The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

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

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

MB90880 Series

F²MC-16LX Hardware Manual

Doc. No. 002-04554 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600
www.cypress.com

Copyrights

© Cypress Semiconductor Corporation, 2007-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Preface

Purpose of This Manual and Intended Readers

Thank you very much for your continued patronage of Cypress semiconductor products.

MB90880 series is a 16-bit microcontroller designed for applications such as consumer devices requiring high-speed real-time processing. MB90880 series functions are suitable for controlling PHS, cellular phones, CD-ROMs, and VTRs.

This manual, intended for engineers developing products using the MB90880 series, explains the MB90880 series functions and operations. Read this manual first, before using the product.

Composition of This Manual

This manual consists of the following 24 chapters and an appendix.

1. Overview Of MB90880 Series

This chapter gives an overview of MB90880 series, including its basic features and basic specifications.

2. CPU Functions

This chapter explains CPU functions of the MB90880 series.

3. Interrupt

This chapter explains the functions and operation of the interrupt, extended intelligent I/O service (EI^{2}OS), and DMA in MB90880 series.

4. Reset

This chapter explains reset for the MB90880 series.

5. Clocks

This chapter describes the clocks of the MB90880 series.

6. Low-power Consumption Mode

This chapter explains the low-power consumption mode of the MB90880 series.

7. Mode Setting

This chapter explains mode setting, mode pins, mode data, external memory access and its operation.

8. I/O Port

This chapter explains the functions and registers of I/O port.

9. Time-base Timer

This chapter provides an overview of the time-base timer explains the configuration, the control register, interrupt, its operation, notes, and program example of the time-base timer.

10. Watchdog Timer

This chapter explains the function and operation of the watchdog timer.

11. Watch Timer

This chapter provides an overview of the watch timer, explains the configuration, the control register, and the operation of the watch timer.

12. 16-bit I/O Timer

This chapter provides an overview of the 16-bit I/O timer, explains the configuration, configuration and functions of its registers, interrupt and its operation.

13. 8/16-bit Up/Down Counter/Timer

This chapter provides an overview of the 8/16-bit up/down counter/timer, explains the configuration, configuration and functions of its registers, interrupt and its operation.

14. PPG Timer

This chapter describes the operations of the PPG timer.

15. Base Timer

This chapter describes the overview of base timer, configuration and function of its registers and operations of the base timer.

16. DTP/External Interrupts

This chapter provides an overview of the DTP/external interrupt unit, explains configuration and functions of its registers and its operation, and shows the precautions on use.

17. 8/10-Bit A/D Converter

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

18. Multi-function Serial Interface

This chapter describes an overview of the multi-function serial interface and the configuration of the registers. For details, refer to sections 18.4 through 18.7 explaining each mode.

19. Chip Selection Facility

This chapter provides an overview of the chip selection facility, explains the configuration, its operation, the configuration and functions of its registers.

20. Address Match Detection Function

This chapter explains the functions and operations of the address match detection.

21. ROM Mirror Function Selection Module

This chapter provides an overview of the ROM mirror function selection module and explains its registers.

22. Flash Memory

This chapter describes the functions and operations of the 2M/3M/4M-bit flash memory.

23. Examples Of Flash Memory Products Serial Programming Connection

This chapter shows an example of a serial programming connection using the AF220/AF210/AF120/AF110 flash microcontroller programmer by Yokogawa Digital Computer Corporation.

24. Delay Interrupt Generation Module

This chapter describes the functions and operations of the delay interrupt generation module.

A. Appendix

The appendix provides the memory map and lists the instructions used in the F^2MC-16LX.

Contents

1. Overview Of MB90880 Series 11
   - 1.1 Features of MB90880 Series 12
   - 1.2 Block Diagram of MB90880 Series 16
   - 1.3 Package Dimensions 17
   - 1.4 Pin Assignment 19
   - 1.5 Pin Functions 21
   - 1.6 I/O Circuit Type 29
   - 1.7 Handling the Device 33
2. CPU Functions 37
   - 2.1 Overview of CPU Specifications 38
   - 2.2 Memory Space 39
   - 2.3 CPU Registers 43
     - 2.3.1 Accumulator (A) 45
     - 2.3.2 User Stack Pointer (USP) and System Stack Pointer (SSP) 46
     - 2.3.3 Processor Status (PS) 47
     - 2.3.4 Program Counter (PC) 49
     - 2.3.5 Program Counter Bank Register (PCB) 50
     - 2.3.6 Direct Page Register (DPR) 51
     - 2.3.7 General-Purpose Register (Register Bank) 52
   - 2.4 Prefix Codes 53
3. Interrupt 57
   - 3.1 Overview of Interrupt 58
   - 3.2 Interrupt Vectors 61
   - 3.3 Interrupt Control Registers (ICR00 to ICR15) 62
   - 3.4 Interrupt Flow 65
   - 3.5 Hardware Interrupt 67
     - 3.5.1 Hardware Interrupt Operation 68
     - 3.5.2 Hardware Interrupt Generation and Release 69
     - 3.5.3 Multiple Interrupt 71
   - 3.6 Software Interrupt 72
   - 3.7 Extended Intelligent I/O Services (EI^{2}OS) 74
     - 3.7.1 Extended Intelligent I/O Service Descriptor (ISD) 76
     - 3.7.2 EI^{2}OS Status Register (ISCS) 78
   - 3.8 Operational Flow and Procedure of Extended Intelligent I/O Services (EI^{2}OS) 79
   - 3.9 μDMA Controller (μDMAC) 82
   - 3.10 μDMAC Register 83
     - 3.10.1 DMAC Descriptor Channel Select Register (DCSR) 85
     - 3.10.2 DMAC Status Register (DSRL/DSRH) 86
     - 3.10.3 DMAC Stop Status Register (DSSR) 87
     - 3.10.4 DMAC Enable Register (DERL/DERH) 88
   - 3.11 DMA Descriptor Window Register (DDWR) 89
     - 3.11.1 Data Counter (DCTL/DCTH) 90
     - 3.11.2 I/O Register Address Pointer (IOAL/IOAH) 91

3.11.3 DMA Control Register (DMACS) 92
3.11.4 Buffer Address Pointer (BAPL/BAPM/BAPH) 94
3.12 μDMAC Operation 95
3.13 Exceptions 98
4. Reset 99
4.1 Overview of Reset 100
4.2 Reset Factors and Oscillation Stabilization Wait Time 102
4.3 External-Reset Pin 103
4.4 Reset Operation 104
4.5 Reset-Factor Bits 106
4.6 State of Pins by Reset 108
5. Clocks 109
5.1 Overview of Clocks 110
5.2 Block Diagram of Clock Generator 112
5.3 Clock Selection Register (CKSCR) and PLL Output Selection Register (PLLOS) 114
5.4 Clock Modes 120
5.5 Oscillation Stabilization Wait Time 124
5.6 Connecting Oscillator and External Clock 125
6. Low-power Consumption Mode 127
6.1 Overview of Low-Power Consumption Mode 128
6.2 Block Diagram of Low-Power Consumption Control Circuit 130
6.3 Low-Power Consumption Mode Control Register (LPMCR) 132
6.4 CPU Intermittent Operation Mode 135
6.5 Standby Mode 136
6.5.1 Sleep Mode 137
6.5.2 Time-base Timer Mode 139
6.5.3 Watch Mode 140
6.5.4 Stop Mode 142
6.6 State Transition Diagram in Standby Mode 144
6.7 Pin State in Standby Mode, Hold, and Reset 146
6.8 Caution on Using Low-Power Consumption Mode 151
7. Mode Setting 155
7.1 Mode Setting 156
7.2 Mode Pins (MD2 to MD0) 157
7.3 Mode Data 158
7.4 External Memory Access 162
7.4.1 Automatic Ready Function Selection Register (ARSR) 164
7.4.2 External Address Output Control Register (HACR) 165
7.4.3 Bus Control Signal Selection Register (EPCR) 166
7.5 Operation of Each Mode for Mode Setting 168
7.5.1 External Memory Access Control Signals 169
7.5.2 Ready Function 172
7.5.3 Hold Function 175
8. I/O Port 177
8.1 Functions of I/O Port 178
8.2 Registers for I/O Port 179
8.2.1 Port Data Registers (PDR0 to PDRA) 180
8.2.2 Port Direction Registers (DDR0 to DDRA) 181
8.2.3 Other Registers 182
9. Time-base Timer 187
9.1 Overview of Time-base Timer 188
9.2 Configuration of Time-base Timer 189
9.3 Time-base Timer Control Register (TBTC) 190

9.4 Interrupt of Time-base Timer 192
9.5 Operation of Time-base Timer 193
9.6 Notes on Using Time-base Timer 196
10. Watchdog Timer 197
10.1 Overview of Watchdog Timer 198
10.2 Configuration of Watchdog Timer 199
10.3 Register for Watchdog Timer 201
10.3.1 Watchdog Timer Control Register (WDTC) 202
10.4 Operation of Watchdog Timer 204
10.5 Precautions for Using Watchdog Timer 207
11. Watch Timer 209
11.1 Overview of Watch Timer 210
11.2 Configuration of Watch Timer 211
11.3 Watch Timer Control Register (WTC) 212
11.4 Operation of Watch Timer 214
12. 16-bit I/O Timer 215
12.1 Overview of 16-bit I/O timer 216
12.2 Configuration of 16-bit I/O timer 217
12.3 Configuration and Function of 16-bit I/O timer Register 221
12.3.1 Free-Run Timer 222
12.3.2 Output Compare 226
12.3.3 Input Capture 230
12.4 Interrupt of 16-Bit I/O timer 234
12.5 Operation of 16-Bit I/O timer 236
12.5.1 Operation of Free-Run Timer 237
12.5.2 Operation of Output Compare 239
12.5.3 Operation of Input Capture 241
12.5.4 Timing of Free-Run Timer 242
12.5.5 Timing of Output Compare 243
12.5.6 Timing of Input Capture 244
13. 8/16-bit Up/Down Counter/Timer 245
13.1 Overview of 8/16-Bit Up/Down Counter/Timer 246
13.2 Configuration of 8/16-Bit Up/Down Counter/Timer 247
13.3 Configuration and Functions of Registers for 8/16-Bit Up/Down Counter/Timer 250
13.3.1 Counter Control Register (ch.0) Upper (CCRH0) 251
13.3.2 Counter Control Register (ch.1) Upper (CCRH1) 253
13.3.3 Counter Control Register (ch.0/ch.1) Lower (CCRL0/CCRL1) 255
13.3.4 Counter Status Register 0/1 (CSR0/CSR1) 257
13.3.5 Up/Down Count Register (ch.0/ch.1) (UDCR0/UDCR1) 259
13.3.6 Reload/Compare Register (ch.0/ch.1) (RCR0/RCR1) 260
13.4 Interrupt of 8/16-Bit Up/Down Counter/Timer 261
13.5 Operation of 8/16-Bit Up/Down Counter/Timer 263
13.5.1 Reload/Compare Function 266
13.5.2 Writing Data to Up/Down Count Register (UDCR) 269
14. PPG Timer 273
14.1 Overview of PPG Timer 274
14.2 Block Diagram of PPG Timer 275
14.3 Registers of PPG Timer 277
14.3.1 List of Registers of PPG Timer 278
14.3.2 Details of Registers of PPG Timer 281
14.4 Interrupts of PPG Timer 288
14.5 Operational Description of PPG Timer 289
15. Base Timer 293

15.1 Overview of Base Timer 294
15.2 Block Diagram of Base Timer 296
15.3 Registers of Base Timer 299
15.4 Operations of Base Timer 303
15.5 Operation of 32-Bit Mode 305
15.6 Precautions when Using Base Timer 307
15.7 Interrupts of Base Timer 309
15.8 Functional Description of Base Timer 310
15.8.1 PWM Timer Function 311
15.8.2 PPG Timer Function 324
15.8.3 Reload Timer Function 337
15.8.4 PWC Timer Function 348
16. DTP/External Interrupts 363
16.1 Overview of DTP/External Interrupt Unit 364
16.2 Configuration and Functions of DTP/External Interrupt Unit Registers 366
16.3 DTP/External Interrupt 370
16.4 Operations of DTP/External Interrupt Circuit 372
16.4.1 External Interrupt Function 375
16.4.2 DTP Function 376
16.5 Precautions on Use of DTP/External Interrupt Unit 377
17. 8/10-Bit A/D Converter 379
17.1 Overview of the 8/10-Bit A/D Converter 380
17.2 Configuration of the 8/10-Bit A/D Converter 381
17.3 Pins of 8/10-Bit A/D Converter 383
17.4 Registers of 8/10-Bit A/D Converter 384
17.4.1 A/D Control Status Register (Upper) (ADCSH) 385
17.4.2 A/D Control Status Register (Lower) (ADCSL) 388
17.4.3 A/D Conversion Channel Setting Register (ADSR) 391
17.4.4 A/D Data Register (ADCRH/ADCRL) 394
17.5 Interrupts of 8/10-Bit A/D Converter 395
17.6 Operation of the 8/10-Bit A/D Converter 396
17.6.1 Conversion Operation Using EI^{2}OS 399
17.6.2 A/D Conversion Data Protection Function 400
17.7 Precautions When Using the 8/10-Bit A/D Converter 404
18. Multi-function Serial Interface 405
18.1 Overview of the Multi-function Serial Interface 406
18.2 Pins of the Multi-function Serial Interface 409
18.3 Registers of the Multi-function Serial Interface 410
18.4 UART (Asynchronous Serial Interface) 412
18.4.1 Registers of UART (Asynchronous Serial Interface) 414
18.4.2 Interrupts of the UART 427
18.4.3 Operation of the UART 430
18.4.4 Dedicated Baud Rate Generator 435
18.4.5 Setting Procedure and the Program Flow for Operation Mode 0 (Asynchronous Normal Mode) 440
18.4.6 Setting Procedure and the Program Flow for Operation Mode 1 (Asynchronous Multiprocessor Mode) 442
18.5 CSIO (Clock Synchronous Serial Interface) 444
18.5.1 Overview of the CSIO (Clock Synchronous Serial Interface) 445
18.5.2 Registers of the CSIO (Clock Synchronous Serial Interface) 446
18.5.3 Interrupts of the CSIO (Clock Synchronous Serial Interface) 459
18.5.4 Dedicated Baud Rate Generator 474
18.6 LIN-UART (LIN Communication Control UART) 479

18.6.1 Overview of the LIN-UART (LIN Communication Control UART) 480
18.6.2 Registers of the LIN-UART 481
18.6.3 Interrupts of the LIN-UART 493
18.6.4 Dedicated Baud Rate Generator 497
18.6.5 Operation of the LIN-UART 502
18.6.6 Setting Procedure and the Program Flow for Operation Mode 3 (LIN Communication Mode) 510
18.7 I^2C Interface 513
18.7.1 Overview of the I^2C Interface 514
18.7.2 Registers of the I^2C Interface 515
18.7.3 Interrupts of the I^2C Interface 532
18.7.4 Dedicated Baud Rate Generator 551
18.7.5 Precautions When Using I^2C Mode 556
19. Chip Selection Facility 557
19.1 Overview of Chip Selection Facility 558
19.2 Configuration of Chip Selection Facility 559
19.3 Configuration and Functions of Registers for Chip Selection Facility 561
19.3.1 Chip Selection Area MASK Registers (CMR0 to CMR3) 562
19.3.2 Chip Selection Area Registers (CAR0 to CAR3) 563
19.3.3 Chip Selection Control Register (CSCR) 564
19.3.4 Chip Selection Active Level Register (CALR) 565
19.4 Operation of the Chip Selection Facility 566
20. Address Match Detection Function 569
20.1 Overview of Address Match Detection Function 570
20.2 Block Diagram of Address Match Detection Function 571
20.3 Configuration for Address Match Detection Function 572
20.3.1 Address Detection Control Registers (PACSR0/PACSR1) 573
20.3.2 Detection Address Set Registers (PADR0 to PADR5) 577
20.4 Explanation of Operation of Address Match Detection Function 579
20.4.1 Example of Using Address Match Detection Function 580
21. ROM Mirror Function Selection Module 585
21.1 Overview of ROM Mirror Function Selection Module 586
21.2 ROM Mirror Function Selection Register (ROMM) 587
22. Flash Memory 589
22.1 Overview of Flash Memory 590
22.2 Sector Configuration of Flash Memory 591
22.3 Flash Memory Control Status Register (FMCS) 594
22.4 Flash Memory Write Control Registers (FWR0/FWR1) 596
22.5 Starting the Flash Memory Automatic Algorithm 602
22.6 Confirming the Automatic Algorithm Execution State 603
22.6.1 Data Polling Flag (DQ7) 604
22.6.2 Toggle Bit Flag (DQ6) 605
22.6.3 Timing Limit Exceeded Flag (DQ5) 606
22.6.4 Sector Erase Timer Flag (DQ3) 607
22.7 Writing Data to and Erasing Data from Flash Memory 608
22.7.1 Setting Flash Memory to the Read/Reset State 609
22.7.2 Write Data to Flash Memory 610
22.7.3 Erasing All Data of Flash Memory (Chip Erase) 612
22.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase) 613
22.7.5 Suspending Sector Erase of Flash Memory 615
22.7.6 Restarting Sector Erase of Flash Memory 616
22.8 Flash Security Function 617
22.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems 618

22.10 Notes on Using Flash Memory 621
23. Examples Of Flash Memory Products Serial Programming Connection 623
23.1 Basic Configuration of Flash Memory Products Serial Programming Connection ... 624
23.2 Examples of Serial Programming Connections 627
24. Delay Interrupt Generation Module 631
24.1 Overview of Delay Interrupt Generation Module 632
24.2 Operation of Delay Interrupt Generation Module 633
A. Appendix 635
A.1 Memory Map 636
A.2 I/O Map 638
A.3 Interrupt Source, Interrupt Vector, and Interrupt Control Register 650
A.4 Instructions 652
A.4.1 Instruction Types 653
A.4.2 Addressing 654
A.4.3 Direct Addressing 656
A.4.4 Indirect Addressing 662
A.4.5 Execution Cycle Count 669
A.4.6 Effective address field 671
A.4.7 How to Read the Instruction List 672
A.4.8 F^2MC-16LX Instruction List 674
A.4.9 Instruction Map 688
Major Changes 711
Revision History 713

1. Overview Of MB90880 Series

This chapter gives an overview of MB90880 series, including its basic features and basic specifications.

1.1 Features of MB90880 Series

1.2 Block Diagram of MB90880 Series

1.3 Package Dimensions

1.4 Pin Assignment

1.5 Pin Functions

1.6 I/O Circuit Type

1.7 Handling the Device

1.1 Features of MB90880 Series

MB90880 series is a 16-bit microcontroller designed for applications requiring high-speed real-time processing.

Features of MB90880 Series

The MB90880 series has the following features:

Minimum instruction execution time

- 30.3 ns/4.125 MHz oscillation multiplied by 8 (33 MHz/3.3 V ± 0.3 V for internal operation)
- PLL clock multiply system

Maximum memory space: 16 Mbytes

Instruction set optimized for control applications

- Available data types: bit, byte, word, long-word
- Standard addressing modes: 23 types
- Improved high-precision operation using a 32-bit accumulator
- Signed multiply and divide operations, extensive RETI instruction

Instruction set supporting multitasking in high-level languages (such as C)

- Use of a system stack pointer
- Symmetry of instruction sets and barrel shift instructions

Non-multiplex bus and multiplex bus support

Improved execution speed: 4-byte queue

Improved interrupt function (priority is a programmable setting of up to 8 levels):
24 external interrupts

Data transfer function (μDMAC): maximum of 16 channels

Built-in ROM:Flash version: 256 Kbytes/384 Kbytes/512 Kbytes,
Mask version: 256 Kbytes/384 Kbytes/512 Kbytes

Built-in RAM:Flash version:16 Kbytes/24 Kbytes/30 Kbytes,
Mask version:16 Kbytes/24 Kbytes/30 Kbytes

General-purpose ports:maximum of 83 ports
Single clock model (product without S-suffix): maximum of 83 ports
Dual clock model (product without S-suffix): maximum of 81 ports

A/D converter (RC successive approximation type): 20 channels (resolution: 8/10 bits)

16-bit PPG: 8 channels
(8 bits × 6 channels and 16 bits × 3 channels with mode switching function)

8/16-bit up/down timer: 1 channel
(8 bits × 2 channels or 16 bits × 1 channel with mode switching function)

16-bit I/O timer
(input capture × 2 channels; output compare × 6 channels; free-run timer × 1 channel)

Built-in dual-system clock generator (support product without S-suffix only)

Low-power consumption mode
(stop mode, sleep mode, CPU intermittent operation mode, watch timer/time-base timer mode)

Package: QFP100/LQFP100

CMOS technology

3V single power supply

Product Configuration

Table 1-1 lists the MB90880 series product configuration.

Table 1-1. MB90880 Series Product Configuration (Sheet 1 of 2)

		MB90882(S)	MB90F882A(S)	MB90F883B(S)/ MB90F883BH(S)/ MB90F883C(S)	MB90F884B(S)/ MB90F884BH(S)/ MB90F884C(S)	MB90V880A-101, 102
Class		MASK ROM product	Flash memory product			Evaluation product
ROM size		256 Kbytes	256 Kbytes	384 Kbytes	512 Kbytes	-
RAM size		16 Kbytes	16 Kbytes	24 Kbytes	30 Kbytes	30 Kbytes
CPU functions		Number of instructions : 351 Instruction bit length : 8 bits, 16 bits Instruction length : 1to7 bytes Data bit length : 1 bit, 8 bits, 16 bits Minimum instruction execution time : 30.3 ns (machine clock: 33 MHz) The maximum operating frequency of MB90F883B(S) and MB90F884B(S) is 25 MHz.				
Ports		General-purpose I/O ports: up to 81 for dual clock product (product without S-suffix), up to 83 for single clock product (product with S-suffix) Pins X0A and X1A cannot be used as I/O ports in dual clock model. General-purpose I/O ports (CMOS output)				
Multi-function serial		7 channels (software switchable between SIO, UART & I^2C)				
16-bit PPG timer		8 channels				
8/16-bit up/down counter/timer		Event input pins: 6, 8-bit up/down counters: 2 8-bit reload/compare registers: 2				
16-bit I/O timer	16-bit free-run timer	1 channel Overflow interrupt				
	Output compare (OCU)	6 channels Pin input source: Match signal of compare register				
	Input capture (ICU)	2 channels Rewriting register by pin input (rising, falling or both edges)				
DTP/external interrupt circuit		External interrupt pins: 24 channels (edge/level support)				
Base timer		4 channels (software switchable between 16-bit reload timer, PWC timer, PPG timer, and PWM timer)				
Time-base timer		18-bit counter Interrupt interval: 1.0 ms, 4.1 ms, 16.4 ms, 131.1 ms (source oscillation: 4 MHz)				
A/D converter		Conversion accuracy: 8 or 10 bits can be switched Single conversion mode (Selected channel converted only once) Scan conversion mode (Multiple successive channels converted) Successive conversion mode (Selected channel converted repeatedly) Stop conversion mode (Selected channel converted and stopped repeatedly)				
Watchdog timer		Reset generation interval: 3.58 ms, 14.33 ms, 57.23ms, 458.75 ms (source oscillation: 4 MHz, minimum value)				
Low power consumption (standby) modes		Sleep, stop, CPU intermittent operation, watch timer, time-base timer				

Table 1-1. MB90880 Series Product Configuration (Sheet 2 of 2)

	MB90882(S)	MB90F882A(S)	MB90F883B(S)/ MB90F883BH(S)/ MB90F883C(S)	MB90F884B(S)/ MB90F884BH(S)/ MB90F884C(S)	MB90V880A-101, 102
Flash memory	-	Flash security/write-protect features (not implemented in MB90F883B(S), MB90F884B(S), MB90F883BH(S), MB90F884BH(S))			-
Process	CMOS technology				
Emulator-dedicated power supply*	-				Included

*: This refers to the setting of jumper switch (TOOL VCC) when using the emulator.

For details, see "3.3 Emulator-dedicated Power Supply Switching" in MB2147-01 or MB2147-20 hardware manual.

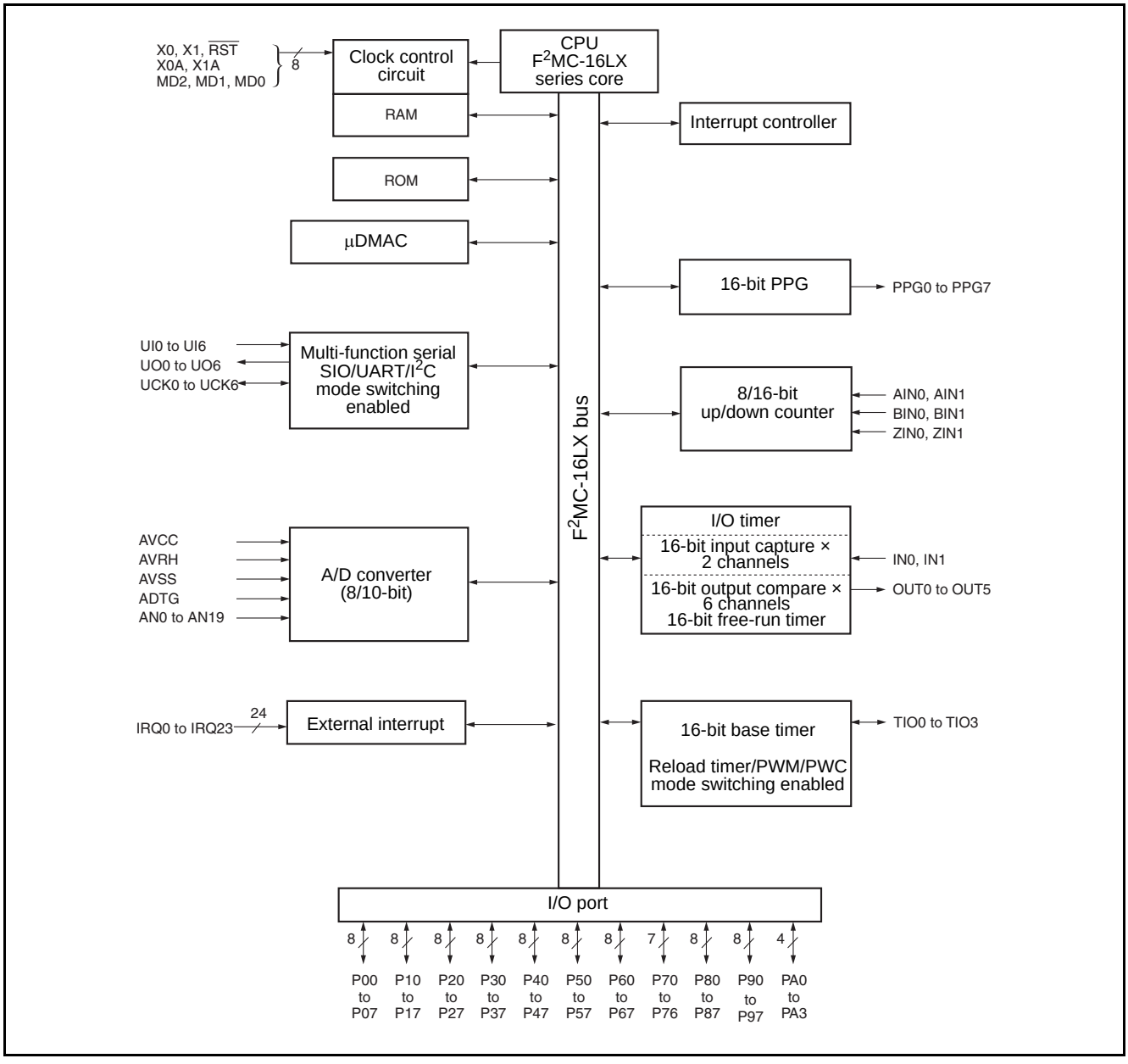
1.2 Block Diagram of MB90880 Series

This section shows a block diagram of the MB90880 series.

Block Diagram of MB90880 Series

Figure 1-1 is a block diagram of the MB90880 series.

Figure 1-1. Block Diagram of MB90880 Series



Note:

In the Figure 1-1, the I/O port shares a pin with each built-in function block. The pin cannot be used as an I/O port if it is used as a built-in module pin.

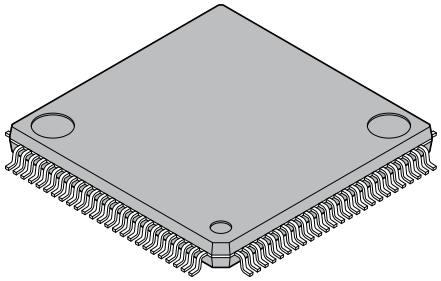
1.3 Package Dimensions

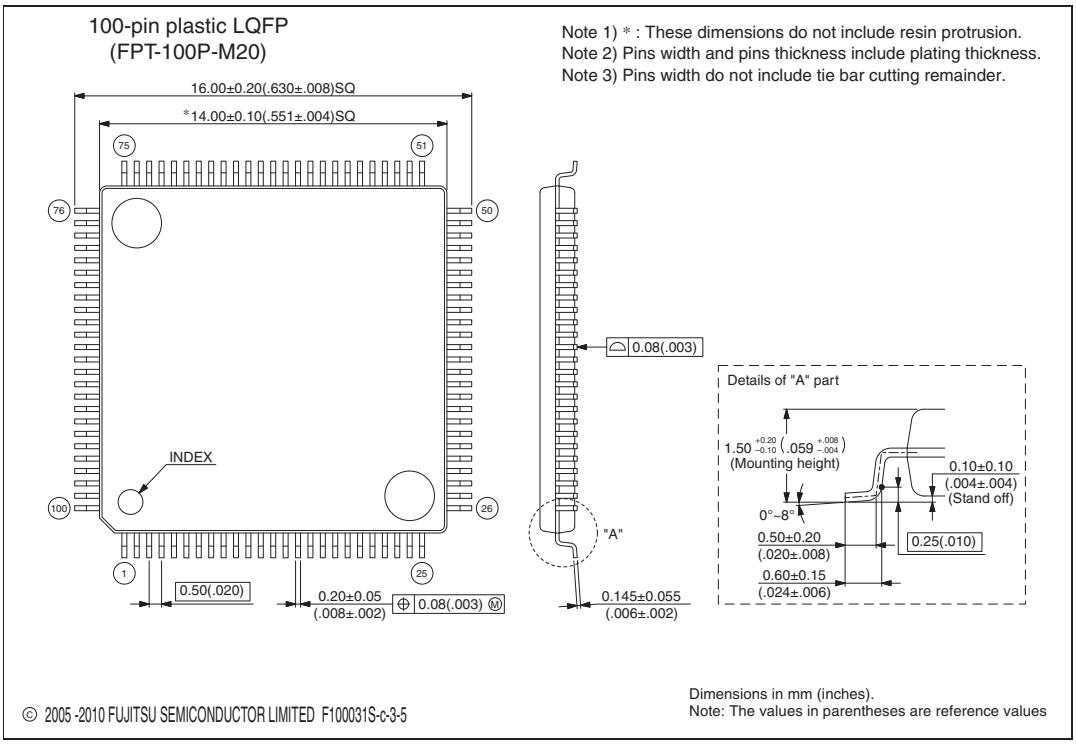
MB90880 series has two types of packages.

Package Dimension (LQFP-100)

Figure 1-2 is the package dimension of the LQFP-100 type.

Figure 1-2. Package Dimension of LQFP-100 Type

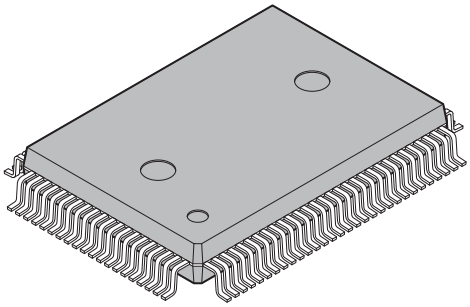
100-pin plastic LQFP		
	Lead pitch	0.50 mm
	Package width × package length	14.0 mm × 14.0 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm Max
	Weight	0.65 g
(FPT-100P-M20)	Code (Reference)	P-LFQFP100-14×14-0.50

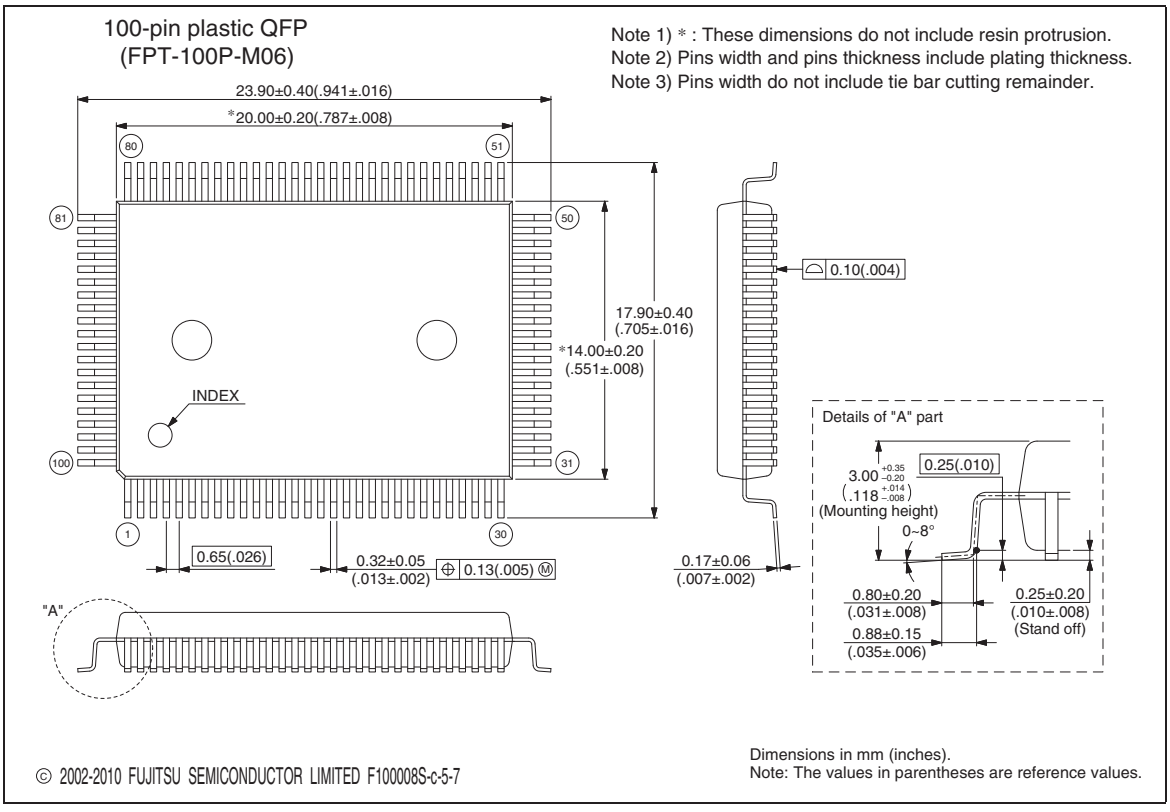


Package Dimension (QFP-100)

Figure 1-3 is the package dimension of the QFP-100 type.

Figure 1-3. Package Dimension of QFP-100 Type

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



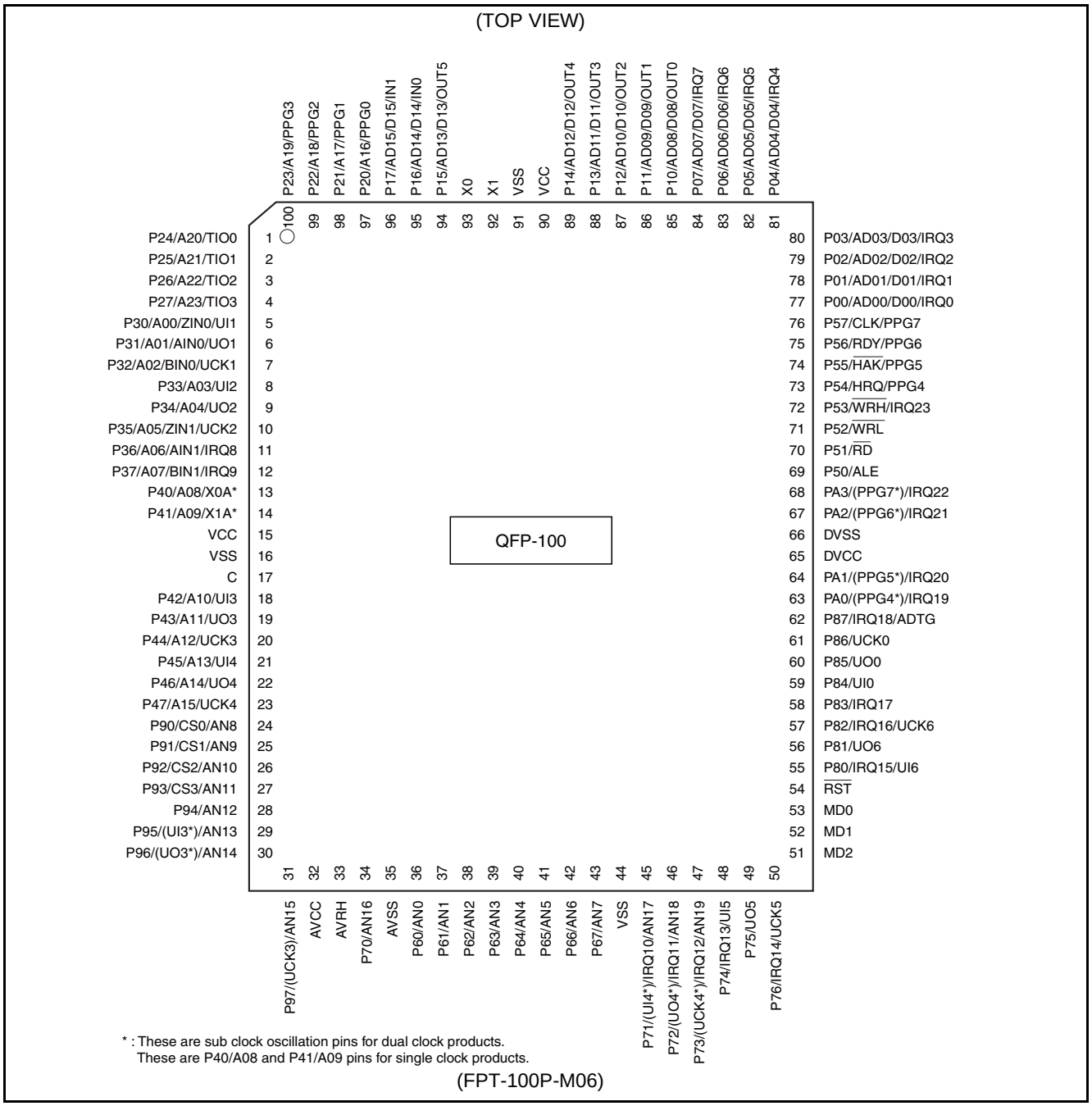
1.4 Pin Assignment

This section shows the MB90880 series pin assignments for two types of packages.

Pin Assignment (QFP-100)

Figure 1-4 is a pin assignment for the QFP-100 type.

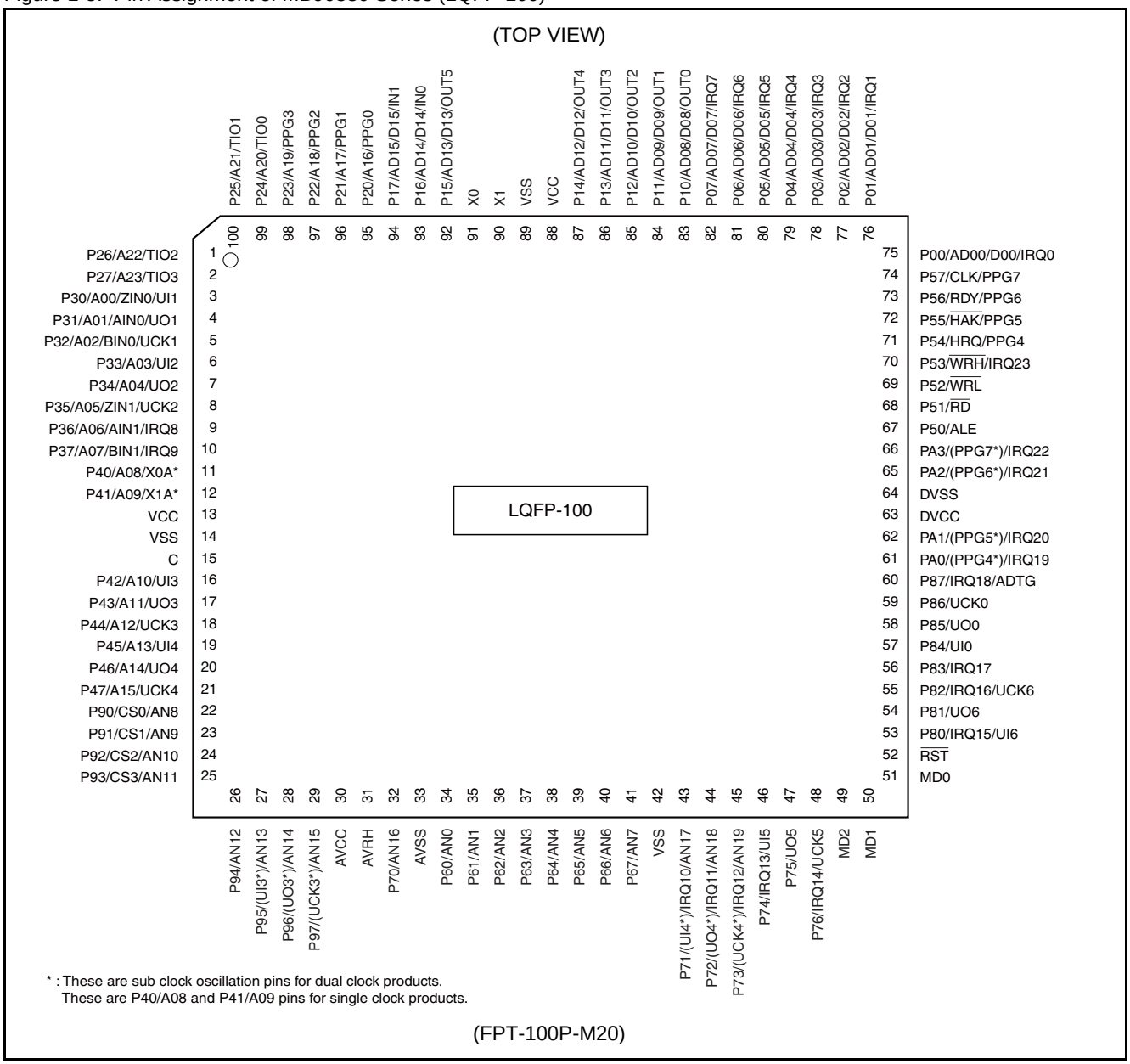
Figure 1-4. Pin Assignment of MB90880 Series (QFP-100)



Pin Assignment (LQFP-100)

Figure 1-5 is a pin assignment for the LQFP-100 type.

Figure 1-5. Pin Assignment of MB90880 Series (LQFP-100)



1.5 Pin Functions

This section explains the pin functions of the MB90880 series.

Pin Functions

Table 1-2 explains the pin functions of MB90880 series.

Table 1-2. Pin Functions (Sheet 1 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
1	3	P26	D	General-purpose I/O port
		A22		In multiplex mode, this pin serves as higher address output pin (A22) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A22) when corresponding bit in external address output control register (HACR) is set to "0".
		TIO2		Base timer I/O pin (ch.2)
2	4	P27	D	General-purpose I/O port
		A23		In multiplex mode, this pin serves as higher address output pin (A23) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A23) when corresponding bit in external address output control register (HACR) is set to "0".
		TIO3		Base timer I/O pin (ch.3)
3	5	P30	E	General-purpose I/O port
		A00		Serves as an external address pin in non-multiplex mode.
		ZIN0		8/16-bit up/down timer input pin (ch.1)
		UI1		Multi-function serial input pin
4	6	P31	E	General-purpose I/O port
		A01		Serves as an external address pin in non-multiplex mode.
		AIN0		8/16-bit up/down timer input pin (ch.1)
		UO1/ (SDA1)		Multi-function serial output pin
5	7	P32	E	General-purpose I/O port
		A02		Serves as an external address pin in non-multiplex mode.
		BIN0		8/16-bit up/down timer input pin (ch.1)
		UCK1/ (SCL1)		Multi-function serial clock I/O pin
6	8	P33	E	General-purpose I/O port
		A03		Serves as an external address pin in non-multiplex mode.
		UI2		Multi-function serial input pin
7	9	P34	E	General-purpose I/O port
		A04		Serves as an external address pin in non-multiplex mode.
		UO2/ (SDA2)		Multi-function serial output pin
8	10	P35	E	General-purpose I/O port
		A05		Serves as an external address pin in non-multiplex mode.
		ZIN1		8/16-bit up/down timer input pin (ch.0)
		UCK2/ (SCL2)		Multi-function serial clock I/O pin

Table 1-2. Pin Functions (Sheet 2 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
9	11	P36	D	General-purpose I/O port
		A06		Serves as an external address pin in non-multiplex mode.
		AIN1		8/16-bit up/down timer input pin (ch.0)
		IRQ8		External interrupt input pin
10	12	P37	D	General-purpose I/O port
		A07		Serves as an external address pin in non-multiplex mode.
		BIN1		8/16-bit up/down timer input pin (ch.0)
		IRQ9		External interrupt input pin
11	13	P40	A/D	General-purpose I/O port (Only for single clock devices)
		A08		Serves as an external address pin in non-multiplex mode. (Only for single clock devices)
		X0A		32 kHz oscillator connecting pin (Only for dual clock devices)
12	14	P41	A/D	General-purpose I/O port (Only for single clock devices)
		A09		Serves as an external address pin in non-multiplex mode. (Only for single clock devices)
		X1A		32 kHz oscillator connecting pin (Only for dual clock devices)
13	15	VCC	-	Power supply pin
14	16	VSS	-	Power supply pin (GND)
15	17	C	-	Regulator stabilization capacity connecting pin
16	18	P42	E	General-purpose I/O port
		A10		Serves as an external address pin in non-multiplex mode.
		UI3		Multi-function serial input pin
17	19	P43	E	General-purpose I/O port
		A11		Serves as an external address pin in non-multiplex mode.
		UO3/ (SDA3)		Multi-function serial output pin
18	20	P44	E	General-purpose I/O port
		A12		Serves as an external address pin in non-multiplex mode.
		UCK3/ (SCL3)		Multi-function serial clock I/O pin
19	21	P45	E	General-purpose I/O port
		A13		Serves as an external address pin in non-multiplex mode.
		UI4		Multi-function serial input pin
20	22	P46	E	General-purpose I/O port
		A14		Serves as an external address pin in non-multiplex mode.
		UO4/ (SDA4)		Multi-function serial output pin
21	23	P47	E	General-purpose I/O port
		A15		Serves as an external address pin in non-multiplex mode.
		UCK4/ (SCL4)		Multi-function serial clock I/O pin
22	24	P90	H	General-purpose I/O port
		CS0		Chip select 0
		AN8		Analog input pin

Table 1-2. Pin Functions (Sheet 3 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
23	25	P91	H	General-purpose I/O port
		CS1		Chip select 1
		AN9		Analog input pin
24	26	P92	H	General-purpose I/O port
		CS2		Chip select 2
		AN10		Analog input pin
25	27	P93	H	General-purpose I/O port
		CS3		Chip select 3
		AN11		Analog input pin
26	28	P94	H	General-purpose I/O port
		AN12		Analog input pin
27	29	P95	K	General-purpose I/O port
		AN13		Analog input pin
		(UI3*)		Multi-function serial input pin (when set by P9FSR register)
28	30	P96	K	General-purpose I/O port
		AN14		Analog input pin
		(UO3*)/ (SDA3)		Multi-function serial output pin (when set by P9FSR register)
29	31	P97	K	General-purpose I/O port
		AN15		Analog input pin
		(UCK3*)/ (SCL3)		Multi-function serial clock I/O pin (when set by P9FSR register)
30	32	AVCC	-	A/D converter power supply pin
31	33	AVRH	-	Reference voltage input pin for A/D converter This power supply must be turned on or off while a voltage higher than or equal to AVRH is applied to AVCC.
32	34	P70	H	General-purpose I/O port
		AN16		Analog input pin
33	35	AVSS	-	A/D converter analog GND pin
34	36	P60	H	General-purpose I/O port
		AN0		Analog input pin
35	37	P61	H	General-purpose I/O port
		AN1		Analog input pin
36	38	P62	H	General-purpose I/O port
		AN2		Analog input pin
37	39	P63	H	General-purpose I/O port
		AN3		Analog input pin
38	40	P64	H	General-purpose I/O port
		AN4		Analog input pin
39	41	P65	H	General-purpose I/O port
		AN5		Analog input pin
40	42	P66	H	General-purpose I/O port
		AN6		Analog input pin

Table 1-2. Pin Functions (Sheet 4 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
41	43	P67	H	General-purpose I/O port
		AN7		Analog input pin
42	44	VSS	-	Power supply pin (GND)
43	45	P71	K	General-purpose I/O port
		IRQ10		External interrupt input pin
		AN17		Analog input pin
		(UI4*)		Multi-function serial input pin (when set by P7FSR register)
44	46	P72	K	General-purpose I/O port
		IRQ11		External interrupt input pin
		AN18		Analog input pin
		(UO4*)/ (SDA4)		Multi-function serial output pin (when set by P7FSR register)
45	47	P73	K	General-purpose I/O port
		IRQ12		External interrupt input pin
		AN19		Analog input pin
		(UCK4*)/ (SCL4)		Multi-function serial clock I/O pin (when set by F7FSR register)
46	48	P74	G	General-purpose I/O port
		IRQ13		External interrupt input pin
		UI5		Multi-function serial input pin
47	49	P75	G	General-purpose I/O port
		UO5/ (SDA5)		Multi-function serial output pin
48	50	P76	G	General-purpose I/O port
		IRQ14		External interrupt input pin
		UCK5/ (SCL5)		Multi-function serial clock I/O pin
49	51	MD2	M	Operation mode specification input pin
50	52	MD1	L	Operation mode specification input pin
51	53	MD0	L	Operation mode specification input pin
52	54	RST	B	Reset input pin
53	55	P80	G	General-purpose I/O port
		IRQ15		External interrupt input pin
		UI6		Multi-function serial input pin
54	56	P81	G	General-purpose I/O port
		UO6/ (SDA6)		Multi-function serial output pin
55	57	P82	G	General-purpose I/O port
		IRQ16		External interrupt input pin
		UCK6/ (SCL6)		Multi-function serial clock I/O pin
56	58	P83	I	General-purpose I/O port
		IRQ17		External interrupt input pin
57	59	P84	G	General-purpose I/O port
		UI0		Multi-function serial input pin

Table 1-2. Pin Functions (Sheet 5 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
58	60	P85	G	General-purpose I/O port
		UO0/ (SDA0)		Multi-function serial output pin
59	61	P86	G	General-purpose I/O port
		UCK0/ (SCL0)		Multi-function serial clock I/O pin
60	62	P87	I	General-purpose I/O port
		IRQ18		External interrupt input pin
		ADTG		External trigger input pin when A/D converter is used
61	63	PA0	J	General-purpose I/O port
		IRQ19		External interrupt input pin
		(PPG4*)		PPG timer output pin (when set by PAFSR register)
62	64	PA1	J	General-purpose I/O port
		IRQ20		External interrupt input pin
		(PPG5*)		PPG timer output pin (when set by PAFSR register)
63	65	DVCC	-	PA port power supply pin
64	66	DVSS	-	PA port power supply pin (GND)
65	67	PA2	J	General-purpose I/O port
		IRQ21		External interrupt input pin
		(PPG6*)		PPG timer output pin (when set by PAFSR register)
66	68	PA3	J	General-purpose I/O port
		IRQ22		External interrupt input pin
		(PPG7*)		PPG timer output pin (when set by PAFSR register)
67	69	P50	F	General-purpose I/O port
		ALE		Serves as address latch enable signal (ALE) pin in external bus mode.
68	70	P51	F	General-purpose I/O port
		RD		Serves as read strobe output ($\overline{RD}$) pin in external bus mode.
69	71	P52	F	General-purpose I/O port
		WRL		Serves as lower data write strobe output ($\overline{WRL}$) pin in external bus mode, and serves as a general-purpose I/O port when WRE bit in EPCR register is "0".
70	72	P53	F	General-purpose I/O port
		WRH		Serves as higher data write strobe output ($\overline{WRH}$) pin in external bus mode with 16-bit bus width, and serves as a general-purpose I/O port when WRE bit in EPCR register is "0".
		IRQ23		External interrupt input pin
71	73	P54	F	General-purpose I/O port
		HRQ		Serves as hold request input (HRQ) pin in external bus mode, and serves as a general-purpose I/O port when HDE bit in EPCR register is "0".
		PPG4		PPG timer output pin
72	74	P55	F	General-purpose I/O port
		HAK		Serves as hold acknowledge output ($\overline{HAK}$) pin in external bus mode, and serves as a general-purpose I/O port when HDE bit in EPCR register is "0".
		PPG5		PPG timer output pin

Table 1-2. Pin Functions (Sheet 6 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
73	75	P56	F	General-purpose I/O port
		RDY		Serves as external ready input (RDY) pin in external bus mode, and serves as a general-purpose I/O port when RYE bit in EPCR register is "0".
		PPG6		PPG timer output pin
74	76	P57	F	General-purpose I/O port
		CLK		Serves as machine cycle clock output (CLK) pin in external bus mode, and serves as a general-purpose I/O port when CKE bit in EPCR register is "0".
		PPG7		PPG timer output pin
75	77	P00	C	General-purpose I/O port
		AD00/D00		In multiplex mode, it serves as lower external address/data bus I/O pin.
				Serves as lower external data bus output pin in non-multiplex mode.
		IRQ0		External interrupt input pin
76	78	P01	C	General-purpose I/O port
		AD01/D01		Serves as an external address/lower data bus I/O pin in multiplex mode.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ1		External interrupt input pin
77	79	P02	C	General-purpose I/O port
		AD02/D02		Serves as an external address/lower data bus I/O pin in multiplex mode.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ2		External interrupt input pin
78	80	P03	C	General-purpose I/O port
		AD03/D03		Serves as an external address/lower data bus I/O pin in multiplex mode.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ3		External interrupt input pin
79	81	P04	C	General-purpose I/O port
		AD04/D04		In multiplex mode, it serves as lower external address/data bus I/O pin.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ4		External interrupt input pin
80	82	P05	C	General-purpose I/O port
		AD05/D05		In multiplex mode, it serves as lower external address/data bus I/O pin.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ5		External interrupt input pin
81	83	P06	C	General-purpose I/O port
		AD06/D06		In multiplex mode, it serves as lower external address/data bus I/O pin.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ6		External interrupt input pin
82	84	P07	C	General-purpose I/O port
		AD07/D07		In multiplex mode, it serves as lower external address/data bus I/O pin.
				Serves as a lower external data bus output pin in non-multiplex mode.
		IRQ7		External interrupt input pin

Table 1-2. Pin Functions (Sheet 7 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
83	85	P10	C	General-purpose I/O port
		AD08/D08		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		OUT0		Output compare event output pin
84	86	P11	C	General-purpose I/O port
		AD09/D09		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		OUT1		Output compare event output pin
85	87	P12	C	General-purpose I/O port
		AD10/D10		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		OUT2		Output compare event output pin
86	88	P13	C	General-purpose I/O port
		AD11/D11		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		OUT3		Output compare event output pin
87	89	P14	C	General-purpose I/O port
		AD12/D12		In non-multiplex mode, it serves as higher external data output pin.
		OUT4		Output compare event output pin
88	90	VCC	-	Power supply pin
89	91	VSS	-	Power supply pin (GND)
90	92	X1	A	Main oscillator connecting pin
91	93	X0	A	Main oscillator connecting pin
92	94	P15	C	General-purpose I/O port
		AD13/D13		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		OUT5		Output compare event output pin
93	95	P16	C	General-purpose I/O port
		AD14/D14		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		IN0		Trigger input pin for input capture ch.0
94	96	P17	C	General-purpose I/O port
		AD15/D15		In multiplex mode, it serves as higher external address/data bus I/O pin.
				In non-multiplex mode, it serves as higher external data output pin.
		IN1		Trigger input pin for input capture ch.1
95	97	P20	D	General-purpose I/O port
		A16		In multiplex mode, this pin serves as higher address output pin (A16) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A16) when corresponding bit in external address output control register (HACR) is set to "0".
		PPG0		PPG timer output pin

Table 1-2. Pin Functions (Sheet 8 of 8)

Pin no.		Pin name	I/O circuit type*3	Function
LQFP *1	QFP *2			
96	98	P21	D	General-purpose I/O port
		A17		In multiplex mode, this pin serves as higher address output pin (A17) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A17) when corresponding bit in external address output control register (HACR) is set to "0".
		PPG1		PPG timer output pin
97	99	P22	D	General-purpose I/O port
		A18		In multiplex mode, this pin serves as higher address output pin (A18) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A18) when corresponding bit in external address output control register (HACR) is set to "0".
		PPG2		PPG timer output pin
98	100	P23	D	General-purpose I/O port
		A19		In multiplex mode, this pin serves as higher address output pin (A19) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A19) when corresponding bit in external address output control register (HACR) is set to "0".
		PPG3		PPG timer output pin
99	1	P24	D	General-purpose I/O port
		A20		In multiplex mode, this pin serves as higher address output pin (A20) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A20) when corresponding bit in external address output control register (HACR) is set to "0".
		TIO0		Base timer I/O pin (ch.0)
100	2	P25	D	General-purpose I/O port
		A21		In multiplex mode, this pin serves as higher address output pin (A21) when corresponding bit in external address output control register (HACR) is set to "0".
				In non-multiplex mode, this pin serves as higher address output pin (A21) when corresponding bit in external address output control register (HACR) is set to "0".
		TIO1		Base timer I/O pin (ch.1)

*1: LQFP: FPT-100P-M20

*2: QFP: FPT-100P-M06

*3: For the I/O circuit type, refer to Section “1.6 I/O Circuit Type”.

1.6 I/O Circuit Type

This section explains the I/O circuit type of pins in MB90880 series.

I/O Circuit Type

Table 1-3 summarizes the I/O circuit type of pins in MB90880 series.

Table 1-3. I/O Circuit Type (Sheet 1 of 4)

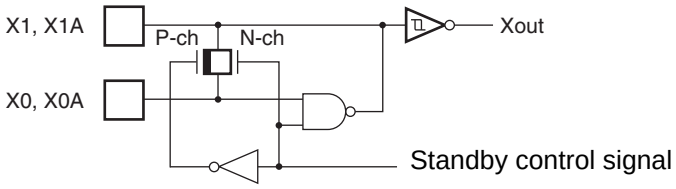
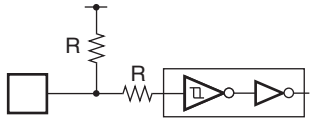
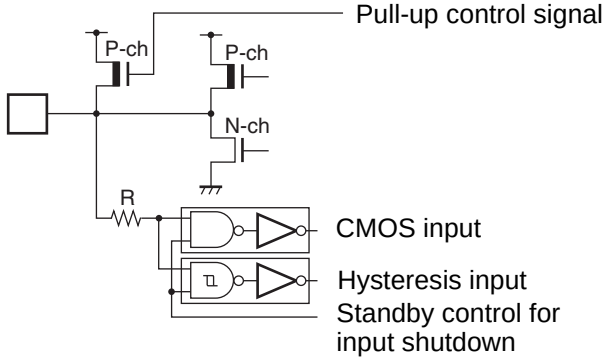
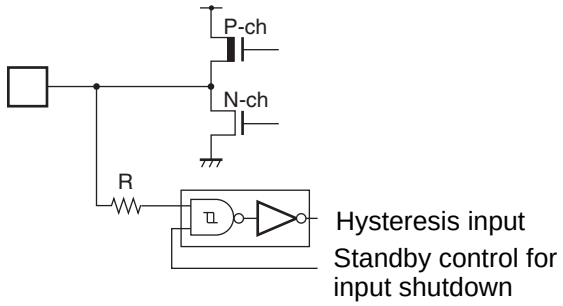
Type	Circuit	Remarks
A		■ Oscillation feedback resistor X1, X0: approx. 1 MΩ X1A, X0A: approx. 10 MΩ ■ Standby control provided
B		Hysteresis input with pull-up resistor
C		■ Input pull-up resistor control provided ■ CMOS level output ■ Hysteresis input ■ CMOS input (in external bus mode)
D		■ CMOS level output ■ Hysteresis input

Table 1-3. I/O Circuit Type (Sheet 2 of 4)

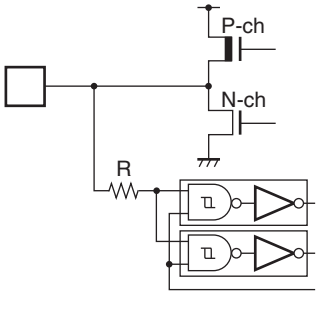
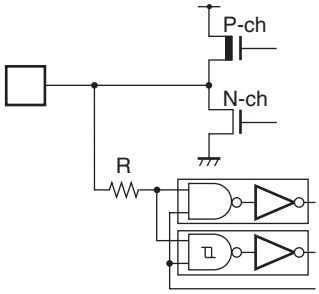
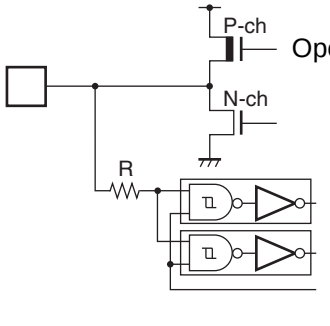
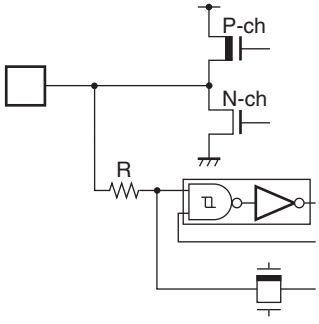
Type	Circuit	Remarks
E		■ CMOS level output ■ Hysteresis input ■ I2C level hysteresis input
F		■ CMOS level output ■ Hysteresis input ■ CMOS input (in external bus mode)
G		■ CMOS level output (Open-drain control provided) ■ 5V tolerant ■ Hysteresis input ■ I2C level hysteresis input
H		■ CMOS level output ■ Hysteresis input ■ Analog input

Table 1-3. I/O Circuit Type (Sheet 3 of 4)

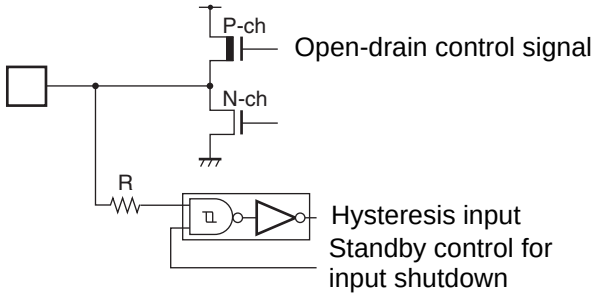
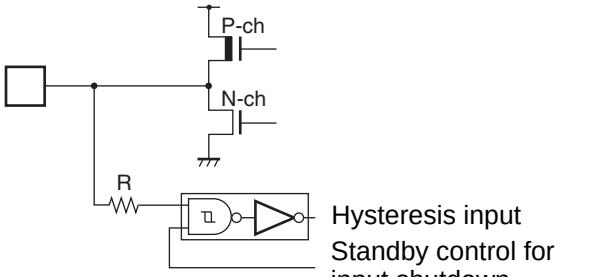
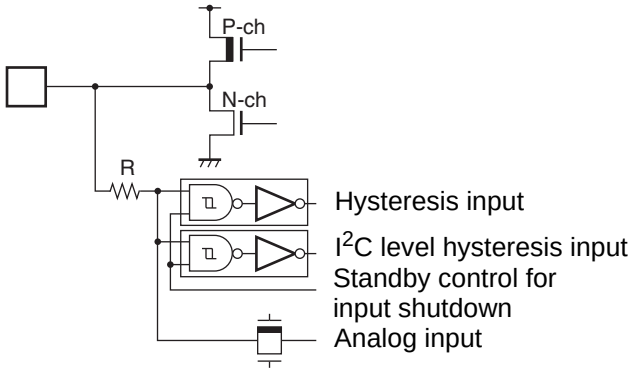
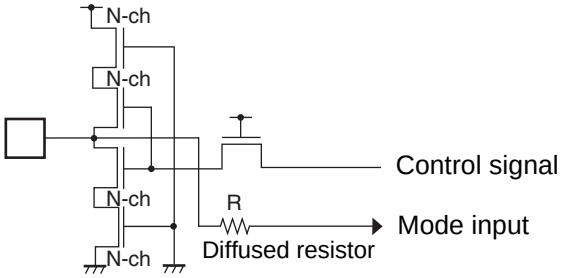
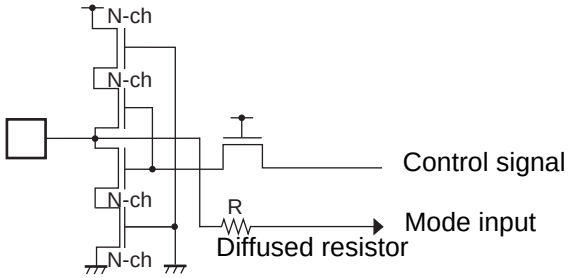
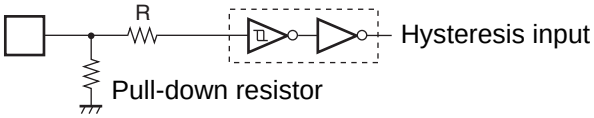
Type	Circuit	Remarks
I		■ CMOS level output (Open-drain control provided) ■ 5V tolerant ■ Hysteresis input
J		■ CMOS level output (high-current type) ■ Hysteresis input
K		■ CMOS level output ■ Hysteresis input ■ Analog input ■ I^2C level hysteresis input

Table 1-3. I/O Circuit Type (Sheet 4 of 4)

Type	Circuit	Remarks
L	(Flash memory product) 	(Flash memory product) ■ CMOS level input ■ High-voltage control for flash test provided
	(MASK ROM product) 	(MASK ROM product) ■ Hysteresis input
M	Flash memory product 	Flash memory product ■ CMOS level input ■ High-voltage control for flash test provided
	MASK ROM product, Evaluation product 	■ With pull-down resistor Hysteresis input

1.7 Handling the Device

This section gives notes on handling the MB90880 series.

Notes on Handling the Device

Latch-up prevention and power-on sequence

Do not apply a voltage higher than V_{CC} or lower than V_{SS} to input pins and output pins of MB90880 series products. Moreover, do not apply a voltage exceeding the rating range between VCC and VSS. When a voltage exceeding the rating is applied, a latch-up may occur. During a latch-up, the power supply current increases rapidly, which can result in heat-induced damage to elements. Ensure that the voltage applied does not exceed the maximum rating.

Once a latch-up occurs, the power supply current increases rapidly, possibly causing heat damage in the element. Be sure to prevent such damage during use.

