A,#imm8	2	2	0	0	byte (A) - imm8	-	-	-	-	-	*	*	*	*	-
CMPW	A	1	1	0	0	word (AH) - (AL)	-	-	-	-	-	*	*	*	*	-
CMPW	A,ear	2	2	1	0	word (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMPW	A,eam	2+	3+(a)	0	(c)	word (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMPW	A,#imm16	3	2	0	0	word (A) - imm16	-	-	-	-	-	*	*	*	*	-
CMPL	A,ear	2	6	2	0	long (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMPL	A,eam	2+	7+(a)	0	(d)	long (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMPL	A,#imm32	5	3	0	0	long (A) - imm32	-	-	-	-	-	*	*	*	*	-

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (d) in the table.

Table C.8-6 11 Unsigned Multiplication/Division Instructions (Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIVU A	1	*1	0	0	word (AH) / byte (AL) quotient → byte (AL) remainder → byte (AH)	-	-	-	-	-	-	-	*	*	-
DIVU A,ear	2	*2	1	0	word (A) / byte (ear) quotient → byte (A) remainder → byte (ear)	-	-	-	-	-	-	-	*	*	-
DIVU A,eam	2+	*3	0	*6	word (A) / byte (eam) quotient → byte (A) remainder → byte (eam)	-	-	-	-	-	-	-	*	*	-
DIVUW A,ear	2	*4	1	0	long (A) / word (ear) quotient → word (A) remainder → word (ear)	-	-	-	-	-	-	-	*	*	-
DIVUW A,eam	2+	*5	0	*7	long (A) / word (eam) quotient → word (A) remainder → word (eam)	-	-	-	-	-	-	-	*	*	-
MULU A	1	*8	0	0	byte (AH) * byte (AL) → word (A)	-	-	-	-	-	-	-	-	-	-
MULU A,ear	2	*9	1	0	byte (A) * byte (ear) → word (A)	-	-	-	-	-	-	-	-	-	-
MULU A,eam	2+	*10	0	(b)	byte (A) * byte (eam) → word (A)	-	-	-	-	-	-	-	-	-	-
MULUW A	1	*11	0	0	word (AH) * word (AL) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULUW A,ear	2	*12	1	0	word (A) * word (ear) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULUW A,eam	2+	*13	0	(c)	word (A) * word (eam) → Long (A)	-	-	-	-	-	-	-	-	-	-

\*1: 3: Division by 0 7: Overflow 15: Normal  
 \*2: 4: Division by 0 8: Overflow 16: Normal  
 \*3: 6+(a): Division by 0 9+(a): Overflow 19+(a): Normal  
 \*4: 4: Division by 0 7: Overflow 22: Normal  
 \*5: 6+(a): Division by 0 8+(a): Overflow 26+(a): Normal  
 \*6: (b): Division by 0 or overflow 2 × (b): Normal  
 \*7: (c): Division by 0 or overflow 2 × (c): Normal  
 \*8: 3: Byte (AH) is 0. 7: Byte (AH) is not 0.  
 \*9: 4: Byte (ear) is 0. 8: Byte (ear) is not 0.  
 \*10: 5+(a): Byte (eam) is 0, 9+(a): Byte (eam) is not 0.  
 \*11: 3: Word (AH) is 0. 11: Word (AH) is not 0.  
 \*12: 4: Word (ear) is 0. 12: Word (ear) is not 0.  
 \*13: 5+(a): Word (eam) is 0. 13+(a): Word (eam) is not 0.

## Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (c) in the table.

Table C.8-7 11 Signed Multiplication/Division Instructions (Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIV A	2	*1	0	0	word (AH) / byte (AL) quotient → byte (AL) remainder → byte (AH)	Z	-	-	-	-	-	-	*	*	-
DIV A,ear	2	*2	1	0	word (A) / byte (ear) quotient → byte (A) remainder → byte (ear)	Z	-	-	-	-	-	-	*	*	-
DIV A,eam	2+	*3	0	*6	word (A) / byte (eam) quotient → byte (A) remainder → byte (eam)	Z	-	-	-	-	-	-	*	*	-
DIVW A,ear	2	*4	1	0	long (A) / word (ear) quotient → word (A) remainder → word (ear)	-	-	-	-	-	-	-	*	*	-
DIVW A,eam	2+	*5	0	*7	long (A) / word (eam) quotient → word (A) remainder → word (eam)	-	-	-	-	-	-	-	*	*	-
MUL A	2	*8	0	0	byte (AH) * byte (AL) → word (A)	-	-	-	-	-	-	-	-	-	-
MUL A,ear	2	*9	1	0	byte (A) * byte (ear) → word (A)	-	-	-	-	-	-	-	-	-	-
MUL A,eam	2+	*10	0	(b)	byte (A) * byte (eam) → word (A)	-	-	-	-	-	-	-	-	-	-
MULW A	2	*11	0	0	word (AH) * word (AL) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULW A,ear	2	*12	1	0	word (A) * word (ear) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULW A,eam	2+	*13	0	(c)	word (A) * word (eam) → Long (A)	-	-	-	-	-	-	-	-	-	-

\*1: 3: Division by 0, 8 or 18: Overflow, 18: Normal  
\*2: 4: Division by 0, 11 or 22: Overflow, 23: Normal  
\*3: 5+(a): Division by 0, 12+(a) or 23+(a): Overflow, 24+(a): Normal  
\*4: When dividend is positive; 4: Division by 0, 12 or 30: Overflow, 31: Normal  
When dividend is negative; 4: Division by 0, 12 or 31: Overflow, 32: Normal  
\*5: When dividend is positive; 5+(a): Division by 0, 12+(a) or 31+(a): Overflow, 32+(a): Normal  
When dividend is negative; 5+(a): Division by 0, 12+(a) or 32+(a): Overflow, 33+(a): Normal  
\*6: (b): Division by 0 or overflow, 2 × (b): Normal  
\*7: (c): Division by 0 or overflow, 2 × (c): Normal  
\*8: 3: Byte (AH) is 0, 12: result is positive, 13: result is negative  
\*9: 4: Byte (ear) is 0, 13: result is positive, 14: result is negative  
\*10: 5+(a): Byte (eam) is 0, 14+(a): result is positive, 15+(a): result is negative  
\*11: 3: Word (AH) is 0, 16: result is positive, 19: result is negative  
\*12: 4: Word (ear) is 0, 17: result is positive, 20: result is negative  
\*13: 5+(a): Word (eam) is 0, 18+(a): result is positive, 21+(a): result is negative

Notes:

- The execution cycle count found when an overflow occurs in a DIV or DIVW instruction may be a pre-operation count or a post-operation count depending on the detection timing.
- When an overflow occurs with DIV or DIVW instruction, the contents of the AL are destroyed.
- See Table C.5-1 and Table C.5-2 for information on (a) to (c) in the table.

Table C.8-8 39 Logic 1 Instructions (Byte, Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
AND A,#imm8	2	2	0	0	byte (A) ← (A) and imm8	-	-	-	-	-	*	*	R	-	-
AND A,ear	2	3	1	0	byte (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
AND A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
AND ear,A	2	3	2	0	byte (ear) ← (ear) and (A)	-	-	-	-	-	*	*	R	-	-
AND eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) and (A)	-	-	-	-	-	*	*	R	-	*
OR A,#imm8	2	2	0	0	byte (A) ← (A) or imm8	-	-	-	-	-	*	*	R	-	-
OR A,ear	2	3	1	0	byte (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
OR A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
OR ear,A	2	3	2	0	byte (ear) ← (ear) or (A)	-	-	-	-	-	*	*	R	-	-
OR eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) or (A)	-	-	-	-	-	*	*	R	-	*
XOR A,#imm8	2	2	0	0	byte (A) ← (A) xor imm8	-	-	-	-	-	*	*	R	-	-
XOR A,ear	2	3	1	0	byte (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XOR A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-
XOR ear,A	2	3	2	0	byte (ear) ← (ear) xor (A)	-	-	-	-	-	*	*	R	-	-
XOR eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) xor (A)	-	-	-	-	-	*	*	R	-	*
NOT A	1	2	0	0	byte (A) ← not (A)	-	-	-	-	-	*	*	R	-	-
NOT ear	2	3	2	0	byte (ear) ← not (ear)	-	-	-	-	-	*	*	R	-	-
NOT eam	2+	5+(a)	0	2 × (b)	byte (eam) ← not (eam)	-	-	-	-	-	*	*	R	-	*
ANDW A	1	2	0	0	word (A) ← (AH) and (A)	-	-	-	-	-	*	*	R	-	-
ANDW A,#imm16	3	2	0	0	word (A) ← (A) and imm16	-	-	-	-	-	*	*	R	-	-
ANDW A,ear	2	3	1	0	word (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
ANDW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
ANDW ear,A	2	3	2	0	word (ear) ← (ear) and (A)	-	-	-	-	-	*	*	R	-	-
ANDW eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) and (A)	-	-	-	-	-	*	*	R	-	*
ORW A	1	2	0	0	word (A) ← (AH) or (A)	-	-	-	-	-	*	*	R	-	-
ORW A,#imm16	3	2	0	0	word (A) ← (A) or imm16	-	-	-	-	-	*	*	R	-	-
ORW A,ear	2	3	1	0	word (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
ORW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
ORW ear,A	2	3	2	0	word (ear) ← (ear) or (A)	-	-	-	-	-	*	*	R	-	-
ORW eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) or (A)	-	-	-	-	-	*	*	R	-	*
XORW A	1	2	0	0	word (A) ← (AH) xor (A)	-	-	-	-	-	*	*	R	-	-
XORW A,#imm16	3	2	0	0	word (A) ← (A) xor imm16	-	-	-	-	-	*	*	R	-	-
XORW A,ear	2	3	1	0	word (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XORW A,eam	2+	4+(a)	0	(c)	word (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-
XORW ear,A	2	3	2	0	word (ear) ← (ear) xor (A)	-	-	-	-	-	*	*	R	-	-
XORW eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) xor (A)	-	-	-	-	-	*	*	R	-	*
NOTW A	1	2	0	0	word (A) ← not (A)	-	-	-	-	-	*	*	R	-	-
NOTW ear	2	3	2	0	word (ear) ← not (ear)	-	-	-	-	-	*	*	R	-	-
NOTW eam	2+	5+(a)	0	2 × (c)	word (eam) ← not (eam)	-	-	-	-	-	*	*	R	-	*

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (c) in the table.

**Table C.8-9 6 Logic 2 Instructions (Long Word)**

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ANDL A,ear	2	6	2	0	long (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
ANDL A,eam	2+	7+(a)	0	(d)	long (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
ORL A,ear	2	6	2	0	long (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
ORL A,eam	2+	7+(a)	0	(d)	long (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
XORL A,ear	2	6	2	0	long (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XORL A,eam	2+	7+(a)	0	(d)	long (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-

Note:

See Table C.5-1 and Table C.5-2 for information on (a) and (d) in the table.

**Table C.8-10 6 Sign Inversion Instructions (Byte, Word)**

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NEG A	1	2	0	0	byte (A) ← 0 - (A)	X	-	-	-	-	*	*	*	*	-
NEG ear	2	3	2	0	byte (ear) ← 0 - (ear)	-	-	-	-	-	*	*	*	*	-
NEG eam	2+	5+(a)	0	2 × (b)	byte (eam) ← 0 - (eam)	-	-	-	-	-	*	*	*	*	*
NEGW A	1	2	0	0	word (A) ← 0 - (A)	-	-	-	-	-	*	*	*	*	-
NEGW ear	2	3	2	0	word (ear) ← 0 - (ear)	-	-	-	-	-	*	*	*	*	-
NEGW eam	2+	5+(a)	0	2 × (c)	word (eam) ← 0 - (eam)	-	-	-	-	-	*	*	*	*	*

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (c) in the table.

**Table C.8-11 1 Normalization Instruction (Long Word)**

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NRML A,R0	2	*1	1	0	long (A) ← Shift left to the position where '1' is set for the first time. byte (R0) ← Shift count at that time	-	-	-	-	-	-	*	-	-	-

\*1: 4 when all accumulators have a value of 0; otherwise, 6+(R0)

Table C.8-12 18 Shift Instructions (Byte, Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
RORC A	2	2	0	0	byte (A) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
ROLC A	2	2	0	0	byte (A) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
RORC ear	2	3	2	0	byte (ear) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
RORC eam	2+	5+(a)	0	2 × (b)	byte (eam) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	*
ROLC ear	2	3	2	0	byte (ear) ← Left rotation with carry	-	-	-	-	-	*	*	-	*	-
ROLC eam	2+	5+(a)	0	2 × (b)	byte (eam) ← Left rotation with carry	-	-	-	-	-	*	*	-	*	*
ASR A,R0	2	*1	1	0	byte (A) ← Arithmetic right shift (A, 1 bit)	-	-	-	-	*	*	*	-	*	-
LSR A,R0	2	*1	1	0	byte (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSL A,R0	2	*1	1	0	byte (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-
ASRW A	1	2	0	0	word (A) ← Arithmetic right shift (A, 1 bit)	-	-	-	-	*	*	*	-	*	-
LSRW A/SHRW A	1	2	0	0	word (A) ← Logical right shift (A, 1 bit)	-	-	-	-	*	R	*	-	*	-
LSLW A/SHLW A	1	2	0	0	word (A) ← Logical left shift (A, 1 bit)	-	-	-	-	-	*	*	-	*	-
ASRW A,R0	2	*1	1	0	word (A) ← Arithmetic right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSRW A,R0	2	*1	1	0	word (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSLW A,R0	2	*1	1	0	word (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-
ASRL A,R0	2	*2	1	0	long (A) ← Arithmetic right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSRL A,R0	2	*2	1	0	long (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSLL A,R0	2	*2	1	0	long (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-

\*1: 6 when R0 is 0; otherwise, 5 + (R0)

\*2: 6 when R0 is 0; otherwise, 6 + (R0)

## Note:

See Table C.5-1 and Table C.5-2 for information on (a) and (b) in the table.

Table C.8-13 31 Branch 1 Instructions

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
BZ/BEQ rel	2	*1	0	0	Branch on (Z) = 1	-	-	-	-	-	-	-	-	-	-
BNZ/ rel BNE	2	*1	0	0	Branch on (Z) = 0	-	-	-	-	-	-	-	-	-	-
BC/BLO rel	2	*1	0	0	Branch on (C) = 1	-	-	-	-	-	-	-	-	-	-
BNC/ rel BHS	2	*1	0	0	Branch on (C) = 0	-	-	-	-	-	-	-	-	-	-
BN rel	2	*1	0	0	Branch on (N) = 1	-	-	-	-	-	-	-	-	-	-
BP rel	2	*1	0	0	Branch on (N) = 0	-	-	-	-	-	-	-	-	-	-
BV rel	2	*1	0	0	Branch on (V) = 1	-	-	-	-	-	-	-	-	-	-
BNV rel	2	*1	0	0	Branch on (V) = 0	-	-	-	-	-	-	-	-	-	-
BT rel	2	*1	0	0	Branch on (T) = 1	-	-	-	-	-	-	-	-	-	-
BNT rel	2	*1	0	0	Branch on (T) = 0	-	-	-	-	-	-	-	-	-	-
BLT rel	2	*1	0	0	Branch on (V) xor (N) = 1	-	-	-	-	-	-	-	-	-	-
BGE rel	2	*1	0	0	Branch on (V) xor (N) = 0	-	-	-	-	-	-	-	-	-	-
BLE rel	2	*1	0	0	Branch on ((V) xor (N)) or (Z) = 1	-	-	-	-	-	-	-	-	-	-
BGT rel	2	*1	0	0	Branch on ((V) xor (N)) or (Z) = 0	-	-	-	-	-	-	-	-	-	-
BLS rel	2	*1	0	0	Branch on (C) or (Z) = 1	-	-	-	-	-	-	-	-	-	-
BHI rel	2	*1	0	0	Branch on (C) or (Z) = 0	-	-	-	-	-	-	-	-	-	-
BRA rel	2	*1	0	0	Unconditional branch	-	-	-	-	-	-	-	-	-	-
JMP @A	1	2	0	0	word (PC) ← (A)	-	-	-	-	-	-	-	-	-	-
JMP addr16	3	3	0	0	word (PC) ← addr16	-	-	-	-	-	-	-	-	-	-
JMP @ear	2	3	1	0	word (PC) ← (ear)	-	-	-	-	-	-	-	-	-	-
JMP @eam	2+	4+(a)	0	(c)	word (PC) ← (eam)	-	-	-	-	-	-	-	-	-	-
JMPP @ear *3	2	5	2	0	word (PC) ← (ear), (PCB) ← (ear+2)	-	-	-	-	-	-	-	-	-	-
JMPP @eam *3	2+	6+(a)	0	(d)	word (PC) ← (eam), (PCB) ← (eam+2)	-	-	-	-	-	-	-	-	-	-
JMPP addr24	4	4	0	0	word (PC) ← ad24 0-15, (PCB) ← ad24 16-23	-	-	-	-	-	-	-	-	-	-
CALL @ear *4	2	6	1	(c)	word (PC) ← (ear)	-	-	-	-	-	-	-	-	-	-
CALL @eam *4	2+	7+(a)	0	2 × (c)	word (PC) ← (eam)	-	-	-	-	-	-	-	-	-	-
CALL addr16 *5	3	6	0	(c)	word (PC) ← addr16	-	-	-	-	-	-	-	-	-	-
CALLV #vct4 *5	1	7	0	2 × (c)	Vector call instruction	-	-	-	-	-	-	-	-	-	-
CALLP @ear *6	2	10	2	2 × (c)	word (PC) ← (ear), (PCB) ← (ear+2)	-	-	-	-	-	-	-	-	-	-
CALLP @eam *6	2+	11+(a)	0	*2	word (PC) ← (eam), (PCB) ← (eam+2)	-	-	-	-	-	-	-	-	-	-
CALLP addr24 *7	4	10	0	2 × (c)	word (PC) ← ad24 0-15, (PCB) ← ad24 16-23	-	-	-	-	-	-	-	-	-	-

\*1: 4 when a branch is made; otherwise, 3

\*2:  $3 \times (c) + (b)$

\*3: Read (word) of branch destination address

\*4: W: Save to stack (word) R: Read (word) of branch destination address

\*5: Save to stack (word)

\*6: W: Save to stack (long word), R: Read (long word) of branch destination address

\*7: Save to stack (long word)

Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (d) in the table.

Table C.8-14 19 Branch 2 Instructions

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CBNE A,#imm8,rel	3	*1	0	0	Branch on byte (A) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CWBNE A,#imm16,rel	4	*1	0	0	Branch on word (A) not equal to imm16	-	-	-	-	-	*	*	*	*	-
CBNE ear,#imm8,rel	4	*2	1	0	Branch on byte (ear) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CBNE eam,#imm8,rel *9	4+	*3	0	(b)	Branch on byte (eam) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CWBNE ear,#imm16,rel	5	*4	1	0	Branch on word (ear) not equal to imm16	-	-	-	-	-	*	*	*	*	-
CWBNE eam,#imm16,rel*9	5+	*3	0	(c)	Branch on word (eam) not equal to imm16	-	-	-	-	-	*	*	*	*	-
DBNZ ear,rel	3	*5	2	0	byte (ear) ← (ear) - 1, Branch on (ear) not equal to 0	-	-	-	-	-	*	*	*	-	-
DBNZ eam,rel	3+	*6	2	2 × (b)	byte (eam) ← (eam) - 1, Branch on (eam) not equal to 0	-	-	-	-	-	*	*	*	-	*
DWBNZ ear,rel	3	*5	2	0	word (ear) ← (ear) - 1, Branch on (ear) not equal to 0	-	-	-	-	-	*	*	*	-	-
DWBNZ eam,rel	3+	*6	2	2 × (c)	word (eam) ← (eam) - 1, Branch on (eam) not equal to 0	-	-	-	-	-	*	*	*	-	*
INT #vct8	2	20	0	8 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INT addr16	3	16	0	6 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INTP addr24	4	17	0	6 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INT9	1	20	0	8 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
RETI	1	*8	0	*7	Return from interrupt	-	-	*	*	*	*	*	*	*	-
LINK #imm8	2	6	0	(c)	Saves the old frame pointer in the stack upon entering the function, then sets the new frame pointer and reserves the local pointer area.	-	-	-	-	-	-	-	-	-	-
UNLINK	1	5	0	(c)	Recovers the old frame pointer from the stack upon exiting the function.	-	-	-	-	-	-	-	-	-	-
RET *10	1	4	0	(c)	Return from subroutine	-	-	-	-	-	-	-	-	-	-
RETP *11	1	6	0	(d)	Return from subroutine	-	-	-	-	-	-	-	-	-	-

\*1: 5 when a branch is made; otherwise, 4

\*2: 13 when a branch is made; otherwise, 12

\*3: 7+(a) when a branch is made; otherwise, 6+(a)

\*4: 8 when a branch is made; otherwise, 7

\*5: 7 when a branch is made; otherwise, 6

\*6: 8+(a) when a branch is made; otherwise, 7+(a)

\*7: 3 × (b) + 2 × (c) when jumping to the next interruption request; 6 × (c) when returning from the current interruption

\*8: 15 when jumping to the next interruption request; 17 when returning from the current interruption

\*9: Do not use RWj+ addressing mode with a CBNE or CWBNE instruction.

\*10: Return from stack (word)

\*11: Return from stack (long word)

## Note:

See Table C.5-1 and Table C.5-2 for information on (a) to (d) in the table.



Table C.8-15 28 Other Control Instructions (Byte, Word, Long Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
PUSHW A	1	4	0	(c)	word (SP) $\leftarrow$ (SP) - 2, ((SP)) $\leftarrow$ (A)	-	-	-	-	-	-	-	-	-	-
PUSHW AH	1	4	0	(c)	word (SP) $\leftarrow$ (SP) - 2, ((SP)) $\leftarrow$ (AH)	-	-	-	-	-	-	-	-	-	-
PUSHW PS	1	4	0	(c)	word (SP) $\leftarrow$ (SP) - 2, ((SP)) $\leftarrow$ (PS)	-	-	-	-	-	-	-	-	-	-
PUSHW rlst	2	*3	*5	*4	(SP) $\leftarrow$ (SP) - 2n, ((SP)) $\leftarrow$ (rlst)	-	-	-	-	-	-	-	-	-	-
POPW A	1	3	0	(c)	word (A) $\leftarrow$ ((SP)), (SP) $\leftarrow$ (SP) + 2	-	*	-	-	-	-	-	-	-	-
POPW AH	1	3	0	(c)	word (AH) $\leftarrow$ ((SP)), (SP) $\leftarrow$ (SP) + 2	-	-	-	-	-	-	-	-	-	-
POPW PS	1	4	0	(c)	word (PS) $\leftarrow$ ((SP)), (SP) $\leftarrow$ (SP) + 2	-	-	*	*	*	*	*	*	*	-
POPW rlst	2	*2	*5	*4	(rlst) $\leftarrow$ ((SP)), (SP) $\leftarrow$ (SP) + 2n	-	-	-	-	-	-	-	-	-	-
JCTX @A	1	14	0	6 $\times$ (c)	Context switch instruction	-	-	*	*	*	*	*	*	*	-
AND CCR,#imm8	2	3	0	0	byte (CCR) $\leftarrow$ (CCR) and imm8	-	-	*	*	*	*	*	*	*	-
OR CCR,#imm8	2	3	0	0	byte (CCR) $\leftarrow$ (CCR) or imm8	-	-	*	*	*	*	*	*	*	-
MOV RP,#imm8	2	2	0	0	byte (RP) $\leftarrow$ imm8	-	-	-	-	-	-	-	-	-	-
MOV ILM,#imm8	2	2	0	0	byte (ILM) $\leftarrow$ imm8	-	-	-	-	-	-	-	-	-	-
MOVEA RWi,ear	2	3	1	0	word (RWi) $\leftarrow$ ear	-	-	-	-	-	-	-	-	-	-
MOVEA RWi,eam	2+	2+(a)	1	0	word (RWi) $\leftarrow$ eam	-	-	-	-	-	-	-	-	-	-
MOVEA A,ear	2	1	0	0	word (A) $\leftarrow$ ear	-	*	-	-	-	-	-	-	-	-
MOVEA A,eam	2+	1+(a)	0	0	word (A) $\leftarrow$ eam	-	*	-	-	-	-	-	-	-	-
ADDSP #imm8	2	3	0	0	word (SP) $\leftarrow$ (SP) + ext(imm8)	-	-	-	-	-	-	-	-	-	-
ADDSP #imm16	3	3	0	0	word (SP) $\leftarrow$ (SP) + imm16	-	-	-	-	-	-	-	-	-	-
MOV A,brg1	2	*1	0	0	byte (A) $\leftarrow$ (brg1)	Z	*	-	-	-	*	*	-	-	-
MOV brg2,A	2	1	0	0	byte (brg2) $\leftarrow$ (A)	-	-	-	-	-	*	*	-	-	-
NOP	1	1	0	0	No operation	-	-	-	-	-	-	-	-	-	-
ADB	1	1	0	0	Prefix code for AD space access	-	-	-	-	-	-	-	-	-	-
DTB	1	1	0	0	Prefix code for DT space access	-	-	-	-	-	-	-	-	-	-
PCB	1	1	0	0	Prefix code for PC space access	-	-	-	-	-	-	-	-	-	-
SPB	1	1	0	0	Prefix code for SP space access	-	-	-	-	-	-	-	-	-	-
NCC	1	1	0	0	Prefix code for flag no-change	-	-	-	-	-	-	-	-	-	-
CMR	1	1	0	0	Prefix code for common register bank	-	-	-	-	-	-	-	-	-	-

\*1: PCB, ADB, SSB, USB, SPB: 1, DTB, DPR: 2

\*2:  $7 + 3 \times (\text{POP count}) + 2 \times (\text{POP last register number})$ , 7 when RLST = 0 (no transfer register)

\*3:  $29 + 3 \times (\text{PUSH count}) - 3 \times (\text{PUSH last register number})$ , 8 when RLST = 0 (no transfer register)

\*4: (POP count)  $\times$  (c) or (PUSH count)  $\times$  (c)

\*5: (POP count) or (PUSH count)

Note:

See Table C.5-1 and Table C.5-2 for information on (a) and (c) in the table.

Table C.8-16 21 Bit Operand Instructions

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVB A,dir:bp	3	5	0	(b)	byte (A) $\leftarrow$ (dir:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB A,addr16:bp	4	5	0	(b)	byte (A) $\leftarrow$ (addr16:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB A,io:bp	3	4	0	(b)	byte (A) $\leftarrow$ (io:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB dir:bp,A	3	7	0	$2 \times (b)$	bit (dir:bp)b $\leftarrow$ (A)	-	-	-	-	-	*	*	-	-	*
MOVB addr16:bp,A	4	7	0	$2 \times (b)$	bit (addr16:bp)b $\leftarrow$ (A)	-	-	-	-	-	*	*	-	-	*
MOVB io:bp,A	3	6	0	$2 \times (b)$	bit (io:bp)b $\leftarrow$ (A)	-	-	-	-	-	*	*	-	-	*
SETB dir:bp	3	7	0	$2 \times (b)$	bit (dir:bp)b $\leftarrow$ 1	-	-	-	-	-	-	-	-	-	*
SETB addr16:bp	4	7	0	$2 \times (b)$	bit (addr16:bp)b $\leftarrow$ 1	-	-	-	-	-	-	-	-	-	*
SETB io:bp	3	7	0	$2 \times (b)$	bit (io:bp)b $\leftarrow$ 1	-	-	-	-	-	-	-	-	-	*
CLRB dir:bp	3	7	0	$2 \times (b)$	bit (dir:bp)b $\leftarrow$ 0	-	-	-	-	-	-	-	-	-	*
CLRB addr16:bp	4	7	0	$2 \times (b)$	bit (addr16:bp)b $\leftarrow$ 0	-	-	-	-	-	-	-	-	-	*
CLRB io:bp	3	7	0	$2 \times (b)$	bit (io:bp)b $\leftarrow$ 0	-	-	-	-	-	-	-	-	-	*
BBC dir:bp,rel	4	*1	0	(b)	Branch on (dir:bp) b = 0	-	-	-	-	-	-	*	-	-	-
BBC addr16:bp,rel	5	*1	0	(b)	Branch on (addr16:bp) b = 0	-	-	-	-	-	-	*	-	-	-
BBC io:bp,rel	4	*2	0	(b)	Branch on (io:bp) b = 0	-	-	-	-	-	-	*	-	-	-
BBS dir:bp,rel	4	*1	0	(b)	Branch on (dir:bp) b = 1	-	-	-	-	-	-	*	-	-	-
BBS addr16:bp,rel	5	*1	0	(b)	Branch on (addr16:bp) b = 1	-	-	-	-	-	-	*	-	-	-
BBS io:bp,rel	4	*2	0	(b)	Branch on (io:bp) b = 1	-	-	-	-	-	-	*	-	-	-
SBBS addr16:bp,rel	5	*3	0	$2 \times (b)$	Branch on (addr16:bp) b = 1, bit (addr16:bp) b $\leftarrow$ 1	-	-	-	-	-	-	*	-	-	*
WBTS io:bp	3	*4	0	*5	Waits until (io:bp) b = 1	-	-	-	-	-	-	-	-	-	-
WBTC io:bp	3	*4	0	*5	Waits until (io:bp) b = 0	-	-	-	-	-	-	-	-	-	-

\*1: 8 when a branch is made; otherwise, 7

\*2: 7 when a branch is made; otherwise, 6

\*3: 10 when the condition is met; otherwise, 9

\*4: Undefined count

\*5: Until the condition is met

Note:

See Table C.5-1 and Table C.5-2 for information on (b) in the table.

Table C.8-17 6 Accumulator Operation Instructions (Byte, Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
SWAP	1	3	0	0	byte (A)0-7 $\leftrightarrow$ (A)8-15	-	-	-	-	-	-	-	-	-	-
SWAPW	1	2	0	0	word (AH) $\leftrightarrow$ (AL)	-	*	-	-	-	-	-	-	-	-
EXT	1	1	0	0	Byte sign extension	X	-	-	-	-	*	*	-	-	-
EXTW	1	2	0	0	Word sign extension	-	X	-	-	-	*	*	-	-	-
ZEXT	1	1	0	0	Byte zero extension	Z	-	-	-	-	R	*	-	-	-
ZEXTW	1	1	0	0	Word zero extension	-	Z	-	-	-	R	*	-	-	-

**Table C.8-18 10 String Instructions**

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVS / MOVSI	2	*2	*5	*3	byte transfer @AH+ ← @AL+, counter = RW0	-	-	-	-	-	-	-	-	-	-
MOVSD	2	*2	*5	*3	byte transfer @AH- ← @AL-, counter = RW0	-	-	-	-	-	-	-	-	-	-
SCEQ / SCEQI	2	*1	*8	*4	byte search @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
SCEQD	2	*1	*8	*4	byte search @AH- ← AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
FILS / FILSI	2	6m+6	*8	*3	byte fill @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	-	-	-
MOVSW / MOVSWI	2	*2	*5	*6	word transfer @AH+ ← @AL+, counter = RW0	-	-	-	-	-	-	-	-	-	-
MOVSWD	2	*2	*5	*6	word transfer @AH- ← @AL-, counter = RW0	-	-	-	-	-	-	-	-	-	-
SCWEQ / SCWEQI	2	*1	*8	*7	word search @AH+ - AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
SCWEQD	2	*1	*8	*7	word search @AH- - AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
FILSW / FILSWI	2	6m+6	*8	*6	word fill @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	-	-	-

\*1: 5 when RW0 is 0,  $4 + 7 \times (RW0)$  when the counter expires, or  $7n + 5$  when a match occurs

\*2: 5 when RW0 is 0; otherwise,  $4 + 8 \times (RW0)$

\*3:  $(b) \times (RW0) + (b) \times (RW0)$  When the source and destination access different areas, calculate the (b) item individually.

\*4:  $(b) \times n$

\*5:  $2 \times (b) \times (RW0)$

\*6:  $(c) \times (RW0) + (c) \times (RW0)$  When the source and destination access different areas, calculate the (c) item individually.

\*7:  $(c) \times n$

\*8:  $(b) \times (RW0)$

**Note:**

m: RW0 value (counter value), n: Loop count

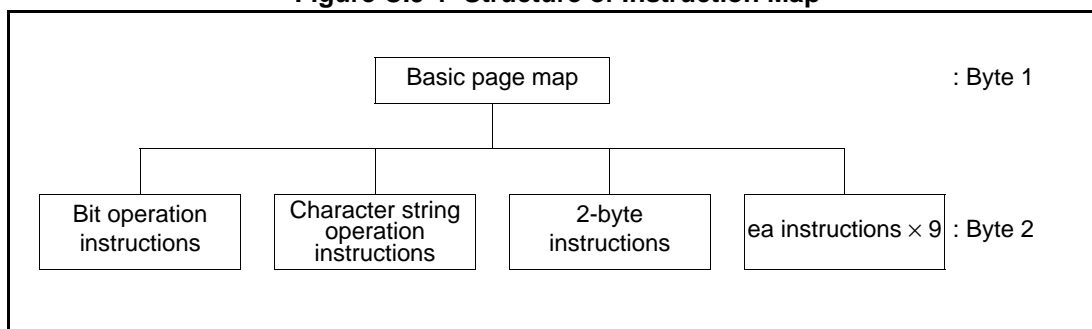
See Table C.5-1 and Table C.5-2 for information on (b) and (c) in the table.