Analog voltage must be supplied at the same time as V_{CC}, or it must be applied after the digital power supply is turned on. (Turn off the analog power supply before or at the same time as the power supply is turned off.)

Processing unused input pins

Leaving an unused input pin open may cause permanent damage because of malfunction and latch-up.
Keeping an unused input pin open may cause an error in operation or apply pull-up or pull-down to such pins with 2 kΩ or more resistor as necessary.
For an unused A/D converter, connect it so that AV_{CC} = AVRH = V_{CC}, and AV_{SS} = V_{SS}.

Handling a power supply pin (VCC/VSS)

If multiple VCC and/or VSS are used, all power supply pins must be connected with a power supply or ground externally in consideration of device design in order to decrease latch-up and unnecessary radiation and to prevent the malfunction of the strobe signal due to a rise of ground level. Be sure to connect all power supply pins to the power supply or ground so that the total output current specification is not exceeded.

As much as possible, the power supply source must be connected with VCC/VSS of this device at the lowest impedance.

Cypress recommends placing a bypass condenser of 0.1 μF between VCC and VSS.

Crystal oscillation circuit

Noise near the X0, X1, X0A, or X1A pin causes this device to malfunction. Design PC boards so that the X0 and X1 (X0A and A1A) pins, crystal oscillators (or ceramic oscillators), and bypass capacitors to the ground can be places as close possible.

In the interest of stable operation, it is strongly recommended that a PC board artwork that encloses the surroundings of the X0, X1, X0A, and X1A pins with the ground should be used.

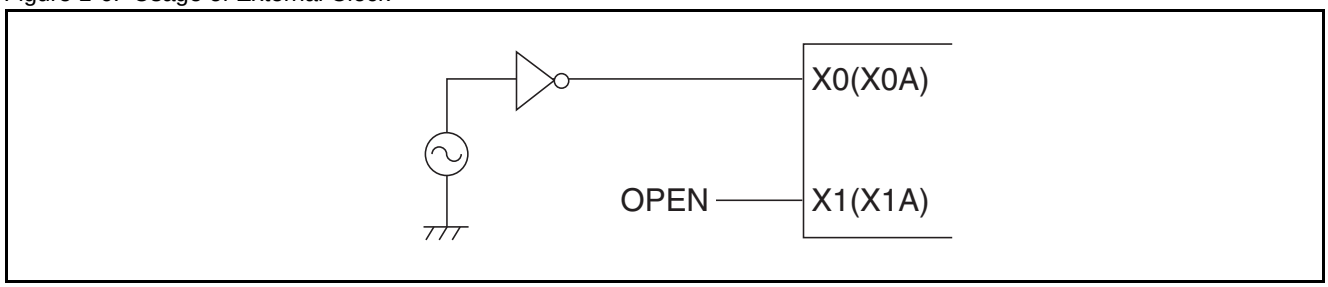
Please ask the crystal maker to evaluate the oscillational characteristics of the crystal and this device.

Notes on using external clock

To use external clock, drive only pin X0(X0A). Be sure to set up pin X1(X1A) to be opened.

Figure 1-6 shows a usage of the external clock (f=25 MHz or below).

Figure 1-6. Usage of External Clock



Note on operations during PLL clock mode

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

Turning-on Sequence of Power Supply to A/D Converter and Analog Inputs

Make sure to turn on the A/D converter power supply (AV_{CC}, AVRH) and analog inputs (AN0 to AN19)

after turning-on the digital power supply (V_{CC}).

Turn-off the digital power supply after turning off the A/D converter power supply (AV_{CC}, AVRH) and analog inputs (AN0 to AN19).

In this case, make sure that the AVRH does not exceed AV_{CC}.

The input voltage must not exceed AV_{CC} even when you use the analog input and the pin used combinable as an input port either.

Connection of Unused Pins of A/D Converter

Connect unused pins of A/D converter as AV_{CC} = AVRH = V_{CC}, AV_{SS} = V_{SS}.

Notes on Energization

To prevent malfunction of the internal step-down circuit, the voltage rise time at power-on should be 50 µs or more (0.2 V to 2.7 V).

Stabilizing the power supply

Even in the range of V_{CC} power supply voltage, a rapid change in the power supply may cause a misoperation. Therefore, the V_{CC} supply voltage should be stabilized. For stabilization reference, the supply voltage should be controlled so that V_{CC} ripple variations (P-P values) at commercial frequencies (50 Hz/60 Hz) fall below 10% of the standard V_{CC} supply voltage and the transient fluctuation rate does not exceed 0.1 V/ms in the momentary fluctuation, such as switching the power supply.

Notes on using MB90F883B(S) and MB90F884B(S)

- The maximum operating frequency is 25 MHz.
- Flash security and write-protect features are not implemented.

Note on using MB90F883BH(S), MB90F884BH(S)

Flash security and write-protect features are not implemented.

Serial communication

At serial communication, incorrect data may be received by noise, etc. To prevent such incorrectly receiving, the board should be designed to reduce the noise. Considering the incorrect data receiving rarely by noise, etc., retransmit the data, for example, adding the checksum data, etc. finally when an error occurs.

When using a dual clock product as a dual clock product

X0A pin must be connected toV_{SS}, and X1A pin must be open.

2. CPU **Functions**

This chapter explains CPU functions of the MB90880 series.

2.1 Overview of CPU Specifications

2.2 Memory Space

2.3 CPU Registers

2.4 Prefix Codes

2.1 Overview of CPU Specifications

This section gives an overview of the CPU specifications.

Overview of the CPU Specifications

The F^{2}MC-16LX CPU core is a 16-bit CPU designed for applications such as consumer devices that requires high-speed real-time processing. The F^{2}MC-16LX instruction set is designed for controller applications, providing high-speed and high-efficiency control processes.

In addition to 16-bit data processing, the F^{2}MC-16LX CPU core can provide 32-bit data processing with an installed internal 32-bit accumulator. Memory spaces are a maximum of 16 M bytes (expandable) and can be accessed by using a linear pointer or bank. Based on the F^{2}MC-8L AT architecture, its instruction system is improved because of adding the instructions supporting high-level languages, expanding addressing modes, improving multiply and divide operation instructions, and enhancing bit processing. The followings are features of F^{2}MC-16LX CPU.

Minimum instruction execution time

- 30.3 ns/4.125 MHz multiplied by 8 of source oscillation
  (when internal operation: 33 MHz/3.3 V ± 0.3V)
- PLL clock multiply system

Maximum memory space: 16 M bytes, accessing by using a linear pointer or bank

Instruction set optimized for control applications

- Data types available: bit, byte, word, long-word
- Standard addressing mode: Use of 23-type, 32-bit accumulator for enhancing high-precision operation
- Signed multiply and divide operations, expanded RETI instruction

Enhanced interrupt function: 8 priority levels (programmable)

CPU independent automatic transfer function: Up to 16 channels of μDMAC

Multitasking-compatible instruction set in high-level language (C)

Use of system stack pointers, symmetrical instruction set, and barrel shift instruction

Improved execution speed: 4-byte queue

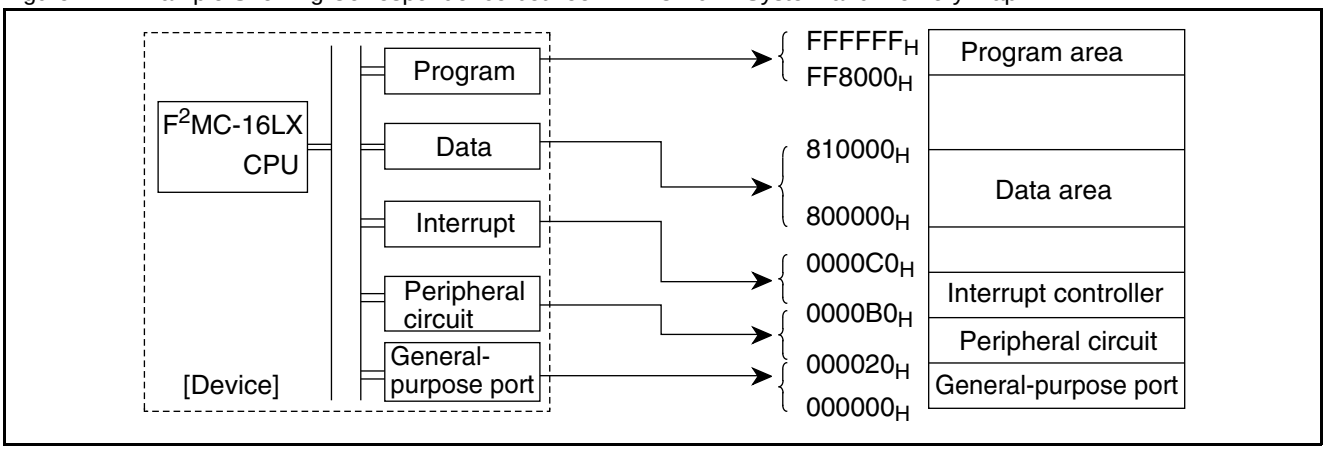
2.2 Memory Space

The F^{2}MC-16LX CPU has a 16M bytes memory space, to which all input to and output from the F^{2}MC-16LX CPU controlled data program is allocated. CPU has a 24-bit address bus to access each resource.

Memory Map

Figure 2-1 shows the F^{2}MC-16LX system and the associated memory map.

Figure 2-1. Example Showing Correspondence between F^{2}MC-16LX System and Memory Map



Address Generation Type

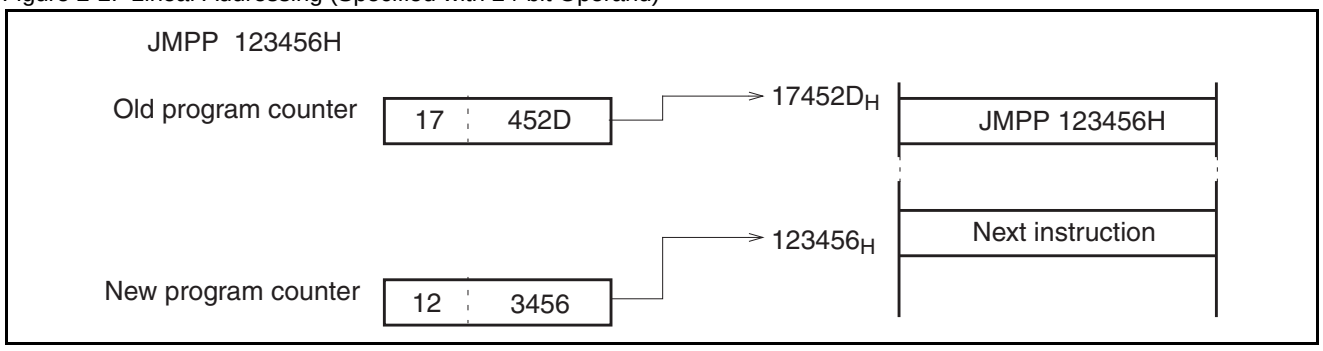
The F^{2}MC-16LX CPU has two types of address generation. One is linear addressing that specifies all 24-bit addresses with instructions. The other is bank addressing that specifies upper 8-bit addresses with appropriate bank registers and lower 16-bit addresses with instructions.

Linear addressing has two types: one uses operands to directly specify 24-bit addresses; the other refers to contents of the lower 24 bits in a 32-bit general-purpose register as addresses.

Linear addressing (specified with 24-bit operand)

Figure 2-2 shows an example of linear addressing scheme specified with 24-bit operands.

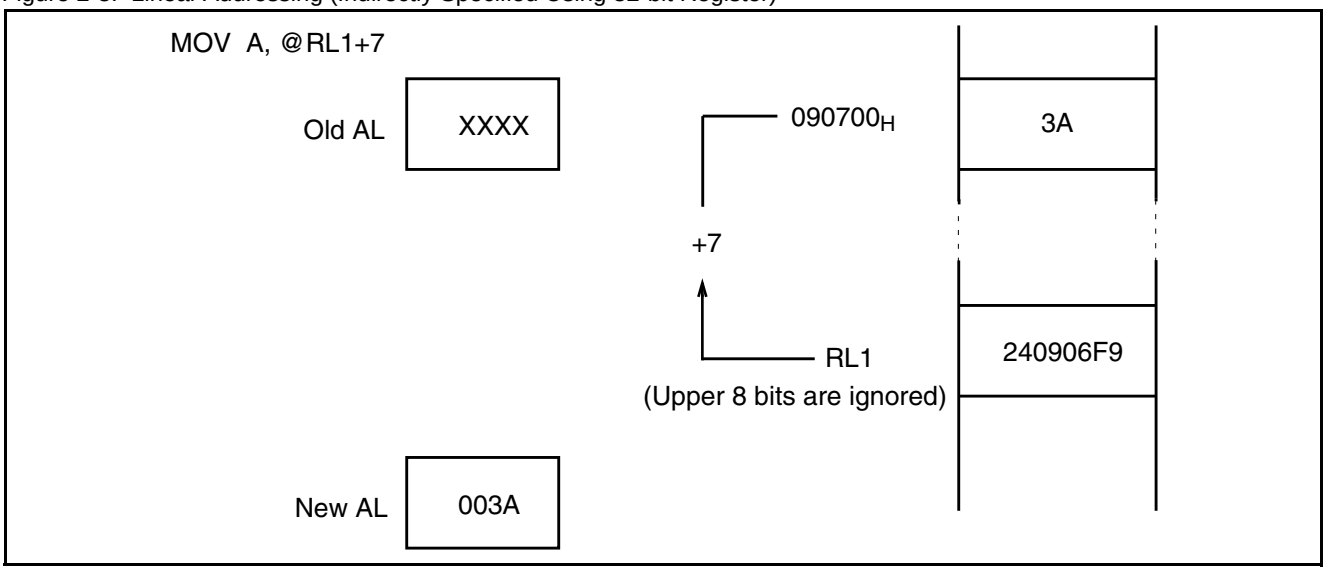
Figure 2-2. Linear Addressing (Specified with 24-bit Operand)



Linear addressing (indirectly specified using 32-bit register)

Figure 2-3 shows an example of linear addressing scheme indirectly specified using a 32-bit register.

Figure 2-3. Linear Addressing (Indirectly Specified Using 32-bit Register)



Addressing Type by Bank

Bank addressing divides a 16M bytes space into 256 banks of 64 Kbytes each, using five bank registers to specify banks for each space.

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

A 64 Kbytes bank specified with PCB is called the program (PC) space. The PC space includes such information as instruction codes, vector tables, and immediate data.

A 64 Kbytes bank specified with DTB is called the data (DT) space. The DT space includes writable data, and internal and external resource control/data registers.

A 64 Kbytes bank specified with USB or SSB is called the stack (SP) space. The SP space is accessed if a stack access occurs by saving the push/pop instruction or interrupt register. The stack space to be accessed is determined by the S-flag in the condition code register.

A 64 Kbytes bank specified with ADB is called the additional (AD) space. The AD space includes, for example, the data that cannot be included in the DT space.

As shown in Table 2-1, each addressing mode uses a default space defined in advance to improve the efficiency of coding instructions. If an addressing mode uses a space other than the default space, a prefix code corresponding to the bank must be specified prior to the instruction code, enabling access to any bank space corresponding to the prefix code.

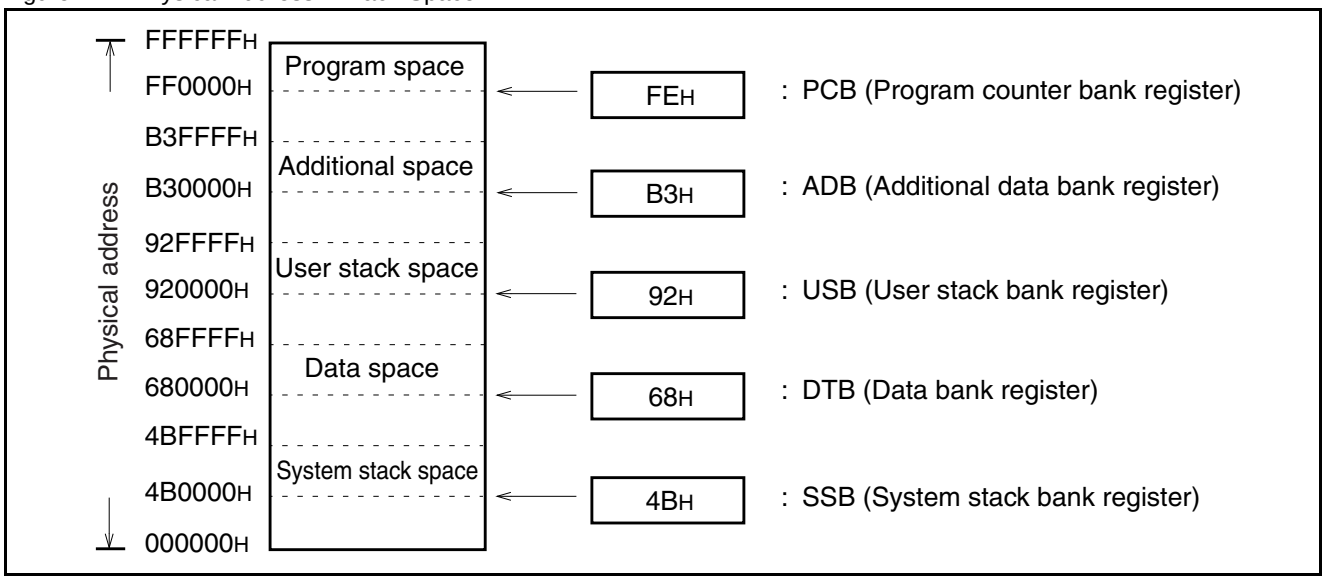
After resetting, DTB, USB, SSB and ADB are initialized to 00_H, and PCB is initialized to the value specified by a reset vector. After resetting, each space for DT, SP and AD is allocated to bank 00_H (000000_H to $00FFFF_H$), and each space for PC is allocated to the bank specified by the reset vector.

Table 2-1. Default Space

Default space	Addressing mode
Program space	PC indirect, program access, branch instruction
Data space	Addressing mode using @RW0, @RW1, @RW4, and @RW5; @A; addr16; dir
Stack space	Addressing mode using PUCHW, POPW, @RW3, and @RW7
Additional space	Addressing mode using @RW2 and @RW6

Figure 2-4 shows an example of a memory space divided for a register bank.

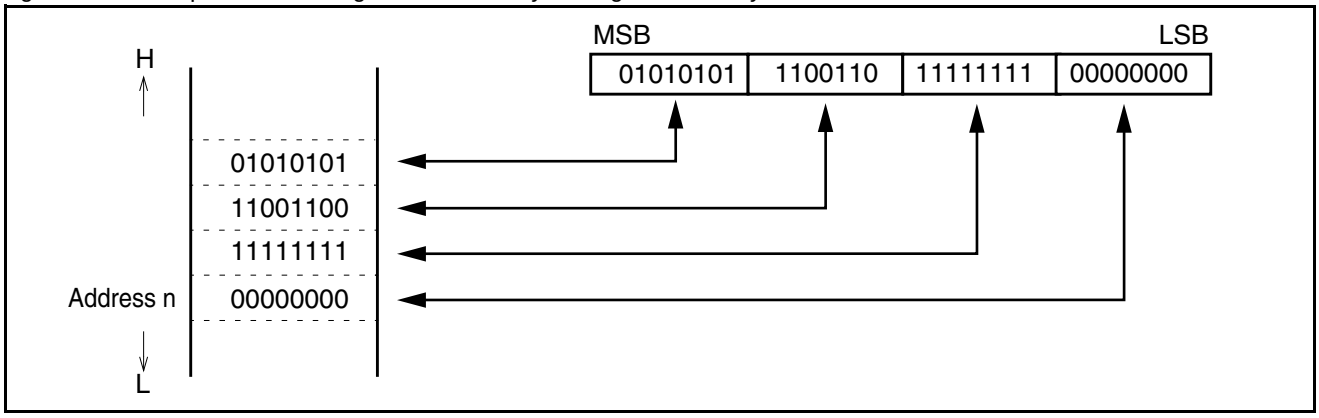
Figure 2-4. Physical Address in Each Space



Allocation for Data of Multi-Byte Length in Memory Space

Figure 2-5 shows the configuration for data of a multi-byte length in memory. The lower 8 bits of a data item are stored at address n, then address n+1, address n+2, address n+3, etc.

Figure 2-5. Example for Allocating Data of Multi-Byte Length in Memory



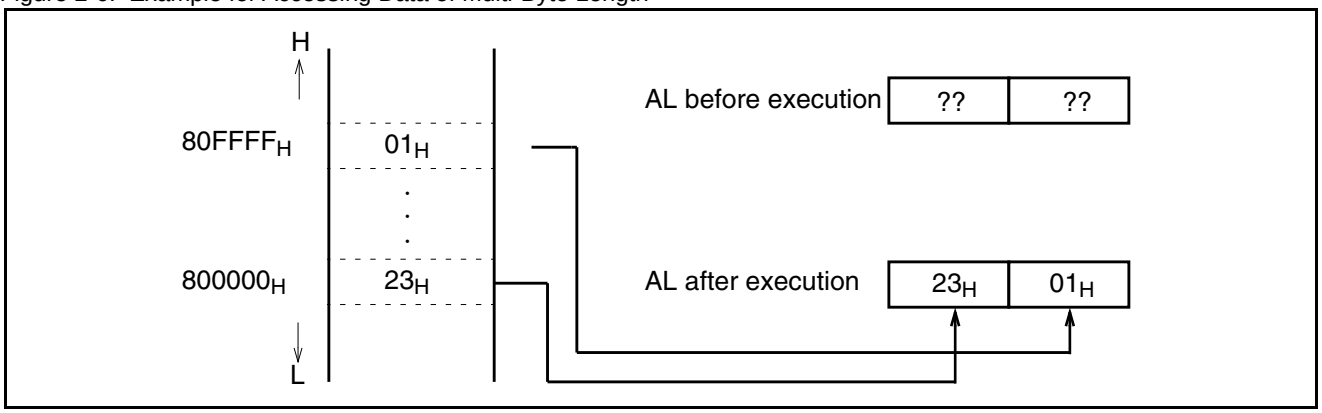
Data is written to memory in sequence starting from the lower addresses. Thus, the lower 16 bits of a 32-bit data item are transferred first, followed by the upper 16 bits. If a reset signal is input immediately after writing the lower bit, writing the upper bit may fail.

Access to Data of Multi-Byte Length

Figure 2-6 shows an example for accessing data of a multi-byte length.

In this example, MOVW A, 030FFFFH is executed.

Figure 2-6. Example for Accessing Data of Multi-Byte Length



2.3 CPU Registers

The F^{2}MC-16LX registers are divided into special registers inside CPU and general-purpose registers on memory. The former is dedicated hardware inside the CPU, and its use is limited because of the CPU architecture. The latter shares CPU address spaces with RAM. General-purpose registers are same as special registers so that they can be accessed without specifying the address. However, the user can specify the use same as the normal memory space.

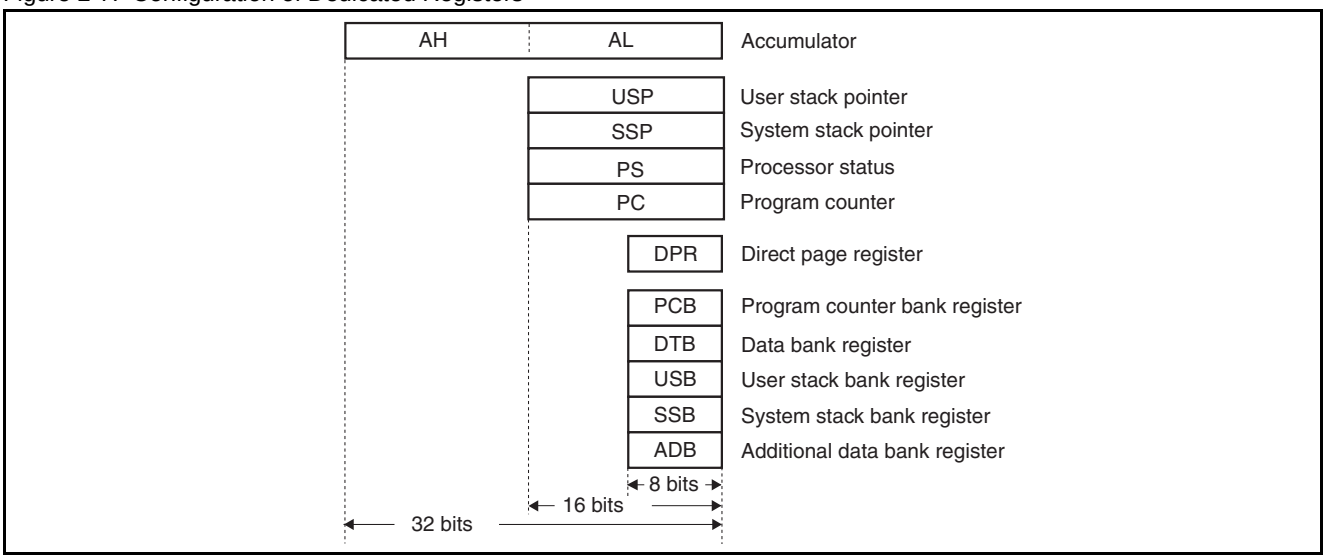
Dedicated Registers

The F^{2}MC-16LX has the following 11 types of dedicated registers:

- Accumulator (A = AH: AL) : Two 16-bit accumulators (usable as single 32-bit accumulator)
- User stack pointer (USP) : 16-bit pointer pointing to user stack area
- System stack pointer (SSP) : 16-bit pointer pointing to system stack area
- Processor status (PS) : 16-bit register indicating system status
- Program counter (PC) : 16-bit register containing a program address
- Direct page register (DPR) : 8-bit register indicating a direct page
- Program counter bank register (PCB) : 8-bit register indicating a PC space
- Data bank register (DTB) : 8-bit register indicating a DT space
- User stack bank register (USB) : 8-bit register indicating a user stack space
- System stack bank register (SSB) : 8-bit register indicating a system stack space
- Additional data bank register (ADB) : 8-bit register indicating an AD space

Figure 2-7 shows the configuration of the dedicated registers.

Figure 2-7. Configuration of Dedicated Registers



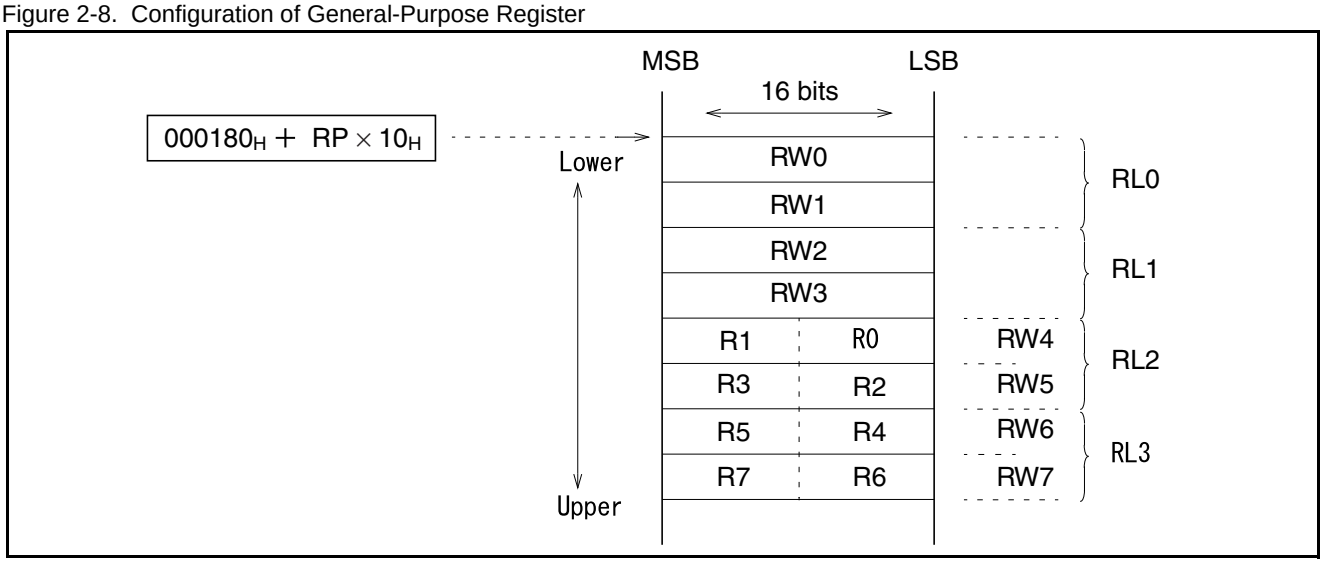
General-Purpose Register

The F^{2}MC-16LX general-purpose register resides on the main memory addresses: 000180_H to $00037F_H$ (maximum configuration). It uses a register bank register (RP) to indicate which part of addresses are currently used for register banks. Each bank has the three types of registers listed below. They are dependent on one another, as shown in Figure 2-8.

- R0 to R7 : 8-bit general-purpose register
- RW0 to RW7 : 16-bit general-purpose register
- RL0 to RL3 : 32-bit general-purpose register

Figure 2-8 shows the configuration of a general-purpose register.

Figure 2-8. Configuration of General-Purpose Register



The relationship between upper and lower bytes in a byte register and word register is represented with the following formula: RW(i + 4) = RW(i × 2 + 1) × 256 + R(i × 2) [i = 0 to 3]. The relationship of upper and lower bytes in RLi is represented with the following formula:

RW(i) = RW(i × 2 + 1) × 65536 + RW(i × 2) [i = 0 to 3].

2.3.1 Accumulator (A)

This section explains the functions of accumulator (A).

Accumulator (A)

An accumulator (A) consists of two 16-bit arithmetic operation registers (AH/AL) that are used to store operation results and temporarily store data transfer results. For 32-bit data processing, AH is connected with AL. For word processing in the 16-bit data processing mode and for byte processing in the 8-bit data processing mode, only AL is used. Data stored in an accumulator (A) is used together with that in memory and registers (Ri, RWi, RLi), the data item with a smaller word length is transferred to AL. This enables data items in AL before the transfer to be automatically transferred to AH (data hold function). The data hold function and operation between AL-AH support improvements in processing efficiency.

During a transfer of a data item with a lower byte length to AL, a sign extension or zero extension is added to the data, and the data is saved in AL as a 16-bit data item. Also, data in AL is handled in either word lengths or byte lengths.

If an arithmetic operation instruction of byte processing is executed in AL, the upper 8 bits in AL before the operation are ignored, and the upper 8 bits of operation results are reset to zero. Resetting an accumulator (A) does not initialized it, and it has an undefined value after the reset.

Figure 2-9 shows 32-bit data transfer processing, and Figure 2-10 shows AL-AH transfer processing.

Figure 2-9. 32-Bit Data Transfer

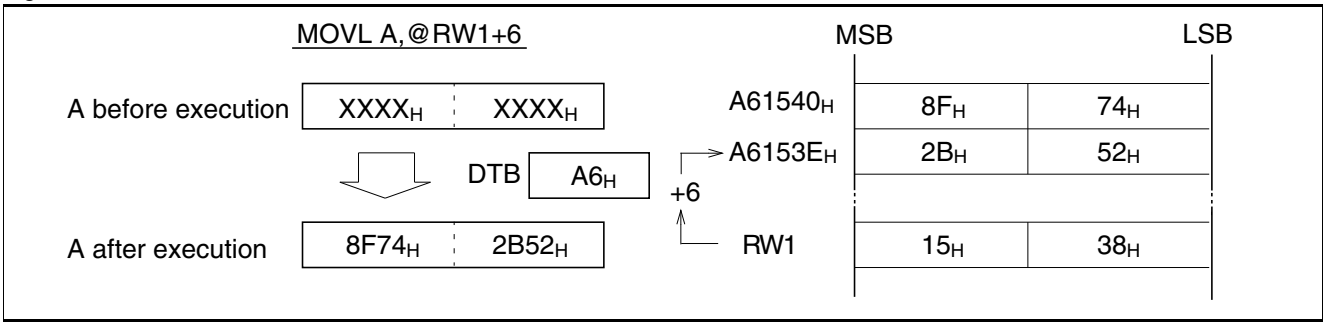
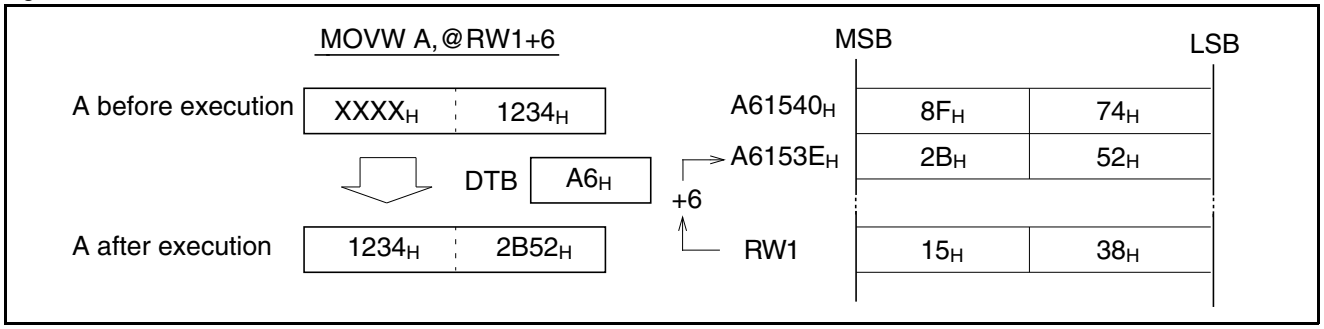


Figure 2-10. AL-AH Transfer



2.3.2 User Stack Pointer (USP) and System Stack Pointer (SSP)

This section explains the functions of the user stack pointer (USP) and system stack pointer (SSP).

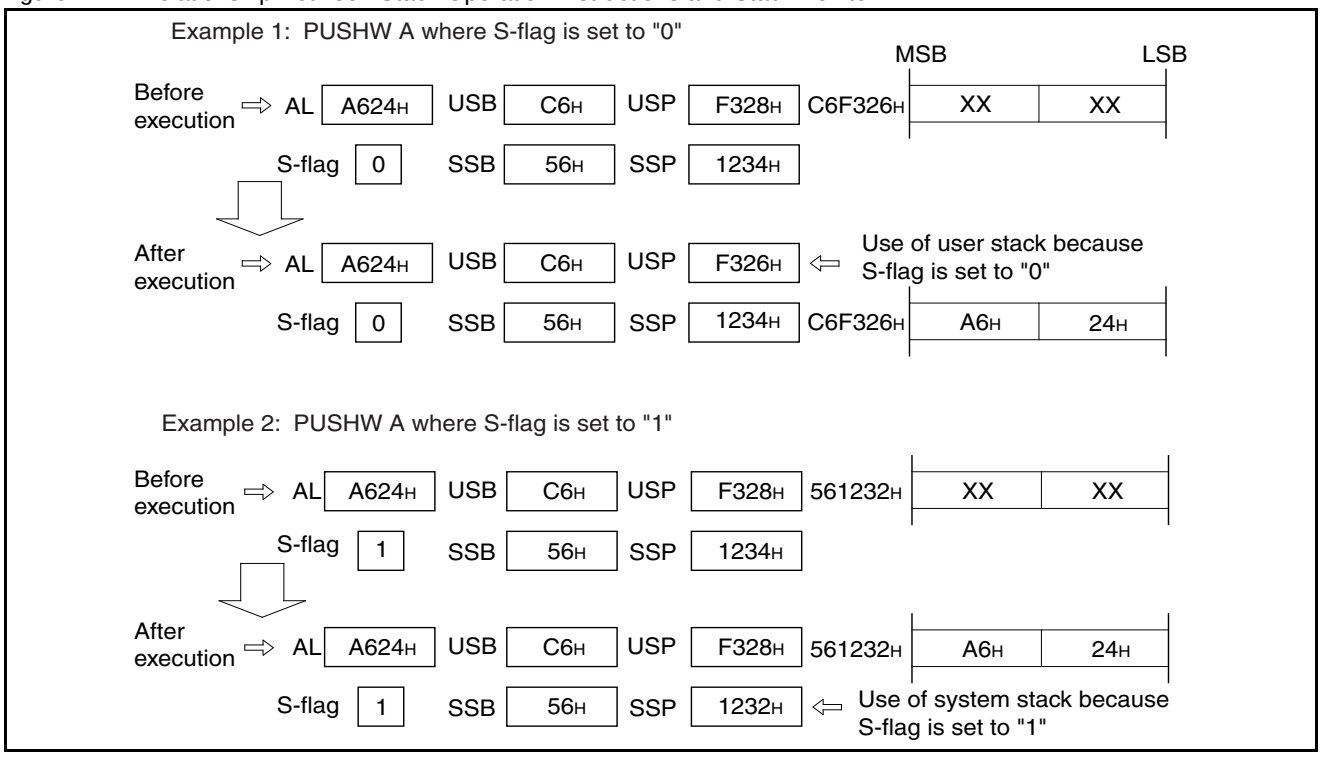
User Stack Pointer (USP) and System Stack Pointer (SSP)

The user stack pointer (USP) and system stack pointer (SSP) are 16-bit registers indicating the push/pop instruction or the memory address to which data is saved or restored at subroutine execution. The USP register and SSP register are used in stack-type instructions. If the S-flag in the processor status register is set to “0”, the USP register is enabled. If the S-flag is set to "1", SSP register is enabled (see Figure 2-11). If an interrupt is accepted, the S-flag is set and then the register value is saved in the memory area indicated by SSP in interrupt processing. SSP is used to execute stack processing of interrupt routines, and USP is used to execute stack processing other than interrupt routines. Only SSP is used if stack space is not divided.

In stack processing, the address of upper 8 bits is indicated with SSP → SSB and USP → USB. Resetting USP and SSP does not initialize them, but each then has an undefined value.

Figure 2-11 shows the relationship between stack operation instructions and the stack pointer where the S-flag is set to “0” and “1”.

Figure 2-11. Relationship Between Stack Operation Instructions and Stack Pointer



Note:

Use an even-numbered address for a stack pointer, in principle.

2.3.3 Processor Status (PS)

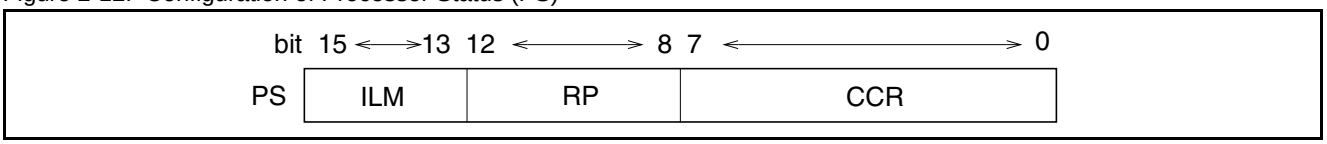
This section explains the functions of processor status (PS).

Processor Status (PS)

Processor status (PS) consists of bits used to execute CPU operations and bits indicating the CPU state. As shown in Figure 2-12, the upper byte in the PS register consists of a register bank pointer (RP) and interrupt level mask register (ILM). RP indicates the header address of a register bank. The lower byte of PS register is the condition code register (CCR) that includes a flag that is set and reset depending on execution results or interrupt events.

Figure 2-12 shows the configuration of processor status (PS).

Figure 2-12. Configuration of Processor Status (PS)



Condition Code Register (CCR)

Figure 2-13 shows the configuration of the condition code register.

Figure 2-13. Configuration of Condition Code Register

bit	7	6	5	4	3	2	1	0	
	-	I	S	T	N	Z	V	C	
Initial value	-	0	1	X	X	X	X	X	X: Undefined

I: Interrupt permission flag

An interrupt other than software interrupt is permitted if the I-flag is set to “1” and disabled if set to “0”.

The I-flag is cleared if reset.

S: Stack flag

If the S-flag is set to “0”, USP is enabled as the stack operation pointer, and if it is set to “1”, SSP is enabled. The S-flag is set if an interrupt or reset occurs.

T: Sticky bit flag

If there is one or more “1” in the data shifted out by a carry operation after the logic right shift instruction or arithmetic right shift instruction, the T-flag is set to “1”. Otherwise, it is set to “0”. If the shift amount is zero, it is set to “0”.

N: Negative flag

If MSB in operation results indicates “1”, the N-flag is set. Otherwise, it is cleared.

Z: Zero flag

If all operation results indicate “0”, the Z-flag is set. Otherwise, it is cleared.

V: Overflow flag

If an overflow with a signed value occurs as an operation execution result, the V-flag is set. Otherwise, it is cleared.

C: Carry flag

If a shift-in/shift-out operation occurs from MSB as operation execution results, the C-flag is set. Otherwise, it is cleared.

Register Bank Pointer (RP)

The register bank pointer (RP) shows the relationship between the F^{2}MC-16LX general-purpose register and internal RAM addresses. RP indicates the header memory address in the currently used register bank with the conversion formula [00180_H + RP x 10_H].

RP consists of 5 bits, with an address ranging from 00_H to $1F_H$.

A register bank can be allocated to a memory address in a range of 000180_H to $00037F_H$. Even in this range, however, a register bank cannot be used as a general-purpose register if a register bank is not an internal RAM. An instruction transfers an immediate value of 8 bits to RP, but only the lower 5 bits are actually used.

Figure 2-14. Configuration of Register Bank Pointer (RP)

	B4	B3	B2	B1	B0
Initial value	0	0	0	0	0

Interrupt Level Mask Register (ILM)

The interrupt level mask register (ILM) consists of 3 bits indicating the level of the CPU interrupt mask. Only interrupt request of an interrupt level higher than that represented with the 3 bits is accepted. The highest level is indicated with "0", the lowest level is indicated with “7” (see Table 2-2). Thus, to accept an interrupt, its level must be lower than the current ILM value. If an interrupt is accepted, its interrupt level value is set to ILM, and then any interrupts with the same or lower level of the interrupt priority are not accepted. ILM is initialized to zero by a reset. An instruction can transfer an 8-bit immediate value to the ILM register, but only the lower 3 bits are actually used.

Figure 2-15 shows the configuration of the interrupt level mask register. Table 2-2 shows the level indicated in the interrupt level mask register (ILM).

Figure 2-15. Configuration of Interrupt Level Mask Register

	ILM2	ILM1	ILM0
Initial value	0	0	0

Table 2-2. Level Indicated by Interrupt Level Mask Register (ILM)

ILM2	ILM1	ILM0	Level value	Permitted interrupt level
0	0	0	0	Interrupt prohibited
0	0	1	1	"0" only
0	1	0	2	Level value less than 1
0	1	1	3	Level value less than 2
1	0	0	4	Level value less than 3
1	0	1	5	Level value less than 4
1	1	0	6	Level value less than 5
1	1	1	7	Level value less than 6

2.3.4 Program Counter (PC)

This section explains the functions of program counter (PC).

Program Counter (PC)

PC is a 16-bit counter indicating the lower 16 bits in the memory address of an instruction code to be executed by CPU. An upper 8-bit address is indicated with the program counter bank register (PCB). PC contents are updated by condition branch instructions, subroutine call instructions, interrupts, or resets. It may also be used as a base pointer for operand access.

Figure 2-16 explains the functions of program counter (PC).

Figure 2-16. Functions of Program Counter (PC)



2.3.5 Program Counter Bank Register (PCB)

This section explains the functions of program counter bank register (PCB).

Program Counter Bank Register (PCB) <Initial Value: Value in Reset Vector>

The program counter bank register (PCB) consists of the following registers:

- Data bank register (DTB) < Initial value: 00_H >
- User stack bank register (USB) < Initial value: 00_H >
- System stack bank register (SSB) < Initial value: 00_H >
- Additional data bank register (ADB) < Initial value: 00_H >

Each bank register indicates memory banks to which PC, DT, SP (user), SP (system), and AD space are allocated.

All bank registers has a length of 1 byte. PCB is initialized to "00_H" by a reset. Bank registers can only be read.

PCB is updated when the JMPP, CALLP, RETP, RETI, or RETF instruction that branches to an entire 16M bytes space is executed. PCB is also updated when an interrupt occurs. For information on the operation of each register, see Section "2.2 Memory Space".

2.3.6 Direct Page Register (DPR)

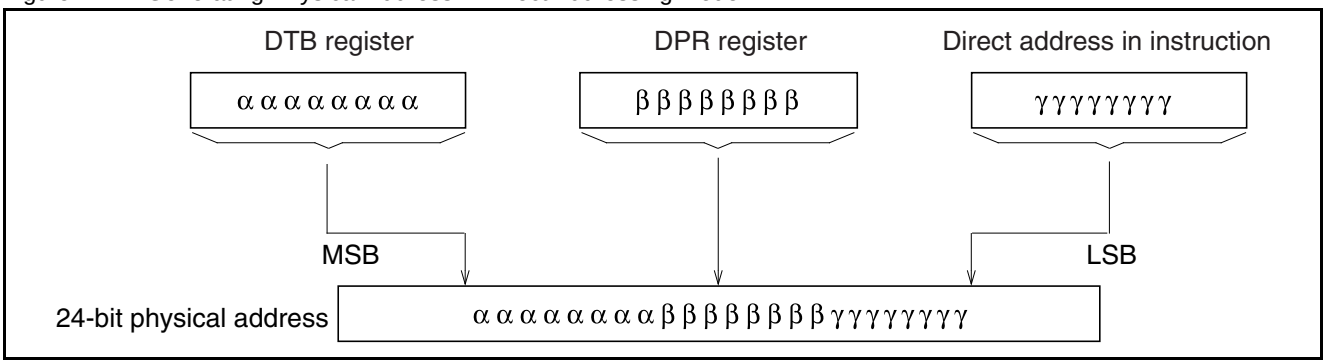
This section explains the functions of direct page register (DPR).

Direct Page Register (DPR) <Initial Value: 01_H>

The direct page register (DPR) specifies, as shown in Figure 2-17, addresses 8 to 15 of an instruction operand in the direct addressing mode. DPR has a length of 8 bits, and is initialized to 01_H by a reset. It also allows reading and writing by instructions.

Figure 2-17 illustrates the generation of a physical address in the direct addressing mode.

Figure 2-17. Generating Physical Address in Direct Addressing Mode



2.3.7 General-Purpose Register (Register Bank)

This section explains the functions of general-purpose register (register bank).

General-Purpose Register (Register Bank)

A register bank consists of 8 words and is used as a general-purpose register for arithmetic operation in the byte register (R0 to R7), word register (RW0 to RW7), and long-word register (RL0 to RL3). A register bank is also used as an instruction pointer. Table 2-3 lists the register functions and Table 2-4 shows the relationship among registers.

Register bank values are not initialized by a reset, the same as for RAM area, but the state before resetting is kept. At power-on, however, the values are undefined.

Table 2-3. Register Functions

R0 to R7	Used as operand in different instructions Note: R0 is used as the barrel shift counter or normalization instruction counter.
RW0 to RW7	Used as a pointer or operand in different instructions Note: RW0 is used as a string instruction counter.
RL0 to RL3	Used as a long pointer or operand in different instructions

Table 2-4. Relationship Among Registers

	RW0	RL0
	RW1	
	RW2	RL1
	RW3	
R0	RW4	RL2
R1		
R2	RW5	
R3		
R4	RW6	RL3
R5		
R6	RW7	
R7		

2.4 Prefix Codes

By inserting a prefix code before an instruction, part of an instruction operation may change. Three types of prefix codes are provided: bank select prefixes, common register bank prefixes, and flag change suppress prefixes.

Bank Select Prefix (PCB, DTB, ADB, SPB)

Memory space used in data access is determined according to the addressing mode.

By inserting a bank select prefix before an instruction, the instruction selects the memory space used for data access regardless of the addressing mode in use.

Table 2-5 shows the relationship between the bank select prefix and a selected space.

Table 2-5. Bank Select Prefix

Bank select prefix	Selected space
PCB	PC space
DTB	Data space
ADB	AD space
SPB	Either SSB or USB space is used depending on the stack flag value.

Be careful when using the following instructions:

String instruction (MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW)

The bank registers specified with an operand is used regardless of a prefix.

Stack operation instruction (PUSHW, POPW)

SSB or USB is used depending on the S-flag, regardless of a prefix.

I/O access instruction

MOVA A, io/MOV io, A/MOVX A, io/MOVW A, io/MOVW io, A

MOV io, #imm8/MOVW io, #imm8/MOBV A, io: bp/MOVB io: bp, A

SETB io: bp/CLRB io: bp/BBC io:bp, rel/BBS io:bp, rel WBTC

WBTS

The I/O space of a bank is used regardless of whether a prefix is in an instruction.

Flag change instruction (AND CCR,#imm8, OR CCR,#imm8)

An instruction operation is normal, but a prefix affects the next instruction.

POPW PS

Regardless of prefix, SSB or USB is used depending on the S-flag. A prefix affects the next instruction.

MOV ILM, #imm8

An instruction operation is normal, but a prefix affects the next instruction.

RETI

SSB is used regardless of prefix.

Common Register Bank Prefix (CMR)

To facilitate data exchange between multiple tasks, the same register bank needs to be easily accessed regardless of each register bank pointer (RP) value. If CMR is inserted before an instruction that accesses a register bank, the instruction accesses the common bank with addresses ranging from 000180_H to $00018F_H$ (register bank selected if RP = 0) regardless of the current RP value. However, be careful when using the following instructions:

String instruction (MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW)

If an interrupt request occurs while a string instruction with a prefix code is executed, the prefix code is disabled when the string instruction is returned after the interrupt is processed. Thus, with interrupt processing, the string instruction causes an error. Do not add a CMR prefix to the above string instruction.

Flag change instruction (AND CCR, #imm8, OR CCR, #imm8)

An instruction operation is normal, but a prefix affects the next instruction.

MOV ILM, #imm8

An instruction operation is normal, but a prefix affects the next instruction.

Flag Change Suppress Prefix (NCC)

To suppress a flag change, specify the flag change suppress prefix code (NCC). By inserting NCC before an instruction, the flag change caused by an instruction is suppressed. However, be careful when using the following instructions:

String instruction (MOVS, MOVSW, SCEQ, SCWEQ, FILS, FILSW)

If an interrupt request occurs while a string instruction with prefix code is executed, the prefix code is disabled when the string instruction is returned after the interrupt is processed. Thus, with interrupt processing, the string instruction causes an error. Do not add a CMR prefix to the above string instruction.

Flag change instruction (AND CCR, #imm8, OR CCR, #imm8)

An instruction operation is normal, but a prefix affects the next instruction.

Interrupt instruction (INT #vct8, INT9, INT addr16, INTO addr24, POPW PS)

CCR changes according to the instruction specification regardless of the prefix.

JCTX@A

CCR change according to the instruction specification regardless of the prefix.

MOV ILM, #imm8

An instruction operation is normal, but a prefix affects the next instruction.

Interrupt Suppress Instruction

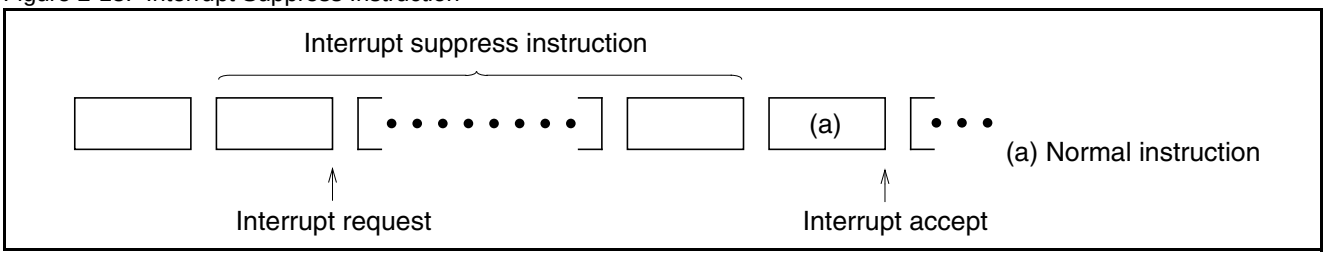
No interrupt requests are accepted on ten types of instruction as follows.

MOV ILM, #imm8/PCB/SPB/OR CCR, #imm8/NCC

AND CCR, #imm8/ADB/CMR/POPW PS/DTB

When an effective interrupt request is issued while any of above instructions is executed, an interrupt may be processed only if instructions other than the above are executed. For more information, see Figure 2-18.

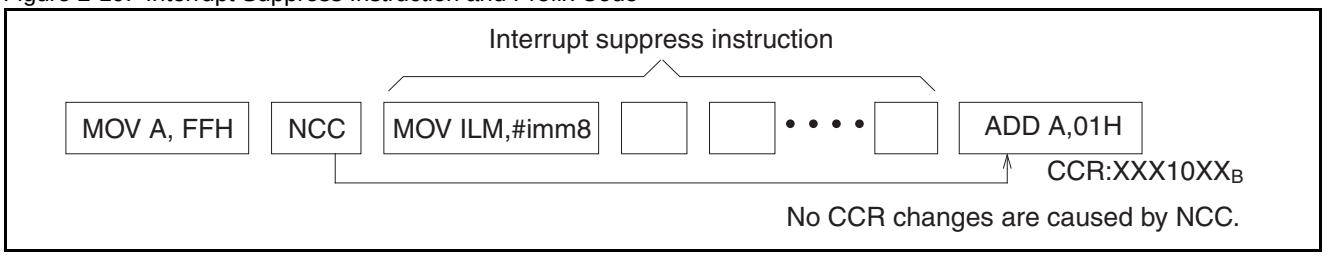
Figure 2-18. Interrupt Suppress Instruction



Restrictions on Interrupt Suppress Instruction and Prefix Instruction

If a prefix code is inserted before an interrupt is suppressed, the prefix code affects up to the first instruction that appears after any code other than interrupt suppress instructions, as shown in Figure 2-19.

Figure 2-19. Interrupt Suppress Instruction and Prefix Code

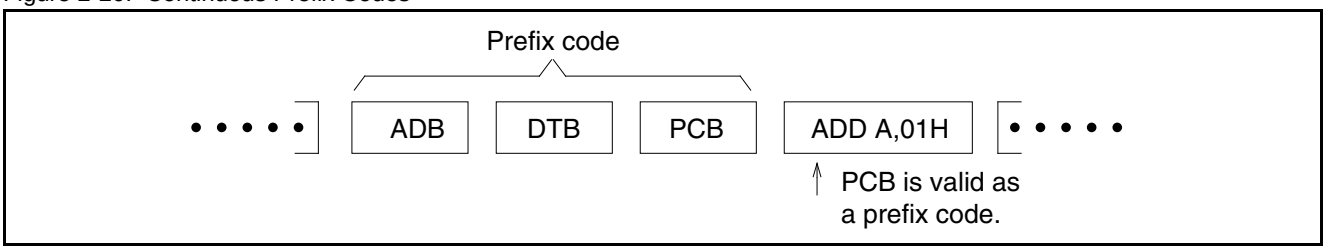


Continuous Prefix Codes

If continuous prefix codes conflict, the latest ones are valid, as shown in Figure 2-20.

Such conflicting prefix codes mean PCB, ADB, DTB, and SPB, as shown in Figure 2-20.

Figure 2-20. Continuous Prefix Codes



3. Interrupt

This chapter explains the functions and operation of the interrupt, extended intelligent I/O service (EI^{2}OS), and DMA in MB90880 series.

3.1 Overview of Interrupt

3.2 Interrupt Vectors

3.3 Interrupt Control Registers (ICR00 to ICR15)

3.4 Interrupt Flow

3.5 Hardware Interrupt

3.6 Software Interrupt

3.7 Extended Intelligent I/O Services (EI^{2}OS)

3.8 Operational Flow and Procedure of Extended Intelligent I/O Services (EI^{2}OS)

3.9 μDMA Controller (μDMAC)

3.10 μDMAC Register

3.11 DMA Descriptor Window Register (DDWR)

3.12 μDMAC Operation

3.13 Exceptions

3.1 Overview of Interrupt

F^{2}MC-16LX has an interrupt function that aborts the running process according to such as an event to forward the control to another specified program. This interrupt function will be divided into five parts as follows.

- Hardware Interrupt:Interrupt process by an event of the embedded source.
- Software Interrupt:Interrupt process by a software event generation instruction.
- Extended Intelligent I/O Services (EI^{2}OS):Forward service by an event of the embedded resource.
- μDMAC: Data forwarding service by an event of the embedded resource.
- Exceptions: Abort service by exceptional operation.

Hardware Interrupt

Hardware interrupt is booted by an interrupt request from internal resource. Hardware interrupt request is generated when both the interrupt request flag and the interrupt enable flag in the internal resource are set. Therefore, there should be an interrupt request flag and an interrupt enable flag in the internal resource.

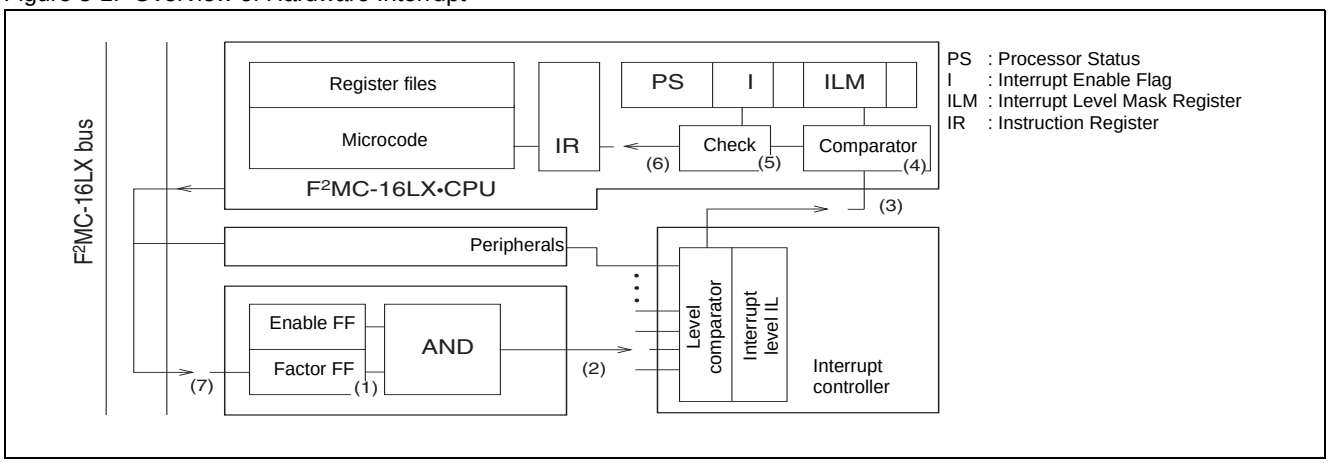
Interrupt Level Assignment

Hardware interrupt can specify the interrupt level. Level setting bits (IL0, IL1, IL2) for the interrupt controller to set the interrupt level.

Hardware Interrupt Request Mask

Hardware interrupt request can be masked by using the I-flag of the processor status (PS) register in the CPU and ILM bits (ILM0, ILM1, ILM2). CPU saves 12 bytes data that consist of registers PS, PC, PCB, DTB, ADB, DPR and A to the memory area directed by SSB and SSP registers, when an interrupt request is generated which is not masked.

Figure 3-1. Overview of Hardware Interrupt



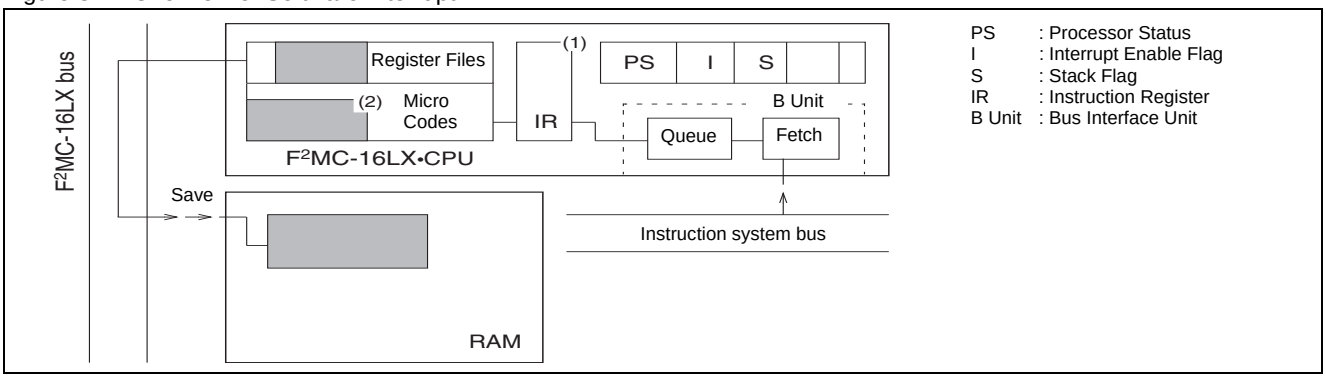
Software Interrupt

Software interrupt is a function of CPU that forwards the control from the running program to the interrupt process program that user defined.

Software interrupt is requested by the INT instruction execution. The interrupt that is requested by the INT instruction does not consist of interrupt request flag or interrupt enable flag. Execution of INT instruction generates interrupt request at any time.

INT instruction does not have even any assignment for interrupt level. This does not update the ILM on INT instructions. Instead of it, clearing I-flag suspends the continuous interrupt request.

Figure 3-2. Overview of Software Interrupt



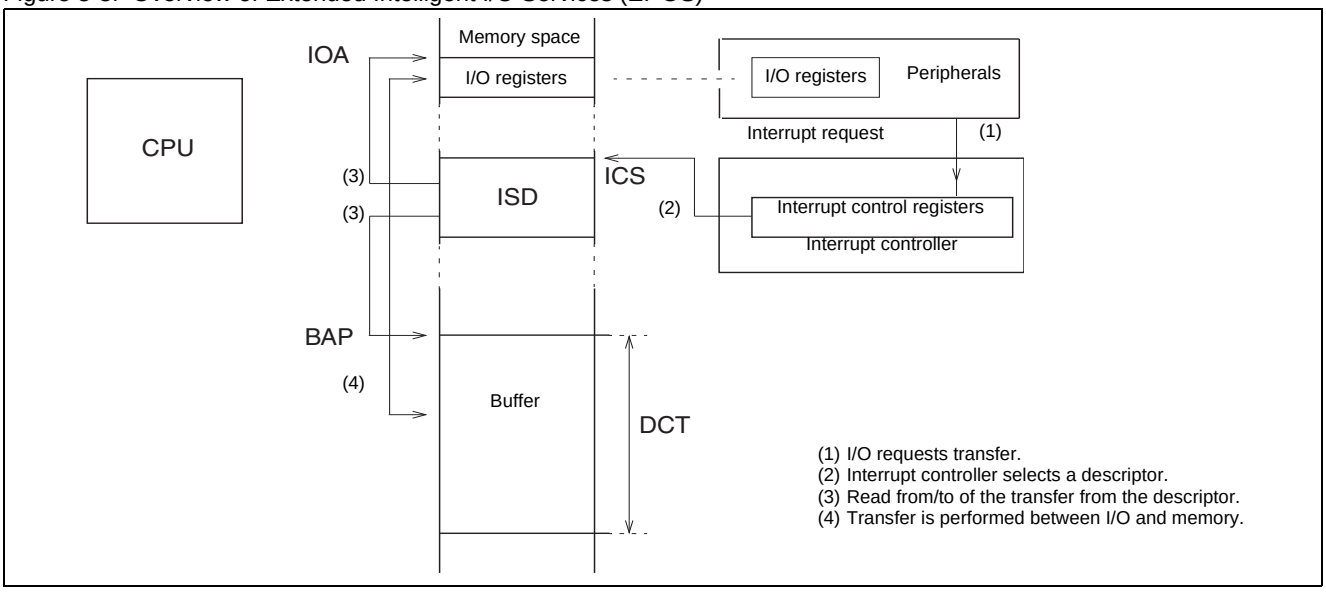
Extended Intelligent I/O Services (EI^{2}OS)

Extended intelligent I/O services are transferred data between the internal resource and the memory automatically. In this kind of processes, interrupt process program has been used traditionally, but EI^{2}OS is able to execute the data transfer like DMA (Direct Memory Access).

There should be an extended intelligent I/O service enable bit (ISE) in the interrupt control register (ICR) of the interrupt controller to activate the extended intelligent I/O service function from the internal resource.

The extended intelligent I/O service is booted when the ISE flag is set to “1” and an interrupt request is occurred. Set the ISE flag to “0” in order to generate the usual interrupt by the hardware interrupt request.

Figure 3-3. Overview of Extended Intelligent I/O Services (EI^{2}OS)

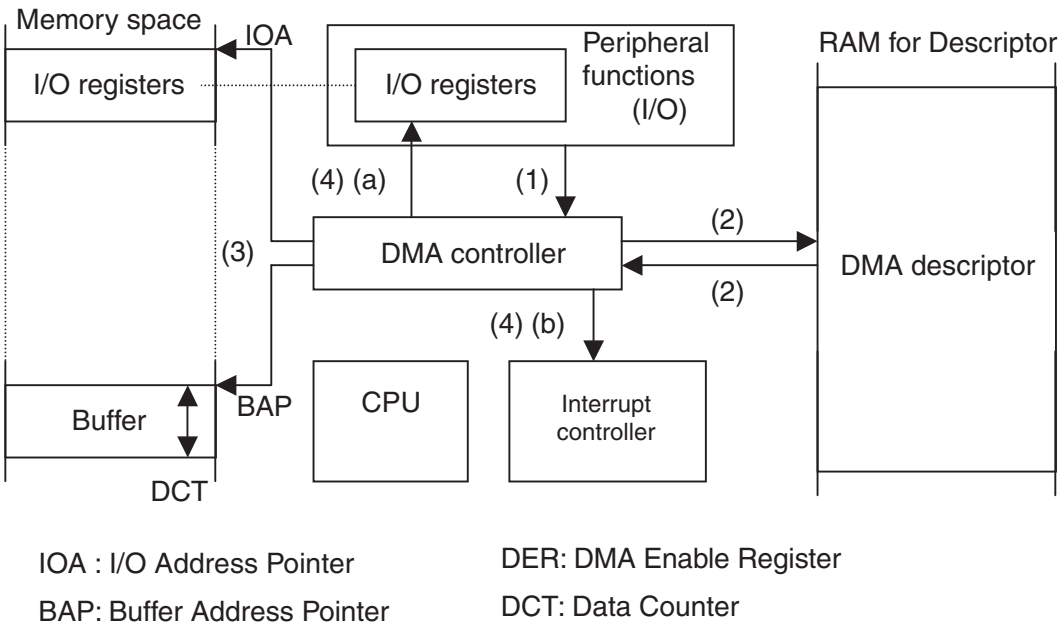


Direct Memory Access (DMA)

µDMA is a function that transfers data between peripheral functions and memory automatically. µDMA acts like a direct memory access (DMA) for the data transfer which has been processed by interrupt process program traditionally. µDMA executes interrupt process program automatically after given number of the data transfer is terminated.

The interrupt of µDMA is one of hardware interrupts.

Figure 3-4. Overview of Direct Memory Access (DMA)



(1) Peripheral resource (I/O) requests DMA transfer.
(2) If target bit of DMA enable register (DER) is "1", DMAC controller reads the transfer data (from/to address and transfer channel) from the descriptor.
(3) DMA transfer is performed between I/O and memory space.
(4) After an item (one of byte data or word data) is transferred:
  (a) The transfer is not completed (not DCT = 0): Requests peripherals to clear the DMA transfer requests.
  (b) The transfer is completed (DCT = 0): Sets a transfer completion flag to the DMA status register after the DMA transfer is completed, and generates an interrupt request to interrupt controller.

Note: Use read-modify-write (RMW) instructions if writing is conducted to the internal registers (DSRH, DSRL, DSSR, DERH and DERL).

Exceptions

Exceptional handling is basically the same as an interrupt and is executed by aborting the general processes at the time that an exception is detected between the instructions. Generally, exceptional handling is generated as a result of exceptional operation so that this handling shall be used to boot recovery software on debugging or in emergency.

3.2 Interrupt Vectors

Interrupt vectors use the same area as that hardware interrupt and software interrupt uses. For example, interrupt request number INT 42 is used for delayed hardware interrupt and software interrupt INT #42. Therefore, both delayed interrupt and INT #42 call same interrupt process routine. Interrupt vectors are assigned between the address $FFFC00_H$ and $FFFFFF_H$ shown in Table 3-1.

Interrupt Vectors

Table 3-1. Interrupt Vectors

Interrupt Request	Interrupt Factor	Interrupt Control Registers		Lower Vector Address	Middle Vector Address	Upper Vector Address	Mode Registers
		Number	Address				
INT 0 *	--	--	--	$FFFFFC_H$	$FFFFFD_H$	$FFFFFE_H$	Not Used
INT 1 *	--	--	--	$FFFFF8_H$	$FFFFF9_H$	$FFFFFA_H$	Not Used
. . .	--	--	--	. . .	. . .	. . .	. . .
INT 7*	--	--	--	$FFFFE0_H$	$FFFFE1_H$	$FFFFE2_H$	Not Used
INT 8	Reset	--	--	$FFFFDC_H$	$FFFFDD_H$	$FFFFDE_H$	$FFFFDF_H$
INT 9	INT9 instruction	--	--	$FFFFD8_H$	$FFFFD9_H$	$FFFFDA_H$	Not Used
INT 10	Exceptional Handling	--	--	$FFFFD4_H$	$FFFFD5_H$	$FFFFD6_H$	Not Used
INT 11	Hardware Interrupt #0	ICR00	$0000B0_H$	$FFFFD0_H$	$FFFFD1_H$	$FFFFD2_H$	Not Used
INT 12	Hardware Interrupt #1			$FFFFCC_H$	$FFFFCD_H$	$FFFFCE_H$	Not Used
INT 13	Hardware Interrupt #2	ICR01	$0000B1_H$	$FFFFC8_H$	$FFFFC9_H$	$FFFFCA_H$	Not Used
INT 14	Hardware Interrupt #3			$FFFFC4_H$	$FFFFC5_H$	$FFFFC6_H$	Not Used
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .
INT 41	Hardware Interrupt #30	ICR15	$0000BF_H$	$FFFF58_H$	$FFFF59_H$	$FFFF5A_H$	Not Used
INT 42	Hardware Interrupt #31			$FFFF54_H$	$FFFF55_H$	$FFFF56_H$	Not Used
INT 43	--	--	--	$FFFF50_H$	$FFFFF51_H$	$FFFF52_H$	Not Used
. . .	--	--	--	. . .	. . .	. . .	. . .
INT 254	--	--	--	$FFFC04_H$	$FFFC05_H$	$FFFC06_H$	Not Used
INT 255	--	--	--	$FFFC00_H$	$FFFC01_H$	$FFFC02_H$	Not Used

*:If PCB has an address FF_H, the vector area for CALLV instruction duplicates the vector area for INT #vct8 (#0 to #7). Pay attention if CALLV instruction is used.

Interrupt Factors, Interrupt Vectors and Interrupt Control Registers

Refer to Table A-3 in "A.3 Interrupt Source, Interrupt Vector, and Interrupt Control Register".

3.3 Interrupt Control Registers (ICR00 to ICR15)

Interrupt control registers are employed in the Interrupt controller and in all I/O's that have interrupt function. These registers have three of functions as follows.

- Interrupt level setting for the peripherals that apply.
- Selection of the general interrupts for the peripherals that apply or the extended intelligent I/O service.
- Selection of extended intelligent I/O services channel

Avoid accessing to these registers by read-modify-write (RMW) instruction to cause malfunction.

Interrupt Control Registers (ICR00 to ICR15)

Bit mapping of the interrupt control registers is shown in the Figure 3-5.

Figure 3-5. Interrupt Control Registers (ICR00 to ICR15)

On resetting the Interrupt Control Registers

bit 15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0	
ICS3	ICS2	ICS1 or S1	ICS0 or S0	ISE	IL2	IL1	IL0	00000111_B
W	W	*	*	R/W	R/W	R/W	R/W	

*:ICS1 and ICS0 are valid only when write, and S1 and S0 are valid only when read.

Note:

ICS3 to ICS0 are valid only when EI^{2}OS is activated. Set “1” to ISE to activate EI^{2}OS and set “0” not to activate EI^{2}OS. Any value can be set to ICS3 to ICS0 if EI^{2}OS will not be activated.

[bit10 to bit8, bit2 to bit0] IL0, IL1, IL2 (interrupt level setting bits)

These are interrupt level setting bits. These bits specify interrupt level of internal resource that applies. These bits can be read from and written to. These are initialized to level 7 (no interrupt) by reset.

The relation between the interrupt level setting bits and interrupt level is shown in the Table 3-2.

Table 3-2. Interrupt Level Setting Bits and Interrupt Level

IL2	IL1	IL0	Interrupt Level Value
0	0	0	0 (Highest Priority Interrupt)
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6 (Lowest Priority Interrupt)
1	1	1	7 (No Interrupt)

[bit11, bit3] ISE (extended intelligent I/O service enable bit)

It is an EI^{2}OS enable bit. EI^{2}OS will be booted if this bit is “1” interrupt request occurs and interrupt sequence will be booted if this bit is “0”, when interrupt request occurs. The ISE bit becomes “0” when EI^{2}OS is completed (in case of count completion and a request from embedded resource). This bit should be set to “0” by software if there is no applicable EI^{2}OS function in the embedded resource. This bit can be read from and written to.

This bit is initialized to “0” by reset.

[bit15 to bit12, bit7 to bit4] ICS3 to ICS0 (extended intelligent I/O services channel selection bits)

These are an EI^{2}OS channel selection bits. These bits are write only bits and select EI^{2}OS channel. Set value by these bits determines the address of the extended intelligent I/O service descriptor on the memory. Bits ICS3 to ICS0 are initialized to “0000_B” by reset.

Relationship between ICS bit, channel number and descriptor address is shown in the Table 3-3.

Table 3-3. ICS Bit, Channel Number and Descriptor Address

ICS3	ICS2	ICS1	ICS0	Selected Channel	Descriptor Address
0	0	0	0	0	000100_H
0	0	0	1	1	000108_H
0	0	1	0	2	000110_H
0	0	1	1	3	000118_H
0	1	0	0	4	000120_H
0	1	0	1	5	000128_H
0	1	1	0	6	000130_H
0	1	1	1	7	000138_H
1	0	0	0	8	000140_H
1	0	0	1	9	000148_H
1	0	1	0	10	000150_H
1	0	1	1	11	000158_H
1	1	0	0	12	000160_H
1	1	0	1	13	000168_H
1	1	1	0	14	000170_H
1	1	1	1	15	000178_H

[bit13, bit12, bit5, bit4] S0, S1 (extended intelligent I/O service status)

These are EI^{2}OS completion status bits. These bits are read only bits and will determine the status of the completion when EI^{2}OS is completed.

These bits are initialized to "00_B" by reset.

Relationship between S bit and completion status is shown in the Table 3-4.

Table 3-4. S Bit and Completion Status

S1	S0	Completion Status
0	0	When EI^{2}OS is running or not booting
0	1	Pause condition by count completion
1	0	Reserve
1	1	Pause condition by a request from embedded resource

3.4 Interrupt Flow

Interrupt flow is shown in the Figure 3-6.

Interrupt Flow

Figure 3-6. Interrupt Flow

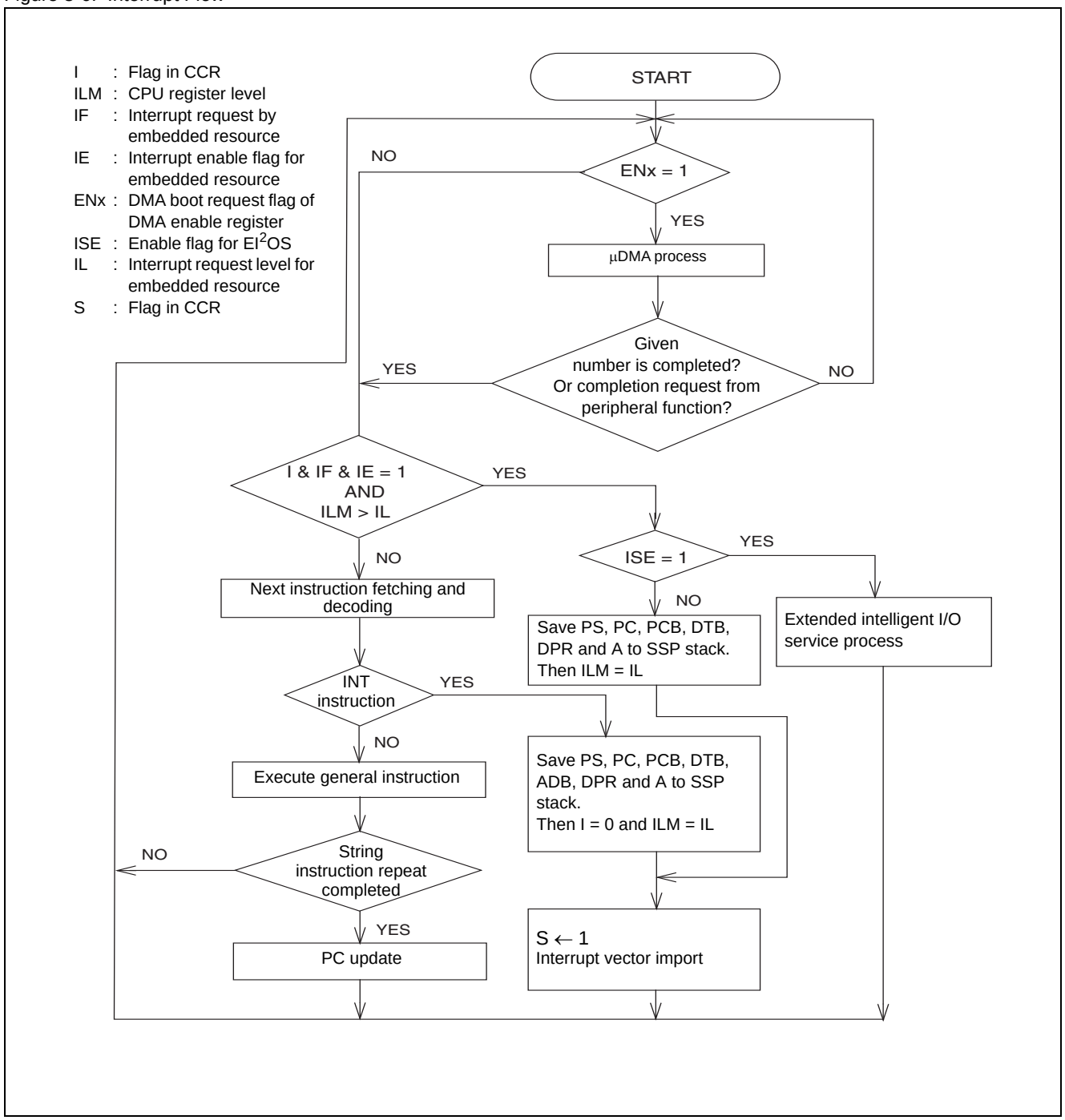
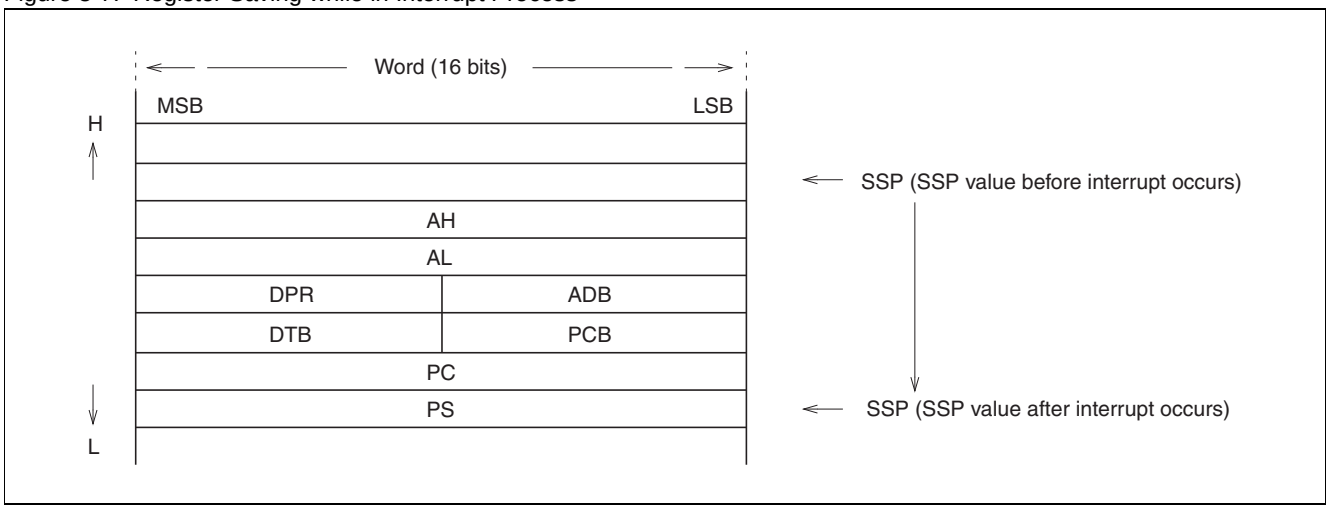


Figure 3-7. Register Saving while in Interrupt Process



3.5 Hardware Interrupt

Hardware interrupt is a function of CPU that halts the running program in response to a interrupt request signal from embedded resource, and forwards the control from the running program to the interrupt process program that user defined.

Hardware Interrupt

A hardware interrupt occurs when the relevant conditions are satisfied as a result of two operations:
comparison between the interrupt request level and the value in the interrupt level mask register (ILM) of PS in the CPU, and hardware reference to the I-flag value of PS.

The CPU performs the following processing when a hardware interrupt occurs:

- Saves the values in the PC, PS, AH, AL, PCB, DTB, ADB, and DPR registers of the CPU to the system stack.
- Set the ILM in the PS registers. The registers become the same interrupt level as of that has been requested.
- Import the content of corresponding interrupt vector and branch to the vector.

Construction of Hardware Interrupt

Construction of hardware interrupt is divided by following three parts.

Embedded Resource

Interrupt enable bit, Interrupt request bit: Control of the interrupt request from the resources

Interrupt Controller

ICR: Interrupt level assignment, Priority determination of simultaneous request interrupt

CPU

I, ILM: Compares request interrupt level with current level and determines the status of the interrupt enable.

Microcode: Steps for interrupt process

Each feature appears as a content of: resource control register for embedded resource, ICR for interrupt controller and CCR for CPU. Three of settings should be prepared by software prior to using hardware interrupt.

The interrupt vector table for referring to in case of interrupt process is allocated to the memory area from $FFFC00_H$ to $FFFFFF_H$ and shared with the software interrupt. The allocation is shown in the A.3 Interrupt Source, Interrupt Vector, and Interrupt Control Register “Table A-3”.

3.5.1 Hardware Interrupt Operation

The embedded resources that have a hardware interrupt request employ “Interrupt Request Flag” that indicates the existence of an interrupt request and “Interrupt Enable Flag” that selects the permission to the request to the CPU. Interrupt request flag will be set by a unique event of the embedded resource and the resource will generate interrupt request to interrupt controller if the interrupt enable flag is “enable”.

Hardware Interrupt Operation

Interrupt controller compares with the interrupt levels (IL), which have been received at the same time, in the ICR each other, and selects a highest level request (minimum IL value) to forward to CPU. The request with small interrupt number will be selected if there are multiple requests with same level. The relation between each interrupt request and each ICR is determined by the hardware.

CPU compares received interrupt level (IL) with ILM in the PS register, and if interrupt level (IL) < (ILM) and I bit in the PS register is “1”, CPU boots the microcode for interrupt processing after executing instruction is completed. After referring to the ISE bit in the ICR of the interrupt controller at the top of the microcode for interrupt process, CPU confirms the ISE bit is “0” (i.e. interrupt) to boot main interrupt process.

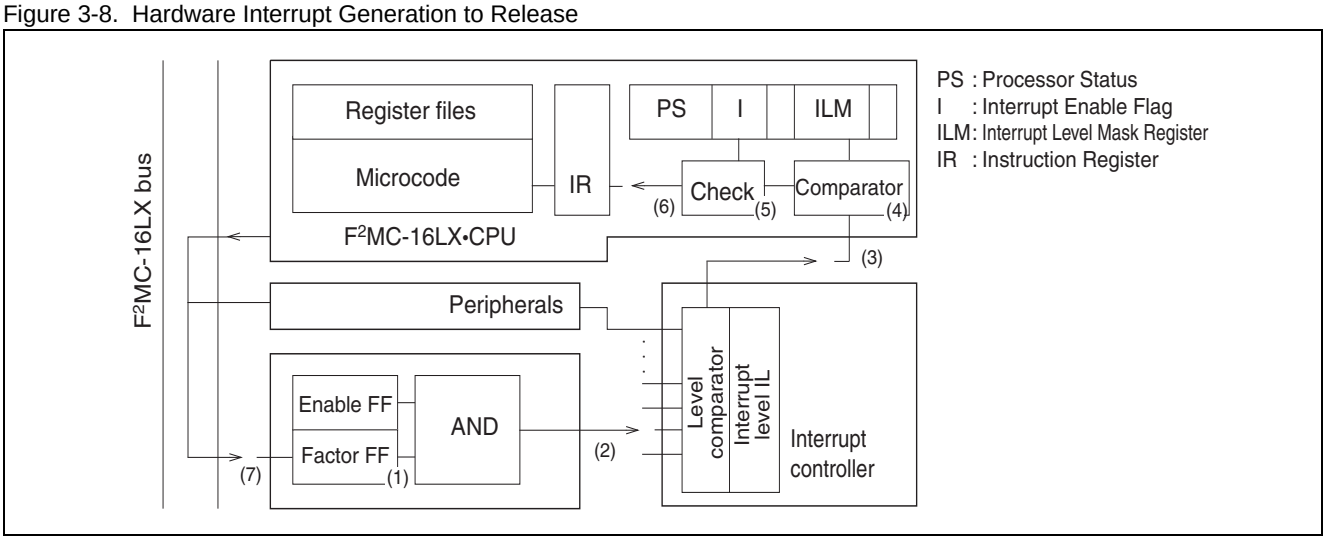
Main interrupt process saves 12 bytes of PS, PC, PCB, DTB, ADB, DPR and A to the memory that SSB and SSP directed, reads 3 bytes of interrupt vector and loads to PC and PCB, updates the ILM in the PC to received interrupt request level, and sets the S flag to “1” to perform branch process. In consequence, the instruction that will be executed next becomes the interrupt process program that user defined.

3.5.2 Hardware Interrupt Generation and Release

The flowchart, that describes the sequence from hardware interrupt generation until all the request in the interrupt process program is processed, is shown in Figure 3-8.

Hardware Interrupt Generation and Release

Figure 3-8. Hardware Interrupt Generation to Release



(1) Interrupt factor is generated in the peripherals.

(2) If interrupt enable bit in the peripheral is enabled, the peripheral generates an interrupt request to the interrupt controller.

(3) The interrupt controller that received the request forwards the applicable interrupt level to CPU upon the determination of the priority to the simultaneous requests.

(4) CPU compares the interrupt level requested by interrupt controller with ILM bit in the processor status register.

(5) Checks the content of I-flag in the same processor status register only if the result of the comparison had higher priority than current interrupt process level.

(6) Only if the result of checking (5) is that I-flag is the interrupt enable status, sets the content of ILM bit to requested level then performs the interrupt process after current executing instruction is terminated to forward the control to interrupt process routine.

(7) Interrupt request is terminated by clearing the interrupt factor that occurred in (1) at the software in the user's interrupt process routine.

The formula the CPU executes for calculating the interrupt process execution time at (6) and (7) is shown below. Refer to the Table 3-5 for the cycle number compensation value.

Interrupt boot : 24 + 6 × compensation value of cycle number

Interrupt return : 15 + 6 × compensation value of cycle number (RETI command)

Table 3-5. Cycle Number Compensation Value on Interrupt Process

Address that directed by stack pointer	Compensation value [cycle]
External 8 bits	+4
External even address	+1
External odd address	+4
Internal even address	0
Internal odd address	+2

3.5.3 Multiple Interrupt

In a special case, no hardware interrupt request is accepted while writing to the data input/output area. This case will happen because CPU avoids any malfunction related to interrupt request that is issued while rewriting interrupt control registers for each resource.

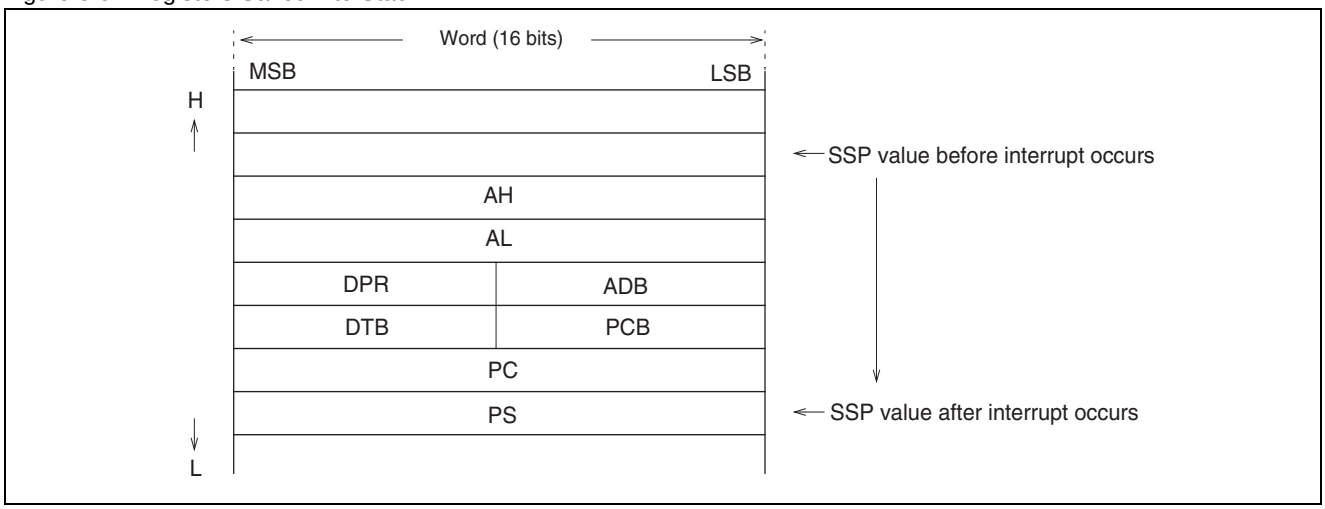
And the process with higher interrupt level will be overridden when an interrupt occurs while executing interrupt process.

Multiple Interrupt

F^{2}MC-16LX CPU supports multiple interrupt function. Therefore, if an interrupt occurs with higher level than that of currently executing interrupt process has, the control will move to the higher level after currently executing instruction is terminated. The control will be returned to the process that has been processed after interrupt with higher level is terminated. If an interrupt occurs with equal or lower level than that of currently executing interrupt process has, new interrupt request will be suspended until current interrupt process is terminated unless there is no change by content of the ILM or I-flag instruction. In addition, extended intelligent I/O service will not be booted multi-ply and interrupt requests or other extended intelligent I/O service request will be suspended while an extended intelligent I/O service is processing.

The sequence of the registers that will be saved to the stack is shown in Figure 3-9.

Figure 3-9. Registers Saved into Stack



3.6 Software Interrupt

Software interrupt is a function that CPU forwards the control from the currently executing program to the user defined interrupt process program, responding to a special instruction execution. Booting software interrupt is generally occurred by execution of the software interrupt instruction.

Software Interrupt

The CPU performs the following processing when a software interrupt occurs:

- Save the values in the PC, PS, AH, AL, PCB, DTB, ADB, and DPR registers of the CPU to the system stack.
- Sets I-flag of PS register to “0”. This prohibits interrupt automatically.
- Imports the content of the interrupt vector and branches to the place where the content indicates.

Interrupt request, executed by INT instruction as one of software interrupts, does not have an interrupt request flag or an enable flag and interrupt request generally occurs by executing INT instruction.

INT instruction does not have an interrupt level. This will not update ILM by INT instruction and suspends continuous interrupt requests by setting I-flag to “0”.

Construction of Software Interrupt

All the constructions related to software interrupts are employed in the CPU.

CPU..... Microcode: Steps for interrupt processing

Interrupt Vector List

Interrupt vector list for MB90880 series is shown Table C-1 in Appendix C.

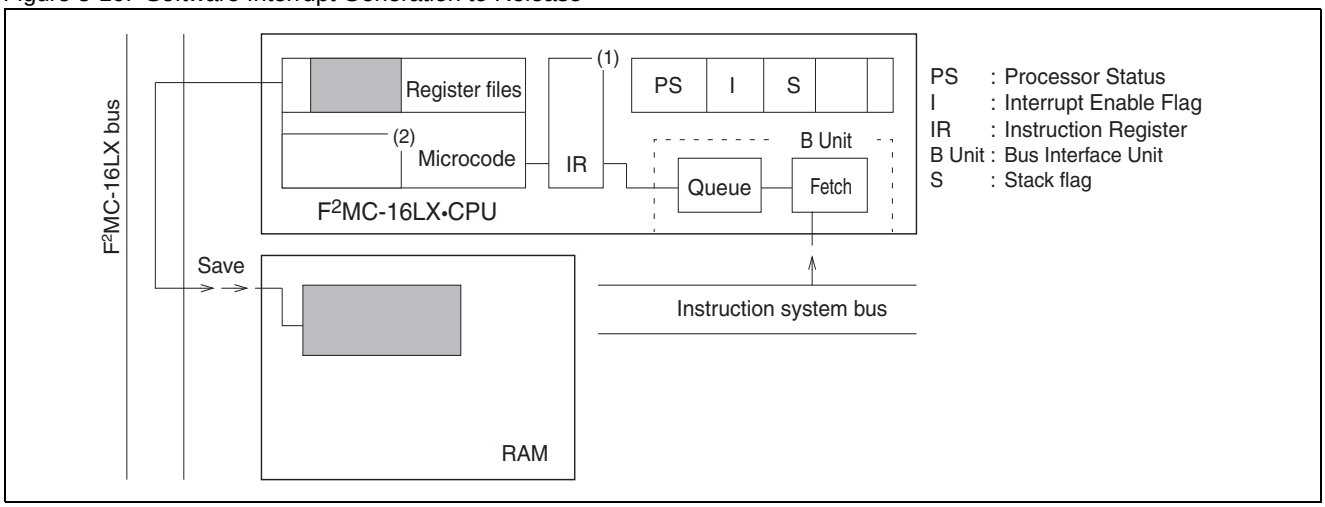
As shown in the Table, software interrupts share the same interrupt vector area with hardware interrupt. For example, interrupt number INT 12 is used as software interrupt INT#12 as well as hardware interrupt. Therefore, both hardware interrupt and INT#12 call same interrupt process routine.

Software Interrupt Operation

CPU executes importing software interrupt instruction to boot a microcode for processing software interrupt. Software interrupt process microcode saves 12 bytes of PS, PC, PCB, DTB, ADB, DPR and A to the memory directed by SSB and SSP, then reads 3 bytes of interrupt vector and save them to PC and PCB, and sets I-flag to “0” and S flag to “1” to process branch. This results the next instruction to be executed is user-defined interrupt process program.

The flow, from the occurrence of software interrupt to the empty of the interrupt request in the interrupt process program, is shown in the Figure 3-10.

Figure 3-10. Software Interrupt Generation to Release



(1). Software interrupt instruction will be executed.

(2). Dedicated registers in the CPU in the register file will be saved following microcode applicable to software interrupt instructions.

(3). Interrupt process at the RETI command in the user's interrupt process routine will be terminated.

Others

The vector area of CALLV instruction is duplicated with INT#vct8 instruction table when program bank register (PCB) is located at FF_H. Avoid using CALLV instruction and INT#vct8 instruction that can share the same address when coding software.

The relation between interrupt factors, interrupt vectors and interrupt control registers of MB90880 series is shown in Table A-3.

3.7 Extended Intelligent I/O Services (EI^{2}OS)

Extended intelligent I/O service (EI^{2}OS) is one of hardware interrupt operations and employs automatic data transfer function between I/O and memory. This achieves DMA-like process with I/O instead of interrupt process program traditionally.

Extended Intelligent I/O Services (EI^{2}OS)

There are advantages below comparing with the interrupt process method traditionally.

- Program size can be minimized as no description for transfer is required.
- Transfer speed is fast as saving register is not required without internal registers for transfer.
- Unnecessary data transfer does not occur as the transfer can be halted by I/O condition.
- No increment nor update mode for buffer address is selectable.
- No increment nor update mode for I/O register address is selectable (in case of buffer address update mode).

In addition, when EI^{2}OS is completed, user can determine the class of completion condition as branches to interrupt process routine automatically after the determination of completion condition.

In order to accomplish EI^{2}OS, hardware is distributed into two places and each block has following registers and descriptors.

Interrupt Control Register

Located in the interrupt controller and indicates the address for ISD.

Extended Intelligent I/O Service descriptors

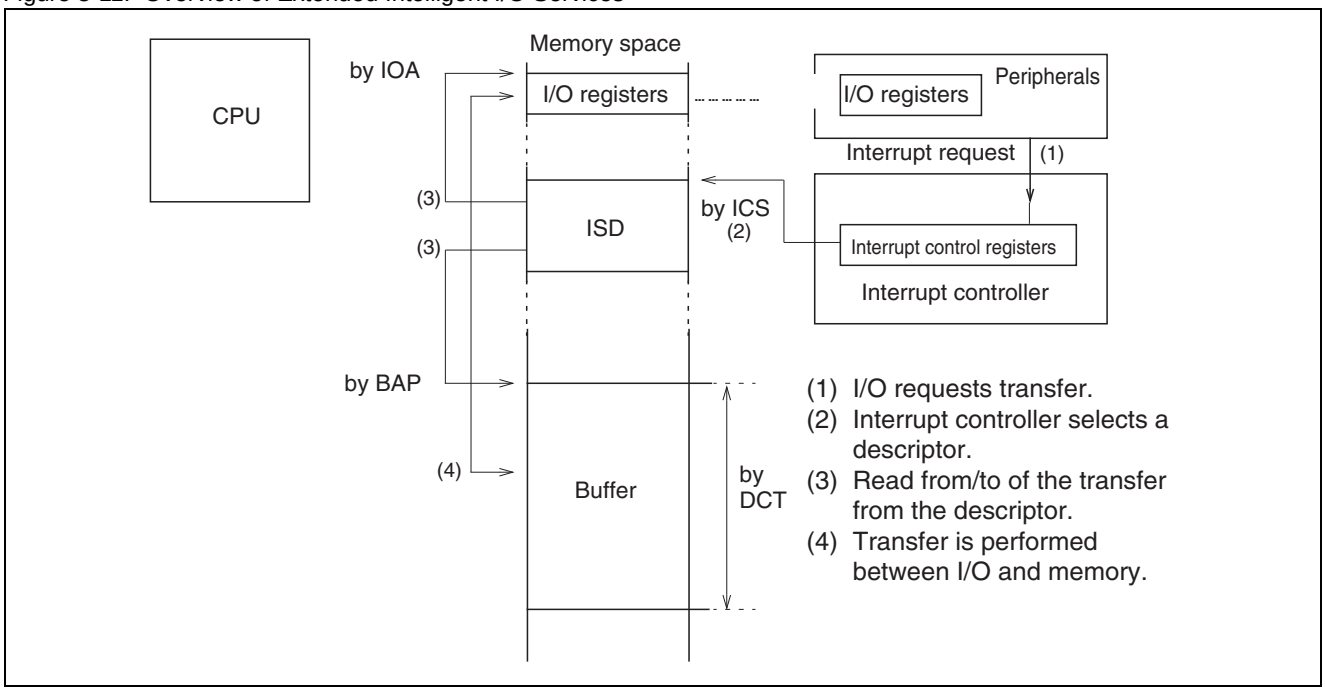
Located on the RAM and has transfer mode, I/O address, number of transfers and buffer address.

Note:

In case of using REALOS, extended intelligent I/O service (EI^{2}OS) is not available.

Overview of EI^{2}OS is shown in the Figure 3-11.

Figure 3-11. Overview of Extended Intelligent I/O Services



Notes:

- Available area that IOA can specify is between 000000_H and $00FFFF_H$.
- Available area that BAP can specify is between 000000_H and $FFFFFF_H$.
- Available maximum transfer count that DCT can specify is 65536.

Construction of Extended Intelligent I/O Services (EI^{2}OS)

Construction of EI^{2}OS is parted into four parts.

- Embedded resource .....Interrupt enable bit, Interrupt request bit: Control of the interrupt request from the resources
- Interrupt controller .....ICR: Interrupt level assignment, Priority determination of simultaneous request interrupt and selection of EI^{2}OS operation
- CPU .....I, ILM: Compares request interrupt level with current level determine the status of the interrupt enable
  Microcode: Steps for EI^{2}OS processing
- RAM ..... Descriptor: Describes transfer information of EI^{2}OS

3.7.1 Extended Intelligent I/O Service Descriptor (ISD)

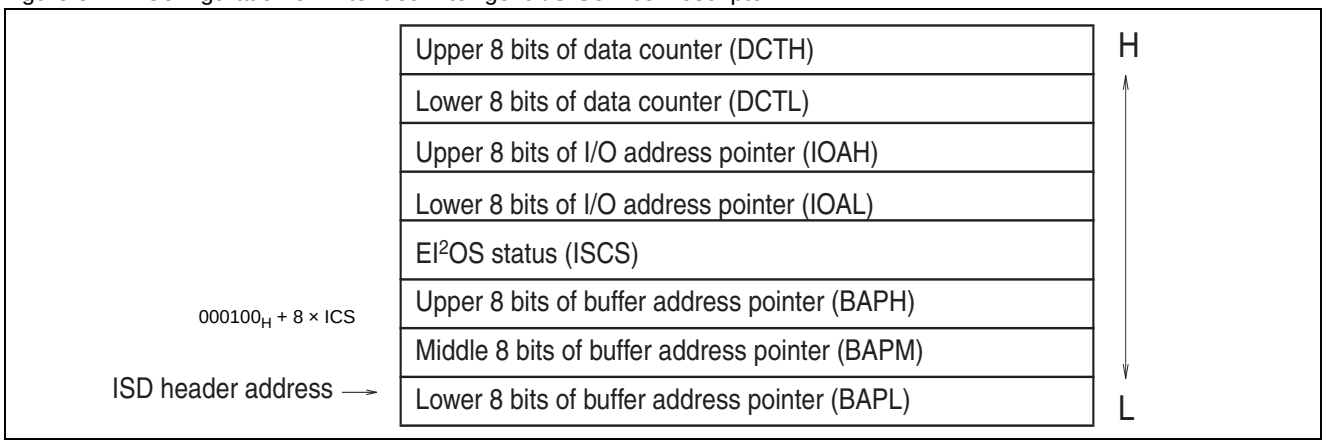
Extended intelligent I/O service descriptor is employed between 000100_H and $00017F_H$ in the internal RAM and consists of followings.

- Various control data for transfer
- Status data
- Buffer address pointer

Extended Intelligent I/O Service Descriptor (ISD)

Configuration of the extended intelligent I/O service descriptor is shown in the Figure 3-12.

Figure 3-12. Configuration of Extended Intelligent I/O Service Descriptor

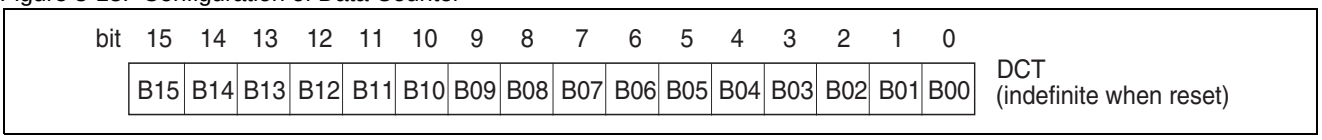


Data Counter (DCT)

Data counter (DCT) is a 16 bits length register and a counter applicable the number of transfer data. The counter will be decremented by 1 after the data is transferred. EI^{2}OS will be terminated after this counter reached zero.

Configuration of the data counter is shown in the Figure 3-13.

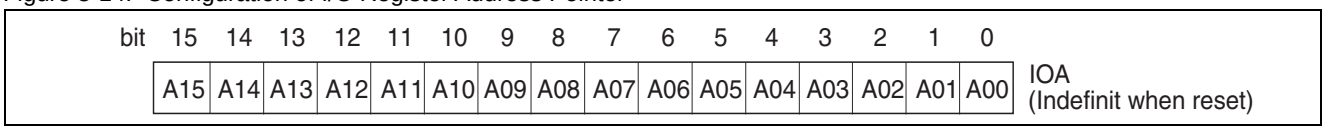
Figure 3-13. Configuration of Data Counter



I/O Register Address Pointer (IOA)

I/O register address pointer (IOA) is a 16 bits length register and indicates lower addresses (A15 to A00) of the I/O register that transfers the buffer and the data by the 16 bits of the I/O register address pointer. All of upper addresses (A23 to A16) are “0” and is able to determine any I/O in between 000000_H and $00FFFF_H$. Configuration of IOA is shown in Figure 3-14.

Figure 3-14. Configuration of I/O Register Address Pointer



Buffer Address Pointer (BAP)

Buffer address pointer is a 24 bits length register and holds the address for next EI^{2}OS transfer. BAP is employed independently for each channel for EI^{2}OS so that each EI^{2}OS is able to transfer to the 16 M bytes of any space. In case of BF bit of ISCS is set to “0” (updatable), only lower 16 bits of BAP can be changed but BAPH cannot.

3.7.2 EI^{2}OS Status Register (ISCS)

EI^{2}OS status register (ISCS) is a 8 bits length register that updates/fixes the buffer address pointer and I/O register address pointer, and indicates the transfer data length (byte/word) and transfer direction.

EI^{2}OS Status Register (ISCS)

Configuration of ISCS is shown in Figure 3-15.

Note that bit7 to bit5 should be written to “0”.

Figure 3-15. ISCS Configuration

bit	7	6	5	4	3	2	1	0	
	Reserved	Reserved	Reserved	IF	BW	BF	DIR	SE	ISCS (Invalid when reset)

Explanation for each bit is described as follows.

[bit4] IF

Specifies update/fix of the I/O register address pointer.

0: I/O register address pointer will be updated (incremented) after the data is transferred.

1: I/O register address pointer will be fixed after the data is transferred.

[bit3] BW

Determines the transfer data length.

0: Byte

1: Word

[bit2] BF

Specifies update/fix of the buffer address pointer.

0: Buffer address pointer will be updated (incremented) after the data is transferred.

1: Buffer address pointer will be fixed after the data is transferred.

Note:

Lower 16 bits of the buffer address pointer will only be changed on updates.

[bit1] DIR

Specifies the transfer direction of data.

0: I/O address pointer → Buffer address pointer

1: Buffer address pointer → I/O address pointer

[bit0] SE

Controls the termination of the extended intelligent I/O service by the request from embedded resource.

0: Does not terminate by the request from embedded resource.

1: Terminates by the request from embedded resource.

3.8 Operational Flow and Procedure of Extended Intelligent I/O Services (EI^{2}OS)

Operational flow of extended intelligent I/O services (EI^{2}OS) is shown in Figure 3-16, the procedure in Figure 3-17.

Operational Flow of Extended Intelligent I/O Services (EI^{2}OS)

Figure 3-16. Operational Flow of Extended Intelligent I/O Services (EI^{2}OS)

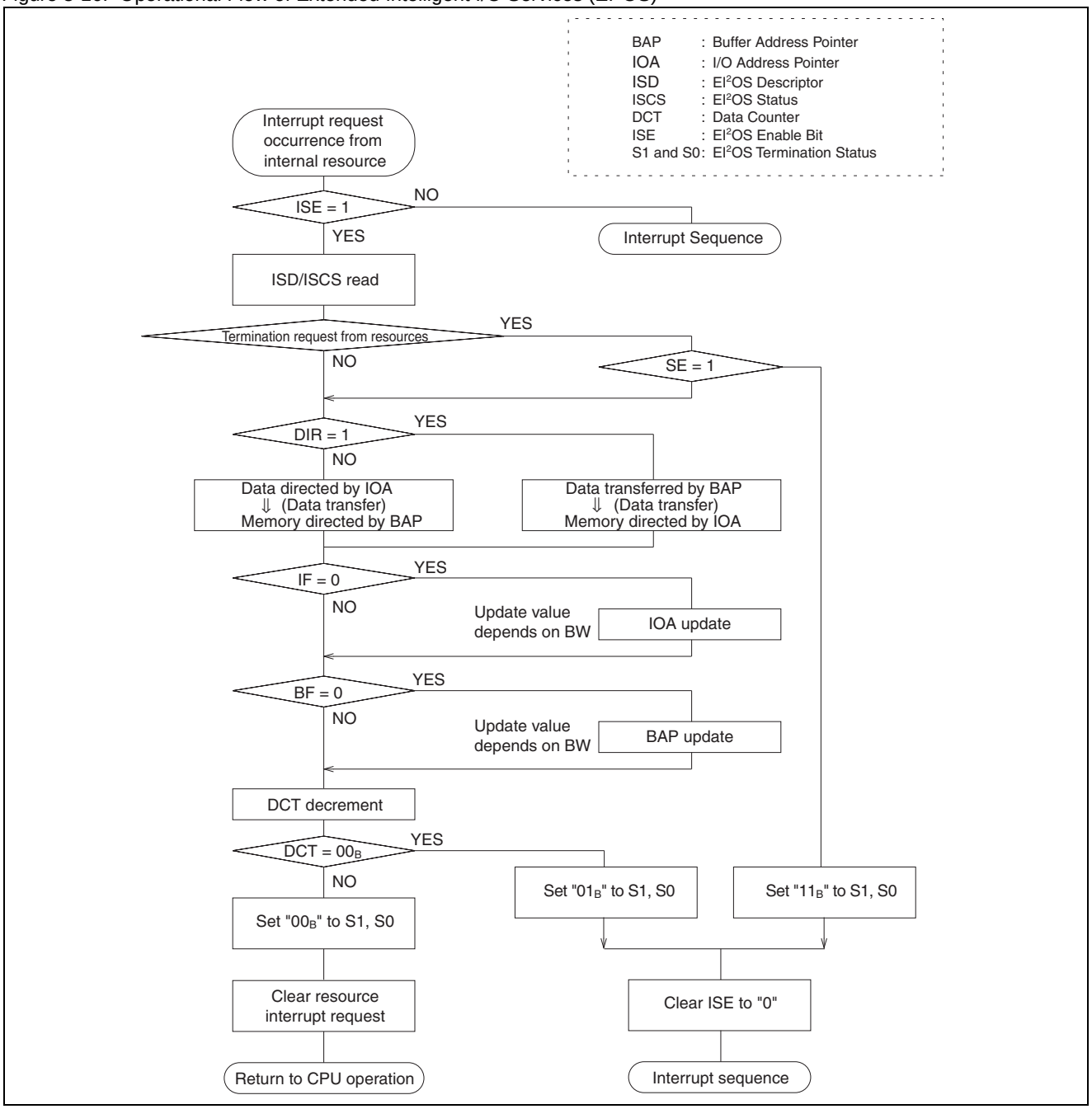
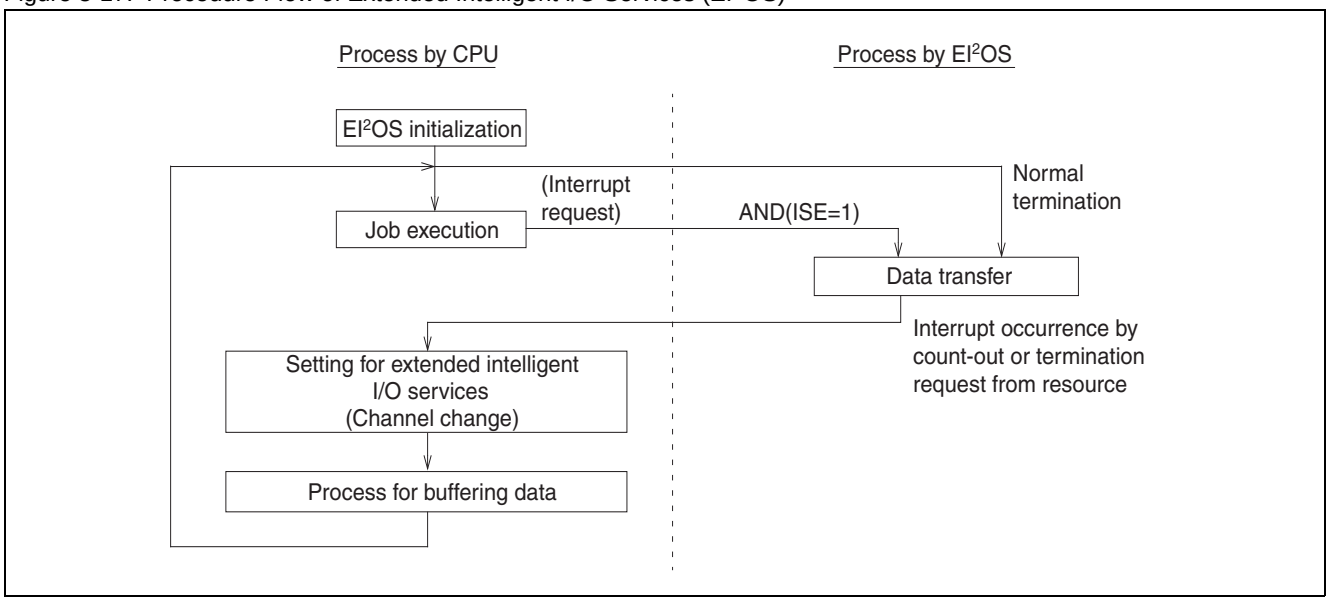


Figure 3-17. Procedure Flow of Extended Intelligent I/O Services (EI^{2}OS)



Process time for extended EI^{2}OS in each flow is as follows.

Continuous Data Transfer (termination condition is not met)

(Table 3-6 + Table 3-7) machine cycle

Termination Request from Resource

(36 + 6 × Table 3-8) machine cycle

Count Termination

(Table 3-6 + Table 3-7 + (21 + 6 × Table 3-8)) machine cycle

Table 3-6. Process Time for Continuous EI^{2}OS

ISCS/SE bit		Set to "0"		Set to "1"	
I/O Address Pointer		Fix	Update	Fix	Update
Buffer Address Pointer	Fix	32	34	33	35
	Update	34	36	35	37

Table 3-7. Compensation Value for Data Transfer of EI^{2}OS Process Time

I/O Address Pointer			Internal Access	
			Even	B/Odd
Buffer Address Pointer	Internal Access	Even	0	+2
		B/Odd	+2	+4

B : Byte data transfer
Even : Even address/word transfer
Odd : Odd address/word transfer

Table 3-8. Compensation Value of Interrupt Handling Time

Address directed by Stack Pointer	Compensation Value [cycle]
External 8 bits	+4
External Even Address	+1
External Odd Address	+4
Internal Even Address	0
Internal Odd Address	+2

3.9 µDMA Controller (µDMAC)

µDMA controller (µDMAC) is a simplified DMA with the same function as EI^{2}OS.
µDMAC accomplishes faster data transfer than EI^{2}OS by employing descriptor registers.

Overview of µDMA

µDMAC employs following functions.

- Automatic data transfer is performed between peripheral resource (I/O) and memory.
- Program execution of CPU will be stopped while DMA is activated.
- Watchdog timer operates while in DMA transfer.
- DMA transfer channel consists of 16 (smallest channel number has higher priority for DMA transfer).
- With/without address increment of transfer from/to is selectable.
- DMA transfer is booted interrupt trigger from peripheral resource (I/O).
- DMA transfer is controlled by (a) DMA enable register (DER), (b) DMA stop status register (DSSR), (c) DMA status register (DSR), (d) DMA descriptor channel select register (DCSR) and (e) descriptor (DMACS).
- STOP request is employed as a method to stop the DMA transfer from resources.
- After DMA transfer is terminated, a flag will be set to applicable bit of DMA Status Register (DSR) to output the interrupt to interrupt controller.

3.10 µDMAC Register

µDMAC employs four registers: DCSR, DSR, DSSR and DER. DMA descriptor used for DMA transfer setting is described in "3.11 DMA Descriptor Window Register (DDWR)".

Register List

- DMAC Descriptor Channel Select Register (DCSR)

bit	15	14	13	12	11	10	9	8	
$00009B_H$	STPctrl	reserved	reserved	reserved	DCSR3	DCSR2	DCSR1	DCSR0	DCSR
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

- DMAC Status Register (DSRL/DSRH)

bit	15	14	13	12	11	10	9	8	
$00009D_H$	DTE15	DTE14	DTE13	DTE12	DTE11	DTE10	DTE9	DTE8	DSRH
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B
bit	7	6	5	4	3	2	1	0	
$00009C_H$	DTE7	DTE6	DTE5	DTE4	DTE3	DTE2	DTE1	DTE0	DSRL
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

- DMAC Stop Status Register (DSSR)

bit	7	6	5	4	3	2	1	0	
$0000A4_H$	STP15	STP14	STP13	STP12	STP11	STP10	STP9	STP8	DSSR
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B
bit	7	6	5	4	3	2	1	0	
$0000A4_H$	STP7	STP6	STP5	STP4	STP3	STP2	STP1	STP0	DSSR
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

Use bit15 to bit8 of DSSR when STPctrl bit of DCSR is “0” and bit7 to bit0 when that bit is “1”.

■ DMAC Enable Register (DERL/DERH)

	bit	15	14	13	12	11	10	9	8	
$0000AD_H$		EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	DERH
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial Value 00000000_B
	bit	7	6	5	4	3	2	1	0	
$0000AC_H$		EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	DERL
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

3.10.1 DMAC Descriptor Channel Select Register (DCSR)

DMAC descriptor channel select register (DCSR) is a register that changes over descriptors for each channel. Set the descriptor after specifying a channel by this register.

DMAC Descriptor Channel Select Register (DCSR)

bit	15	14	13	12	11	10	9	8	
$00009B_H$	STPctrl	reserved	reserved	reserved	DCSR3	DCSR2	DCSR1	DCSR0	DCSR
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

[bit15] STPctrl: STP Control Bit

STPctrl bit	Function
0 [Initial value]	Select STP8 to STP15 as DSSR.
1	Select STP0 to STP7 as DSSR.

[bit14 to bit12] reserved: reserved bits

These bits are reserved.

These bits are always “0” when read.

Note that these bits should be written to “0”.

[bit11 to bit8] DCSR3 to DCSR0: DMAC descriptor channel select

Table 3-9. Relation between DCSR and Channel Selection

DCSR3 to DCSR0	Selected Channel	Resource Interrupt Request
0000_B	0	IRQ0/IRQ1
0001_B	1	Base Timer ch.0 (source 0, 1)
0010_B	2	Base Timer ch.1 (source 0, 1)
0011_B	3	Base Timer ch.2 (source 0, 1)
0100_B	4	Base Timer ch.3 (source 0, 1)
0101_B	5	PPG0/PPG4 Counter Borrow
0110_B	6	PPG1/PPG5 Counter Borrow
0111_B	7	PPG2/PPG6 Counter Borrow
1000_B	8	PPG3/PPG7 Counter Borrow
1001_B	9	16-bit Free-run Timer Overflow/Compare clear/Multi-Function Serial ch.4/ch.5/ch.6 (status)
1010_B	10	Multi-Function Serial ch.4 (reception)
1011_B	11	Multi-Function Serial ch.4 (transmission)
1100_B	12	Multi-Function Serial ch.5 (reception)
1101_B	13	Multi-Function Serial ch.5 (transmission)
1110_B	14	Multi-Function Serial ch.6 (reception)
1111_B	15	Multi-Function Serial ch.6 (transmission)

Setting DCSR will select descriptor for one of 16 channels. Refer to Section “3.11 DMA Descriptor Window Register (DDWR)” for detail.

3.10.2 DMAC Status Register (DSRL/DSRH)

DMAC status register (DSRL/DSRH) is a register that shows DMA transfer has terminated. An interrupt will occur at the same time as “1” is set to this register.

DMAC Status Register (DSRL/DSRH)

	bit	15	14	13	12	11	10	9	8	
$00009D_H$		DTE15	DTE14	DTE13	DTE12	DTE11	DTE10	DTE9	DTE8	DSRH
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B
	bit	7	6	5	4	3	2	1	0	
$00009C_H$		DTE7	DTE6	DTE5	DTE4	DTE3	DTE2	DTE1	DTE0	DSRL
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

[bit15 to bit0] DTE15 to DTE0: DMAC Status

DTE15 to DTE0 bits	Function
0 [Initial value]	No interrupt by DMA transfer termination. Note that these bits should be written to “0” when DTE15 through DTE0 are “0”.
1	Indicates that interrupt is requesting after DMA transfer is terminated. “1” cannot be set by DMA transfer STOP request except for the last transfer. Writing “0” to this bit clears to “0” when DTE15 to DTE0 are all “1” and writing “1” will holds previous data.

Note:

In case of writing to DSRL/DSRH, use read-modify-write (RMW) instruction.

3.10.3 DMAC Stop Status Register (DSSR)

DMAC stop status register (DSSR) is a register that indicates DMA transfer is stopped by STOP request. The meaning of this bit in this register varies depending on STPctrl bit of DMAC descriptor channel select register (DCSR).

DMAC Stop Status Register (DSSR)

When DCSR.STPctrl bit = 0

	bit	7	6	5	4	3	2	1	0	
$0000A4_H$		STP15	STP14	STP13	STP12	STP11	STP10	STP9	STP8	DSSR
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

When DCSR.STPctrl bit = 1

	bit	7	6	5	4	3	2	1	0	
$0000A4_H$		STP7	STP6	STP5	STP4	STP3	STP2	STP1	STP0	DSSR
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

[bit15 to bit0] STP15 to STP0: DMAC Stop Status

STP15 to STP0 bits	Function
0 [Initial value]	STOP request from resources is not accepted while DMA transfer. Note that these bits should be written to “0” when STP15 through STP0 are “0”.
1	Indicates that DMA transfer is stopped after received STOP request from resources while in DMA transfer. However, “1” cannot be set to STP15 through STP0 even if STOP request is accepted on the last transfer. Applicable bit of DMAC enable register will be cleared to “0” if a STOP request to applicable channel is accepted when SE bit of DMA control register is “1”. Writing “0” to this bit clears to “0” when STP15 to STP0 are all “1” and writing “1” will holds previous data.

Note:

In case of writing to DSSR, use read-modify-write (RMW) instruction.

Following 5 channels are applicable to STOP request.

Channel	Applicable STPx bit	Resource
ch.10	STP10	Multi-Function Serial ch.4 (reception)
ch.12	STP12	Multi-Function Serial ch.5 (reception)
ch.14	STP14	Multi-Function Serial ch.6 (reception)

Bit except for above means nothing.

Note:

DSSR is controlled by MSB bit of DCSR (STPctrl). Bit8 to bit15 are selected as DSSR when STPctrl is “0”, bit0 to bit7 are selected as DSSR when STPctrl is “1”. Bit8 to bit15 are selected at first as initial value of STPctrl is “0”.

3.10.4 DMAC Enable Register (DERL/DERH)

DMAC enable register (DERL/DERH) is a register that permits DMA transfer. If “1” is set to this register and an interrupt request occurs to applicable channel, starts DMA transfer, identified as DMA transfer request.

DMAC Enable Register (DERL/DERH)

bit	15	14	13	12	11	10	9	8	
$0000AD_H$	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	DERH
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B
bit	7	6	5	4	3	2	1	0	
$0000AC_H$	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	DERL
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Initial value 00000000_B

[bit15 to bit0] EN15 to EN0: DMAC Enable

EN15 to EN0 bits	Function
0 [Initial value]	DMA transfer is not executed.
1	Interrupt request from resource is treated as a DMA boot request to output interrupt request to interrupt controller after DMA transfer has been terminated. If the number of the DMA transfer becomes 0 or STOP request from resource is received, it will be cleared to "0" at point that DMA transfer stopped.

Notes:

- In case of writing to DERL/DERH use read-modify-write (RMW) instruction.
- Be sure to change the mode after setting EN15 to EN0 to “0” if change from stop to sleep mode is required.

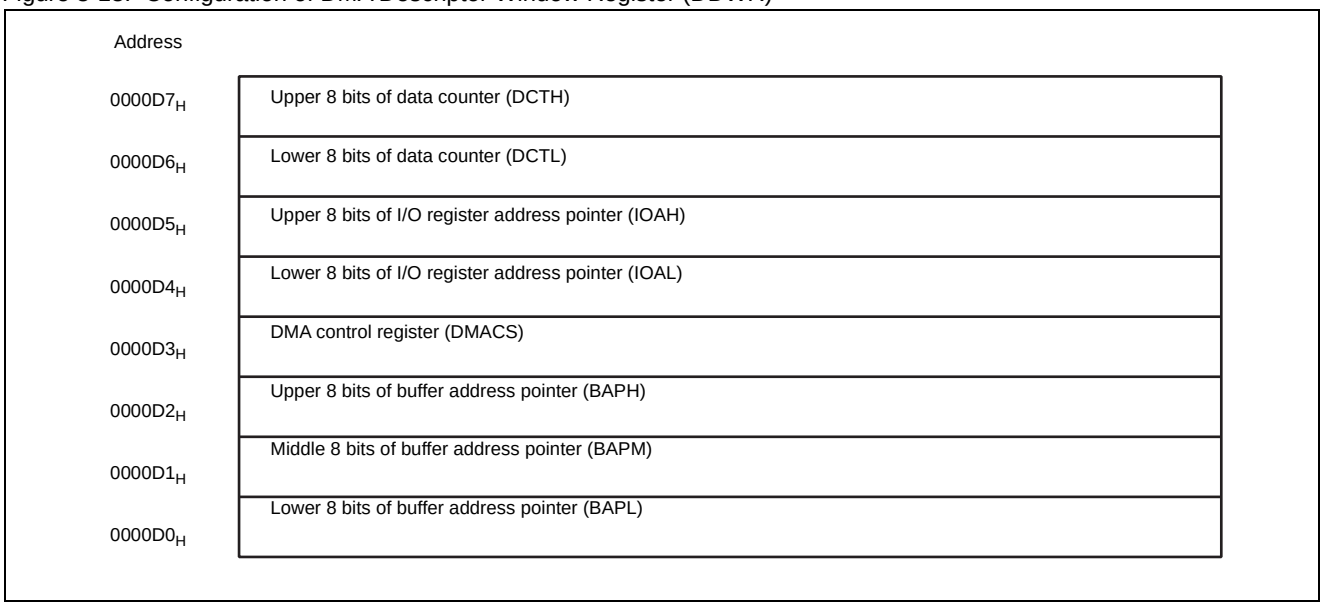
3.11 DMA Descriptor Window Register (DDWR)

DMA descriptor consists of 8 bytes x 16 channels to be used by setting the DMA transfer. Selected one of 16 channels will be accessible by mapping it in DMA descriptor window register (DDWR). Address of DDWR is $0000D0_H$ through $0000D7_H$.

Configuration of DMA Descriptor Window Register (DDWR)

DMA descriptor consists of 8 bytes x 16 channels and configuration of each channel is shown in Figure 3-18. A descriptor of the channel, selected by DMA descriptor channel select register (DCSR) or interrupt request channel number, will be mapped on DMA descriptor window register (DDWR). Refer to Table 3-9 for relation between DMA descriptor channel select register (DCSR) and selected channel.

Figure 3-18. Configuration of DMA Descriptor Window Register (DDWR)



Registers for DMA Descriptor

Registers that consists of DMA descriptor will be explained in the following page or later. Be sure to initialize before setting “1” to EN15 to EN0 as the initial value of the each register becomes invalid when reset.

Note:

Accessing to DMA descriptor window register (DDWR) is prohibited within 2 machine cycles if descriptor of the channel is changed by DMA descriptor channel select register (DCSR).

3.11.1 Data Counter (DCTL/DCTH)

Data counter (DCTL/DCTH) is a register that sets the number of data transfers.

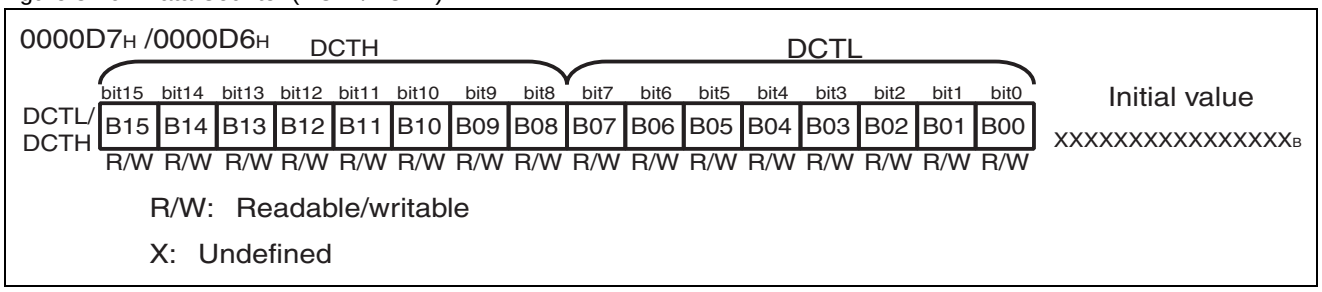
DMA transfer will be terminated when data counter (DCTL/DCTH) becomes “0”.

Data Counter (DCTL/DCTH)

Data counter (DCTL/DCTH) is a 16 bits length register and matched with number of transfers. The number of the counter will be decremented by 1 after each data transfer despite of the data is word or byte transfer. DMA transfer terminates when this counter becomes zero. The configuration of DCTL/DCTH is shown in Figure 3-19.

Setting DCT to “0” will set the maximum data transfer count at “65536”.

Figure 3-19. Data Counter (DCTL/DCTH)



Data Counter (DCTL/DCTH) Setting

The relation between number of transfer bytes and data counter (DCTL/DCTH) is as follows.

Table 3-10. The Relation Between Number of Transfer Bytes and Data Counter (DCTL/DCTH)

DMACS		DCTL/DCTH
BW Bit	BYTEL Bit	
0	-	N
1	0	N/2
1	1	(N+1)/2

N: Number of transfer bytes

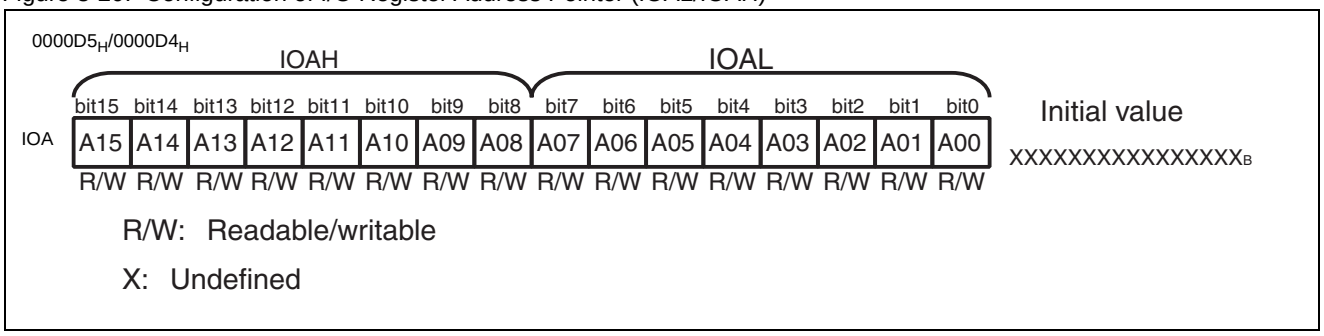
3.11.2 I/O Register Address Pointer (IOAL/IOAH)

I/O register address pointer (IOAL/IOAH) is a register that sets I/O address pointer. Upper addresses A23 to A16 are fixed with "00_H".

I/O Register Address Pointer (IOAL/IOAH)

I/O register address pointer (IOAL/IOAH) is a 16 bit length register to indicates lower 16 bits (A15 to A00) of I/O register address. Upper addresses (A23 through A16) are all "0" that can indicate any I/O address space from "000000_H" to "$00FFFF_H$". If IF bit (IOAL/IOAH update/fix selection bit) of DMA control register (DMACS) is set to "update enabled", IOAL/IOAH becomes +1 when byte transfer and +2 when word transfer, and if set to "update inhibited", IOAL/IOAH is fixed. Configuration of IOAL/IOAH is shown in Figure 3-20.

Figure 3-20. Configuration of I/O Register Address Pointer (IOAL/IOAH)



3.11.3 DMA Control Register (DMACS)

DMA control register (DMACS) controls the DMA transfer.

This register controls:

- Directional control (IOA → BAP, BAP → IOA)
- Transfer Bit length (byte, word)
- Address update (available, prohibited)
- Transfer interval
- Odd byte control on word transfer

DMA Control Register (DMACS)

DMA control register (DMACS) is a 8 bits length register and specifies update/fix mode for buffer address pointer and I/O register address point, transfer data length (byte/word), transfer direction, byte transfer, and wait command. Configuration of DMACS is shown in the Figure 3-21.