## C.9 Instruction Map

Each F<sup>2</sup>MC-16LX instruction code consists of 1 or 2 bytes. Therefore, the instruction map consists of multiple pages. Table C.9-2 to Table C.9-21 summarize the F<sup>2</sup>MC-16LX instruction map.

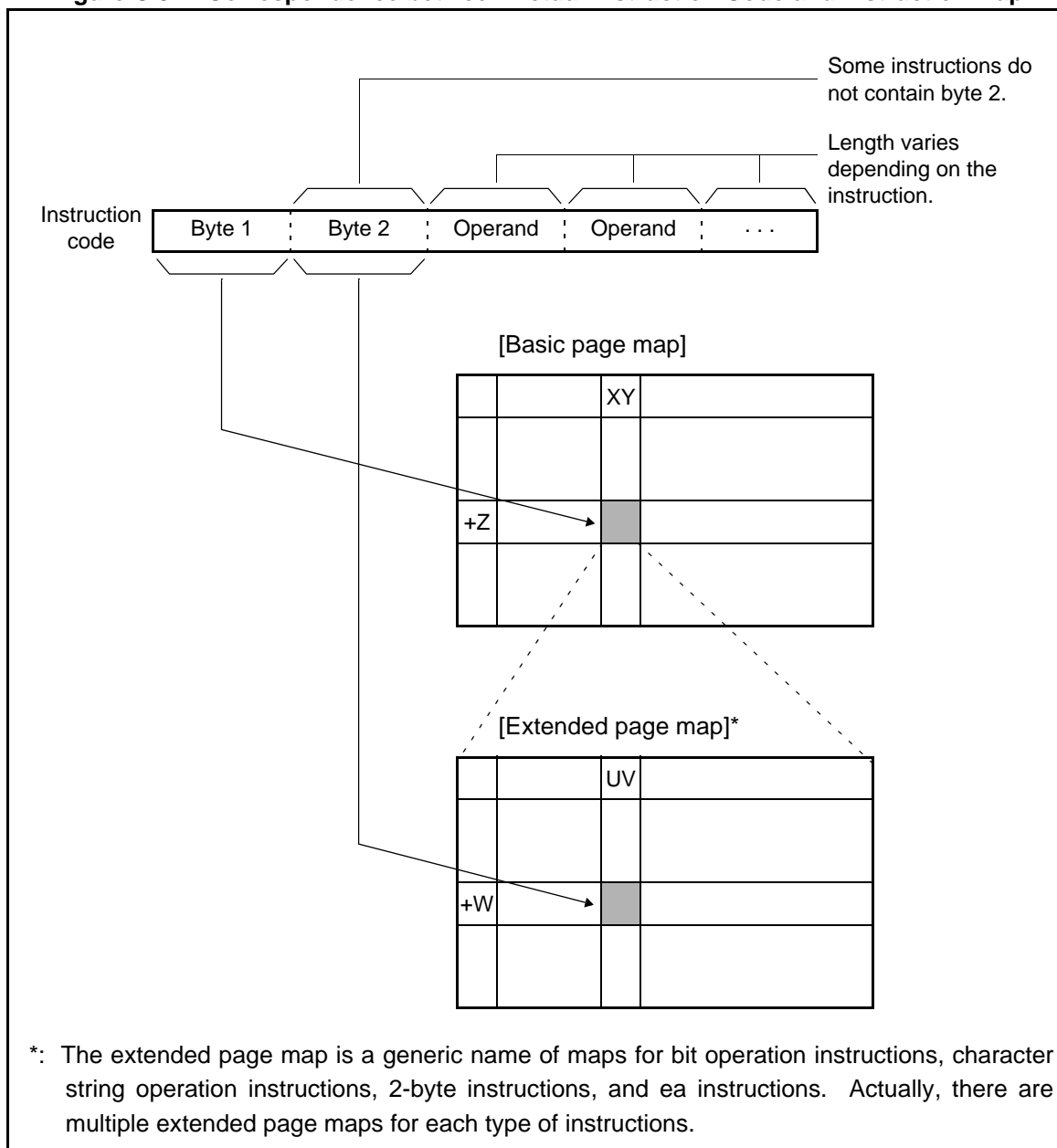
### ■ Structure of Instruction Map

Figure C.9-1 Structure of Instruction Map



An instruction such as the NOP instruction that ends in one byte is completed within the basic page. An instruction such as the MOVS instruction that requires two bytes recognizes the existence of byte 2 when it references byte 1, and can check the following one byte by referencing the map for byte 2. Figure C.9-2 shows the correspondence between an actual instruction code and instruction map.

**Figure C.9-2 Correspondence between Actual Instruction Code and Instruction Map**



An example of an instruction code is shown in Table C.9-1 .

**Table C.9-1 Example of an Instruction Code**

Instruction	Byte 1 (from basic page map)	Byte 2 (from extended page map)
NOP	00 +0=00	-
AND A, #8	30 +4=34	-
MOV A, ADB	60 +F=6F	00 +0=00
@RW2+d8, #8, rel	70 +0=70	F0 +2=F2

Table C.9-2 Basic Page Map

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	NOP	CMR	ADD A, dir	ADD A, #8	MOV A, dir	MOV A, io	BRA rel	ea instruction 1	MOV A, Ri	MOV Ri, A	MOV Ri, #8	MOV A, Ri	MOVX A, @RiH+d8	MOV A, #4	CALL #4	BZ/BEQ rel
+1	INT9	NCC	SUB A, dir	SUB A, #8	MOV dir, A	MOV io, A	JMP @A	ea instruction 2								BNZ/BNE rel
+2	ADDDC A	SUBDC A	ADDC A	SUBC A	MOV A, #8	MOV A, addr16	JMP addr16	ea instruction 3								BC/BLO rel
+3	NEG A	JCTX @A	CMP A	CMP A, #8	MOVX A, #8	MOV addr16, A	JMPP addr24	ea instruction 4								BNC/BHS rel
+4	PCB	EXT	AND CCR, #8	AND A, #8	MOV dir, #8	MOV io, #8	CALL addr16	ea instruction 5								BN rel
+5	DTB	ZEXT	OR CCR, #8	OR A, #8	MOVX A, dir	MOVX A, io	CALLP addr24	ea instruction 6								BP rel
+6	ADB	SWAP	DIVU A	XOR A, #8	MOVW A, SP	MOVW io, #16	RETP	ea instruction 7								BV rel
+7	SPB	ADDSP #8	MULU A	NOT A	MOVW SP, A	MOVX A, addr16	RET	ea instruction 8								BNV rel
+8	LINK #imm8	ADDL A, #32	ADDW A	ADDW A, #16	MOVW A, dir	MOVW A, io	INT #vct8	ea instruction 9	MOVW A, RWi	MOVW RWi, A	MOVW RWi, #16	MOVX A, @RWiH+d8	MOVW @RWiH+d8, A			BT rel
+9	UNLINK	SUBL A, #32	SUBW A	SUBW A, #16	MOVW dir, A	MOVW io, A	INT	MOVEA RWi, ea								BNT rel
+A	MOV RP, #8	MOV ILM, #8	CBNE A, #8, rel	CWBN A, #16, rel	MOVW A, #16	MOVW A, addr16	INTP	MOV Ri, ea								BLT rel
+B	NEGW A	CMPL A, #32	CMPW A	CMPW A, #16	MOVL A, #32	MOVW addr16, A	RETI	MOVW RWi, ea								BGE rel
+C	LSLW A	EXTW A	ANDW A	ANDW A, #16	PUSHW A	POPW A	Bit operation instruction	MOV ea, Ri								BLE rel
+D		ZEXTW	ORW A	ORW A, #16	PUSHW AH	POPW AH		MOVW ea, RWi								BGT rel
+E	ASRW A	SWAPW A	XORW A	XORW A, #16	PUSHW PS	POPW PS	Character string operation instruction	XCH Ri, ea								BLS rel
+F	LSRW A	ADDSP #16	MULW A	NOTW A	PUSHW r1st	POPW r1st	2-byte instruction	XCHW RWi, ea								BHI rel

**Table C.9-3 Bit Operation Instruction Map (First Byte = 6C<sub>H</sub>)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVB A, io:bp		MOVB io:bp, A		CLRB io:bp		SETB io:bp		BBC io:bp, rel		BBS io:bp, rel		WBTS io:bp		WBTC io:bp	
+1																
+2																
+3																
+4																
+5																
+6																
+7																
+8	MOVB A, dir:bp	MOVB A, addr16:bp	MOVB dir:bp, A	MOVB addr16:bp, A	CLRB dir:bp	CLRB addr16:bp	SETB dir:bp	SETB addr16:bp	BBC dir:bp, rel	BBC addr16:bp, rel	BBS dir:bp, rel	BBS addr16:bp, rel				SBBS addr16:bp
+9																
+A																
+B																
+C																
+D																
+E																
+F																

Table C.9-4 Character String Operation Instruction Map (First Byte = 6E<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVSI PCB, PCB	MOVSD PCB, PCB	MOVSWI PCB, PCB	MOVSWD PCB, PCB					SCWEQI PCB	SCEQD PCB	SCWEQI PCB	SCWEQD PCB	FILSI PCB		FILSI PCB	
+1	PCB, DTB								DTB	DTB	DTB	DTB	DTB		DTB	
+2	PCB, ADB								ADB	ADB	ADB	ADB	ADB		ADB	
+3	PCB, SPB								SPB	SPB	SPB	SPB	SPB		SPB	
+4	DTB, PCB															
+5	DTB, DTB															
+6	DTB, ADB															
+7	DTB, SPB															
+8	ADB, PCB															
+9	ADB, DTB															
+A	ADB, ADB															
+B	ADB, SPB															
+C	SPB, PCB															
+D	SPB, DTB															
+E	SPB, ADB															
+F	SPB, SPB															



**Table C.9-5 2-byte Instruction Map (First Byte = 6F<sub>H</sub>)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOV A, DTB	MOV DTB, A	MOVX A, @RL0+d8	MOV @RL0+d8, A	MOV A, @RL0+d8											
+1	MOV A, ADB	MOV ADB, A														
+2	MOV A, SSB	MOV SSB, A	MOVX A, @RL1+d8	MOV @RL1+d8, A	MOV A, @RL1+d8											
+3	MOV A, USB	MOV USB, A														
+4	MOV A, DPR	MOV DPR, A	MOVX A, @RL2+d8	MOV @RL2+d8, A	MOV A, @RL2+d8											
+5	MOV A, @A	MOV @AL, AH														
+6	MOV A, PCB	MOV A, @A	MOVX A, @RL3+d8	MOV @RL3+d8, A	MOV A, @RL3+d8											
+7	ROL A	ROL A														
+8				MOVW @RL0+d8, A	MOVW A, @RL0+d8		MUL A									
+9							MULW A									
+A				MOVW @RL1+d8, A	MOVW A, @RL1+d8		DIVU A									
+B																
+C	LSLW A, R0	LSLL A, R0	LSL A, R0	MOVW @RL2+d8, A	MOVW A, @RL2+d8											
+D	MOVW A, @A	MOVW @AL, AH	NRML A, R0													
+E	ASRW A, R0	ASRL A, R0	ASR A, R0	MOVW @RL3+d8, A	MOVW A, @RL3+d8											
+F	LSRW A, R0	LSRL A, R0	LSR A, R0													

Table C.9-6 ea Instruction 1 (First Byte = 70<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
					CWNE ↓, CBNE ↓										CBNE ↓, CBNE ↓	
+0	ADDL A, A, RLO', @RW0+d8	ADDL A, A, RLO', @RW0+d8	SUBL A, A, RLO', @RW0+d8	SUBL A, A, RLO', @RW0+d8	RW0', @RW0+d8 #16, rel'	CMPL A, A, RLO', @RW0+d8	CMPL A, A, RLO', @RW0+d8	CMPL A, A, RLO', @RW0+d8	ANDL A, A, RLO', @RW0+d8	ANDL A, A, RLO', @RW0+d8	ORL A, A, RLO', @RW0+d8	ORL A, A, RLO', @RW0+d8	XORL A, A, RLO', @RW0+d8	XORL A, A, RLO', @RW0+d8	R0', @RW0+d8 #8, rel'	R0', @RW0+d8 #8, rel'
+1	ADDL A, A, RLO', @RW1+d8	ADDL A, A, RLO', @RW1+d8	SUBL A, A, RLO', @RW1+d8	SUBL A, A, RLO', @RW1+d8	RW1', @RW1+d8 #16, rel'	CMPL A, A, RLO', @RW1+d8	CMPL A, A, RLO', @RW1+d8	CMPL A, A, RLO', @RW1+d8	ANDL A, A, RLO', @RW1+d8	ANDL A, A, RLO', @RW1+d8	ORL A, A, RLO', @RW1+d8	ORL A, A, RLO', @RW1+d8	XORL A, A, RLO', @RW1+d8	XORL A, A, RLO', @RW1+d8	R1', @RW1+d8 #8, rel'	R1', @RW1+d8 #8, rel'
+2	ADDL A, A, RL1', @RW2+d8	ADDL A, A, RL1', @RW2+d8	SUBL A, A, RL1', @RW2+d8	SUBL A, A, RL1', @RW2+d8	RW2', @RW2+d8 #16, rel'	CMPL A, A, RL1', @RW2+d8	CMPL A, A, RL1', @RW2+d8	CMPL A, A, RL1', @RW2+d8	ANDL A, A, RL1', @RW2+d8	ANDL A, A, RL1', @RW2+d8	ORL A, A, RL1', @RW2+d8	ORL A, A, RL1', @RW2+d8	XORL A, A, RL1', @RW2+d8	XORL A, A, RL1', @RW2+d8	R2', @RW2+d8 #8, rel'	R2', @RW2+d8 #8, rel'
+3	ADDL A, A, RL1', @RW3+d8	ADDL A, A, RL1', @RW3+d8	SUBL A, A, RL1', @RW3+d8	SUBL A, A, RL1', @RW3+d8	RW3', @RW3+d8 #16, rel'	CMPL A, A, RL1', @RW3+d8	CMPL A, A, RL1', @RW3+d8	CMPL A, A, RL1', @RW3+d8	ANDL A, A, RL1', @RW3+d8	ANDL A, A, RL1', @RW3+d8	ORL A, A, RL1', @RW3+d8	ORL A, A, RL1', @RW3+d8	XORL A, A, RL1', @RW3+d8	XORL A, A, RL1', @RW3+d8	R3', @RW3+d8 #8, rel'	R3', @RW3+d8 #8, rel'
+4	ADDL A, A, RL2', @RW4+d8	ADDL A, A, RL2', @RW4+d8	SUBL A, A, RL2', @RW4+d8	SUBL A, A, RL2', @RW4+d8	RW4', @RW4+d8 #16, rel'	CMPL A, A, RL2', @RW4+d8	CMPL A, A, RL2', @RW4+d8	CMPL A, A, RL2', @RW4+d8	ANDL A, A, RL2', @RW4+d8	ANDL A, A, RL2', @RW4+d8	ORL A, A, RL2', @RW4+d8	ORL A, A, RL2', @RW4+d8	XORL A, A, RL2', @RW4+d8	XORL A, A, RL2', @RW4+d8	R4', @RW4+d8 #8, rel'	R4', @RW4+d8 #8, rel'
+5	ADDL A, A, RL2', @RW5+d8	ADDL A, A, RL2', @RW5+d8	SUBL A, A, RL2', @RW5+d8	SUBL A, A, RL2', @RW5+d8	RW5', @RW5+d8 #16, rel'	CMPL A, A, RL2', @RW5+d8	CMPL A, A, RL2', @RW5+d8	CMPL A, A, RL2', @RW5+d8	ANDL A, A, RL2', @RW5+d8	ANDL A, A, RL2', @RW5+d8	ORL A, A, RL2', @RW5+d8	ORL A, A, RL2', @RW5+d8	XORL A, A, RL2', @RW5+d8	XORL A, A, RL2', @RW5+d8	R5', @RW5+d8 #8, rel'	R5', @RW5+d8 #8, rel'
+6	ADDL A, A, RL3', @RW6+d8	ADDL A, A, RL3', @RW6+d8	SUBL A, A, RL3', @RW6+d8	SUBL A, A, RL3', @RW6+d8	RW6', @RW6+d8 #16, rel'	CMPL A, A, RL3', @RW6+d8	CMPL A, A, RL3', @RW6+d8	CMPL A, A, RL3', @RW6+d8	ANDL A, A, RL3', @RW6+d8	ANDL A, A, RL3', @RW6+d8	ORL A, A, RL3', @RW6+d8	ORL A, A, RL3', @RW6+d8	XORL A, A, RL3', @RW6+d8	XORL A, A, RL3', @RW6+d8	R6', @RW6+d8 #8, rel'	R6', @RW6+d8 #8, rel'
+7	ADDL A, A, RL3', @RW7+d8	ADDL A, A, RL3', @RW7+d8	SUBL A, A, RL3', @RW7+d8	SUBL A, A, RL3', @RW7+d8	RW7', @RW7+d8 #16, rel'	CMPL A, A, RL3', @RW7+d8	CMPL A, A, RL3', @RW7+d8	CMPL A, A, RL3', @RW7+d8	ANDL A, A, RL3', @RW7+d8	ANDL A, A, RL3', @RW7+d8	ORL A, A, RL3', @RW7+d8	ORL A, A, RL3', @RW7+d8	XORL A, A, RL3', @RW7+d8	XORL A, A, RL3', @RW7+d8	R7', @RW7+d8 #8, rel'	R7', @RW7+d8 #8, rel'
+8	ADDL A, A, @RW0', @RW0+d16	ADDL A, A, @RW0', @RW0+d16	SUBL A, A, @RW0', @RW0+d16	SUBL A, A, @RW0', @RW0+d16	@RW0', @RW0+d16 #16, rel'	CMPL A, A, @RW0', @RW0+d16	CMPL A, A, @RW0', @RW0+d16	CMPL A, A, @RW0', @RW0+d16	ANDL A, A, @RW0', @RW0+d16	ANDL A, A, @RW0', @RW0+d16	ORL A, A, @RW0', @RW0+d16	ORL A, A, @RW0', @RW0+d16	XORL A, A, @RW0', @RW0+d16	XORL A, A, @RW0', @RW0+d16	@RW0', @RW0+d16 #8, rel'	@RW0', @RW0+d16 #8, rel'
+9	ADDL A, A, @RW1', @RW1+d16	ADDL A, A, @RW1', @RW1+d16	SUBL A, A, @RW1', @RW1+d16	SUBL A, A, @RW1', @RW1+d16	@RW1', @RW1+d16 #16, rel'	CMPL A, A, @RW1', @RW1+d16	CMPL A, A, @RW1', @RW1+d16	CMPL A, A, @RW1', @RW1+d16	ANDL A, A, @RW1', @RW1+d16	ANDL A, A, @RW1', @RW1+d16	ORL A, A, @RW1', @RW1+d16	ORL A, A, @RW1', @RW1+d16	XORL A, A, @RW1', @RW1+d16	XORL A, A, @RW1', @RW1+d16	@RW1', @RW1+d16 #8, rel'	@RW1', @RW1+d16 #8, rel'
+A	ADDL A, A, @RW2', @RW2+d16	ADDL A, A, @RW2', @RW2+d16	SUBL A, A, @RW2', @RW2+d16	SUBL A, A, @RW2', @RW2+d16	@RW2', @RW2+d16 #16, rel'	CMPL A, A, @RW2', @RW2+d16	CMPL A, A, @RW2', @RW2+d16	CMPL A, A, @RW2', @RW2+d16	ANDL A, A, @RW2', @RW2+d16	ANDL A, A, @RW2', @RW2+d16	ORL A, A, @RW2', @RW2+d16	ORL A, A, @RW2', @RW2+d16	XORL A, A, @RW2', @RW2+d16	XORL A, A, @RW2', @RW2+d16	@RW2', @RW2+d16 #8, rel'	@RW2', @RW2+d16 #8, rel'
+B	ADDL A, A, @RW3', @RW3+d16	ADDL A, A, @RW3', @RW3+d16	SUBL A, A, @RW3', @RW3+d16	SUBL A, A, @RW3', @RW3+d16	@RW3', @RW3+d16 #16, rel'	CMPL A, A, @RW3', @RW3+d16	CMPL A, A, @RW3', @RW3+d16	CMPL A, A, @RW3', @RW3+d16	ANDL A, A, @RW3', @RW3+d16	ANDL A, A, @RW3', @RW3+d16	ORL A, A, @RW3', @RW3+d16	ORL A, A, @RW3', @RW3+d16	XORL A, A, @RW3', @RW3+d16	XORL A, A, @RW3', @RW3+d16	@RW3', @RW3+d16 #8, rel'	@RW3', @RW3+d16 #8, rel'
+C	ADDL A, A, @RW0+', @RW0+RW7	ADDL A, A, @RW0+', @RW0+RW7	SUBL A, A, @RW0+', @RW0+RW7	SUBL A, A, @RW0+', @RW0+RW7	Use prohibited	CMPL A, A, @RW0+', @RW0+RW7	CMPL A, A, @RW0+', @RW0+RW7	CMPL A, A, @RW0+', @RW0+RW7	ANDL A, A, @RW0+', @RW0+RW7	ANDL A, A, @RW0+', @RW0+RW7	ORL A, A, @RW0+', @RW0+RW7	ORL A, A, @RW0+', @RW0+RW7	XORL A, A, @RW0+', @RW0+RW7	XORL A, A, @RW0+', @RW0+RW7	Use prohibited	@RW0+RW7 #8, rel'
+D	ADDL A, A, @RW1+', @RW1+RW7	ADDL A, A, @RW1+', @RW1+RW7	SUBL A, A, @RW1+', @RW1+RW7	SUBL A, A, @RW1+', @RW1+RW7	Use prohibited	CMPL A, A, @RW1+', @RW1+RW7	CMPL A, A, @RW1+', @RW1+RW7	CMPL A, A, @RW1+', @RW1+RW7	ANDL A, A, @RW1+', @RW1+RW7	ANDL A, A, @RW1+', @RW1+RW7	ORL A, A, @RW1+', @RW1+RW7	ORL A, A, @RW1+', @RW1+RW7	XORL A, A, @RW1+', @RW1+RW7	XORL A, A, @RW1+', @RW1+RW7	Use prohibited	@RW1+RW7 #8, rel'
+E	ADDL A, A, @RW2+', @PC+d16	ADDL A, A, @RW2+', @PC+d16	SUBL A, A, @RW2+', @PC+d16	SUBL A, A, @RW2+', @PC+d16	Use prohibited	CMPL A, A, @RW2+', @PC+d16	CMPL A, A, @RW2+', @PC+d16	CMPL A, A, @RW2+', @PC+d16	ANDL A, A, @RW2+', @PC+d16	ANDL A, A, @RW2+', @PC+d16	ORL A, A, @RW2+', @PC+d16	ORL A, A, @RW2+', @PC+d16	XORL A, A, @RW2+', @PC+d16	XORL A, A, @RW2+', @PC+d16	Use prohibited	@PC+d16 #8, rel'
+F	ADDL A, A, @RW3+', addr16	ADDL A, A, @RW3+', addr16	SUBL A, A, @RW3+', addr16	SUBL A, A, @RW3+', addr16	Use prohibited	CMPL A, A, @RW3+', addr16	CMPL A, A, @RW3+', addr16	CMPL A, A, @RW3+', addr16	ANDL A, A, @RW3+', addr16	ANDL A, A, @RW3+', addr16	ORL A, A, @RW3+', addr16	ORL A, A, @RW3+', addr16	XORL A, A, @RW3+', addr16	XORL A, A, @RW3+', addr16	Use prohibited	addr16 #8, rel'

**Table C.9-7 ea Instruction 2 (First Byte = 71<sub>H</sub>)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	JMPP @RLO, @RW0+d8	JMPP @ @RLO, @RW0+d8	CALLP @RLO, @RW0+d8	CALLP @ @RLO, @RW0+d8	INCL RLO, @RW0+d8	INCL RLO, @RW0+d8	DECL RLO, @RW0+d8	DECL RLO, @RW0+d8	MOVL A, RLO, @RW0+d8	MOVL A, @RW0+d8	MOVL RLO, A, @RW0+d8	MOVL RLO, A, @RW0+d8	MOV R0, #8, @RW0+d8	MOV R0, #8, @RW0+d8	MOVEA A, RWO, @RW0+d8	MOVEA A, RWO, @RW0+d8
+1	JMPP @RLO, @RW1+d8	JMPP @ @RLO, @RW1+d8	CALLP @RLO, @RW1+d8	CALLP @ @RLO, @RW1+d8	INCL RLO, @RW1+d8	INCL RLO, @RW1+d8	DECL RLO, @RW1+d8	DECL RLO, @RW1+d8	MOVL A, RLO, @RW1+d8	MOVL A, @RW1+d8	MOVL RLO, A, @RW1+d8	MOVL RLO, A, @RW1+d8	MOV R1, #8, @RW1+d8	MOV R1, #8, @RW1+d8	MOVEA A, RW1, @RW1+d8	MOVEA A, RW1, @RW1+d8
+2	JMPP @RL1, @RW2+d8	JMPP @ @RL1, @RW2+d8	CALLP @RL1, @RW2+d8	CALLP @ @RL1, @RW2+d8	INCL RL1, @RW2+d8	INCL RL1, @RW2+d8	DECL RL1, @RW2+d8	DECL RL1, @RW2+d8	MOVL A, RL1, @RW2+d8	MOVL A, @RW2+d8	MOVL RL1, A, @RW2+d8	MOVL RL1, A, @RW2+d8	MOV R2, #8, @RW2+d8	MOV R2, #8, @RW2+d8	MOVEA A, RW2, @RW2+d8	MOVEA A, RW2, @RW2+d8
+3	JMPP @RL1, @RW3+d8	JMPP @ @RL1, @RW3+d8	CALLP @RL1, @RW3+d8	CALLP @ @RL1, @RW3+d8	INCL RL1, @RW3+d8	INCL RL1, @RW3+d8	DECL RL1, @RW3+d8	DECL RL1, @RW3+d8	MOVL A, RL1, @RW3+d8	MOVL A, @RW3+d8	MOVL RL1, A, @RW3+d8	MOVL RL1, A, @RW3+d8	MOV R3, #8, @RW3+d8	MOV R3, #8, @RW3+d8	MOVEA A, RW3, @RW3+d8	MOVEA A, RW3, @RW3+d8
+4	JMPP @RL2, @RW4+d8	JMPP @ @RL2, @RW4+d8	CALLP @RL2, @RW4+d8	CALLP @ @RL2, @RW4+d8	INCL RL2, @RW4+d8	INCL RL2, @RW4+d8	DECL RL2, @RW4+d8	DECL RL2, @RW4+d8	MOVL A, RL2, @RW4+d8	MOVL A, @RW4+d8	MOVL RL2, A, @RW4+d8	MOVL RL2, A, @RW4+d8	MOV R4, #8, @RW4+d8	MOV R4, #8, @RW4+d8	MOVEA A, RW4, @RW4+d8	MOVEA A, RW4, @RW4+d8
+5	JMPP @RL2, @RW5+d8	JMPP @ @RL2, @RW5+d8	CALLP @RL2, @RW5+d8	CALLP @ @RL2, @RW5+d8	INCL RL2, @RW5+d8	INCL RL2, @RW5+d8	DECL RL2, @RW5+d8	DECL RL2, @RW5+d8	MOVL A, RL2, @RW5+d8	MOVL A, @RW5+d8	MOVL RL2, A, @RW5+d8	MOVL RL2, A, @RW5+d8	MOV R5, #8, @RW5+d8	MOV R5, #8, @RW5+d8	MOVEA A, RW5, @RW5+d8	MOVEA A, RW5, @RW5+d8
+6	JMPP @RL3, @RW6+d8	JMPP @ @RL3, @RW6+d8	CALLP @RL3, @RW6+d8	CALLP @ @RL3, @RW6+d8	INCL RL3, @RW6+d8	INCL RL3, @RW6+d8	DECL RL3, @RW6+d8	DECL RL3, @RW6+d8	MOVL A, RL3, @RW6+d8	MOVL A, @RW6+d8	MOVL RL3, A, @RW6+d8	MOVL RL3, A, @RW6+d8	MOV R6, #8, @RW6+d8	MOV R6, #8, @RW6+d8	MOVEA A, RW6, @RW6+d8	MOVEA A, RW6, @RW6+d8
+7	JMPP @RL3, @RW7+d8	JMPP @ @RL3, @RW7+d8	CALLP @RL3, @RW7+d8	CALLP @ @RL3, @RW7+d8	INCL RL3, @RW7+d8	INCL RL3, @RW7+d8	DECL RL3, @RW7+d8	DECL RL3, @RW7+d8	MOVL A, RL3, @RW7+d8	MOVL A, @RW7+d8	MOVL RL3, A, @RW7+d8	MOVL RL3, A, @RW7+d8	MOV R7, #8, @RW7+d8	MOV R7, #8, @RW7+d8	MOVEA A, RW7, @RW7+d8	MOVEA A, RW7, @RW7+d8
+8	JMPP @RW0, @RW0+d16	JMPP @ @RW0, @RW0+d16	CALLP @RW0, @RW0+d16	CALLP @ @RW0, @RW0+d16	INCL @RW0, @RW0+d16	INCL @RW0, @RW0+d16	DECL @RW0, @RW0+d16	DECL @RW0, @RW0+d16	MOVL A, @RW0, @RW0+d16	MOVL A, @RW0+d16	MOVL @RW0, A, @RW0+d16	MOVL @RW0, A, @RW0+d16	MOV @RW0, #8, @RW0+d16	MOV @RW0, #8, @RW0+d16	MOVEA A, @RW0, @RW0+d16	MOVEA A, @RW0, @RW0+d16
+9	JMPP @RW1, @RW1+d16	JMPP @ @RW1, @RW1+d16	CALLP @RW1, @RW1+d16	CALLP @ @RW1, @RW1+d16	INCL @RW1, @RW1+d16	INCL @RW1, @RW1+d16	DECL @RW1, @RW1+d16	DECL @RW1, @RW1+d16	MOVL A, @RW1, @RW1+d16	MOVL A, @RW1+d16	MOVL @RW1, A, @RW1+d16	MOVL @RW1, A, @RW1+d16	MOV @RW1, #8, @RW1+d16	MOV @RW1, #8, @RW1+d16	MOVEA A, @RW1, @RW1+d16	MOVEA A, @RW1, @RW1+d16
+A	JMPP @RW2, @RW2+d16	JMPP @ @RW2, @RW2+d16	CALLP @RW2, @RW2+d16	CALLP @ @RW2, @RW2+d16	INCL @RW2, @RW2+d16	INCL @RW2, @RW2+d16	DECL @RW2, @RW2+d16	DECL @RW2, @RW2+d16	MOVL A, @RW2, @RW2+d16	MOVL A, @RW2+d16	MOVL @RW2, A, @RW2+d16	MOVL @RW2, A, @RW2+d16	MOV @RW2, #8, @RW2+d16	MOV @RW2, #8, @RW2+d16	MOVEA A, @RW2, @RW2+d16	MOVEA A, @RW2, @RW2+d16
+B	JMPP @RW3, @RW3+d16	JMPP @ @RW3, @RW3+d16	CALLP @RW3, @RW3+d16	CALLP @ @RW3, @RW3+d16	INCL @RW3, @RW3+d16	INCL @RW3, @RW3+d16	DECL @RW3, @RW3+d16	DECL @RW3, @RW3+d16	MOVL A, @RW3, @RW3+d16	MOVL A, @RW3+d16	MOVL @RW3, A, @RW3+d16	MOVL @RW3, A, @RW3+d16	MOV @RW3, #8, @RW3+d16	MOV @RW3, #8, @RW3+d16	MOVEA A, @RW3, @RW3+d16	MOVEA A, @RW3, @RW3+d16
+C	JMPP @RW0+, @RW0+RW7	JMPP @ @RW0+, @RW0+RW7	CALLP @RW0+, @RW0+RW7	CALLP @ @RW0+, @RW0+RW7	INCL @RW0+, @RW0+RW7	INCL @RW0+, @RW0+RW7	DECL @RW0+, @RW0+RW7	DECL @RW0+, @RW0+RW7	MOVL A, @RW0+, @RW0+RW7	MOVL A, @RW0+RW7	MOVL @RW0+, A, @RW0+RW7	MOVL @RW0+, A, @RW0+RW7	MOV @RW0+, #8, @RW0+RW7	MOV @RW0+, #8, @RW0+RW7	MOVEA A, @RW0+, @RW0+RW7	MOVEA A, @RW0+, @RW0+RW7
+D	JMPP @RW1+, @RW1+RW7	JMPP @ @RW1+, @RW1+RW7	CALLP @RW1+, @RW1+RW7	CALLP @ @RW1+, @RW1+RW7	INCL @RW1+, @RW1+RW7	INCL @RW1+, @RW1+RW7	DECL @RW1+, @RW1+RW7	DECL @RW1+, @RW1+RW7	MOVL A, @RW1+, @RW1+RW7	MOVL A, @RW1+RW7	MOVL @RW1+, A, @RW1+RW7	MOVL @RW1+, A, @RW1+RW7	MOV @RW1+, #8, @RW1+RW7	MOV @RW1+, #8, @RW1+RW7	MOVEA A, @RW1+, @RW1+RW7	MOVEA A, @RW1+, @RW1+RW7
+E	JMPP @RW2+, @PC+d16	JMPP @ @RW2+, @PC+d16	CALLP @RW2+, @PC+d16	CALLP @ @RW2+, @PC+d16	INCL @RW2+, @PC+d16	INCL @RW2+, @PC+d16	DECL @RW2+, @PC+d16	DECL @RW2+, @PC+d16	MOVL A, @RW2+, @PC+d16	MOVL A, @PC+d16	MOVL @RW2+, A, @PC+d16	MOVL @RW2+, A, @PC+d16	MOV @RW2+, #8, @PC+d16	MOV @RW2+, #8, @PC+d16	MOVEA A, @RW2+, @PC+d16	MOVEA A, @RW2+, @PC+d16
+F	JMPP @RW3+, @addr16	JMPP @ @RW3+, @addr16	CALLP @RW3+, @addr16	CALLP @ @RW3+, @addr16	INCL @RW3+, @addr16	INCL @RW3+, @addr16	DECL @RW3+, @addr16	DECL @RW3+, @addr16	MOVL A, @RW3+, @addr16	MOVL A, @addr16	MOVL @RW3+, A, @addr16	MOVL @RW3+, A, @addr16	MOV @RW3+, #8, @addr16	MOV @RW3+, #8, @addr16	MOVEA A, @RW3+, @addr16	MOVEA A, @RW3+, @addr16