Figure 3-21. Configuration of DMA Control Register (DMACS)

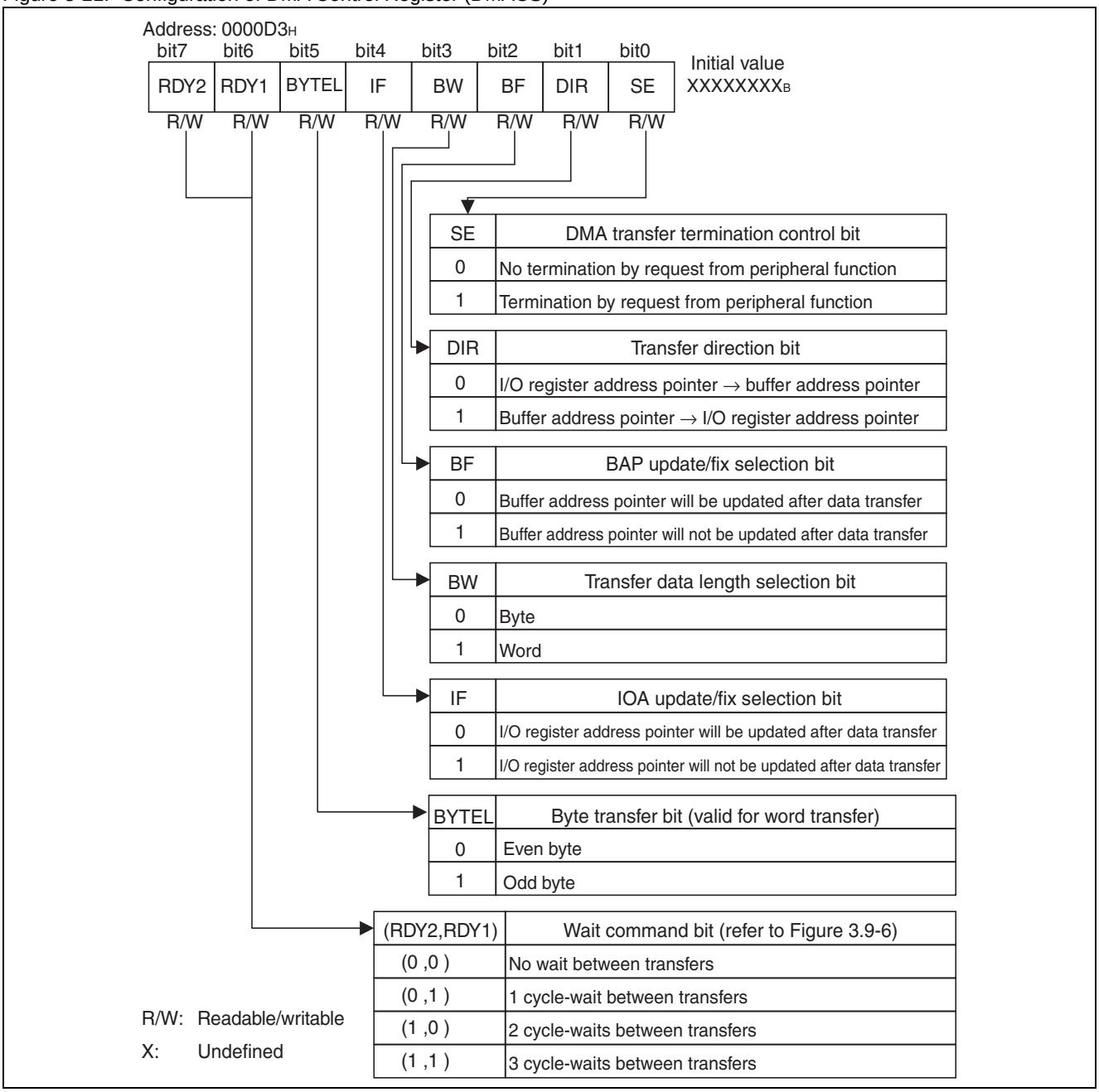
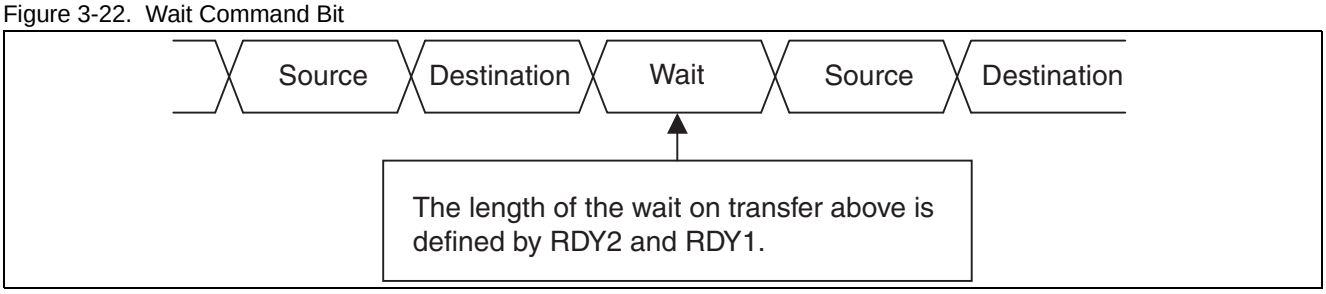


Figure 3-22. Wait Command Bit



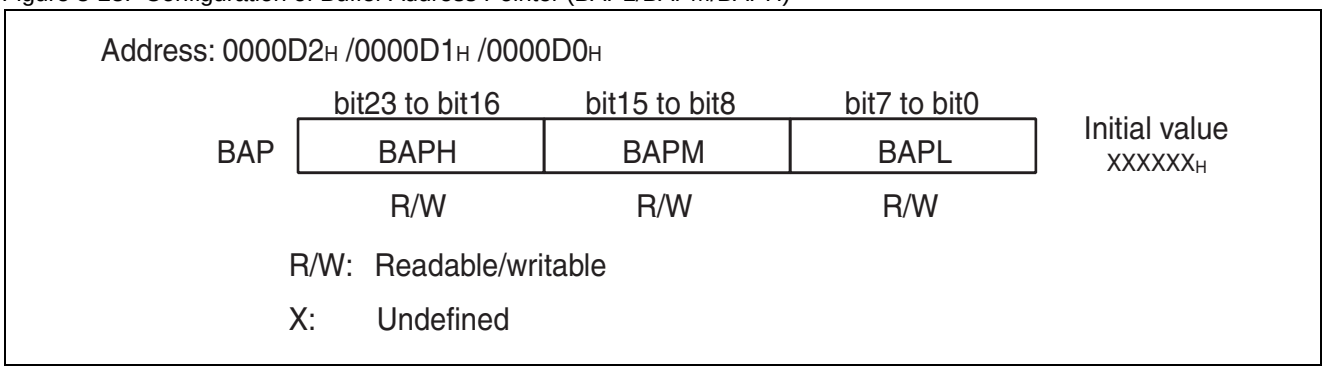
3.11.4 Buffer Address Pointer (BAPL/BAPM/BAPH)

Buffer address pointer (BAPL/BAPM/BAPH) is a register that sets the buffer address pointer. Buffer address pointer (BAPL/BAPM/BAPH) is available from A23 through A00.

Buffer Address Pointer (BAPL/BAPM/BAPH)

Buffer address pointer (BAPL/BAPM/BAPH) is a register that consists of 24 bits and saves the address used for DMA transfer. BAPL/BAPM/BAPH is independent for each channel of DMA so that each channel of DMA is able to transfer data between any address and I/O. If BF bit (BAPL/BAPM/BAPH update/fix selection bit) of DMA control register (DMACS) is set to “update enabled”, BAPL/BAPM/BAPH becomes +1 when lower 16 bits (BAPM, BAPL) is byte transfer and +2 when word transfer, and upper 8 bits (BAPH) will not change. Configuration of buffer address pointer (BAPL/BAPM/BAPH) is shown in Figure 3-23.

Figure 3-23. Configuration of Buffer Address Pointer (BAPL/BAPM/BAPH)



Notes:

- The area that I/O register address pointer (IOA) specifies is “000000_H” to “$00FFFF_H$”.
- The area that buffer address pointer (BAP) specifies is “000000_H” to “$FFFFFF_H$”.
- Setting an address for internal registers in μDMAC: DCSR, DSRH, DSRL, DSSR, DERH, DERL and DMA descriptor window register DDWR, to IOA and BAP is prohibited.

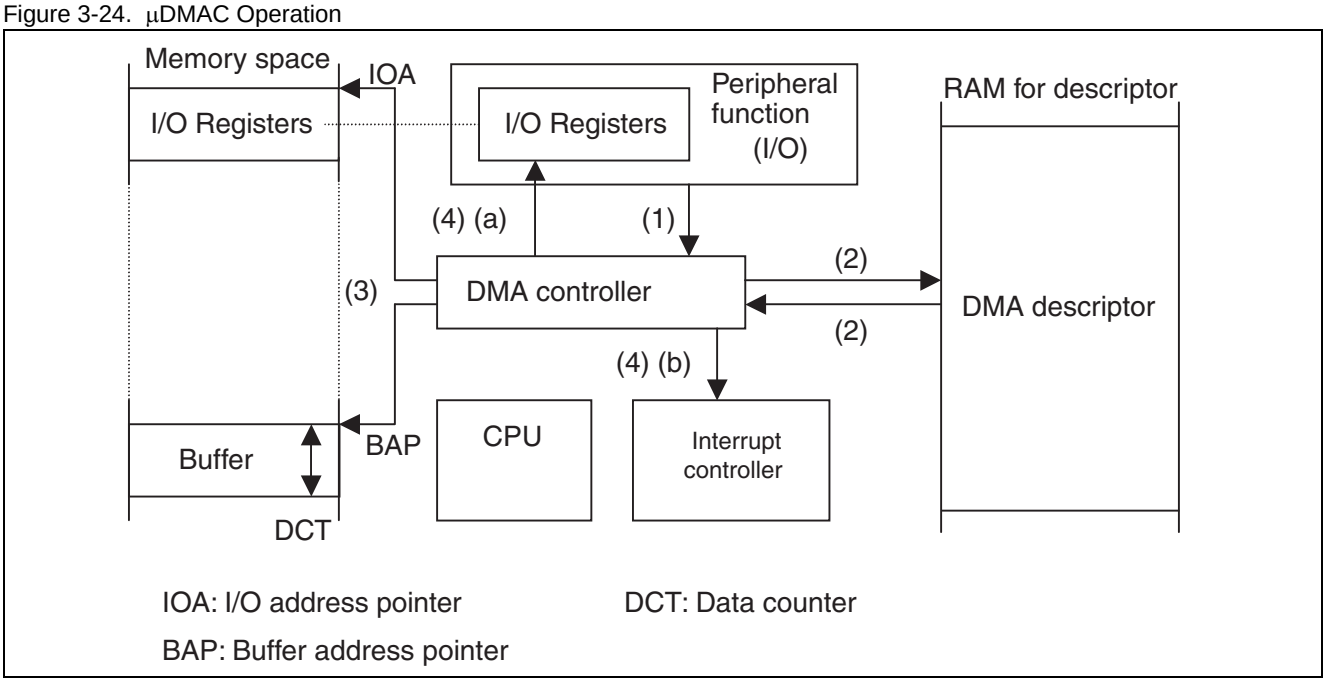
3.12 μDMAC Operation

μDMAC operation is described as follows.

μDMAC Operation

Explanation of μDMAC operation is shown in Figure 3-24.

Figure 3-24. μDMAC Operation



Data transfer using DMAC is performed with the sequence below.

(1) Peripheral resource (I/O) requests DMA transfer.

(2) DMAC reads transfer data: from/to addresses for specified channel and number of transfers, from descriptor when DMA enable register (DER) is “1”.

(3) Starts DMA data transfer between I/O and memory.

(4) After 1 byte or 1 word transfer is executed:

(a) When transfer is not terminated (data counter DCT ≠ 0)
Requests peripheral resource to clear the DMA transfer request.

(b) When transfer is terminated (data counter DCT = 0)

Sets a transfer termination flag to DMA status register to output an interrupt request to interrupt controller after DMA transfer is terminated.

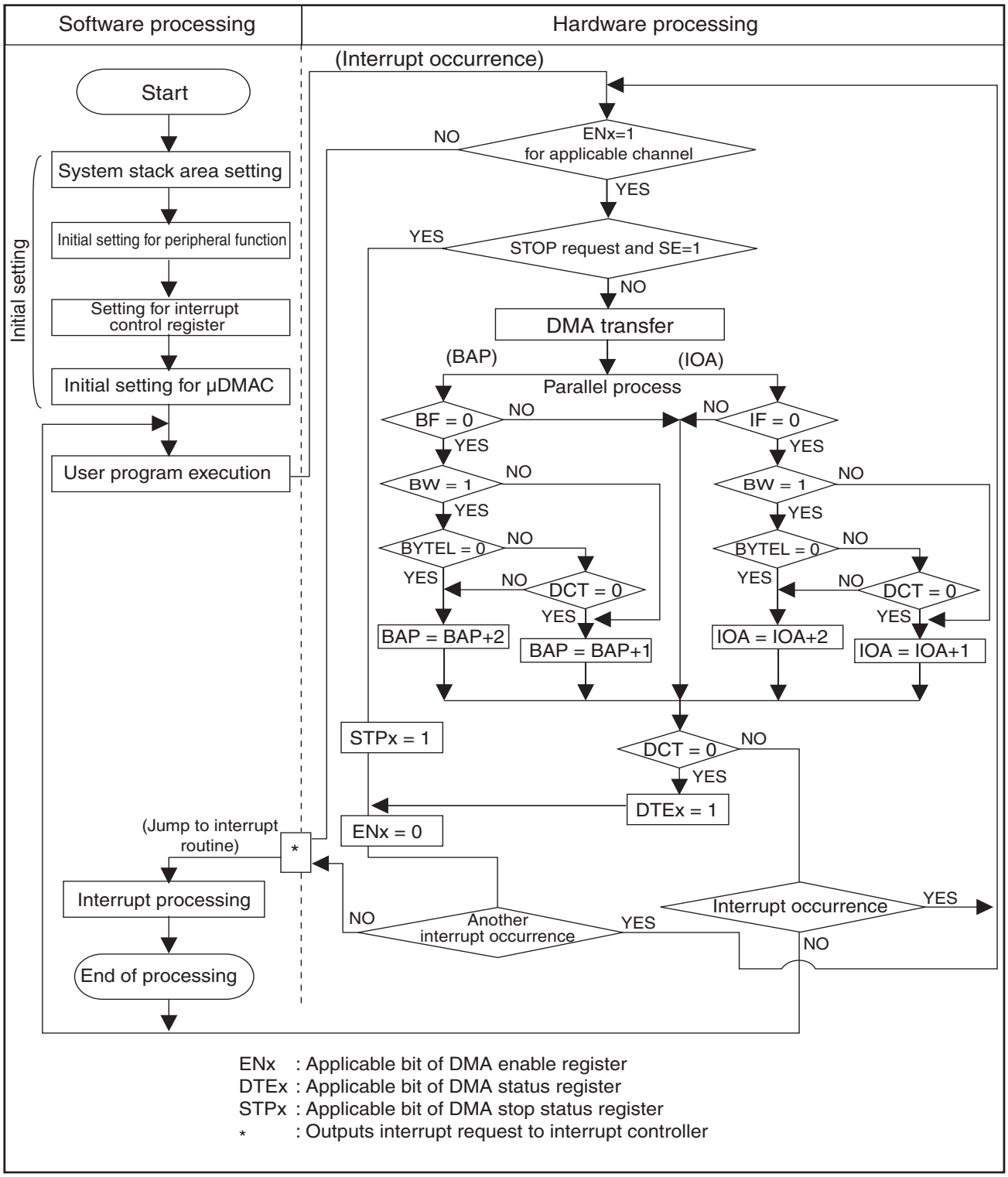
Note:

In case of writing to the internal registers DSRH, DSRL, DSSR, DERH and DERL, use read-modify-write (RMW) instruction.

DMAC Operation Procedure

Operation procedure of DMAC is shown in Figure 3-25.

Figure 3-25. DMAC Operation Procedure



Number of Data Transfer Cycle (Internal Transfer)

The number of transfer cycle on data transfer in the LSI, after µDMAC retains bus occupation, is as follows.

Table 3-11. Transfer Start after Bus Occupation

Match between DCSR3 to DCSR0 of DCSR and interrupt request channel	Mismatch between DCSR3 to DCSR0 of DCSR and interrupt request channel
1 Machine Cycle	2 Machine Cycles

Table 3-12. Transfer Cycles

Address Pointer		DMACS		Number of Cycles
BAP	IOA	BW	BYTEL	
-	-	0	-	4 + (RDY2, RDY1)*1 machine cycles
Odd	Even	1	-	6 + (RDY2, RDY1) *2 machine cycles
Even	Odd			
Odd	Odd	1	-	8 + (RDY2, RDY1) *2 machine cycles
Even	Even	1	-	4 + (RDY2, RDY1) *1 machine cycles

*1: (RDY2, RDY1) becomes “0” when in last transfer.

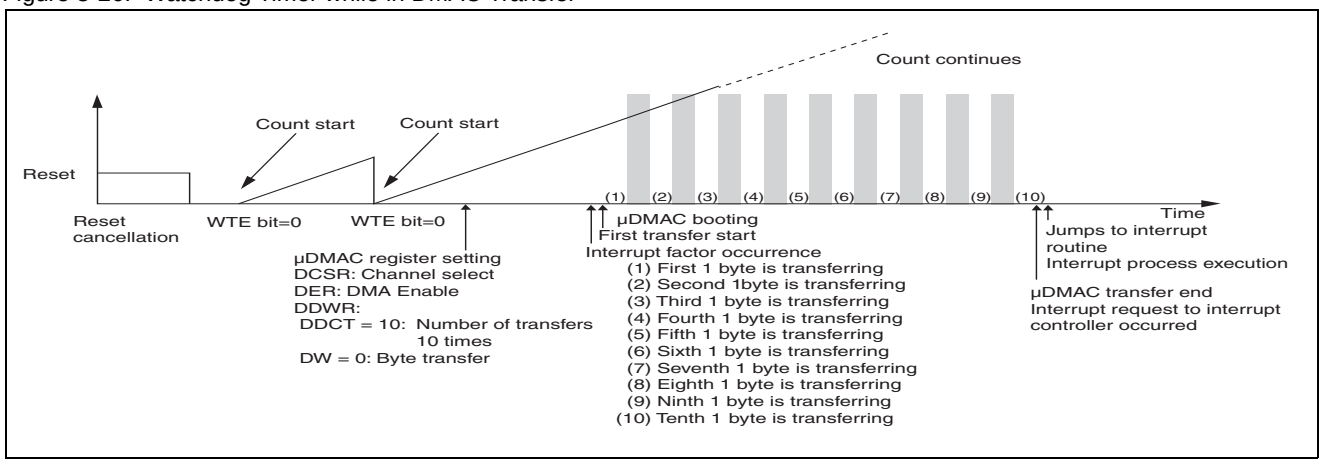
*2: Becomes 4 cycles and (RDY2, RDY1) becomes “0” when BYTEL = 1.
(RDY2, RDY0) becomes “0” when BYTEL = 0.

Odd : Odd Address
Even : Even Address

Watchdog Timer

Operation of watchdog timer, while in DMAC transfer, is shown in Figure 3-26. If the transfer exceeds the interval time of the watchdog timer which is set by WT1/WT0 bit in watchdog timer control register (WDTC), reset will be generated while in transfer.

Figure 3-26. Watchdog Timer while in DMAC Transfer



3.13 Exceptions

Exceptional handling is performed in F^{2}MC-16LX as exceptions occur by following factors.

Execution of Undefined Instructions

Exceptional handling is basically the same as of interrupt and exceptional handling is executed separated with ordinal process when detected an exceptional job at the boundary of instructions. Generally, it is recommended to be used only in debugging or booting recovery software on emergency, as an exceptional handling is required in results of unexpected operation.

Exception by Executing Undefined Instructions

F^{2}MC-16LX handles all the codes that are not defined in instruction map as undefined instructions. Executing undefined instruction is processed equivalent to software interrupt instruction: "INT10". This results in saving the content of AL, AH, DPR, DTB, ADB, PCB, PC and PS to system stack then setting I flag to "0", S flag to "1" and branching to the routine that is directed by the vector with the interrupt number 10. The value of the PC that is saved to stack means the address itself for stored undefined instruction. In case the instruction code is more than 2 bytes, the address becomes the value of which code determined as an undefined command is stored. Therefore, it is possible to restore the system by RETI command but it means nothing as it causes another exception.

4. Reset

This chapter explains reset for the MB90880 series.

4.1 Overview of Reset

4.2 Reset Factors and Oscillation Stabilization Wait Time

4.3 External-Reset Pin

4.4 Reset Operation

4.5 Reset-Factor Bits

4.6 State of Pins by Reset

4.1 Overview of Reset

If a reset factor occurs, the CPU immediately stops the processing currently in progress and stands by for cancellation of the reset. After the reset is canceled, processing starts at the address specified by the reset vector.

A reset is triggered by the following four factors:

- Power-on reset
- Watchdog timer overflow
- External reset request from $\overline{\text{RST}}$ pin
- Software reset request

Reset Factors

Table 4-1 summarizes the reset factors.

Table 4-1. Reset Factors

Reset	Reset factor	Machine clock	Watchdog timer	Oscillation stabilization wait
Power on	When power is turned on	Main clock (MCLK)	Stopped	Yes
Watchdog timer	Watchdog timer overflow	Main clock (MCLK)	Stopped	No
External pin	"L" level input to $\overline{\text{RST}}$ pin	Main clock (MCLK)	Stopped	No
Software	"0" is written in internal reset signal bit (RST) of low-power consumption mode control register (LPMCR)	Main clock (MCLK)	Stopped	No

Main clock: clock of oscillation clock divided by two

Power-on reset

A power-on reset occurs when the power is turned on. Oscillation stabilization wait time is fixed to 2^{18}/HCLK (approximately 65.54 ms: when oscillation clock is 4 MHz). A reset is performed after the end of the oscillation stabilization wait time.

Watchdog reset

A watchdog reset is triggered by a watchdog timer overflow if "0" is not written in the watchdog control bit (WTE) of the watchdog timer control register (WDTC) within a preset time after the watchdog timer is activated. The watchdog reset is performed before the oscillation stabilization wait time.

External reset

An external reset is triggered by input of the “L” level to the external-reset pin ($\overline{\text{RST}}$ pin). More than 16 machine cycles (16/ϕ) is required for the “L” level input time to $\overline{\text{RST}}$ pin.

An external reset ($\overline{\text{RST}}$ pin input reset) does not require the oscillation stabilization wait time.

After an instruction processing ends, the reset cancellation waiting state is set only when a reset request is issued via $\overline{\text{RST}}$ pin because of the event where a reset factor is triggered during writing (such as the MOV instruction while a transfer instruction is being executed). Writing thus ends normally even if a reset is input during writing.

However, string instructions (such as the MOVS instruction) accept a reset before a transfer completes at the specified counter, so the transfer of all data cannot be assured. Reset are also accepted when a bus cycle extension with RDY pin continues for more than 16 machine cycles during external bus access.

Software reset

In a software reset, an internal reset is triggered by writing “0” in the internal reset signal bit (RST) of the low-power consumption mode control register (LPMCR). A software reset does not require the oscillation stabilization wait time.

Reference:

Definition of clock

HCLK: Oscillation clock (clock supplied via high-speed oscillation pin)

MCLK: Main clock (clock of HCLK divided by two)

SCLK: Sub clock (clock divided by four, supplied via low-speed oscillation pin)

ϕ: Machine clock (CPU operation clock)

1/ϕ: Machine cycle (CPU operation clock period)

Refer to Section “5.1 Overview of Clocks” for a detailed information on machine clocks.

Note:

The oscillation stabilization wait time of 2^{17}/HCLK (about 32.77ms where the oscillation clock is 4MHz) is required if a reset is triggered in the stop mode or the sub clock mode.

Refer to Section "5.4 Clock Modes" for a detailed information on clock modes.

4.2 Reset Factors and Oscillation Stabilization Wait Time

The four types of reset factors can occur in the MB90880 series. The oscillation stabilization wait time during a reset varies depending on the reset factor.

Reset Factors and Oscillation Stabilization Wait Time

Table 4-2 summarizes the reset factors and the oscillation stabilization wait time.

Table 4-2. Reset Factors and Oscillation Stabilization Wait Time

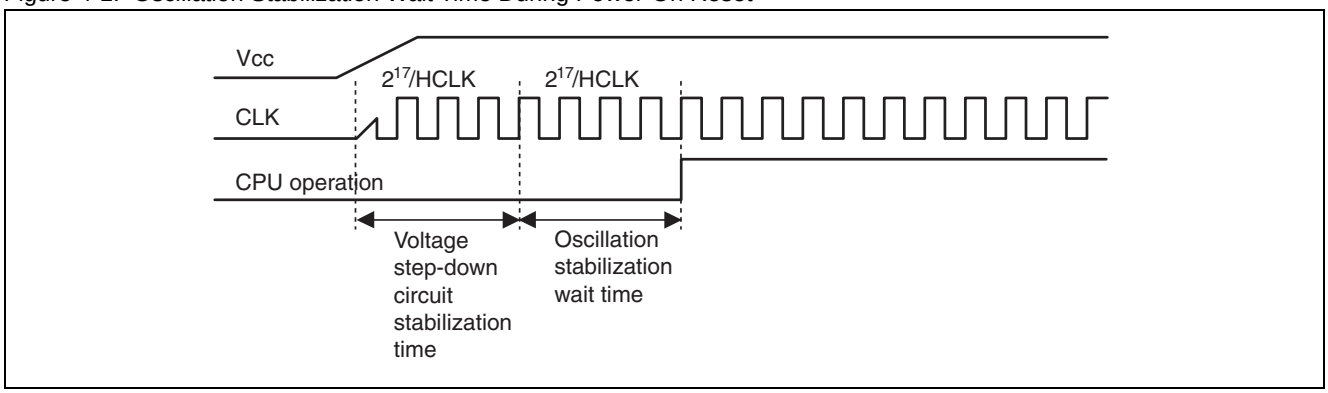
Reset factors	Oscillation stabilization wait time The value in parentheses ( ) is a period when oscillation clock is 4MHz
Power-on reset	2^{18}/HCLK (about 65.54ms)
Watchdog timer	None: WS1 and WS0 bits are initialized to "11_B".
External reset via $\overline{RST}$ pin	None: WS1 and WS0 bits are initialized to "11_B".
Software reset	None: WS1 and WS0 bits are initialized to "11_B".

HCLK: Oscillation clock

WS1, WS0: Oscillation stabilization wait time selection bits of clock selection register (CKSCR)

Figure 4-1 shows the oscillation stabilization wait time during a power-on reset.

Figure 4-1. Oscillation Stabilization Wait Time During Power-On Reset



Note:

Ceramic and crystal oscillators generally require an oscillation stabilization wait time ranging from several milliseconds to several ten milliseconds after the start of oscillation until oscillation stabilizes at a specific frequency. Therefore, specify a oscillation stabilization wait time suitable for the oscillator used.

Refer to Section "5.5 Oscillation Stabilization Wait Time" for more information.

Oscillation Stabilization Wait Reset State

A reset during the power-on sequence and a reset in response to a reset in the stop and sub clock modes is performed after the end of the oscillation stabilization wait time created by the time-base timer. In this event, a reset is performed after an external reset is canceled unless external reset input is cleared.

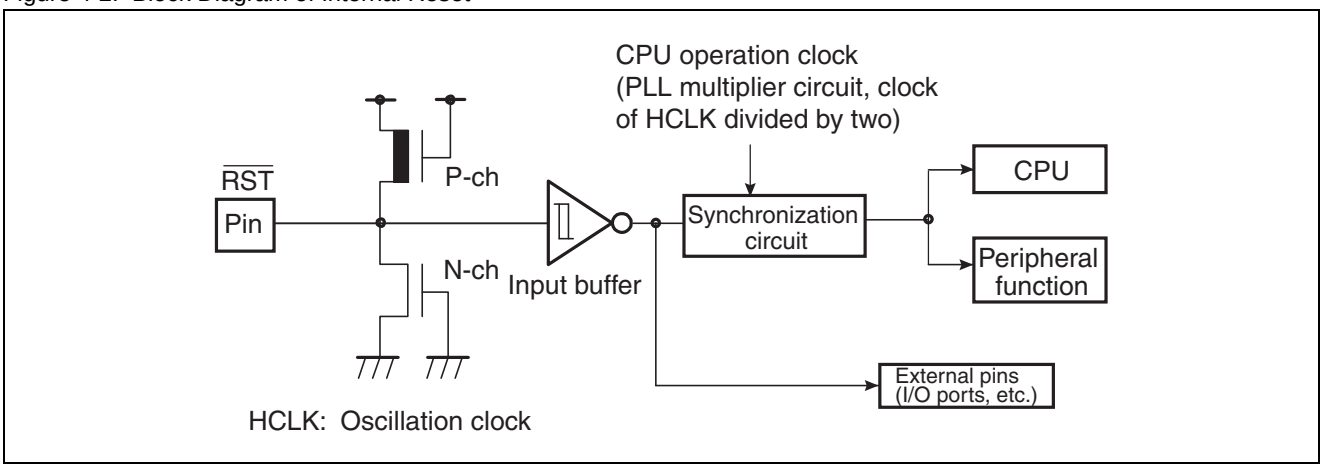
4.3 External-Reset Pin

The external-reset pin ($\overline{\text{RST}}$ pin) is a pin dedicated for the input of resets, and it triggers an internal reset by input of the “L” level. The MB90880 series have resets synchronized to the CPU operation clock. However, only external pins (e.g., I/O ports) change asynchronously to a reset state.

Block Diagram of External-Reset Pin

Figure 4-2 shows the block diagram of internal reset.

Figure 4-2. Block Diagram of Internal Reset



Note:

To prevent damage to the memory contents by a reset during writing, input to $\overline{\text{RST}}$ pin is accepted in a cycle that precludes damage to memory contents. A clock is required to initialize internal circuits. Input of a clock is required during input of a reset when an external clock is used for operation.

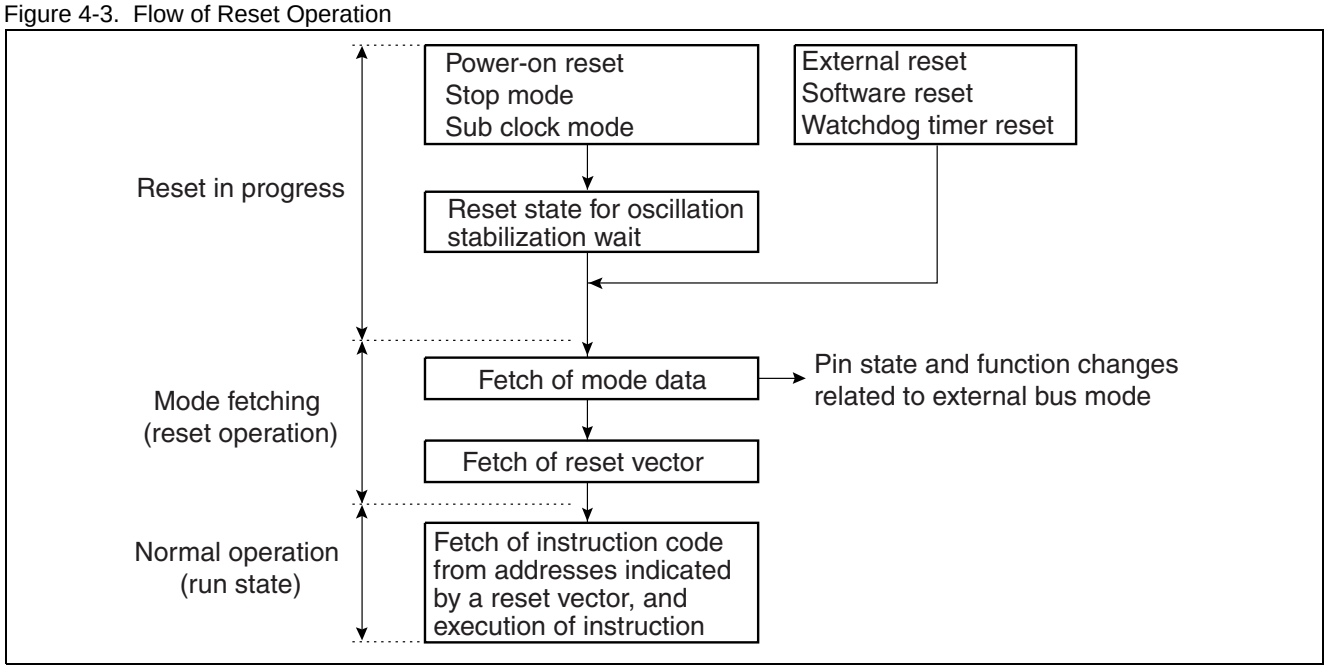
4.4 Reset Operation

After the cancellation of a reset, a read from operation of mode data and the reset vector can be selected by setting the mode pin to perform mode fetching. Mode fetching determines the CPU operation mode and the start address of execution after the end of a reset. When the power is turned on or when the system is returned from the stop mode or the sub clock mode by a reset, perform mode fetching after the end of the oscillation stabilization wait time.

Overview of Reset Operation

Figure 4-3 shows the flow of reset operation.

Figure 4-3. Flow of Reset Operation



Mode Pins

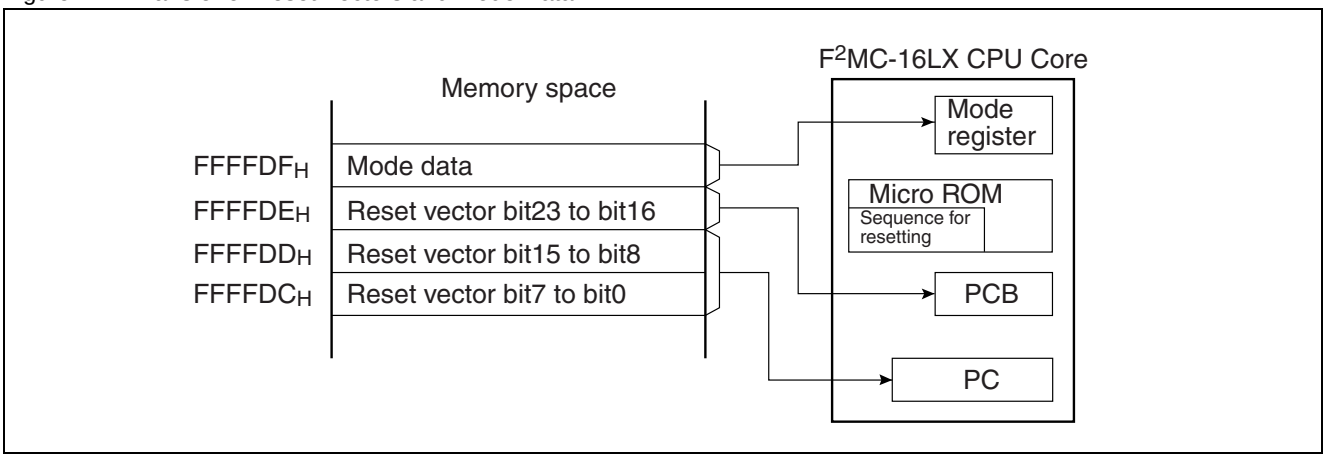
Mode pins (MD2 to MD0) specify a method to fetch reset vectors and mode data. A reset vector and mode data are fetched in a sequence for resetting. Refer to Section “7.2 Mode Pins (MD2 to MD0)” for details of the mode pins.

Mode Fetch

After a reset is canceled, the CPU transfers the reset vectors and mode data to the applicable registers in the CPU core by hardware. The reset vectors and mode data are allocated to four bytes, namely $FFFFDC_H$ to $FFFFDF_H$. Upon a reset cancellation, the CPU immediately outputs these addresses to a bus and fetches reset vectors and mode data. During mode fetching, the CPU starts processing from the address specified by the reset vector.

Figure 4-4 shows transfer of reset vectors and mode data.

Figure 4-4. Transfer of Reset Vectors and Mode Data



Reference:

Use a mode pin, from which reset vectors and mode data are read, to specify either internal ROM or external memory. If the external vector mode is specified with a mode pin, however, external memory is accessed to read reset vectors and mode data. Cypress recommends specifying the internal vector mode with a mode pin when the single-chip mode or internal ROM external bus mode is used.

Mode data (Address: $FFFFDF_H$)

The data in the mode register can be modified only by a reset, and the mode register settings become effective after a reset. Refer to Section "7.3 Mode Data" for details on mode data.

Reset vector (Address: $FFFFDC_H$ to $FFFFDE_H$)

Write the execution start address after the end of a reset. Execution starts from this address.

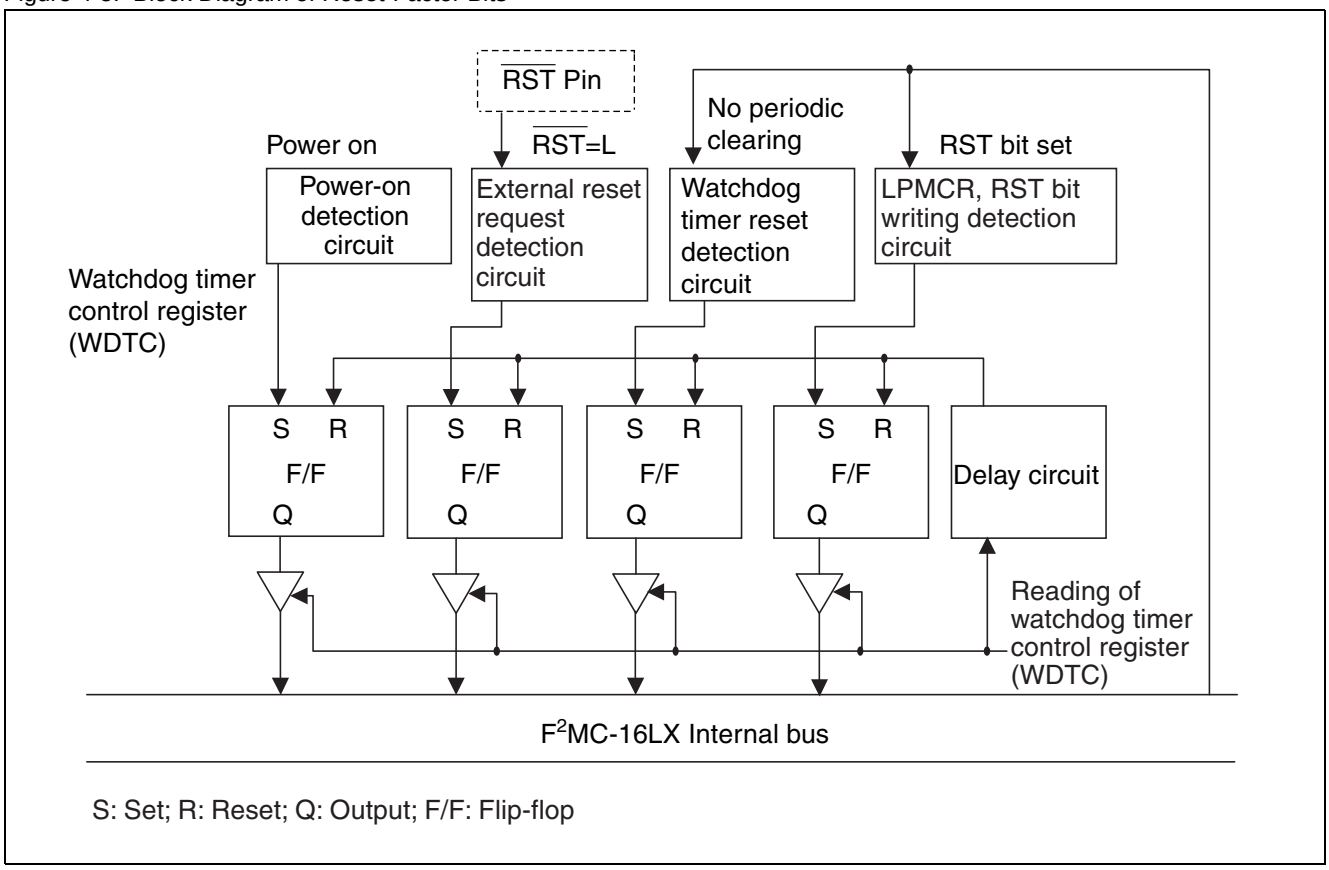
4.5 Reset-Factor Bits

Reset factors can be determined by reading the watchdog timer control register (WDTC).

Reset-Factor Bits

As shown in the Figure 4-5, each reset factor has a corresponding flip-flop circuit assigned to it. This information can be obtained by reading the watchdog timer control register (WDTC). If a reset factor must be determined after a reset cancellation, read the value of the WDTC register by software, and branch to an appropriate program.

Figure 4-5. Block Diagram of Reset-Factor Bits



Correspondence Between Reset-Factor Bits and Reset Factors

Figure 4-6 shows the configuration of the reset-factor bits for the watchdog timer control register (WDTC). Table 4-3 shows the correspondence between reset-factor bits and reset factors.

For details, refer to Section "10.2 Configuration of Watchdog Timer".

Figure 4-6. Configuration of Reset-Factor Bits (Watchdog Timer Control Register)

bit	15 to 8	7	6	5	4	3	2	1	0	
$0000A8_H$	(TBTC)	PONR	Reserved	WRST	ERST	SRST	WTE	WT1	WT0	
		X	X	X	X	X	1	1	1	Initial value
		R	R	R	R	R	W	W	W	R/W

R: Read only
W: Write only

Table 4-3. Correspondence Between Reset-Factor Bits and Reset Factors

Reset factor	PONR	WRST	ERST	SRST
Power-on reset	1	X	X	X
Watchdog timer overflow	*	1	*	*
External reset request via $\overline{\text{RST}}$ pin	*	*	1	*
Software reset request	*	*	*	1

X: Undefined

* : Preserving the previous state

Cautions about Reset-Factor Bits

If more than one reset factor occurs

If more than one reset factor occurs, the individual reset-factor bits of the WDTC register are set to “1”. For example, if an external reset via $\overline{\text{RST}}$ pin is requested at the same time as a watchdog timer overflow occurs, ERST and WRST bits of the reset-factor bits are set to “1”.

Power-on reset

During a power-on reset, PONR bit of the reset-factor bits is set to “1”. However, the reset-factor bits other than PONR bit are undefined. Therefore, if PONR bit is “1”, create program so that reset-factor bits other than PONR bit are ignored.

Clearing reset-factor bits

The reset-factor bits are cleared only if the data in the WDTC register is read. Bits corresponding to reset factors that have occurred once are not cleared even if a reset is triggered (remains “1”).

Note:

The values of the WDTC register may not be assured if the power is turned on under a condition that precludes a power-on reset.

4.6 State of Pins by Reset

This section explains the states of pins by a reset.

Pin States During a Reset

States of the pins during a reset are determined by the settings of mode pins (MD2 to MD0).

If the internal vector mode is set (MD2 to MD0 = 011_B)

All I/O pins (resource pins) become set at the high-impedance state, and mode data is read from internal ROM.

Refer to Section "6.7 Pin State in Standby Mode, Hold, and Reset" for the states of pins during a reset.

Pin States after Mode Data is Read

The states of the pins after mode data is read are determined by mode data (M1, M0).

If the single-chip mode is set (M1, M0 = 00_B)

All I/O pins (resource pins) become set at the high-impedance state, and mode data is read from the internal ROM.

Note:

Take care with the pins that have the high-impedance state during a reset so that equipment connected to the pins do not malfunction.

5. Clocks

This chapter describes the clocks of the MB90880 series.

5.1 Overview of Clocks

5.2 Block Diagram of Clock Generator

5.3 Clock Selection Register (CKSCR) and PLL Output Selection Register (PLLOS)

5.4 Clock Modes

5.5 Oscillation Stabilization Wait Time

5.6 Connecting Oscillator and External Clock

5.1 Overview of Clocks

The clock generator controls the operations of internal clocks, which are the operation clocks of the CPU and peripheral functions. In this document, the clocks are called as follows according to clock type:

- Machine clock: Defined as an internal clock.
- Machine cycle: Defined as one period of a machine clock.
- Oscillation clock: Defined as a clock supplied via a high-speed oscillation pin.
- PLL clock: Defined as a clock using internal PLL oscillation.
- Sub clock: Clock divided by four or two, provided from a low-speed oscillation pin.

Overview of Clocks

The clock generator contains an oscillation circuit and generates an oscillation clock and sub clock by using an external connection to an oscillator. The generator generates an oscillation clock by inputting a clock generated externally. The generator contains a PLL clock multiplier circuit and generates four multiplication clocks of an oscillation clock. The clock generator controls the oscillation stabilization wait time, PLL clock multiplication, and operations of internal clocks by changing the clock of the clock selector.

Oscillation clock (HCLK)

This clock is generated by connecting an oscillator to the high-speed oscillation pin or by inputting an external clock.

Sub clock (SCLK)

This clock operates the watch timer. It can also be used as a low-speed machine clock.

This clock is divided by four or two and created by connecting an oscillator to the low-speed oscillation pin or by inputting an external clock.

Main clock (MCLK)

This is a clock of the oscillation clock divided by two, and is used as an input clock to the time-base timer and clock selector.

PLL clock (PCLK)

This clock is a clock obtained by multiplying with built-in PLL clock multiplier circuit (PLL oscillation circuit). Four types of the clocks can be selected.

Machine clock (ϕ)

This clock is an operation clock of the CPU and peripheral functions. One period of this clock is used as a machine cycle (1/ϕ). One clock can be selected from among the main clock (clock of oscillation clock divided by two), sub clock, and four types of multiplication clocks.

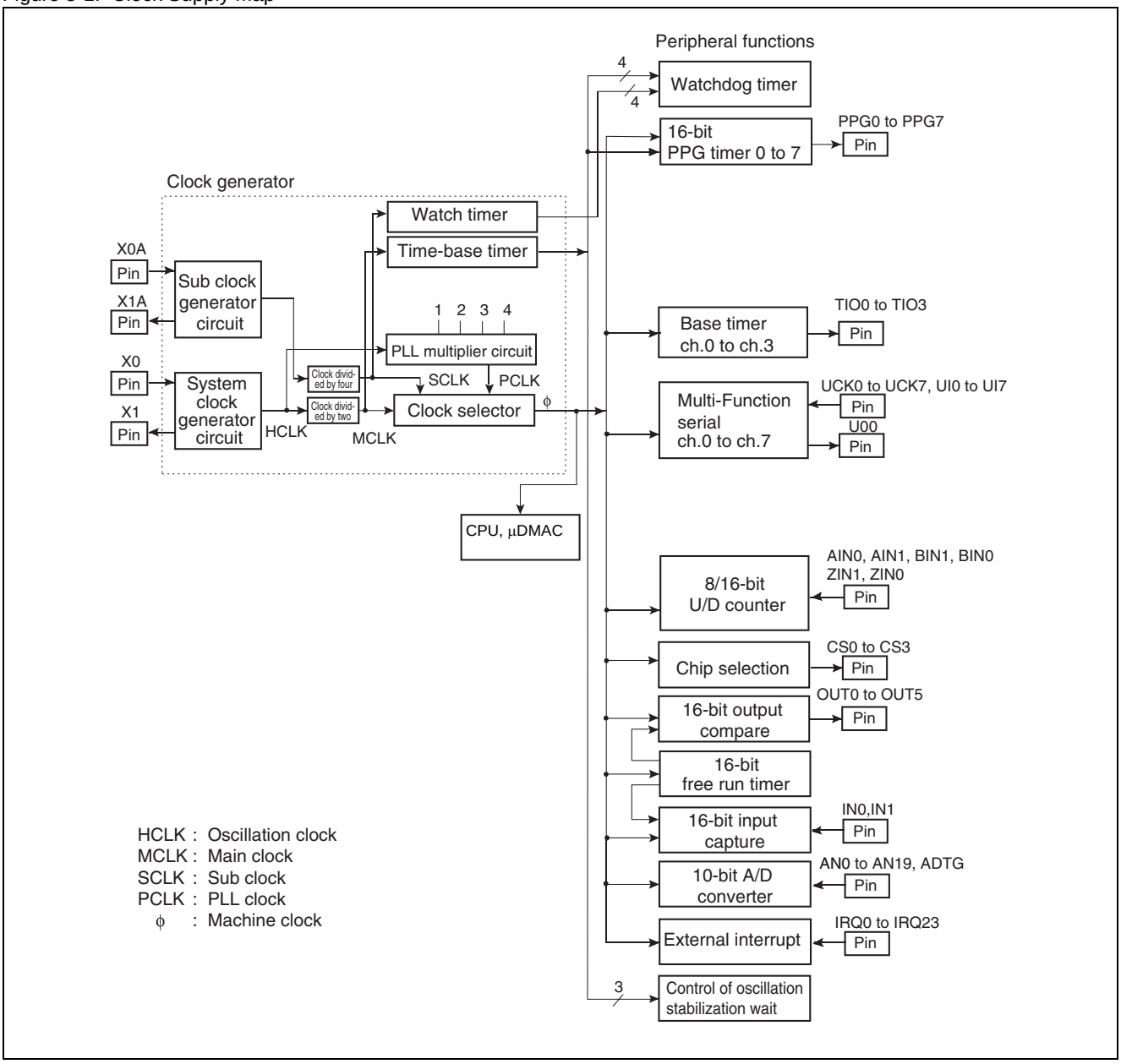
Note:

The maximum operating frequency of the CPU and peripheral functions is 33 MHz
(MB90F883B(S), MB90F884B(S): 25 MHz).
If a multiply-by rate exceeding the maximum operating frequency is specified, the device will not operate correctly.

Clock Supply Map

Machine clocks generated by the clock generator are supplied as operation clocks of the CPU and peripheral functions. Therefore, operations of the CPU and peripheral functions are affected by changes between the main clock and PLL clock (clock mode) and by changes in the PLL clock multiplication rate. The clock-divided outputs of the time-base timer are supplied to some peripheral functions, and the peripheral functions can select their own operation clocks. Figure 5-1 shows a clock supply map.

Figure 5-1. Clock Supply Map



5.2 Block Diagram of Clock Generator

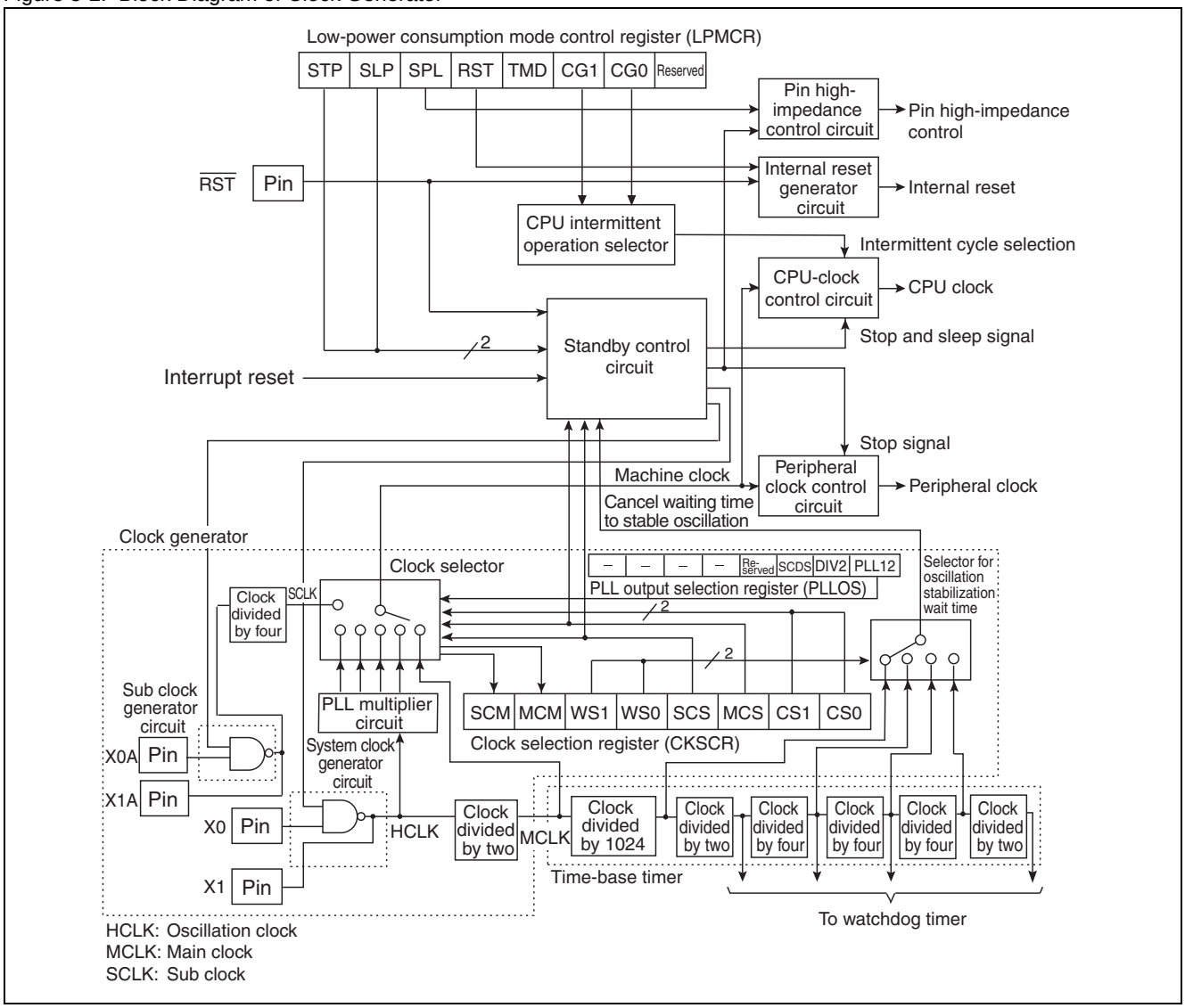
The clock generator consists of the following five blocks:

- System clock generator circuit/sub clock generator circuit
- PLL multiplier circuit
- Clock selector
- Clock selection register (CKSCR) and PLL output selection register (PLLOS)
- Selector for oscillation stabilization wait time

Block Diagram of Clock Generator

Figure 5-2 is a block diagram of the clock generator. Figure 5-2 also includes the standby control circuits and time-base timer circuit.

Figure 5-2. Block Diagram of Clock Generator



System clock generator circuit

This circuit generates an oscillation clock (HCLK) by using an oscillator connected to the high-speed oscillation pin. Also, an external clock can be input to it.

Sub clock generator circuit

This circuit generates a sub clock (SCLK) by using an oscillator connected to the low-speed oscillation pin. Also, an external clock can be input to it.

PLL multiplier circuit

This circuit multiplies an oscillation clock by using PLL oscillation and supplies it to the CPU clock selector.

Clock selector

This circuit selects clocks supplied to the CPU clock control circuit and peripheral clock control circuit from among the main clock, sub clock, and four PLL clocks.

Clock selection register (CKSCR)

This register changes between the oscillation clock and PLL clocks, selects the oscillation stabilization wait time, and selects the multiplication rate of the PLL clocks.

PLL output selection register (PLLOS)

This register specifies doubling of the multiply-by rate for the PLL specified in the CKSCR register.

Selector for oscillation stabilization wait time

This circuit selects the oscillation stabilization wait time of the oscillation clock when the stop mode is reset and during watch-dog reset. Four types of the time-base timer output are selected.

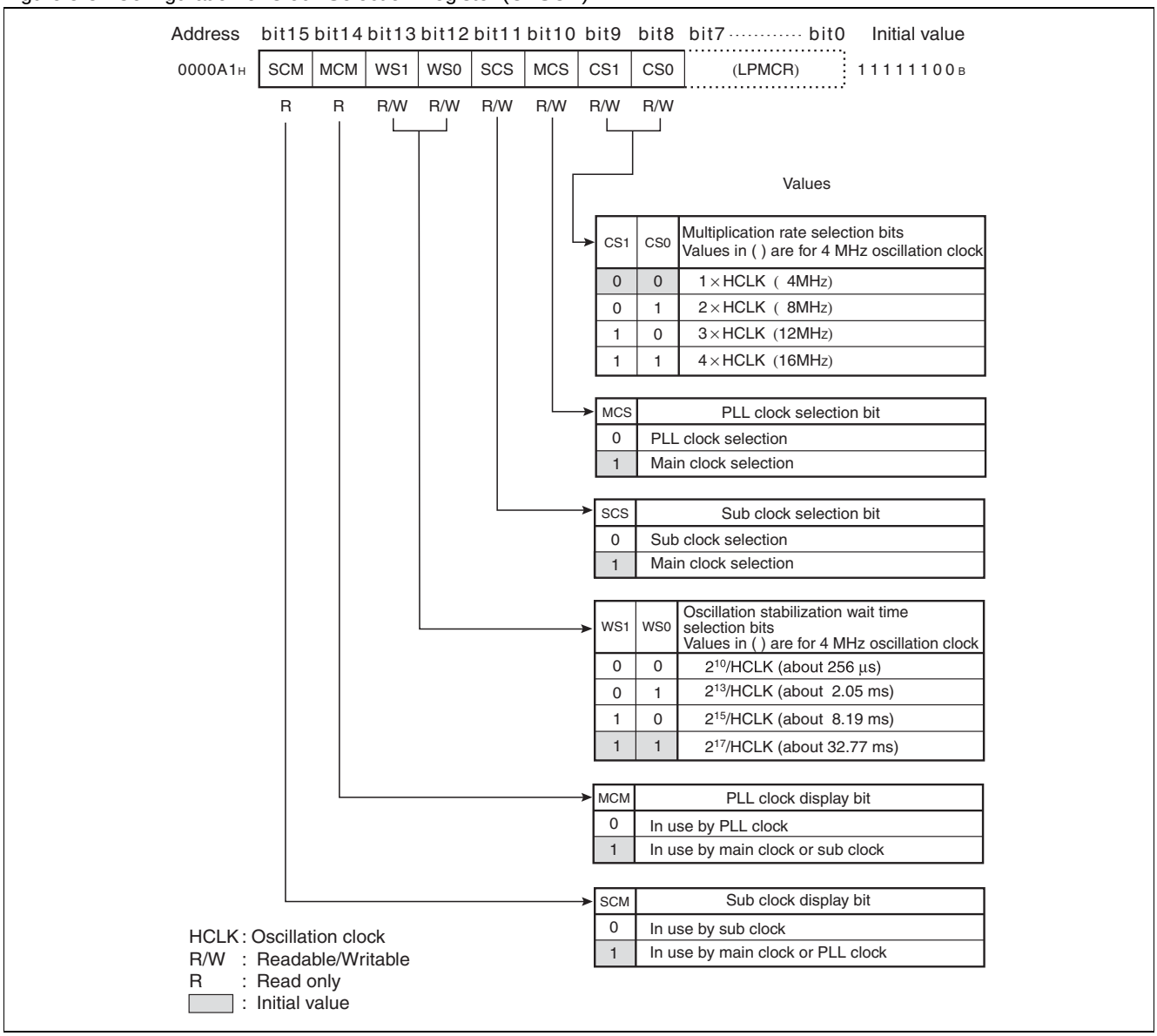
5.3 Clock Selection Register (CKSCR) and PLL Output Selection Register (PLLOS)

The clock selection register (CKSCR) switches among the main clock, sub clock, and PLL clock, and it selects the oscillation stabilization wait time and PLL clock multiplication rate.

Configuration of Clock Selection Register (CKSCR)

Figure 5-3 shows the configuration of the clock selection register (CKSCR). Table 5-1 explains the functions of bits in the clock selection register.

Figure 5-3. Configuration of Clock Selection Register (CKSCR)



Note:

When reset, the machine clock selection bit (MCS) is initialized to the main clock selection.

Table 5-1. Functions of Bits in Clock Selection Register (CKSCR) (Sheet 1 of 2)

Bit name		Function
bit15	SCM: Sub clock display bit	This bit displays whether the main clock or sub clock is selected as a machine clock. ■ If the bit is “0”, the sub clock is selected. If “1”, the main clock or PLL clock is selected. ■ If SCS = 1 and SCM = 0, the mode is the waiting time for stable oscillation of the main clock. ■ Writing has no effect on operation.
bit14	MCM: PLL clock display bit	This bit displays whether the main clock or PLL clock is selected as a machine clock. ■ If this bit is “0”, the PLL clock is selected. If “1”, the main clock or sub clock is selected. ■ If PLL clock selection bit (MCS) = 0 and MCM = 1, the mode is the waiting time for stable oscillation of the PLL clock. ■ Writing has no effect on operation.
bit13, bit12	WS1, WS0: Oscillation stabilization wait time selection bits	Selects the oscillation stabilization wait time of oscillation clock in a change from the sub clock mode to the main clock mode or from the sub clock mode to the PLL clock mode if the stop mode is canceled. Initialized to “11_B” by all reset factors. **Note**: The specified value of the oscillation stabilization wait time must be suitable for the oscillator used. Refer to Section “4.2 Reset Factors and Oscillation Stabilization Wait Time”, for more information. Specify “00_B” only if the mode is the main clock mode. When the main clock mode is switched to PLL clock mode, the oscillation stabilization wait time is fixed at 2^{14}/HCLK. When the mode is switched to the sub clock mode or when PLL stop mode is returned to PLL clock mode, the oscillation stabilization wait time uses the specified values in the WS1 and WS0 bits. For PLL clock oscillation stabilization wait time, at least 2^{14} /HCLK is required. Accordingly, when sub clock mode is switched to PLL clock mode, or when the transition to the PLL stop mode occurs, set WS1 and WS0 bits to “10_B” or “11_B”.

Table 5-1. Functions of Bits in Clock Selection Register (CKSCR) (Sheet 2 of 2)

Bit name		Function
bit11	SCS: Sub clock selection bit	This bit specifies selection of the main clock or sub clock as a machine clock. ■ If this bit is “0”, the sub clock is selected. If “1”, the main clock is selected. ■ When this bit is rewritten from “1” to “0”, the mode is switched to the sub clock mode synchronizing the sub clock (approx. 130 μs.). ■ Writing “1” to this bit when it is “0” generates the oscillation stabilization wait time of the main clock. The time-base timer is automatically cleared. ■ Initialized to “1” by all reset factors. Note: Use the sub clock as an operation clock when the sub clock is selected. (The machine clock changes to a frequency of 8 kHz during low-speed oscillation at 32 kHz) If both SCS and MCS are “0”, SCS is assigned with priority, and the sub clock is selected.
bit10	MCS: PLL clock selection bit	This bit specifies selection of the main clock or PLL clock as a machine clock. ■ If this bit is “0”, the PLL clock is selected. If “1”, the main clock is selected. ■ Writing “0” to this bit when it is “1” generates the oscillation stabilization wait time of the PLL clock. The time-base timer is automatically cleared. The interrupt request flag bit (TBOF) of the time-base timer control register (TBTC) is also cleared. ■ When the main clock is switched to PLL clock mode, the oscillation stabilization wait time is fixed at 2^{14}/HCLK. (The oscillation stabilization wait time is about 4.1 ms if the oscillation clock has a frequency of 4 MHz.) When sub clock mode is switched to PLL clock, the oscillation stabilization wait time uses the specified values in the oscillation stabilization wait time selection bits (CKSCR: WS1, WS0). ■ When the main clock is selected, the operation clock is the oscillation clock divided by 2. (The operation clock is 2 MHz if the oscillation clock is 4 MHz.) ■ Initialized to “1” by all reset factors. Note: Writing “0” to the MCS bit when it is “1”, write while the time-base timer interrupt is masked by using the interrupt request enable bit (TBIE) of the TBTC register or the interrupt level mask register (ILM).
bit9, bit8	CS1, CS0: Multiplication rate selection bits	These bits select the multiplication rate of the PLL clocks. One of four multiplication rates can be selected. Initialized to “00_B” by all reset factors. Note: Writing is disabled if the MCS bit or MCM bit is “0”. Rewrite the CS1 and CS0 bits after setting the MCS bit to “1” (main clock mode).

HCLK: Oscillation clock

Configuration of PLL Output Selection Register (PLLOS)

Figure 5-4 shows the configuration of the PLL output selection register (PLLOS). Table 5-2 explains the functions of the bits for the PLL output selection register.

Figure 5-4. Configuration of PLL Output Selection Register (PLLOS)

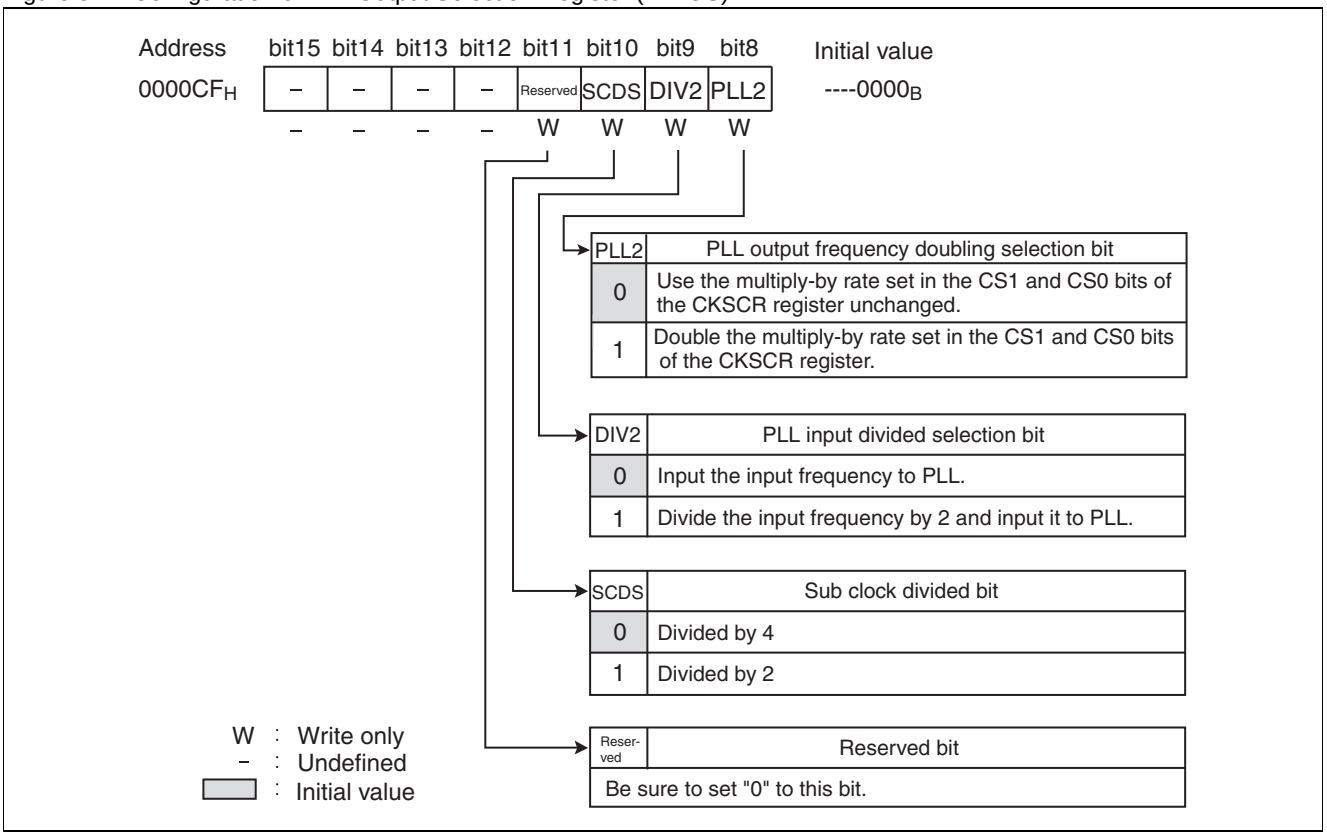


Table 5-2. Functions of Bits for PLL Output Selection Register (PLLOS)

Bit name		Function
bit15 to bit12	Undefined bits	Not used
bit11	Reserved bit	■ Always write “0”. ■ Read value is always “1”.
bit10	SCDS: Sub clock divided bit	This bit selects a division ratio of sub clock. ■ When “0” is written in this bit, divided by 4 is selected. ■ When “1” is written in this bit, divided by 2 is selected. ■ Read value is always “1”. ■ It is initialized to “0” by all reset sources.
bit9	DIV2: PLL input divided selection bit	■ This bit selects dividing of input clock to PLL or input as it is. ■ It is initialized to “0” by all reset sources. ■ Read value is always “1”. ■ Do not change this bit when you use the clock of PLL. ■ For possible settings of PLL multiplication, original oscillation clock, internal clock based on DIV2, PLL2, CS1, and CS0 bits, see Table 5-3 and Table 5-4.
bit8	PLL2: PLL output frequency doubling selection bit	■ This bit specifies doubling of the multiply-by rate for the PLL. ■ Initialized to “0” by all reset sources. ■ Read value is always “1”. Do not change this bit when a PLL clock is being used. ■ For possible settings of PLL multiplication, original oscillation clock, internal clock based on DIV2, PLL2, CS1, and CS0 bits, see Table 5-3 and Table 5-4.

Table 5-3. Example of setting PLLOS bit and CKSCR bit

Original oscillation clock	Internal clock	PLL multiplication	PLLOS, CKSCR bit		
			PLL2	CS1	CS0
4MHz	12MHz	3	0	1	0
4MHz	16MHz	4	0	1	1
4MHz	24MHz	6	1	1	0
4MHz	32MHz	8	1	1	1
6MHz	12MHz	2	0	0	1
6MHz	18MHz	3	0	1	0
6MHz	24MHz	4	0	1	1
8MHz	16MHz	2	0	0	1
8MHz	24MHz	3	0	1	0
8MHz	32MHz	4	1	0	1
16MHz	16MHz	1	0	0	0
16MHz	32MHz	2	1	0	0

* : When DIV2 bit is set to "0".

When setting not listed in Table 5-3, see Table 5-4.

Table 5-4. Relationship between PLL multiplication setting, possible original oscillation clock frequency, and internal clock frequency for PLLOS and CKSCR registers

PLL multiplication	DIV2	PLL2	CS1	CS0	Original oscillation clock	Internal clock
1	0	0	0	0	4MHz to 25MHz	4MHz to 25MHz
1	1	1	0	0	8MHz to 25MHz	8MHz to 25MHz
2	0	0	0	1	3MHz to 12.5MHz	6MHz to 25MHz
2	0	1	0	0	4MHz to 16.5MHz	8MHz to 33MHz
2	1	0	1	1	6MHz to 12.5MHz	12MHz to 25MHz
2	1	1	0	1	6MHz to 16.5MHz	12MHz to 33MHz
3	0	0	1	0	3MHz to 8.33MHz	9MHz to 25MHz
3	1	1	1	0	6MHz to 11MHz	18MHz to 33MHz
4	0	0	1	1	3MHz to 6.25MHz	12MHz to 25MHz
4	0	1	0	1	3MHz to 8.25MHz	12MHz to 33MHz
4	1	1	1	1	6MHz to 8.25MHz	24MHz to 33MHz
6	0	1	1	0	3MHz to 5.5MHz	18MHz to 33MHz
8	0	1	1	1	3MHz to 4.125MHz	24MHz to 33MHz

5.4 Clock Modes

The clock modes are the main clock, PLL clock, and sub clock modes.

Main Clock Mode, PLL Clock Mode, and Sub Clock Mode

Main clock mode

The main clock mode uses a clock obtained by dividing the oscillation clock by two as the operation clock of the CPU and peripheral resources.

PLL clock mode

The PLL clock mode uses the PLL clock obtained as the operation clock of the CPU and peripheral functions. The multiplication rate of the PLL clock can be selected with the clock selection register (CKSCR).

Sub clock mode

The sub clock mode uses a sub clock as the operation clock of the CPU and peripheral resources.

Change of Clock Mode

The clock mode changes to the main clock, PLL clock, or sub clock mode according to the writing of the PLL clock selection bit (MCS) and sub clock selection bit (SCS) in the CKSCR register.

Change from the main clock mode to the PLL clock mode

Rewriting the MCS bit in the CKSCR register from “1” to “0” in the main clock mode changes the main clock to the PLL clock after the end of the oscillation stabilization wait time of the PLL clock (2^{14}/HCLK).

Change from the PLL clock mode to the main clock mode

Rewriting the MCS bit in the CKSCR register from “0” to “1” in the PLL clock mode changes the PLL clock to the main clock adjusted to the timing where the edges of the PLL clock and main clock match (after 1 to 8 PLL clocks).

Change from the main clock mode to the sub clock mode

Rewriting the SCS bit in the CKSCR register from “1” to “0” in the main clock mode changes the main clock to the sub clock synchronizing with the sub clock (approx. 130 μs.).

Change from the sub clock mode to the main clock mode

Rewriting the SCS bit in the CKSCR register from “0” to “1” in the sub clock mode changes the sub clock to the main clock after end of the oscillation stabilization wait time of the main clock. Select the oscillation stabilization wait time by using the oscillation stabilization wait time selection bits (WS1, WS0) of the CKSCR register.

Change from the PLL clock mode to the sub clock mode

Rewriting the sub clock selection bit (SCS) of the clock selection register (CKSCR) from “1” to “0” in the PLL clock mode changes the PLL clock to the sub clock.

Change from the sub clock mode to the PLL clock mode

Rewriting the SCS bit in the CKSCR register from “0” to “1” in the sub clock mode changes the sub clock to the PLL clock after the end of the oscillation stabilization wait time of the main clock. Select the oscillation stabilization wait time by using the oscillation stabilization wait time selection bits (WS1, WS0) of the CKSCR register.

Notes:

- Rewriting the PLL clock selection bit (MCS) or SCS bit in the CKSCR register does not change the machine clock immediately. When operating a resource that depends on a machine clock, make sure that the intended machine clock change has completed by checking the PLL clock display bit (MCM) and sub clock display bit (SCM) in the CKSCR register. Then operate the resource.
- If both of the SCS and MCS bits are "0", SCS is assigned with priority, and the sub clock mode is set.
- When the clock mode is switched, do not switch to low-power consumption mode and other clock mode before this switching is completed. Confirm the completion of clock mode switching by referring to the MCM and SCM bits of the clock selection register (CKSCR). If the mode is switched to other clock mode and low-power consumption mode before completion of switching, the mode may not be switched.

Selection of PLL Clock Multiplication Rate

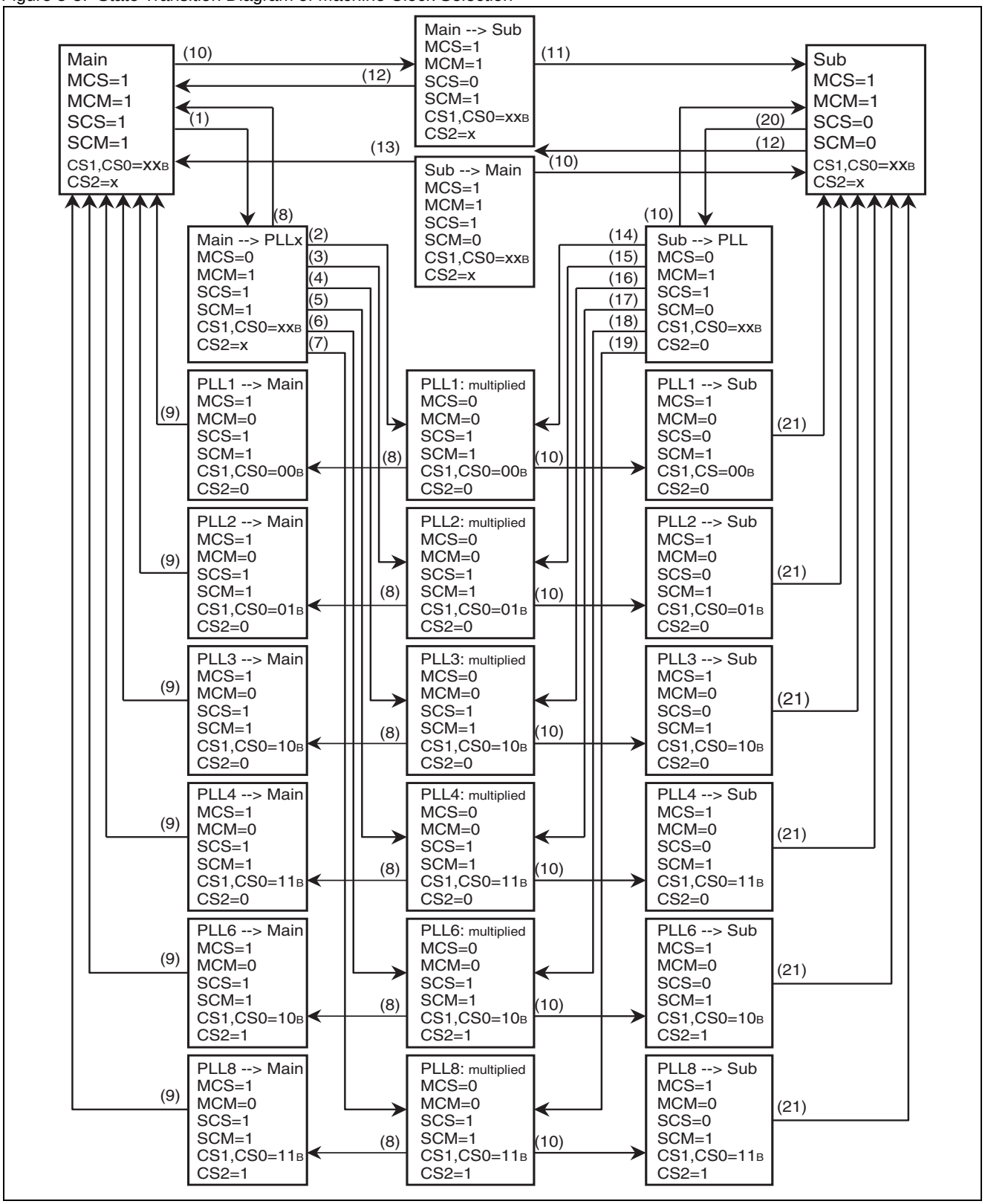
If 00_B to 11_B are written to the CS1 and CS0 bits of the CKSCR register, four types of PLL clock multiply-by rates can be selected: multiply-by 1 to 4.

Machine Clock

The PLL clock, main clock, and sub clock output by the PLL multiplier circuit are machine clocks, which are supplied to the CPU and peripheral functions. The main clock, PLL clock, and sub clock can be selected by writing in the SCS or MCS bit of the CKSCR register.

Figure 5-5 is a state transition diagram of machine clock selection.

Figure 5-5. State Transition Diagram of Machine Clock Selection



(Continued)

(Continued)

(1) Write “0” to MCS bit

(2) End of main clock oscillation stabilization wait time. & CS1, CS0 = 00_B & CS2 = 0

(3) End of main clock oscillation stabilization wait time. & CS1, CS0 = 01_B & CS2 = 0

(4) End of main clock oscillation stabilization wait time. & CS1, CS0 = 10_B & CS2 = 0

(5) End of main clock oscillation stabilization wait time. & CS1, CS0 = 11_B & CS2 = 0

(6) End of main clock oscillation stabilization wait time. & CS1, CS0 = 10_B & CS2 = 1

(7) End of main clock oscillation stabilization wait time. & CS1, CS0 = 11_B & CS2 = 1

(8) Write “1” to MCS bit (includes reset)

(9) Synchronization timing of PLL and main clocks

(10) Write “0” to SCS bit

(11) Synchronization timing of sub clocks and main clocks

(12) SCS bit “1” write (MCS=1)

(13) End of main clock oscillation stabilization wait time.

(14) End of main clock oscillation stabilization wait time. & CS1, CS0 = 00_B & CS2 = 0

(15) End of main clock oscillation stabilization wait time. & CS1, CS0 = 01_B & CS2 = 0

(16) End of main clock oscillation stabilization wait time. & CS1, CS0 = 10_B & CS2 = 0

(17) End of main clock oscillation stabilization wait time. & CS1, CS0 = 11_B & CS2 = 0

(18) End of main clock oscillation stabilization wait time. & CS1, CS0 = 10_B & CS2 = 1

(19) End of main clock oscillation stabilization wait time. & CS1, CS0 = 11_B & CS2 = 1

(20) Write “1” to SCS bit (MCS=0)

(21) Synchronization timing of PLL and sub clocks

MCS:	PLL clock selection bit of clock selection register (CKSCR)
MCM:	PLL clock display bit of clock selection register (CKSCR)
SCS:	Sub clock selection bit of clock selection register (CKSCR)
SCM:	Sub clock display bit of clock selection register (CKSCR)
CS1, CS0:	Multiplication rate selection bits of clock selection register (CKSCR)
CS2:	Multiplication rate selection bit of PLL/sub clock control register (PSCCR)

Note:

The initial value of the machine clocks is the main clock (MCS = 1, SCS = 1).

If both of the SCS and MCS bits are “0”, SCS is assigned with priority and the sub clock is set.

When sub clock mode is switched to PLL clock mode, set “10_B” or “11_B” in the oscillation stabilization wait time selection bits (WS1, WS0) of the CKSCR register.

5.5 Oscillation Stabilization Wait Time

When the power is turned on, when stop mode is released, or switching from the sub clock to the main clock or from sub clock to the PLL clock occurs, an oscillation stabilization wait time is required after oscillation begins because the oscillation clock is stopped. When switching from the main clock to the PLL clock or from the main clock to the sub clock occurs, an oscillation stabilization wait time is required.

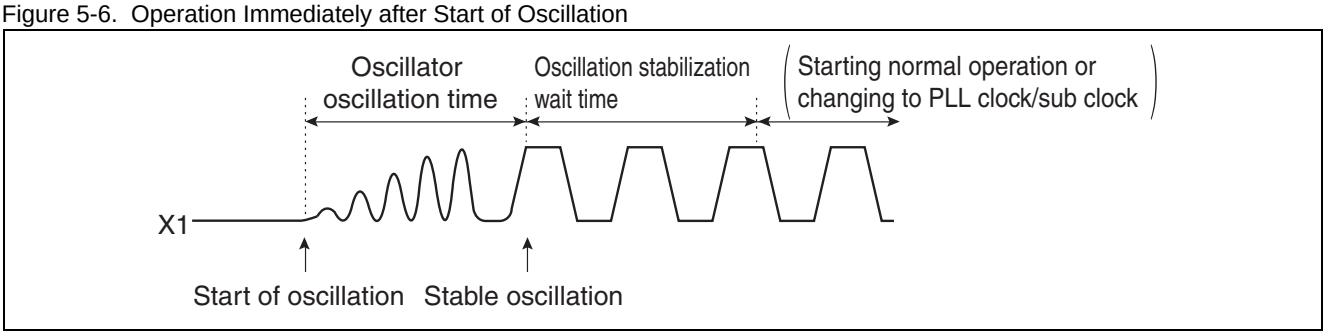
Oscillation Stabilization Wait Time

Ceramic and crystal oscillators generally require a waiting time ranging from several milliseconds to several ten milliseconds after the start of oscillation until oscillation stabilizes to a natural frequency (oscillation frequency). Therefore, disable CPU operation immediately after the start of oscillation, and supply a clock to the CPU when oscillation completely stabilizes following the elapse of the oscillation stabilization wait time. Specify a oscillation stabilization wait time suitable for the oscillator used because the time required for oscillation to stabilize varies depending on the type of oscillator (crystal, ceramic, or other material). The oscillation stabilization wait time can be selected by defining the clock selection register (CKSCR).

When the clock mode is switched from the main clock to the PLL clock, the main clock to the sub clock, the sub clock to the main clock, or the sub clock to the PLL clock, the CPU runs in the clock mode set before switching. After the oscillation stabilization wait time, the CPU changes to the selected clock mode.

Figure 5-6 illustrates operation immediately after the start of oscillation.

Figure 5-6. Operation Immediately after Start of Oscillation



5.6 Connecting Oscillator and External Clock

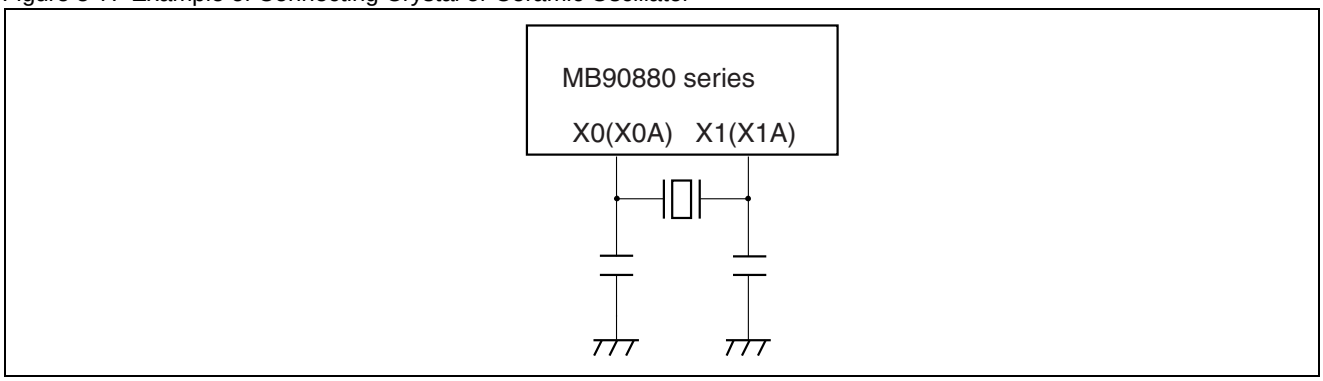
MB90880 series contain a system clock generator circuit and generate clocks using an externally connected oscillator. Also, an external clock can be input to it.

Connection of Oscillator and External Clock

Example of connecting crystal or ceramic oscillator

Connect a crystal oscillator or a ceramic oscillator as shown in the example in Figure 5-7.

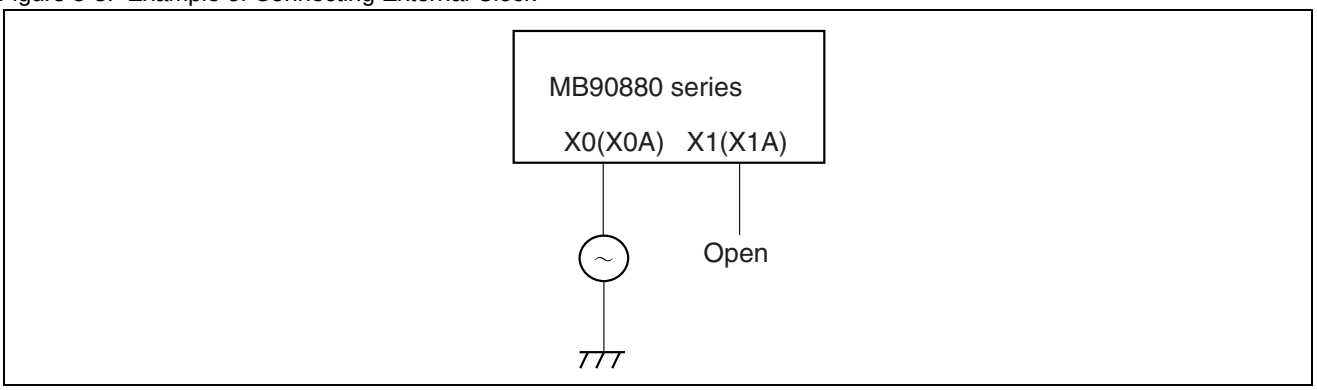
Figure 5-7. Example of Connecting Crystal or Ceramic Oscillator



Example of connecting external clock

Connect an external clock to X0 pin and set up X1 pin to be open as shown in the example in the Figure 5-8.

Figure 5-8. Example of Connecting External Clock



6. Low-power Consumption Mode

This chapter explains the low-power consumption mode of the MB90880 series.

6.1 Overview of Low-Power Consumption Mode

6.2 Block Diagram of 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 Transition Diagram in Standby Mode

6.7 Pin State in Standby Mode, Hold, and Reset

6.8 Caution on Using Low-Power Consumption Mode

6.1 Overview of Low-Power Consumption Mode

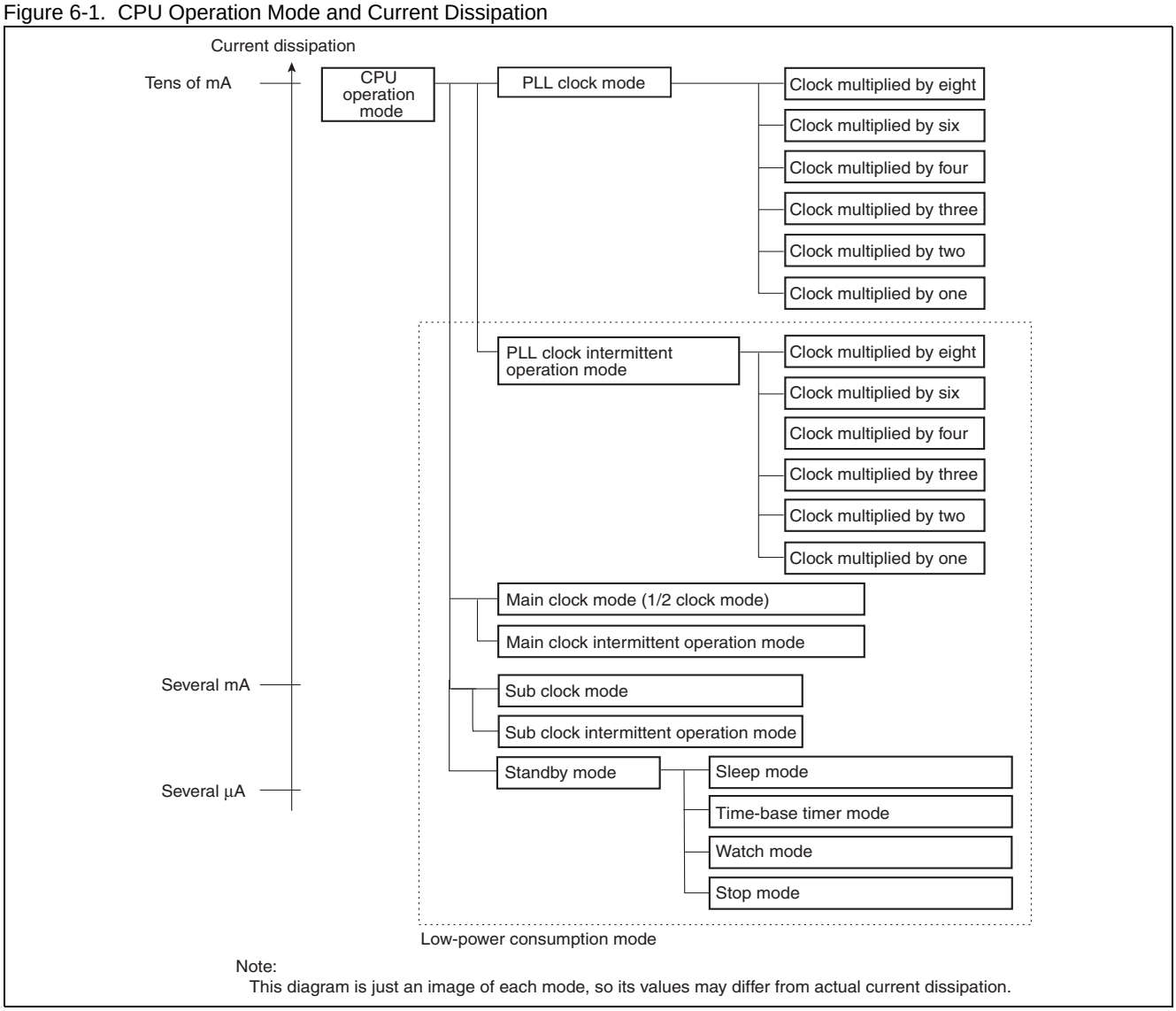
The low-power consumption mode reduces power dissipation by selecting a suitable operation clock and by controlling clock oscillation.
MB90880 series has the following low-power consumption mode.

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

CPU Operation Mode and Current Dissipation

Figure 6-1 illustrates the relationship between CPU operation mode and current dissipation.

Figure 6-1. CPU Operation Mode and Current Dissipation



Note:
This diagram is just an image of each mode, so its values may differ from actual current dissipation.

Clock Modes

PLL clock mode

In PLL clock mode, the CPU and peripheral function operate on a PLL multiplying clock of oscillation clock (HCLK).

Main clock mode

This mode operates the CPU and peripheral functions by using the clock of the oscillation clock (HCLK) divided by two. The PLL multiplier circuit stops its operation in the main clock mode.

Sub clock mode

This mode operates the CPU and peripheral functions by using the sub clock (SCLK). The main clock and the PLL multiplier circuit stop their operation in the sub clock mode.

Reference:

The clock mode includes the PLL clock mode except the low power consumption mode. Refer to Section "5.4 Clock Modes" for more information on clock modes.

CPU Intermittent Operation Mode

The CPU intermittent operation mode (PLL clock intermittent operation mode, main clock intermittent operation mode, and sub clock intermittent operation mode) operates the CPU intermittently while supplying a high-speed clock to the peripheral functions, thereby reducing power dissipation.

Standby Mode

The standby mode reduces power dissipation by stopping supply of a clock to the CPU by using the low-power consumption control circuit (sleep mode), stopping supply of a clock to the CPU and peripheral functions (time-base timer mode), and stopping oscillation clocks (stop mode).

Sleep mode

The sleep mode stops the CPU operation clock in each clock mode. The CPU stops, and the peripheral functions operate with the clock before the sleep mode shifts.

The clock mode when shifting to sleep mode divides into the main sleep mode, PLL sleep mode, and the sub sleep mode.

Time-base timer mode

The time-base timer mode stops operations other than the oscillation clock, time-base timer, and watch timer. Functions other than the time-base timer and watch timer are stopped.

Watch mode

The watch mode operates only the watch timer. In this mode, only the sub clock operates. The main clock and PLL multiplier circuit are stopped.

Stop mode

The stop mode stops source oscillation, and all functions are stopped. In the stop mode, data can be retained with the lowest power dissipation.

6.2 Block Diagram of Low-Power Consumption Control Circuit

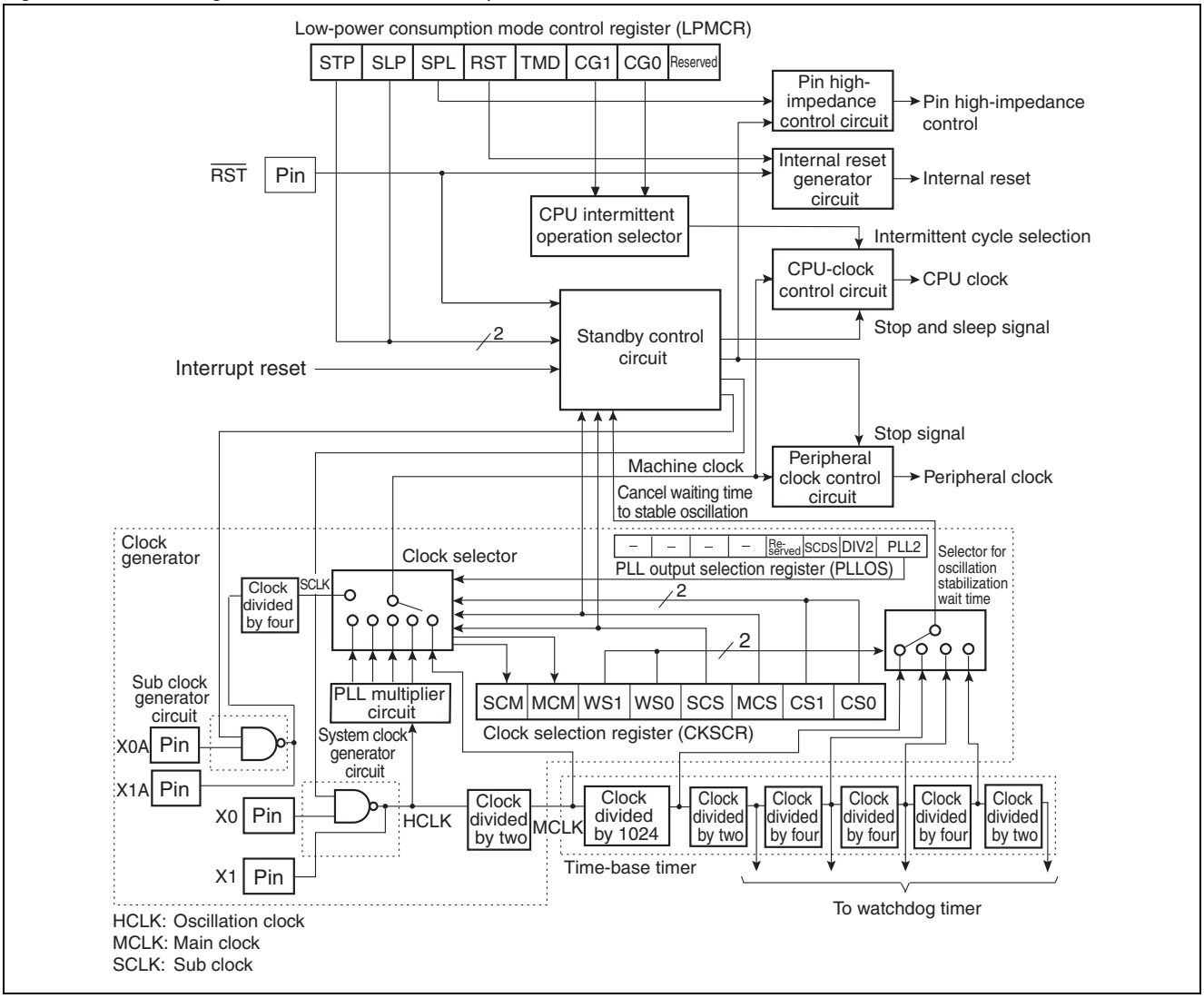
The low-power consumption control circuit is composed of the following seven blocks:

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

Block Diagram of Low-Power Consumption Control Circuit

Figure 6-2 shows the block diagram of the low-power consumption control circuit.

Figure 6-2. Block Diagram of Low-Power Consumption Control Circuit



CPU intermittent operation selector

The CPU intermittent operation selector selects the number of pause cycles in the CPU intermittent operation mode.

Standby control circuit

The standby control circuit controls the CPU-clock control circuit and peripheral clock control circuit for resetting and changing to the low-power consumption mode.

CPU-clock control circuit

The CPU-clock control circuit controls clocks supplied to the CPU.

Peripheral clock control circuit

The peripheral clock control circuit controls clocks supplied to peripheral functions.

Pin high-impedance control circuit

The pin high-impedance control circuit changes the states of external pins to high impedance in the time-base timer mode and stop mode. In the stop mode, the circuit invalidates pull-up resistance with pins for which the pull-up option is selected.

Internal reset generator circuit

The internal reset generator circuit generates an internal reset signal.

Low-power consumption mode control register (LPMCR)

The low-power consumption mode control register (LPMCR) performs functions, such as resetting and changing to the standby mode and defining the CPU intermittent operation function.

6.3 Low-Power Consumption Mode Control Register (LPMCR)

This section explains the registers used for low-power consumption mode settings.

Low-Power Consumption Mode Control Register (LPMCR)

Figure 6-3 shows the configuration of the low-power consumption mode control register (LPMCR).

Figure 6-3. Configuration of Low-Power Consumption Mode Control Register (LPMCR)

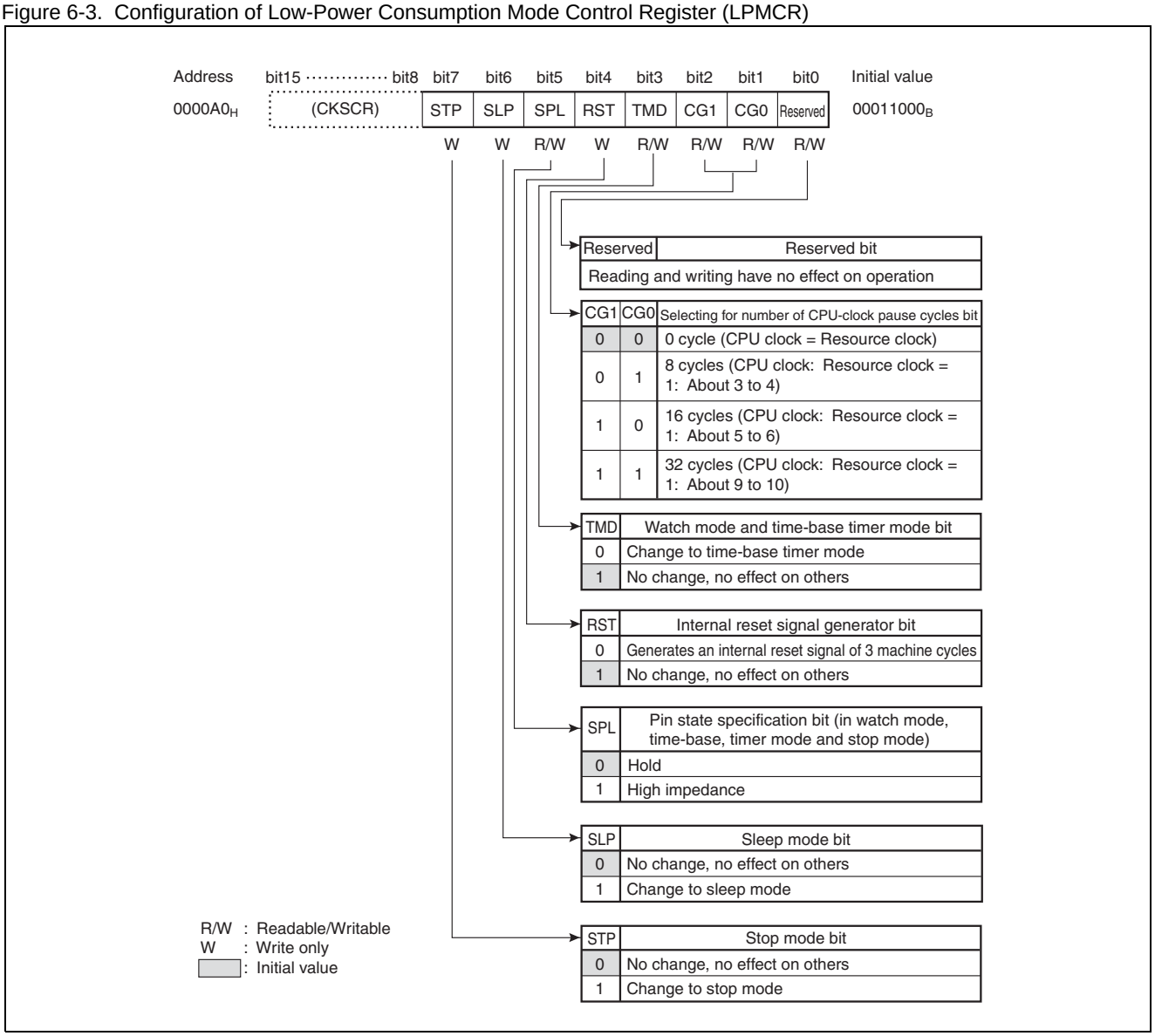


Table 6-1. Functions of Bits in Low-Power Consumption Mode Control Register (LPMCR)

Bit name		Function
bit7	STP: Stop mode bit	This bit instructs a change to the stop mode. ■ Write “1” to this bit to change the mode to the stop mode. ■ Writing “0” to this bit does not affect operation. ■ Initialized to “0” by a reset or if an interrupt request is generated. ■ “0” is always read when this bit is read.
bit6	SLP: Sleep mode bit	This bit instructs a change to the sleep mode. ■ Write “1” to this bit to change the mode to the sleep mode. ■ Writing “0” to this bit does not affect operation. ■ Initialized to “0” by a reset or if an interrupt request is generated. ■ “0” is always read when this bit is read.
bit5	SPL: Pin state specification bit (in watch mode, time-base timer mode, and stop mode)	This bit sets the state of external pins only in the watch, time-base timer, and stop modes. ■ If this bit is “0”, the levels of external pins are retained. ■ If this bit is “1”, external pins are changed to high impedance. ■ Initialized to “0” by a reset.
bit4	RST: Internal reset signal generator bit	This bit generates the software reset. ■ Write “0” to this bit to generate an internal reset signal of 3 machine cycles. ■ Writing “1” to this bit does not affect operation. ■ “1” is always read when this bit is read.
bit3	TMD: Watch mode and time-base timer mode bit	This bit instructs a change to the watch or time-base timer mode. ■ Write “0” to this bit at the main clock or PLL clock mode to change the mode to the time-base timer mode. ■ Write “0” to this bit at the sub clock mode to change the mode to the watch mode. ■ Initialized to “1” by a reset or if an interrupt request is generated. ■ “1” is always read when this bit is read.
bit2, bit1	CG1, CG0: Selecting for number of CPU-clock pause cycles bit	These bits specify the number of pause cycles of the CPU clock in the CPU intermittent operation function. ■ Stops supply of CPU clocks for the specified number of cycles per instruction. ■ Capable selected from four types of clocks. ■ Initialized to “00_B” by a reset.
bit0	Reserved: Reserved bit	Reading and writing have no effect on operation.

Accessing Low-Power Consumption Mode Control Register

Writing in the low-power consumption mode control register executes a change to the standby mode (stop, sleep, time-base timer and watch modes). Use the instructions listed in Table 6-2.

The low-power consumption mode transition instruction in Table 6-2 must always be followed by an array of instructions highlighted by a line below.

```
MOV   LPMCR,#H'xx    ; The low-power consumption mode transition instruction in Table 6-2.
  NOP
  NOP
  JMP    $+3              ; Jump to next instruction

 MOV    A,#H'10          ; Any instruction
```

The devices does not guarantee its operation after releasing from the standby mode if you place an array of instructions other than the one enclosed in the dotted line.

To access the low-power consumption mode control register (LPMCR) with C language, refer to " Notes on Accessing the Low-Power Consumption Mode Control Register (LPMCR) to Enter the Standby Mode" in "6.8 Caution on Using Low-Power Consumption Mode". When writing to the low-power consumption mode control register with a length of words, use even addresses only. Performing transition by using an odd address for writing may result in operation errors.

Any instruction may be used to control functions not listed in Table 6-1.

Table 6-2. Instructions Used for Change 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	

Priority of STP, SLP, and TMD Bits

Requests are processed with the following order of priority in the event that stop mode, sleep mode, and time-base timer mode requests are issued simultaneously.

Stop mode request > Time-base timer mode request > Sleep mode request

6.4 CPU Intermittent Operation Mode

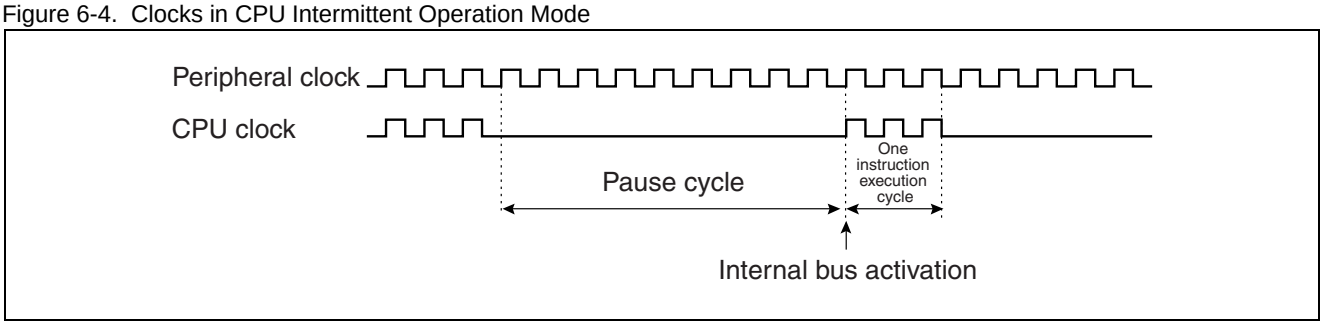
The CPU intermittent operation mode reduces power dissipation by intermittently operating the CPU while operating external buses and peripheral functions at high speeds.

CPU Intermittent Operation Mode

To delay activation of the internal bus cycle, the CPU intermittent operation mode (PLL clock intermittent operation mode, main clock intermittent operation mode, and sub clock intermittent operation mode) stops clocks supplied to the CPU for a preset period for each instruction during access to registers, embedded memory (ROM or RAM), I/O, peripheral functions, and external buses. Low-power dissipation processing is possible by lowering the CPU execution speed while high-speed peripheral clocks are supplied to peripheral functions.

- Select the number of clock pause cycles supplied to the CPU using Selecting for number of CPU-clock pause cycles bit (CG1 and CG0) of the low-power consumption mode control register (LPMCR).
- Use the same clock as that for the peripheral functions when operating external buses.
- The instruction execution time when the CPU intermittent operation mode is set can be calculated by dividing the number of instruction executions for accessing registers, embedded memory, embedded peripheral functions, and external buses by the number of pauses cycles. The correction value thus obtained is added to the usual execution time. Figure 6-4 illustrates operation clocks in the CPU intermittent operation mode.

Figure 6-4. Clocks in CPU Intermittent Operation Mode



6.5 Standby Mode

Standby mode reduces power dissipation by stopping supplying an operation clock to the CPU or peripheral function and stopping oscillation clock.
The standby mode is divided into four modes, namely, the sleep (PLL sleep, main sleep, and sub sleep), time-base timer, watch, and stop modes.

Operational States in Standby Mode

Table 6-3 lists operational states in the standby mode.

Table 6-3. Operational States in Standby Mode

Standby mode		Change condition	Main clock	Sub clock	Machine clock	CPU	Peripheral	Pin	Cancellation method
Sleep mode	PLL sleep mode	SCS = 1 MCS = 0 SLP = 1	In operation	In operation	In operation	Stopped	In operation	In operation	Reset or interrupt
	Main sleep mode	SCS = 1 MCS = 1 SLP = 1							
	Sub sleep mode	SCS = 0 SLP = 1	Stopped						
Time-base timer mode	Time-base timer mode (SPL = 0)	SCS = 1 TMD = 0	In operation		Stopped		Stopped *1	Hold	
	Time-base timer mode (SPL = 1)	SCS = 1 TMD = 0						Hi-Z	
Watch mode	Watch mode (SPL = 0)	SCS = 0 TMD = 0	Stopped				Stopped *2	Hold	
	Watch mode (SPL = 1)	SCS = 0 TMD = 0						Hi-Z	
Stop mode	Stop mode (SPL = 0)	STP = 1		Stopped			Stopped	Hold	
	Stop mode (SPL = 1)	STP = 1						Hi-Z	

*1: The time-base timer and watch timer are operating.

*2: The watch timer is operating

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

SLP: Sleep mode bit of low-power consumption mode control register (LPMCR)

STP: Stop mode bit of low-power consumption mode control register (LPMCR)

TMD: Watch/time-base timer mode bit of low-power consumption mode control register (LPMCR)

MCS: PLL clock selection bit of the clock selection register (CKSCR)

SCS: Sub clock selection bit of the clock selection register (CKSCR)

Hi-Z: High impedance

$\overline{RST}$: External-reset pin

6.5.1 Sleep Mode

The sleep mode stops CPU operation clocks. The CPU stops, and the peripheral functions operate with the clock before the sleep mode shifts.

Change to Sleep Mode

Writing “1” in the sleep mode bit (SLP), “1” in the watch/time-base timer mode bit (TMD), and “0” in the stop mode bit (STP) of the low-power consumption mode control register (LPMCR) changes the mode to the sleep mode.

Note:

If “1” is simultaneously written in the SLP and STP bits of the LPMCR register, the STP bit has the priority and the device is changed to the stop mode.

If writing “1” in the SLP bit and writing “0” in the TMD bit of the low-power consumption mode control register are performed at the same time, the TMD bit has the priority and the device is changed to the time-base timer mode or watch mode.

Data hold function

This function in the sleep mode holds data of the internal RAM and dedicated registers such as an accumulator.

Hold function

The external bus hold function operates in the sleep mode. A hold state is set if a hold request is issued.

Operation during interrupt request

The sleep mode is not set if an interrupt request is issued while “1” is written in the SLP bit of the LPMCR register. The CPU executes a next instruction if an interrupt is not accepted. If the CPU can accept an interrupt, the request is immediately branched to an interrupt processing routine.

Pin state

In the sleep mode, the previous states are maintained except for pins used for bus input and output or for bus control.

Canceling the Sleep Mode

The low-power consumption control circuit cancels the sleep mode by input of a reset or by an interrupt.

Return by a reset

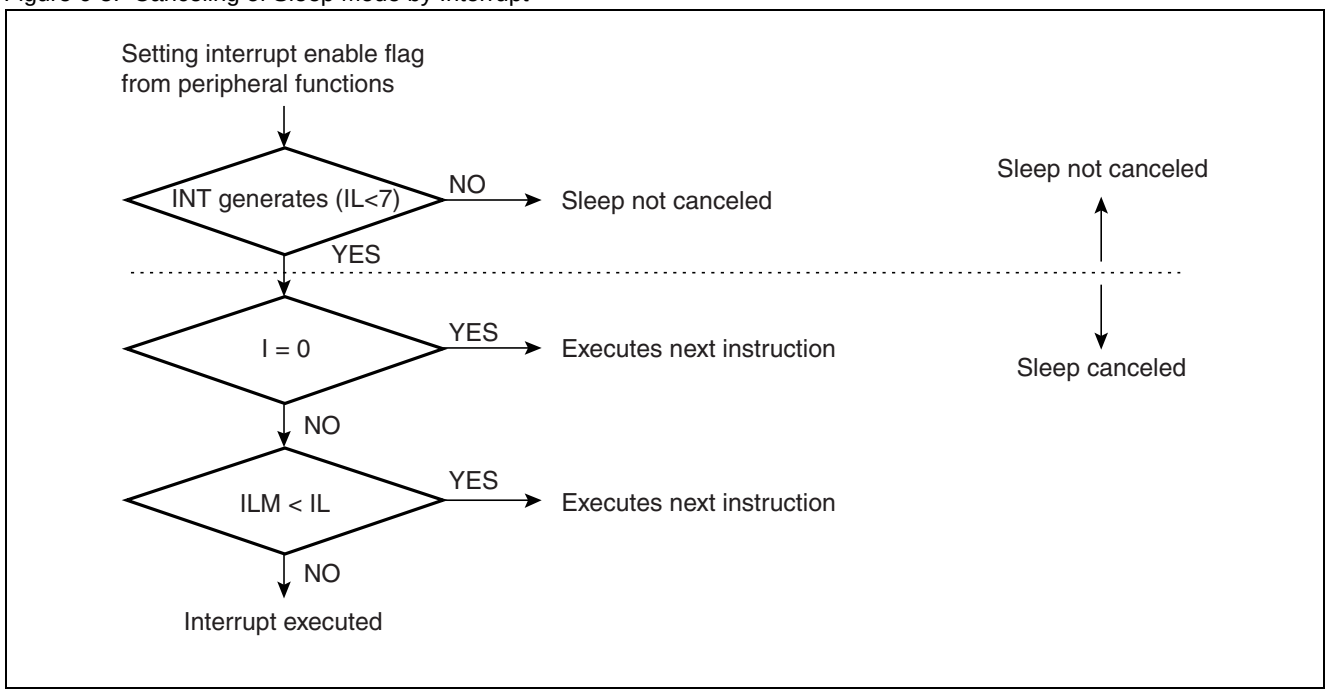
Reset initializes to the main clock mode.

Return by interrupt

The sleep mode is canceled if an interrupt request whose interrupt level is higher than 7 is generated in a peripheral circuit, etc., in the sleep mode. After the sleep mode is canceled, the interrupt requests are judged by setting the I-flag of the condition code register (CCR), interrupt level mask register (ILM), or interrupt control register (ICR) as in the case with ordinary interrupt processing. If the interrupts can be accepted, the CPU executes the interrupts. If the interrupts cannot be accepted, the CPU continues processing beginning from an instruction next to the instruction specifying the sleep mode.

Figure 6-5 illustrates the canceling of the sleep mode by an interrupt.

Figure 6-5. Canceling of Sleep Mode by Interrupt



Note:

When executing an interrupt, an instruction next to the instruction that specified the sleep mode is normally executed first before an interrupt is processed. If a change to the sleep mode occurs at the same time as an external bus hold request is received, an interrupt may be executed first before the next instruction is executed.

6.5.2 Time-base Timer Mode

The time-base timer mode stops operations except for oscillation clock, time-base timer and watch timer. All functions except the time-base timer and watch timer are stopped.

Change to Time-base Timer Mode

To change the mode to the time-base timer mode, write “0” in watch/time-base timer mode bit (TMD) of the low-power consumption mode control register (LPMCR) in the PLL clock mode or the main clock mode (sub clock display bit (SCM) of the clock selection register (CKSCR) = 1).

Data hold function

This function in the time-base timer mode holds data of the internal RAM and dedicated registers such as an accumulator.

Hold function

In the time-base timer mode, the external bus hold function is stopped and hold requests cannot be accepted even if they are input. If a hold request is input during a change to the time-base timer mode, the level of the $\overline{\text{HAK}}$ signal may not change to “L” while the bus is set to the high-impedance state.

Operation during interrupt request

The time-base timer mode is not set if an interrupt request is issued while “0” is written to the TMD bit of the low-power consumption mode control register (LPMCR).

Pin state

Pin state specification bit (SPL) of the LPMCR register can control whether to maintain the state of an external pin in the time-base timer mode in the previous state or in the high-impedance state.

Canceling the Time-base Timer Mode

The low-power consumption control circuit cancels the time-base timer mode by input of a reset or by an interrupt request.

Return by external reset

If the time-base timer mode is canceled by external reset, initializes to the main clock mode.

Return by interrupt

The time-base timer mode is canceled by the low-power consumption control circuit if an interrupt request whose interrupt level is higher than 7 (other than IL2, IL1, and IL0 of the interrupt control register (ICR)=111_B) is generated in a peripheral circuit, etc., in the time-base timer mode.
After the time-base timer mode is canceled, the interrupt requests are judged by setting the I-flag of the condition code register (CCR), interrupt level mask register (ILM), or interrupt control register (ICR), as in the case with ordinary interrupt processing. If the interrupts can be accepted, the CPU executes the interrupts. If an interrupt cannot be accepted, the CPU continues processing beginning from an instruction that was processed before the time-base timer mode was set.

Note:

When executing an interrupt, an instruction next to the instruction specifying the time-base timer mode is normally executed first before an interrupt is processed. If a change to the time-base timer mode occurs at the same time as an external bus hold request is received, an interrupt may be executed first before the next instruction is executed.

6.5.3 Watch Mode

The watch mode stops operations other than those of the sub clock and watch timer.

Change to Watch Mode

To change the mode to the watch mode, write “0” in watch/time-base timer mode bit (TMD) of the low-power consumption mode control register (LPMCR) in the sub clock mode (sub clock display bit (SCS) of the clock selection register (CKSCR) = 0).

Data hold function

This function in the watch mode holds data of the internal RAM and dedicated registers such as an accumulator.

Hold function

In the watch mode, the external bus hold function is stopped and hold requests cannot be accepted even if they are input.

Note:

If a hold request is input during a change to the watch mode, the level of the $\overline{\text{HAK}}$ signal may not change to “L” while the bus is set to the high-impedance state.

Operation during interrupt request

The watch mode is not set if an interrupt request is issued while “0” is set in the TMD bit of the LPMCR register.

Pin state setting

Pin state specification bit (SPL) of the LPMCR register can control whether to maintain the state of an external pin in the watch mode in the previous state or in the high-impedance state.

Canceling the Watch Mode

The low-power consumption control circuit cancels the watch mode by input of a reset or by an interrupt request.

Return by a reset

When the watch mode is canceled by a reset factor, the watch mode is canceled first, and a reset state for standing by for stable oscillation is set. The sequence for resetting is executed after the end of the oscillation stabilization wait time.

Return by interrupt

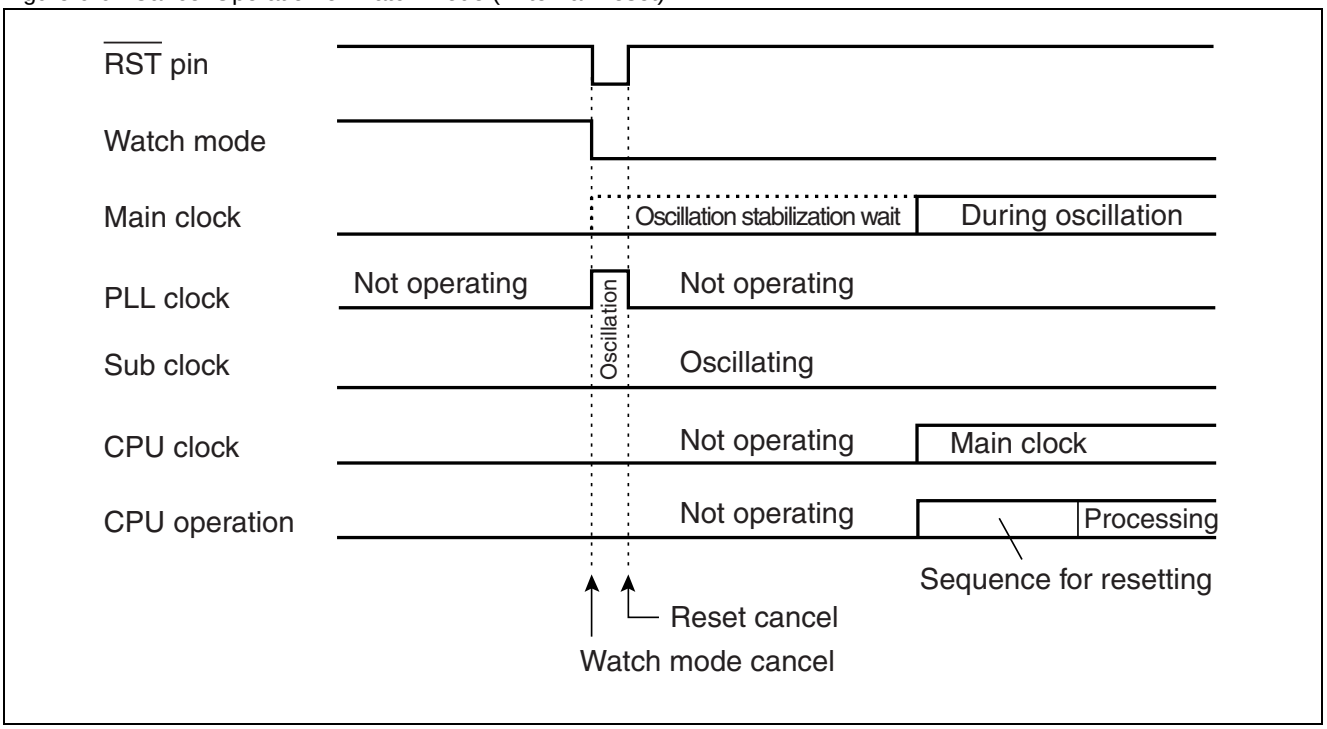
The watch mode is canceled by the low-power consumption control circuit if an interrupt request whose interrupt level is higher than 7 (other than IL2, IL1, and IL0 of the interrupt control register (ICR)=111_B) is generated in a peripheral circuit, etc., in the watch mode. The mode immediately changes to the sub clock mode. After the change to the sub clock mode, the interrupt requests are judged by setting the I-flag of the condition code register (CCR), interrupt level mask register (ILM), or interrupt control register (ICR), as in the case with ordinary interrupt processing. If the interrupts can be accepted, the CPU executes the interrupts. If an interrupt cannot be accepted, the CPU continues processing beginning from an instruction next to the instruction that was processed before the watch mode was set.

Note:

When executing an interrupt, an instruction next to the instruction specifying the watch mode is normally executed first before an interrupt is processed. If a change to the watch mode occurs at the same time as an external bus hold request is received, an interrupt may be executed first before the next instruction is executed.

Figure 6-6 illustrates the cancel operation of the watch mode.

Figure 6-6. Cancel Operation of Watch Mode (External Reset)



6.5.4 Stop Mode

The stop mode stops source oscillation and stops all functions, thereby enabling retention of data with the lowest power dissipation.

Change to Stop Mode

Write “1” in the stop mode bit (STP) of the low-power consumption mode control register (LPMCR) to change the mode to the stop mode.

Data hold function

This function in the stop mode holds data of the internal RAM and dedicated registers such as an accumulator.

Hold function

In the stop mode, the external bus hold function is stopped and hold requests cannot be accepted even if they are input. If a hold request is input during a change to the stop mode, the level of the $\overline{HAK}$ signal may not change to “L” while the bus is set to the high-impedance state.

Operation during interrupt request

The stop mode is not set if an interrupt request is issued while “1” is set in the STP bit of the LPMCR register.

Pin state setting

Pin state specification bit (SPL) of the LPMCR register can specify whether to maintain the state of an external pin in the stop mode in the previous state or in the high-impedance state.

Canceling the Stop Mode

The low-power consumption control circuit releases the stop mode when a reset is input or an interrupt occurs. Because the oscillation clock (HCLK) and sub clock (SCLK) are halted, the stop mode is released after the oscillation stabilization wait time of the main clock or sub clock.

Return by a reset

When the stop mode is canceled by a reset factor, the stop mode is canceled first and the reset state for standing by for stable oscillation is set. The sequence for resetting is executed after the end of the oscillation stabilization wait time.

Return by interrupt

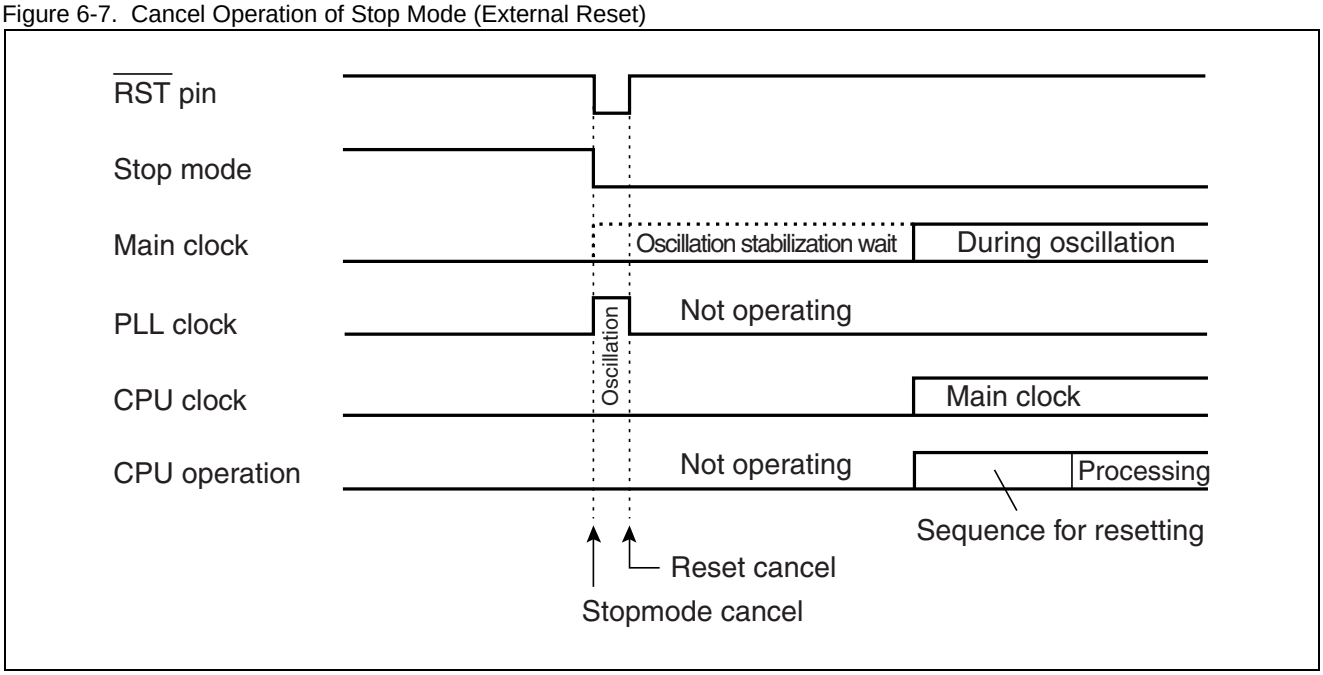
The stop mode is canceled by the low-power consumption control circuit if an interrupt request whose interrupt level is higher than 7 (other than IL2, IL1, and IL0 of the interrupt control register (ICR)=111_B) is generated in a peripheral circuit, etc., in the stop mode. After the stop mode is canceled, oscillation stabilization wait time for the main clock specified by the oscillation stabilization wait time selection bits (WS1, WS0) of the clock selection register (CKSCR) is elapsed. Then the interrupt requests are judged by setting the I-flag of the condition code register (CCR), interrupt level mask register (ILM), or interrupt control register (ICR) as in the case with ordinary interrupt processing. If the interrupts can be accepted, the CPU executes interrupts. If an interrupt cannot be accepted, the CPU continues processing beginning from an instruction next to the instruction that was processed before the stop mode was set.

Notes:

- When executing an interrupt, an instruction next to the instruction that specified the stop mode is normally executed first before an interrupt request is processed. If a change to the stop mode occurs at the same time as an external bus hold request is received, an interrupt may be executed first before the next instruction is executed.
- In PLL stop mode, the main clock and PLL multiplication circuit stop. During recovery from PLL stop mode, it is necessary to allot the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time. The oscillation stabilization wait times for the main clock and PLL clock are counted simultaneously according to the value specified in the oscillation stabilization wait time selection bits in the clock selection register (CKSCR: WS1, WS0). The oscillation stabilization wait time selection bits in the clock selection register (CKSCR: WS1, WS0) must be selected accordingly to account for the longer of main clock and PLL clock oscillation stabilization wait time. The PLL clock oscillation stabilization wait time, however, requires 2^{14}/HCLK or more. Set the oscillation stabilization wait time selection bits in the clock selection register (CKSCR: WS1, WS0) to "10_B" or "11_B".

Figure 6-7 shows the cancel operation of the stop mode.

Figure 6-7. Cancel Operation of Stop Mode (External Reset)



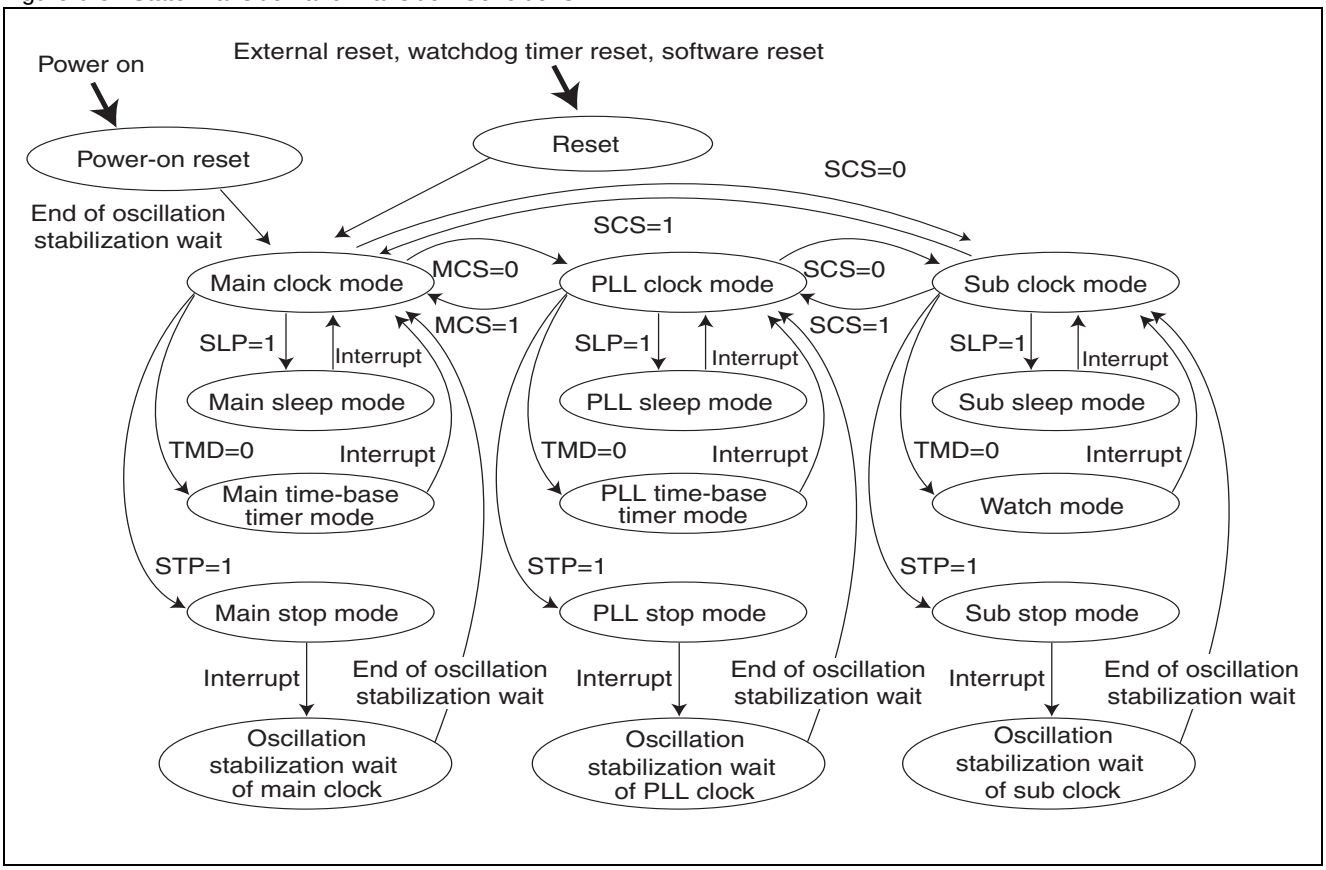
6.6 State Transition Diagram in Standby Mode

This section explains the transition of operational states for the MB90880 series and describes the transition conditions.

State Transition Diagram

Figure 6-8 illustrates the transition of operational states for the MB90880 series and the transition conditions.

Figure 6-8. State Transition and Transition Conditions



Operational State in Low-Power Consumption Mode

Table 6-4 lists operational states in the low-power consumption mode.

Table 6-4. Operational States in Low-Power Consumption Mode

Operational state	Main clock	Sub clock	PLL clock	CPU	Peripheral	Watch	Time-base timer	Clock source
PLL clock mode	Operating	Operating	Operating	Operating	Operating	Operating	Operating	PLL clock
PLL sleep mode				Not operating				
PLL time-base timer mode					Not operating			
PLL stop mode	Not operating	Not operating	Not operating			Not operating	Not operating	
Oscillation stabilization wait of PLL clock mode	Operating	Operating	Operating			Operating	Operating	
Main clock mode	Operating	Operating	Not operating	Operating	Operating	Operating	Operating	Main clock
Main sleep mode				Not operating				
Main time-base timer mode					Not operating			
Main stop mode	Not operating	Not operating				Not operating	Not operating	
Oscillation stabilization wait of main clock	Operating	Operating				Operating	Operating	
Sub clock mode	Not operating	Operating	Not operating	Operating	Operating	Operating	Not operating	Sub clock
Sub sleep mode				Not operating				
Watch mode					Not operating			
Sub stop mode		Not operating				Not operating		
Oscillation stabilization wait of sub clock		Operating				Operating		
Power-on reset	Operating	Operating	Not operating	Not operating	Not operating	Operating	Operating	Main clock
Reset			Operating					

6.7 Pin State in Standby Mode, Hold, and Reset

The states of the pins in the standby mode and in the hold and reset states are described for each memory access mode.

Pin State in Single-Chip Mode

Table 6-5 lists the pin states in the single-chip mode.