Table C.9-8 ea Instruction 3 (First Byte = 72<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ROL R0, @RW0+d8	ROL R0, @RW0+d8	ROR R0, @RW0+d8	ROR R0, @RW0+d8	INC R0, @RW0+d8	INC R0, @RW0+d8	DEC R0, @RW0+d8	DEC R0, @RW0+d8	MOV A, R0, @RW0+d8	MOV A, R0, @RW0+d8	MOV R0, A, @RW0+d8	MOV R0, A, @RW0+d8	MOVX A, R0, @RW0+d8	MOVX A, R0, @RW0+d8	XCH A, R0, @RW0+d8	XCH A, R0, @RW0+d8
+1	ROL R1, @RW1+d8	ROL R1, @RW1+d8	ROR R1, @RW1+d8	ROR R1, @RW1+d8	INC R1, @RW1+d8	INC R1, @RW1+d8	DEC R1, @RW1+d8	DEC R1, @RW1+d8	MOV A, R1, @RW1+d8	MOV A, R1, @RW1+d8	MOV R1, A, @RW1+d8	MOV R1, A, @RW1+d8	MOVX A, R1, @RW1+d8	MOVX A, R1, @RW1+d8	XCH A, R1, @RW1+d8	XCH A, R1, @RW1+d8
+2	ROL R2, @RW2+d8	ROL R2, @RW2+d8	ROR R2, @RW2+d8	ROR R2, @RW2+d8	INC R2, @RW2+d8	INC R2, @RW2+d8	DEC R2, @RW2+d8	DEC R2, @RW2+d8	MOV A, R2, @RW2+d8	MOV A, R2, @RW2+d8	MOV R2, A, @RW2+d8	MOV R2, A, @RW2+d8	MOVX A, R2, @RW2+d8	MOVX A, R2, @RW2+d8	XCH A, R2, @RW2+d8	XCH A, R2, @RW2+d8
+3	ROL R3, @RW3+d8	ROL R3, @RW3+d8	ROR R3, @RW3+d8	ROR R3, @RW3+d8	INC R3, @RW3+d8	INC R3, @RW3+d8	DEC R3, @RW3+d8	DEC R3, @RW3+d8	MOV A, R3, @RW3+d8	MOV A, R3, @RW3+d8	MOV R3, A, @RW3+d8	MOV R3, A, @RW3+d8	MOVX A, R3, @RW3+d8	MOVX A, R3, @RW3+d8	XCH A, R3, @RW3+d8	XCH A, R3, @RW3+d8
+4	ROL R4, @RW4+d8	ROL R4, @RW4+d8	ROR R4, @RW4+d8	ROR R4, @RW4+d8	INC R4, @RW4+d8	INC R4, @RW4+d8	DEC R4, @RW4+d8	DEC R4, @RW4+d8	MOV A, R4, @RW4+d8	MOV A, R4, @RW4+d8	MOV R4, A, @RW4+d8	MOV R4, A, @RW4+d8	MOVX A, R4, @RW4+d8	MOVX A, R4, @RW4+d8	XCH A, R4, @RW4+d8	XCH A, R4, @RW4+d8
+5	ROL R5, @RW5+d8	ROL R5, @RW5+d8	ROR R5, @RW5+d8	ROR R5, @RW5+d8	INC R5, @RW5+d8	INC R5, @RW5+d8	DEC R5, @RW5+d8	DEC R5, @RW5+d8	MOV A, R5, @RW5+d8	MOV A, R5, @RW5+d8	MOV R5, A, @RW5+d8	MOV R5, A, @RW5+d8	MOVX A, R5, @RW5+d8	MOVX A, R5, @RW5+d8	XCH A, R5, @RW5+d8	XCH A, R5, @RW5+d8
+6	ROL R6, @RW6+d8	ROL R6, @RW6+d8	ROR R6, @RW6+d8	ROR R6, @RW6+d8	INC R6, @RW6+d8	INC R6, @RW6+d8	DEC R6, @RW6+d8	DEC R6, @RW6+d8	MOV A, R6, @RW6+d8	MOV A, R6, @RW6+d8	MOV R6, A, @RW6+d8	MOV R6, A, @RW6+d8	MOVX A, R6, @RW6+d8	MOVX A, R6, @RW6+d8	XCH A, R6, @RW6+d8	XCH A, R6, @RW6+d8
+7	ROL R7, @RW7+d8	ROL R7, @RW7+d8	ROR R7, @RW7+d8	ROR R7, @RW7+d8	INC R7, @RW7+d8	INC R7, @RW7+d8	DEC R7, @RW7+d8	DEC R7, @RW7+d8	MOV A, R7, @RW7+d8	MOV A, R7, @RW7+d8	MOV R7, A, @RW7+d8	MOV R7, A, @RW7+d8	MOVX A, R7, @RW7+d8	MOVX A, R7, @RW7+d8	XCH A, R7, @RW7+d8	XCH A, R7, @RW7+d8
+8	ROL @RW0, @RW0+d16	ROL @RW0, @RW0+d16	ROR @RW0, @RW0+d16	ROR @RW0, @RW0+d16	INC @RW0, @RW0+d16	INC @RW0, @RW0+d16	DEC @RW0, @RW0+d16	DEC @RW0, @RW0+d16	MOV A, @RW0, @RW0+d16	MOV A, @RW0, @RW0+d16	MOV @RW0, A, @RW0+d16	MOV @RW0, A, @RW0+d16	MOVX A, @RW0, @RW0+d16	MOVX A, @RW0, @RW0+d16	XCH A, @RW0, @RW0+d16	XCH A, @RW0, @RW0+d16
+9	ROL @RW1, @RW1+d16	ROL @RW1, @RW1+d16	ROR @RW1, @RW1+d16	ROR @RW1, @RW1+d16	INC @RW1, @RW1+d16	INC @RW1, @RW1+d16	DEC @RW1, @RW1+d16	DEC @RW1, @RW1+d16	MOV A, @RW1, @RW1+d16	MOV A, @RW1, @RW1+d16	MOV @RW1, A, @RW1+d16	MOV @RW1, A, @RW1+d16	MOVX A, @RW1, @RW1+d16	MOVX A, @RW1, @RW1+d16	XCH A, @RW1, @RW1+d16	XCH A, @RW1, @RW1+d16
+A	ROL @RW2, @RW2+d16	ROL @RW2, @RW2+d16	ROR @RW2, @RW2+d16	ROR @RW2, @RW2+d16	INC @RW2, @RW2+d16	INC @RW2, @RW2+d16	DEC @RW2, @RW2+d16	DEC @RW2, @RW2+d16	MOV A, @RW2, @RW2+d16	MOV A, @RW2, @RW2+d16	MOV @RW2, A, @RW2+d16	MOV @RW2, A, @RW2+d16	MOVX A, @RW2, @RW2+d16	MOVX A, @RW2, @RW2+d16	XCH A, @RW2, @RW2+d16	XCH A, @RW2, @RW2+d16
+B	ROL @RW3, @RW3+d16	ROL @RW3, @RW3+d16	ROR @RW3, @RW3+d16	ROR @RW3, @RW3+d16	INC @RW3, @RW3+d16	INC @RW3, @RW3+d16	DEC @RW3, @RW3+d16	DEC @RW3, @RW3+d16	MOV A, @RW3, @RW3+d16	MOV A, @RW3, @RW3+d16	MOV @RW3, A, @RW3+d16	MOV @RW3, A, @RW3+d16	MOVX A, @RW3, @RW3+d16	MOVX A, @RW3, @RW3+d16	XCH A, @RW3, @RW3+d16	XCH A, @RW3, @RW3+d16
+C	ROL @RW0+, @RW0+RW7	ROL @RW0+, @RW0+RW7	ROR @RW0+, @RW0+RW7	ROR @RW0+, @RW0+RW7	INC @RW0+, @RW0+RW7	INC @RW0+, @RW0+RW7	DEC @RW0+, @RW0+RW7	DEC @RW0+, @RW0+RW7	MOV A, @RW0+, @RW0+RW7	MOV A, @RW0+, @RW0+RW7	MOV @RW0+, A, @RW0+RW7	MOV @RW0+, A, @RW0+RW7	MOVX A, @RW0+, @RW0+RW7	MOVX A, @RW0+, @RW0+RW7	XCH A, @RW0+, @RW0+RW7	XCH A, @RW0+, @RW0+RW7
+D	ROL @RW1+, @RW1+RW7	ROL @RW1+, @RW1+RW7	ROR @RW1+, @RW1+RW7	ROR @RW1+, @RW1+RW7	INC @RW1+, @RW1+RW7	INC @RW1+, @RW1+RW7	DEC @RW1+, @RW1+RW7	DEC @RW1+, @RW1+RW7	MOV A, @RW1+, @RW1+RW7	MOV A, @RW1+, @RW1+RW7	MOV @RW1+, A, @RW1+RW7	MOV @RW1+, A, @RW1+RW7	MOVX A, @RW1+, @RW1+RW7	MOVX A, @RW1+, @RW1+RW7	XCH A, @RW1+, @RW1+RW7	XCH A, @RW1+, @RW1+RW7
+E	ROL @RW2+, @PC+d16	ROL @RW2+, @PC+d16	ROR @RW2+, @PC+d16	ROR @RW2+, @PC+d16	INC @RW2+, @PC+d16	INC @RW2+, @PC+d16	DEC @RW2+, @PC+d16	DEC @RW2+, @PC+d16	MOV A, @RW2+, @PC+d16	MOV A, @RW2+, @PC+d16	MOV @RW2+, A, @PC+d16	MOV @RW2+, A, @PC+d16	MOVX A, @RW2+, @PC+d16	MOVX A, @RW2+, @PC+d16	XCH A, @RW2+, @PC+d16	XCH A, @RW2+, @PC+d16
+F	ROL @RW3+, addr16	ROL @RW3+, addr16	ROR @RW3+, addr16	ROR @RW3+, addr16	INC @RW3+, addr16	INC @RW3+, addr16	DEC @RW3+, addr16	DEC @RW3+, addr16	MOV A, @RW3+, addr16	MOV A, @RW3+, addr16	MOV @RW3+, A, addr16	MOV @RW3+, A, addr16	MOVX A, @RW3+, addr16	MOVX A, @RW3+, addr16	XCH A, @RW3+, addr16	XCH A, @RW3+, addr16

**Table C.9-9 ea Instruction 4 (First Byte = 73<sub>H</sub>)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	JMP @RW0, @RW0+d8	JMP @RW0, @RW0+d8	CALL RW0, @RW0+d8	CALL RW0, @RW0+d8	INCW RW0, @RW0+d8	INCW RW0, @RW0+d8	DECW RW0, @RW0+d8	DECW RW0, @RW0+d8	MOVW A, RW0, @RW0+d8	MOVW A, RW0, @RW0+d8	MOVW RW0, #16, @RW0+d8, #16	MOVW RW0, #16, @RW0+d8, #16	MOVW RW0, #16, @RW0+d8, #16	MOVW RW0, #16, @RW0+d8, #16	XCHW A, RW0, @RW0+d8	XCHW A, RW0, @RW0+d8
+1	JMP @RW1, @RW1+d8	JMP @RW1, @RW1+d8	CALL RW1, @RW1+d8	CALL RW1, @RW1+d8	INCW RW1, @RW1+d8	INCW RW1, @RW1+d8	DECW RW1, @RW1+d8	DECW RW1, @RW1+d8	MOVW A, RW1, @RW1+d8	MOVW A, RW1, @RW1+d8	MOVW RW1, #16, @RW1+d8, #16	MOVW RW1, #16, @RW1+d8, #16	MOVW RW1, #16, @RW1+d8, #16	MOVW RW1, #16, @RW1+d8, #16	XCHW A, RW1, @RW1+d8	XCHW A, RW1, @RW1+d8
+2	JMP @RW2, @RW2+d8	JMP @RW2, @RW2+d8	CALL RW2, @RW2+d8	CALL RW2, @RW2+d8	INCW RW2, @RW2+d8	INCW RW2, @RW2+d8	DECW RW2, @RW2+d8	DECW RW2, @RW2+d8	MOVW A, RW2, @RW2+d8	MOVW A, RW2, @RW2+d8	MOVW RW2, #16, @RW2+d8, #16	MOVW RW2, #16, @RW2+d8, #16	MOVW RW2, #16, @RW2+d8, #16	MOVW RW2, #16, @RW2+d8, #16	XCHW A, RW2, @RW2+d8	XCHW A, RW2, @RW2+d8
+3	JMP @RW3, @RW3+d8	JMP @RW3, @RW3+d8	CALL RW3, @RW3+d8	CALL RW3, @RW3+d8	INCW RW3, @RW3+d8	INCW RW3, @RW3+d8	DECW RW3, @RW3+d8	DECW RW3, @RW3+d8	MOVW A, RW3, @RW3+d8	MOVW A, RW3, @RW3+d8	MOVW RW3, #16, @RW3+d8, #16	MOVW RW3, #16, @RW3+d8, #16	MOVW RW3, #16, @RW3+d8, #16	MOVW RW3, #16, @RW3+d8, #16	XCHW A, RW3, @RW3+d8	XCHW A, RW3, @RW3+d8
+4	JMP @RW4, @RW4+d8	JMP @RW4, @RW4+d8	CALL RW4, @RW4+d8	CALL RW4, @RW4+d8	INCW RW4, @RW4+d8	INCW RW4, @RW4+d8	DECW RW4, @RW4+d8	DECW RW4, @RW4+d8	MOVW A, RW4, @RW4+d8	MOVW A, RW4, @RW4+d8	MOVW RW4, #16, @RW4+d8, #16	MOVW RW4, #16, @RW4+d8, #16	MOVW RW4, #16, @RW4+d8, #16	MOVW RW4, #16, @RW4+d8, #16	XCHW A, RW4, @RW4+d8	XCHW A, RW4, @RW4+d8
+5	JMP @RW5, @RW5+d8	JMP @RW5, @RW5+d8	CALL RW5, @RW5+d8	CALL RW5, @RW5+d8	INCW RW5, @RW5+d8	INCW RW5, @RW5+d8	DECW RW5, @RW5+d8	DECW RW5, @RW5+d8	MOVW A, RW5, @RW5+d8	MOVW A, RW5, @RW5+d8	MOVW RW5, #16, @RW5+d8, #16	MOVW RW5, #16, @RW5+d8, #16	MOVW RW5, #16, @RW5+d8, #16	MOVW RW5, #16, @RW5+d8, #16	XCHW A, RW5, @RW5+d8	XCHW A, RW5, @RW5+d8
+6	JMP @RW6, @RW6+d8	JMP @RW6, @RW6+d8	CALL RW6, @RW6+d8	CALL RW6, @RW6+d8	INCW RW6, @RW6+d8	INCW RW6, @RW6+d8	DECW RW6, @RW6+d8	DECW RW6, @RW6+d8	MOVW A, RW6, @RW6+d8	MOVW A, RW6, @RW6+d8	MOVW RW6, #16, @RW6+d8, #16	MOVW RW6, #16, @RW6+d8, #16	MOVW RW6, #16, @RW6+d8, #16	MOVW RW6, #16, @RW6+d8, #16	XCHW A, RW6, @RW6+d8	XCHW A, RW6, @RW6+d8
+7	JMP @RW7, @RW7+d8	JMP @RW7, @RW7+d8	CALL RW7, @RW7+d8	CALL RW7, @RW7+d8	INCW RW7, @RW7+d8	INCW RW7, @RW7+d8	DECW RW7, @RW7+d8	DECW RW7, @RW7+d8	MOVW A, RW7, @RW7+d8	MOVW A, RW7, @RW7+d8	MOVW RW7, #16, @RW7+d8, #16	MOVW RW7, #16, @RW7+d8, #16	MOVW RW7, #16, @RW7+d8, #16	MOVW RW7, #16, @RW7+d8, #16	XCHW A, RW7, @RW7+d8	XCHW A, RW7, @RW7+d8
+8	JMP @RW0, @RW0+d16	JMP @RW0, @RW0+d16	CALL @RW0, @RW0+d16	CALL @RW0, @RW0+d16	INCW @RW0, @RW0+d16	INCW @RW0, @RW0+d16	DECW @RW0, @RW0+d16	DECW @RW0, @RW0+d16	MOVW A, @RW0, @RW0+d16	MOVW A, @RW0, @RW0+d16	MOVW @RW0, #16, @RW0+d16, #16	MOVW @RW0, #16, @RW0+d16, #16	MOVW @RW0, #16, @RW0+d16, #16	MOVW @RW0, #16, @RW0+d16, #16	XCHW A, @RW0, @RW0+d16	XCHW A, @RW0, @RW0+d16
+9	JMP @RW1, @RW1+d16	JMP @RW1, @RW1+d16	CALL @RW1, @RW1+d16	CALL @RW1, @RW1+d16	INCW @RW1, @RW1+d16	INCW @RW1, @RW1+d16	DECW @RW1, @RW1+d16	DECW @RW1, @RW1+d16	MOVW A, @RW1, @RW1+d16	MOVW A, @RW1, @RW1+d16	MOVW @RW1, #16, @RW1+d16, #16	MOVW @RW1, #16, @RW1+d16, #16	MOVW @RW1, #16, @RW1+d16, #16	MOVW @RW1, #16, @RW1+d16, #16	XCHW A, @RW1, @RW1+d16	XCHW A, @RW1, @RW1+d16
+A	JMP @RW2, @RW2+d16	JMP @RW2, @RW2+d16	CALL @RW2, @RW2+d16	CALL @RW2, @RW2+d16	INCW @RW2, @RW2+d16	INCW @RW2, @RW2+d16	DECW @RW2, @RW2+d16	DECW @RW2, @RW2+d16	MOVW A, @RW2, @RW2+d16	MOVW A, @RW2, @RW2+d16	MOVW @RW2, #16, @RW2+d16, #16	MOVW @RW2, #16, @RW2+d16, #16	MOVW @RW2, #16, @RW2+d16, #16	MOVW @RW2, #16, @RW2+d16, #16	XCHW A, @RW2, @RW2+d16	XCHW A, @RW2, @RW2+d16
+B	JMP @RW3, @RW3+d16	JMP @RW3, @RW3+d16	CALL @RW3, @RW3+d16	CALL @RW3, @RW3+d16	INCW @RW3, @RW3+d16	INCW @RW3, @RW3+d16	DECW @RW3, @RW3+d16	DECW @RW3, @RW3+d16	MOVW A, @RW3, @RW3+d16	MOVW A, @RW3, @RW3+d16	MOVW @RW3, #16, @RW3+d16, #16	MOVW @RW3, #16, @RW3+d16, #16	MOVW @RW3, #16, @RW3+d16, #16	MOVW @RW3, #16, @RW3+d16, #16	XCHW A, @RW3, @RW3+d16	XCHW A, @RW3, @RW3+d16
+C	JMP @RW0+, @RW0+RW7	JMP @RW0+, @RW0+RW7	CALL @RW0+, @RW0+RW7	CALL @RW0+, @RW0+RW7	INCW @RW0+, @RW0+RW7	INCW @RW0+, @RW0+RW7	DECW @RW0+, @RW0+RW7	DECW @RW0+, @RW0+RW7	MOVW A, @RW0+, @RW0+RW7	MOVW A, @RW0+, @RW0+RW7	MOVW @RW0+, #16, @RW0+RW7, #16	MOVW @RW0+, #16, @RW0+RW7, #16	MOVW @RW0+, #16, @RW0+RW7, #16	MOVW @RW0+, #16, @RW0+RW7, #16	XCHW A, @RW0+, @RW0+RW7	XCHW A, @RW0+, @RW0+RW7
+D	JMP @RW1+, @RW1+RW7	JMP @RW1+, @RW1+RW7	CALL @RW1+, @RW1+RW7	CALL @RW1+, @RW1+RW7	INCW @RW1+, @RW1+RW7	INCW @RW1+, @RW1+RW7	DECW @RW1+, @RW1+RW7	DECW @RW1+, @RW1+RW7	MOVW A, @RW1+, @RW1+RW7	MOVW A, @RW1+, @RW1+RW7	MOVW @RW1+, #16, @RW1+RW7, #16	MOVW @RW1+, #16, @RW1+RW7, #16	MOVW @RW1+, #16, @RW1+RW7, #16	MOVW @RW1+, #16, @RW1+RW7, #16	XCHW A, @RW1+, @RW1+RW7	XCHW A, @RW1+, @RW1+RW7
+E	JMP @RW2+, @RW2+PC+d16	JMP @RW2+, @RW2+PC+d16	CALL @RW2+, @RW2+PC+d16	CALL @RW2+, @RW2+PC+d16	INCW @RW2+, @RW2+PC+d16	INCW @RW2+, @RW2+PC+d16	DECW @RW2+, @RW2+PC+d16	DECW @RW2+, @RW2+PC+d16	MOVW A, @RW2+, @RW2+PC+d16	MOVW A, @RW2+, @RW2+PC+d16	MOVW @RW2+, #16, @RW2+PC+d16, #16	MOVW @RW2+, #16, @RW2+PC+d16, #16	MOVW @RW2+, #16, @RW2+PC+d16, #16	MOVW @RW2+, #16, @RW2+PC+d16, #16	XCHW A, @RW2+, @RW2+PC+d16	XCHW A, @RW2+, @RW2+PC+d16
+F	JMP @RW3+, @RW3+addr16	JMP @RW3+, @RW3+addr16	CALL @RW3+, @RW3+addr16	CALL @RW3+, @RW3+addr16	INCW @RW3+, @RW3+addr16	INCW @RW3+, @RW3+addr16	DECW @RW3+, @RW3+addr16	DECW @RW3+, @RW3+addr16	MOVW A, @RW3+, @RW3+addr16	MOVW A, @RW3+, @RW3+addr16	MOVW @RW3+, #16, @RW3+addr16, #16	MOVW @RW3+, #16, @RW3+addr16, #16	MOVW @RW3+, #16, @RW3+addr16, #16	MOVW @RW3+, #16, @RW3+addr16, #16	XCHW A, @RW3+, @RW3+addr16	XCHW A, @RW3+, @RW3+addr16

Table C.9-10 ea Instruction 5 (First Byte = 74<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADD A, R0', @RW0+d8	SUB A, R0', @RW0+d8	SUB A, R0', @RW0+d8	SUB A, R0', @RW0+d8	ADDC A, R0', @RW0+d8	ADDC A, R0', @RW0+d8	CMP A, R0', @RW0+d8	CMP A, R0', @RW0+d8	AND A, R0', @RW0+d8	AND A, R0', @RW0+d8	OR A, R0', @RW0+d8	OR A, R0', @RW0+d8	XOR A, R0', @RW0+d8	XOR A, R0', @RW0+d8	DBNZ R0, r'RW0+d8, r	DBNZ @R0, r'RW0+d8, r
+1	ADD A, R1', @RW1+d8	SUB A, R1', @RW1+d8	SUB A, R1', @RW1+d8	SUB A, R1', @RW1+d8	ADDC A, R1', @RW1+d8	ADDC A, R1', @RW1+d8	CMP A, R1', @RW1+d8	CMP A, R1', @RW1+d8	AND A, R1', @RW1+d8	AND A, R1', @RW1+d8	OR A, R1', @RW1+d8	OR A, R1', @RW1+d8	XOR A, R1', @RW1+d8	XOR A, R1', @RW1+d8	DBNZ R1, r'RW1+d8, r	DBNZ @R1, r'RW1+d8, r
+2	ADD A, R2', @RW2+d8	SUB A, R2', @RW2+d8	SUB A, R2', @RW2+d8	SUB A, R2', @RW2+d8	ADDC A, R2', @RW2+d8	ADDC A, R2', @RW2+d8	CMP A, R2', @RW2+d8	CMP A, R2', @RW2+d8	AND A, R2', @RW2+d8	AND A, R2', @RW2+d8	OR A, R2', @RW2+d8	OR A, R2', @RW2+d8	XOR A, R2', @RW2+d8	XOR A, R2', @RW2+d8	DBNZ R2, r'RW2+d8, r	DBNZ @R2, r'RW2+d8, r
+3	ADD A, R3', @RW3+d8	SUB A, R3', @RW3+d8	SUB A, R3', @RW3+d8	SUB A, R3', @RW3+d8	ADDC A, R3', @RW3+d8	ADDC A, R3', @RW3+d8	CMP A, R3', @RW3+d8	CMP A, R3', @RW3+d8	AND A, R3', @RW3+d8	AND A, R3', @RW3+d8	OR A, R3', @RW3+d8	OR A, R3', @RW3+d8	XOR A, R3', @RW3+d8	XOR A, R3', @RW3+d8	DBNZ R3, r'RW3+d8, r	DBNZ @R3, r'RW3+d8, r
+4	ADD A, R4', @RW4+d8	SUB A, R4', @RW4+d8	SUB A, R4', @RW4+d8	SUB A, R4', @RW4+d8	ADDC A, R4', @RW4+d8	ADDC A, R4', @RW4+d8	CMP A, R4', @RW4+d8	CMP A, R4', @RW4+d8	AND A, R4', @RW4+d8	AND A, R4', @RW4+d8	OR A, R4', @RW4+d8	OR A, R4', @RW4+d8	XOR A, R4', @RW4+d8	XOR A, R4', @RW4+d8	DBNZ R4, r'RW4+d8, r	DBNZ @R4, r'RW4+d8, r
+5	ADD A, R5', @RW5+d8	SUB A, R5', @RW5+d8	SUB A, R5', @RW5+d8	SUB A, R5', @RW5+d8	ADDC A, R5', @RW5+d8	ADDC A, R5', @RW5+d8	CMP A, R5', @RW5+d8	CMP A, R5', @RW5+d8	AND A, R5', @RW5+d8	AND A, R5', @RW5+d8	OR A, R5', @RW5+d8	OR A, R5', @RW5+d8	XOR A, R5', @RW5+d8	XOR A, R5', @RW5+d8	DBNZ R5, r'RW5+d8, r	DBNZ @R5, r'RW5+d8, r
+6	ADD A, R6', @RW6+d8	SUB A, R6', @RW6+d8	SUB A, R6', @RW6+d8	SUB A, R6', @RW6+d8	ADDC A, R6', @RW6+d8	ADDC A, R6', @RW6+d8	CMP A, R6', @RW6+d8	CMP A, R6', @RW6+d8	AND A, R6', @RW6+d8	AND A, R6', @RW6+d8	OR A, R6', @RW6+d8	OR A, R6', @RW6+d8	XOR A, R6', @RW6+d8	XOR A, R6', @RW6+d8	DBNZ R6, r'RW6+d8, r	DBNZ @R6, r'RW6+d8, r
+7	ADD A, R7', @RW7+d8	SUB A, R7', @RW7+d8	SUB A, R7', @RW7+d8	SUB A, R7', @RW7+d8	ADDC A, R7', @RW7+d8	ADDC A, R7', @RW7+d8	CMP A, R7', @RW7+d8	CMP A, R7', @RW7+d8	AND A, R7', @RW7+d8	AND A, R7', @RW7+d8	OR A, R7', @RW7+d8	OR A, R7', @RW7+d8	XOR A, R7', @RW7+d8	XOR A, R7', @RW7+d8	DBNZ R7, r'RW7+d8, r	DBNZ @R7, r'RW7+d8, r
+8	ADD A, @RW0, @RW0+d16	SUB A, @RW0, @RW0+d16	SUB A, @RW0, @RW0+d16	SUB A, @RW0, @RW0+d16	ADDC A, @RW0, @RW0+d16	ADDC A, @RW0, @RW0+d16	CMP A, @RW0, @RW0+d16	CMP A, @RW0, @RW0+d16	AND A, @RW0, @RW0+d16	AND A, @RW0, @RW0+d16	OR A, @RW0, @RW0+d16	OR A, @RW0, @RW0+d16	XOR A, @RW0, @RW0+d16	XOR A, @RW0, @RW0+d16	DBNZ @RW0, r'W0+d16, r	DBNZ @R0, r'W0+d16, r
+9	ADD A, @RW1, @RW1+d16	SUB A, @RW1, @RW1+d16	SUB A, @RW1, @RW1+d16	SUB A, @RW1, @RW1+d16	ADDC A, @RW1, @RW1+d16	ADDC A, @RW1, @RW1+d16	CMP A, @RW1, @RW1+d16	CMP A, @RW1, @RW1+d16	AND A, @RW1, @RW1+d16	AND A, @RW1, @RW1+d16	OR A, @RW1, @RW1+d16	OR A, @RW1, @RW1+d16	XOR A, @RW1, @RW1+d16	XOR A, @RW1, @RW1+d16	DBNZ @RW1, r'W1+d16, r	DBNZ @R1, r'W1+d16, r
+A	ADD A, @RW2, @RW2+d16	SUB A, @RW2, @RW2+d16	SUB A, @RW2, @RW2+d16	SUB A, @RW2, @RW2+d16	ADDC A, @RW2, @RW2+d16	ADDC A, @RW2, @RW2+d16	CMP A, @RW2, @RW2+d16	CMP A, @RW2, @RW2+d16	AND A, @RW2, @RW2+d16	AND A, @RW2, @RW2+d16	OR A, @RW2, @RW2+d16	OR A, @RW2, @RW2+d16	XOR A, @RW2, @RW2+d16	XOR A, @RW2, @RW2+d16	DBNZ @RW2, r'W2+d16, r	DBNZ @R2, r'W2+d16, r
+B	ADD A, @RW3, @RW3+d16	SUB A, @RW3, @RW3+d16	SUB A, @RW3, @RW3+d16	SUB A, @RW3, @RW3+d16	ADDC A, @RW3, @RW3+d16	ADDC A, @RW3, @RW3+d16	CMP A, @RW3, @RW3+d16	CMP A, @RW3, @RW3+d16	AND A, @RW3, @RW3+d16	AND A, @RW3, @RW3+d16	OR A, @RW3, @RW3+d16	OR A, @RW3, @RW3+d16	XOR A, @RW3, @RW3+d16	XOR A, @RW3, @RW3+d16	DBNZ @RW3, r'W3+d16, r	DBNZ @R3, r'W3+d16, r
+C	ADD A, @RW0+, @RW0+RW7	SUB A, @RW0+, @RW0+RW7	SUB A, @RW0+, @RW0+RW7	SUB A, @RW0+, @RW0+RW7	ADDC A, @RW0+, @RW0+RW7	ADDC A, @RW0+, @RW0+RW7	CMP A, @RW0+, @RW0+RW7	CMP A, @RW0+, @RW0+RW7	AND A, @RW0+, @RW0+RW7	AND A, @RW0+, @RW0+RW7	OR A, @RW0+, @RW0+RW7	OR A, @RW0+, @RW0+RW7	XOR A, @RW0+, @RW0+RW7	XOR A, @RW0+, @RW0+RW7	DBNZ @RW0+, r'W0+RW7, r	DBNZ @R0, r'W0+RW7, r
+D	ADD A, @RW1+, @RW1+RW7	SUB A, @RW1+, @RW1+RW7	SUB A, @RW1+, @RW1+RW7	SUB A, @RW1+, @RW1+RW7	ADDC A, @RW1+, @RW1+RW7	ADDC A, @RW1+, @RW1+RW7	CMP A, @RW1+, @RW1+RW7	CMP A, @RW1+, @RW1+RW7	AND A, @RW1+, @RW1+RW7	AND A, @RW1+, @RW1+RW7	OR A, @RW1+, @RW1+RW7	OR A, @RW1+, @RW1+RW7	XOR A, @RW1+, @RW1+RW7	XOR A, @RW1+, @RW1+RW7	DBNZ @RW1+, r'W1+RW7, r	DBNZ @R1, r'W1+RW7, r
+E	ADD A, @RW2+, @PC+d16	SUB A, @RW2+, @PC+d16	SUB A, @RW2+, @PC+d16	SUB A, @RW2+, @PC+d16	ADDC A, @RW2+, @PC+d16	ADDC A, @RW2+, @PC+d16	CMP A, @RW2+, @PC+d16	CMP A, @RW2+, @PC+d16	AND A, @RW2+, @PC+d16	AND A, @RW2+, @PC+d16	OR A, @RW2+, @PC+d16	OR A, @RW2+, @PC+d16	XOR A, @RW2+, @PC+d16	XOR A, @RW2+, @PC+d16	DBNZ @RW2+, r'PC+d16, r	DBNZ @R2, r'PC+d16, r
+F	ADD A, @RW3+, A, addr16	SUB A, @RW3+, A, addr16	SUB A, @RW3+, A, addr16	SUB A, @RW3+, A, addr16	ADDC A, @RW3+, A, addr16	ADDC A, @RW3+, A, addr16	CMP A, @RW3+, A, addr16	CMP A, @RW3+, A, addr16	AND A, @RW3+, A, addr16	AND A, @RW3+, A, addr16	OR A, @RW3+, A, addr16	OR A, @RW3+, A, addr16	XOR A, @RW3+, A, addr16	XOR A, @RW3+, A, addr16	DBNZ @RW3+, r', addr16, r	DBNZ @R3, r', addr16, r