Table 6-5. Pin States in Single-Chip Mode

Pin name	In sleep state	In stop state		In hold state	When reset
		SPL = 0	SPL = 1		
P07 to P00	Maintains the previous state *2	Input cutoff/Maintains the previous state *2, *3	Input cutoff/Output Hi-Z *3	This state does not exist	Input disabled/Output Hi-Z
P17 to P10					
P27 to P20					
P37 to P30					
P47 to P40					
P57 to P50					
P67 to P60					
P76 to P70					
P87 to P80					
P97 to P90 PA3 to PA0					
When used as External Interrupt Pin (IRQX)	Input enabled *1	Input enabled *1			Input disabled

*1: Same as in other ports when used in the output state. “Input enabled” means that input functions are ready and require pull up/pull down or external input.

*2: The state output immediately before this mode was set is output as it is. If it is input, input is disabled.
“Input disabled” means that operations of input gates located very close to the pins are enabled, but pin states cannot be accepted internally because internal circuits are not operating.

*3: In the input cutoff state, the input is masked and the “L” level is passed to the internal. Output Hi-Z means to disable the transistors to drive the pin, and make the pin to be high-impedance.

Pin States in External Bus 16-Bit Data Bus Mode and Multiplex 16-Bit External Bus Mode

Table 6-6 summarizes pin states in the external bus 16-bit data bus mode and multiplex 16-bit external bus mode.

Table 6-6. Pin States in External Bus 16-Bit Data Bus Mode and Multiplex 16-Bit External Bus Mode

Pin name	In sleep state	In stop state		In hold state	When reset	Internal ROM access immediately after reset cancellation	Internal ROM access after external ROM access
		SPL = 0	SPL = 1				
P07 to P00 (AD07 to AD00)	Input disabled/ Output Hi-Z	Input cutoff/ Output Hi-Z	Input cutoff/ Output Hi-Z *6	Input disabled/ Output Hi-Z	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/Input enabled
P17 to P10 (AD15 to AD08)							
P27 to P20 (A23 to A16)	Output state *1, *4	Output state *1, *4		Input disabled/ Output Hi-Z *4	Output state *1	Output state *1	Maintains the previous address
P57 (CLK)	Input disabled/ Output enabled *2, *4	Input disabled/ Output state *1, *4		Input disabled/ Output enabled *2, *4	Input disabled/Output enabled *2	CLK output	CLK output
P56 (RDY)	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5, *6		Input disabled *4	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P55 ($\overline{HAK}$)				"L" output			
P54 (HRQ)				"H" input			
P53 ($\overline{WRH}$)	"H" output *4	"H" output *4		Input disabled/ Output Hi-Z *4	"H" output	"H" output	"H" output
P52 ($\overline{WRL}$)							
P51 ($\overline{RD}$)	"H" output	"H" output		Input disabled/Output Hi-Z			
P50 (ALE)	"L" output	"L" output			Output enabled *2	"L" output	"L" output
P37 to P30	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5		Maintains the previous state *5	Input disabled/Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/Input enabled
P47 to P40							
P17 to P10							
P67 to P60							
P76 to P70							
P97 to P91							
PA3 to PA0							
P90 (CS0)					"H" output		
When used as External Interrupt Pin (IRQX)	Input enabled *3	Input enabled *3	Input enabled *3		Input disabled		

*1: "Output state" means that the pin-drive transistors are set in a drive-enabled state, but internal circuits are stopped so that a fixed value, "H" or "L", is output. When internal peripheral circuits are in operation and output functions are used, output varies except when reset. Output does not vary during a reset.

*2: "Output enabled" means that the pin-drive transistors are set in a drive state and that internal circuit operations are enabled. Operational states are therefore conveyed on the pins.

*3: Same as in other ports when used in the output state. "Input enabled" means that input functions are ready and require pull up/pull down or external input.

*4: The previous values are retained if used as an output port.

*5: The state output immediately before this mode was set is output as it is. If it is input, input is disabled. "Input disabled" means that operations of input gates located very close to the pins are enabled, but pin states cannot be accepted internally because internal circuits are not operating.

*6: In the input cutoff state, the input is masked and the "L" level is passed to the internal. Output Hi-Z means to disable the transistors to drive the pin, and make the pin to be high-impedance.

Pin States in External Bus 8-Bit Data Bus Mode and Multiplex 8-Bit External Bus Mode

Table 6-7 lists pin states in the external bus 8-bit data bus mode and multiplex 8-bit external bus mode.

Table 6-7. Pin States in External Bus 8-Bit Data Bus Mode and Multiplex 8-Bit External Bus Mode

Pin name	In sleep state	In stop state		In hold state	When reset	Internal ROM access immediately after reset cancellation	Internal ROM access after external ROM access
		SPL = 0	SPL = 1				
P07 to P00 (AD07 to AD00)	Input disabled/ Output Hi-Z	Input cutoff/ Output Hi-Z	Input cutoff/ Output Hi-Z *6	Input disabled/ Output Hi-Z	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P17 to P10 (AD15 to AD08)	Output state *1	Output state *1			Output state *1	Output state *1	Maintains the previous address
P27 to P20 (A23 to A16)	Output state *1, *4	Output state *1, *4		Input disabled/ Output Hi-Z *4			
P57 (CLK)	Input disabled/ Output enabled *2, *4	Input disabled/ Output state *1, *4		Input disabled/ Output enabled *2, *4	Input disabled/ Output enabled *2	CLK output	CLK output
P56 (RDY)	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5		Input disabled *4	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P55 ($\overline{HAK}$)				"L" output			
P54 (HRQ)				"H" input			
P53 ($\overline{WRH}$)				Maintains the previous state *5			
P52 ($\overline{WRL}$)	"H" output *5	"H" output *5		Input disabled/ Output Hi-Z *4	"H" output	"H" output	"H" output
P51 ($\overline{RD}$)	"H" output	"H" output		Input disabled/ Output Hi-Z			
P50 (ALE)	"L" output	"L" output			"L" output	"L" output	"L" output
P37 to P30	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5		Maintains the previous state *5	Input disabled/ Output Hi-Z	Output Hi-Z /Input enabled	Output Hi-Z /Input enabled
P47 to P40							
P67 to P60							
P76 to P70							
P97 to P91							
PA3 to PA0							
P90 (CS0)					"H" output		
When used as External Interrupt Pin (IRQX)	Input enabled *3	Input enabled *3	Input enabled *3		Input disabled		

*1: "Output state" means that the pin-drive transistors are set in a drive-enabled state, but internal circuits are stopped so that a fixed value, "H" or "L", is output. When internal peripheral circuits are in operation and output functions are used, output varies except when reset. Output does not vary during a reset.

*2: "Output enabled" means that the pin-drive transistors are set in a drive state and that internal circuit operations are enabled. Operational states are therefore conveyed on the pins.

*3: Same as in other ports when used in the output state. "Input enabled" means that input functions are ready and require pull up/pull down or external input.

*4: The previous values are retained if used as an output port.

*5: The state output immediately before this mode was set is output as it is. If it is input, input is disabled. "Input disabled" means that operations of input gates located very close to the pins are enabled, but pin states cannot be accepted internally because internal circuits are not operating.

*6: In the input cutoff state, the input is masked and the "L" level is passed to the internal. Output Hi-Z means to disable the transistors to drive the pin, and make the pin to be high-impedance.

Pin States in External Bus 16-Bit Data Bus Mode and Non-Multiplex 16-Bit External Bus Mode

Table 6-8 lists pin states in the external bus 16-bit data bus mode and non-multiplex 16-bit external bus mode.

Table 6-8. Pin States in External Bus 16-Bit Data Bus Mode and Non-Multiplex 16-Bit External Bus Mode

<table>
<tr><th rowspan="2">Pin name</th><th rowspan="2">In sleep state</th><th colspan="2">In stop state</th><th rowspan="2">In hold state</th><th rowspan="2">When reset</th><th rowspan="2">Internal ROM access immediately after reset cancellation</th><th rowspan="2">Internal ROM access after external ROM access</th></tr>
<tr><th>SPL = 0</th><th>SPL = 1</th></tr>
<tr><td>P07 to P00 (AD07 to AD00)</td><td rowspan="2">Input disabled/ Output Hi-Z</td><td rowspan="2">Input cutoff/ Output Hi-Z</td><td rowspan="14">Input cutoff/ Output Hi-Z *6</td><td rowspan="3">Input disabled/ Output Hi-Z</td><td>Input disabled/ Output Hi-Z</td><td>Output Hi-Z/ Input enabled</td><td>Output Hi-Z/ Input enabled</td></tr>
<tr><td>P17 to P10 (AD15 to AD08)</td></tr>
<tr><td>P37 to P30 (A07 to A00)</td><td rowspan="2">Output state *1</td><td rowspan="2">Output state *1</td><td rowspan="3">Output state *1</td><td rowspan="3">Output state *1</td><td rowspan="3">Maintains the previous address</td></tr>
<tr><td>P47 to P40 (A15 to A08)</td></tr>
<tr><td>P27 to P20 (A23 to A16)</td><td>Output state *1, *4</td><td>Output state *1, *4</td><td>Input disabled/ Output Hi-Z *4</td></tr>
<tr><td>P57 (CLK)</td><td>Input disabled/ Output enabled *2, *4</td><td>Input disabled/ Output state *1, *4</td><td>Input disabled/ Output enabled *2, *4</td><td>Input disabled/ Output enabled *2</td><td>CLK output</td><td>CLK output</td></tr>
<tr><td>P56 (RDY)</td><td rowspan="3">Maintains the previous state *5</td><td rowspan="3">Input cutoff/ Maintains the previous state *5</td><td>Input disabled *4</td><td rowspan="3">Input disabled/ Output Hi-Z</td><td rowspan="3">Output Hi-Z/ Input enabled</td><td rowspan="3">Output Hi-Z/ Input enabled</td></tr>
<tr><td>P55 (HAK)</td><td>“L” output</td></tr>
<tr><td>P54 (HRQ)</td><td>“H” input</td></tr>
<tr><td>P53 (WRH)</td><td rowspan="2">“H” output *5</td><td rowspan="2">“H” output *5</td><td>Input disabled/ Output Hi-Z *4</td><td rowspan="2">“H” output</td><td rowspan="2">“H” output</td><td rowspan="2">“H” output</td></tr>
<tr><td>P52 (WRL)</td><td rowspan="2">Input disabled/ Output Hi-Z</td></tr>
<tr><td>P51 (RD)</td><td>“H” output</td><td>“H” output</td></tr>
<tr><td>P50 (ALE)</td><td>“L” output</td><td>“L” output</td><td>Output enabled *2</td><td>“L” output</td><td>“L” output</td></tr>
<tr><td>P67 to P60</td><td rowspan="5">Maintains the previous state *5</td><td rowspan="5">Input cutoff/ Maintains the previous state *5</td><td rowspan="6">Maintains the previous state *5</td><td rowspan="4">Input disabled/ Output Hi-Z</td><td rowspan="5">Output Hi-Z/ Input enabled</td><td rowspan="6">Output Hi-Z/ Input enabled</td></tr>
<tr><td>P76 to P70</td></tr>
<tr><td>P97 to P91</td></tr>
<tr><td>PA3 to PA0</td></tr>
<tr><td>P90 (CS0)</td><td>“H” output</td></tr>
<tr><td>When used as External Interrupt Pin (IRQX)</td><td>Input enabled *3</td><td>Input enabled *3</td><td>Input enabled *3</td><td>Input disabled</td></tr>
</table>

*1: “Output state” means that the pin-drive transistors are set in a drive-enabled state, but internal circuits are stopped so that a fixed value, “H” or “L”, is output. When internal peripheral circuits are in operation and output functions are used, output varies except when reset. Output does not vary during a reset.

*2: “Output enabled” means that the pin-drive transistors are set in a drive state and that internal circuit operations are enabled. Operational states are therefore conveyed on the pins.

*3: Same as in other ports when used in the output state. “Input enabled” means that input functions are ready and require pull up/pull down or external input.

*4: The previous values are retained if used as an output port.

*5: The state output immediately before this mode was set is output as it is. If it is input, input is disabled. “Input disabled” means that operations of input gates located very close to the pins are enabled, but pin states cannot be accepted internally because internal circuits are not operating.

*6: In the input cutoff state, the input is masked and the “L” level is passed to the internal. Output Hi-Z means to disable the transistors to drive the pin, and make the pin to be high-impedance.

Pin States in External Bus 8-Bit Data Bus Mode and Non-Multiplex 8-Bit External Bus Mode

Table 6-9 summarizes pin states in the external bus 8-bit data bus mode and non-multiplex 8-bit external bus mode.

Table 6-9. Pin States in External Bus 8-Bit Data Bus Mode and Non-Multiplex 8-Bit External Bus Mode

Pin name	In sleep state	In stop state		In hold state	When reset	Internal ROM access immediately after reset cancellation	Internal ROM access after external ROM access
		SPL = 0	SPL = 1				
P07 to P00 (AD07 to AD00)	Input disabled/ Output Hi-Z	Input cutoff/ Output Hi-Z	Input cutoff/ Output Hi-Z *6	Input disabled/ Output Hi-Z	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P37 to P30 (A07 to A00)	Output state *1	Output state *1			Output state *1	Output state *1	Maintains the previous address
P47 to P40 (A15 to A08)							
P27 to P20 (A23 to A16)	Output state *1, *4	Output state *1, *4		Input disabled/ Output Hi-Z *4			
P57 (CLK)	Input disabled/ Output enabled *2, *4	Input disabled/ Output state *1, *4		Input disabled/ Output enabled *2, *4	Input disabled/ Output enabled *2	CLK output	CLK output
P56 (RDY)	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5		Input disabled *4	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P55 ($\overline{\text{HAK}}$)				"L" output			
P54 (HRQ)				"H" input			
P52 ($\overline{\text{WRL}}$)	"H" output *4	"H" output *4		Input disabled/ Output Hi-Z *4	"H" output	"H" output	"H" output
P51 ($\overline{\text{RD}}$)	"H" output	"H" output		Input disabled/ Output Hi-Z			
P50 (ALE)	"L" output	"L" output			Output enabled *2	"L" output	"L" output
P17 to P10	Maintains the previous state *5	Input cutoff/ Maintains the previous state *5		Maintains the previous state *5	Input disabled/ Output Hi-Z	Output Hi-Z/ Input enabled	Output Hi-Z/ Input enabled
P67 to P60							
P76 to P70							
P87 to P80							
P97 to P91							
PA3 to PA0							
P90 (CS0)					"H" output		
When used as External Interrupt Pin (IRQX)	Input enabled *3	Input enabled *3	Input enabled *3		Input disabled		

*1: "Output state" means that the pin-drive transistors are set in a drive-enabled state, but internal circuits are stopped so that a fixed value, "H" or "L", is output. When internal peripheral circuits are in operation and output functions are used, output varies except when reset. Output does not vary during a reset.

*2: "Output enabled" means that the pin-drive transistors are set in a drive state and that internal circuit operations are enabled. Operational states are therefore conveyed on the pins.

*3: Same as in other ports when used in the output state. "Input enabled" means that input functions are ready and require pull up/pull down or external input.

*4: The previous values are retained if used as an output port.

*5: The state output immediately before this mode was set is output as it is. If it is input, input is disabled. "Input disabled" means that operations of input gates located very close to the pins are enabled, but pin states cannot be accepted internally because internal circuits are not operating.

*6: In the input cutoff state, the input is masked and the "L" level is passed to the internal. Output Hi-Z means to disable the transistors to drive the pin, and make the pin to be high-impedance.

6.8 Caution on Using Low-Power Consumption Mode

When operating in the low-power consumption mode, exercise reasonable care concerning the following:

- Change to the standby mode and interrupts
- Cancellation of standby mode by interrupt
- Cancellation of stop mode
- Oscillation stabilization wait time
- Notes on accessing the low-power consumption mode control register (LPMCR) to enter the standby mode

Change to Standby Mode and Interrupts

When a peripheral function issues an interrupt request to the CPU, the setting of “1” to the stop mode bit (STP) and sleep mode bit (SLP) and “0” to the watch/time-base timer mode bit (TMD) in the low-power consumption mode control register (LPMCR) is ignored. A change to the appropriate standby mode is not executed. (Neither is a change to the standby mode executed after interrupt processing.)

Even when the CPU is processing an interrupt, the interrupt request flag bit is cleared, and a change to the standby mode is possible unless there is another interrupt request.

Cancellation of Standby Mode by Interrupt

The standby mode is canceled if a peripheral function or other device issues an interrupt request whose interrupt level is higher than 7 in the sleep, time-base timer, or stop mode. (This is irrelevant to whether or not the CPU accepts an interrupt.)

After the standby mode is canceled by an interrupt, branching to an interrupt processing routine is performed as in a normal interrupt operation if the priority of interrupt level setting bit (bits IL2, IL1, and IL0 of the ICR register) corresponding to an interrupt request is higher than the interrupt level mask register (ILM) and if interrupts are enabled (I=1) by the I-flag of the condition code register (CCR). If an interrupt is not accepted, operation is resumed beginning from an instruction next to the instruction specifying the standby mode.

In the execution of interrupt processing, interrupt processing is normally started after execution of an instruction next to the instruction specifying the standby mode. However, depending on the conditions under which the mode is changed to the standby mode, interrupt processing may be started before the next instruction is executed.

Disabling of interrupts or other actions are required before setting the standby mode if branching to an interrupt processing routine is not performed immediately after returning to the standby mode.

Oscillation Stabilization Wait Time

Oscillation stabilization wait time of oscillation clock

The oscillator for source oscillation is stopped in the stop mode, and a oscillation stabilization wait time must be provided. Specify the oscillation stabilization wait time selected with the selection bits (WS1 and WS0) for the oscillation stabilization wait time of the clock selection register (CKSCR).

Set “00_B” in the selection bits (WS1 and WS0) for the oscillation stabilization wait time of the clock selection register (CKSCR) only in the main clock mode.

Oscillation stabilization wait time of PLL clock

In main clock mode, the PLL multiplication circuit stops. When changing to PLL clock mode, it is necessary to reserve the PLL clock oscillation stabilization wait time. The CPU runs in main clock mode till the PLL clock oscillation stabilization wait time has elapsed.

When the main clock mode is switched to PLL clock mode, the PLL clock oscillation stabilization wait time is fixed at 2^{14}/HCLK (HCLK: oscillation clock).

In sub clock mode, the main clock and PLL multiplication circuit stop. When changing to PLL clock mode, it is necessary to reserve the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time. The oscillation stabilization wait times for main clock and PLL clock are counted simultaneously according to the value specified in the oscillation stabilization wait time selection bits (CKSCR: WS1, WS0) in the clock selection register. The oscillation stabilization wait time selection bits (CKSCR: WS1, WS0) in the clock selection register must be selected accordingly to account for the longer of the main clock and PLL clock oscillation stabilization wait times. The PLL clock oscillation stabilization wait time, however, requires 2^{14}/HCLK or more. Set the oscillation stabilization wait time selection bits in the clock selection register (CKSCR: WS1, WS0) to “10_B” or “11_B”.

In PLL stop mode, the main clock and PLL multiplication circuit stop. During recovery from PLL stop mode, it is necessary to allot the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time. The oscillation stabilization wait times for the main clock and PLL clock are counted simultaneously according to the value specified in the oscillation stabilization wait time selection bits (CKSCR: WS1, WS0) in the clock selection register. The oscillation stabilization wait time selection bits (CKSCR: WS1, WS0) in the clock selection register must be selected accordingly to account for the longer of main clock and PLL clock oscillation stabilization wait time. The PLL clock oscillation stabilization wait time, however, requires 2^{14}/HCLK or more. Set the oscillation stabilization wait time selection bits in the clock selection register (CKSCR: WS1, WS0) to “10_B” or “11_B”.

Switching the Clock Mode

When the clock mode is switched, do not switch to low-power consumption mode and other clock mode before this switching is completed. Confirm the completion of clock mode switching by referring to the MCM and SCM bits of the clock selection register (CKSCR). If the mode is switched to other clock mode and low-power consumption mode before completion of switching, the mode may not be switched.

Notes on Accessing the Low-Power Consumption Mode Control Register (LPMCR) to Enter the Standby Mode

- To access the low-power consumption mode control register (LPMCR) with assemble language

  To enter the standby mode using the low-power consumption mode control register (LPMCR), use the instruction listed in Table 6-2.

  The low-power consumption mode transition instruction in Table 6-2 must always be followed by an array of instructions highlighted by a line below.

```
MOV     LPMCR,#H'xx     ; the low-power consumption mode transition instruction in Table 6-2.
NOP
NOP
JMP     $+3             ; jump to next instruction
MOV     A,#'10          ; any instruction
```

  The devices does not guarantee its operation after releasing from the standby mode if you place an array of instructions other than the one enclosed in the dotted line.

- To access the low-power consumption mode control register (LPMCR) with C language

  To enter the standby mode using the low-power consumption mode control register (LPMCR), use one of the following methods (1) to (3) to access the register:

  (1) Specify the standby mode transition instruction as a function and insert two _wait_nop() built-in functions after that instruction. If any interrupt other than the interrupt to return from the standby mode can occur within the function, optimize the function during compilation to suppress the LINK and UNLINK instructions from occurring.

  Example: Watch mode or time-base timer mode transition function

```
void enter_watch(){
    IO_LPMCR.byte = 0x10;          /* Set TMD bit of LPMCR to “0” */
    _wait_nop();
    _wait_nop();
    }
```

  (2) Define the standby mode transition instruction using _asm statements and insert two NOP and JMP instructions after that instruction.

  Example: Transition to sleep mode

```
_asm(“  MOVI: _IO_LPMCR,#H'58);   /* Set SLP bit of LPMCR to “1” */
_asm(“  NOP”);
_asm(“  NOP”);
_asm(“  JMP   $+3”);              /* Jump to next instruction */
```

(3) Define the standby mode transition instruction between #pragma asm and #pragma endasm and insert two NOP and JMP instructions after that instruction.

Example: Transition to stop mode

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

7. Mode Setting

This chapter explains mode setting, mode pins, mode data, external memory access and its operation.

7.1 Mode Setting

7.2 Mode Pins (MD2 to MD0)

7.3 Mode Data

7.4 External Memory Access

7.5 Operation of Each Mode for Mode Setting

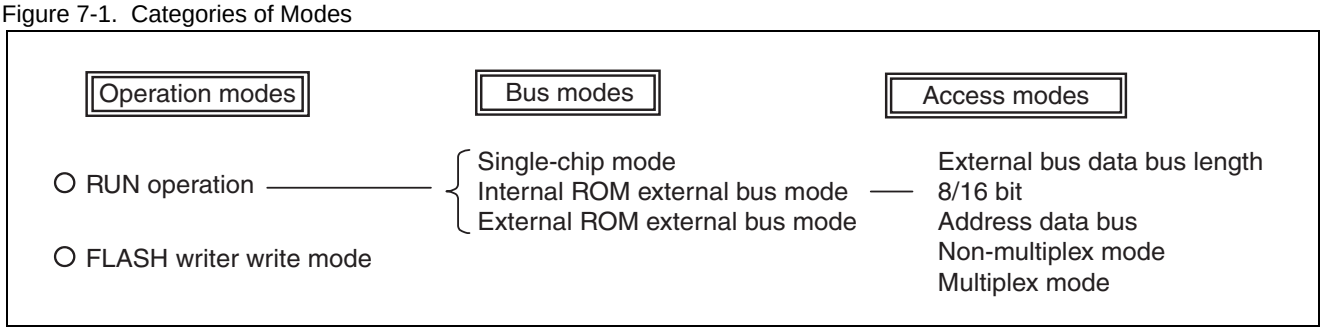
7.1 Mode Setting

The F^{2}MC-16LX has different modes in each access system and access area. Each mode is set according to a mode pin at the reset state and according to mode data obtained by mode-fetch.

Categories of Modes

The F^{2}MC-16LX has different modes in each access system and access area. In this module, they are categorized as shown in Figure 7-1.

Figure 7-1. Categories of Modes



Operation Modes

Operation modes are used to control the operating conditions of devices, and they are set by mode setting pins (MDx) and with the contents of the Mx bits in mode data. By selecting an operation mode, normal operation activation or flash memory programming can be performed.

Bus Modes

Bus modes are used to control the operation of internal ROMs and of external access functions, and they are specified by mode setting pins (MDx) and with the contents of the Mx bits in mode data. Mode setting pins (MDx) specify the reset vector as well as set the bus mode for reading mode data. The Mx bits in mode data specify the bus mode during normal operation.

Access Modes

Access modes are used to control the external data bus width, and they are set by mode setting pins (MDx) and with the contents of the Sx bits in mode data. Selection of an access mode specifies either an 8-bit length or 16-bit length for the external data bus. It also specifies either the non-multiplex mode or multiplex mode for the address data bus.

7.2 Mode Pins (MD2 to MD0)

Mode pins are three external pins MD2 to MD0 that specify the reset vector and mode data fetching method.

Settings of Mode Pins (MD2 to MD0)

Mode pins (MD2 to MD0) are used to select the source, either the external or internal data bus when reset vectors are read, and to select the bus width when the external data bus is used. For a device with built-in FLASH ROM, the mode pins are also used to set the FLASH ROM write mode for writing program to built-in ROM.

Table 7-1 lists the contents of mode pin settings.

Table 7-1. Contents of Mode Pin Settings

P01	P00	MD2	MD1	MD0	Mode name	Reset vector access area	External data bus width	Remarks
-	-	0	0	0	External vector mode 0	External	Multiplex mode	Reset vector 8-bit bus width access
-	-	0	0	1	External vector mode 1	External	Multiplex mode	Reset vector 16-bit bus width access
-	-	0	1	0	External vector mode 2	External	Non-multiplex mode	Reset vector 8-bit bus width access
-	-	0	1	1	Internal vector mode	Internal	Mode data	Operation after reset sequence is controlled with mode data
-	-	1	0	0	Setting is prohibited			
-	-	1	0	1				
1	0	1	1	0	FLASH serial writing			
-	-	1	1	1	FLASH writer write mode	-	-	-

Note: MD2 to MD0: Specify 0=V_{SS} or 1=V_{CC}. For external vector mode 2, the data bus width also has the initial value of 8 bits. To specify 16 bits as the data bus width, specify mode data for the non-multiplex external data bus 16-bit mode, and then the IOBS and LMBS areas are set up for 16-bit size access. To set up the HMBS area for 16-bit size access, change the HMBS setting.

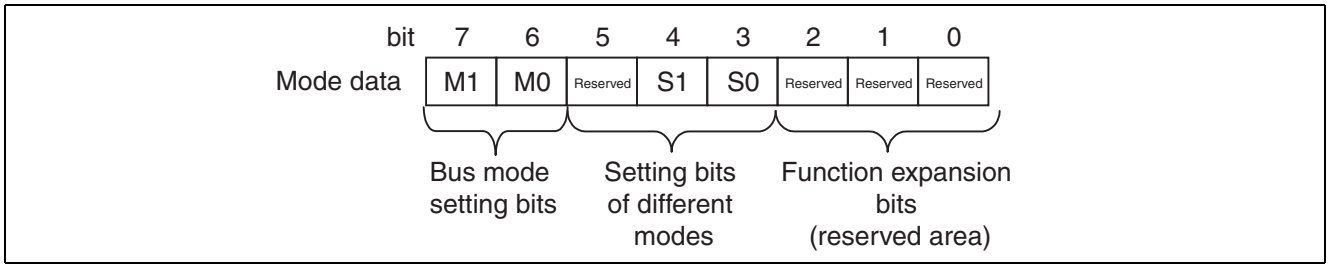
7.3 Mode Data

Mode data stored at address $FFFFDF_H$ in memory specifies the operation immediately after the reset sequence. Mode data is fetched in the CPU automatically by mode fetching.

Mode Data

During the reset sequence, mode data at address "$FFFFDF_H$" is sent to the mode register in the CPU core. The CPU uses this mode data to set the memory access mode. The contents of the mode register can be changed only by the reset sequence. Furthermore, mode data settings become valid only after the reset sequence.

The configuration of mode data is shown in the figure below.



*: Set reserved bit to "0".

Setting Bits of Different Modes (S1, S0)

S1 and S0 bits specify the bus mode and access mode that is set after completion of the reset sequence.

Table 7-2 lists the contents of the settings for S1 and S0 bits.

Table 7-2. Contents of S1 and S0 Bit Settings

S1	S0	Functions	
0	0	External data bus 8-bit mode	Address data bus multiplex
0	1	External data bus 16-bit mode	
1	0	External data bus 8-bit mode	Address data bus non-multiplex
1	1	External data bus 16-bit mode	

Bus Mode Setting Bits (M1, M0)

M1 and M0 bits specify the operation mode that is set after completion of the reset sequence.

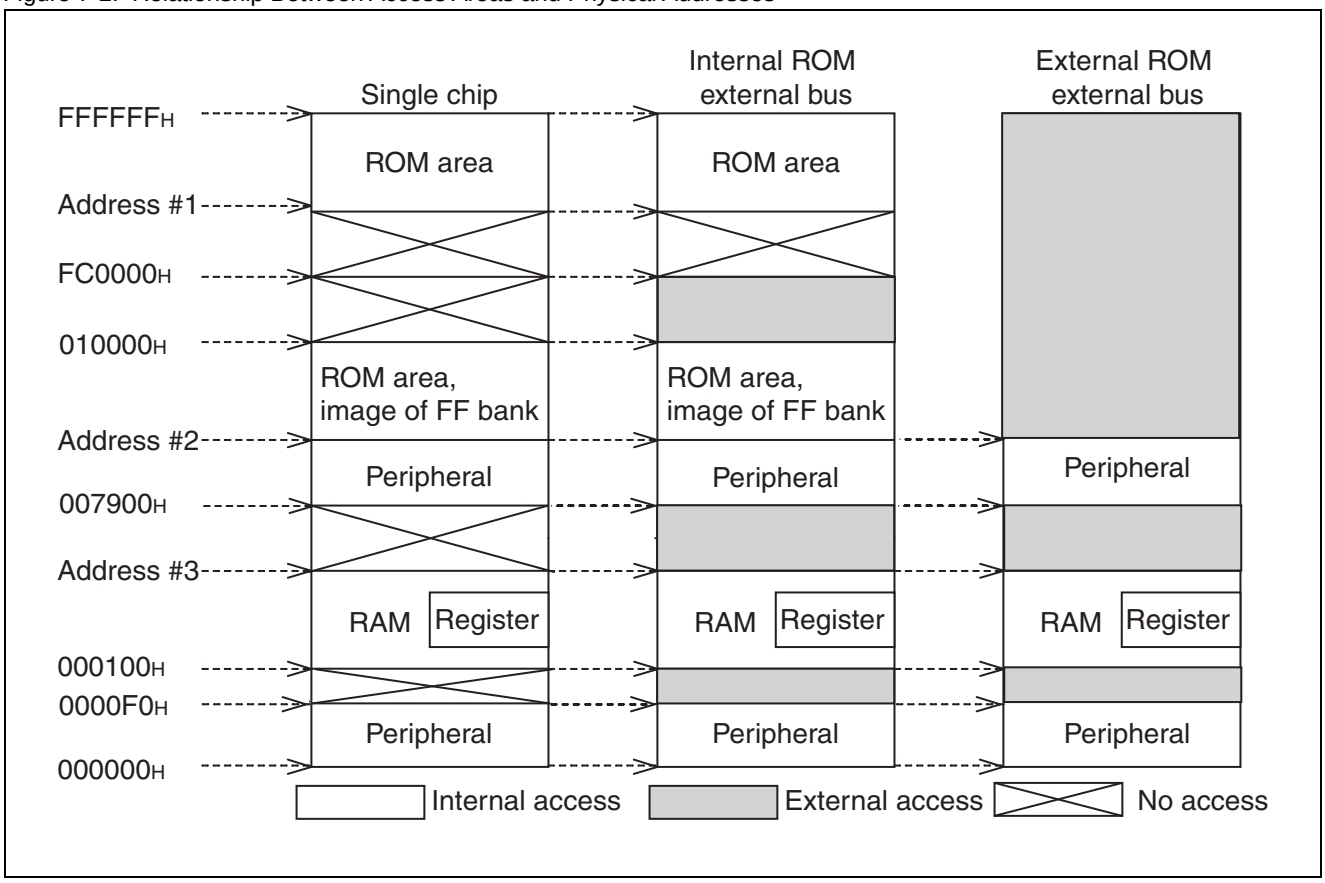
Table 7-3 lists the contents of the settings for M1 and M0 bits.

Table 7-3. Contents of M1 and M0 Bit Settings

M1	M0	Functions
0	0	Single-chip mode
0	1	Internal ROM external bus mode
1	0	External ROM external bus mode
1	1	(Setting is prohibited)

Figure 7-2 shows the correspondence between access areas and physical addresses.

Figure 7-2. Relationship Between Access Areas and Physical Addresses



Note:

“Address #1”, “Address #2” and “Address #3” are determined based on individual models. See “A.1 Memory Map”, for details.

Relationship Between Mode Pins and Mode Data (Recommended Example)

Table 7-4 shows the relationship between mode pins and mode data.

Table 7-4. Relationship Between Mode Pins and Mode Data

Mode	MD2	MD1	MD0	M1	M0	S1	S0
Single chip	0	1	1	0	0	X	X
Internal ROM external bus mode, 8-bit (address data multiplex)	0	1	1	0	1	0	0
Internal ROM external bus mode, 16-bit (address data multiplex)	0	1	1	0	1	0	1
Internal ROM external bus mode, 8-bit (address data non-multiplex)	0	1	1	0	1	1	0
Internal ROM external bus mode, 16-bit (address data non-multiplex)	0	1	1	0	1	1	1
External ROM external bus mode, 16-bit, bus vector with 16-bit width (address data multiplex)	0	0	1	1	0	0	1
External ROM external bus mode, 8-bit (address data multiplex)	0	0	0	1	0	0	0
External ROM external bus mode, 8-bit (address data non-multiplex)	0	1	0	1	0	1	0

Note: If the output for high-order addresses of A23 to A16 is suppressed, the maximum capacity of accessible memory is 64 Kbytes.

Operation of External Pins in Each Mode

Table 7-5 shows the operation of each external pins in the non-multiplex mode and multiplex mode.

Table 7-5. Operation of External Pins in Each Mode

	Functions							
	Non-multiplex mode				Multiplex mode			
	External address control				External address control			
	Permitted (address)		Prohibited (port)		Permitted (address)		Prohibited (port)	
	External bus extension		External bus extension		External bus extension		External bus extension	
	8-bit	16-bit	8-bit	16-bit	8-bit	16-bit	8-bit	16-bit
P07 to P00/ D07 to D00/ AD07 to AD00	D07 to D00				AD07 to AD00			
P17 to P10/ D15 to D08/ AD15 to AD08	Port	D15 to D08	Port	D15 to D08	A15 to A08	AD15 to AD08	A15 to A08	AD15 to AD08
P27 to P20	A23 to A16		Port		A23 to A16		Port	
P37 to P30	A07 to A00		A07 to A00		Port			
P47 to P40	A15 to A08		A15 to A08					
ALE	ALE				ALE			
RD	RD				RD			
P52/$\overline{\text{WRL}}$	WRL				WRL			
P53/$\overline{\text{WRH}}$	Port	WRH	Port	WRH	Port	WRH	Port	$\overline{\text{WRH}}$
P54/HRQ	HRQ				HRQ			
P55/$\overline{\text{HAK}}$	HAK				HAK			
P56/RDY	RDY				RDY			
P57/CLK	CLK				CLK			

- In the single-chip mode, all addresses can be used as ports.
- High-order addresses, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$, $\overline{\text{HAK}}$, HRQ, RDY, and CLK can be used as ports depending on the selected function.
- In the non-multiplex mode, the up/down-counter cannot be used. They function as addresses.
- For dual clock products, A09 and A08 in non-multiplex mode cannot be used as they are the connection pins for the sub clock oscillator.

7.4 External Memory Access

This section contains block diagrams about external memory access, the configuration and functions of registers, and operation of external memory access.

I/O Signal Pins for External Memory Access

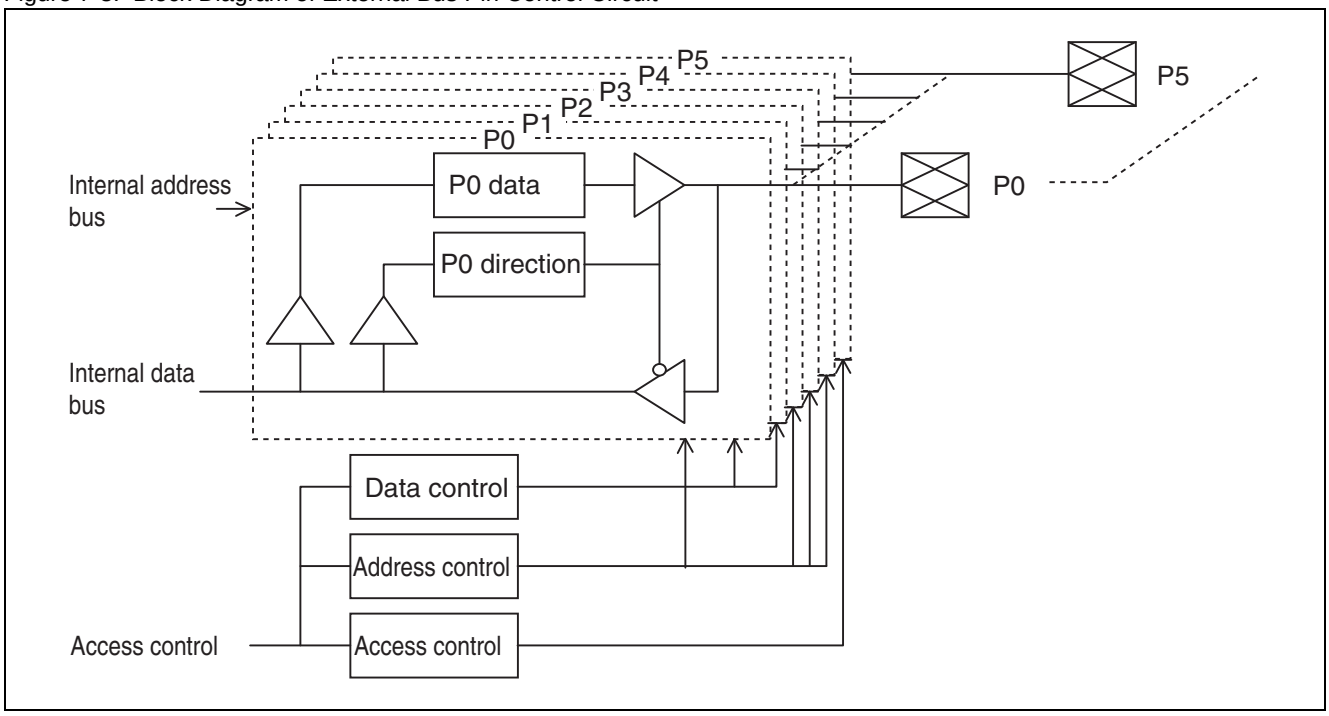
For accessing external memory and peripheral devices, the F^{2}MC-16LX supplies the following address, data, and control signals:

- CLK (P57): Outputs the machine cycle clock
- RDY (P56): External ready input pin
- $\overline{\text{HAK}}$ (P55): Hold acknowledge output pin
- HRQ (P54): Hold request input pin
- $\overline{\text{WRH}}$ (P53): Write signal for the high-order 8 bits on the data bus
- $\overline{\text{WRL}}$ (P52): Write signal for the low-order 8 bits on the data bus
- $\overline{\text{RD}}$ (P51): Read signal
- ALE (P50): Address latch enable signal (effective in the multiplex mode)

Block Diagram

Figure 7-3 is a block diagram of the external bus pin control circuit.

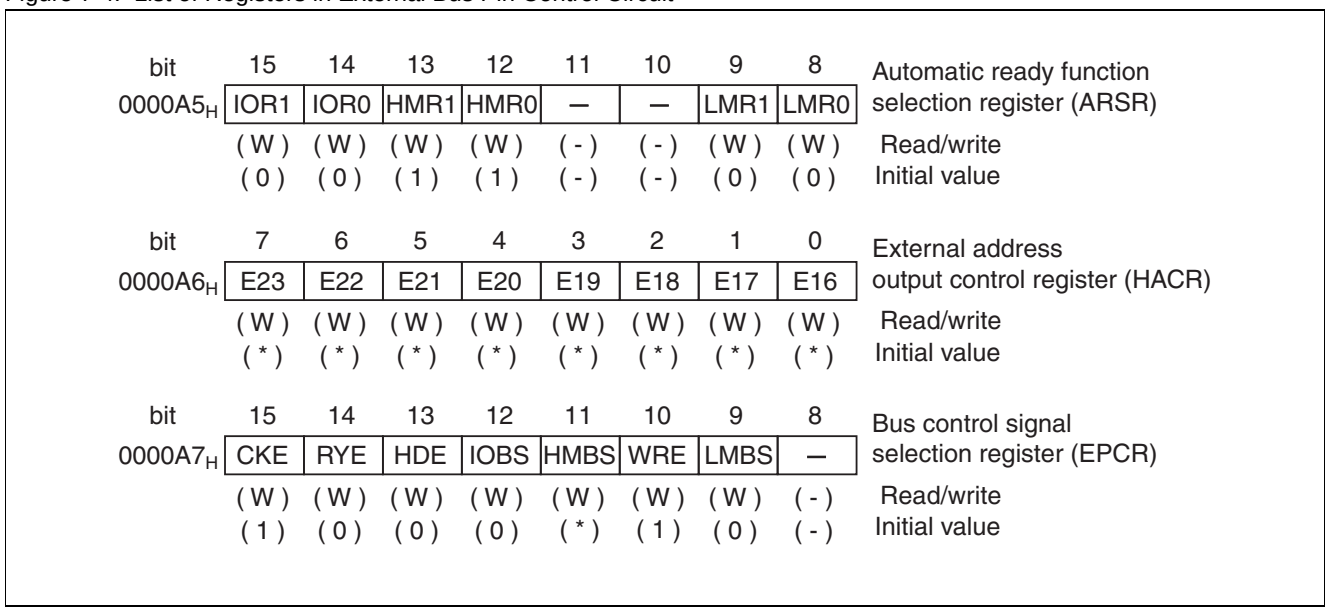
Figure 7-3. Block Diagram of External Bus Pin Control Circuit



List of Registers

Figure 7-4 shows a list of registers in the external bus pin control circuit.

Figure 7-4. List of Registers in External Bus Pin Control Circuit



7.4.1 Automatic Ready Function Selection Register (ARSR)

This section explains the configuration and function of the automatic ready function selection register (ARSR)

Automatic Ready Function Selection Register (ARSR)

The bit configuration of the automatic ready function selection register (ARSR) is shown in the figure below.

bit	15	14	13	12	11	10	9	8	
$0000A5_H$	IOR1	IOR0	HMR1	HMR0	—	—	LMR1	LMR0	Automatic ready function selection register
	(W)	(W)	(W)	(W)	(-)	(-)	(W)	(W)	Read/write
	(0)	(0)	(1)	(1)	(-)	(-)	(0)	(0)	Initial value

Functions of each bit in the automatic ready function selection register (ARSR) are described below.

[bit15, bit14] IOR1, IOR0

These bits are used to select the automatic wait function for external access to areas in a range of $0000F0_H$ to $0000FF_H$. Contents of settings are listed below.

IOR1	IOR0	Setting
0	0	Automatic wait prohibited [Initial value]
0	1	Automatic wait in 1 machine cycle during external access
1	0	Automatic wait in 2 machine cycles during external access
1	1	Automatic wait in 3 machine cycles during external access

[bit13, bit12] HMR1, HMR0

These bits are used to select the automatic wait function for external access to areas in a range of 800000_H to $FFFFFF_H$. Contents of settings are listed below.

HMR1	HMR0	Setting
0	0	Automatic wait prohibited
0	1	Automatic wait in 1 machine cycle during external access
1	0	Automatic wait in 2 machine cycles during external access
1	1	Automatic wait in 3 machine cycles during external access [Initial value]

[bit9, bit8] LMR1, LMR0

These bits are used to select the automatic wait function for external access to areas in a range of 002000_H to $7FFFFF_H$. Contents of settings are listed below.

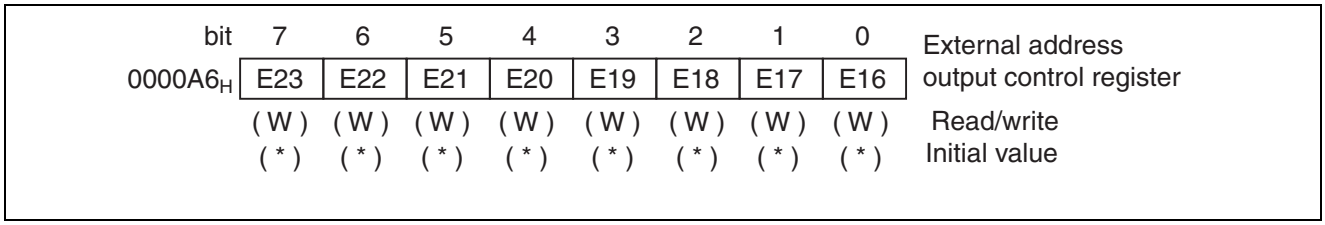
LMR1	LMR0	Setting
0	0	Automatic wait prohibited [Initial value]
0	1	Automatic wait in 1 machine cycle during external access
1	0	Automatic wait in 2 machine cycles during external access
1	1	Automatic wait in 3 machine cycles during external access

7.4.2 External Address Output Control Register (HACR)

This section explains the configuration and function of the external address output control register.

External Address Output Control Register (HACR)

The bit configuration of the external address output control register is shown in the figure below.



The external address output control register controls external output of addresses (A23 to A16). One bit corresponds to each of addresses A23 to A16 and controls each address output pin as follows.

0	The corresponding pin is for address output (AXX).
1	The corresponding pin is as an I/O port (PXX). Please set “1” when using it as peripheral resource.

This register cannot be accessed while the device is set to the single-chip mode. In this event, all ports function as I/O ports regardless of the values in this register. All bits of this register are dedicated for writing, and the readout value is “1”. Furthermore, if addresses are expected to be output with address output selected, specify the value of DDR to “0”.

The initial value is “1” if the device is activated in internal vector mode. Otherwise, the initial value is “0”.

Note:

When using PPG and base timer, set them to “1” (setting for an I/O port).

7.4.3 Bus Control Signal Selection Register (EPCR)

This section explains the configuration and function of the bus control signal selection register.

Bus Control Signal Selection Register (EPCR)

The bus control signal selection register (EPCR) sets the bus operation control function in external bus mode.

The bit configuration of the bus control signal selection register is shown in the figure below.

bit	15	14	13	12	11	10	9	8	
$0000A7_H$	CKE	RYE	HDE	IOBS	HMBS	WRE	LMBS	—	Bus control signal selection register
	(W)	(W)	(W)	(W)	(W)	(W)	(W)	(-)	Read/write
	(1)	(0)	(0)	(0)	(*)	(1)	(0)	(-)	Initial value

This register cannot be accessed while the device is set to the single-chip mode. In the single-chip mode, all pins function as I/O ports regardless of the values in this register. All bits of this register are dedicated for writing, and the readout value is "1".

Functions of each bit in the bus control signal selection register are described below.

[bit15] CKE

This bit controls the external clock (CLK) output.

0	I/O port (P57) operation (clock prohibited)
1	Clock signal (CLK) output permitted [Initial value]

[bit14] RYE

This bit controls the external ready (RDY) input.

0	I/O port (P56) operation (external RDY input prohibited) [Initial value]
1	External ready (RDY) input permitted

[bit13] HDE

This bit specifies I/O enable for hold-related pins. Hold request input (HRQ) and hold acknowledge output ($\overline{HAK}$) are controlled with this bit setting.

0	I/O port (P55, 54) operation (hold function I/O prohibited) [Initial value]
1	Hold request (HRQ) input/hold acknowledge ($\overline{HAK}$) output permitted

[bit12] IOBS

This bit specifies the bus width for accessing external buses corresponding to areas in a range of $0000D0_H$ to $0000FF_H$ in the external data bus 16-bit mode.

0	16-bit bus width access [Initial value]
1	8-bit bus width access

[bit11] HMBS

This bit specifies the bus width for accessing external buses corresponding to areas in a range of 800000_H to $FFFFFF_H$ in the external data bus 16-bit mode.

0	16-bit bus width access [Initial value in external vector mode 1]
1	8-bit bus width access [Initial value in external vector modes 0 and 2]

[bit10] WRE

This bit controls the output of external write signals (both the $\overline{WRH}$ and $\overline{WRL}$ pins in the external data bus 16-bit mode or the $\overline{WRL}$ pin in the external data bus 8-bit mode).

0	I/O port (P53, P52) operation (write signal output prohibited)
1	Write strobe signal ($\overline{WRH}$ and $\overline{WRL}$, or only $\overline{WRL}$) output permitted [Initial value]

[bit9] LMBS

This bit specifies the bus width for accessing external buses corresponding to areas in a range of 002000_H to $7FFFFF_H$ in the external data bus 16-bit mode.

0	16-bit bus width access [Initial value]
1	8-bit bus width access

Note:

Even when RDY and HRQ input is permitted by RYE and HDE bits, the I/O port function of a port is enabled. Therefore, be sure to set “0” (input mode) to the bit in DDR5 corresponding to that port.

7.5 Operation of Each Mode for Mode Setting

This section has a timing chart showing the operation of each mode for mode setting.

Types of Mode

Operation with the following items is categorized by function as follows:

- External memory access control signals
  - External data bus 8-bit mode (non-multiplex mode)
  - External data bus 8-bit mode (multiplex mode)
  - External data bus 16-bit mode (non-multiplex mode)
  - External data bus 16-bit mode (multiplex mode)
- Ready function
  - Non-multiplex mode
  - Multiplex mode
- Hold function
  - Non-multiplex mode
  - Multiplex mode

7.5.1 External Memory Access Control Signals

Access to external memory is performed in 3 cycles when the ready function is not used.

External Memory Access Control Signal

Timing charts for external access in each mode are shown in Figure 7-5 to Figure 7-8. Access with an 8-bit bus width in the external data bus 16-bit mode is a function for reading from and writing to peripheral chips of an 8-bit width when a mixture of peripheral chips of an 8-bit width and 16-bit width are connected to the external bus. Because access with an 8-bit bus width is performed using the low-order 8 bits of the data bus, connect the peripheral chips of an 8-bit bus width to the low-order 8 bits of data. Access with either a 16-bit bus width or an 8-bit bus width in the external data bus 16-bit mode is determined by the specification of the HMBS, LMBS, and/or IOBS bit of EPCR.

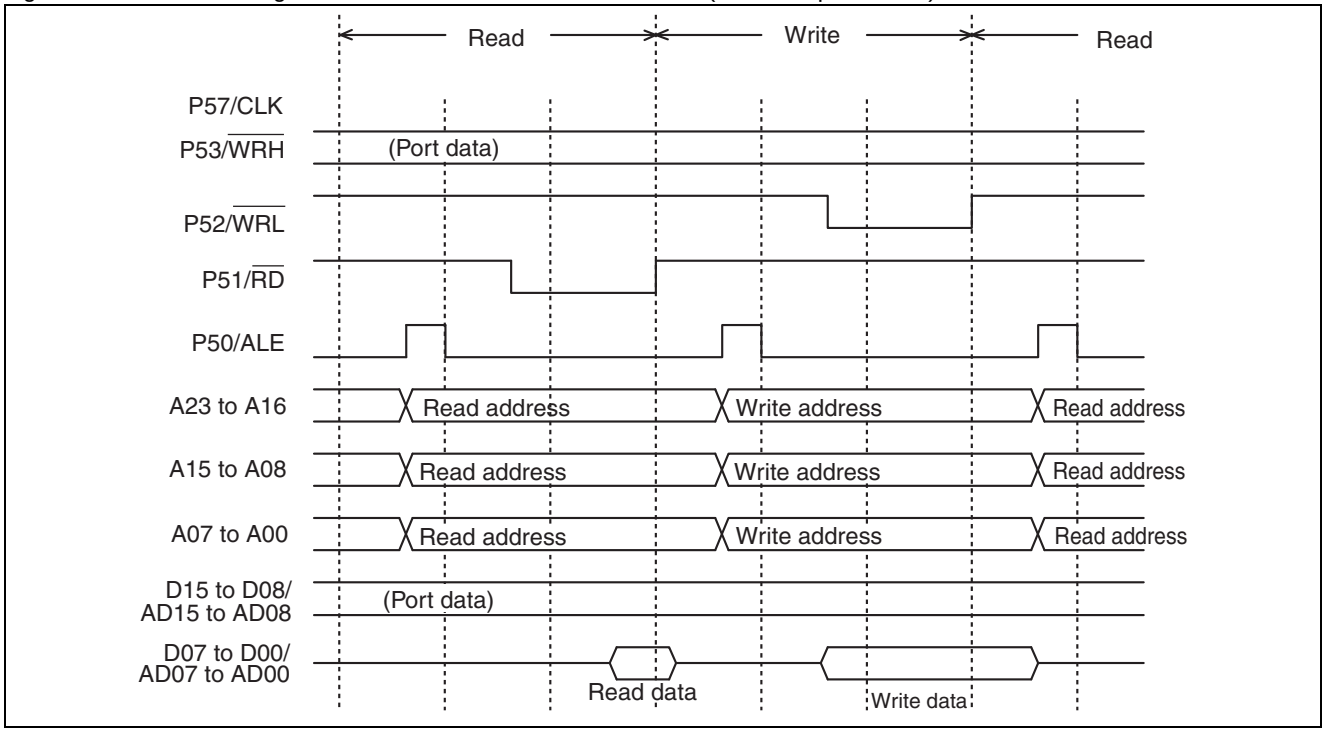
Incidentally, there is a case where bus operation is not actually done by only outputting addresses and ALE assert results in the multiplex mode without assert for $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$.

Note:

Be sure not to perform access to peripheral chips with only ALE signals.

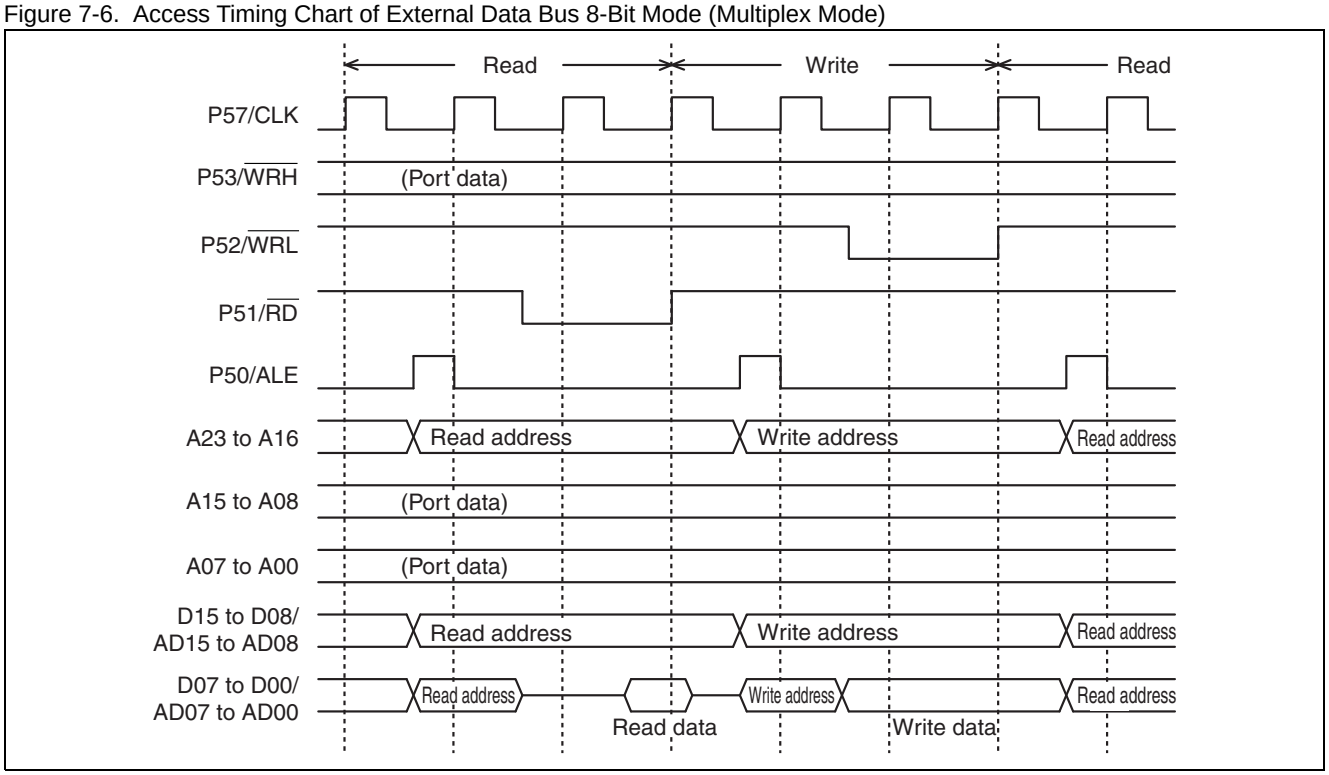
External data bus 8-bit mode (non-multiplex mode)

Figure 7-5. Access Timing Chart of External Data Bus 8-Bit Mode (Non-multiplex Mode)



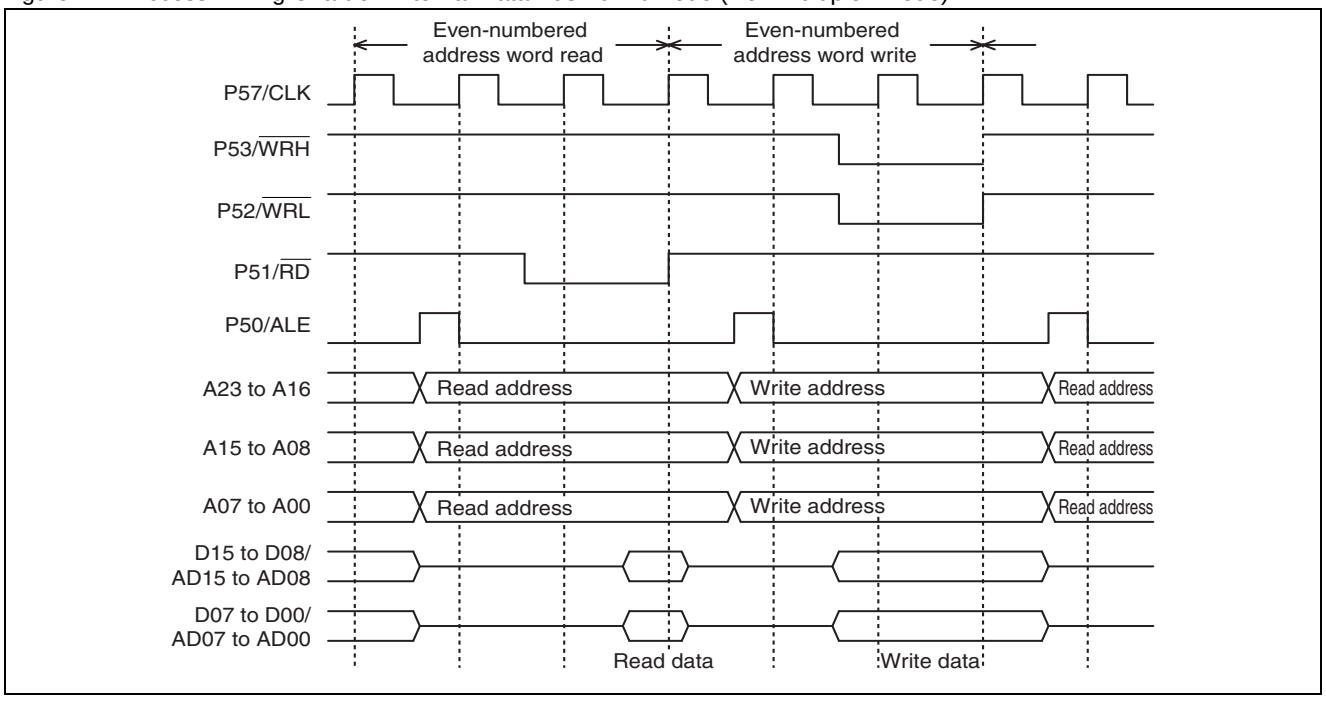
External data bus 8-bit mode (multiplex mode)

Figure 7-6. Access Timing Chart of External Data Bus 8-Bit Mode (Multiplex Mode)



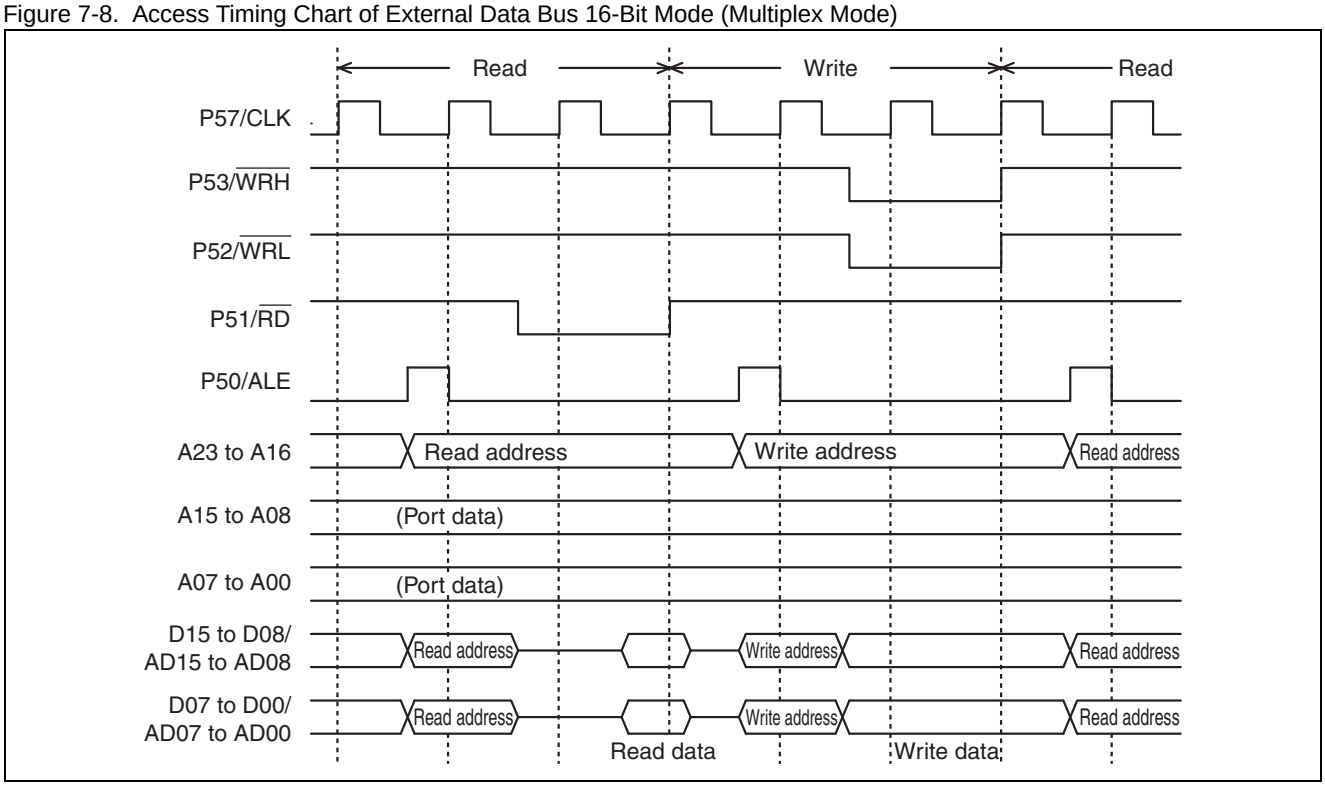
External data bus 16-bit mode (non-multiplex mode)

Figure 7-7. Access Timing Chart of External Data Bus 16-Bit Mode (Non-multiplex Mode)



External data bus 16-bit mode (multiplex mode)

Figure 7-8. Access Timing Chart of External Data Bus 16-Bit Mode (Multiplex Mode)



7.5.2 Ready Function

By setting the P56/RDY pin or defining the automatic ready function selection register (ARSR), access to low-speed memory and peripheral circuits is enabled. If the RYE bit in the bus control signal selection register (EPCR) is set to “1”, wait cycles are generated during the period where the “L” level is input to the P56/RDY pin while access to the external area is in progress. Thus, the access cycle can be extended.

Ready Function

The F^{2}MC-16LX has two types of built-in auto-ready functions for external memory. The auto-ready functions enable the access cycle to be extended by inserting 1 to 3 wait cycles automatically without an external circuit when access occurs to the external area within the following address ranges: a low-order address allocated between 002000_H and $7FFFFF_H$, and a high-order address allocated between 800000_H and $FFFFFF_H$. These functions are evoked by setting the LMR1 and LMR0 bits of ARSR (external area of a low-order address) and the HMR1 and HMR0 bits of ARSR (external area of a high-order address).

Furthermore, the F^{2}MC-16LX has built-in auto-ready function for external I/O that is independent of those for external memory. This function enables the access cycle to be extended by inserting 1 to 3 wait cycles automatically without an external circuit when access occurs to the external area between addresses $0000F0_H$ and $0000FF_H$. This function is evoked by setting the IOR1 and IOR0 bits in ARSR.

With the auto-ready functions for either external memory or external I/O, if the RYE bit of EPCR is set to “1” when the “L” level is input to the P56/RDY pin upon completion of the wait cycle generated by auto-ready, the wait cycle continues as it is.