**Table C.9-11 ea Instruction 6 (First Byte = 75<sub>H</sub>)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADD R0, A, @RW0+d8, A	SUB R0, A, @RW0+d8, A	SUB R0, A, @RW0+d8, A	SUB A, R0, @RW0+d8, A	SUBC A, R0, @RW0+d8, A	SUBC A, R0, @RW0+d8, A	NEG R0, @RW0+d8, A	NEG A, @RW0+d8, A	AND R0, A, @RW0+d8, A	AND R0, A, @RW0+d8, A	OR R0, A, @RW0+d8, A	OR A, @RW0+d8, A	XOR R0, A, @RW0+d8, A	XOR A, @RW0+d8, A	NOT R0, @RW0+d8, A	NOT R0, @RW0+d8, A
+1	ADD R1, A, @RW1+d8, A	SUB R1, A, @RW1+d8, A	SUB R1, A, @RW1+d8, A	SUB A, R1, @RW1+d8, A	SUBC A, R1, @RW1+d8, A	SUBC A, R1, @RW1+d8, A	NEG R1, @RW1+d8, A	NEG A, @RW1+d8, A	AND R1, A, @RW1+d8, A	AND R1, A, @RW1+d8, A	OR R1, A, @RW1+d8, A	OR A, @RW1+d8, A	XOR R1, A, @RW1+d8, A	XOR A, @RW1+d8, A	NOT R1, @RW1+d8, A	NOT R1, @RW1+d8, A
+2	ADD R2, A, @RW2+d8, A	SUB R2, A, @RW2+d8, A	SUB R2, A, @RW2+d8, A	SUB A, R2, @RW2+d8, A	SUBC A, R2, @RW2+d8, A	SUBC A, R2, @RW2+d8, A	NEG R2, @RW2+d8, A	NEG A, @RW2+d8, A	AND R2, A, @RW2+d8, A	AND R2, A, @RW2+d8, A	OR R2, A, @RW2+d8, A	OR A, @RW2+d8, A	XOR R2, A, @RW2+d8, A	XOR A, @RW2+d8, A	NOT R2, @RW2+d8, A	NOT R2, @RW2+d8, A
+3	ADD R3, A, @RW3+d8, A	SUB R3, A, @RW3+d8, A	SUB R3, A, @RW3+d8, A	SUB A, R3, @RW3+d8, A	SUBC A, R3, @RW3+d8, A	SUBC A, R3, @RW3+d8, A	NEG R3, @RW3+d8, A	NEG A, @RW3+d8, A	AND R3, A, @RW3+d8, A	AND R3, A, @RW3+d8, A	OR R3, A, @RW3+d8, A	OR A, @RW3+d8, A	XOR R3, A, @RW3+d8, A	XOR A, @RW3+d8, A	NOT R3, @RW3+d8, A	NOT R3, @RW3+d8, A
+4	ADD R4, A, @RW4+d8, A	SUB R4, A, @RW4+d8, A	SUB R4, A, @RW4+d8, A	SUB A, R4, @RW4+d8, A	SUBC A, R4, @RW4+d8, A	SUBC A, R4, @RW4+d8, A	NEG R4, @RW4+d8, A	NEG A, @RW4+d8, A	AND R4, A, @RW4+d8, A	AND R4, A, @RW4+d8, A	OR R4, A, @RW4+d8, A	OR A, @RW4+d8, A	XOR R4, A, @RW4+d8, A	XOR A, @RW4+d8, A	NOT R4, @RW4+d8, A	NOT R4, @RW4+d8, A
+5	ADD R5, A, @RW5+d8, A	SUB R5, A, @RW5+d8, A	SUB R5, A, @RW5+d8, A	SUB A, R5, @RW5+d8, A	SUBC A, R5, @RW5+d8, A	SUBC A, R5, @RW5+d8, A	NEG R5, @RW5+d8, A	NEG A, @RW5+d8, A	AND R5, A, @RW5+d8, A	AND R5, A, @RW5+d8, A	OR R5, A, @RW5+d8, A	OR A, @RW5+d8, A	XOR R5, A, @RW5+d8, A	XOR A, @RW5+d8, A	NOT R5, @RW5+d8, A	NOT R5, @RW5+d8, A
+6	ADD R6, A, @RW6+d8, A	SUB R6, A, @RW6+d8, A	SUB R6, A, @RW6+d8, A	SUB A, R6, @RW6+d8, A	SUBC A, R6, @RW6+d8, A	SUBC A, R6, @RW6+d8, A	NEG R6, @RW6+d8, A	NEG A, @RW6+d8, A	AND R6, A, @RW6+d8, A	AND R6, A, @RW6+d8, A	OR R6, A, @RW6+d8, A	OR A, @RW6+d8, A	XOR R6, A, @RW6+d8, A	XOR A, @RW6+d8, A	NOT R6, @RW6+d8, A	NOT R6, @RW6+d8, A
+7	ADD R7, A, @RW7+d8, A	SUB R7, A, @RW7+d8, A	SUB R7, A, @RW7+d8, A	SUB A, R7, @RW7+d8, A	SUBC A, R7, @RW7+d8, A	SUBC A, R7, @RW7+d8, A	NEG R7, @RW7+d8, A	NEG A, @RW7+d8, A	AND R7, A, @RW7+d8, A	AND R7, A, @RW7+d8, A	OR R7, A, @RW7+d8, A	OR A, @RW7+d8, A	XOR R7, A, @RW7+d8, A	XOR A, @RW7+d8, A	NOT R7, @RW7+d8, A	NOT R7, @RW7+d8, A
+8	ADD @RW0, A, @RW0+d16, A	SUB @RW0, A, @RW0+d16, A	SUB @RW0, A, @RW0+d16, A	SUB A, @RW0, @RW0+d16, A	SUBC A, @RW0, @RW0+d16, A	SUBC A, @RW0, @RW0+d16, A	NEG @RW0, @RW0+d16, A	NEG A, @RW0+d16, A	AND @RW0, A, @RW0+d16, A	AND @RW0, A, @RW0+d16, A	OR @RW0, A, @RW0+d16, A	OR A, @RW0+d16, A	XOR @RW0, A, @RW0+d16, A	XOR A, @RW0+d16, A	NOT @RW0, @RW0+d16, A	NOT @RW0, @RW0+d16, A
+9	ADD @RW1, A, @RW1+d16, A	SUB @RW1, A, @RW1+d16, A	SUB @RW1, A, @RW1+d16, A	SUB A, @RW1, @RW1+d16, A	SUBC A, @RW1, @RW1+d16, A	SUBC A, @RW1, @RW1+d16, A	NEG @RW1, @RW1+d16, A	NEG A, @RW1+d16, A	AND @RW1, A, @RW1+d16, A	AND @RW1, A, @RW1+d16, A	OR @RW1, A, @RW1+d16, A	OR A, @RW1+d16, A	XOR @RW1, A, @RW1+d16, A	XOR A, @RW1+d16, A	NOT @RW1, @RW1+d16, A	NOT @RW1, @RW1+d16, A
+A	ADD @RW2, A, @RW2+d16, A	SUB @RW2, A, @RW2+d16, A	SUB @RW2, A, @RW2+d16, A	SUB A, @RW2, @RW2+d16, A	SUBC A, @RW2, @RW2+d16, A	SUBC A, @RW2, @RW2+d16, A	NEG @RW2, @RW2+d16, A	NEG A, @RW2+d16, A	AND @RW2, A, @RW2+d16, A	AND @RW2, A, @RW2+d16, A	OR @RW2, A, @RW2+d16, A	OR A, @RW2+d16, A	XOR @RW2, A, @RW2+d16, A	XOR A, @RW2+d16, A	NOT @RW2, @RW2+d16, A	NOT @RW2, @RW2+d16, A
+B	ADD @RW3, A, @RW3+d16, A	SUB @RW3, A, @RW3+d16, A	SUB @RW3, A, @RW3+d16, A	SUB A, @RW3, @RW3+d16, A	SUBC A, @RW3, @RW3+d16, A	SUBC A, @RW3, @RW3+d16, A	NEG @RW3, @RW3+d16, A	NEG A, @RW3+d16, A	AND @RW3, A, @RW3+d16, A	AND @RW3, A, @RW3+d16, A	OR @RW3, A, @RW3+d16, A	OR A, @RW3+d16, A	XOR @RW3, A, @RW3+d16, A	XOR A, @RW3+d16, A	NOT @RW3, @RW3+d16, A	NOT @RW3, @RW3+d16, A
+C	ADD @RW0+, A, @RW0+RW7, A	SUB @RW0+, A, @RW0+RW7, A	SUB @RW0+, A, @RW0+RW7, A	SUB A, @RW0+, @RW0+RW7, A	SUBC A, @RW0+, @RW0+RW7, A	SUBC A, @RW0+, @RW0+RW7, A	NEG @RW0+, @RW0+RW7, A	NEG A, @RW0+RW7, A	AND @RW0+, A, @RW0+RW7, A	AND @RW0+, A, @RW0+RW7, A	OR @RW0+, A, @RW0+RW7, A	OR A, @RW0+RW7, A	XOR @RW0+, A, @RW0+RW7, A	XOR A, @RW0+RW7, A	NOT @RW0+, @RW0+RW7, A	NOT @RW0+, @RW0+RW7, A
+D	ADD @RW1+, A, @RW1+RW7, A	SUB @RW1+, A, @RW1+RW7, A	SUB @RW1+, A, @RW1+RW7, A	SUB A, @RW1+, @RW1+RW7, A	SUBC A, @RW1+, @RW1+RW7, A	SUBC A, @RW1+, @RW1+RW7, A	NEG @RW1+, @RW1+RW7, A	NEG A, @RW1+RW7, A	AND @RW1+, A, @RW1+RW7, A	AND @RW1+, A, @RW1+RW7, A	OR @RW1+, A, @RW1+RW7, A	OR A, @RW1+RW7, A	XOR @RW1+, A, @RW1+RW7, A	XOR A, @RW1+RW7, A	NOT @RW1+, @RW1+RW7, A	NOT @RW1+, @RW1+RW7, A
+E	ADD @RW2+, A, @PC+d16, A	SUB @RW2+, A, @PC+d16, A	SUB @RW2+, A, @PC+d16, A	SUB A, @RW2+, @PC+d16, A	SUBC A, @RW2+, @PC+d16, A	SUBC A, @RW2+, @PC+d16, A	NEG @RW2+, @PC+d16, A	NEG A, @PC+d16, A	AND @RW2+, A, @PC+d16, A	AND @RW2+, A, @PC+d16, A	OR @RW2+, A, @PC+d16, A	OR A, @PC+d16, A	XOR @RW2+, A, @PC+d16, A	XOR A, @PC+d16, A	NOT @RW2+, @PC+d16, A	NOT @RW2+, @PC+d16, A
+F	ADD @RW3+, A, addr16, A	SUB @RW3+, A, addr16, A	SUB @RW3+, A, addr16, A	SUB A, @RW3+, addr16, A	SUBC A, @RW3+, addr16, A	SUBC A, @RW3+, addr16, A	NEG @RW3+, addr16, A	NEG A, addr16, A	AND @RW3+, A, addr16, A	AND @RW3+, A, addr16, A	OR @RW3+, A, addr16, A	OR A, addr16, A	XOR @RW3+, A, addr16, A	XOR A, addr16, A	NOT @RW3+, addr16, A	NOT @RW3+, addr16, A

Table C.9-12 ea Instruction 7 (First Byte = 76<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADDW A, A, RW0, @RW0+d8	ADDW A, SUBW A, A, RW0, @RW0+d8	SUBW A, SUBW A, A, RW0, @RW0+d8	SUBW A, SUBW A, A, RW0, @RW0+d8	ADDCW A, A, RW0, @RW0+d8	ADDCW A, A, RW0, @RW0+d8	CMPW A, A, RW0, @RW0+d8	CMPW A, A, RW0, @RW0+d8	ANDW A, A, RW0, @RW0+d8	ANDW A, A, RW0, @RW0+d8	ORW A, A, RW0, @RW0+d8	ORW A, A, RW0, @RW0+d8	XORW A, A, RW0, @RW0+d8	XORW A, A, RW0, @RW0+d8	DWBNZ A, RW0, r' @RW0+d8, r	DWBNZ A, RW0, r' @RW0+d8, r
+1	ADDW A, A, RW1, @RW1+d8	ADDW A, SUBW A, A, RW1, @RW1+d8	SUBW A, SUBW A, A, RW1, @RW1+d8	SUBW A, SUBW A, A, RW1, @RW1+d8	ADDCW A, A, RW1, @RW1+d8	ADDCW A, A, RW1, @RW1+d8	CMPW A, A, RW1, @RW1+d8	CMPW A, A, RW1, @RW1+d8	ANDW A, A, RW1, @RW1+d8	ANDW A, A, RW1, @RW1+d8	ORW A, A, RW1, @RW1+d8	ORW A, A, RW1, @RW1+d8	XORW A, A, RW1, @RW1+d8	XORW A, A, RW1, @RW1+d8	DWBNZ A, RW1, r' @RW1+d8, r	DWBNZ A, RW1, r' @RW1+d8, r
+2	ADDW A, A, RW2, @RW2+d8	ADDW A, SUBW A, A, RW2, @RW2+d8	SUBW A, SUBW A, A, RW2, @RW2+d8	SUBW A, SUBW A, A, RW2, @RW2+d8	ADDCW A, A, RW2, @RW2+d8	ADDCW A, A, RW2, @RW2+d8	CMPW A, A, RW2, @RW2+d8	CMPW A, A, RW2, @RW2+d8	ANDW A, A, RW2, @RW2+d8	ANDW A, A, RW2, @RW2+d8	ORW A, A, RW2, @RW2+d8	ORW A, A, RW2, @RW2+d8	XORW A, A, RW2, @RW2+d8	XORW A, A, RW2, @RW2+d8	DWBNZ A, RW2, r' @RW2+d8, r	DWBNZ A, RW2, r' @RW2+d8, r
+3	ADDW A, A, RW3, @RW3+d8	ADDW A, SUBW A, A, RW3, @RW3+d8	SUBW A, SUBW A, A, RW3, @RW3+d8	SUBW A, SUBW A, A, RW3, @RW3+d8	ADDCW A, A, RW3, @RW3+d8	ADDCW A, A, RW3, @RW3+d8	CMPW A, A, RW3, @RW3+d8	CMPW A, A, RW3, @RW3+d8	ANDW A, A, RW3, @RW3+d8	ANDW A, A, RW3, @RW3+d8	ORW A, A, RW3, @RW3+d8	ORW A, A, RW3, @RW3+d8	XORW A, A, RW3, @RW3+d8	XORW A, A, RW3, @RW3+d8	DWBNZ A, RW3, r' @RW3+d8, r	DWBNZ A, RW3, r' @RW3+d8, r
+4	ADDW A, A, RW4, @RW4+d8	ADDW A, SUBW A, A, RW4, @RW4+d8	SUBW A, SUBW A, A, RW4, @RW4+d8	SUBW A, SUBW A, A, RW4, @RW4+d8	ADDCW A, A, RW4, @RW4+d8	ADDCW A, A, RW4, @RW4+d8	CMPW A, A, RW4, @RW4+d8	CMPW A, A, RW4, @RW4+d8	ANDW A, A, RW4, @RW4+d8	ANDW A, A, RW4, @RW4+d8	ORW A, A, RW4, @RW4+d8	ORW A, A, RW4, @RW4+d8	XORW A, A, RW4, @RW4+d8	XORW A, A, RW4, @RW4+d8	DWBNZ A, RW4, r' @RW4+d8, r	DWBNZ A, RW4, r' @RW4+d8, r
+5	ADDW A, A, RW5, @RW5+d8	ADDW A, SUBW A, A, RW5, @RW5+d8	SUBW A, SUBW A, A, RW5, @RW5+d8	SUBW A, SUBW A, A, RW5, @RW5+d8	ADDCW A, A, RW5, @RW5+d8	ADDCW A, A, RW5, @RW5+d8	CMPW A, A, RW5, @RW5+d8	CMPW A, A, RW5, @RW5+d8	ANDW A, A, RW5, @RW5+d8	ANDW A, A, RW5, @RW5+d8	ORW A, A, RW5, @RW5+d8	ORW A, A, RW5, @RW5+d8	XORW A, A, RW5, @RW5+d8	XORW A, A, RW5, @RW5+d8	DWBNZ A, RW5, r' @RW5+d8, r	DWBNZ A, RW5, r' @RW5+d8, r
+6	ADDW A, A, RW6, @RW6+d8	ADDW A, SUBW A, A, RW6, @RW6+d8	SUBW A, SUBW A, A, RW6, @RW6+d8	SUBW A, SUBW A, A, RW6, @RW6+d8	ADDCW A, A, RW6, @RW6+d8	ADDCW A, A, RW6, @RW6+d8	CMPW A, A, RW6, @RW6+d8	CMPW A, A, RW6, @RW6+d8	ANDW A, A, RW6, @RW6+d8	ANDW A, A, RW6, @RW6+d8	ORW A, A, RW6, @RW6+d8	ORW A, A, RW6, @RW6+d8	XORW A, A, RW6, @RW6+d8	XORW A, A, RW6, @RW6+d8	DWBNZ A, RW6, r' @RW6+d8, r	DWBNZ A, RW6, r' @RW6+d8, r
+7	ADDW A, A, RW7, @RW7+d8	ADDW A, SUBW A, A, RW7, @RW7+d8	SUBW A, SUBW A, A, RW7, @RW7+d8	SUBW A, SUBW A, A, RW7, @RW7+d8	ADDCW A, A, RW7, @RW7+d8	ADDCW A, A, RW7, @RW7+d8	CMPW A, A, RW7, @RW7+d8	CMPW A, A, RW7, @RW7+d8	ANDW A, A, RW7, @RW7+d8	ANDW A, A, RW7, @RW7+d8	ORW A, A, RW7, @RW7+d8	ORW A, A, RW7, @RW7+d8	XORW A, A, RW7, @RW7+d8	XORW A, A, RW7, @RW7+d8	DWBNZ A, RW7, r' @RW7+d8, r	DWBNZ A, RW7, r' @RW7+d8, r
+8	ADDW A, A, RW0, @RW0+d16	ADDW A, SUBW A, A, RW0, @RW0+d16	SUBW A, SUBW A, A, RW0, @RW0+d16	SUBW A, SUBW A, A, RW0, @RW0+d16	ADDCW A, A, RW0, @RW0+d16	ADDCW A, A, RW0, @RW0+d16	CMPW A, A, RW0, @RW0+d16	CMPW A, A, RW0, @RW0+d16	ANDW A, A, RW0, @RW0+d16	ANDW A, A, RW0, @RW0+d16	ORW A, A, RW0, @RW0+d16	ORW A, A, RW0, @RW0+d16	XORW A, A, RW0, @RW0+d16	XORW A, A, RW0, @RW0+d16	DWBNZ A, RW0, r' @RW0+d16, r	DWBNZ A, RW0, r' @RW0+d16, r
+9	ADDW A, A, RW1, @RW1+d16	ADDW A, SUBW A, A, RW1, @RW1+d16	SUBW A, SUBW A, A, RW1, @RW1+d16	SUBW A, SUBW A, A, RW1, @RW1+d16	ADDCW A, A, RW1, @RW1+d16	ADDCW A, A, RW1, @RW1+d16	CMPW A, A, RW1, @RW1+d16	CMPW A, A, RW1, @RW1+d16	ANDW A, A, RW1, @RW1+d16	ANDW A, A, RW1, @RW1+d16	ORW A, A, RW1, @RW1+d16	ORW A, A, RW1, @RW1+d16	XORW A, A, RW1, @RW1+d16	XORW A, A, RW1, @RW1+d16	DWBNZ A, RW1, r' @RW1+d16, r	DWBNZ A, RW1, r' @RW1+d16, r
+A	ADDW A, A, RW2, @RW2+d16	ADDW A, SUBW A, A, RW2, @RW2+d16	SUBW A, SUBW A, A, RW2, @RW2+d16	SUBW A, SUBW A, A, RW2, @RW2+d16	ADDCW A, A, RW2, @RW2+d16	ADDCW A, A, RW2, @RW2+d16	CMPW A, A, RW2, @RW2+d16	CMPW A, A, RW2, @RW2+d16	ANDW A, A, RW2, @RW2+d16	ANDW A, A, RW2, @RW2+d16	ORW A, A, RW2, @RW2+d16	ORW A, A, RW2, @RW2+d16	XORW A, A, RW2, @RW2+d16	XORW A, A, RW2, @RW2+d16	DWBNZ A, RW2, r' @RW2+d16, r	DWBNZ A, RW2, r' @RW2+d16, r
+B	ADDW A, A, RW3, @RW3+d16	ADDW A, SUBW A, A, RW3, @RW3+d16	SUBW A, SUBW A, A, RW3, @RW3+d16	SUBW A, SUBW A, A, RW3, @RW3+d16	ADDCW A, A, RW3, @RW3+d16	ADDCW A, A, RW3, @RW3+d16	CMPW A, A, RW3, @RW3+d16	CMPW A, A, RW3, @RW3+d16	ANDW A, A, RW3, @RW3+d16	ANDW A, A, RW3, @RW3+d16	ORW A, A, RW3, @RW3+d16	ORW A, A, RW3, @RW3+d16	XORW A, A, RW3, @RW3+d16	XORW A, A, RW3, @RW3+d16	DWBNZ A, RW3, r' @RW3+d16, r	DWBNZ A, RW3, r' @RW3+d16, r
+C	ADDW A, A, RW0+, @RW0+RW7	ADDW A, SUBW A, A, RW0+, @RW0+RW7	SUBW A, SUBW A, A, RW0+, @RW0+RW7	SUBW A, SUBW A, A, RW0+, @RW0+RW7	ADDCW A, A, RW0+, @RW0+RW7	ADDCW A, A, RW0+, @RW0+RW7	CMPW A, A, RW0+, @RW0+RW7	CMPW A, A, RW0+, @RW0+RW7	ANDW A, A, RW0+, @RW0+RW7	ANDW A, A, RW0+, @RW0+RW7	ORW A, A, RW0+, @RW0+RW7	ORW A, A, RW0+, @RW0+RW7	XORW A, A, RW0+, @RW0+RW7	XORW A, A, RW0+, @RW0+RW7	DWBNZ A, RW0+, r' @RW0+RW7, r	DWBNZ A, RW0+, r' @RW0+RW7, r
+D	ADDW A, A, RW1+, @RW1+RW7	ADDW A, SUBW A, A, RW1+, @RW1+RW7	SUBW A, SUBW A, A, RW1+, @RW1+RW7	SUBW A, SUBW A, A, RW1+, @RW1+RW7	ADDCW A, A, RW1+, @RW1+RW7	ADDCW A, A, RW1+, @RW1+RW7	CMPW A, A, RW1+, @RW1+RW7	CMPW A, A, RW1+, @RW1+RW7	ANDW A, A, RW1+, @RW1+RW7	ANDW A, A, RW1+, @RW1+RW7	ORW A, A, RW1+, @RW1+RW7	ORW A, A, RW1+, @RW1+RW7	XORW A, A, RW1+, @RW1+RW7	XORW A, A, RW1+, @RW1+RW7	DWBNZ A, RW1+, r' @RW1+RW7, r	DWBNZ A, RW1+, r' @RW1+RW7, r
+E	ADDW A, A, RW2+, @PC+d16	ADDW A, SUBW A, A, RW2+, @PC+d16	SUBW A, SUBW A, A, RW2+, @PC+d16	SUBW A, SUBW A, A, RW2+, @PC+d16	ADDCW A, A, RW2+, @PC+d16	ADDCW A, A, RW2+, @PC+d16	CMPW A, A, RW2+, @PC+d16	CMPW A, A, RW2+, @PC+d16	ANDW A, A, RW2+, @PC+d16	ANDW A, A, RW2+, @PC+d16	ORW A, A, RW2+, @PC+d16	ORW A, A, RW2+, @PC+d16	XORW A, A, RW2+, @PC+d16	XORW A, A, RW2+, @PC+d16	DWBNZ A, RW2+, r' @PC+d16, r	DWBNZ A, RW2+, r' @PC+d16, r
+F	ADDW A, A, RW3+, addr 16	ADDW A, SUBW A, A, RW3+, addr 16	SUBW A, SUBW A, A, RW3+, addr 16	SUBW A, SUBW A, A, RW3+, addr 16	ADDCW A, A, RW3+, addr 16	ADDCW A, A, RW3+, addr 16	CMPW A, A, RW3+, addr 16	CMPW A, A, RW3+, addr 16	ANDW A, A, RW3+, addr 16	ANDW A, A, RW3+, addr 16	ORW A, A, RW3+, addr 16	ORW A, A, RW3+, addr 16	XORW A, A, RW3+, addr 16	XORW A, A, RW3+, addr 16	DWBNZ A, RW3+, r' addr 16, r	DWBNZ A, RW3+, r' addr 16, r



	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADDW RW0, A, @RW0+d8, A	SUBW RW0, A, @RW0+d8, A	SUBW RW0, A, @RW0+d8, A	SUBCW A, RW0, @RW0+d8	SUBCW A, RW0, @RW0+d8	NEGW RW0, @RW0+d8	NEGW RW0, @RW0+d8	NEGW RW0, @RW0+d8	ANDW RW0, A, @RW0+d8, A	ANDW RW0, A, @RW0+d8, A	ORW RW0, A, @RW0+d8, A	ORW RW0, A, @RW0+d8, A	XORW RW0, A, @RW0+d8, A	XORW RW0, A, @RW0+d8, A	NOTW RW0, @RW0+d8	NOTW @RW0+d8
+1	ADDW RW1, A, @RW1+d8, A	SUBW RW1, A, @RW1+d8, A	SUBW RW1, A, @RW1+d8, A	SUBCW A, RW1, @RW1+d8	SUBCW A, RW1, @RW1+d8	NEGW RW1, @RW1+d8	NEGW RW1, @RW1+d8	NEGW RW1, @RW1+d8	ANDW RW1, A, @RW1+d8, A	ANDW RW1, A, @RW1+d8, A	ORW RW1, A, @RW1+d8, A	ORW RW1, A, @RW1+d8, A	XORW RW1, A, @RW1+d8, A	XORW RW1, A, @RW1+d8, A	NOTW RW1, @RW1+d8	NOTW @RW1+d8
+2	ADDW RW2, A, @RW2+d8, A	SUBW RW2, A, @RW2+d8, A	SUBW RW2, A, @RW2+d8, A	SUBCW A, RW2, @RW2+d8	SUBCW A, RW2, @RW2+d8	NEGW RW2, @RW2+d8	NEGW RW2, @RW2+d8	NEGW RW2, @RW2+d8	ANDW RW2, A, @RW2+d8, A	ANDW RW2, A, @RW2+d8, A	ORW RW2, A, @RW2+d8, A	ORW RW2, A, @RW2+d8, A	XORW RW2, A, @RW2+d8, A	XORW RW2, A, @RW2+d8, A	NOTW RW2, @RW2+d8	NOTW @RW2+d8
+3	ADDW RW3, A, @RW3+d8, A	SUBW RW3, A, @RW3+d8, A	SUBW RW3, A, @RW3+d8, A	SUBCW A, RW3, @RW3+d8	SUBCW A, RW3, @RW3+d8	NEGW RW3, @RW3+d8	NEGW RW3, @RW3+d8	NEGW RW3, @RW3+d8	ANDW RW3, A, @RW3+d8, A	ANDW RW3, A, @RW3+d8, A	ORW RW3, A, @RW3+d8, A	ORW RW3, A, @RW3+d8, A	XORW RW3, A, @RW3+d8, A	XORW RW3, A, @RW3+d8, A	NOTW RW3, @RW3+d8	NOTW @RW3+d8
+4	ADDW RW4, A, @RW4+d8, A	SUBW RW4, A, @RW4+d8, A	SUBW RW4, A, @RW4+d8, A	SUBCW A, RW4, @RW4+d8	SUBCW A, RW4, @RW4+d8	NEGW RW4, @RW4+d8	NEGW RW4, @RW4+d8	NEGW RW4, @RW4+d8	ANDW RW4, A, @RW4+d8, A	ANDW RW4, A, @RW4+d8, A	ORW RW4, A, @RW4+d8, A	ORW RW4, A, @RW4+d8, A	XORW RW4, A, @RW4+d8, A	XORW RW4, A, @RW4+d8, A	NOTW RW4, @RW4+d8	NOTW @RW4+d8
+5	ADDW RW5, A, @RW5+d8, A	SUBW RW5, A, @RW5+d8, A	SUBW RW5, A, @RW5+d8, A	SUBCW A, RW5, @RW5+d8	SUBCW A, RW5, @RW5+d8	NEGW RW5, @RW5+d8	NEGW RW5, @RW5+d8	NEGW RW5, @RW5+d8	ANDW RW5, A, @RW5+d8, A	ANDW RW5, A, @RW5+d8, A	ORW RW5, A, @RW5+d8, A	ORW RW5, A, @RW5+d8, A	XORW RW5, A, @RW5+d8, A	XORW RW5, A, @RW5+d8, A	NOTW RW5, @RW5+d8	NOTW @RW5+d8
+6	ADDW RW6, A, @RW6+d8, A	SUBW RW6, A, @RW6+d8, A	SUBW RW6, A, @RW6+d8, A	SUBCW A, RW6, @RW6+d8	SUBCW A, RW6, @RW6+d8	NEGW RW6, @RW6+d8	NEGW RW6, @RW6+d8	NEGW RW6, @RW6+d8	ANDW RW6, A, @RW6+d8, A	ANDW RW6, A, @RW6+d8, A	ORW RW6, A, @RW6+d8, A	ORW RW6, A, @RW6+d8, A	XORW RW6, A, @RW6+d8, A	XORW RW6, A, @RW6+d8, A	NOTW RW6, @RW6+d8	NOTW @RW6+d8
+7	ADDW RW7, A, @RW7+d8, A	SUBW RW7, A, @RW7+d8, A	SUBW RW7, A, @RW7+d8, A	SUBCW A, RW7, @RW7+d8	SUBCW A, RW7, @RW7+d8	NEGW RW7, @RW7+d8	NEGW RW7, @RW7+d8	NEGW RW7, @RW7+d8	ANDW RW7, A, @RW7+d8, A	ANDW RW7, A, @RW7+d8, A	ORW RW7, A, @RW7+d8, A	ORW RW7, A, @RW7+d8, A	XORW RW7, A, @RW7+d8, A	XORW RW7, A, @RW7+d8, A	NOTW RW7, @RW7+d8	NOTW @RW7+d8
+8	ADDW @RW0, A, @RW0+d16, A	SUBW @RW0, A, @RW0+d16, A	SUBW @RW0, A, @RW0+d16, A	SUBCW A, @RW0, @RW0+d16	SUBCW A, @RW0, @RW0+d16	NEGW @RW0, @RW0+d16	NEGW @RW0, @RW0+d16	NEGW @RW0, @RW0+d16	ANDW @RW0, A, @RW0+d16, A	ANDW @RW0, A, @RW0+d16, A	ORW @RW0, A, @RW0+d16, A	ORW @RW0, A, @RW0+d16, A	XORW @RW0, A, @RW0+d16, A	XORW @RW0, A, @RW0+d16, A	NOTW @RW0, @RW0+d16	NOTW @RW0+d16
+9	ADDW @RW1, A, @RW1+d16, A	SUBW @RW1, A, @RW1+d16, A	SUBW @RW1, A, @RW1+d16, A	SUBCW A, @RW1, @RW1+d16	SUBCW A, @RW1, @RW1+d16	NEGW @RW1, @RW1+d16	NEGW @RW1, @RW1+d16	NEGW @RW1, @RW1+d16	ANDW @RW1, A, @RW1+d16, A	ANDW @RW1, A, @RW1+d16, A	ORW @RW1, A, @RW1+d16, A	ORW @RW1, A, @RW1+d16, A	XORW @RW1, A, @RW1+d16, A	XORW @RW1, A, @RW1+d16, A	NOTW @RW1, @RW1+d16	NOTW @RW1+d16
+A	ADDW @RW2, A, @RW2+d16, A	SUBW @RW2, A, @RW2+d16, A	SUBW @RW2, A, @RW2+d16, A	SUBCW A, @RW2, @RW2+d16	SUBCW A, @RW2, @RW2+d16	NEGW @RW2, @RW2+d16	NEGW @RW2, @RW2+d16	NEGW @RW2, @RW2+d16	ANDW @RW2, A, @RW2+d16, A	ANDW @RW2, A, @RW2+d16, A	ORW @RW2, A, @RW2+d16, A	ORW @RW2, A, @RW2+d16, A	XORW @RW2, A, @RW2+d16, A	XORW @RW2, A, @RW2+d16, A	NOTW @RW2, @RW2+d16	NOTW @RW2+d16
+B	ADDW @RW3, A, @RW3+d16, A	SUBW @RW3, A, @RW3+d16, A	SUBW @RW3, A, @RW3+d16, A	SUBCW A, @RW3, @RW3+d16	SUBCW A, @RW3, @RW3+d16	NEGW @RW3, @RW3+d16	NEGW @RW3, @RW3+d16	NEGW @RW3, @RW3+d16	ANDW @RW3, A, @RW3+d16, A	ANDW @RW3, A, @RW3+d16, A	ORW @RW3, A, @RW3+d16, A	ORW @RW3, A, @RW3+d16, A	XORW @RW3, A, @RW3+d16, A	XORW @RW3, A, @RW3+d16, A	NOTW @RW3, @RW3+d16	NOTW @RW3+d16
+C	ADDW @RW0+, A, @RW0+RW7, A															

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MULU A, R0' @RW0+d8	MULU A, R0' @RW0+d8	MULUW A, RW0' @RW0+d8	MULUW A, RW0' @RW0+d8	MUL A, R0' @RW0+d8	MUL A, R0' @RW0+d8	MULW A, RW0' @RW0+d8	MULW A, RW0' @RW0+d8	DIVU A, R0' @RW0+d8	DIVU A, R0' @RW0+d8	DIVUW A, RW0' @RW0+d8	DIVUW A, RW0' @RW0+d8	DIV A, R0' @RW0+d8	DIV A, R0' @RW0+d8	DIVW A, RW0' @RW0+d8	DIVW A, RW0' @RW0+d8
+1	MULU A, R1' @RW1+d8	MULU A, R1' @RW1+d8	MULUW A, RW1' @RW1+d8	MULUW A, RW1' @RW1+d8	MUL A, R1' @RW1+d8	MUL A, R1' @RW1+d8	MULW A, RW1' @RW1+d8	MULW A, RW1' @RW1+d8	DIVU A, R1' @RW1+d8	DIVU A, R1' @RW1+d8	DIVUW A, RW1' @RW1+d8	DIVUW A, RW1' @RW1+d8	DIV A, R1' @RW1+d8	DIV A, R1' @RW1+d8	DIVW A, RW1' @RW1+d8	DIVW A, RW1' @RW1+d8
+2	MULU A, R2' @RW2+d8	MULU A, R2' @RW2+d8	MULUW A, RW2' @RW2+d8	MULUW A, RW2' @RW2+d8	MUL A, R2' @RW2+d8	MUL A, R2' @RW2+d8	MULW A, RW2' @RW2+d8	MULW A, RW2' @RW2+d8	DIVU A, R2' @RW2+d8	DIVU A, R2' @RW2+d8	DIVUW A, RW2' @RW2+d8	DIVUW A, RW2' @RW2+d8	DIV A, R2' @RW2+d8	DIV A, R2' @RW2+d8	DIVW A, RW2' @RW2+d8	DIVW A, RW2' @RW2+d8
+3	MULU A, R3' @RW3+d8	MULU A, R3' @RW3+d8	MULUW A, RW3' @RW3+d8	MULUW A, RW3' @RW3+d8	MUL A, R3' @RW3+d8	MUL A, R3' @RW3+d8	MULW A, RW3' @RW3+d8	MULW A, RW3' @RW3+d8	DIVU A, R3' @RW3+d8	DIVU A, R3' @RW3+d8	DIVUW A, RW3' @RW3+d8	DIVUW A, RW3' @RW3+d8	DIV A, R3' @RW3+d8	DIV A, R3' @RW3+d8	DIVW A, RW3' @RW3+d8	DIVW A, RW3' @RW3+d8
+4	MULU A, R4' @RW4+d8	MULU A, R4' @RW4+d8	MULUW A, RW4' @RW4+d8	MULUW A, RW4' @RW4+d8	MUL A, R4' @RW4+d8	MUL A, R4' @RW4+d8	MULW A, RW4' @RW4+d8	MULW A, RW4' @RW4+d8	DIVU A, R4' @RW4+d8	DIVU A, R4' @RW4+d8	DIVUW A, RW4' @RW4+d8	DIVUW A, RW4' @RW4+d8	DIV A, R4' @RW4+d8	DIV A, R4' @RW4+d8	DIVW A, RW4' @RW4+d8	DIVW A, RW4' @RW4+d8
+5	MULU A, R5' @RW5+d8	MULU A, R5' @RW5+d8	MULUW A, RW5' @RW5+d8	MULUW A, RW5' @RW5+d8	MUL A, R5' @RW5+d8	MUL A, R5' @RW5+d8	MULW A, RW5' @RW5+d8	MULW A, RW5' @RW5+d8	DIVU A, R5' @RW5+d8	DIVU A, R5' @RW5+d8	DIVUW A, RW5' @RW5+d8	DIVUW A, RW5' @RW5+d8	DIV A, R5' @RW5+d8	DIV A, R5' @RW5+d8	DIVW A, RW5' @RW5+d8	DIVW A, RW5' @RW5+d8
+6	MULU A, R6' @RW6+d8	MULU A, R6' @RW6+d8	MULUW A, RW6' @RW6+d8	MULUW A, RW6' @RW6+d8	MUL A, R6' @RW6+d8	MUL A, R6' @RW6+d8	MULW A, RW6' @RW6+d8	MULW A, RW6' @RW6+d8	DIVU A, R6' @RW6+d8	DIVU A, R6' @RW6+d8	DIVUW A, RW6' @RW6+d8	DIVUW A, RW6' @RW6+d8	DIV A, R6' @RW6+d8	DIV A, R6' @RW6+d8	DIVW A, RW6' @RW6+d8	DIVW A, RW6' @RW6+d8
+7	MULU A, R7' @RW7+d8	MULU A, R7' @RW7+d8	MULUW A, RW7' @RW7+d8	MULUW A, RW7' @RW7+d8	MUL A, R7' @RW7+d8	MUL A, R7' @RW7+d8	MULW A, RW7' @RW7+d8	MULW A, RW7' @RW7+d8	DIVU A, R7' @RW7+d8	DIVU A, R7' @RW7+d8	DIVUW A, RW7' @RW7+d8	DIVUW A, RW7' @RW7+d8	DIV A, R7' @RW7+d8	DIV A, R7' @RW7+d8	DIVW A, RW7' @RW7+d8	DIVW A, RW7' @RW7+d8
+8	MULU A, @RW0' @RW0+d16	MULU A, @RW0' @RW0+d16	MULUW A, RW0' @RW0+d16	MULUW A, RW0' @RW0+d16	MUL A, @RW0' @RW0+d16	MUL A, @RW0' @RW0+d16	MULW A, @RW0' @RW0+d16	MULW A, @RW0' @RW0+d16	DIVU A, @RW0' @RW0+d16	DIVU A, @RW0' @RW0+d16	DIVUW A, RW0' @RW0+d16	DIVUW A, RW0' @RW0+d16	DIV A, @RW0' @RW0+d16	DIV A, @RW0' @RW0+d16	DIVW A, @RW0' @RW0+d16	DIVW A, @RW0' @RW0+d16
+9	MULU A, @RW1' @RW1+d16	MULU A, @RW1' @RW1+d16	MULUW A, RW1' @RW1+d16	MULUW A, RW1' @RW1+d16	MUL A, @RW1' @RW1+d16	MUL A, @RW1' @RW1+d16	MULW A, @RW1' @RW1+d16	MULW A, @RW1' @RW1+d16	DIVU A, @RW1' @RW1+d16	DIVU A, @RW1' @RW1+d16	DIVUW A, RW1' @RW1+d16	DIVUW A, RW1' @RW1+d16	DIV A, @RW1' @RW1+d16	DIV A, @RW1' @RW1+d16	DIVW A, @RW1' @RW1+d16	DIVW A, @RW1' @RW1+d16
+A	MULU A, @RW2' @RW2+d16	MULU A, @RW2' @RW2+d16	MULUW A, RW2' @RW2+d16	MULUW A, RW2' @RW2+d16	MUL A, @RW2' @RW2+d16	MUL A, @RW2' @RW2+d16	MULW A, @RW2' @RW2+d16	MULW A, @RW2' @RW2+d16	DIVU A, @RW2' @RW2+d16	DIVU A, @RW2' @RW2+d16	DIVUW A, RW2' @RW2+d16	DIVUW A, RW2' @RW2+d16	DIV A, @RW2' @RW2+d16	DIV A, @RW2' @RW2+d16	DIVW A, @RW2' @RW2+d16	DIVW A, @RW2' @RW2+d16
+B	MULU A, @RW3' @RW3+d16	MULU A, @RW3' @RW3+d16	MULUW A, RW3' @RW3+d16	MULUW A, RW3' @RW3+d16	MUL A, @RW3' @RW3+d16	MUL A, @RW3' @RW3+d16	MULW A, @RW3' @RW3+d16	MULW A, @RW3' @RW3+d16	DIVU A, @RW3' @RW3+d16	DIVU A, @RW3' @RW3+d16	DIVUW A, RW3' @RW3+d16	DIVUW A, RW3' @RW3+d16	DIV A, @RW3' @RW3+d16	DIV A, @RW3' @RW3+d16	DIVW A, @RW3' @RW3+d16	DIVW A, @RW3' @RW3+d16
+C	MULU A, @RW0+ @RW0+RW7	MULU A, @RW0+ @RW0+RW7	MULUW A, RW0+ @RW0+RW7	MULUW A, RW0+ @RW0+RW7	MUL A, @											

[illegible]

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOV R0, R0 @RW0+d8	MOV R0, R1 @RW0+d8	MOV R1, R0 @RW0+d8	MOV R2, R0 @RW0+d8	MOV R3, R0 @RW0+d8	MOV R4, R0 @RW0+d8	MOV R5, R0 @RW0+d8	MOV R6, R0 @RW0+d8	MOV R7, R0 @RW0+d8	MOV R8, R0 @RW0+d8	MOV R9, R0 @RW0+d8	MOV R10, R0 @RW0+d8	MOV R11, R0 @RW0+d8	MOV R12, R0 @RW0+d8	MOV R13, R0 @RW0+d8	MOV R14, R0 @RW0+d8
+1	MOV R0, R1 @RW1+d8	MOV R1, R0 @RW1+d8	MOV R2, R1 @RW1+d8	MOV R3, R1 @RW1+d8	MOV R4, R1 @RW1+d8	MOV R5, R1 @RW1+d8	MOV R6, R1 @RW1+d8	MOV R7, R1 @RW1+d8	MOV R8, R1 @RW1+d8	MOV R9, R1 @RW1+d8	MOV R10, R1 @RW1+d8	MOV R11, R1 @RW1+d8	MOV R12, R1 @RW1+d8	MOV R13, R1 @RW1+d8	MOV R14, R1 @RW1+d8	MOV R15, R1 @RW1+d8
+2	MOV R0, R2 @RW2+d8	MOV R1, R2 @RW2+d8	MOV R2, R1 @RW2+d8	MOV R3, R2 @RW2+d8	MOV R4, R2 @RW2+d8	MOV R5, R2 @RW2+d8	MOV R6, R2 @RW2+d8	MOV R7, R2 @RW2+d8	MOV R8, R2 @RW2+d8	MOV R9, R2 @RW2+d8	MOV R10, R2 @RW2+d8	MOV R11, R2 @RW2+d8	MOV R12, R2 @RW2+d8	MOV R13, R2 @RW2+d8	MOV R14, R2 @RW2+d8	MOV R15, R2 @RW2+d8
+3	MOV R0, R3 @RW3+d8	MOV R1, R3 @RW3+d8	MOV R2, R3 @RW3+d8	MOV R3, R2 @RW3+d8	MOV R4, R3 @RW3+d8	MOV R5, R3 @RW3+d8	MOV R6, R3 @RW3+d8	MOV R7, R3 @RW3+d8	MOV R8, R3 @RW3+d8	MOV R9, R3 @RW3+d8	MOV R10, R3 @RW3+d8	MOV R11, R3 @RW3+d8	MOV R12, R3 @RW3+d8	MOV R13, R3 @RW3+d8	MOV R14, R3 @RW3+d8	MOV R15, R3 @RW3+d8
+4	MOV R0, R4 @RW4+d8	MOV R1, R4 @RW4+d8	MOV R2, R4 @RW4+d8	MOV R3, R4 @RW4+d8	MOV R4, R3 @RW4+d8	MOV R5, R4 @RW4+d8	MOV R6, R4 @RW4+d8	MOV R7, R4 @RW4+d8	MOV R8, R4 @RW4+d8	MOV R9, R4 @RW4+d8	MOV R10, R4 @RW4+d8	MOV R11, R4 @RW4+d8	MOV R12, R4 @RW4+d8	MOV R13, R4 @RW4+d8	MOV R14, R4 @RW4+d8	MOV R15, R4 @RW4+d8
+5	MOV R0, R5 @RW5+d8	MOV R1, R5 @RW5+d8	MOV R2, R5 @RW5+d8	MOV R3, R5 @RW5+d8	MOV R4, R5 @RW5+d8	MOV R5, R4 @RW5+d8	MOV R6, R5 @RW5+d8	MOV R7, R5 @RW5+d8	MOV R8, R5 @RW5+d8	MOV R9, R5 @RW5+d8	MOV R10, R5 @RW5+d8	MOV R11, R5 @RW5+d8	MOV R12, R5 @RW5+d8	MOV R13, R5 @RW5+d8	MOV R14, R5 @RW5+d8	MOV R15, R5 @RW5+d8
+6	MOV R0, R6 @RW6+d8	MOV R1, R6 @RW6+d8	MOV R2, R6 @RW6+d8	MOV R3, R6 @RW6+d8	MOV R4, R6 @RW6+d8	MOV R5, R6 @RW6+d8	MOV R6, R5 @RW6+d8	MOV R7, R6 @RW6+d8	MOV R8, R6 @RW6+d8	MOV R9, R6 @RW6+d8	MOV R10, R6 @RW6+d8	MOV R11, R6 @RW6+d8	MOV R12, R6 @RW6+d8	MOV R13, R6 @RW6+d8	MOV R14, R6 @RW6+d8	MOV R15, R6 @RW6+d8
+7	MOV R0, R7 @RW7+d8	MOV R1, R7 @RW7+d8	MOV R2, R7 @RW7+d8	MOV R3, R7 @RW7+d8	MOV R4, R7 @RW7+d8	MOV R5, R7 @RW7+d8	MOV R6, R7 @RW7+d8	MOV R7, R6 @RW7+d8	MOV R8, R7 @RW7+d8	MOV R9, R7 @RW7+d8	MOV R10, R7 @RW7+d8	MOV R11, R7 @RW7+d8	MOV R12, R7 @RW7+d8	MOV R13, R7 @RW7+d8	MOV R14, R7 @RW7+d8	MOV R15, R7 @RW7+d8
+8	MOV R0, R8 @RW0+d16	MOV R1, R8 @RW0+d16	MOV R2, R8 @RW0+d16	MOV R3, R8 @RW0+d16	MOV R4, R8 @RW0+d16	MOV R5, R8 @RW0+d16	MOV R6, R8 @RW0+d16	MOV R7, R8 @RW0+d16	MOV R8, R7 @RW0+d16	MOV R9, R8 @RW0+d16	MOV R10, R8 @RW0+d16	MOV R11, R8 @RW0+d16	MOV R12, R8 @RW0+d16	MOV R13, R8 @RW0+d16	MOV R14, R8 @RW0+d16	MOV R15, R8 @RW0+d16
+9	MOV R0, R9 @RW1+d16	MOV R1, R9 @RW1+d16	MOV R2, R9 @RW1+d16	MOV R3, R9 @RW1+d16	MOV R4, R9 @RW1+d16	MOV R5, R9 @RW1+d16	MOV R6, R9 @RW1+d16	MOV R7, R9 @RW1+d16	MOV R8, R9 @RW1+d16	MOV R9, R8 @RW1+d16	MOV R10, R9 @RW1+d16	MOV R11, R9 @RW1+d16	MOV R12, R9 @RW1+d16	MOV R13, R9 @RW1+d16	MOV R14, R9 @RW1+d16	MOV R15, R9 @RW1+d16
+A	MOV R0, R10 @RW2+d16	MOV R1, R10 @RW2+d16	MOV R2, R10 @RW2+d16	MOV R3, R10 @RW2+d16	MOV R4, R10 @RW2+d16	MOV R5, R10 @RW2+d16	MOV R6, R10 @RW2+d16	MOV R7, R10 @RW2+d16	MOV R8, R10 @RW2+d16	MOV R9, R10 @RW2+d16	MOV R10, R10 @RW2+d16	MOV R11, R10 @RW2+d16	MOV R12, R10 @RW2+d16	MOV R13, R10 @RW2+d16	MOV R14, R10 @RW2+d16	MOV R15, R10 @RW2+d16
+B	MOV R0, R11 @RW3+d16	MOV R1, R11 @RW3+d16	MOV R2, R11 @RW3+d16	MOV R3, R11 @RW3+d16	MOV R4, R11 @RW3+d16	MOV R5, R11 @RW3+d16	MOV R6, R11 @RW3+d16	MOV R7, R11 @RW3+d16	MOV R8, R11 @RW3+d16	MOV R9, R11 @RW3+d16	MOV R10, R11 @RW3+d16	MOV R11, R11 @RW3+d16	MOV R12, R11 @RW3+d16	MOV R13, R11 @RW3+d16	MOV R14, R11 @RW3+d16	MOV R15, R11 @RW3+d16
+C	MOV R0, R12 @RW0+RW7	MOV R1, R12 @RW0+RW7	MOV R2, R12 @RW0+RW7	MOV R3, R12 @RW0+RW7	MOV R4, R12 @RW0+RW7	MOV R5, R12 @RW0+RW7	MOV R6, R12 @RW0+RW7	MOV R7, R12 @RW0+RW7	MOV R8, R12 @RW0+RW7	MOV R9, R12 @RW0+RW7	MOV R10, R12 @RW0+RW7	MOV R11, R12 @RW0+RW7	MOV R12, R12 @RW0+RW7	MOV R13, R12 @RW0+RW7	MOV R14, R12 @RW0+RW7	MOV R15, R12 @RW0+RW7
+D	MOV R0, R13 @RW1+RW7	MOV R1, R13 @RW1+RW7	MOV R2, R13 @RW1+RW7	MOV R3, R13 @RW1+RW7	MOV R4, R13 @RW1+RW7	MOV R5, R13 @RW1+RW7	MOV R6, R13 @RW1+RW7	MOV R7, R13 @RW1+RW7	MOV R8, R13 @RW1+RW7	MOV R9, R13 @RW1+RW7	MOV R10, R13 @RW1+RW7	MOV R11, R13 @RW1+RW7	MOV R12, R13 @RW1+RW7	MOV R13, R13 @RW1+RW7	MOV R14, R13 @RW1+RW7	MOV R15, R13 @RW1+RW7
+E	MOV R0															

[illegible]

Table C.9-18 MOV ea, Ri Instruction (First Byte = 7CH)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOV R0, R0, @RW0+d8, R0	MOV R0, R1, @RW0+d8, R1	MOV R0, R2, @RW0+d8, R2	MOV R0, R3, @RW0+d8, R3	MOV R0, R4, @RW0+d8, R4	MOV R0, R5, @RW0+d8, R5	MOV R0, R6, @RW0+d8, R6	MOV R0, R7, @RW0+d8, R7	MOV R0, R8, @RW0+d8, R8	MOV R0, R9, @RW0+d8, R9	MOV R0, R10, @RW0+d8, R10	MOV R0, R11, @RW0+d8, R11	MOV R0, R12, @RW0+d8, R12	MOV R0, R13, @RW0+d8, R13	MOV R0, R14, @RW0+d8, R14	MOV R0, R15, @RW0+d8, R15
+1	MOV R1, R0, @RW1+d8, R0	MOV R1, R1, @RW1+d8, R1	MOV R1, R2, @RW1+d8, R2	MOV R1, R3, @RW1+d8, R3	MOV R1, R4, @RW1+d8, R4	MOV R1, R5, @RW1+d8, R5	MOV R1, R6, @RW1+d8, R6	MOV R1, R7, @RW1+d8, R7	MOV R1, R8, @RW1+d8, R8	MOV R1, R9, @RW1+d8, R9	MOV R1, R10, @RW1+d8, R10	MOV R1, R11, @RW1+d8, R11	MOV R1, R12, @RW1+d8, R12	MOV R1, R13, @RW1+d8, R13	MOV R1, R14, @RW1+d8, R14	MOV R1, R15, @RW1+d8, R15
+2	MOV R2, R0, @RW2+d8, R0	MOV R2, R1, @RW2+d8, R1	MOV R2, R2, @RW2+d8, R2	MOV R2, R3, @RW2+d8, R3	MOV R2, R4, @RW2+d8, R4	MOV R2, R5, @RW2+d8, R5	MOV R2, R6, @RW2+d8, R6	MOV R2, R7, @RW2+d8, R7	MOV R2, R8, @RW2+d8, R8	MOV R2, R9, @RW2+d8, R9	MOV R2, R10, @RW2+d8, R10	MOV R2, R11, @RW2+d8, R11	MOV R2, R12, @RW2+d8, R12	MOV R2, R13, @RW2+d8, R13	MOV R2, R14, @RW2+d8, R14	MOV R2, R15, @RW2+d8, R15
+3	MOV R3, R0, @RW3+d8, R0	MOV R3, R1, @RW3+d8, R1	MOV R3, R2, @RW3+d8, R2	MOV R3, R3, @RW3+d8, R3	MOV R3, R4, @RW3+d8, R4	MOV R3, R5, @RW3+d8, R5	MOV R3, R6, @RW3+d8, R6	MOV R3, R7, @RW3+d8, R7	MOV R3, R8, @RW3+d8, R8	MOV R3, R9, @RW3+d8, R9	MOV R3, R10, @RW3+d8, R10	MOV R3, R11, @RW3+d8, R11	MOV R3, R12, @RW3+d8, R12	MOV R3, R13, @RW3+d8, R13	MOV R3, R14, @RW3+d8, R14	MOV R3, R15, @RW3+d8, R15
+4	MOV R4, R0, @RW4+d8, R0	MOV R4, R1, @RW4+d8, R1	MOV R4, R2, @RW4+d8, R2	MOV R4, R3, @RW4+d8, R3	MOV R4, R4, @RW4+d8, R4	MOV R4, R5, @RW4+d8, R5	MOV R4, R6, @RW4+d8, R6	MOV R4, R7, @RW4+d8, R7	MOV R4, R8, @RW4+d8, R8	MOV R4, R9, @RW4+d8, R9	MOV R4, R10, @RW4+d8, R10	MOV R4, R11, @RW4+d8, R11	MOV R4, R12, @RW4+d8, R12	MOV R4, R13, @RW4+d8, R13	MOV R4, R14, @RW4+d8, R14	MOV R4, R15, @RW4+d8, R15
+5	MOV R5, R0, @RW5+d8, R0	MOV R5, R1, @RW5+d8, R1	MOV R5, R2, @RW5+d8, R2	MOV R5, R3, @RW5+d8, R3	MOV R5, R4, @RW5+d8, R4	MOV R5, R5, @RW5+d8, R5	MOV R5, R6, @RW5+d8, R6	MOV R5, R7, @RW5+d8, R7	MOV R5, R8, @RW5+d8, R8	MOV R5, R9, @RW5+d8, R9	MOV R5, R10, @RW5+d8, R10	MOV R5, R11, @RW5+d8, R11	MOV R5, R12, @RW5+d8, R12	MOV R5, R13, @RW5+d8, R13	MOV R5, R14, @RW5+d8, R14	MOV R5, R15, @RW5+d8, R15
+6	MOV R6, R0, @RW6+d8, R0	MOV R6, R1, @RW6+d8, R1	MOV R6, R2, @RW6+d8, R2	MOV R6, R3, @RW6+d8, R3	MOV R6, R4, @RW6+d8, R4	MOV R6, R5, @RW6+d8, R5	MOV R6, R6, @RW6+d8, R6	MOV R6, R7, @RW6+d8, R7	MOV R6, R8, @RW6+d8, R8	MOV R6, R9, @RW6+d8, R9	MOV R6, R10, @RW6+d8, R10	MOV R6, R11, @RW6+d8, R11	MOV R6, R12, @RW6+d8, R12	MOV R6, R13, @RW6+d8, R13	MOV R6, R14, @RW6+d8, R14	MOV R6, R15, @RW6+d8, R15
+7	MOV R7, R0, @RW7+d8, R0	MOV R7, R1, @RW7+d8, R1	MOV R7, R2, @RW7+d8, R2	MOV R7, R3, @RW7+d8, R3	MOV R7, R4, @RW7+d8, R4	MOV R7, R5, @RW7+d8, R5	MOV R7, R6, @RW7+d8, R6	MOV R7, R7, @RW7+d8, R7	MOV R7, R8, @RW7+d8, R8	MOV R7, R9, @RW7+d8, R9	MOV R7, R10, @RW7+d8, R10	MOV R7, R11, @RW7+d8, R11	MOV R7, R12, @RW7+d8, R12	MOV R7, R13, @RW7+d8, R13	MOV R7, R14, @RW7+d8, R14	MOV R7, R15, @RW7+d8, R15
+8	MOV @RW0, R0, @RW0+d16, R0	MOV @RW0, R1, @RW0+d16, R1	MOV @RW0, R2, @RW0+d16, R2	MOV @RW0, R3, @RW0+d16, R3	MOV @RW0, R4, @RW0+d16, R4	MOV @RW0, R5, @RW0+d16, R5	MOV @RW0, R6, @RW0+d16, R6	MOV @RW0, R7, @RW0+d16, R7	MOV @RW0, R8, @RW0+d16, R8	MOV @RW0, R9, @RW0+d16, R9	MOV @RW0, R10, @RW0+d16, R10	MOV @RW0, R11, @RW0+d16, R11	MOV @RW0, R12, @RW0+d16, R12	MOV @RW0, R13, @RW0+d16, R13	MOV @RW0, R14, @RW0+d16, R14	MOV @RW0, R15, @RW0+d16, R15
+9	MOV @RW1, R0, @RW1+d16, R0	MOV @RW1, R1, @RW1+d16, R1	MOV @RW1, R2, @RW1+d16, R2	MOV @RW1, R3, @RW1+d16, R3	MOV @RW1, R4, @RW1+d16, R4	MOV @RW1, R5, @RW1+d16, R5	MOV @RW1, R6, @RW1+d16, R6	MOV @RW1, R7, @RW1+d16, R7	MOV @RW1, R8, @RW1+d16, R8	MOV @RW1, R9, @RW1+d16, R9	MOV @RW1, R10, @RW1+d16, R10	MOV @RW1, R11, @RW1+d16, R11	MOV @RW1, R12, @RW1+d16, R12	MOV @RW1, R13, @RW1+d16, R13	MOV @RW1, R14, @RW1+d16, R14	MOV @RW1, R15, @RW1+d16, R15
+A	MOV @RW2, R0, @RW2+d16, R0	MOV @RW2, R1, @RW2+d16, R1	MOV @RW2, R2, @RW2+d16, R2	MOV @RW2, R3, @RW2+d16, R3	MOV @RW2, R4, @RW2+d16, R4	MOV @RW2, R5, @RW2+d16, R5	MOV @RW2, R6, @RW2+d16, R6	MOV @RW2, R7, @RW2+d16, R7	MOV @RW2, R8, @RW2+d16, R8	MOV @RW2, R9, @RW2+d16, R9	MOV @RW2, R10, @RW2+d16, R10	MOV @RW2, R11, @RW2+d16, R11	MOV @RW2, R12, @RW2+d16, R12	MOV @RW2, R13, @RW2+d16, R13	MOV @RW2, R14, @RW2+d16, R14	MOV @RW2, R15, @RW2+d16, R15
+B	MOV @RW3, R0, @RW3+d16, R0	MOV @RW3, R1, @RW3+d16, R1	MOV @RW3, R2, @RW3+d16, R2	MOV @RW3, R3, @RW3+d16, R3	MOV @RW3, R4, @RW3+d16, R4	MOV @RW3, R5, @RW3+d16, R5	MOV @RW3, R6, @RW3+d16, R6	MOV @RW3, R7, @RW3+d16, R7	MOV @RW3, R8, @RW3+d16, R8	MOV @RW3, R9, @RW3+d16, R9	MOV @RW3, R10, @RW3+d16, R10	MOV @RW3, R11, @RW3+d16, R11	MOV @RW3, R12, @RW3+d16, R12	MOV @RW3, R13, @RW3+d16, R13	MOV @RW3, R14, @RW3+d16, R14	MOV @RW3, R15, @RW3+d16, R15
+C	MOV @RW0+, R0, @RW0+RW7, R0	MOV @RW0+, R1, @RW0+RW7, R1	MOV @RW0+, R2, @RW0+RW7, R2	MOV @RW0+, R3, @RW0+RW7, R3	MOV @RW0+, R4, @RW0+RW7, R4	MOV @RW0+, R5, @RW0+RW7, R5	MOV @RW0+, R6, @RW0+RW7, R6	MOV @RW0+, R7, @RW0+RW7, R7	MOV @RW0+, R8, @RW0+RW7, R8	MOV @RW0+, R9, @RW0+RW7, R9	MOV @RW0+, R10, @RW0+RW7, R10	MOV @RW0+, R11, @RW0+RW7, R11	MOV @RW0+, R12, @RW0+RW7, R12	MOV @RW0+, R13, @RW0+RW7, R13	MOV @RW0+, R14, @RW0+RW7, R14	MOV @RW0+, R15, @RW0+RW7, R15
+D	MOV @RW1+, R0, @RW1+RW7, R0	MOV @RW1+, R1, @RW1+RW7, R1	MOV @RW1+, R2, @RW1+RW7, R2	MOV @RW1+, R3, @RW1+RW7, R3	MOV @RW1+, R4, @RW1+RW7, R4	MOV @RW1+, R5, @RW1+RW7, R5	MOV @RW1+, R6, @RW1+RW7, R6	MOV @RW1+, R7, @RW1+RW7, R7	MOV @RW1+, R8, @RW1+RW7, R8	MOV @RW1+, R9, @RW1+RW7, R9	MOV @RW1+, R10, @RW1+RW7, R10	MOV @RW1+, R11, @RW1+RW7, R11	MOV @RW1+, R12, @RW1+RW7, R12	MOV @RW1+, R13, @RW1+RW7, R13	MOV @RW1+, R14, @RW1+RW7, R14	MOV @RW1+, R15, @RW1+RW7, R15
+E	MOV @RW2+, R0, @RW2+PC+d16, R0	MOV @RW2+, R1, @RW2+PC+d16, R1	MOV @RW2+, R2, @RW2+PC+d16, R2	MOV @RW2+, R3, @RW2+PC+d16, R3	MOV @RW2+, R4, @RW2+PC+d16, R4	MOV @RW2+, R5, @RW2+PC+d16, R5	MOV @RW2+, R6, @RW2+PC+d16, R6	MOV @RW2+, R7, @RW2+PC+d16, R7	MOV @RW2+, R8, @RW2+PC+d16, R8	MOV @RW2+, R9, @RW2+PC+d16, R9	MOV @RW2+, R10, @RW2+PC+d16, R10	MOV @RW2+, R11, @RW2+PC+d16, R11	MOV @RW2+, R12, @RW2+PC+d16, R12	MOV @RW2+, R13, @RW2+PC+d16, R13	MOV @RW2+, R14, @RW2+PC+d16, R14	MOV @RW2+, R15, @RW2+PC+d16, R15
+F	MOV @RW3+, R0, @RW3+addr16, R0	MOV @RW3+, R1, @RW3+addr16, R1	MOV @RW3+, R2, @RW3+addr16, R2	MOV @RW3+, R3, @RW3+addr16, R3	MOV @RW3+, R4, @RW3+addr16, R4	MOV @RW3+, R5, @RW3+addr16, R5	MOV @RW3+, R6, @RW3+addr16, R6	MOV @RW3+, R7, @RW3+addr16, R7	MOV @RW3+, R8, @RW3+addr16, R8	MOV @RW3+, R9, @RW3+addr16, R9	MOV @RW3+, R10, @RW3+addr16, R10	MOV @RW3+, R11, @RW3+addr16, R11	MOV @RW3+, R12, @RW3+addr16, R12	MOV @RW3+, R13, @RW3+addr16, R13	MOV @RW3+, R14, @RW3+addr16, R14	MOV @RW3+, R15, @RW3+addr16, R15