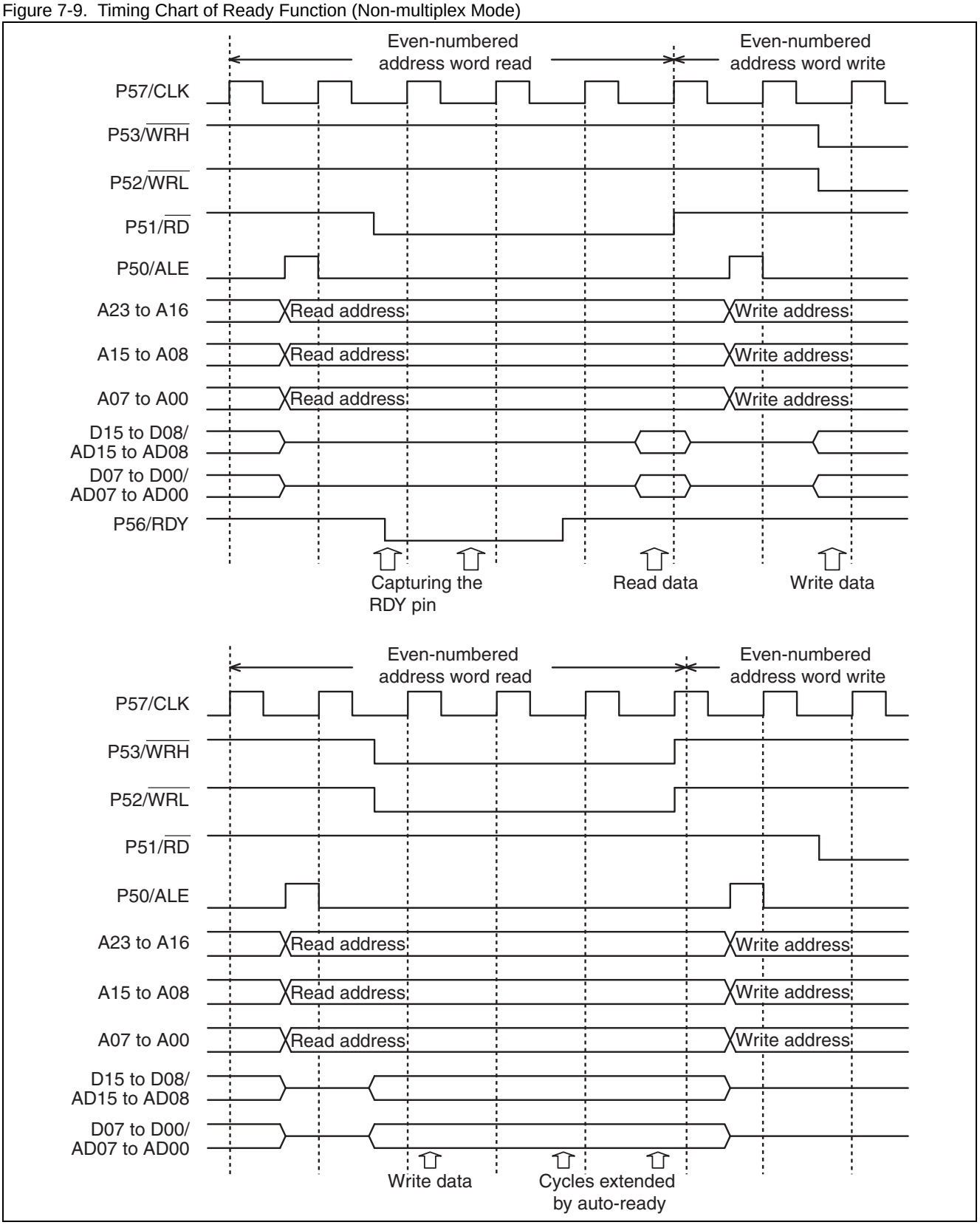
The timing charts of the ready function in the non-multiplex mode and multiplex mode are shown below. In both modes, the top figure shows the case where the ready function is not set and the bottom figure shows the case where the ready function is set.

Note:

If the AC rating is not satisfied for input from the RDY pin, be careful because this device may enter the runaway state.

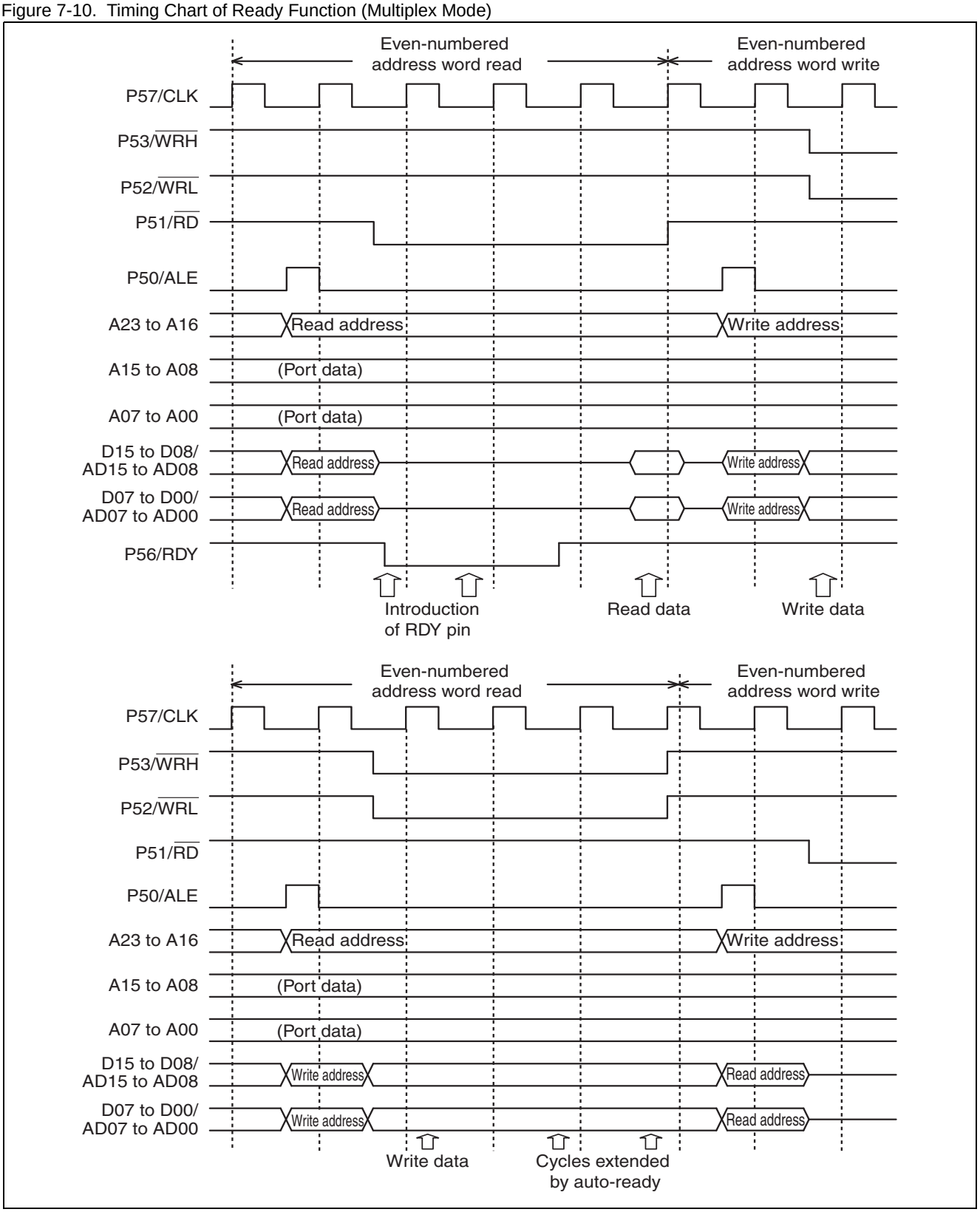
Non-multiplex mode

Figure 7-9. Timing Chart of Ready Function (Non-multiplex Mode)



Multiplex mode

Figure 7-10. Timing Chart of Ready Function (Multiplex Mode)



7.5.3 Hold Function

This section uses timing charts to describe the operation of the hold function.

Operation of Hold Function

When the HDE bit of EPCR is set to “1”, the external bus hold function specified by both the P54/HRQ and P55/$\overline{\text{HAK}}$ pins becomes effective. When the “H” level is input to the P54/HRQ pin, the hold state is set upon completion of an instruction by the CPU (after data of 1 element is processed in the case of the string instruction), and the "L" level is output from P55/$\overline{\text{HAK}}$ to set the following pins to a high-impedance state:

Non-multiplex mode

- Address output: A23 to A00
- Data input/output: D15/AD15 to D00/AD00
- Bus control signal: P51/$\overline{\text{RD}}$, P52/$\overline{\text{WRL}}$, P53/$\overline{\text{WRH}}$

Multiplex mode

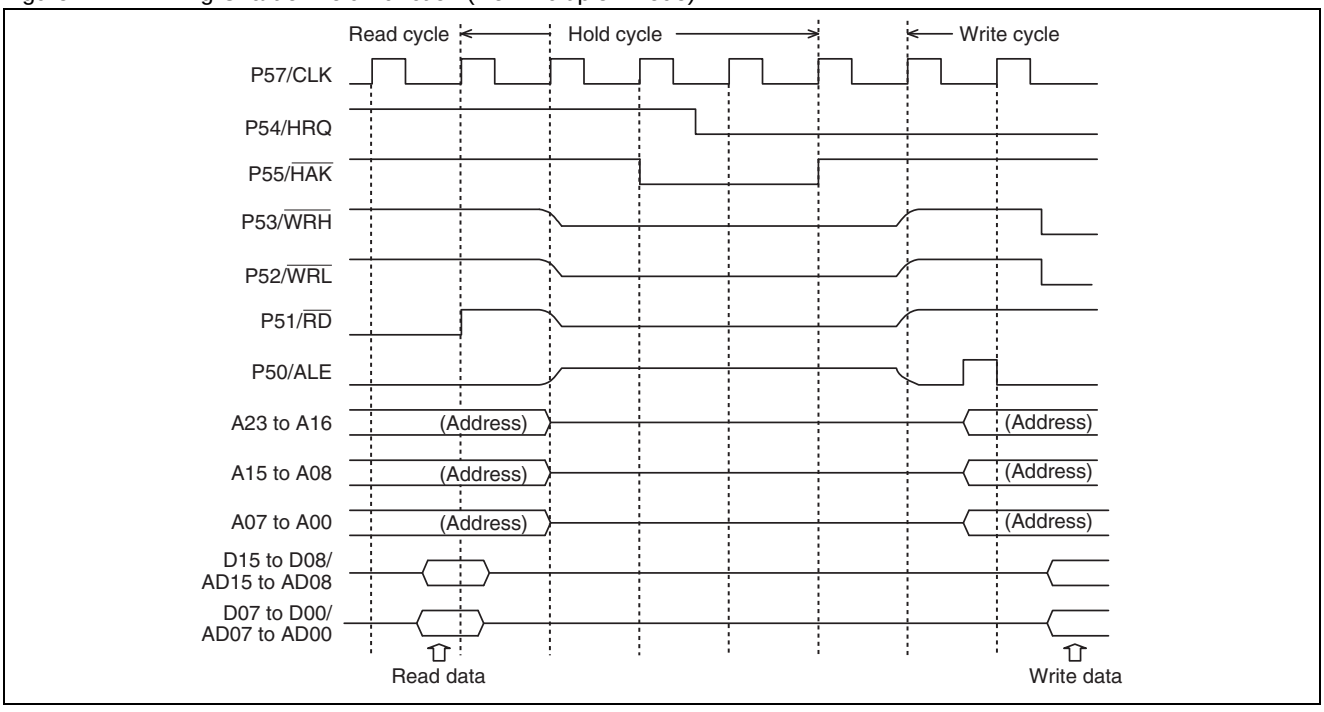
- Address output: A23 to A16
- Address output, Data input/output: D15/AD15 to D00/AD00
- Bus control signal: P51/$\overline{\text{RD}}$, P52/$\overline{\text{WRL}}$, P53/$\overline{\text{WRH}}$

This operation enables use of the external bus via the device external circuit. When the “L” level is input to the P54/HRQ pin, the P55/$\overline{\text{HAK}}$ pin outputs the “H” level to restore the external pin state, and the CPU restarts operation. In the STOP state, requests for hold are rejected.

Non-Multiplex Mode

Figure 7-11 shows a timing chart of the non-multiplex-mode hold function in the external data bus 16-bit mode.

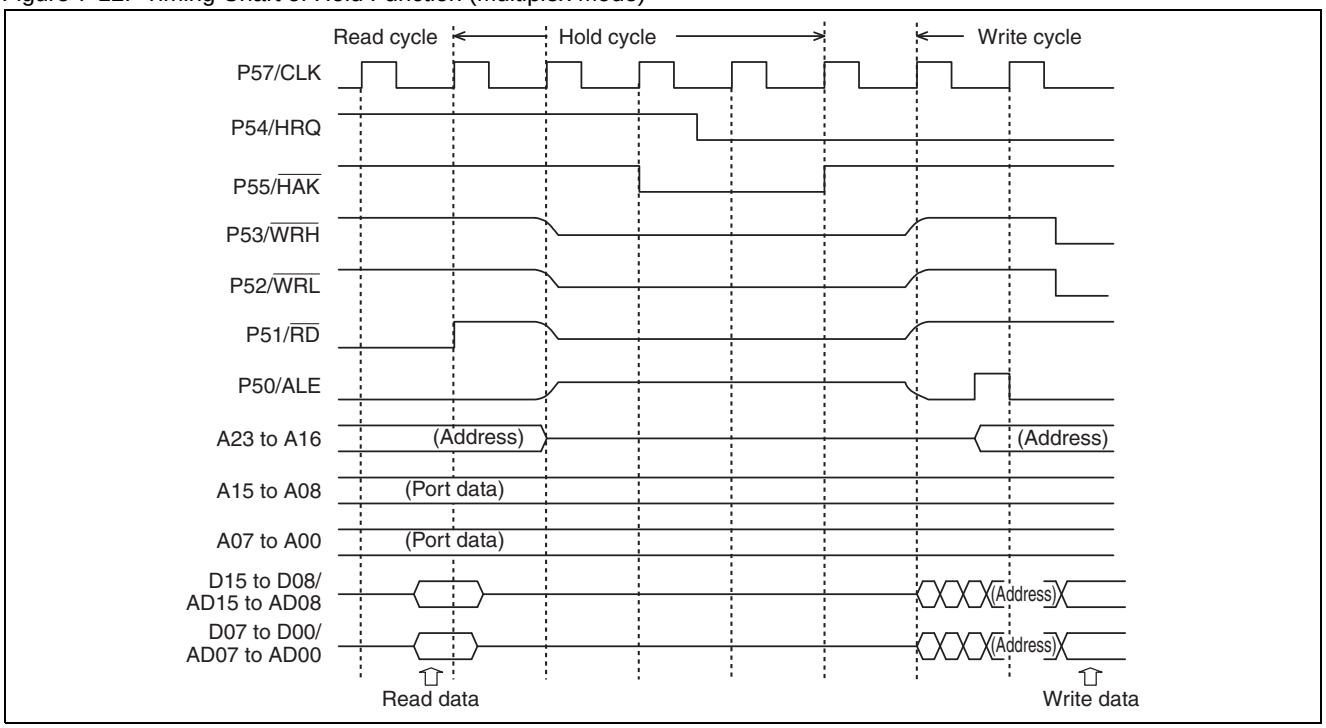
Figure 7-11. Timing Chart of Hold Function (Non-multiplex Mode)



Multiplex Mode

Figure 7-12 shows a timing chart of the multiplex-mode hold function in the external data bus 16-bit mode.

Figure 7-12. Timing Chart of Hold Function (Multiplex Mode)



Notes:

- After inputting the “H” level to the P54/HRQ pin, retain it to the “H” level until the P55/$\overline{\text{HAK}}$ pin is set to the “L” level.
- The watchdog timer continues to operate without getting the counter cleared even when the P55/$\overline{\text{HAK}}$ pin is at the “L” level. A watchdog reset occurs, if the timer is held for longer than the watchdog timer interval time set by the WT1/WT0 bits in the watchdog timer control register (WDTC).

8. I/O Port

This chapter explains the functions and registers of I/O port.

8.1 Functions of I/O Port

8.2 Registers for I/O Port

8.1 Functions of I/O Port

This section outlines the functions of the I/O port.

Functions of I/O Port

The I/O port has functions to output data from the CPU to I/O pins and fetched the signals input to I/O pins to the CPU by using the port data register (PDR). Furthermore, the I/O port enables input to and output from I/O pins to be set in any direction in units of bits by using the port direction register (DDR).

The MB90880 series has 83 input/output pins.

8.2 Registers for I/O Port

This section explains the configuration and functions of the registers used for the I/O port.

Registers for I/O Port

The registers for the I/O port are listed below:

- Port data registers (PDR0 to PDRA)
- Port direction registers (DDR0 to DDRA)
- Port input resistor registers (RDR0, RDR1)
- Output level set registers (OLSR0, OLSR1)
- Serial input level select registers (ILSR0, ILSR1, ILSR2)
- Analog input enable registers (ADER0, ADER1, ADER2)
- Up/down timer input enable register (UDER)
- PPG terminal assignment select register (PAFSR)
- Serial terminal assignment select register 0 (P7FSR)
- Serial terminal assignment select register 1 (P9FSR)

8.2.1 Port Data Registers (PDR0 to PDRA)

This section explains the configuration and functions of port data registers (PDR0 to PDRA)

Port Data Registers (PDR0 to PDRA)

Figure 8-1 shows a list of port data registers (PDR0 to PDRA).

Figure 8-1. List of Port Data Registers (PDR0 to PDRA)

PDR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:000000$_H$		P07	P06	P05	P04	P03	P02	P01	P00	Undefined	R/W *
PDR1	bit	7	6	5	4	3	2	1	0		
Address:000001$_H$		P17	P16	P15	P14	P13	P12	P11	P10	Undefined	R/W *
PDR2	bit	7	6	5	4	3	2	1	0		
Address:000002$_H$		P27	P26	P25	P24	P23	P22	P21	P20	Undefined	R/W *
PDR3	bit	7	6	5	4	3	2	1	0		
Address:000003$_H$		P37	P36	P35	P34	P33	P32	P31	P30	Undefined	R/W *
PDR4	bit	7	6	5	4	3	2	1	0		
Address:000004$_H$		P47	P46	P45	P44	P43	P42	P41	P40	Undefined	R/W *
PDR5	bit	7	6	5	4	3	2	1	0		
Address:000005$_H$		P57	P56	P55	P54	P53	P52	P51	P50	Undefined	R/W *
PDR6	bit	7	6	5	4	3	2	1	0		
Address:000006$_H$		P67	P66	P65	P64	P63	P62	P61	P60	Undefined	R/W *
PDR7	bit	7	6	5	4	3	2	1	0		
Address:000007$_H$		P77	P76	P75	P74	P73	P72	P71	P70	Undefined	R/W *
PDR8	bit	7	6	5	4	3	2	1	0		
Address:000008$_H$		P87	P86	P85	P84	P83	P82	P81	P80	Undefined	R/W *
PDR9	bit	7	6	5	4	3	2	1	0		
Address:000009$_H$		P97	P96	P95	P94	P93	P92	P91	P90	Undefined	R/W *
PDRA	bit	7	6	5	4	3	2	1	0		
Address:00000A$_H$		-	-	-	-	PA3	PA2	PA1	PA0	Undefined	R/W *

*: The R/W access to the I/O port operates slightly different from the R/W access to the memory.

The read value when accessing the I/O port are shown below.

	When normal port is used		When resource output is enabled	
	Input mode (DDR=0)	Output mode (DDR=1)	Input mode (DDR=0)	Output mode (DDR=1)
PDR read	Pin status	PDR value	Pin status	PDR value
PDR RMW	PDR value	PDR value	PDR value	PDR value
DDR read	DDR value	DDR value	DDR value	DDR value
DDR RMW	DDR value	DDR value	DDR value	DDR value

8.2.2 Port Direction Registers (DDR0 to DDRA)

This section explains the configuration and functions of port direction registers (DDR0 to DDRA).

Port Direction Registers (DDR0 to DDRA)

Figure 8-2 shows a list of port direction registers (DDR0 to DDRA).

Figure 8-2. List of Port Direction Registers (DDR0 to DDRA)

DDR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:000010H		D07	D06	D05	D04	D03	D02	D01	D00	00000000B	R/W
DDR1	bit	7	6	5	4	3	2	1	0		
Address:000011H		D17	D16	D15	D14	D13	D12	D11	D10	00000000B	R/W
DDR2	bit	7	6	5	4	3	2	1	0		
Address:000012H		D27	D26	D25	D24	D23	D22	D21	D20	00000000B	R/W
DDR3	bit	7	6	5	4	3	2	1	0		
Address:000013H		D37	D36	D35	D34	D33	D32	D31	D30	00000000B	R/W
DDR4	bit	7	6	5	4	3	2	1	0		
Address:000014H		D47	D46	D45	D44	D43	D42	D41	D40	00000000B	R/W
DDR5	bit	7	6	5	4	3	2	1	0		
Address:000015H		D57	D56	D55	D54	D53	D52	D51	D50	00000000B	R/W
DDR6	bit	7	6	5	4	3	2	1	0		
Address:000016H		D67	D66	D65	D64	D63	D62	D61	D60	00000000B	R/W
DDR7	bit	7	6	5	4	3	2	1	0		
Address:000017H		-	D76	D75	D74	D73	D72	D71	D70	-0000000B	R/W
DDR8	bit	7	6	5	4	3	2	1	0		
Address:000018H		D87	D86	D85	D84	D83	D82	D81	D80	00000000B	R/W
DDR9	bit	7	6	5	4	3	2	1	0		
Address:000019H		D97	D96	D95	D94	D93	D92	D91	D90	00000000B	R/W
DDRA	bit	7	6	5	4	3	2	1	0		
Address:00001AH		-	-	-	-	DA3	DA2	DA1	DA0	----0000B	R/W

When each pin functions as a port

When each pin functions as a port, it controls the corresponding pin as follows:

- 0: Input mode
- 1: Output mode, which can be set to “0” by a reset.

Note:

When a pin for input is switched to a pin for output, be sure to define DDR after writing the desired value to PDR, and then switch it to a pin for output.

8.2.3 Other Registers

This section explains the configuration and functions of registers other than port data registers (PDR0 to PDRA) and port direction registers (DDR0 to DDRA).

Port Input Resistor Registers (RDR0, RDR1)

The bit configuration of port input resistor registers (RDR0, RDR1) is shown in the figure below.

RDR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:00001E$_H$		RD07	RD06	RD05	RD04	RD03	RD02	RD01	RD00	00000000$_B$	R/W
RDR1	bit	7	6	5	4	3	2	1	0		
Address:00001F$_H$		RD17	RD16	RD15	RD14	RD13	RD12	RD11	RD10	00000000$_B$	R/W

Port input resistor registers (RDR0, RDR1) specify whether or not there is pull-up resistor in the input mode.

- 0: No pull-up resistor in the input mode
- 1: Pull-up resistor in the input mode

These registers have no function in the output mode (no pull-up resistor).

The input or output mode is determined by the setting on the port direction register (DDR).

During stop (SPL = 1), the lack of pull-up resistor is specified (high impedance).

This function is prohibited if an external bus is used. Do not write to these registers.

Output Level Set Registers (OLSR0, OLSR1)

The bit configuration of output level set registers (OLSR0, OLSR1) is shown in the figure below.

OLSR0	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:00003C$_H$		-	OD76	OD75	OD74	-	-	-	-	-000----$_B$	R/W
relevant pins			P76	P75	P74						
			UCK5	UO5	-						
			*	*	*						
OLSR1	bit	7	6	5	4	3	2	1	0		
Address:00003D$_H$		OD87	OD86	OD85	OD84	OD83	OD82	OD81	OD80	00000000$_B$	R/W
relevant pins		P87	P86	P85	P84	P83	P82	P81	P80		
		-	UCK0	UO0	-	-	UCK6	UO6	-		
		*	*	*	*	*	*	*	*		

*:5V tolerant corresponding pin

Output level set registers (OLSR0, OLSR1) control open drain when port 7 (P76 to P74) and port 8 (P87 to P80) are in output mode.

- 0: Sets a standard output port in the output mode
- 1: Sets an open-drain output port in the output mode

Port output pin registers (OLSR0, OLSR1) have no function in the input mode (Output Hi-Z).

The input or output mode is determined by the setting of the port direction register (DDR).

Serial Input Level Select Registers (ILSR0, ILSR1, ILSR2)

The bit configuration of the serial input level select registers (ILSR0, ILSR1, ILSR2) is shown in the figure below.

	bit	7	6	5	4	3	2	1	0	Initial value	Access
ILSR0											
Address:00000CH		ILS7	ILS6	IL5	ILS4	ILS3	ILS2	ILS1	ILS0	00000000_B	R/W
relevant pins		P34	P33	P32	P31	P30	P86	P85	P84		
		UO2	UI2	UCK1	UO1	UI1	UCK0	UO0	UI0		

	bit	15	14	13	12	11	10	9	8	Initial value	Access
ILSR1											
Address:00000DH		ILS15	ILS14	ILS13	ILS12	ILS11	ILS10	ILS9	ILS8	00000000_B	R/W
relevant pins		P74	P47	P46	P45	P44	P43	P42	P35		
		UI5	UCK4	UO4	UI4	UCK3	UO3	UI3	UCK2		

	bit	7	6	5	4	3	2	1	0	Initial value	Access
ILSR2											
Address:00000EH		ILS23	ILS22	ILS21	ILS20	ILS19	ILS18	ILS17	ILS16	$---00000_B$	R/W
relevant pins		-	-	-	P82	P81	P80	P76	P75		
		-	-	-	UCK6	UO6	UI6	UCK5	UO5		

The serial input level selection registers (ILSR0, ILSR1, and ILSR2) select the input level of the multi-function serial.

- 0: Sets the input level to CMOS hysteresis.
- 1: Sets the input level to I^2C-standard CMOS hysteresis.

Analog Input Enable Registers (ADER0, ADER1, ADER2)

The bit configuration of the analog input enable registers (ADER0, ADER1, ADER2) is shown in the figure below.

	bit	7	6	5	4	3	2	1	0	Initial value	Access
ADER0											
Address:00001BH		ADE7	ADE6	ADE5	ADE4	ADE3	ADE2	ADE1	ADE0	11111111_B	R/W

	bit	15	14	13	12	11	10	9	8	Initial value	Access
ADER1											
Address:00001CH		ADE15	ADE14	ADE13	ADE12	ADE11	ADE10	ADE9	ADE8	11111111_B	R/W

	bit	7	6	5	4	3	2	1	0	Initial value	Access
ADER2											
Address:00001DH		-	-	-	-	ADE19	ADE18	ADE17	ADE16	$----1111_B$	R/W

Analog input enable registers (ADER0, ADER1 and ADER2) control each pin of port 6 (P67 to P60), port 9 (P97 to P90) and port 7 (P73 to P70) as follows.

- 0: Sets the port I/O mode
- 1: Sets the analog input mode. “1” is restored by a reset.

In the MB90880 series, each bit is set as follows:

- ADE0: P60/AN0 ADE8: P90/AN8 ADE16: P70/AN16
- ADE1: P61/AN1 ADE9: P91/AN9 ADE17: P71/AN17
- ADE2: P62/AN2 ADE10: P92/AN10 ADE18: P72/AN18
- ADE3: P63/AN3 ADE11: P93/AN11 ADE19: P73/AN19
- ADE4: P64/AN4 ADE12: P94/AN12
- ADE5: P65/AN5 ADE13: P95/AN13
- ADE6: P66/AN6 ADE14: P96/AN14
- ADE7: P67/AN7 ADE15: P97/AN15

Up/Down Timer Input Enable Register (UDER)

The bit configuration of the up/down timer input enable register (UDER) is shown in the figure below.

UDER	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:00000B$_H$		-	-	UDE5	UDE4	UDE3	UDE2	UDE1	UDE0	XX000000$_B$	R/W

The up/down timer input enable register (UDER) controls the pins of port 3 (P37 to P35, P32 to P30) as follows:

- 0: Sets the port input mode
- 1: Sets the up/down timer input mode. “0” is restored by a reset.

In the MB90880 series, each bit is set as follows:

- UDE0: P30/ZIN0
- UDE1: P31/AIN0
- UDE2: P32/BIN0
- UDE3: P35/ZIN1
- UDE4: P36/AIN1
- UDE5: P37/BIN1

PPG Terminal Assignment Select Register (PAFSR)

The bit configuration of the PPG terminal assignment select register (PAFSR) is shown in the figure below.

PAFSR	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:00799A$_H$		-	-	-	-	PA3FS	PA2FS	PA1FS	PA0FS	----0000$_B$	R/W
relevant channel						PPG7	PPG6	PPG5	PPG4		

The PPG terminal assignment select register (PAFSR) controls the pin assignment of PPG7 to PPG4.

- 0:Selects the normal pin assignment (PPG7:P57, PPG6:P56, PPG5:P55, PPG4:P54)
- 1:Switches the pin assignment (PPG7:PA3, PPG6:PA2, PPG5:PA1, PPG4:PA0)

Serial Terminal Assignment Select Register 0 (P7FSR)

The bit configuration of the serial terminal assignment select register 0 (P7FSR) is shown in the figure below.

P7FSR	bit	7	6	5	4	3	2	1	0	Initial value	Access
Address:0079A2H		—	—	—	—	P73FS	P72FS	P71FS	RESV	----000XB	R/W
relevant channel						UCK4	UO4	UI4			

The serial terminal assignment select register 0 (P7FSR) controls the pin assignment of UCK4, UO4, and UI4.

- 0:Selects the normal pin assignment (UCK4:P47, UO4:P46, UI4:P45)
- 1:Switches the pin assignment (UCK4:P73, UO4:P72, UI4:P71)

Be sure to set “0” to RESEV bit (bit0).

Serial Terminal Assignment Select Register 1 (P9FSR)

The bit configuration of the serial terminal assignment select register 1 (P9FSR) is shown in the figure below.

P9FSR	bit	15	14	13	12	11	10	9	8	Initial value	Access
Address:00799DH		—	—	—	—	—	P97FS	P96FS	P95FS	-----000B	R/W
relevant channel							UCK3	UO3	UI3		

The serial terminal assignment select register 1 (P9FSR) controls the pin assignment of UCK3, UO3, and UI3.

- 0:Selects the normal pin assignment (UCK3:P44, UO3:P43, UI3:P42)
- 1:Switches the pin assignment (UCK3:P97, UO3:P96, UI3:P95)

9. Time-base Timer

This chapter provides an overview of the time-base timer explains the configuration, the control register, interrupt, its operation, notes, and program example of the time-base timer.

9.1 Overview of Time-base Timer

9.2 Configuration of Time-base Timer

9.3 Time-base Timer Control Register (TBTC)

9.4 Interrupt of Time-base Timer

9.5 Operation of Time-base Timer

9.6 Notes on Using Time-base Timer

9.1 Overview of Time-base Timer

The time-base timer, which is an 18-bit free-run counter (time-base timer counter) that counts up in synchronization with the internal count clock (the source oscillation divided by 2), has the interval timer function to enable selection for four types of interval times. Furthermore, it also has functions to supply operation clocks including timer output for the oscillation stabilization wait time as well as the watchdog timer.

Interval Timer Function

The interval timer function generates repetitive interval interrupt requests.

- An interrupt request is generated when the bit for the interval timer in the time-base counter overflows.
- The bit for the interval timer (interval time) can be selected out of four types.

Table 9-1 shows the interval time of the time-base timer.

Table 9-1. Interval Time of Time-base Timer

Internal count clock cycle	Interval cycle
2 / HCLK (0.5 μs)	2^{12} / HCLK (approximately 1.0 ms)
	2^{14} / HCLK (approximately 4.1 ms)
	2^{16} / HCLK (approximately 16.4 ms)
	2^{19} / HCLK (approximately 131.1 ms)

HCLK: Oscillation clock

The value during operation of the oscillation clock at 4 MHz is shown in ( ).

Clock Supplying Function

The clock supplying function is the function supplying the timer for the oscillation stabilization wait time and the operation clocks to some peripheral functions. Table 9-2 lists the cycles of the clocks supplied by the time-base timer to individual peripheral functions.

Table 9-2. Clock Cycles Supplied by Time-base Timer

Function to which clock is supplied	Clock cycle	Remarks
Oscillation stabilization wait time	2^{13} / HCLK (approximately 2.0 ms)	Oscillation stabilization wait time for ceramic resonator
	2^{15} / HCLK (approximately 8.2 ms)	Oscillation stabilization wait time for crystal resonator
	2^{17} / HCLK (approximately 32.8 ms)	
Watchdog timer	2^{12} / HCLK (approximately 1.0 ms)	Count up clock for watchdog timers
	2^{14} / HCLK (approximately 4.1 ms)	
	2^{16} / HCLK (approximately 16.4 ms)	
	2^{19} / HCLK (approximately 131.1 ms)	

HCLK: Oscillation clock

The value during operation of the oscillation clock at 4 MHz is shown in ( ).

Because the oscillation cycle immediately after the start of oscillation is unstable, the oscillation stabilization wait time is merely a guideline.

9.2 Configuration of Time-base Timer

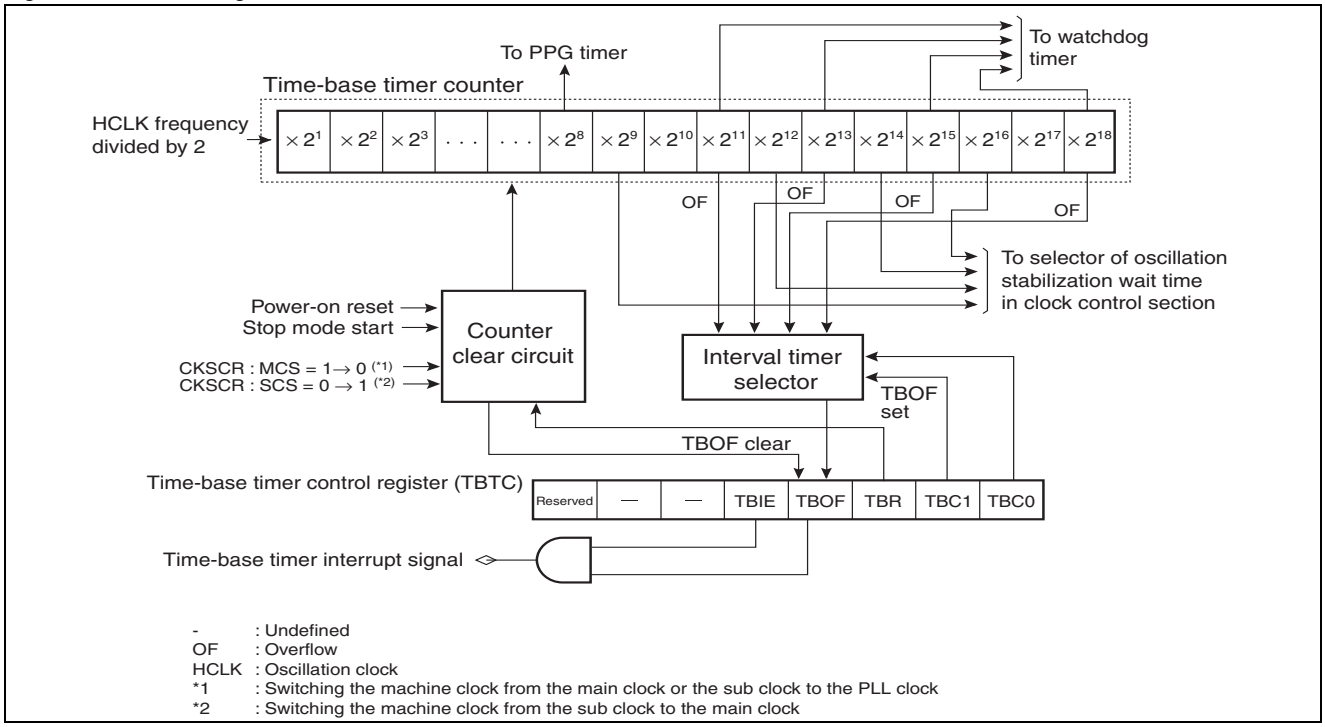
The time-base timer is composed of the following four blocks:

- Time-base timer counter
- Counter clear circuit
- Interval timer selector
- Time-base timer control register (TBTC)

Block Diagram of Time-base Timer

Figure 9-1 is a block diagram of the time-base timer.

Figure 9-1. Block Diagram of Time-base Timer



Time-base timer counter

This is an 18-bit up-counter that uses the oscillation clock (HCLK) frequency divided by 2 as the count clock.

Counter clear circuit

This circuit clears the counter at the time of writing of "0" to the time-base timer initializing bit (TBR) in the time-base timer control register (TBTC), a power-on reset, a transition to the main stop mode, a transition to the PLL stop mode, switching from the main clock mode to the PLL clock mode, switching from sub clock mode to the PLL clock mode, and switching from the sub clock mode to main clock mode.

Interval timer selector

Selects one of the four types for time-base timer counter output. Overflow of the selected bit causes an interrupt.

Time-base timer control register (TBTC)

Selects the interval time, clears the counter, controls interrupt requests, and checks the current state.

9.3 Time-base Timer Control Register (TBTC)

This register selects the interval time, clears the counter, controls interrupt requests, and checks the state.

Time-base Timer Control Register (TBTC)

Figure 9-2. Time-base Timer Control Register (TBTC)

Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7 ... bit0	Initial value
0000A9$_H$	Reserved	—	—	TBIE	TBOF	TBR	TBC1	TBC0	(WDTC)	1XX00100$_B$
	R/W	—	—	R/W	R/W	W	R/W	R/W		

TBC1	TBC0	Interval time selection bits
0	0	2^{12}/HCLK (approximately 1.0 ms)
0	1	2^{14}/HCLK (approximately 4.1 ms)
1	0	2^{16}/HCLK (approximately 16.4 ms)
1	1	2^{19}/HCLK (approximately 131 ms)

The value during operation of the oscillation clock at 4 MHz is shown in ( ).

TBR	Time-base timer initializing bit	
	Read	Write
0	—	Clears TBOF bit of time-base timer counter.
1	"1" is always read	No change, no effect on other functions

TBOF	Interrupt request flag bit	
	Read	Write
0	No overflow of specified bit	Clears this bit.
1	Overflow of specified bit	No change, no effect on other functions

TBIE	Interrupt request permit bit
0	Prohibits interrupt request output.
1	Permits interrupt request output.

Reserved	Reserved bit
Always write "1" to this bit.	

R/W : Readable/Writable
W : Write only
- : Undefined
x : Indefinite
HCLK : Oscillation clock
: Initial value

Table 9-3. Functions of Bits in Time-base Timer Control Register (TBTC)

Bit name		Function
bit15	Reserved bit	Always write “1” to this bit.
bit14, bit13	Undefined bits	■ Read value is undefined ■ Writing has no effect on operation
bit12	TBIE: Interrupt request permit bit	This bit permits or prohibits output of interrupt requests to the CPU. ■ An interrupt request is output if this bit and the interrupt request flag bit (TBOF) are set to “1”.
bit11	TBOF: Interrupt request flag bit	This bit is set to “1” when the bit specified by the time-base timer counter overflows. ■ An interrupt request is output if this bit and the interrupt request permit bit (TBIE) are set to “1”. ■ This bit is cleared by writing “0” but is not changed by writing “1” so that this operation does not affect other functions. **Notes:** ■ Clearing of the interrupt request flag bit (TBOF) must be implemented in the state where the time-base timer interrupt is prohibited by the interrupt request permit bit (TBIE) or by the interrupt level mask register (ILM) setting of the processor status (PS). ■ This bit is cleared to “0” by writing “0”, a transition to the main stop mode, a transition to the PLL stop mode, a transition from the sub clock mode to the main clock mode, a transition from the sub clock mode to the PLL clock mode, a transition from the main clock mode to the PLL clock mode, writing "0" to the time-base timer initializing bit (TBR) or a reset.
bit10	TBR: Time-base timer initializing bit	This bit clears the time-base timer counter. ■ When “0” is written to this bit, the counter is cleared and the TBOF bit is cleared. This bit is not changed by writing “1” so that this operation does not affect other functions. The readout value is always “1”.
bit9, bit8	TBC1, TBC0: Interval time selection bits	This bit specifies the cycle of the interval timer. ■ The bit for the interval timer in the time-base timer counter is specified. ■ The interval time can be selected out of four types.

9.4 Interrupt of Time-base Timer

The time-base timer can generate the interrupt request caused by an overflow of the specified bit in the time-base timer counter (interval timer function).

Interrupt of Time-base Timer

When the time-base timer counter counts up using the internal count clock and the bit for the selected interval timer overflows, the interrupt request flag bit (TBOF) of the time-base timer control register (TBTC) is set to “1”. In this event, if an interrupt request is permitted because the interrupt request permit bit (TBIE) is set to “1”, an interrupt request is generated in the CPU. Clear this interrupt request by writing “0” to the TBOF bit using the interrupt processing routine. Incidentally, TBOF bit is set when the specified bit overflows regardless for the value for the interrupt request permit bit (TBIE).

Notes:

- Clearing of the interrupt request flag bit (TBOF) in the time-base timer control register (TBTC) must be implemented in the state where the time-base timer interrupt is prohibited by the interrupt request permit bit (TBIE) or by the interrupt level mask register (ILM) setting of the processor status (PS).
- If the TBOF bit is set to “1”, an interrupt request is generated immediately when the TBIE bit is switched from prohibit (0) to permit (1).
- μDMAC cannot be used in the time-base timer.

Time-base Timer Interrupt and μDMAC

Table 9-4 lists time-base timer interrupt and μDMAC.

Table 9-4. Time-base Timer Interrupt and μDMAC

Interrupt No.	Interrupt level setting register		Address of the vector table			μDMAC
	Register name	Address	Low-order	High-order	Bank	
#41	ICR15	$0000BF_H$	$FFFF58_H$	$FFFF59_H$	$FFFF5A_H$	Not used

Note:

ICR15 is commonly used by the time-base timer interrupt, the watch timer interrupt, and flash write. Although interrupt can be used for these three purposes, the interrupt level is the same.

9.5 Operation of Time-base Timer

The time-base timer has the interval timer function, the timer function for the oscillation stabilization wait time, and the clock supplying function for some peripheral functions.

Operation of Interval Timer Function (Time-base Timer)

The interval timer function generates interrupt requests at any defined interval times. For its operation as an interval timer, the settings shown in Figure 9-3 are required.

Figure 9-3. Time-base Timer Settings

Address		bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit0
0000A9H	TBTC	Reserved	—	—	TBIE	TBOF	TBR	TBC1	TBC0	(WDTC)	
		1	–	–	◎	0	0	◎	◎		

◎ : Used bits
- : Unused bits
0 : "0" is set
1 : "1" is set

- The time-base timer counter continues counting up in synchronization with the internal count clock (oscillation clock divided by 2) as long as the clock is oscillating.
- When the counter has been cleared, counting starts from “0”, and an overflow of the bit used for the interval timer sets the interrupt request flag bit (TBOF) to “1”. In this event, if the interrupt request output is permitted (TBIE = 1), interrupts are generated at the selected interval times with the clearing time used as a reference time point.
- The interval time may become longer than the specified time, when the time-base timer was cleared.

Timer Function for Oscillation Stabilization Wait Time

The time-base timer can also be used as the oscillation clock as well as the timer for the oscillation stabilization wait time of the PLL clock. The oscillation stabilization wait time is the time in which counting starts when the counter is set to “0” (clearing of counter) and continues until an overflow occurs in the bit for the oscillation stabilization wait time. However, during return from the time-base timer mode to the PLL clock mode or main clock mode, the waiting time differs because the time-base timer counter is not cleared and the time count does not start from zero. Table 9-5 explains the time-base timer counter clear operation and oscillation stabilization wait time.

Table 9-5. Time-base Timer Counter Clear Operation and Oscillation Stabilization Wait Time

Operation	Time-base timer counter	TBOF bit	Oscillation stabilization wait time
Writing “0” to time-base timer initializing bit (TBR) for time-base timer control register (TBTC)	○	○	None
Power-on reset	○	○	Oscillation stabilization wait time of main clock
Watchdog reset	×	○	
Release of the main stop mode	○	○	
Release of the PLL stop mode	○	○	
Release of the sub stop mode	×	×	Oscillation stabilization wait time of sub clock
Switching from main clock mode to PLL clock mode (MCS: transition from 1 to 0)	○	○	Oscillation stabilization wait time of PLL clock
Transition from sub clock mode to main clock mode (SCM: transition from 0 to 1)	○	○	Oscillation stabilization wait time of main clock
Release of time-base timer mode	×	×	None
Release of sleep mode	×	×	None

○: Cleared
×: Not cleared

Clock Supplying Function

The time-base timer supplies a clock to the watchdog timer. Clearing of the time-base timer counter, the interval time of watchdog timer may be longer than the specified time.

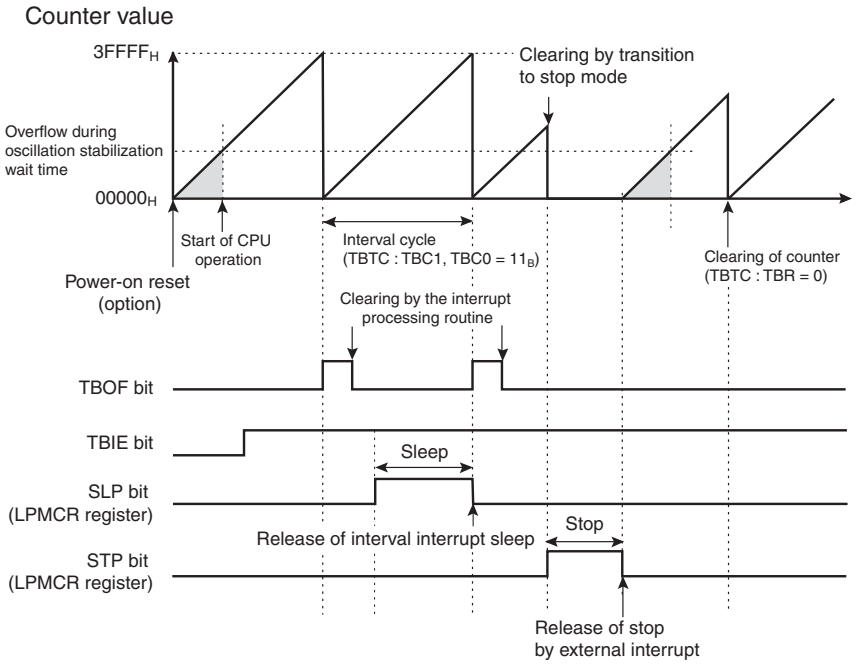
Operation Example of Time-base Timer

Operations in the following states are shown in Figure 9-4:

- Where the power-on reset has occurred
- Where transition to the sleep mode has occurred during processing for the interval timer function
- Where transition to the stop mode has occurred
- Where clearing of the counter is requested

Transition to the stop mode clears the time-base timer to stop operation. After restoration from the stop mode, the time-base timer starts an count up of the oscillation stabilization wait time.

Figure 9-4. Operation Example of Time-base Timer



"11$_B$" is set to the interval time selection bits (TBTC; TBC1, TBC0) in the time-base timer control register (2^{19}/HCLK).

: Oscillation stabilization wait time

HCLK : Oscillation clock

9.6 Notes on Using Time-base Timer

This section explains notes on using the time-base timer, including the effects of clearing an interrupt request or clearing the time-base timer on peripheral functions.

Notes on Using Time-base Timer

Clearing an interrupt request

Clearing the interrupt request flag bit (TBOF) of the time-base timer control register (TBTC) must be implemented in the state where the time-base timer interrupt is prohibited by the interrupt request permit bit (TBIE) or by the interrupt level mask register (ILM) setting of the processor status (PS).

Effect of clearing the time-base timer

Clearing the time-base timer counter affects the following:

- Operations where the interval timer function (interval interrupt) is used by the time-base timer, the interval cycle may become longer.
- Operations using the watchdog timer, the interval time of the watchdog timer may become longer.

Use of the timer for the oscillation stabilization wait time

When power is turned on, the oscillation clock is stopped in the main stop mode. In such a case, after the oscillator starts operating, the oscillation stabilization wait time of the oscillation clock must be provided by using as a timing reference the operation clock supplied by the time-base timer. An appropriate oscillation stabilization wait time must be selected depending on the type of oscillator (resonator) connected to the high-speed oscillation pin. See Section “5.5 Oscillation Stabilization Wait Time”, for details.

Caution on using peripheral functions whose operation clock is supplied from the time-base timer

In a mode where the main clock stops, the counter is cleared and the time-base timer stops operating. Furthermore, because the clock supplied by the time-base timer is reset to the initial state and is supplied again when the time-base timer counter is cleared, the period of “H” level may become shorter or the period of “L” level may become longer by a maximum of a 1/2 cycle. Although the clock for the watchdog timer is also supplied from the initial state, the watchdog timer operates at normal cycles because the watchdog time counter is cleared at the same time.

10. Watchdog Timer

This chapter explains the function and operation of the watchdog timer.

10.1 Overview of Watchdog Timer

10.2 Configuration of Watchdog Timer

10.3 Register for Watchdog Timer

10.4 Operation of Watchdog Timer

10.5 Precautions for Using Watchdog Timer

10.1 Overview of Watchdog Timer

Watchdog timer is a 2 bit-counter that uses time-base timer or watch timer as a count clock. CPU will be reset if the counter is not cleared within interval time.

Watchdog Timer Function

- Watchdog timer is a timer counter that observes with malfunction of program. On booting watchdog timer, it should continue clearing within the interval time set at the counter of watchdog timer. If the interval time reaches the set time without clearing the counter of the watchdog timer, CPU will be reset. This is so called a watchdog timer.
- Interval time of watchdog timer is based on the clock period input as a count clock and watchdog reset will occur between minimum time and maximum time.
- Output target of clock source is set at the watchdog clock select bit of watch timer control register (WTC: WDCS).
- Interval time of watchdog timer is set at the time-base timer output select bit/watch timer output select bit of watchdog timer control register (WDTC: WT1, WT0).

Interval time of watchdog timer is shown in the Table 10-1.

Table 10-1. Interval Time of Watchdog Timer

Minimum	Maximum	Clock Period	Minimum	Maximum	Clock Period
Approximately 3.58 ms	Approximately 4.61 ms	$(2^{14} \pm 2^{11})$/ HCLK	Approximately 0.457 s	Approximately 0.576 s	$(2^{12} \pm 2^{9})$/ SCLK
Approximately 14.33 ms	Approximately 18.3 ms	$(2^{16} \pm 2^{13})$/ HCLK	Approximately 3.584 s	Approximately 4.608 s	$(2^{15} \pm 2^{12})$/ SCLK
Approximately 57.23 ms	Approximately 73.73 ms	$(2^{18} \pm 2^{15})$/ HCLK	Approximately 7.168 s	Approximately 9.216 s	$(2^{16} \pm 2^{13})$/ SCLK
Approximately 458.75 ms	Approximately 589.82 ms	$(2^{21} \pm 2^{18})$/ HCLK	Approximately 14.336 s	Approximately 18.432 s	$(2^{17} \pm 2^{14})$/ SCLK

HCLK: Oscillation Clock (4 MHz), SCLK: Sub clock (8.192 kHz)

Notes:

- In case that count clock of watchdog timer is used for time-base timer output (a carry signal), the generation time of the watchdog reset of may delayed on clearing time-base timer.
- In case of using sub clock as machine clock, be sure to set watchdog timer clock source select bit (WDCS) of watch timer control register (WTC) to “0” to select the output of watch timer.

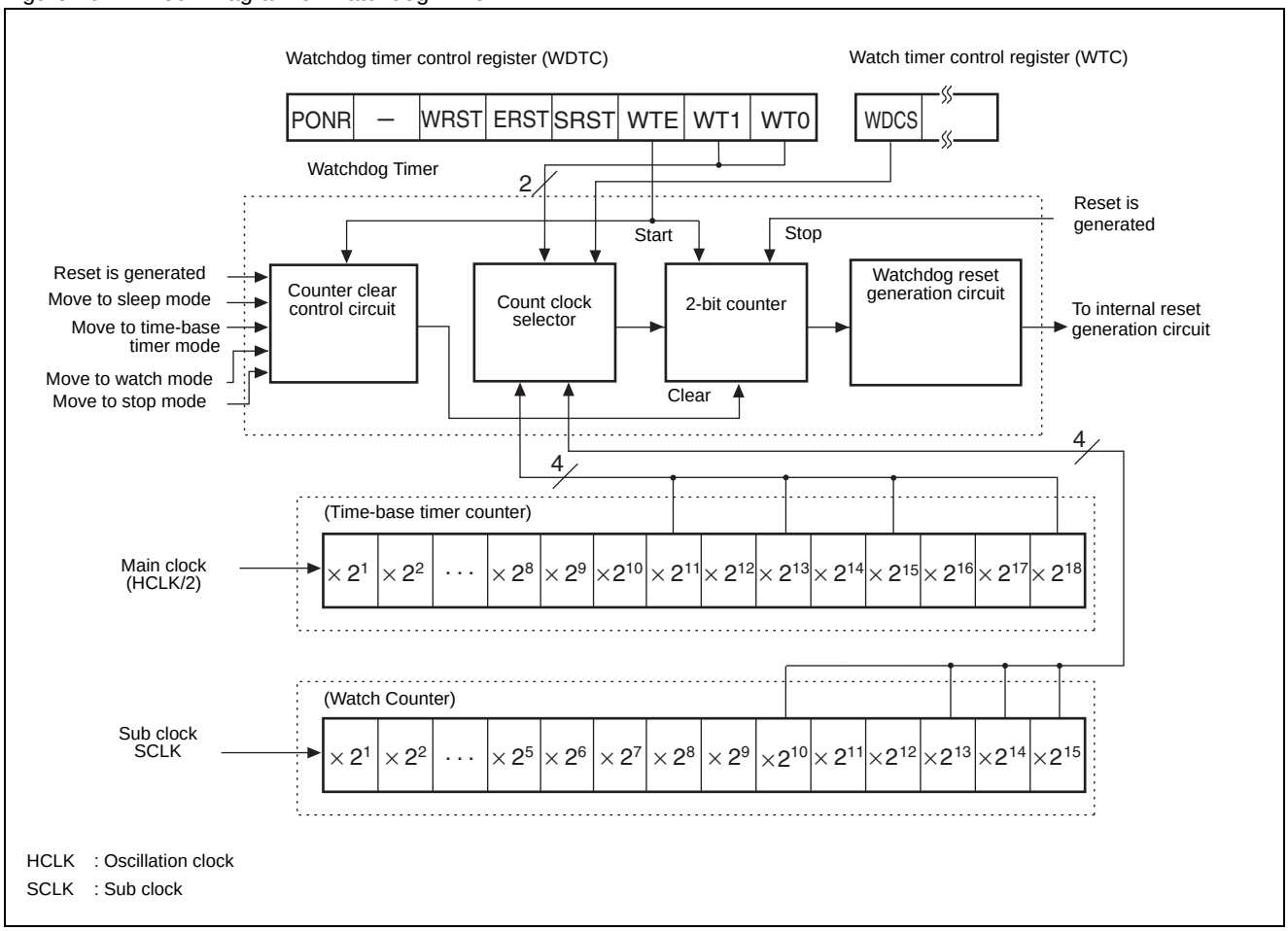
10.2 Configuration of Watchdog Timer

Watchdog timer consists of following blocks.

- Count clock selector
- Watchdog timer counter (2-bit counter)
- Watchdog reset generation circuit
- Counter clear control circuit
- Watchdog timer control register (WDTC)

Block Diagram of Watchdog Timer

Figure 10-1. Block Diagram of Watchdog Timer



Count Clock Selector

Selects count clock input to watchdog timer from time-base timer or watch timer. 4 types of interval time can be set from each output of timer.

Watchdog Timer Counter (2-bit counter)

2-bit up counter that handles output of time-base timer or watch timer as count clock. Target of the clock source output is set by watchdog clock select bit of watch timer control register (WTC: WDCS).

Watchdog Reset Generation Circuit

Generates reset signal by overflow (carry) of the watchdog timer.

Counter Clear Circuit

Clears counter of watchdog timer.

Watchdog Timer Control Register (WDTC)

Boots and clears the watchdog timer, sets the interval time and holds the reset factor.

10.3 Register for Watchdog Timer

Register for setting the watchdog timer is explained as follows.

List of Watchdog Timer Register and Initial Value

Figure 10-2. List of Watchdog Timer Register and Initial Value

Watchdog timer control register (WDTC)

bit	7	6	5	4	3	2	1	0
	×	×	×	×	×	1	1	1

× : Undefined

10.3.1 Watchdog Timer Control Register (WDTC)

Boots and clears watchdog timer, sets the interval time and holds the reset factors.

Watchdog Timer Control Register (WDTC)

Figure 10-3. Watchdog Timer Control Register (WDTC)

bit	7	6	5	4	3	2	1	0
Address 0000A8H	PONR	–	WRST	ERST	SRST	WTE	WT1	WT0
	R	–	R	R	R	W	W	W

Initial value: XXXXX111B

WT1	WT0	Interval time select bits (Time-base timer output select)		
		Interval time		Clock period
		Minimum	Maximum	
0	0	Approximately 3.58 ms	Approximately 4.61 ms	$(2^{14} \pm 2^{11})$/HCLK
0	1	Approximately 14.33 ms	Approximately 18.3 ms	$(2^{16} \pm 2^{13})$/HCLK
1	0	Approximately 57.23 ms	Approximately 73.73 ms	$(2^{18} \pm 2^{15})$/HCLK
1	1	Approximately 458.75 ms	Approximately 589.82 ms	$(2^{21} \pm 2^{18})$/HCLK

HCLK: Oscillation Clock
Parenthesis is interval time in case of operation at HCLK 4 MHz.

WT1	WT0	Interval time select bits (Watch timer output select)		
		Interval time		Clock period
		Minimum	Maximum	
0	0	Approximately 0.457 s	Approximately 0.576 s	$(2^{12} \pm 2^{9})$/SCLK
0	1	Approximately 3.584 s	Approximately 4.608 s	$(2^{15} \pm 2^{12})$/SCLK
1	0	Approximately 7.168 s	Approximately 9.216 s	$(2^{16} \pm 2^{13})$/SCLK
1	1	Approximately 14.336 s	Approximately 18.432 s	$(2^{17} \pm 2^{14})$/SCLK

SCLK: Sub clock
Parenthesis is interval time in case of operation at HCLK 8.192 kHz.

WTE	Watchdog timer control bit	
0	First write after reset: Boots watchdog timer	Second or later write after reset: Clears watchdog timer
1	No effect	

Reset factor bits				Reset factor bits
PONR	WRST	ERST	SRST	
1	X	X	X	Power ON reset
*	1	*	*	Watchdog reset
*	*	1	*	External reset (input “L” to $\overline{\text{RST}}$ pin)
*	*	*	1	Software reset (write “1” to RST bit)

R: Read only
W: Write only
*: Holds previous state
X: Undefined
: Initial value

Note: SCLK is the divide-by-two or divide-by-four of the clock signal input to low-speed oscillation X0A and X1A pins.
The division ratio is set by the SCDS bit in the PLL/sub clock control register (PSCCR). (Refer to "5. Clocks").

Table 10-2. Function of Watchdog Timer Control Register (WDTC)

Bit name		Function
bit7, bit5 to bit3	PONR, WRST, ERST, SRST: Reset factor bits	Indicates reset factors. ■ If reset occurs, applicable bit for reset factors will be set to “1”. The reset factors can be checked by reading watchdog timer control register (WDTC) after reset. ■ Reset factor bits will be cleared after reading watchdog timer control register (WDTC). **Note**: The content except for PONR is not guaranteed after power on reset. In case the PONR is set when read, ignore other contents.
bit6	Undefined Bit	**When read** : Value is undefined. **When write** : No effect
bit2	WTE: Watchdog timer control bit	Boots or clears the watchdog timer. **In case “0” is set.** **(First after reset)** : will be booted. **In case “0” is set** **(Second or later after reset)** : will be cleared.
bit1, bit0	WT1, WT0: Interval time select bits	Set interval time of watchdog timer. Watch timer control register (WTC) changes the interval time as shown in Figure 10-3, when the clock source of the watchdog timer is watch timer (watchdog clock select bit WDCS = 0) and when main clock mode or PLL clock mode is selected as clock mode and WDCS bit of WTC is “1”. When in sub clock mode, be sure to set watchdog clock select bit (WDCS) of watch timer control register (WTC) to “0” to select output of watch timer. ■ The data when watchdog timer is booted is only valid. ■ The write data after watchdog timer is booted is omitted. ■ These bits are write only.

10.4 Operation of Watchdog Timer

After the watchdog timer is activated if the interval time reaches the set time without clearing the counter of the watchdog timer, watchdog reset is generated.

State Transition Diagram for Watchdog Timer

Watchdog timer consists of following four states.

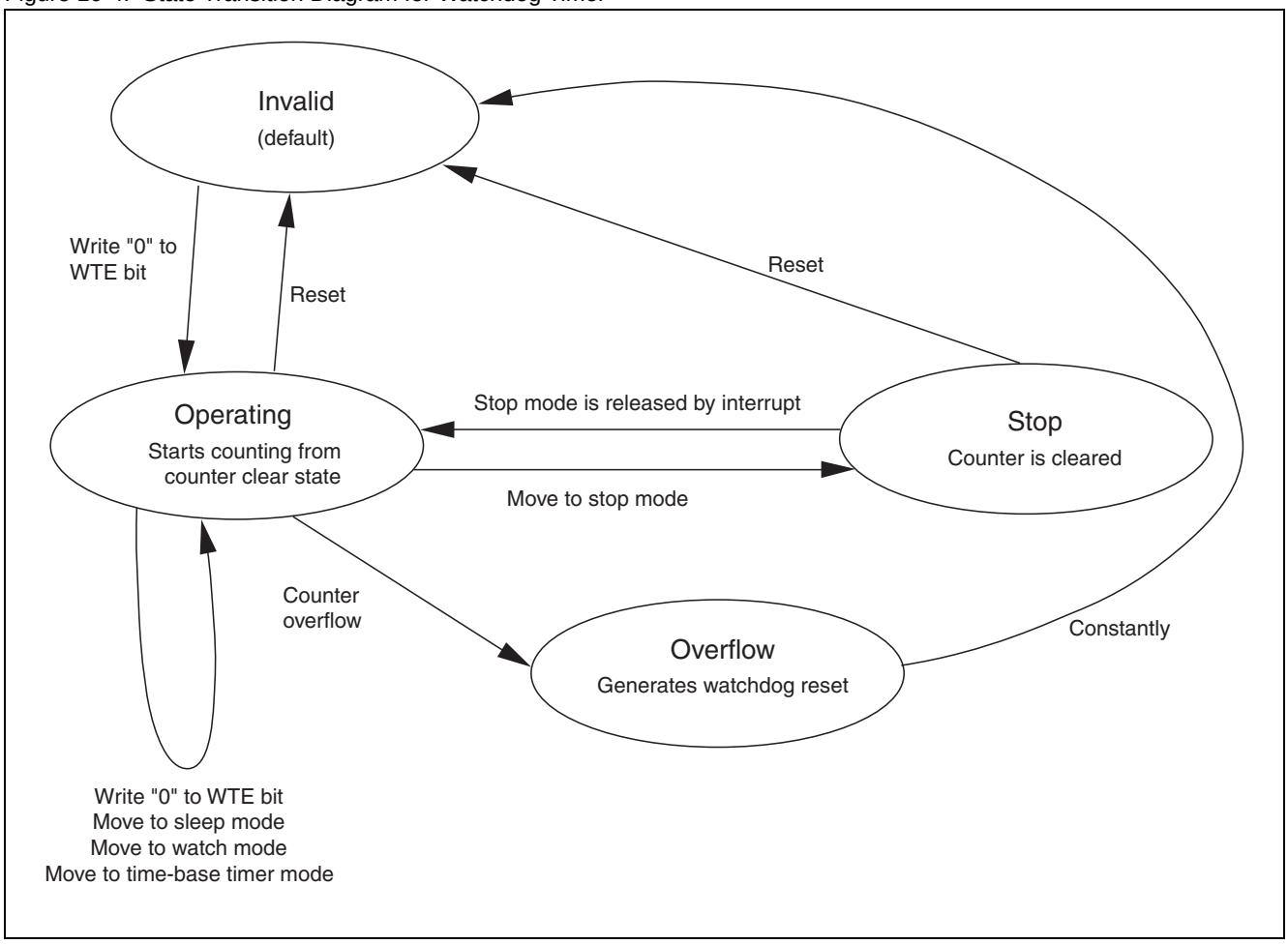
Invalid: Not operating.

Operation: Starts counting from counter clear state.

Stop: Continues counter clear state.

Overflow: Generates watchdog reset.

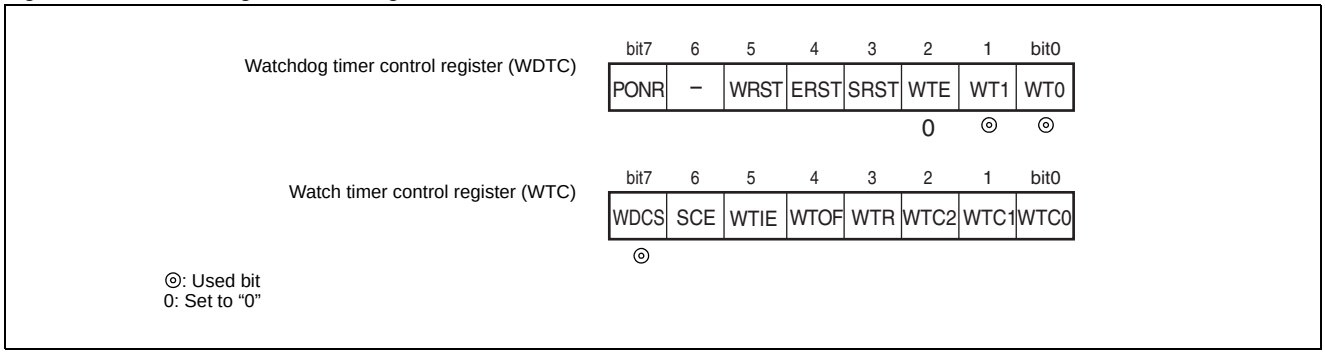
Figure 10-4. State Transition Diagram for Watchdog Timer



Operation of Watchdog Timer

For watchdog timer operation, the setting shown in the Figure 10-5 is required.

Figure 10-5. Watchdog Timer Setting



Selecting Clock Input Source

- Clock input source for count clock of watchdog timer will be selected for time-base timer or watch timer. Time-base timer is selected when watchdog clock select bit (WTC: WDCS) is set to “1” and watch timer is selected when “0”. Reset returns the bit to “1” (time-base timer).
- Select clock timer by setting WDCS bit to “0” when operating in sub clock mode.

Interval Time Setting

- Interval time of watchdog timer is selected by setting interval time select bit (WDTC: WT1, WT0).
- Set the interval time at the same time of booting. Writing to this bit after the watchdog timer is booted will be ignored.

Booting Watchdog Timer

Watchdog timer is booted and starts counting up when “0” is set to watchdog timer control bit (WDTC: WTE) after reset.

Clearing Watchdog Timer

- After booting the watchdog timer, writing “0” to watchdog timer control bit (WDTC: WTE) again within interval time will clear the watchdog timer. If the watchdog timer is not cleared within interval time, it causes overflow and CPU will be reset.
- Generating reset, moving to standby mode (sleep mode, stop mode, watch mode and time-base timer mode) will clear the watchdog timer.
- Counter of watchdog timer will be cleared when time-base timer is running, watch mode is running or is in sleep mode but the watchdog timer will still be activated.
- Relation between clearing timing of watchdog timer and the interval time is shown in the Figure 10-6. The interval time varies depending on the clearing timing of watchdog timer.

Reset Factor Confirmation

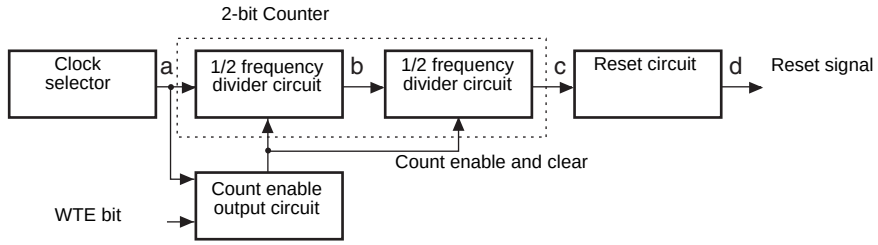
Reset factor can be identified by reading the reset factor bits (WDTC: PONR, WRST, ERST, SRST) of watchdog timer control register (WDTC).