Table C.9-19 MOVW ea, Rwi Instruction (First Byte = 7D<sub>H</sub>)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVW RW0, RW0, @RW0+8, RW0	MOVW RW0, RW1, @RW0+8, RW1	MOVW RW0, RW2, @RW0+8, RW2	MOVW RW0, RW3, @RW0+8, RW3	MOVW RW0, RW4, @RW0+8, RW4	MOVW RW0, RW5, @RW0+8, RW5	MOVW RW0, RW6, @RW0+8, RW6	MOVW RW0, RW7, @RW0+8, RW7	MOVW RW0, RW8, @RW0+8, RW8	MOVW RW0, RW9, @RW0+8, RW9	MOVW RW0, RW10, @RW0+8, RW10	MOVW RW0, RW11, @RW0+8, RW11	MOVW RW0, RW12, @RW0+8, RW12	MOVW RW0, RW13, @RW0+8, RW13	MOVW RW0, RW14, @RW0+8, RW14	MOVW RW0, RW15, @RW0+8, RW15
+1	MOVW RW1, RW0, @RW1+8, RW0	MOVW RW1, RW1, @RW1+8, RW1	MOVW RW1, RW2, @RW1+8, RW2	MOVW RW1, RW3, @RW1+8, RW3	MOVW RW1, RW4, @RW1+8, RW4	MOVW RW1, RW5, @RW1+8, RW5	MOVW RW1, RW6, @RW1+8, RW6	MOVW RW1, RW7, @RW1+8, RW7	MOVW RW1, RW8, @RW1+8, RW8	MOVW RW1, RW9, @RW1+8, RW9	MOVW RW1, RW10, @RW1+8, RW10	MOVW RW1, RW11, @RW1+8, RW11	MOVW RW1, RW12, @RW1+8, RW12	MOVW RW1, RW13, @RW1+8, RW13	MOVW RW1, RW14, @RW1+8, RW14	MOVW RW1, RW15, @RW1+8, RW15
+2	MOVW RW2, RW0, @RW2+8, RW0	MOVW RW2, RW1, @RW2+8, RW1	MOVW RW2, RW2, @RW2+8, RW2	MOVW RW2, RW3, @RW2+8, RW3	MOVW RW2, RW4, @RW2+8, RW4	MOVW RW2, RW5, @RW2+8, RW5	MOVW RW2, RW6, @RW2+8, RW6	MOVW RW2, RW7, @RW2+8, RW7	MOVW RW2, RW8, @RW2+8, RW8	MOVW RW2, RW9, @RW2+8, RW9	MOVW RW2, RW10, @RW2+8, RW10	MOVW RW2, RW11, @RW2+8, RW11	MOVW RW2, RW12, @RW2+8, RW12	MOVW RW2, RW13, @RW2+8, RW13	MOVW RW2, RW14, @RW2+8, RW14	MOVW RW2, RW15, @RW2+8, RW15
+3	MOVW RW3, RW0, @RW3+8, RW0	MOVW RW3, RW1, @RW3+8, RW1	MOVW RW3, RW2, @RW3+8, RW2	MOVW RW3, RW3, @RW3+8, RW3	MOVW RW3, RW4, @RW3+8, RW4	MOVW RW3, RW5, @RW3+8, RW5	MOVW RW3, RW6, @RW3+8, RW6	MOVW RW3, RW7, @RW3+8, RW7	MOVW RW3, RW8, @RW3+8, RW8	MOVW RW3, RW9, @RW3+8, RW9	MOVW RW3, RW10, @RW3+8, RW10	MOVW RW3, RW11, @RW3+8, RW11	MOVW RW3, RW12, @RW3+8, RW12	MOVW RW3, RW13, @RW3+8, RW13	MOVW RW3, RW14, @RW3+8, RW14	MOVW RW3, RW15, @RW3+8, RW15
+4	MOVW RW4, RW0, @RW4+8, RW0	MOVW RW4, RW1, @RW4+8, RW1	MOVW RW4, RW2, @RW4+8, RW2	MOVW RW4, RW3, @RW4+8, RW3	MOVW RW4, RW4, @RW4+8, RW4	MOVW RW4, RW5, @RW4+8, RW5	MOVW RW4, RW6, @RW4+8, RW6	MOVW RW4, RW7, @RW4+8, RW7	MOVW RW4, RW8, @RW4+8, RW8	MOVW RW4, RW9, @RW4+8, RW9	MOVW RW4, RW10, @RW4+8, RW10	MOVW RW4, RW11, @RW4+8, RW11	MOVW RW4, RW12, @RW4+8, RW12	MOVW RW4, RW13, @RW4+8, RW13	MOVW RW4, RW14, @RW4+8, RW14	MOVW RW4, RW15, @RW4+8, RW15
+5	MOVW RW5, RW0, @RW5+8, RW0	MOVW RW5, RW1, @RW5+8, RW1	MOVW RW5, RW2, @RW5+8, RW2	MOVW RW5, RW3, @RW5+8, RW3	MOVW RW5, RW4, @RW5+8, RW4	MOVW RW5, RW5, @RW5+8, RW5	MOVW RW5, RW6, @RW5+8, RW6	MOVW RW5, RW7, @RW5+8, RW7	MOVW RW5, RW8, @RW5+8, RW8	MOVW RW5, RW9, @RW5+8, RW9	MOVW RW5, RW10, @RW5+8, RW10	MOVW RW5, RW11, @RW5+8, RW11	MOVW RW5, RW12, @RW5+8, RW12	MOVW RW5, RW13, @RW5+8, RW13	MOVW RW5, RW14, @RW5+8, RW14	MOVW RW5, RW15, @RW5+8, RW15
+6	MOVW RW6, RW0, @RW6+8, RW0	MOVW RW6, RW1, @RW6+8, RW1	MOVW RW6, RW2, @RW6+8, RW2	MOVW RW6, RW3, @RW6+8, RW3	MOVW RW6, RW4, @RW6+8, RW4	MOVW RW6, RW5, @RW6+8, RW5	MOVW RW6, RW6, @RW6+8, RW6	MOVW RW6, RW7, @RW6+8, RW7	MOVW RW6, RW8, @RW6+8, RW8	MOVW RW6, RW9, @RW6+8, RW9	MOVW RW6, RW10, @RW6+8, RW10	MOVW RW6, RW11, @RW6+8, RW11	MOVW RW6, RW12, @RW6+8, RW12	MOVW RW6, RW13, @RW6+8, RW13	MOVW RW6, RW14, @RW6+8, RW14	MOVW RW6, RW15, @RW6+8, RW15
+7	MOVW RW7, RW0, @RW7+8, RW0	MOVW RW7, RW1, @RW7+8, RW1	MOVW RW7, RW2, @RW7+8, RW2	MOVW RW7, RW3, @RW7+8, RW3	MOVW RW7, RW4, @RW7+8, RW4	MOVW RW7, RW5, @RW7+8, RW5	MOVW RW7, RW6, @RW7+8, RW6	MOVW RW7, RW7, @RW7+8, RW7	MOVW RW7, RW8, @RW7+8, RW8	MOVW RW7, RW9, @RW7+8, RW9	MOVW RW7, RW10, @RW7+8, RW10	MOVW RW7, RW11, @RW7+8, RW11	MOVW RW7, RW12, @RW7+8, RW12	MOVW RW7, RW13, @RW7+8, RW13	MOVW RW7, RW14, @RW7+8, RW14	MOVW RW7, RW15, @RW7+8, RW15
+8	MOVW @RW0, RW0, +d16, RW0	MOVW @RW0, RW1, +d16, RW1	MOVW @RW0, RW2, +d16, RW2	MOVW @RW0, RW3, +d16, RW3	MOVW @RW0, RW4, +d16, RW4	MOVW @RW0, RW5, +d16, RW5	MOVW @RW0, RW6, +d16, RW6	MOVW @RW0, RW7, +d16, RW7	MOVW @RW0, RW8, +d16, RW8	MOVW @RW0, RW9, +d16, RW9	MOVW @RW0, RW10, +d16, RW10	MOVW @RW0, RW11, +d16, RW11	MOVW @RW0, RW12, +d16, RW12	MOVW @RW0, RW13, +d16, RW13	MOVW @RW0, RW14, +d16, RW14	MOVW @RW0, RW15, +d16, RW15
+9	MOVW @RW1, RW1, +d16, RW1	MOVW @RW1, RW1, +d16, RW1	MOVW @RW1, RW2, +d16, RW2	MOVW @RW1, RW3, +d16, RW3	MOVW @RW1, RW4, +d16, RW4	MOVW @RW1, RW5, +d16, RW5	MOVW @RW1, RW6, +d16, RW6	MOVW @RW1, RW7, +d16, RW7	MOVW @RW1, RW8, +d16, RW8	MOVW @RW1, RW9, +d16, RW9	MOVW @RW1, RW10, +d16, RW10	MOVW @RW1, RW11, +d16, RW11	MOVW @RW1, RW12, +d16, RW12	MOVW @RW1, RW13, +d16, RW13	MOVW @RW1, RW14, +d16, RW14	MOVW @RW1, RW15, +d16, RW15
+A	MOVW @RW2, RW2, +d16, RW2	MOVW @RW2, RW2, +d16, RW2	MOVW @RW2, RW3, +d16, RW3	MOVW @RW2, RW4, +d16, RW4	MOVW @RW2, RW5, +d16, RW5	MOVW @RW2, RW6, +d16, RW6	MOVW @RW2, RW7, +d16, RW7	MOVW @RW2, RW8, +d16, RW8	MOVW @RW2, RW9, +d16, RW9	MOVW @RW2, RW10, +d16, RW10	MOVW @RW2, RW11, +d16, RW11	MOVW @RW2, RW12, +d16, RW12	MOVW @RW2, RW13, +d16, RW13	MOVW @RW2, RW14, +d16, RW14	MOVW @RW2, RW15, +d16, RW15	MOVW @RW2, RW16, +d16, RW16
+B	MOVW @RW3, RW3, +d16, RW3	MOVW @RW3, RW3, +d16, RW3	MOVW @RW3, RW4, +d16, RW4	MOVW @RW3, RW5, +d16, RW5	MOVW @RW3, RW6, +d16, RW6	MOVW @RW3, RW7, +d16, RW7	MOVW @RW3, RW8, +d16, RW8	MOVW @RW3, RW9, +d16, RW9	MOVW @RW3, RW10, +d16, RW10	MOVW @RW3, RW11, +d16, RW11	MOVW @RW3, RW12, +d16, RW12	MOVW @RW3, RW13, +d16, RW13	MOVW @RW3, RW14, +d16, RW14	MOVW @RW3, RW15, +d16, RW15	MOVW @RW3, RW16, +d16, RW16	MOVW @RW3, RW17, +d16, RW17
+C	MOVW @RW0+, RW0, +RW7, RW0	MOVW @RW0+, RW1, +RW7, RW1	MOVW @RW0+, RW2, +RW7, RW2	MOVW @RW0+, RW3, +RW7, RW3	MOVW @RW0+, RW4, +RW7, RW4	MOVW @RW0+, RW5, +RW7, RW5	MOVW @RW0+, RW6, +RW7, RW6	MOVW @RW0+, RW7, +RW7, RW7	MOVW @RW0+, RW8, +RW7, RW8	MOVW @RW0+, RW9, +RW7, RW9	MOVW @RW0+, RW10, +RW7, RW10	MOVW @RW0+, RW11, +RW7, RW11	MOVW @RW0+, RW12, +RW7, RW12	MOVW @RW0+, RW13, +RW7, RW13	MOVW @RW0+, RW14, +RW7, RW14	MOVW @RW0+, RW15, +RW7, RW15
+D	MOVW @RW1+, RW1, +RW7, RW1	MOVW @RW1+, RW1, +RW7, RW1	MOVW @RW1+, RW2, +RW7, RW2	MOVW @RW1+, RW3, +RW7, RW3	MOVW @RW1+, RW4, +RW7, RW4	MOVW @RW1+, RW5, +RW7, RW5	MOVW @RW1+, RW6, +RW7, RW6	MOVW @RW1+, RW7, +RW7, RW7	MOVW @RW1+, RW8, +RW7, RW8	MOVW @RW1+, RW9, +RW7, RW9	MOVW @RW1+, RW10, +RW7, RW10	MOVW @RW1+, RW11, +RW7, RW11	MOVW @RW1+, RW12, +RW7, RW12	MOVW @RW1+, RW13, +RW7, RW13	MOVW @RW1+, RW14, +RW7, RW14	MOVW @RW1+, RW15, +RW7, RW15
+E	MOVW @RW2+, RW2, +d16, RW2	MOVW @RW2+, RW2, +d16, RW2	MOVW @RW2+, RW3, +d16, RW3	MOVW @RW2+, RW4, +d16, RW4	MOVW @RW2+, RW5, +d16, RW5	MOVW @RW2+, RW6, +d16, RW6	MOVW @RW2+, RW7, +d16, RW7	MOVW @RW2+, RW8, +d16, RW8	MOVW @RW2+, RW9, +d16, RW9	MOVW @RW2+, RW10, +d16, RW10	MOVW @RW2+, RW11, +d16, RW11	MOVW @RW2+, RW12, +d16, RW12	MOVW @RW2+, RW13, +d16, RW13	MOVW @RW2+, RW14, +d16, RW14	MOVW @RW2+, RW15, +d16, RW15	MOVW @RW2+, RW16, +d16, RW16
+F	MOVW @RW3+, RW3, +d16, RW3	MOVW @RW3+, RW3, +d16, RW3	MOVW @RW3+, RW4, +d16, RW4	MOVW @RW3+, RW5, +d16, RW5	MOVW @RW3+, RW6, +d16, RW6	MOVW @RW3+, RW7, +d16, RW7	MOVW @RW3+, RW8, +d16, RW8	MOVW @RW3+, RW9, +d16, RW9	MOVW @RW3+, RW10, +d16, RW10	MOVW @RW3+, RW11, +d16, RW11	MOVW @RW3+, RW12, +d16, RW12	MOVW @RW3+, RW13, +d16, RW13	MOVW @RW3+, RW14, +d16, RW14	MOVW @RW3+, RW15, +d16, RW15	MOVW @RW3+, RW16, +d16, RW16	MOVW @RW3+, RW17, +d16, RW17

Table C.9-20 XCH Ri, ea Instruction (First Byte = 7EH)

	00	10	20	30	40	50	60	70	80	90	A	B0	C0	D0	E0	F0
+0	XCH R0, R0, @RW0+d8	XCH R0, R1, @RW0+d8	XCH R1, R0, @RW0+d8	XCH R1, R2, @RW0+d8	XCH R2, R0, @RW0+d8	XCH R2, R3, @RW0+d8	XCH R3, R0, @RW0+d8	XCH R3, R4, @RW0+d8	XCH R4, R0, @RW0+d8	XCH R4, R5, @RW0+d8	XCH R5, R0, @RW0+d8	XCH R5, R1, @RW0+d8	XCH R6, R0, @RW0+d8	XCH R6, R7, @RW0+d8	XCH R7, R0, @RW0+d8	XCH R7, R1, @RW0+d8
+1	XCH R0, R1, @RW1+d8	XCH R1, R0, @RW1+d8	XCH R1, R2, @RW1+d8	XCH R2, R1, @RW1+d8	XCH R2, R3, @RW1+d8	XCH R3, R1, @RW1+d8	XCH R3, R2, @RW1+d8	XCH R4, R1, @RW1+d8	XCH R4, R2, @RW1+d8	XCH R5, R1, @RW1+d8	XCH R5, R2, @RW1+d8	XCH R6, R1, @RW1+d8	XCH R6, R2, @RW1+d8	XCH R7, R1, @RW1+d8	XCH R7, R2, @RW1+d8	XCH R7, R3, @RW1+d8
+2	XCH R0, R2, @RW2+d8	XCH R2, R0, @RW2+d8	XCH R2, R1, @RW2+d8	XCH R2, R3, @RW2+d8	XCH R3, R2, @RW2+d8	XCH R3, R4, @RW2+d8	XCH R4, R2, @RW2+d8	XCH R4, R3, @RW2+d8	XCH R5, R2, @RW2+d8	XCH R5, R3, @RW2+d8	XCH R6, R2, @RW2+d8	XCH R6, R3, @RW2+d8	XCH R7, R2, @RW2+d8	XCH R7, R3, @RW2+d8	XCH R7, R4, @RW2+d8	XCH R7, R5, @RW2+d8
+3	XCH R0, R3, @RW3+d8	XCH R3, R0, @RW3+d8	XCH R3, R1, @RW3+d8	XCH R3, R2, @RW3+d8	XCH R4, R3, @RW3+d8	XCH R4, R4, @RW3+d8	XCH R5, R3, @RW3+d8	XCH R5, R4, @RW3+d8	XCH R6, R3, @RW3+d8	XCH R6, R4, @RW3+d8	XCH R7, R3, @RW3+d8	XCH R7, R4, @RW3+d8	XCH R7, R5, @RW3+d8	XCH R7, R6, @RW3+d8	XCH R7, R7, @RW3+d8	XCH R7, R8, @RW3+d8
+4	XCH R0, R4, @RW4+d8	XCH R4, R0, @RW4+d8	XCH R4, R1, @RW4+d8	XCH R4, R2, @RW4+d8	XCH R5, R4, @RW4+d8	XCH R5, R5, @RW4+d8	XCH R6, R4, @RW4+d8	XCH R6, R5, @RW4+d8	XCH R7, R4, @RW4+d8	XCH R7, R5, @RW4+d8	XCH R8, R4, @RW4+d8	XCH R8, R5, @RW4+d8	XCH R9, R4, @RW4+d8	XCH R9, R5, @RW4+d8	XCH R9, R6, @RW4+d8	XCH R9, R7, @RW4+d8
+5	XCH R0, R5, @RW5+d8	XCH R5, R0, @RW5+d8	XCH R5, R1, @RW5+d8	XCH R5, R2, @RW5+d8	XCH R6, R5, @RW5+d8	XCH R6, R6, @RW5+d8	XCH R7, R5, @RW5+d8	XCH R7, R6, @RW5+d8	XCH R8, R5, @RW5+d8	XCH R8, R6, @RW5+d8	XCH R9, R5, @RW5+d8	XCH R9, R6, @RW5+d8	XCH R10, R5, @RW5+d8	XCH R10, R6, @RW5+d8	XCH R10, R7, @RW5+d8	XCH R10, R8, @RW5+d8
+6	XCH R0, R6, @RW6+d8	XCH R6, R0, @RW6+d8	XCH R6, R1, @RW6+d8	XCH R6, R2, @RW6+d8	XCH R7, R6, @RW6+d8	XCH R7, R7, @RW6+d8	XCH R8, R6, @RW6+d8	XCH R8, R7, @RW6+d8	XCH R9, R6, @RW6+d8	XCH R9, R7, @RW6+d8	XCH R10, R6, @RW6+d8	XCH R10, R7, @RW6+d8	XCH R11, R6, @RW6+d8	XCH R11, R7, @RW6+d8	XCH R11, R8, @RW6+d8	XCH R11, R9, @RW6+d8
+7	XCH R0, R7, @RW7+d8	XCH R7, R0, @RW7+d8	XCH R7, R1, @RW7+d8	XCH R7, R2, @RW7+d8	XCH R8, R7, @RW7+d8	XCH R8, R8, @RW7+d8	XCH R9, R7, @RW7+d8	XCH R9, R8, @RW7+d8	XCH R10, R7, @RW7+d8	XCH R10, R8, @RW7+d8	XCH R11, R7, @RW7+d8	XCH R11, R8, @RW7+d8	XCH R12, R7, @RW7+d8	XCH R12, R8, @RW7+d8	XCH R12, R9, @RW7+d8	XCH R12, R10, @RW7+d8
+8	XCH R0, @RW0, @RW0+d16	XCH R0, R1, @RW0+d16	XCH R1, R0, @RW0+d16	XCH R1, R2, @RW0+d16	XCH R2, R0, @RW0+d16	XCH R2, R3, @RW0+d16	XCH R3, R0, @RW0+d16	XCH R3, R4, @RW0+d16	XCH R4, R0, @RW0+d16	XCH R4, R5, @RW0+d16	XCH R5, R0, @RW0+d16	XCH R5, R1, @RW0+d16	XCH R6, R0, @RW0+d16	XCH R6, R7, @RW0+d16	XCH R7, R0, @RW0+d16	XCH R7, R1, @RW0+d16
+9	XCH R0, @RW1, @RW1+d16	XCH R1, R0, @RW1+d16	XCH R1, R2, @RW1+d16	XCH R2, R1, @RW1+d16	XCH R2, R3, @RW1+d16	XCH R3, R1, @RW1+d16	XCH R3, R2, @RW1+d16	XCH R4, R1, @RW1+d16	XCH R4, R2, @RW1+d16	XCH R5, R1, @RW1+d16	XCH R5, R2, @RW1+d16	XCH R6, R1, @RW1+d16	XCH R6, R2, @RW1+d16	XCH R7, R1, @RW1+d16	XCH R7, R2, @RW1+d16	XCH R7, R3, @RW1+d16
+A	XCH R0, @RW2, W2+d16, A	XCH R2, R0, @RW2, W2+d16, A	XCH R2, R1, @RW2, W2+d16, A	XCH R2, R2, @RW2, W2+d16, A	XCH R3, R2, @RW2, W2+d16, A	XCH R3, R3, @RW2, W2+d16, A	XCH R4, R2, @RW2, W2+d16, A	XCH R4, R3, @RW2, W2+d16, A	XCH R5, R2, @RW2, W2+d16, A	XCH R5, R3, @RW2, W2+d16, A	XCH R6, R2, @RW2, W2+d16, A	XCH R6, R3, @RW2, W2+d16, A	XCH R7, R2, @RW2, W2+d16, A	XCH R7, R3, @RW2, W2+d16, A	XCH R7, R4, @RW2, W2+d16, A	XCH R7, R5, @RW2, W2+d16, A
+B	XCH R0, @RW3, @RW3+d16	XCH R3, R0, @RW3+d16	XCH R3, R1, @RW3+d16	XCH R3, R2, @RW3+d16	XCH R4, R3, @RW3+d16	XCH R4, R4, @RW3+d16	XCH R5, R3, @RW3+d16	XCH R5, R4, @RW3+d16	XCH R6, R3, @RW3+d16	XCH R6, R4, @RW3+d16	XCH R7, R3, @RW3+d16	XCH R7, R4, @RW3+d16	XCH R8, R3, @RW3+d16	XCH R8, R4, @RW3+d16	XCH R9, R3, @RW3+d16	XCH R9, R4, @RW3+d16
+C	XCH R0, @RW0+, @RW0+RW7	XCH R7, R0, @RW0+, @RW0+RW7	XCH R7, R1, @RW0+, @RW0+RW7	XCH R7, R2, @RW0+, @RW0+RW7	XCH R8, R7, @RW0+, @RW0+RW7	XCH R8, R8, @RW0+, @RW0+RW7	XCH R9, R7, @RW0+, @RW0+RW7	XCH R9, R8, @RW0+, @RW0+RW7	XCH R10, R7, @RW0+, @RW0+RW7	XCH R10, R8, @RW0+, @RW0+RW7	XCH R11, R7, @RW0+, @RW0+RW7	XCH R11, R8, @RW0+, @RW0+RW7	XCH R12, R7, @RW0+, @RW0+RW7	XCH R12, R8, @RW0+, @RW0+RW7	XCH R12, R9, @RW0+, @RW0+RW7	XCH R12, R10, @RW0+, @RW0+RW7
+D	XCH R0, @RW1+, @RW1+RW7	XCH R7, R1, @RW1+, @RW1+RW7	XCH R7, R2, @RW1+, @RW1+RW7	XCH R8, R7, @RW1+, @RW1+RW7	XCH R8, R8, @RW1+, @RW1+RW7	XCH R9, R7, @RW1+, @RW1+RW7	XCH R9, R8, @RW1+, @RW1+RW7	XCH R10, R7, @RW1+, @RW1+RW7	XCH R10, R8, @RW1+, @RW1+RW7	XCH R11, R7, @RW1+, @RW1+RW7	XCH R11, R8, @RW1+, @RW1+RW7	XCH R12, R7, @RW1+, @RW1+RW7	XCH R12, R8, @RW1+, @RW1+RW7	XCH R12, R9, @RW1+, @RW1+RW7	XCH R12, R10, @RW1+, @RW1+RW7	XCH R12, R11, @RW1+, @RW1+RW7
+E	XCH R0, @RW2+, @PC+d16	XCH R2, R0, @RW2+, @PC+d16	XCH R2, R1, @RW2+, @PC+d16	XCH R2, R2, @RW2+, @PC+d16	XCH R3, R2, @RW2+, @PC+d16	XCH R3, R3, @RW2+, @PC+d16	XCH R4, R2, @RW2+, @PC+d16	XCH R4, R3, @RW2+, @PC+d16	XCH R5, R2, @RW2+, @PC+d16	XCH R5, R3, @RW2+, @PC+d16	XCH R6, R2, @RW2+, @PC+d16	XCH R6, R3, @RW2+, @PC+d16	XCH R7, R2, @RW2+, @PC+d16	XCH R7, R3, @RW2+, @PC+d16	XCH R7, R4, @RW2+, @PC+d16	XCH R7, R5, @RW2+, @PC+d16
+F	XCH R0, @RW3+, R0, addr16	XCH R3, R0, @RW3+, R0, addr16	XCH R3, R1, @RW3+, R0, addr16	XCH R3, R2, @RW3+, R0, addr16	XCH R4, R3, @RW3+, R0, addr16	XCH R4, R4, @RW3+, R0, addr16	XCH R5, R3, @RW3+, R0, addr16	XCH R5, R4, @RW3+, R0, addr16	XCH R6, R3, @RW3+, R0, addr16	XCH R6, R4, @RW3+, R0, addr16	XCH R7, R3, @RW3+, R0, addr16	XCH R7, R4, @RW3+, R0, addr16	XCH R8, R3, @RW3+, R0, addr16	XCH R8, R4, @RW3+, R0, addr16	XCH R9, R3, @RW3+, R0, addr16	XCH R9, R4, @RW3+, R0, addr16



**Table C.9-21 XCHW RWi, ea Instruction (First Byte = 7FH)**

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	XCHW RW0, RW0, @RW0+d8	XCHW RW1, RW0, @RW0+d8	XCHW RW1, RW0, @RW0+d8	XCHW RW1, RW0, @RW0+d8	XCHW RW2, RW0, @RW0+d8	XCHW RW3, RW0, @RW0+d8	XCHW RW3, RW0, @RW0+d8	XCHW RW3, RW0, @RW0+d8	XCHW RW4, RW0, @RW0+d8	XCHW RW4, RW0, @RW0+d8	XCHW RW5, RW0, @RW0+d8	XCHW RW5, RW0, @RW0+d8	XCHW RW6, RW0, @RW0+d8	XCHW RW6, RW0, @RW0+d8	XCHW RW7, RW0, @RW0+d8	XCHW RW7, RW0, @RW0+d8
+1	XCHW RW0, RW1, @RW1+d8	XCHW RW1, RW1, @RW1+d8	XCHW RW1, RW1, @RW1+d8	XCHW RW1, RW1, @RW1+d8	XCHW RW2, RW1, @RW1+d8	XCHW RW3, RW1, @RW1+d8	XCHW RW3, RW1, @RW1+d8	XCHW RW3, RW1, @RW1+d8	XCHW RW4, RW1, @RW1+d8	XCHW RW4, RW1, @RW1+d8	XCHW RW5, RW1, @RW1+d8	XCHW RW5, RW1, @RW1+d8	XCHW RW6, RW1, @RW1+d8	XCHW RW6, RW1, @RW1+d8	XCHW RW7, RW1, @RW1+d8	XCHW RW7, RW1, @RW1+d8
+2	XCHW RW0, RW2, @RW2+d8	XCHW RW1, RW2, @RW2+d8	XCHW RW1, RW2, @RW2+d8	XCHW RW1, RW2, @RW2+d8	XCHW RW2, RW2, @RW2+d8	XCHW RW3, RW2, @RW2+d8	XCHW RW3, RW2, @RW2+d8	XCHW RW3, RW2, @RW2+d8	XCHW RW4, RW2, @RW2+d8	XCHW RW4, RW2, @RW2+d8	XCHW RW5, RW2, @RW2+d8	XCHW RW5, RW2, @RW2+d8	XCHW RW6, RW2, @RW2+d8	XCHW RW6, RW2, @RW2+d8	XCHW RW7, RW2, @RW2+d8	XCHW RW7, RW2, @RW2+d8
+3	XCHW RW0, RW3, @RW3+d8	XCHW RW1, RW3, @RW3+d8	XCHW RW1, RW3, @RW3+d8	XCHW RW1, RW3, @RW3+d8	XCHW RW2, RW3, @RW3+d8	XCHW RW3, RW3, @RW3+d8	XCHW RW3, RW3, @RW3+d8	XCHW RW3, RW3, @RW3+d8	XCHW RW4, RW3, @RW3+d8	XCHW RW4, RW3, @RW3+d8	XCHW RW5, RW3, @RW3+d8	XCHW RW5, RW3, @RW3+d8	XCHW RW6, RW3, @RW3+d8	XCHW RW6, RW3, @RW3+d8	XCHW RW7, RW3, @RW3+d8	XCHW RW7, RW3, @RW3+d8
+4	XCHW RW0, RW4, @RW4+d8	XCHW RW1, RW4, @RW4+d8	XCHW RW1, RW4, @RW4+d8	XCHW RW1, RW4, @RW4+d8	XCHW RW2, RW4, @RW4+d8	XCHW RW3, RW4, @RW4+d8	XCHW RW3, RW4, @RW4+d8	XCHW RW3, RW4, @RW4+d8	XCHW RW4, RW4, @RW4+d8	XCHW RW4, RW4, @RW4+d8	XCHW RW5, RW4, @RW4+d8	XCHW RW5, RW4, @RW4+d8	XCHW RW6, RW4, @RW4+d8	XCHW RW6, RW4, @RW4+d8	XCHW RW7, RW4, @RW4+d8	XCHW RW7, RW4, @RW4+d8
+5	XCHW RW0, RW5, @RW5+d8	XCHW RW1, RW5, @RW5+d8	XCHW RW1, RW5, @RW5+d8	XCHW RW1, RW5, @RW5+d8	XCHW RW2, RW5, @RW5+d8	XCHW RW3, RW5, @RW5+d8	XCHW RW3, RW5, @RW5+d8	XCHW RW3, RW5, @RW5+d8	XCHW RW4, RW5, @RW5+d8	XCHW RW4, RW5, @RW5+d8	XCHW RW5, RW5, @RW5+d8	XCHW RW5, RW5, @RW5+d8	XCHW RW6, RW5, @RW5+d8	XCHW RW6, RW5, @RW5+d8	XCHW RW7, RW5, @RW5+d8	XCHW RW7, RW5, @RW5+d8
+6	XCHW RW0, RW6, @RW6+d8	XCHW RW1, RW6, @RW6+d8	XCHW RW1, RW6, @RW6+d8	XCHW RW1, RW6, @RW6+d8	XCHW RW2, RW6, @RW6+d8	XCHW RW3, RW6, @RW6+d8	XCHW RW3, RW6, @RW6+d8	XCHW RW3, RW6, @RW6+d8	XCHW RW4, RW6, @RW6+d8	XCHW RW4, RW6, @RW6+d8	XCHW RW5, RW6, @RW6+d8	XCHW RW5, RW6, @RW6+d8	XCHW RW6, RW6, @RW6+d8	XCHW RW6, RW6, @RW6+d8	XCHW RW7, RW6, @RW6+d8	XCHW RW7, RW6, @RW6+d8
+7	XCHW RW0, RW7, @RW7+d8	XCHW RW1, RW7, @RW7+d8	XCHW RW1, RW7, @RW7+d8	XCHW RW1, RW7, @RW7+d8	XCHW RW2, RW7, @RW7+d8	XCHW RW3, RW7, @RW7+d8	XCHW RW3, RW7, @RW7+d8	XCHW RW3, RW7, @RW7+d8	XCHW RW4, RW7, @RW7+d8	XCHW RW4, RW7, @RW7+d8	XCHW RW5, RW7, @RW7+d8	XCHW RW5, RW7, @RW7+d8	XCHW RW6, RW7, @RW7+d8	XCHW RW6, RW7, @RW7+d8	XCHW RW7, RW7, @RW7+d8	XCHW RW7, RW7, @RW7+d8
+8	XCHW RW0, @RW0	XCHW RW1, @RW0+d16	XCHW RW1, @RW0	XCHW RW1, @RW0+d16	XCHW RW2, @RW0	XCHW RW3, @RW0	XCHW RW3, @RW0	XCHW RW3, @RW0	XCHW RW4, @RW0	XCHW RW4, @RW0+d16	XCHW RW5, @RW0	XCHW RW5, @RW0+d16	XCHW RW6, @RW0	XCHW RW6, @RW0+d16	XCHW RW7, @RW0	XCHW RW7, @RW0+d16
+9	XCHW RW0, @RW1	XCHW RW1, @RW1+d16	XCHW RW1, @RW1	XCHW RW1, @RW1+d16	XCHW RW2, @RW1	XCHW RW3, @RW1	XCHW RW3, @RW1	XCHW RW3, @RW1	XCHW RW4, @RW1	XCHW RW4, @RW1+d16	XCHW RW5, @RW1	XCHW RW5, @RW1+d16	XCHW RW6, @RW1	XCHW RW6, @RW1+d16	XCHW RW7, @RW1	XCHW RW7, @RW1+d16
+A	XCHW RW0, @RW2	XCHW RW1, @RW2+d16	XCHW RW1, @RW2	XCHW RW1, @RW2+d16	XCHW RW2, @RW2	XCHW RW3, @RW2	XCHW RW3, @RW2	XCHW RW3, @RW2	XCHW RW4, @RW2	XCHW RW4, @RW2+d16	XCHW RW5, @RW2	XCHW RW5, @RW2+d16	XCHW RW6, @RW2	XCHW RW6, @RW2+d16	XCHW RW7, @RW2	XCHW RW7, @RW2+d16
+B	XCHW RW0, @RW3	XCHW RW1, @RW3+d16	XCHW RW1, @RW3	XCHW RW1, @RW3+d16	XCHW RW2, @RW3	XCHW RW3, @RW3	XCHW RW3, @RW3	XCHW RW3, @RW3	XCHW RW4, @RW3	XCHW RW4, @RW3+d16	XCHW RW5, @RW3	XCHW RW5, @RW3+d16	XCHW RW6, @RW3	XCHW RW6, @RW3+d16	XCHW RW7, @RW3	XCHW RW7, @RW3+d16
+C	XCHW RW0, @RW0+	XCHW RW1, @RW0+RW7	XCHW RW1, @RW0+	XCHW RW1, @RW0+RW7	XCHW RW2, @RW0+	XCHW RW3, @RW0	XCHW RW3, @RW0	XCHW RW3, @RW0	XCHW RW4, @RW0	XCHW RW4, @RW0+RW7	XCHW RW5, @RW0+	XCHW RW5, @RW0+RW7	XCHW RW6, @RW0+	XCHW RW6, @RW0+RW7	XCHW RW7, @RW0+	XCHW RW7, @RW0+RW7
+D	XCHW RW0, @RW1+	XCHW RW1, @RW1+RW7	XCHW RW1, @RW1+	XCHW RW1, @RW1+RW7	XCHW RW2, @RW1+	XCHW RW3, @RW1	XCHW RW3, @RW1	XCHW RW3, @RW1	XCHW RW4, @RW1	XCHW RW4, @RW1+RW7	XCHW RW5, @RW1+	XCHW RW5, @RW1+RW7	XCHW RW6, @RW1+	XCHW RW6, @RW1+RW7	XCHW RW7, @RW1+	XCHW RW7, @RW1+RW7
+E	XCHW RW0, @RW2+	XCHW RW1, @RW2+d16	XCHW RW1, @RW2+	XCHW RW1, @RW2+d16	XCHW RW2, @RW2+	XCHW RW3, @RW2	XCHW RW3, @RW2	XCHW RW3, @RW2	XCHW RW4, @RW2	XCHW RW4, @RW2+d16	XCHW RW5, @RW2+	XCHW RW5, @RW2+d16	XCHW RW6, @RW2+	XCHW RW6, @RW2+d16	XCHW RW7, @RW2+	XCHW RW7, @RW2+d16
+F	XCHW RW0, @RW3+	XCHW RW1, @RW3+d16	XCHW RW1, @RW3+	XCHW RW1, @RW3+d16	XCHW RW2, @RW3+	XCHW RW3, @RW3	XCHW RW3, @RW3	XCHW RW3, @RW3	XCHW RW4, @RW3+	XCHW RW4, @RW3+d16	XCHW RW5, @RW3+	XCHW RW5, @RW3+d16	XCHW RW6, @RW3+	XCHW RW6, @RW3+d16	XCHW RW7, @RW3+	XCHW RW7, @RW3+d16

**MB90820B Series****INDEX**

---

The index follows on the next page.  
This is listed in alphabetic order.

---

# Index

## Numerics

1024K Bit Flash Memory	
Characteristics of the 512K/1024K Bit	
Flash Memory .....	558
Overview of the 512K/1024K Bit	
Flash Memory .....	558
16-bit Free-run Timer	
16-bit Free-run Timer (x 1) .....	344
16-bit Free-run Timer Interrupts .....	387
16-bit Free-run Timer Interrupts and EI <sup>2</sup> OS .....	387
16-bit Free-run Timer Registers .....	353
Block Diagram of 16-bit Free-run Timer .....	347
Usage Notes on the 16-bit Free-run Timer .....	416
16-bit Input Capture	
16-bit Input Capture (x 4) .....	344
16-bit Input Capture Interrupts .....	389
16-bit Input Capture Interrupts and EI <sup>2</sup> OS .....	389
16-bit Input Capture Operation .....	403
Block Diagram of 16-bit Input Capture .....	348
Usage Notes on the 16-bit Input Capture .....	416
16-bit Input Capture Input	
16-bit Input Capture Input Timing .....	404
16-bit Output Compare	
16-bit Output Compare (x 6) .....	344
16-bit Output Compare Interrupts .....	388
16-bit Output Compare Interrupts and EI <sup>2</sup> OS .....	388
16-bit Output Compare Operation .....	398
16-bit Output Compare Registers .....	354
16-bit Output Compare Timing .....	401
Block Diagram of 16-bit Output Compare .....	348
Usage Notes on the 16-bit Output Compare .....	416
16-bit PPG Timer	
16-bit PPG Timer (x 1) .....	345
16-bit PPG Timer (x 3) .....	322
16-bit PPG Timer Interrupts .....	336
16-bit PPG Timer Interrupts and EI <sup>2</sup> OS .....	337
16-bit PPG Timer Pins .....	324
16-bit PPG Timer Registers .....	326
Block Diagram of 16-bit PPG Timer .....	323
Block Diagram of the 16-bit PPG Timer Pins .....	324
EI <sup>2</sup> OS Function of the 16-bit PPG Timer .....	337
Usage Notes on the 16-bit PPG Timer .....	342
16-Bit Reload Registers	
16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) .....	274
16-bit Reload Timer	
16-bit Reload Timer Settings .....	276
Baud Rates Determined Using the Internal Timer (16-bit Reload Timer) .....	525
Block Diagram of 16-bit Reload Timer .....	265
EI <sup>2</sup> OS Function of 16-bit Reload Timer .....	275
Interrupts Generated by 16-bit Reload Timer .....	275
Interrupts of 16-bit Reload Timer and EI <sup>2</sup> OS .....	275
List of Registers for 16-bit Reload Timer .....	268
Notes on Using the 16-bit Reload Timer .....	286
Operation Mode of 16-bit Reload Timer .....	262
Pins of 16-bit Reload Timer .....	267
16-bit Timer Control Register	
16-bit Timer Control Register (DTCR0/DTCR2) .....	381
16-bit Timer Control Register (DTCR1) .....	383
16-bit Timer Registers	
16-bit Timer Registers (TMR0/TMR1) .....	273
16-bit Timer Registers (TMRR0 to TMRR2) .....	380
24-bit Operand	
Linear Addressing by 24-bit Operand Specification .....	32
32-bit Register	
Addressing by Indirect Specification with a 32-bit Register .....	32
512K	
Characteristics of the 512K/1024K Bit	
Flash Memory .....	558
Overview of the 512K/1024K Bit	
Flash Memory .....	558
8/10-bit A/D converter	
Block diagram of 8/10-bit A/D converter .....	450
Conversion Mode of 8/10-bit A/D Converter .....	449
EI <sup>2</sup> OS of 8/10-bit A/D converter .....	468
Features of 8/10-bit A/D converter .....	448
Interrupt of 8/10-bit A/D converter and EI <sup>2</sup> OS .....	468
Pins of 8/10-bit A/D converter .....	453
Precautions for using the 8/10-bit A/D converter .....	482
Registers of the 8/10-bit A/D converter and their initial value .....	454
The A/D converted data protection function of the 8/10-bit A/D converter .....	478

# MB90820B Series

## A

### A

Accumulator (A) .....	40
A/D control status registers	
A/D control status registers high order (ADCS1) .....	455
A/D control status registers low order (ADCS0) .....	459
A/D converted data protection	
The A/D converted data protection function of the 8/10-bit A/D converter .....	478
A/D converter	
Block diagram of 8/10-bit A/D converter .....	450
Conversion Mode of 8/10-bit A/D Converter.....	449
EI <sup>2</sup> OS of 8/10-bit A/D converter .....	468
Features of 8/10-bit A/D converter .....	448
Interrupt of 8/10-bit A/D converter and EI <sup>2</sup> OS .....	468
Interrupt of A/D converter .....	468
Pins of 8/10-bit A/D converter .....	453
Precautions for using the 8/10-bit A/D converter.....	482
Registers of the 8/10-bit A/D converter and their initial value .....	454
The A/D converted data protection function of the 8/10-bit A/D converter .....	478
A/D data register	
A/D data register (ADCR0/ADCR1).....	461
A/D setting register	
A/D setting register (ADSR0/ADSR1).....	462
Access Space	
Bank Registers and Access Space.....	33
Accumulator	
Accumulator (A) .....	40
ADB	
Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57
ADCR	
A/D data register (ADCR0/ADCR1).....	461
ADCS	
A/D control status registers high order (ADCS1) .....	455
A/D control status registers low order (ADCS0) .....	459
Continuous conversion mode (ADCS: MD1,MD0= 10 <sub>B</sub> ) .....	469
Single Conversion Mode (ADCS: MD1,MD0= 00 <sub>B</sub> or 01 <sub>B</sub> ) .....	469
Stop conversion mode (ADCS: MD1,MD0= 11 <sub>B</sub> ) .....	469
Addressing	
Addressing .....	604

Addressing by Indirect Specification with a 32-bit Register .....	32
Bank Addressing and Default Space.....	34
Direct Addressing.....	606
Indirect Addressing .....	612
Linear Addressing and Bank Addressing .....	31
Linear Addressing by 24-bit Operand Specification .....	32

## ADSR

A/D setting register (ADSR0/ADSR1) .....	462
--	-----

## Analog input enable registers

Analog input enable registers.....	467
------------------------------------	-----

## Arbitrary Data

Deleting Arbitrary Data (Sector Deletion).....	577
--	-----

## Asynchronous Mode

Operation in Asynchronous Mode.....	530
-------------------------------------	-----

## B

### Bank Addressing

Bank Addressing and Default Space.....	34
Linear Addressing and Bank Addressing .....	31

### Bank Registers

Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
Bank Registers and Access Space .....	33

### Bank Select Prefixes

Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57
--	----

### BAP

Buffer address Pointer (BAP) .....	157
------------------------------------	-----

### Baud Rate

UART Baud Rate Selection.....	520
-------------------------------	-----

### Baud Rate Generator

Baud Rates Determined Using the Dedicated Baud Rate Generator .....	522
--	-----

### Baud Rates

Baud Rates Determined Using the Dedicated Baud Rate Generator .....	522
Baud Rates Determined Using the External Clock .....	527
Baud Rates Determined Using the Internal Timer (16-bit Reload Timer) .....	525

### Bidirectional Communication

Bidirectional Communication Function .....	534
--	-----

### Bit Flash Memory

Characteristics of the 512K/1024K Bit Flash Memory .....	558
Overview of the 512K/1024K Bit Flash Memory .....	558

### Block Diagram

Block Diagram of 16-bit Free-run Timer .....	347
Block Diagram of 16-bit Input Capture.....	348
Block Diagram of 16-bit Output Compare .....	348
Block Diagram of 16-bit PPG Timer .....	323

Block Diagram of 16-bit Reload Timer .....	265
Block diagram of 8/10-bit A/D converter .....	450
Block Diagram of Clock Supervisor .....	91
Block Diagram of Multi-functional Timer .....	346
Block Diagram of Multi-functional Timer Pins .....	351
Block Diagram of Port 0 Pins .....	178
Block Diagram of Port 1 Pins .....	185
Block Diagram of Port 2 Pins .....	192
Block Diagram of Port 3 Pins .....	199
Block Diagram of Port 4 Pins .....	206
Block Diagram of Port 5 Pins .....	212
Block Diagram of Port 6 Pins .....	218
Block Diagram of Port 7 Pins .....	224
Block Diagram of Port 8 Pins .....	233
Block Diagram of ROM Correction Function .....	543
Block Diagram of the 16-bit PPG Timer Pins .....	324
Block Diagram of the Clock Generation Block .....	78
Block Diagram of the D/A Converter Pins .....	486
Block Diagram of the Delayed Interrupt Generator Module .....	420
Block Diagram of the DTP/External Interrupt Circuit .....	428
Block Diagram of the DTP/External Interrupt Circuit Pins .....	431
Block Diagram of the External Reset Pin .....	68
Block Diagram of the Low-Power Consumption Control Circuit .....	105
Block Diagram of the PWC Timer .....	289
Block Diagram of the PWC Timer Pins .....	290
Block Diagram of the Time-base Timer .....	242
Block Diagram of the Watchdog Timer .....	253
Block Diagram of UART .....	496
Block Diagram of UART Pins .....	500
Block Diagram of Waveform Generator .....	349
D/A Converter Block Diagram .....	485
MB90820B Series Block Diagram .....	7
ROM Mirroring Function Selection Module Block Diagram .....	554
Buffer address Pointer	
Buffer address Pointer (BAP) .....	157
Bus Mode Setting Bits	
Bus Mode Setting Bits .....	170
<b>C</b>	
Calculating	
Calculating the Execution Cycle Count .....	621
Capture Input	
16-bit Input Capture Input Timing .....	404
CCR	
Condition Code Register (CCR)	
Configuration .....	47
CDCR	
Communication Prescaler Control Register (CDCR) .....	512
Chip	
Deleting the Data (Chip Deletion) .....	576
When the Chip/Sector Deletion Operation is Executed .....	566
When the Write Operation or Chip/Sector Deletion Operation is Executed .....	568, 570
Chip Deletion	
Deleting the Data (Chip Deletion) .....	576
CKSCR	
Configuration of the Clock Selection Register (CKSCR) .....	81
Clock	
Baud Rates Determined Using the External Clock .....	527
Clock .....	76
Clock Mode .....	103
Clock Mode Transition .....	85
Clock Supply Function .....	241
Connection of an Oscillator or an External Clock to the Microcontroller .....	88
Event Count Mode (External Clock Mode) .....	263
External Count Clock Selected .....	397
Internal Clock Mode .....	262
Machine Clock .....	85
Main Clock Mode and PLL Clock Mode .....	85
Supply of Operation Clock .....	248
Clock Generation Block	
Block Diagram of the Clock Generation Block .....	78
Clock Mode	
Clock Mode .....	103
Clock Mode Transition	
Clock Mode Transition .....	85
Clock Selection Register	
Clock Selection Registers .....	80
Configuration of the Clock Selection Register (CKSCR) .....	81
Clock Supervisor	
Block Diagram of Clock Supervisor .....	91
Clock Supervisor Control Register (CSVCR) .....	94
Clock Supervisor Register .....	93
Example Operation Flowchart for the Clock Supervisor .....	97
Example Startup Flowchart when using the Clock Supervisor .....	98
Operations of Clock Supervisor .....	96
Overview of Clock Supervisor .....	90
Precautions when using the Clock Supervisor .....	99
Clock Supply	
Clock Supply Function .....	241

# MB90820B Series

Clock Supply Map		
Clock Supply Map .....	77	
Clock Synchronous Mode		
Operation in Clock Synchronous Mode		
(Operation Mode 2) .....	532	
CMR		
Common Register Bank Prefix (CMR) .....	59	
Command Sequence Table		
Command Sequence Table .....	563	
Common Register Bank Prefix		
Common Register Bank Prefix (CMR) .....	59	
Communication		
Bidirectional Communication Function .....	534	
Master-slave Communication Function .....	536	
Communication Prescaler Control Register		
Communication Prescaler Control Register		
(CDCR) .....	512	
Compare Clear Buffer		
Compare Clear Buffer .....	393	
Compare Clear Buffer Register		
Compare Clear Buffer Register (CPCLRB) .....	357	
Compare Clear Register		
Compare Clear Register (CPCLR) .....	357	
Compare Control Register		
Compare Control Register, Lower Byte		
(OCS0/2/4) .....	369	
Compare Control Register, Upper Byte		
(OCS1/3/5) .....	366	
Compare time		
Compare time setting (CT2 to CT0) .....	466	
Condition Code Register		
Condition Code Register (CCR)		
Configuration .....	47	
Connection		
Example of Minimum Connection with Flash		
Microcomputer Programmer (When Power		
Supplied by User) .....	598	
Example of Minimum Connection with Flash		
Microcontroller Programmer (When Power		
Supplied from Writer) .....	600	
Consecutive Prefix		
Consecutive Prefix Codes .....	62	
Continuous Conversion Mode		
Operation and Usage of the Continuous Conversion		
Mode .....	472	
Setting of Continuous Conversion Mode .....	472	
Continuous conversion mode		
Continuous conversion mode		
(ADCS: MD1, MD0= 10 <sub>B</sub> ) .....	469	
Continuous Measurement Mode		
Single Measurement Mode and Continuous		
Measurement Mode .....	314	
Conversion		
Conversion Using EI <sup>2</sup> OS .....	476	
Operation and Usage of the Continuous Conversion		
Mode .....	472	
Operation and Usage of the Single Conversion		
Mode .....	471	
Operation and Usage of the Stop Conversion		
Mode .....	474	
Setting of Continuous Conversion Mode .....	472	
Setting of Single Conversion Mode .....	470	
Setting of Stop Conversion Mode .....	474	
Single Conversion Mode (ADCS: MD1, MD0= 00 <sub>B</sub> or		
01 <sub>B</sub> ) .....	469	
Stop conversion mode		
(ADCS: MD1, MD0= 11 <sub>B</sub> ) .....	469	
Continuous conversion mode		
(ADCS: MD1, MD0= 10 <sub>B</sub> ) .....	469	
Conversion Mode		
Conversion Mode of 8/10-bit A/D Converter .....	449	
Count Clock Period		
Count Clock Period and Maximum Period .....	312	
Counter Operation		
Counter Operation .....	263	
States of Counter Operation .....	277	
CPCLR		
Compare Clear Register (CPCLR) .....	357	
CPCLRB		
Compare Clear Buffer Register (CPCLRB) .....	357	
CPU		
CPU .....	26	
CPU Intermittent Operation Mode		
CPU Intermittent Operation Mode .....	103, 110	
CPU Operating Modes		
CPU Operating Modes and Current Consumption		
.....	102	
CSVCR		
Clock Supervisor Control Register (CSVCR) .....	94	
CT		
Compare time setting (CT2 to CT0) .....	466	
Current Consumption		
CPU Operating Modes and Current Consumption		
.....	102	
D		
D/A Control Register		
D/A Control Register 0 (DACR0) .....	491	
D/A Control Register 1 (DACR1) .....	490	
D/A Converter		
Block Diagram of the D/A Converter Pins .....	486	
D/A Converter Block Diagram .....	485	
D/A Converter Pins .....	486	
D/A Converter Registers .....	487	

D/A Converter Register	
D/A Converter Register 0 (DAT0).....	489
D/A Converter Register1(DAT1) .....	488
DACR	
D/A Control Register 0 (DACR0) .....	491
D/A Control Register 1 (DACR1) .....	490
DAT	
D/A Converter Register 0 (DAT0).....	489
D/A Converter Register1(DAT1) .....	488
Data Counter	
Data Counter (DCT) .....	156
DCT	
Data Counter (DCT) .....	156
Dedicated Baud Rate Generator	
Baud Rates Determined Using the Dedicated Baud Rate Generator .....	522
Dedicated Register	
Configuration of Dedicated Registers .....	38
Dedicated Registers and General-purpose Registers .....	37
Default Space	
Bank Addressing and Default Space .....	34
Delayed Interrupt Generator Module	
Block Diagram of the Delayed Interrupt Generator Module .....	420
Operation of the Delayed Interrupt Generator Module .....	422
Delayed Interrupt Generator Module Register	
Delayed Interrupt Generator Module Register (DIRR) .....	421
Delayed Interrupt Request Latch	
Usage Notes on the Delayed Interrupt Request Latch .....	423
Delete	
Detailed Explanation on the Flash Memory Write/ Delete .....	572
Deleting	
Deleting Arbitrary Data (Sector Deletion) .....	577
Deleting the Data (Chip Deletion) .....	576
Procedure for Writing/Deleting the Data to the Flash Memory .....	558
Procedure of Deleting a Sector .....	577
Deletion	
Deleting Arbitrary Data (Sector Deletion) .....	577
Deleting the Data (Chip Deletion) .....	576
Restarting the Sector Deletion .....	580
Temporarily Stopping the Sector Deletion .....	579
When the Sector Deletion Temporary Stop is Executed. ....	568
When the Sector Deletion Temporary Stop is Executed. ....	566
When the Sector Deletion Temporary Stop Operation is Executed .....	571
Deletion Operation	
When the Chip/Sector Deletion Operation is Executed .....	566
When the Sector Deletion Operation is Executed .....	571
When the Write Operation or Chip/Sector Deletion Operation is Executed. ....	568, 570
Description	
Description of Instruction Presentation Items and Symbols .....	624
Descriptor	
Configuration of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) Descriptor (ISD) .....	155
Devices	
Notes on Handling Devices .....	21
Direct Addressing	
Direct Addressing .....	606
Direct Page Register	
Direct Page Register (DPR) .....	52
DIRR	
Delayed Interrupt Generator Module Register (DIRR) .....	421
DIV	
Division Ratio Control Register (DIV0/DIV1) .....	301
Division Ratio Control Register	
Division Ratio Control Register (DIV0/DIV1) .....	301
DPR	
Direct Page Register (DPR) .....	52
DTB	
Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57
DTCR	
16-bit Timer Control Register (DTCR0/DTCR2) .....	381
16-bit Timer Control Register (DTCR1) .....	383
DTP	
DTP/External Interrupt Cause Register (EIRR) .....	433
DTP Function	
Operation of the DTP Function .....	442
DTP/External Interrupt Circuit	
Block Diagram of the DTP/External Interrupt Circuit .....	428
Block Diagram of the DTP/External Interrupt Circuit Pins .....	431
DTP/External Interrupt Circuit Pins .....	430
Interrupt of the DTP/External Interrupt Circuit and EI <sup>2</sup> OS .....	427
Operation of the DTP/External Interrupt Circuit .....	439
Setting the DTP/External Interrupt Circuit .....	438

# MB90820B Series

Usage Notes on the DTP/External Interrupt Circuit .....	444
DTP/External Interrupt Functions	
DTP/External Interrupt Functions .....	426
DTP/External Interrupt Interrupt Enable Register	
DTP/External Interrupt Interrupt Enable Register (ENIR) .....	434
DTTI	
DTTI Interrupt .....	415
DTTI pin Input Operation .....	414
DTTI Pin Noise Cancellation Function .....	415
<b>E</b>	
E <sup>2</sup> PROM	
E <sup>2</sup> PROM Memory Map .....	549
Effective Address Field	
Effective Address Field .....	605, 623
EI <sup>2</sup> OS	
16-bit Free-run Timer Interrupts and EI <sup>2</sup> OS .....	387
16-bit Input Capture Interrupts and EI <sup>2</sup> OS .....	389
16-bit Output Compare Interrupts and EI <sup>2</sup> OS .....	388
Configuration of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) Descriptor (ISD) .....	155
Conversion Using EI <sup>2</sup> OS .....	476
EI <sup>2</sup> OS Function of 16-bit Reload Timer .....	275
EI <sup>2</sup> OS Function of the 16-bit PPG Timer .....	337
EI <sup>2</sup> OS Function of the Multi-functional Timer .....	390
EI <sup>2</sup> OS Function of the PWC Timer .....	303
EI <sup>2</sup> OS of 8/10-bit A/D converter .....	468
Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	153
Extended Intelligent I/O Service (EI <sup>2</sup> OS) Status Register (ISCS) .....	157
Interrupt of 8/10-bit A/D converter and EI <sup>2</sup> OS .....	468
Interrupt of the DTP/External Interrupt Circuit and EI <sup>2</sup> OS .....	427
Interrupts of 16-bit Reload Timer and EI <sup>2</sup> OS .....	275
Operation Flow of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	159
Operation of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	154
Procedure for Using the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	160
Processing Time (One Transfer Time) of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	161
PWC Timer Interrupts and EI <sup>2</sup> OS .....	302
Time-base Timer Interrupts and EI <sup>2</sup> OS .....	246
UART EI <sup>2</sup> OS Functions .....	515
UART Interrupt and EI <sup>2</sup> OS .....	495
UART Interrupts and EI <sup>2</sup> OS .....	515
Waveform Generator Interrupts and EI <sup>2</sup> OS .....	390
ELVR	
Request Level Setting Register (ELVR) .....	436
ENIR	
DTP/External Interrupt Interrupt Enable Register (ENIR) .....	434
Error	
If a Program Error Occurs .....	550
Event Count Mode	
Event Count Mode .....	284
Event Count Mode (External Clock Mode) .....	263
Example of Connection	
Example of Connection for Serial Writing (When Power Supplied by User) .....	594
Example of Connection for Serial Writing (When Power Supplied from Writer) .....	596
Exception Processing	
Exception Processing .....	163
Execution Cycle Count	
Calculating the Execution Cycle Count .....	621
Execution Cycle Count .....	620
Extended Intelligent I/O Service	
Configuration of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) Descriptor (ISD) .....	155
Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	153
Extended Intelligent I/O Service (EI <sup>2</sup> OS) Status Register (ISCS) .....	157
Operation Flow of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	159
Operation of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	154
Procedure for Using the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	160
Processing Time (One Transfer Time) of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	161
External Clock	
Baud Rates Determined Using the External Clock .....	527
Connection of an Oscillator or an External Clock to the Microcontroller .....	88
External Clock Mode	
Event Count Mode (External Clock Mode) .....	263
External Count Clock	
External Count Clock Selected .....	397
External Interrupt	
External Interrupt Function .....	441
External Interrupt Cause Register	
DTP/External Interrupt Cause Register (EIRR) .....	433
External Reset	
Block Diagram of the External Reset Pin .....	68
<b>F</b>	
F <sup>2</sup> MC-16LX Instruction List	
F <sup>2</sup> MC-16LX Instruction List .....	627



Fetch	
Mode Data Fetch .....	70
Flag Change Suppression Prefix	
Flag Change Suppression Prefix (NCC) .....	60
Flash Memory	
Characteristics of the 512K/1024K Bit	
Flash Memory .....	558
Detailed Explanation on the Flash Memory Write/	
Delete .....	572
Overview of the 512K/1024K Bit	
Flash Memory .....	558
Procedure for Writing/Deleting the Data to the Flash	
Memory .....	558
Procedure of Writing the Data to the	
Flash Memory .....	574
Register on the Flash Memory .....	558
Flash Memory Control Status Register	
Flash Memory Control Status Register	
(FMCS) .....	560
Flash Microcomputer Programmer	
Example of Minimum Connection with Flash	
Microcomputer Programmer (When Power	
Supplied by User) .....	598
FMCS	
Flash Memory Control Status Register	
(FMCS) .....	560
FPT-80	
FPT-80P-M21/FPT-80P-M22	
Pin Assignment .....	9
FPT-80P-M06	
FPT-80P-M06 Package Dimensions .....	11
FPT-80P-M06 Pin Assignment .....	8
FPT-80P-M21	
FPT-80P-M21 Package Dimensions .....	10
FPT-80P-M22	
FPT-80P-M22 Package Dimensions .....	12
Free-run Timer	
16-bit Free-run Timer (x 1) .....	344
16-bit Free-run Timer Interrupts .....	387
16-bit Free-run Timer Interrupts and	
EI <sup>2</sup> OS .....	387
16-bit Free-run Timer Registers .....	353
Block Diagram of 16-bit Free-run Timer .....	347
Usage Notes on the 16-bit Free-run Timer .....	416
Fujitsu Standard	
Standard Configuration for Fujitsu Standard Serial	
On-board Writing .....	591
<b>G</b>	
Gate	
Gate Triggered PPG0 Output .....	406
Output Condition of RTO0 to RTO5 and	
GATE .....	405
GATE Signal	
Generating GATE Signal During Each RTx is at	
“H” Level When GTENx is Active (DTCR0/	
1/2:TMD2 to TMD0=001 <sub>B</sub> or 111 <sub>B</sub> )	
.....	406
Generating GATE Signal from Rising Edge of Each	
RTx until 16-bit Timer 0/1/2 Underflow	
When GTENx is Active (DTCR0/1/2:TMD2	
to TMD0=010 <sub>B</sub> ) .....	407
Gate Trigger	
Gate Trigger (PPG channel 0 only) .....	340
General-purpose Register	
Configuration of a General-purpose Register .....	54
Dedicated Registers and General-purpose	
Registers .....	37
General-purpose Register Area and Register Bank	
Pointer (RP) .....	49
<b>H</b>	
Hardware Interrupt	
Hardware Interrupt .....	140
Hardware Interrupt Activation .....	143
Hardware Interrupt Operation .....	144
Hardware Interrupt Processing Time .....	149
Hardware Interrupt Structure .....	141
Hardware Interrupt Suppression .....	141
Procedure for Using Hardware Interrupt .....	146
Returning from a Hardware Interrupt .....	143
Hardware Sequence Flag	
Hardware Sequence Flag .....	564
hold Suppression	
Prefix Codes and Interrupt/hold Suppression	
Instructions .....	61
<b>I</b>	
I/O	
I/O Area .....	28
I/O Circuit	
I/O Circuit Types .....	17
I/O Map	
I/O Map .....	584
I/O Pins	
I/O Pins and Pin Functions .....	13
I/O Ports	
I/O Ports Functions .....	174
Registers for I/O Ports .....	176
I/O Register Address Pointer	
I/O Register Address Pointer (IOA) .....	156
ICR	
Bit Configuration of Interrupt Control Registers	
(ICR) .....	137

# MB90820B Series

Interrupt Control Registers (ICR00 to ICR15) .....	135	Operation of Internal Clock Mode (One-shot Mode) .....	281
ICSH		Internal Peripheral Features	
Input Capture Control Status Register,Upper Byte (ICSH23) .....	372	Internal Peripheral Features .....	3
ICSL		Internal Timer	
Input Capture Control Status Register,Lower Byte (ICSL23) .....	374	Baud Rates Determined Using the Internal Timer (16-bit Reload Timer) .....	525
ILM		Interrupt	
Interrupt Level Mask Register (ILM) .....	50	DTTI Interrupt .....	415
Indirect Addressing		External Interrupt Function .....	441
Indirect Addressing .....	612	Hardware Interrupt .....	140
Indirect Specification		Hardware Interrupt Activation .....	143
Addressing by Indirect Specification with a 32-bit Register .....	32	Hardware Interrupt Operation .....	144
Initial		Hardware Interrupt Processing Time .....	149
Registers of the 8/10-bit A/D converter and their initial value .....	454	Hardware Interrupt Structure .....	141
Input Capture		INT9 Interrupt .....	550
16-bit Input Capture (x 4) .....	344	Interrupt Causes and Interrupt Vectors/Interrupt Control Registers .....	131
16-bit Input Capture Interrupts .....	389	Interrupt Mask Function .....	396
16-bit Input Capture Interrupts and EI <sup>2</sup> OS .....	389	Interrupt of 8/10-bit A/D converter and EI <sup>2</sup> OS .....	468
16-bit Input Capture Operation .....	403	Interrupt of A/D converter .....	468
Block Diagram of 16-bit Input Capture .....	348	Interrupt of the DTP/External Interrupt Circuit and EI <sup>2</sup> OS .....	427
Input Capture Registers .....	355	Interrupt Operation .....	129
Usage Notes on the 16-bit Input Capture .....	416	Interrupt Request Generation .....	311, 316
Input Capture Control Status Register		Interrupt Types and Functions .....	128
Input Capture Control Status Register,Lower Byte (ICSL23) .....	374	Prefix Codes and Interrupt/hold Suppression Instructions .....	61
Input Capture Control Status Register,Lower Byte (PICSL01) .....	378	Procedure for Using Hardware Interrupt .....	146
Input Capture Control Status Register,Upper Byte (ICSH23) .....	372	Processing for Interrupt Operation .....	145
Input Capture Register		Reception Interrupt Request Generation and Flag Set Timing .....	516
Input Capture Register (IPCP0 to IPCP3) .....	371	Returning from a Hardware Interrupt .....	143
Instruction		Returning from a Software Interrupt .....	151
Description of Instruction Presentation Items and Symbols .....	624	Software Interrupt Activation .....	151
F <sup>2</sup> MC-16LX Instruction List .....	627	Software Interrupt Operation .....	152
Instruction Types .....	603	Stack Operations at the Start of Interrupt Processing .....	164
Prefix Codes and Interrupt/hold Suppression Instructions .....	61	Stack Operations on Return from Interrupt Processing .....	164
Structure of Instruction Map .....	641	Transmission Interrupt Request Generation and Flag Set Timing .....	518
Instruction Presentation Items and Symbols		UART Interrupt and EI <sup>2</sup> OS .....	495
Description of Instruction Presentation Items and Symbols .....	624	Interrupt Cause	
INT9		Interrupt Causes and Interrupt Vectors/Interrupt Control Registers .....	131
INT9 Interrupt .....	550	Interrupt Control Register	
Internal Clock Mode		Interrupt Causes and Interrupt Vectors/Interrupt Control Registers .....	131
Internal Clock Mode .....	262	Interrupt Control Register Functions .....	134, 138
Operation in Internal Clock Mode (Reload Mode) .....	278	Interrupt Control Registers .....	133
		Interrupt Control Registers (ICR00 to ICR15) .....	135

Bit Configuration of Interrupt Control Registers (ICR).....	137
Interrupt Level Mask Register	
Interrupt Level Mask Register (ILM).....	50
Interrupt Mask Function	
Interrupt Mask Function .....	396
Interrupt Request	
Interrupt Request Generation .....	311, 316
Interrupt Suppression	
Hardware Interrupt Suppression .....	141
Interrupt Vector	
Interrupt Causes and Interrupt Vectors/Interrupt Control Registers .....	131
Interrupt Vectors .....	130
Interrupts	
16-bit Free-run Timer Interrupts .....	387
16-bit Free-run Timer Interrupts and EI <sup>2</sup> OS .....	387
16-bit Input Capture Interrupts .....	389
16-bit Input Capture Interrupts and EI <sup>2</sup> OS .....	389
16-bit Output Compare Interrupts.....	388
16-bit Output Compare Interrupts and EI <sup>2</sup> OS .....	388
16-bit PPG Timer Interrupts .....	336
16-bit PPG Timer Interrupts and EI <sup>2</sup> OS .....	337
Interrupts Generated by 16-bit Reload Timer.....	275
Interrupts of 16-bit Reload Timer and EI <sup>2</sup> OS.....	275
Multiple Interrupts .....	147
PPG Interrupts.....	341
PWC Timer Interrupts.....	302
PWC Timer Interrupts and EI <sup>2</sup> OS .....	302
ROM Correction Interrupts .....	542
Time-base Timer Interrupts .....	246
Time-base Timer Interrupts and EI <sup>2</sup> OS .....	246
Timer Interrupts .....	395
UART Interrupts .....	514
UART Interrupts and EI <sup>2</sup> OS .....	515
Waveform Generator Interrupts.....	389
Waveform Generator Interrupts and EI <sup>2</sup> OS .....	390
Interval Timer	
Interval Timer Function.....	240
Operation of the Interval Timer Function (Time-base Timer) .....	247
Inverted Polarity	
Making Non-overlap Signals by Using PPG in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=111 <sub>B</sub> ) .....	413
Making Non-overlap Signals by using RT1/3/5 in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=100 <sub>B</sub> ) .....	411
IOA	
I/O Register Address Pointer (IOA).....	156
IPCP	
Input Capture Register (IPCP0 to IPCP3).....	371

ISCS	
Extended Intelligent I/O Service (EI <sup>2</sup> OS) Status Register (ISCS) .....	157
ISD	
Configuration of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) Descriptor (ISD).....	155
<b>L</b>	
Latch	
Usage Notes on the Delayed Interrupt Request Latch.....	423
Linear Addressing	
Linear Addressing and Bank Addressing .....	31
Linear Addressing by 24-bit Operand Specification .....	32
Low-Power Consumption	
Block Diagram of the Low-Power Consumption Control Circuit.....	105
Low-Power Consumption Control Circuit	
Block Diagram of the Low-Power Consumption Control Circuit.....	105
Low-Power Consumption Mode	
Low-Power Consumption Mode Operating States .....	120
Low-Power Consumption Mode Control Register	
Access to the Low-Power Consumption Mode Control Register .....	109
Low-Power Consumption Mode Control Register (LPMCR) .....	107
LPMCR	
Low-Power Consumption Mode Control Register (LPMCR) .....	107
<b>M</b>	
Machine Clock	
Machine Clock.....	85
Main Clock	
Main Clock Mode and PLL Clock Mode .....	85
Mask	
Interrupt Mask Function.....	396
Master-slave Communication	
Master-slave Communication Function .....	536
Maximum Period	
Count Clock Period and Maximum Period .....	312
MB902820B Series	
MB902820B Series Features .....	2
MB902820B Series Features	
MB902820B Series Features .....	2
MB90820B Series	
MB90820B Series Block Diagram .....	7
MB90820B Series Product Line-up.....	5

# MB90820B Series

MB90820B Series Product	
MB90820B Series Product Line-up .....	5
MD	
Continuous conversion mode	
(ADCS: MD1,MD0= 10 <sub>B</sub> ) .....	469
Mode Pins (MD2 to MD0) .....	169
Single Conversion Mode (ADCS: MD1,MD0= 00 <sub>B</sub> or 01 <sub>B</sub> ) .....	469
Stop conversion mode	
(ADCS: MD1,MD0= 11 <sub>B</sub> ) .....	469
Measurement	
Flowchart of Pulse Width Measurement	
Operation .....	318
Measurement Mode and Measurement	
Operation .....	316
Measurement Result Data .....	314
Pulse Width Measurement Function .....	305
Pulse Width/Period Measurement Range .....	315
Single Measurement Mode and Continuous	
Measurement Mode .....	314
Starting and Stopping Timer and Pulse Width	
Measurement .....	309
Measurement Mode	
Measurement Mode and Measurement	
Operation .....	316
Measurement Operation	
Measurement Mode and Measurement	
Operation .....	316
Memory	
Storage of Multibyte Data on Memory .....	35
Memory Map	
E <sup>2</sup> PROM Memory Map .....	549
Memory Maps .....	29
Memory Space	
Memory Space .....	27
Microcontroller	
Connection of an Oscillator or an External Clock to the	
Microcontroller .....	88
Minimum Connection	
Example of Minimum Connection with Flash	
Microcomputer Programmer (When Power	
Supplied by User) .....	598
Minimum Input Pulse Width	
Minimum Input Pulse Width .....	315
Mode	
Clock Mode .....	103
Clock Mode Transition .....	85
Continuous conversion mode	
(ADCS: MD1,MD0= 10 <sub>B</sub> ) .....	469
Main Clock Mode and PLL Clock Mode .....	85
Mode Setting .....	168
Operation and Usage of the Continuous Conversion	
Mode .....	472
Operation and Usage of the Single Conversion	
Mode .....	471
Operation and Usage of the Stop Conversion	
Mode .....	474
Setting of Continuous Conversion Mode .....	472
Setting of Single Conversion Mode .....	470
Setting of Stop Conversion Mode .....	474
Single Conversion Mode (ADCS: MD1,MD0= 00 <sub>B</sub> or 01 <sub>B</sub> ) .....	469
Stop conversion mode	
(ADCS: MD1,MD0= 11 <sub>B</sub> ) .....	469
Mode Data	
Mode Data .....	170
Relationship Between Mode Pins and	
Mode Data .....	171
Status of Pins After Mode Data is Read .....	73
Mode Data Fetch	
Mode Data Fetch .....	70
Mode Pins	
Mode Pins .....	69
Mode Pins (MD2 to MD0) .....	169
Relationship Between Mode Pins and	
Mode Data .....	171
Multibyte Data	
Multibyte Data Access .....	36
Storage of Multibyte Data in a Stack .....	36
Storage of Multibyte Data on Memory .....	35
Multibyte Operand	
Storage of Multibyte Operand .....	35
Multi-functional Timer	
Block Diagram of Multi-functional Timer .....	346
Block Diagram of Multi-functional Timer Pins	
.....	351
EI <sup>2</sup> OS Function of the Multi-functional Timer	
.....	390
Multi-functional Timer Pins .....	350
Operation of Multi-functional Timer .....	391
Multiple Interrupts	
Multiple Interrupts .....	147
Multiplier Rate	
Selection of a PLL Clock Multiplier Rate .....	85
<b>N</b>	
NCC	
Flag Change Suppression Prefix (NCC) .....	60
Noise Cancellation Function	
DTTI Pin Noise Cancellation Function .....	415
Non-overlap Signals	
Making Non-overlap Signals by Using PPG in	
Inverted Polarity (DTCR0/1/2:TMD2 to	
TMD0=111 <sub>B</sub> ) .....	413

Making Non-overlap Signals by Using PPG in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=111 <sub>B</sub> ) .....	412	Operation Mode of 16-bit Reload Timer .....	262
Making Non-overlap Signals by using RT1/3/5 in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=100 <sub>B</sub> ) .....	411	Operation Mode Selection .....	307
Making Non-overlap Signals by Using RT1/3/5 in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=100 <sub>B</sub> ) .....	410	Reload Operation Mode .....	311
Normal Polarity		Oscillation Stabilization Wait	
Making Non-overlap Signals by Using PPG in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=111 <sub>B</sub> ) .....	412	Oscillation Stabilization Wait and Reset State .....	67
Making Non-overlap Signals by Using RT1/3/5 in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=100 <sub>B</sub> ) .....	410	Oscillation Stabilization Wait Interval .....	87, 124
		Reset Causes and Oscillation Stabilization Wait Intervals .....	66
		Oscillation Stabilization Time Timer	
		Oscillation Stabilization Time	
		Timer Function .....	248
		Oscillator	
		Connection of an Oscillator or an External Clock to the Microcontroller .....	88
		Output Compare	
		16-bit Output Compare (x 6) .....	344
		16-bit Output Compare Interrupts .....	388
		16-bit Output Compare Interrupts and EI <sup>2</sup> OS .....	388
		16-bit Output Compare Operation .....	398
		16-bit Output Compare Registers .....	354
		16-bit Output Compare Timing .....	401
		Block Diagram of 16-bit Output Compare .....	348
		Usage Notes on the 16-bit Output Compare .....	416
		Output Compare Buffer Registers	
		Output Compare Buffer Registers (OCCPB0 to OCCPB5) .....	364
		Output Compare Registers	
		Output Compare Registers (OCCP0 to OCCP5) .....	365
<b>O</b>		<b>P</b>	
OCCP		Package Dimensions	
Output Compare Registers (OCCP0 to OCCP5) .....	365	FPT-80P-M06 Package Dimensions .....	11
OCCPB		FPT-80P-M21 Package Dimensions .....	10
Output Compare Buffer Registers (OCCPB0 to OCCPB5) .....	364	FPT-80P-M22 Package Dimensions .....	12
OCS		PACSR	
Compare Control Register, Lower Byte (OCS0/2/4) .....	369	Program Address Detection Control Status Register (PACSR) .....	546
Compare Control Register, Upper Byte (OCS1/3/5) .....	366	PADR	
One-shot Mode		Program Address Detection Register 0/1 (PADR0/PADR1) .....	545
Operation of Internal Clock Mode (One-shot Mode) .....	281	PC	
One-shot Operation Mode		Program Counter (PC) .....	51
One-shot Operation Mode .....	311	PCB	
Operand		Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
Linear Addressing by 24-bit Operand Specification .....	32	Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57
Operating Modes		PCKCR	
CPU Operating Modes and Current Consumption .....	102	Configuration of the PLL Clock Control Register (PCKCR) .....	83
Operating Modes .....	168	PCNTH	
Operating States		PPG Control Status Register, Upper Byte (PCNTH0 to PCNTH2) .....	331
Low-Power Consumption Mode			
Operating States .....	120		
Operating Status			
Operating Status During Standby Mode .....	111		
Operation Clock			
Supply of Operation Clock .....	248		
Operation Mode			
CPU Intermittent Operation Mode .....	103, 110		
One-shot Operation Mode .....	311		
Operation in Clock Synchronous Mode (Operation Mode 2) .....	532		

# MB90820B Series

PCNTL	
PPG Control Status Register, Lower Byte (PCNTL1 to PCNTL3) .....	334
PWM Mode (PCNTL: MDSE=0) .....	338
Single-shot Mode (PCNTL: MDSE=1) .....	339
PDCR	
PPG Down Counter Register (PDCR0 to PDCR2) .....	328
Period	
Calculating Pulse Width/Period .....	315
Pulse Width/Period Measurement Range .....	315
PICSH	
PPG Output Control/Input Capture Control Status Register, Upper Byte (PICSH01) .....	376
PICSL	
Input Capture Control Status Register, Lower Byte (PICSL01) .....	378
Pin Assignment	
FPT-80P-M06 Pin Assignment .....	8
FPT-80P-M21/FPT-80P-M22 Pin Assignment .....	9
Pin Functions	
I/O Pins and Pin Functions .....	13
PLL Clock	
Main Clock Mode and PLL Clock Mode .....	85
Selection of a PLL Clock Multiplier Rate .....	85
PLL Clock Control Register	
Configuration of the PLL Clock Control Register (PCKCR) .....	83
Port 0	
Block Diagram of Port 0 Pins .....	178
Functions of Port 0 Registers .....	180
Operation of Port 0 .....	182
Port 0 Configuration .....	177
Port 0 Pins .....	177
Port 0 Registers .....	179
Port 1	
Block Diagram of Port 1 Pins .....	185
Functions of Port 1 Registers .....	187
Operation of Port 1 .....	189
Port 1 Configuration .....	184
Port 1 Pins .....	184
Port 1 Registers .....	186
Port 2	
Block Diagram of Port 2 Pins .....	192
Functions of Port 2 Registers .....	194
Operation of Port 2 .....	196
Port 2 Configuration .....	191
Port 2 Pins .....	191
Port 2 Registers .....	193
Port 3	
Block Diagram of Port 3 Pins .....	199
Functions of Port 3 Registers .....	201
Operation of Port 3 .....	203
Port 3 Configuration .....	198
Port 3 pins .....	198
Port 3 Registers .....	200
Port 4	
Block Diagram of Port 4 Pins .....	206
Functions of Port 4 registers .....	208
Operation of Port 4 .....	209
Port 4 Configuration .....	205
Port 4 Pins .....	205
Port 4 Registers .....	207
Port 5	
Functions of Port 5 registers .....	214
Operation of Port 5 .....	215
Port 5 Configuration .....	211
Port 5 Pins .....	211
Port 5 Registers .....	213
Port 6	
Block Diagram of Port 6 Pins .....	218
Functions of Port 6 Registers .....	219
Operation of Port 6 .....	221
Port 6 Configuration .....	217
Port 6 Pins .....	217
Port 6 Registers .....	218
Port 7	
Block Diagram of Port 7 Pins .....	224
Functions of Port 7 Registers .....	228
Operation of Port 7 .....	230
Port 7 Configuration .....	223
Port 7 Pins .....	223
Port 7 Registers .....	227
Port 8	
Block Diagram of Port 8 Pins .....	233
Functions of Port 8 registers .....	235
Operation of Port 8 .....	236
Port 8 Configuration .....	232
Port 8 Pins .....	232
Port 8 Registers .....	234
PPG	
Gate Trigger (PPG channel 0 only) .....	340
Gate Triggered PPG0 Output .....	406
Making Non-overlap Signals by Using PPG in Inverted Polarity (DTCR0/1/2:TMD2 to TMD0=111 <sub>B</sub> ) .....	413
Making Non-overlap Signals by Using PPG in Normal Polarity (DTCR0/1/2:TMD2 to TMD0=111 <sub>B</sub> ) .....	412
PPG Interrupts .....	341
PPG0 Output Control .....	406
PPG0 Output Pulse from Rising Edge of RT to 16-bit Timer Underflow (DTCR0/1/2:TMD2 to TMD0=010 <sub>B</sub> ) .....	408
PPG Control Status Register	
PPG Control Status Register, Lower Byte (PCNTL1 to PCNTL3) .....	334

PPG Control Status Register,Upper Byte (PCNTH0 to PCNTH2) .....	331	Pulse Width/Period Measurement Range .....	315
PPG Down Counter Register		Pulse Width	
PPG Down Counter Register (PDCR0 to PDCR2) .....	328	Calculating Pulse Width/Period .....	315
PPG Output Control/Input Capture Control Status Register		Flowchart of Pulse Width Measurement Operation .....	318
PPG Output Control/Input Capture Control Status Register,Upper Byte (PICSH01) .....	376	Minimum Input Pulse Width .....	315
PPG Timer		Pulse Width Measurement Function .....	305
16-bit PPG Timer (x 1) .....	345	Pulse Width/Period Measurement Range .....	315
16-bit PPG Timer (x 3) .....	322	Starting and Stopping Timer and Pulse Width Measurement .....	309
16-bit PPG Timer Interrupts .....	336	Pulse Width Measurement	
16-bit PPG Timer Interrupts and EI <sup>2</sup> OS .....	337	Flowchart of Pulse Width Measurement Operation .....	318
16-bit PPG Timer Pins .....	324	Pulse Width Measurement Function .....	305
16-bit PPG Timer Registers .....	326	Starting and Stopping Timer and Pulse Width Measurement .....	309
Block Diagram of 16-bit PPG Timer .....	323	PWC	
Block Diagram of the 16-bit PPG Timer Pins .....	324	PWC Data Buffer Register (PWC0/PWC1) .....	299
EI <sup>2</sup> OS Function of the 16-bit PPG Timer .....	337	PWC Control Status Register	
Usage Notes on the 16-bit PPG Timer .....	342	PWC Control Status Register,Lower Byte (PWCSL0/PWCSH01) .....	297
Prefix		PWC Control Status Register,Upper Byte (PWCSH0/PWCSH01) .....	294
Common Register Bank Prefix (CMR) .....	59	PWC Data Buffer Register	
Consecutive Prefix Codes .....	62	PWC Data Buffer Register (PWC0/PWC1) .....	299
Flag Change Suppression Prefix (NCC) .....	60	PWC Timer	
Prefix Codes .....	56	Block Diagram of the PWC Timer .....	289
Prefix Codes and Interrupt/hold Suppression Instructions .....	61	Block Diagram of the PWC Timer Pins .....	290
Prefixes		EI <sup>2</sup> OS Function of the PWC Timer .....	303
Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57	PWC Timer .....	288
Processor Status		PWC Timer Interrupts .....	302
Processor Status (PS) Configuration .....	46	PWC Timer Interrupts and EI <sup>2</sup> OS .....	302
Program Address Detection		PWC Timer Operation .....	288
Program Address Detection Registers (x 2) .....	542	PWC Timer Pins .....	290
Program Address Detection Control Status Register		PWC Timer Registers .....	293
Program Address Detection Control Status Register (PACSR) .....	546	Usage Notes on the PWC Timer .....	319
Program Address Detection Register		PWCSH	
Program Address Detection Register 0/1 (PADR0/PADR1) .....	545	PWC Control Status Register,Upper Byte (PWCSH0/PWCSH01) .....	294
Program Counter		PWCSL	
Program Counter (PC) .....	51	PWC Control Status Register,Lower Byte (PWCSL0/PWCSH01) .....	297
Program Error		PWM Mode	
If a Program Error Occurs .....	550	PWM Mode (PCNTL: MDSE=0) .....	338
protection		R	
The A/D converted data protection function of the 8/10-bit A/D converter .....	478	RAM	
PS		RAM Area .....	28
Processor Status (PS) Configuration .....	46	Read	
Pull-up Resistor		Setting the Read/Reset Status .....	573
Software Pull-up Resistor .....	122	Reception Interrupt	
Pulse		Reception Interrupt Request Generation and Flag Set Timing .....	516
Calculating Pulse Width/Period .....	315		

# MB90820B Series

Register Bank		Reset Status	
Common Register Bank Prefix (CMR) .....	59	Setting the Read/Reset Status .....	573
Register Bank .....	55	Restarting	
Register Bank Pointer		Restarting the Sector Deletion .....	580
General-purpose Register Area and Register Bank		ROM	
Pointer (RP) .....	49	Block Diagram of ROM Correction Function .....	543
Register Bank Pointer (RP) .....	49	ROM Area .....	28
Reload Mode		ROM Correction Interrupts .....	542
Operation in Internal Clock Mode (Reload Mode)		ROM Correction Function	
.....	278	Block Diagram of ROM Correction Function .....	543
Reload Operation Mode		Operation of the ROM Correction Function .....	548
Reload Operation Mode .....	311	ROM Correction Function Registers .....	544
Reload Registers		ROM Correction Interrupts	
16-Bit Reload Registers (TMRDL0/TMRDL1,		ROM Correction Interrupts .....	542
TMRDH0/TMRDH1) .....	274	ROM Mirroring Function Selection Module	
Reload Timer		ROM Mirroring Function Selection Module Block	
16-bit Reload Timer Settings .....	276	Diagram .....	554
Baud Rates Determined Using the Internal Timer		ROM Mirroring Function Selection Module	
(16-bit Reload Timer) .....	525	Register .....	554
Block Diagram of 16-bit Reload Timer .....	265	ROM Mirroring Function Selection Register	
EI <sup>2</sup> OS Function of 16-bit Reload Timer .....	275	ROM Mirroring Function Selection Register	
Interrupts Generated by 16-bit Reload Timer .....	275	(ROMM) .....	555
Interrupts of 16-bit Reload Timer and		ROMM	
EI <sup>2</sup> OS .....	275	ROM Mirroring Function Selection Register	
List of Registers for 16-bit Reload Timer .....	268	(ROMM) .....	555
Notes on Using the 16-bit Reload Timer .....	286	RP	
Operation Mode of 16-bit Reload Timer .....	262	General-purpose Register Area and Register Bank	
Pins of 16-bit Reload Timer .....	267	Pointer (RP) .....	49
Reload Value		Register Bank Pointer (RP) .....	49
Timer Value and Reload Value .....	311	RT	
Request Level Setting Register		Making Non-overlap Signals by using RT1/3/5 in	
Request Level Setting Register (ELVR) .....	436	Inverted Polarity (DTCR0/1/2:TMD2 to	
Reset		TMD0=100 <sub>B</sub> ) .....	411
Block Diagram of the External Reset Pin .....	68	Making Non-overlap Signals by Using RT1/3/5 in	
Correspondence Between Reset Cause Bits and Reset		Normal Polarity (DTCR0/1/2:TMD2 to	
Causes .....	72	TMD0=100 <sub>B</sub> ) .....	410
Notes About Reset Cause Bits .....	72	RTO	
Oscillation Stabilization Wait and Reset State .....	67	Output Condition of RTO0 to RTO5 and	
Overview of Reset Operation .....	69	GATE .....	405
Reset Cause Bits .....	71	S	
Reset Causes .....	64	Sampling time	
Reset Causes and Oscillation Stabilization Wait		Sampling time setting (ST2 to ST0) .....	465
Intervals .....	66	SCR	
Reset Sequence .....	550	Serial Control Register (SCR0/SCR1) .....	503
Status of Pins During a Reset .....	73	Sector	
Reset Cause Bits		Deleting Arbitrary Data (Sector Deletion) .....	577
Correspondence Between Reset Cause Bits and Reset		Procedure of Deleting a Sector .....	577
Causes .....	72	Restarting the Sector Deletion .....	580
Notes About Reset Cause Bits .....	72	Sector Configuration .....	559
Reset Cause Bits .....	71	Temporarily Stopping the Sector Deletion .....	579
Reset Causes		When the Chip/Sector Deletion Operation is	
Reset Causes .....	64	Executed .....	566
Reset Causes and Oscillation Stabilization Wait			
Intervals .....	66		



When the Sector Deletion Operation is Executed. ....	571
When the Sector Deletion Temporary Stop is Executed. ....	568
When the Sector Deletion Temporary Stop is Executed. ....	566
When the Sector Deletion Temporary Stop Operation is Executed. ....	571
When the Write Operation or Chip/Sector Deletion Operation is Executed. ....	568, 570
<b>Sector Deletion</b>	
Deleting Arbitrary Data (Sector Deletion) .....	577
Restarting the Sector Deletion .....	580
Temporarily Stopping the Sector Deletion .....	579
<b>Sector Deletion Operation</b>	
When the Chip/Sector Deletion Operation is Executed. ....	566
When the Sector Deletion Operation is Executed. ....	571
When the Write Operation or Chip/Sector Deletion Operation is Executed. ....	568, 570
<b>Sector Deletion Temporary Stop</b>	
When the Sector Deletion Temporary Stop is Executed. ....	568
When the Sector Deletion Temporary Stop is Executed. ....	566
<b>Sector Deletion Temporary Stop Operation</b>	
When the Sector Deletion Temporary Stop Operation is Executed. ....	571
<b>Sectors</b>	
Notes on Specifying Two or More Sectors .....	577
<b>Serial Control Register</b>	
Serial Control Register (SCR0/SCR1) .....	503
<b>Serial Input Data Register</b>	
Serial Input Data Register (SIDR0/SIDR1) .....	510
<b>Serial Mode Register</b>	
Serial Mode Register (SMR0/SMR1) .....	505
<b>Serial On-board Writing</b>	
Standard Configuration for Fujitsu Standard Serial On-board Writing .....	591
<b>Serial Output Data Register</b>	
Serial Output Data Register (SODR0/SODR1) .....	510
<b>Serial Status Register</b>	
Serial Status Register (SSR0/SSR1) .....	508
<b>Serial Writing</b>	
Example of Connection for Serial Writing (When Power Supplied by User) .....	594
Example of Connection for Serial Writing (When Power Supplied from Writer) .....	596
<b>SIDR</b>	
Serial Input Data Register (SIDR0/SIDR1) .....	510
<b>SIGCR</b>	
Waveform Control Register (SIGCR) .....	385
<b>Single Conversion Mode</b>	
Operation and Usage of the Single Conversion Mode .....	471
Setting of Single Conversion Mode .....	470
Single Conversion Mode (ADCS: MD1, MD0= 00 <sub>B</sub> or 01 <sub>B</sub> ) .....	469
<b>Single Measurement Mode</b>	
Single Measurement Mode and Continuous Measurement Mode .....	314
<b>Single-chip Mode</b>	
State of Pins in Single-chip Mode .....	122
<b>Single-shot Mode</b>	
Single-shot Mode (PCNTL: MDSE=1) .....	339
<b>Sleep Mode</b>	
Release of Sleep Mode .....	112
Switching to Sleep Mode .....	112
<b>SMR</b>	
Serial Mode Register (SMR0/SMR1) .....	505
<b>SODR</b>	
Serial Output Data Register (SODR0/SODR1) .....	510
<b>Software Interrupt</b>	
Returning from a Software Interrupt .....	151
Software Interrupt Activation .....	151
Software Interrupt Operation .....	152
<b>Software Pull-up Resistor</b>	
Software Pull-up Resistor .....	122
<b>SPB</b>	
Bank Select Prefixes (PCB,DTB,ADB,SPB) .....	57
<b>SSB</b>	
Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
<b>SSP</b>	
System Stack Pointer (SSP) .....	45
<b>SSR</b>	
Serial Status Register (SSR0/SSR1) .....	508
<b>ST</b>	
Sampling time setting (ST2 to ST0) .....	465
<b>Stack</b>	
Stack Area .....	165
Stack Operations at the Start of Interrupt Processing .....	164
Stack Operations on Return from Interrupt Processing .....	164
Stack Selection .....	43
Storage of Multibyte Data in a Stack .....	36
<b>Standby Mode</b>	
Notes on Standby Mode .....	123
Operating Status During Standby Mode .....	111
Standby Mode .....	103

# MB90820B Series

State Change Diagram		Operation of the Interval Timer Function (Time-base Timer).....	247
State Change Diagram.....	119	Operation of the Time-base Timer .....	249
Status		Time-base Timer Interrupts .....	246
Setting the Read/Reset Status.....	573	Time-base Timer Interrupts and EI <sup>2</sup> OS.....	246
Status of Pins		Time-base Timer Usage Notes.....	249
Status of Pins After Mode Data is Read .....	73	Time-base Timer Control Register	
Status Register		Time-base Timer Control Register (TBTC).....	244
Extended Intelligent I/O Service (EI <sup>2</sup> OS) Status Register (ISCS) .....	157	Time-base Timer Mode	
Stop Conversion Mode		Release of Time-base Timer Mode.....	115, 124
Operation and Usage of the Stop Conversion Mode.....	474	Switching to Time-base Timer Mode .....	115
Setting of Stop Conversion Mode.....	474	Timer Control Status Register	
Stop conversion mode (ADCS: MD1,MD0= 11 <sub>B</sub> ).....	469	Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1) .....	271
Stop Mode		Timer Control Status Register,Lower Byte (TCCSL) .....	361
Release of Stop Mode .....	117, 124	Timer Control Status Register,Upper Byte (TCCSH) .....	359
Switching to Stop Mode .....	117	Upper Bits and Bit 7 of Timer Control Status Registers (TMCSRH0/TMCSRH1) .....	269
Structure		Timer Data Register	
Structure of Instruction Map .....	641	Timer Data Register (TCDT).....	358
Synchronous Mode		Timer Function	
Operation in Clock Synchronous Mode (Operation Mode 2) .....	532	Timer Function .....	304
System Configuration		Timer Interrupts	
System Configuration .....	549	Timer Interrupts .....	395
System Stack Pointer		Timer Mode	
System Stack Pointer (SSP) .....	45	Flowchart of Timer Mode Operation .....	313
T		Timer Mode.....	392
TBTC		Timer Period	
Time-base Timer Control Register (TBTC) .....	244	Timer Period.....	312
TCCSH		Timer Registers	
Timer Control Status Register,Upper Byte (TCCSH) .....	359	16-bit Timer Registers (TMR0/TMR1).....	273
TCCSL		Timer Value	
Timer Control Status Register,Lower Byte (TCCSL).....	361	Timer Value and Reload Value .....	311
TCDT		TMCSRH	
Timer Data Register (TCDT) .....	358	Upper Bits and Bit 7 of Timer Control Status Registers (TMCSRH0/TMCSRH1) .....	269
Temporarily Stopping		TMCSRL	
Temporarily Stopping the Sector Deletion.....	579	Lower Bits of Timer Control Status Registers (TMCSRL0/TMCSRL1) .....	271
Temporary Stop		TMR	
When the Sector Deletion Temporary Stop is Executed. ....	568	16-bit Timer Registers (TMR0/TMR1).....	273
When the Sector Deletion Temporary Stop is Executed. ....	566	TMRDH	
Temporary Stop Operation		16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) .....	274
When the Sector Deletion Temporary Stop Operation is Executed.....	571	TMRDL	
Time-base Timer		16-Bit Reload Registers (TMRDL0/TMRDL1, TMRDH0/TMRDH1) .....	274
Block Diagram of the Time-base Timer .....	242	TMRR	
		16-bit Timer Registers (TMRR0 to TMRR2) .....	380

Transfer Time	
Processing Time (One Transfer Time) of the Extended Intelligent I/O Service (EI <sup>2</sup> OS) .....	161
Transmission Interrupt	
Transmission Interrupt Request Generation and Flag Set Timing .....	518
<b>U</b>	
UART	
Block Diagram of UART .....	496
Block Diagram of UART Pins .....	500
Notes on Using UART .....	539
Operation of UART .....	528
UART Baud Rate Selection .....	520
UART EI <sup>2</sup> OS Functions .....	515
UART Functions (x 2) .....	494
UART Interrupt and EI <sup>2</sup> OS .....	495
UART Interrupts .....	514
UART Interrupts and EI <sup>2</sup> OS .....	515
UART Pins .....	499
UART Registers .....	502
USB	
Bank Registers (PCB,DTB,USB,SSB,ADB) .....	53
User Stack Pointer	
User Stack Pointer (USP) .....	45
USP	
User Stack Pointer (USP) .....	45
<b>W</b>	
Watchdog Timer	
Block Diagram of the Watchdog Timer .....	253
Usage Notes on the Watchdog Timer .....	259
Watchdog Timer Function.....	252
Watchdog Timer Operation .....	257
Watchdog Timer Control Register	
Watchdog Timer Control Register (WDTC).....	255
Waveform Control Register	
Waveform Control Register (SIGCR).....	385
Waveform Generator	
Block Diagram of Waveform Generator .....	349
Usage Notes on the Waveform Generator .....	417
Waveform Generator .....	345
Waveform Generator Interrupts .....	389
Waveform Generator Interrupts and EI <sup>2</sup> OS .....	390
Waveform Generator Registers .....	356
WDTC	
Watchdog Timer Control Register (WDTC).....	255
Write	
Detailed Explanation on the Flash Memory Write/ Delete .....	572
Write Operation	
When the Write Operation is Executed. ....	566
When the Write Operation or Chip/Sector Deletion Operation is Executed. ....	568, 570
Writing	
Notes on Writing the Data .....	574
Procedure for Writing/Deleting the Data to the Flash Memory .....	558
Procedure of Writing the Data to the Flash Memory .....	574
Writing the Data.....	574

CM44-10147-2E

---

**FUJITSU MICROELECTRONICS • CONTROLLER MANUAL**

F<sup>2</sup>MC-16LX

16-BIT MICROCONTROLLER

MB90820B Series

HARDWARE MANUAL

---

July 2008 the second edition

Published **FUJITSU MICROELECTRONICS LIMITED**

Edited Business & Media Promotion Dept.

---