Reference:

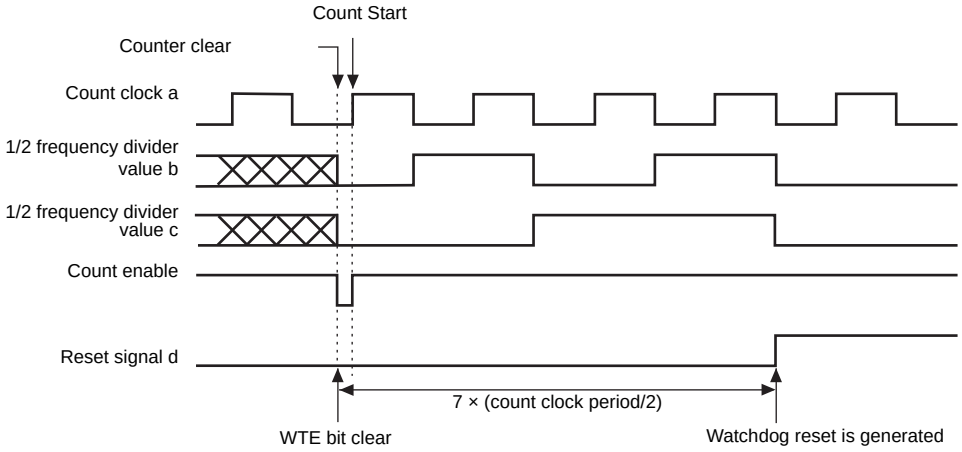
Refer to “4. Reset” for reset factor bit.

Figure 10-6. Clearing Timing of Watchdog Timer and Interval Time

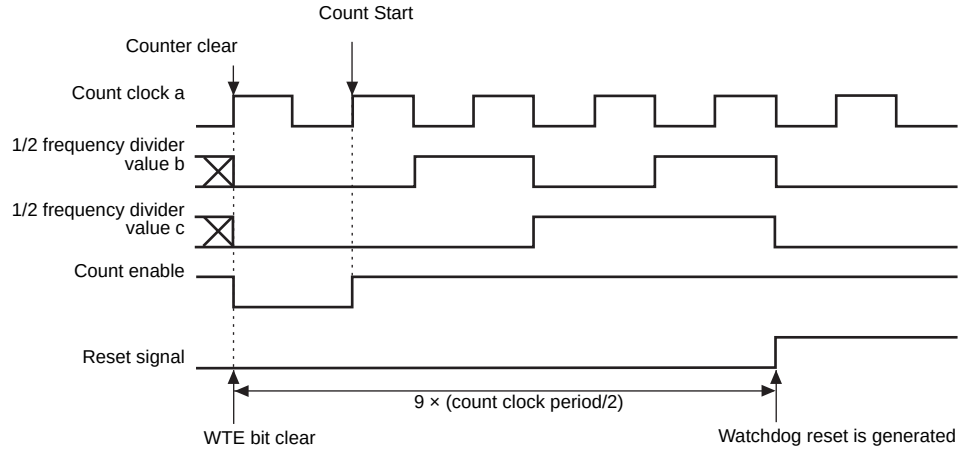
[Block Diagram of Watchdog Timer]



[Minimum Interval Time] When WTE bit is cleared just before the count clock rise



[Maximum Interval Time] When WTE bit is cleared just before the count clock rise



10.5 Precautions for Using Watchdog Timer

Note that following points for using watchdog timer.

Precautions for Using Watchdog Timer

Stopping Watchdog Timer

Watchdog timer will be stopped by all reset factors and moving to stop mode.

Interval Time

- Interval time of watchdog timer may be longer when time-base timer or watch timer is cleared as the count clock uses the carry signal of time-base timer or watch timer. Note that the time-base timer will be cleared when writing “0” to the time-base timer counter clear bit (TBR) of the time-base timer control register (TBTC), moving main clock mode to PLL clock mode, moving from sub clock mode to main clock mode, or moving from sub clock mode to PLL clock mode.
- Be sure to set the interval time at the same time of booting the watchdog timer. Setting except for the booting will be ignored.

Precautions for Programming

Set shorter time than that of interval time of watchdog timer for processing time of main loop including interrupt process time if watchdog timer is cleared repeatedly in the main loop.

In addition, as watchdog timer is running while in DMA transfer, hold condition, sleep mode, time-base timer mode and watch mode, the interval time should be considered about operation time for each mode in user's program.

Precautions for Sub Clock Mode

In case of sub clock mode, be sure to set the watchdog clock select bit (WDCS) of watch timer control register (WTC) to “0” to output the watch timer.

Watchdog Timer Operation while in Sleep Mode, Time-base Timer Mode and Watch Mode

Watchdog timer will be cleared and start counting again (refer to Table 10-3) when moving to sleep mode, time-base timer mode or watch mode.

Watchdog Timer Operation while in DMA Transfer

Watchdog reset may occur while in DMA transfer as watchdog timer is running while in DMA transfer. Consider the prohibition of the reset in user's program (refer to Table 10-3).

Watchdog Timer Operation while in Hold Condition (External Bus Mode)

As watchdog timer is running while CPU is in hold condition, the hold condition may be released by watchdog reset. Consider the prohibition of the reset in user's system (refer to Table 10-3).

Watchdog Timer Operation while in Stop Mode

Watchdog timer will be cleared and stopped after moved to stop mode. Watchdog timer will start counting again after stop mode is released (refer to Table 10-3).

Watchdog Timer Operation while in Resetting

All reset factors disable the watchdog timer. Watchdog timer will still be invalid after reset is released (refer to Table 10-3).

Table 10-3. Watchdog Timer Clear Condition

Operation Mode	Reset	WTE = 0 of WDTC Register	Stop Mode	Sleep Mode	Time-base Timer Mode	Watch Mode	Hold	µDMAC
Clear	When moving	When writing	When moving	When moving	When moving	When moving	N/A	N/A
Watchdog Timer Condition in Mode	Invalid	-	Stop (holds clear condition)	Operate (start counting just after clear)	Operate (start counting just after clear)	Operate (start counting just after clear)	Operate (continues counting)	Operate (continues counting)
Watchdog Reset in Mode	Not generate	-	Not generate	Generate	Generate	Generate	Generate	Generate
Watchdog Timer Condition after Mode Release/Return	Invalid	Operate	Operate (restart counting from clear condition)	Operate (continues counting)	Operate (continues counting)	Operate (continues counting)	Operate (continues counting)	Operate (continues counting)

11. Watch Timer

This chapter provides an overview of the watch timer, explains the configuration, the control register, and the operation of the watch timer.

11.1 Overview of Watch Timer

11.2 Configuration of Watch Timer

11.3 Watch Timer Control Register (WTC)

11.4 Operation of Watch Timer

11.1 Overview of Watch Timer

The watch timer is a 15-bit timer using the sub clock. This timer can generate interval interrupts. Furthermore, depending on the setting, this timer can be used as the clock source for the watchdog timer.

Functions of Watch Timer

The watch timer is composed of a 15-bit timer and a circuit to control interval interrupts.

The watch timer uses the sub clock regardless of the PLL clock selection bit (MCS) or the sub clock selection bit (SCS) in the clock selection register (CKSCR).

Table 11-1 lists the interval times of the watch timer.

Table 11-1. Interval Times of Watch Timer

WTC2	WTC1	WTC0	Interval time *
0	0	0	31.25 ms
0	0	1	62.5 ms
0	1	0	125 ms
0	1	1	250 ms
1	0	0	500 ms
1	0	1	1.000 s
1	1	0	2.000 s
1	1	1	Setting is prohibited

*: Sub clock: 32 kHz frequency divided by 4 (= 8 kHz)

11.2 Configuration of Watch Timer

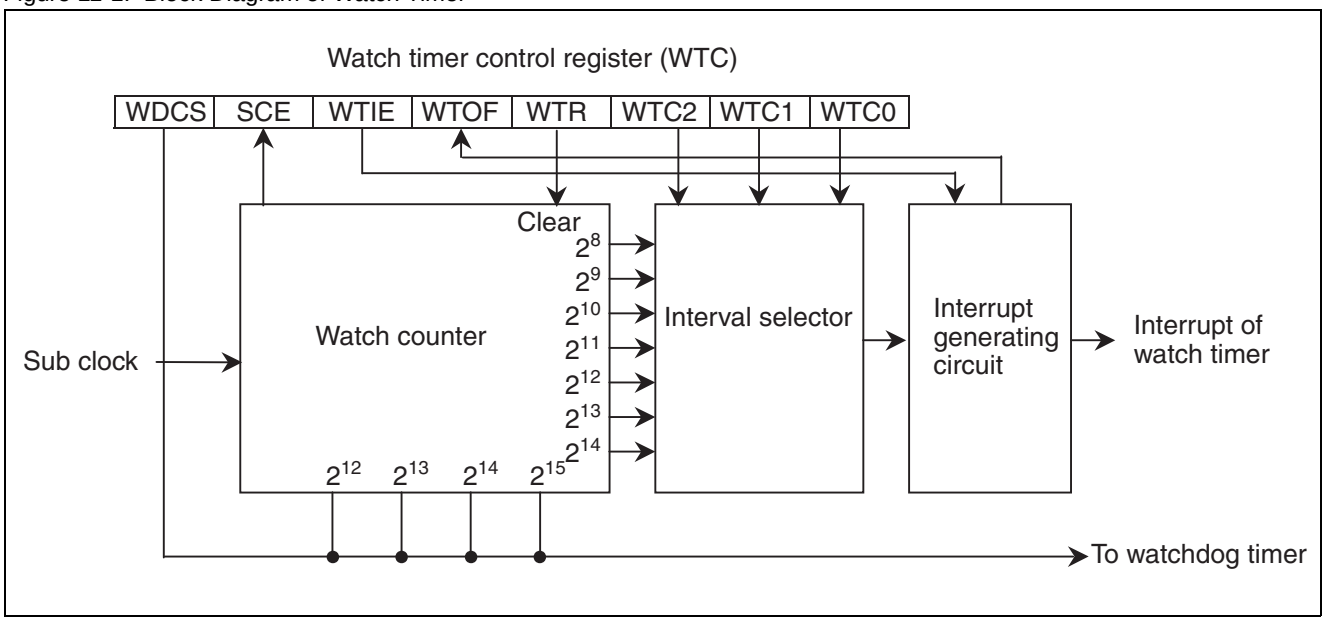
The watch timer is composed of four blocks that include the following:

- Watch counter
- Interval selector
- Watch timer interrupt generating circuit
- Watch timer control register (WTC)

Block Diagram of Watch Timer

Figure 11-1 is a block diagram of the watch timer.

Figure 11-1. Block Diagram of Watch Timer



Watch counter

This is a 15-bit up-counter that uses the sub clock as the clock source.

Interval selector

This selector selects one of seven types for watch timer interrupt intervals.

Interrupt generating circuit

This circuit generates the interval interrupts of the watch timer.

Watch timer control register (WTC)

This register controls operation of the watch timer and watch timer interrupt, and it specifies the clock source for the watchdog timer.

11.3 Watch Timer Control Register (WTC)

The watch timer control register (WTC) controls operation of the watch timer. This register also controls the time of interval interrupts.

Configuration of Watch Timer Control Register (WTC)

Figure 11-2 shows the configuration of the watch timer control register (WTC), and Table 11-2 lists the functions of bits in the watch timer control register (WTC).

Figure 11-2. Configuration of Watch Timer Control Register (WTC)

Address	bit15 ····· bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
$0000AA_H$		WDCS	SCE	WTIE	WTOF	WTR	WTC2	WTC1	WTC0	10001000_B
		R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	

WTC2	WTC1	WTC0	Watch timer interval selection bits Interval time (sub clock 32 kHz)
0	0	0	31.25 ms
0	0	1	62.5 ms
0	1	0	125 ms
0	1	1	250 ms
1	0	0	500 ms
1	0	1	1 s
1	1	0	2 s
1	1	1	Setting is prohibited

WTR	Watch counter clear bit
0	All bits of the watch timer counter are cleared to "0".
1	Nothing happens. This bit is always read during reading.

WTOF	Watch timer interrupt request flag bit
0	No interrupt request is generated.
1	An interrupt request is generated.

WTIE	Watch timer interval interrupt permit bit
0	Interrupt prohibited.
1	Interrupt permitted.

SCE	Bit indicating end of waiting time to stable oscillation of sub clock
0	The waiting time to stable oscillation is ongoing.
1	The waiting time to stable oscillation has ended.

WDCS	Watchdog timer clock source selection bit
0	Select the clock for the watch timer
1	Select the clock for the time-base timer.

R/W : Readable/Writable
R : Read only
: Initial value

Table 11-2. Functions of Bits in Watch Timer Control Register (WTC)

Bit name		Function
bit7	WDCS: Watchdog timer clock source selection bit	This bit is for selecting the clock source for the watchdog timer. ■ If this bit is set to “0”, it specifies clock for the watch timer; if this bit is set to "1", it specifies clock for the time-base timer. If the mode transits to the sub clock mode with setting to "1", the watchdog timer stops. ■ This bit is initialized to “1” by a reset.
bit6	SCE: Bit indicating end of oscillation stabilization wait time for sub clock	This bit indicates that the oscillation stabilization wait time of the sub clock has ended. ■ If this bit is set to “0”, it indicates that the oscillation stabilization wait time of the sub clock is ongoing. ■ The oscillation stabilization wait time of the sub clock is fixed at 2^{14} cycles of the sub clock. ■ This bit is initialized to “0” by a power-on reset or stop.
bit5	WTIE: Watch timer interval interrupt permit bit	This bit is for permitting interval interrupts by the watch timer. ■ If this bit is set to “1”, it permits interrupts; if this bit is set to “0”, it prohibits interrupts. ■ This bit is initialized to “0” by a reset.
bit4	WTOF: Watch timer interrupt request flag bit	This bit indicates that an interrupt request by the watch timer has been issued. ■ If this bit is set to “1” and the WTIE bit is set to “1”, an interrupt request has been issued. ■ This bit is set to “1” at each interval time specified by the WTC2 to WTC0 bits. ■ This bit is cleared to “0” by writing “0”, a transition to the stop mode, and a reset. ■ Writing “1” to this bit has no effect.
bit3	WTR: Watch counter clear bit	This bit is for clearing all bits to “0” in the watch timer counter. ■ Writing “0” to this bit clears the watch timer counter to “0”. ■ Writing “1” to this bit has no effect. ■ During reading, “1” is always read.
bit2 to bit0	WTC2 to WTC0: Watch timer interval selection bits	These bits specify the interval of the watch timer. ■ These bits are initialized to “000_B” by a reset. ■ When changing the bit settings, clear the WTOF bit at the same time.

11.4 Operation of Watch Timer

The watch timer functions as a watch counter, interval interrupt function of the watch timer, clock source setting function for the watchdog timer and the oscillation stabilization wait time function of the sub clock.

Watch Counter

The watch counter is composed of a 15-bit counter to count the sub clock, and it always continues counting as long as the sub clock is input.

Clearing the watch counter

The operation of clearing the watch counter is affected by a power-on reset, a transition to the stop mode, and writing “0” to the watch counter clear bit (WTR) in the watch timer control register (WTC).

Notes:

- Clearing the watch counter affects the watchdog timer and interval interrupts that use watch timer output.
- To clear the watch timer by writing “0” to the WTR bit in the watch timer control register (WTC), set the WTIE bit to “0” and set the watch timer to interrupt inhibited state. Before permitting an interrupt, clear the interrupt request issued by writing “0” to the WTOF flag.

Interval Interrupt Function of Watch Timer

This function generates interrupts at fixed intervals by using the carry signals of the watch counter.

Specification of the interval time

The interval time can be specified with the WTC2, WTC1, and WTC0 bits in the WTC register.

Generation of watch timer interrupts

The watch timer interrupt request flag bit (WTOF) is set at each interval time specified by the WTC2 to WTC0 bits. Consequently, when interrupts are permitted by setting the watch timer interval interrupt permit bit (WTIE) to "1", a watch timer interrupt occurs.

The timing, when the WTOF bit is set, depends on the timing as a time reference when the watch timer was cleared for the last time.

Because the watch timer is used as the timer for the oscillation stabilization wait time of the sub clock after a transition to the stop mode, the WTOF bit is simultaneously cleared at the transition to this mode.

Clock Source Specifying Function for Watchdog Timer

The clock source for the watchdog timer can be specified with the watchdog timer clock source selection bit (WDCS) of the WTC register. If the sub clock is used as the machine clock, set the WDCS bit to "0" and select the output of the watch timer. If the mode transits to the sub clock mode with the WDCS bit setting to "1", the watchdog timer stops.

Sub Clock Oscillation Stabilization Wait Time Function

For a power-on reset or restoration from the stop mode, the watch timer functions as the timer for the oscillation stabilization wait time of the sub clock. The oscillation stabilization wait time of the sub clock is fixed at 2^{14} cycles of the sub clock.

12. 16-bit I/O Timer

This chapter provides an overview of the 16-bit I/O timer, explains the configuration, configuration and functions of its registers, interrupt and its operation.

12.1 Overview of 16-bit I/O timer

12.2 Configuration of 16-bit I/O timer

12.3 Configuration and Function of 16-bit I/O timer Register

12.4 Interrupt of 16-Bit I/O timer

12.5 Operation of 16-Bit I/O timer

12.1 Overview of 16-bit I/O timer

The 16-bit I/O timer consists of one free-run timer, six output compares, and two input captures. An output of six independent waveforms based on the free-run timer can be obtained, and measurement of input pulse widths and external clock intervals are enabled.

Functions of 16-Bit I/O Timer

The 16-bit I/O timer consists of a free-run timer, output compare, and input capture, whose functions are explained below.

Free-run timer (× 1)

Free-run timer consists of 16-bit up-counter, control register, and prescaler.
The output value of the free-run timer uses as a base time (base timer) of input capture and output compare.

- Clock for the count operation can be selected from 8 types of clocks.
- Counter overflow interrupt can be generated.
- The counter initialization is allowed by the match between the value of compare clear register in output compare and value of free-run timer.

Output compare (× 6)

Output compare consists of six 16-bit compare registers, a compare output latch, and a control register. If a free-run timer and compare register have matching values, the output level is reversed with a generation of an interrupt.

- Six compare registers can operate independently. Each compare register has a corresponding output pin and interrupt flag.
- Two compare registers are used as a pair to control the output pin.
- The initial value of output pin can be set.

Input capture (× 2)

Input capture consists of two independent external input pins and corresponding capture registers, control registers, and edge detection circuit. If any edge of a signal input from the external input pin is detected, the free-run timer value is retained in the capture register and, at the same time, an interrupt is generated.

- The edge of an external input signal can be selected from its rising edge, falling edge or both edges.
- The two input capture operate independently.

Interrupts occur at the valid edge of external input signals. Input capture can start EI^{2}OS by the interrupt.

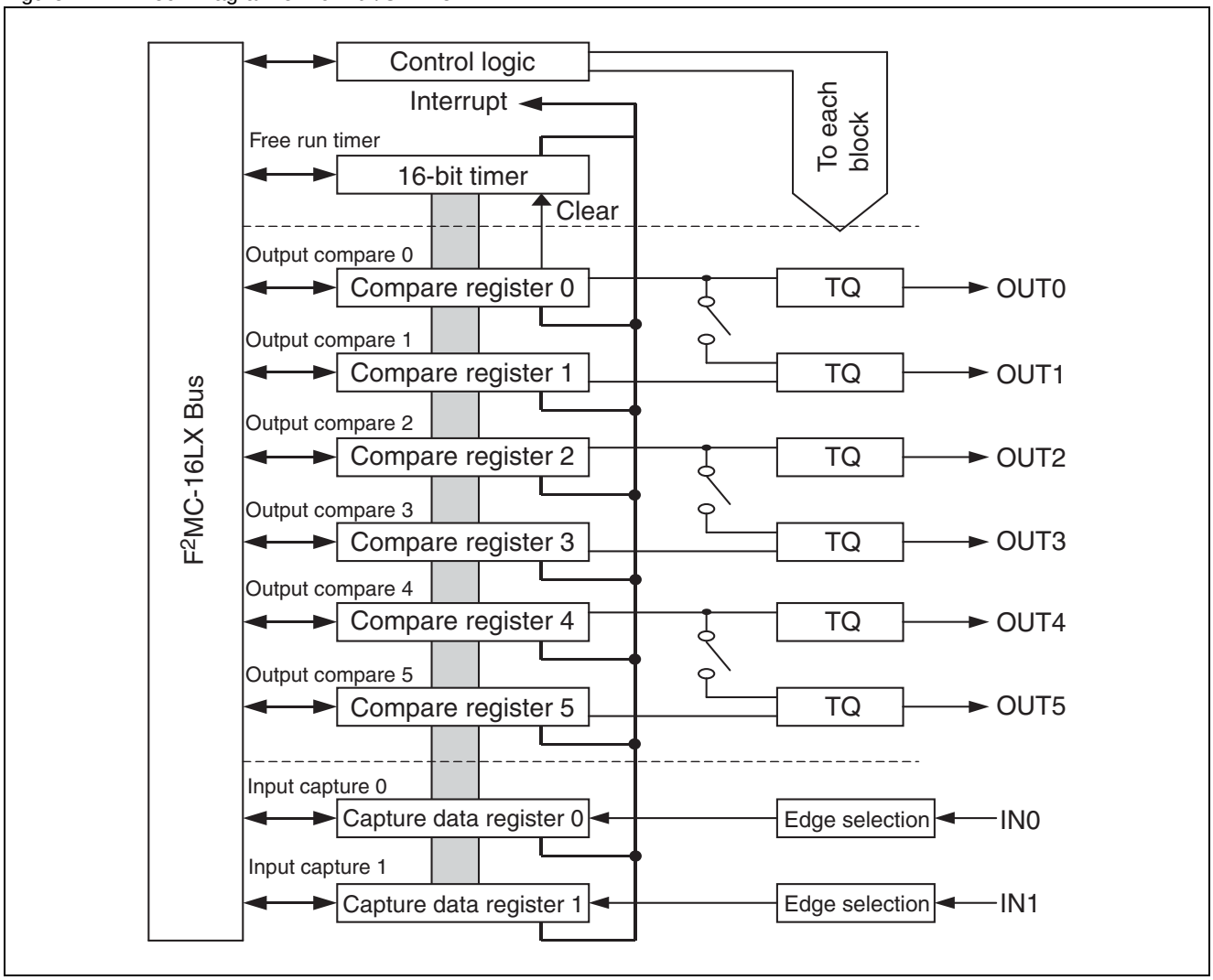
12.2 Configuration of 16-bit I/O timer

The 16-bit I/O timer consists of three modules for the free-run timer, output compare, and input capture.

Block Diagram

Figure 12-1 is a block diagram of the 16-bit I/O timer.

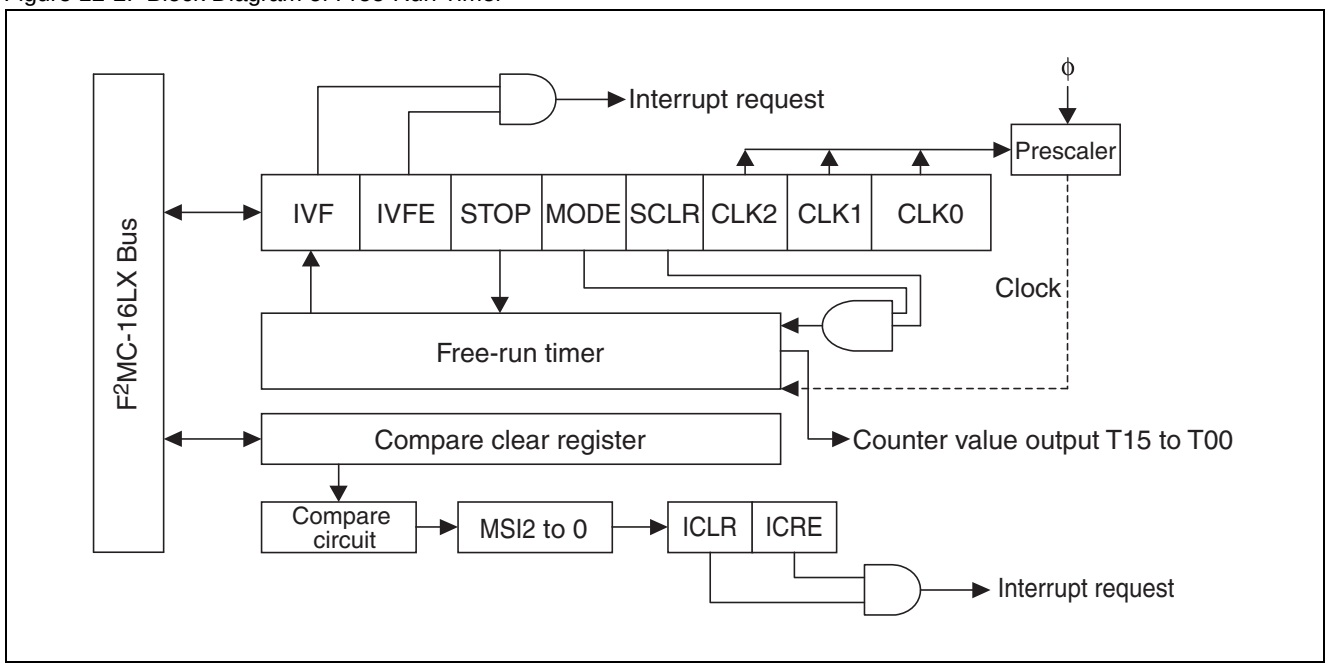
Figure 12-1. Block Diagram of 16-Bit I/O Timer



Block Diagram of Free-Run Timer

Figure 12-2 is a block diagram of the free-run timer.

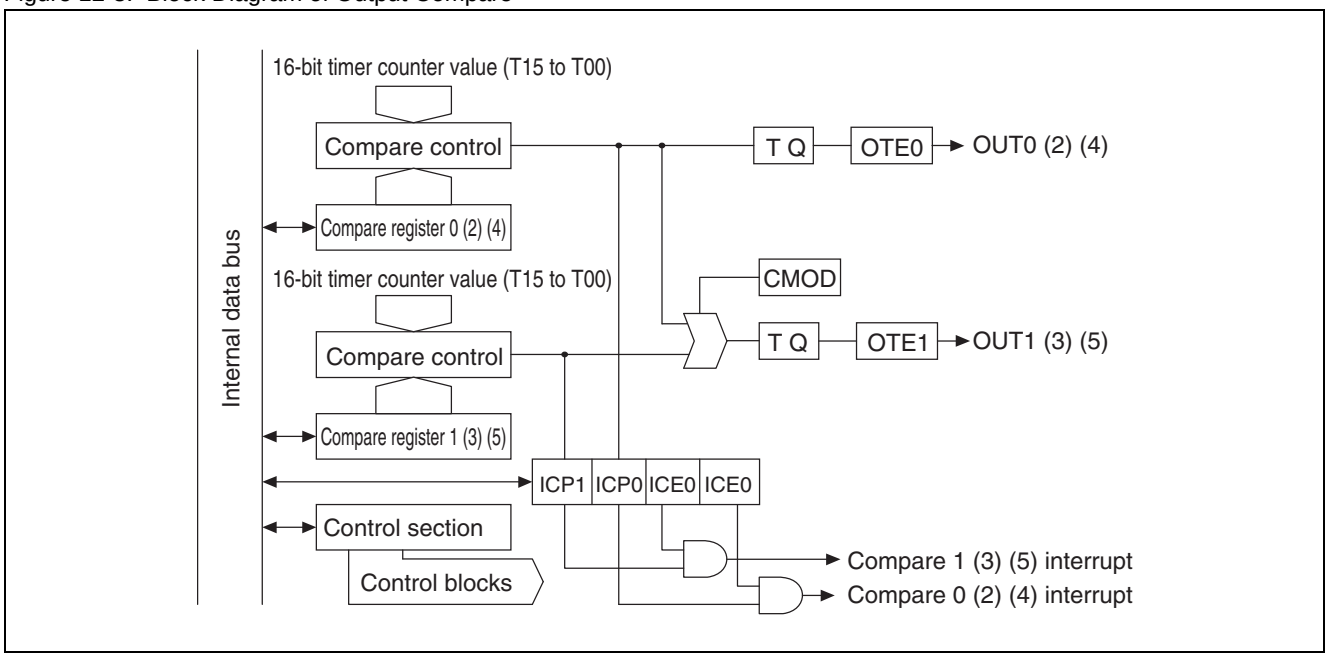
Figure 12-2. Block Diagram of Free-Run Timer



Block Diagram of Output Compare

Figure 12-3 is a block diagram of output compare.

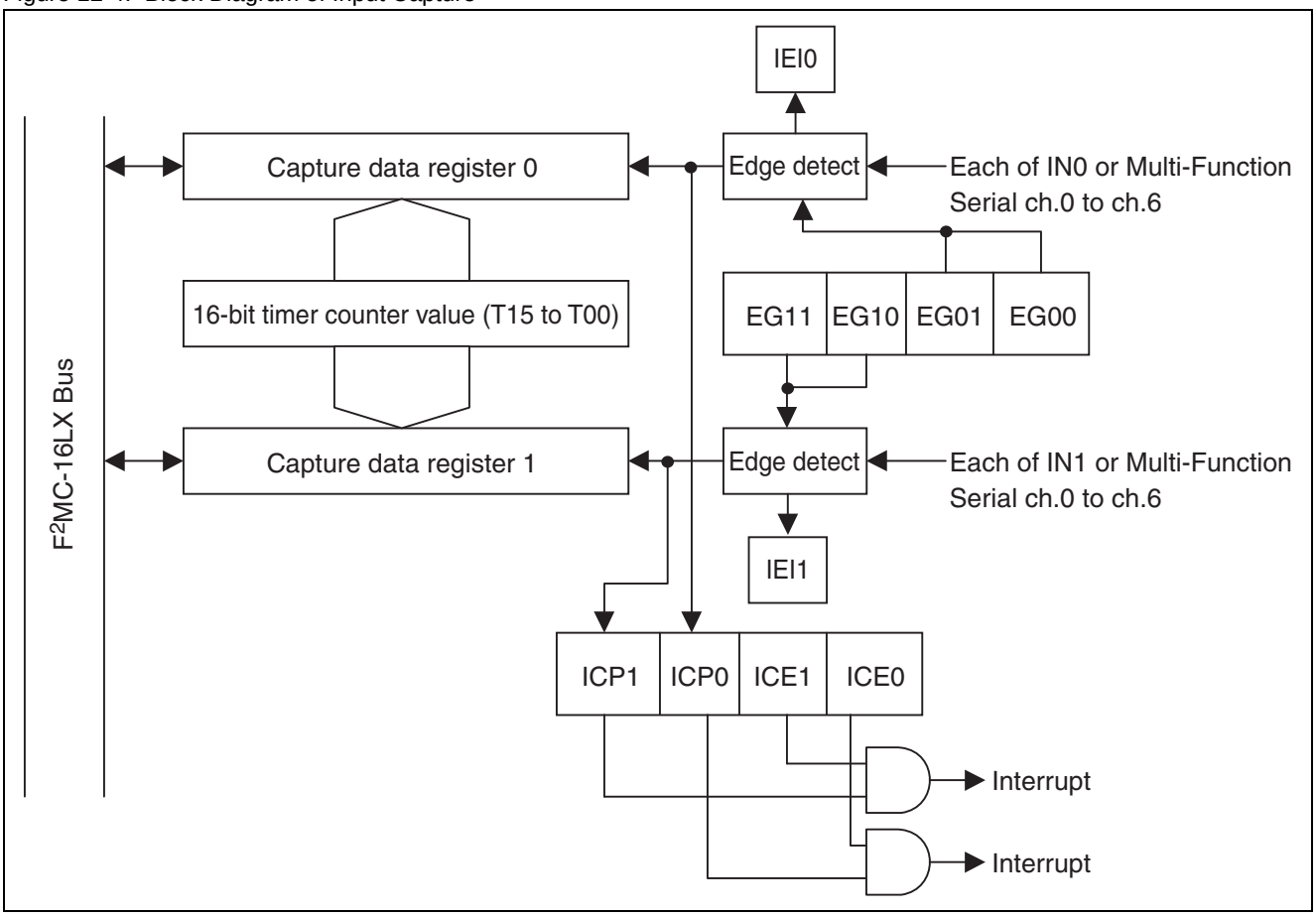
Figure 12-3. Block Diagram of Output Compare



Block Diagram of Input Capture

Figure 12-4 is a block diagram of input capture.

Figure 12-4. Block Diagram of Input Capture



Pin Related to 16-Bit I/O Timer

The pin related to the 16-bit I/O timer has the IN0/IN1 and OUT0/OUT1/OUT2/OUT3/OUT4/OUT5 pins. The IN0/IN1 pins function as the general-purpose I/O port (P16/IN0, P17/IN1) and input capture input pin. The OUT0/OUT1/OUT2/OUT3/OUT4/OUT5 pins function as the general-purpose I/O port (P10/OUT0, P11/OUT1, P12/OUT2, P13/OUT3, P14/OUT4, P15/OUT5) and output compare output pin.

Setting when using as IN0/IN1 pins

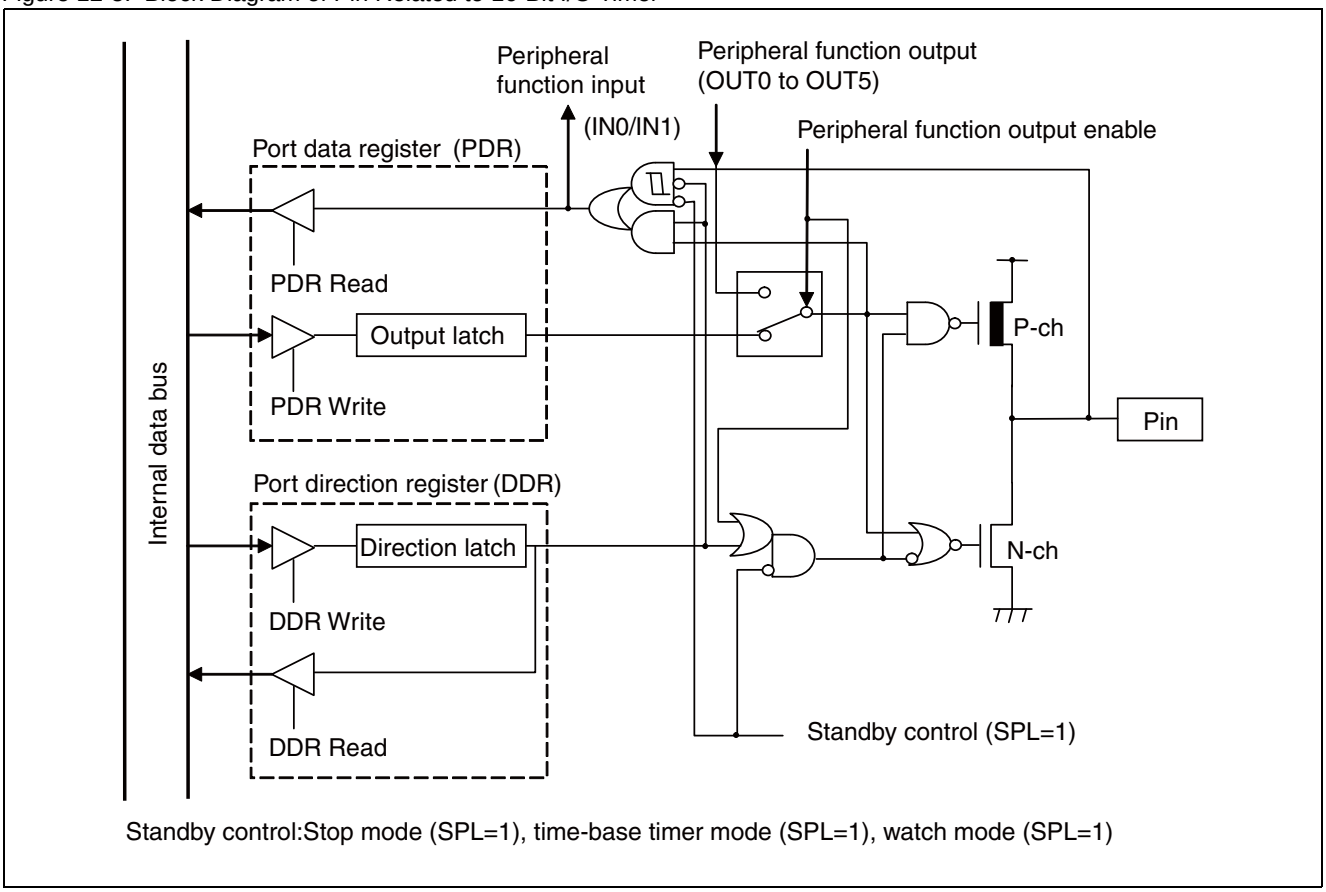
When using as the IN0/IN1 pins, the P16/IN0 and P17/IN1 pins should be set the port direction register to input port (DDR1 bit15, bit14→“0”).

Setting when using as OUT0/OUT1/OUT2/OUT3/OUT4/OUT5 pins

When using the OUT0/OUT1/OUT2/OUT3/OUT4/OUT5 pins as output, be sure to set the control register (OCS01/OCS23/OCS45) to the output compare pin output (OCS01/OCS23/OCS45 bit10, bit11→“1”).

Block Diagram of Pin Related to 16-Bit I/O Timer

Figure 12-5. Block Diagram of Pin Related to 16-Bit I/O Timer

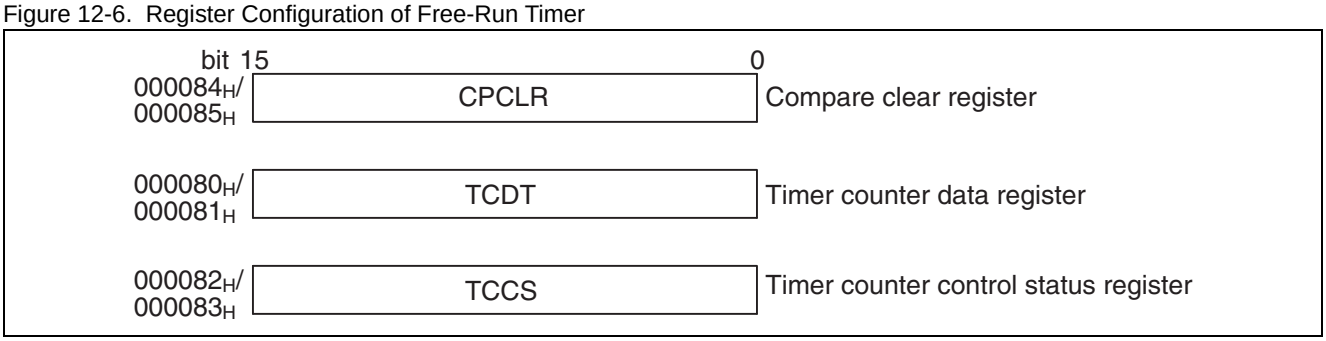


12.3 Configuration and Function of 16-bit I/O timer Register

This section shows the configuration and functions of 16-bit I/O timer registers.

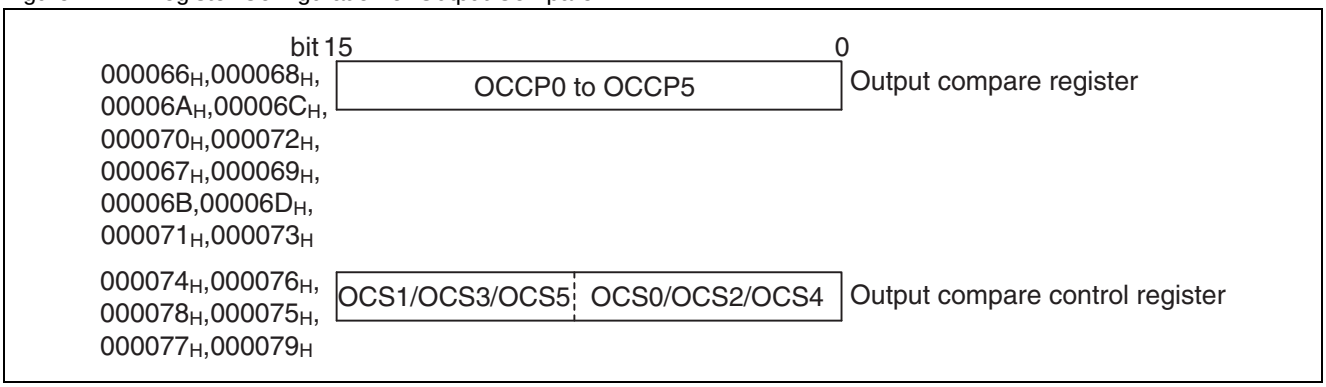
Free-run timer

Figure 12-6. Register Configuration of Free-Run Timer



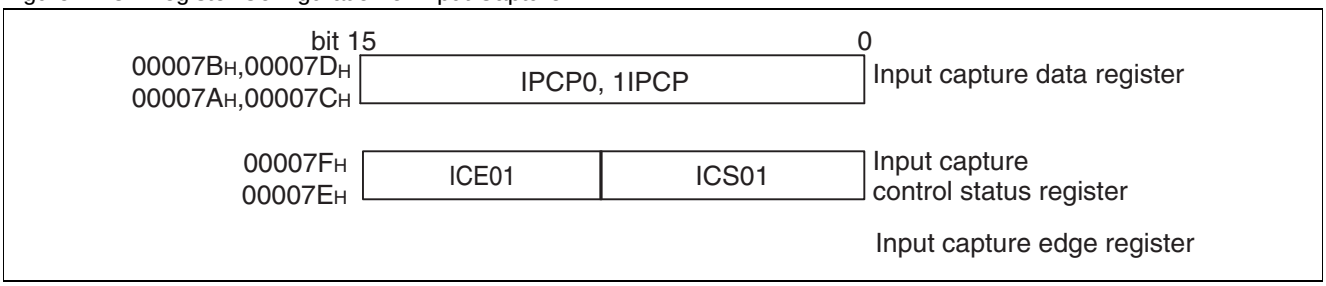
Output compare

Figure 12-7. Register Configuration of Output Compare



Input capture

Figure 12-8. Register Configuration of Input Capture



12.3.1 Free-Run Timer

This section explains the configuration and functions of free-run timer registers.

List of Free-Run Timer Registers

Figure 12-9 shows a list of the free-run timer registers.

Figure 12-9. List of Free-Run Timer Registers

	bit	15	14	13	12	11	10	9	8	CPCLR
000085_H		CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	Compare clear register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	7	6	5	4	3	2	1	0	CPCLR
000084_H		CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	Compare clear register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	15	14	13	12	11	10	9	8	TCDT
000081_H		T15	T14	T13	T12	T11	T10	T09	T08	Timer counter data register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	7	6	5	4	3	2	1	0	TCDT
000080_H		T07	T06	T05	T04	T03	T02	T01	T00	Timer counter data register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	15	14	13	12	11	10	9	8	TCCS
000083_H		Reserved	Reserved	—	MSI2	MSI1	MSI0	ICLR	ICRE	Timer counter control/status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 01100000_B
	bit	7	6	5	4	3	2	1	0	TCCS
000082_H		IVF	IVFE	STOP	MODE	SCLR	CLK2	CLK1	CLK0	Timer counter control/status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

Compare Clear Register (CPCLR)

Figure 12-10 shows the bit configuration of the compare clear register (CPCLR).

Figure 12-10. Bit Configuration of the Compare Clear Register (CPCLR)

	bit	15	14	13	12	11	10	9	8	CPCLR
000085_H		CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	Compare clear register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	7	6	5	4	3	2	1	0	CPCLR
000084_H		CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	Compare clear register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

The compare clear register (CPCLR) is a 16-bit length compare register used to make a comparison with the free-run timer. An initial value of a register is undefined. Therefore, set the initial value, then allow the interrupt operation. This register requires word access.

When the MODE bit of the timer counter control status register (TCCS) is set to “1”, the free-run timer value is initialized to “0000_H” at matching this register value and the value of the free-run timer. When this register value matches the value of the free-run timer, a compare clear interrupt flag is set. When the compare clear interrupt flag is set to “1”, an interrupt request issues to the CPU at allowing the interrupt operation.

Timer Counter Data Register (TCDT)

Bit configuration of the timer counter data register (TCDT) is shown below.

Figure 12-11. Bit Configuration of Timer Counter Data Register (TCDT)

	bit	15	14	13	12	11	10	9	8	TCDT
000081_H		T15	T14	T13	T12	T11	T10	T09	T08	Timer counter data register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B
	bit	7	6	5	4	3	2	1	0	TCDT
000080_H		T07	T06	T05	T04	T03	T02	T01	T00	Timer counter data register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

The timer counter data register (TCDT) is a 16-bit up counter and is used to read the value of the free-run timer counter. The counter value is cleared to "0000_H" by a reset. Writing to this register must be performed in the stopped (STOP = 1) state, and the written value defines the timer value.

This register requires word access. The free-run timer is initialized with the following causes:

- Reset
- Clear bit (SCLR) of timer counter control status register (TCCS)
- Matching between the compare clear register (CPCLR) and timer counter value (TCCS: MODE = 1).

Timer Counter Control Status Register (TCCS)

Bit configuration of the timer counter control status register (TCCS) is shown below.

Figure 12-12. Bit Configuration of Timer Counter Control Status Register (TCCS)

	bit	15	14	13	12	11	10	9	8	TCCS
000083_H		Reserved	Reserved	-	MSI2	MSI1	MSI0	ICLR	ICRE	Timer counter control status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 01100000_B
	bit	7	6	5	4	3	2	1	0	TCCS
000082_H		IVF	IVFE	STOP	MODE	SCLR	CLK2	CLK1	CLK0	Timer counter control status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

Timer counter control status register (TCCS) consists of bits that have the functions explained below.

[bit15] Reserved bit

This bit is a reserved bit. Be sure to write "0" to this bit.

[bit14] Reserved bit

This bit is a reserved bit. Be sure to write "1" to this bit.

[bit13] Unused bit

This bit is not used. When read, "1" is always read.

[bit12 to bit10] MSI2 to MSI0

These bits specify the count with which a compare clear interrupt is masked. It consists of 3-bit reload counter that reloads the count value every time the counter value underflow. During writing to this register (including the other bit), the count value is also loaded, where mask count = specified count (e.g., set to "010_B" when masked twice and the third time is interrupted). However, if "000_B" is set, no interrupt cause is masked. When read, the write value is read.

[bit9] ICLR

This bit is the interrupt request flag for compare clear. When a compare watch between the compare clear register and free-run timer value occurs for the number of times set by the interrupt mask selection bits (bit12 to bit10: MSI2 to MSI0), this bit is set to “1”. If the interrupt request permit bit (bit8: ICRE) is set, an interrupt occurs. This bit is cleared by writing “0”. Writing “1” has no effect. The read-modify-write (RMW) instruction always reads “1”. If the setting “1” and the writing “0” are generated simultaneously, the writing “0” is given priority.

0	No interrupt request issued (initial value)
1	Interrupt request issued

[bit8] ICRE

This bit is the interrupt permit bit for compare clear. If this bit is set to “1” and the interrupt flag (bit9: ICLR) is set to “1”, then an interrupt occurs.

0	Interrupt prohibited (initial value)
1	Interrupt permitted

[bit7] IVF

This bit is the interrupt request flag of the free-run timer. If the free-run timer causes an overflow or mode setting results in a match between compare results of the compare clear register and free-run timer so that the counter is cleared, then the IVF bit is set to “1”. If the interrupt request permit bit (bit5: IVFE) is set, an interrupt occurs. This bit is cleared by setting “0”. Setting “1” has no effect. The read-modify-write (RMW) instruction reads “1”. If the setting “1” and the writing “0” are generated simultaneously, the writing “0” is given priority.

0	No interrupt request used (initial value)
1	Interrupt request used

[bit6] IVFE

This bit is the interrupt permit bit for the free-run timer. If this bit is set to “1” when the write flag (bit5: IVF) is set to “1”, an interrupt occurs.

0	Interrupt prohibit (initial value)
1	Interrupt permit

[bit5] STOP

This bit sets whether to enable or disable the counting by the free-run timer. If this bit is set to “1”, the timer stops counting, and if it is set to “0”, the timer starts counting.

0	Count permit (operation) (initial value)
1	Count prohibit (stop)

If the free-run timer stops counting, the output compare operation also stops.

[bit4] MODE

This bit specifies the initialization conditions of the free-run timer.

If set to “0”, the reset and clear bit (bit3: SCLR) initialize the counter value.

If set to “1”, in addition to the reset and clear bit (bit3: SCLR), matching the compare clear register (CPCLR) value with the free-run timer, initializes the counter value.

0	Initialized by reset and clear bit (initial value)
1	Initialized by reset, clear bit, and compare clear register

The counter value initialization occurs at the point where the counter value changes.

[bit3] SCLR

This bit initializes the value of the free-run timer during operation to “0000_H”.

Writing “1” initializes the counter value to “0000_H”. Writing “0” has no effect. The read value is always “0”. The counter value initialization occurs synchronizing with the counter value change point.

0	No effect (initial value)
1	Initializes the counter value to “0000_H”

Avoid writing “1” to the SCLR bit while in timer is stopped.

If it is initialized at the time of stopping the timer, write “0000_H” to the data register.

[bit2 to bit0] CLK2 to CLK0

These bits select the count clock of the free-run timer. Since the clock changes after this bit is written, change the bit setting when output compare and input capture are in the stopped state.

CLK2	CLK1	CLK0	Count clock	ϕ = 20MHz	ϕ = 16MHz	ϕ = 8MHz	ϕ = 4MHz	ϕ = 1MHz
0	0	0	ϕ	50 ns	62.5 ns	0.125 µs	0.25 µs	1.0 µs
0	0	1	ϕ/2	100 ns	0.125 µs	0.25 µs	0.5 µs	2.0 µs
0	1	0	ϕ/4	0.2 µs	0.25 µs	0.5 µs	1.0 µs	4.0 µs
0	1	1	ϕ/8	0.4 µs	0.5 µs	1.0 µs	2.0 µs	8.0 µs
1	0	0	ϕ/16	0.8 µs	1.0 µs	2.0 µs	4.0 µs	16.0 µs
1	0	1	ϕ/32	1.6 µs	2.0 µs	4.0 µs	8.0 µs	32.0 µs
1	1	0	ϕ/64	3.2 µs	4.0 µs	8.0 µs	16.0 µs	64.0 µs
1	1	1	ϕ/128	6.4 µs	8.0 µs	16.0 µs	32.0 µs	128.0 µs

12.3.2 Output Compare

This section explains the configuration and functions of registers for output compare.

List of Output Compare Registers

Figure 12-13 shows a list of the registers for output compare.

Figure 12-13. List of Registers for Output Compare

ch.0 000067_H
ch.1 000069_H
ch.2 $00006B_H$
ch.3 $00006D_H$
ch.4 000071_H
ch.5 000073_H

bit	15	14	13	12	11	10	9	8	OCCP0 to OCCP5
	C15	C14	C13	C12	C11	C10	C09	C08	Output compare register
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

ch.0 000066_H
ch.1 000068_H
ch.2 $00006A_H$
ch.3 $00006C_H$
ch.4 000070_H
ch.5 000072_H

bit	7	6	5	4	3	2	1	0	OCCP0 to OCCP5
	C07	C06	C05	C04	C03	C02	C01	C00	Output compare register
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

ch.0,ch.1 000075_H
ch.2,ch.3 000077_H
ch.4,ch.5 000079_H

bit	15	14	13	12	11	10	9	8	OCS01/OCS23/OCS45
	-	-	-	CMOD	OTE1	OTE0	OTD1	OTD0	Output compare control register
	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 11100000_B

ch.0,ch.1 000074_H
ch.2,ch.3 000076_H
ch.4,ch.5 000078_H

bit	7	6	5	4	3	2	1	0	OCS01/OCS23/OCS45
	ICP1	ICP0	ICE1	ICE0	-	-	CST1	CST0	Output compare control register
	(R/W)	(R/W)	(R/W)	(R/W)	(-)	(-)	(R/W)	(R/W)	Initial value $0000--00_B$

Note:

To rewrite the output compare register, perform within the compare interrupt routine or compare operation disabled state. Be sure not to occur a compare result match and writing the compare register simultaneously.

Output Compare Registers (OCCP0 to OCCP5)

Output compare register (OCCP0 to OCCP5) has the bit configuration shown below.

Figure 12-14. Bit Configuration of Output Compare Registers (OCCP0 to OCCP5)

	bit	15	14	13	12	11	10	9	8	
ch.0 000067_H ch.1 000069_H ch.2 $00006B_H$ ch.3 $00006D_H$ ch.4 000071_H ch.5 000073_H		C15	C14	C13	C12	C11	C10	C09	C08	OCCP0 to OCCP5 Output compare register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

	bit	7	6	5	4	3	2	1	0	
ch.0 000066_H ch.1 000068_H ch.2 $00006A_H$ ch.3 $00006C_H$ ch.4 000070_H ch.5 000072_H		C07	C06	C05	C04	C03	C02	C01	C00	OCCP0 to OCCP5 Output compare register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

The output compare registers (OCCP0 to OCCP5) are a 16-bit length compare register used to make a comparison with the free-run timer. This register is initialized at reset. This register requires word access. If this register and the free-run timer have matching values, a compare signal is generated to set the output compare interrupt flag. If output enable is given, the output level corresponding to the compare register value is reversed and outputted.

Output Compare Control Register (OCS01/OCS23/OCS45)

The output compare control register (OCS01/OCS23/OCS45) has the bit configuration shown below.

Figure 12-15. Bit Configuration of Output Compare Control Register (OCS01/OCS23/OCS45)

	bit	15	14	13	12	11	10	9	8	
ch.0,ch.1 000075_H ch.2,ch.3 000077_H ch.4,ch.5 000079_H		-	-	-	CMOD	OTE1	OTE0	OTD1	OTD0	OCS01/OCS23/OCS45 Output compare control register
		(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 11100000_B

	bit	7	6	5	4	3	2	1	0	
ch.0,ch.1 000074_H ch.2,ch.3 000076_H ch.4,ch.5 000078_H		ICP1	ICP0	ICE1	ICE0	-	-	CST1	CST0	OCS01/OCS23/OCS45 Output compare control register
		(R/W)	(R/W)	(R/W)	(R/W)	(-)	(-)	(R/W)	(R/W)	Initial value $0000--00_B$

The functions of bits in the output compare control register (OCS01/OCS23/OCS45) are shown below.

[bit15 to bit13] Unused bits

When read, “1” is read. Writing has no effect on the operation.

[bit12] CMOD

This bit switches the pin output level reverse operation mode in compare result matching if pin output is permitted (OTE1 = 1 or OTE0 = 1).

- If CMOD = 0 (initial value), the level corresponding to the compare register value is reversed.
  - OUT0/OUT2/OUT4: Level is reversed if a match for compare register 0/2/4 is found
  - OUT1/OUT3/OUT5: Level is reversed if a match for compare register 1/3/5 is found
- If CMOD = 1, compare register 0 (2/4) inverts the output level in the same manner as in cases where CMOD = 0; however, the pin OUT1 (OUT3/OUT5) output level corresponding to compare register 1 (3/5) inverts the output level only when both compare register 0 (2/4) and compare register 1 (3/5) have a match.
  Also, if compare registers 0 (2/4) and 1 (3/5) have the same value, the operation is the same as that for one compare register.
  - OUT0/OUT2/OUT4: Level is reversed if a match for compare register 0/2/4 is found
  - OUT1/OUT3/OUT5: Level is reversed if a match for compare register 0/2/4 or 1/3/5 is found

[bit11, bit10] OTE1, OTE0

These bits permit pin output of output compare. These bits take an initial value of “0”.

0	Operate as general-purpose port (initial value)
1	Operate as pin output of output compare

- OTE1: Corresponds to output compare 1/3/5
- OTE0: Corresponds to output compare 0/2/4

[bit9, bit8] OTD1, OTD0

These bits change the pin output level if pin output of output compare is permitted. The initial value of compare pin output is set to “0”. A write operation must be performed after the compare operation stops. In a read operation, the output value of the output compare pin is read.

0	Compare pin output set to “0” (initial value)
1	Compare pin output set to “1”

- OTD1: Corresponds to output compare 1/3/5
- OTD0: Corresponds to output compare 0/2/4

[bit7, bit6] ICP1, ICP0

These bits are the interrupt flags for output compare. Set them to “1” if the compare register and free-run timer have matching values. If the interrupt request bits (ICE1, ICE0) are set to “enabled”, and these bits are set, an output compare interrupt occurs. These bits are cleared if “0” is written. Writing “1” has no effect. A read-modify-write (RMW) instruction only reads “1”. If the setting “1” and the writing “0” are generated simultaneously, the writing “0” is given priority.

0	No compare result match (initial value)
1	Compare result match

- ICP1: Corresponds to output compare 1/3/5
- ICP0: Corresponds to output compare 0/2/4

[bit5, bit4] ICE1, ICE0

These bits are the interrupt permit bits of output compare. If these bits are set to “1” and an interrupt flags (ICP1, ICP0) are set, an output compare interrupt occurs.

0	Output compare interrupt prohibit (initial value)
1	Output compare interrupt permit

- ICE1: Corresponds to output compare 1/3/5
- ICE0: Corresponds to output compare 0/2/4

[bit3, bit2] Unused bits

When read, “1” is read. Writing has no effect on the operation.

[bit1, bit0] CST1, CST0

These bits permit a matching operation with the compare register and free-run timer.

0	Compare operation prohibit (initial value)
1	Compare operation permit

- CST1: Corresponds to output compare 1/3/5
- CST0: Corresponds to output compare 0/2/4

Before a compare operation is permitted, specify the compare register value.

Note:

Output compare operates in synch with the free-run timer clock. Thus, if the free-run timer stops, the compare operation also stops.

12.3.3 Input Capture

This section describes the configuration and functions of the registers for the input capture.

List of Input Capture Registers

Figure 12-16 shows a list of the input capture registers.

Figure 12-16. List of Input Capture Registers

	bit	15	14	13	12	11	10	9	8	
ch.0 $00007B_H$ ch.1 $00007D_H$		CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08	IPCP0,IPCP1 Input capture data register
		(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	Initial value 00000000_B

	bit	7	6	5	4	3	2	1	0	
ch.0 $00007A_H$ ch.1 $00007C_H$		CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00	IPCP0,IPCP1 Input capture data register
		(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	Initial value 00000000_B

	bit	15	14	13	12	11	10	9	8	
$00007F_H$		-	-	-	-	-	-	IEI1	IEI0	ICE01 Input capture edge register
		-	-	-	-	-	-	(R)	(R)	Initial value 11111100_B

	bit	7	6	5	4	3	2	1	0	
$00007E_H$		ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	ICS01 Input capture control status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000_B

	bit	15	14	13	12	11	10	9	8	
$0079A3_H$		-	ICIN0S2	ICIN0S1	ICIN0S0	-	ICIN1S2	ICIN1S1	ICIN1S0	LSYNS LIN Synch Field Switching Register
		(-)	(R/W)	(R/W)	(R/W)	(-)	(-)	(R/W)	(R/W)	Initial value 10001000_B

Input Capture Data Registers (IPCP0, IPCP1)

The input capture data registers (IPCP0, IPCP1) have the bit configuration shown below.

Figure 12-17. Bit Configuration of Input Capture Data Registers (IPCP0, IPCP1)

	bit	15	14	13	12	11	10	9	8	
ch.0 $00007B_H$ ch.1 $00007D_H$		CP15	CP14	CP13	CP12	CP11	CP10	CP09	CP08	IPCP0,IPCP1 Input capture data register
		(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	Initial value 00000000_B

	bit	7	6	5	4	3	2	1	0	
ch.0 $00007A_H$ ch.1 $00007C_H$		CP07	CP06	CP05	CP04	CP03	CP02	CP01	CP00	IPCP0,IPCP1 Input capture data register
		(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	Initial value 00000000_B

The input capture data registers (IPCP0, IPCP1) are a register containing the free-run timer value when a valid edge of the external pin input waveform is detected.

These registers require word access. Writing to these registers is not permitted.

Input Capture Control Status Register (ICS01)

The input capture control status register (ICS01) has the bit configuration shown below.

Figure 12-18. Bit Configuration of Input Capture Control Status Register (ICS01)

	bit	7	6	5	4	3	2	1	0	ICS01
00007E$_H$		ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	Input capture control status register
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Initial value 00000000$_B$

The input capture control status register (ICS01) consists of bits that have the functions explained below.

[bit7, bit6] ICP1, ICP0

These bits are the interrupt flags of input capture. When a valid edge of the external input pin is detected, these bits are set to “1”. If the interrupt permit bits (ICE1, ICE0) are set, an interrupt may occur when a valid edge is detected.

Writing “0” clears these bits. Writing “1” has no effect. The read-modify-write (RMW) instruction always reads “1”. If the setting “1” and the writing “0” are generated simultaneously, the writing “0” is given priority.

0	No valid edge detected (initial value)
1	Valid edge detected

- ICP1: Corresponds to input capture 1
- ICP0: Corresponds to input capture 0

[bit5, bit4] ICE1, ICE0

These bits are used as the interrupt permit bits of input capture. If these bits are set to “1” and the interrupt flags (ICP1, ICP0) are set, then an input capture interrupt occurs.

0	Interrupt prohibit (initial value)
1	Interrupt permit

- ICE1: Corresponds to input capture 1
- ICE0: Corresponds to input capture 0

[bit3 to bit0] EG11, EG10, EG01, EG00

These bits specify the valid edge polarity of external input. Also, they are used to specify the enable of input capture operations.

EG11/EG01	EG10/EG00	Edge detection polarity
0	0	No edge detected (stop state) (initial value)
0	1	Rising edge detected
1	0	Falling edge detected
1	1	Both edges detected

- EG11/EG10: Corresponds to input capture 1
- EG01/EG00: Corresponds to input capture 0

Note:

When the input capture is used, set port direction register (DDRx) that shared the general-purpose I/O port to input port.

Input Capture Edge Register (ICE01)

The following figure shows the bit configuration of the input capture edge register (ICE01).

Figure 12-19. Bit Configuration of Input Capture Edge Register (ICE01)

bit	15	14	13	12	11	10	9	8	ICE01
$00007F_H$	-	-	-	-	-	-	IEI1	IEI0	Input capture edge register
	(-)	(-)	(-)	(-)	(-)	(-)	(R)	(R)	Initial value 11111100_B

The following describes the function of each bit in the input capture edge register (ICE01).

[bit15 to bit10] Unused bits

These are unused bits. The read value is “1”. Writing to the bits has no effect on operation.

[bit9, bit8] IEI1 and IEI0

These bits indicate the type of the edge (rising/falling) detected by the input capture function. Bit9 and bit8 are read-only bits.

Note:

The bit value is invalid when capture operation is stopped (in case of ch.1 -> ICS01: EG11 and EG10 = 00_B; in case of ch.0 -> ICS01: EG01 and EG00 = 00_B)

0	Indicates that a falling edge has been detected.
1	Indicates that a rising edge has been detected.

- IEI1: Corresponds to input capture 1.
- IEI0: Corresponds to input capture 0.

LIN Synch Field Switching Register (LSYNS)

The following figure shows the bit configuration of the LIN Synch Field switching register (LSYNS).

Figure 12-20. Bit Configuration of LIN Synch Field Switching Register (LSYNS)

bit	15	14	13	12	11	10	9	8	LSYNS
$0079A3_H$	-	ICIN0S2	ICIN0S1	ICIN0S0	-	ICIN1S2	ICIN1S1	ICIN1S0	LIN Synch Field switching register
	(-)	(R/W)	(R/W)	(R/W)	(-)	(-)	(R/W)	(R/W)	Initial value 10001000_B

This register contains bits that select the input of input capture. The following describes the function of each bit in the LIN Synch Field switching register (LSYNS).

[bit15, bit11] Unused bits

These are unused bits. “1” is always read.

Writing to the bits has no effect on operation.

[bit14 to bit12] ICIN0S2 to ICIN0S0

These bits select the IN0 input of input capture ch.0.

000_B: IN0 pin

001_B: LIN synch field detection signal input for multi-function serial ch.0

010_B: LIN synch field detection signal input for multi-function serial ch.1

011_B: LIN synch field detection signal input for multi-function serial ch.2

100_B: LIN synch field detection signal input for multi-function serial ch.3

101_B: LIN synch field detection signal input for multi-function serial ch.4

110_B: LIN synch field detection signal input for multi-function serial ch.5

111_B: LIN synch field detection signal input for multi-function serial ch.6

[bit10 to bit8] CIN1S2 to ICIN1S0

These bits select the IN1 input of input capture ch.1.

000_B: IN1 pin

001_B: LIN synch field detection signal input for multi-function serial ch.0

010_B: LIN synch field detection signal input for multi-function serial ch.1

011_B: LIN synch field detection signal input for multi-function serial ch.2

100_B: LIN synch field detection signal input for multi-function serial ch.3

101_B: LIN synch field detection signal input for multi-function serial ch.4

110_B: LIN synch field detection signal input for multi-function serial ch.5

111_B: LIN synch field detection signal input for multi-function serial ch.6

Note:

Set this register while counting is stopped (TCCS:STOP=1) and interrupts are disabled (ICS01:ICE1,ICE0=00_B).

12.4 Interrupt of 16-Bit I/O timer

The interrupt of the 16-bit I/O timer occurs when the counter value of the free-run timer overflows, the trigger edge input to the input capture input pin is performed, and a match with the output compare is detected.

The extended intelligent I/O service (EI^{2}OS) can be activated for the interrupt of the input capture and output compare.

Interrupt of 16-Bit I/O Timer

Table 12-1 shows the interrupt control bit and interrupt source of the 16-bit I/O timer.

Table 12-1. Interrupt of 16-Bit I/O Timer

	Interrupt request flag	Interrupt request output enable bit	Interrupt generation factor
Timer counter overflow interrupt	TCCS: IVF (bit7)	TCCS: IVF E(bit6)	16-bit free-run timer overflow or counter clear by compare match when MODE=1
	TCCS: ICLR (bit9)	TCCS: ICRE (bit8)	When a compare match between the16-bit free-run timer and compare clear register occurs for the number of times set by the interrupt mask selection bits (TCCS:MSI2-MSI0 (bit12-bit10)
Input capture interrupt	ICS01: ICP1(bit7) ch.1 ICS01: ICP1(bit6) ch.0	ICS01: ICE1(bit5) ch.1 ICS01: ICE0(bit4) ch.0	Input the valid edge to input capture input pin
Output compare interrupt	OCS01/OCS23/OCS45: ICP1(bit7) ch.1, ch.3, ch.5 OCS01/OCS23/OCS45: ICP0(bit6) ch.0, ch.2, ch.4	OCS01/OCS23/OCS45: ICE1(bit5) ch.1, ch.3, ch.5 OCS01/OCS23/OCS45: ICE0(bit4) ch.0, ch.2, ch.4	Match of the output compare register value and counter value

ICS01: ICP0/ICE0 correspond to input capture pin (IN0).

ICS01: ICP1/ICE1 correspond to input capture pin (IN1).

OCS01/OCS23/OCS45: ICP0/ICE0 correspond to output compare pins (OUT0/OUT2/OUT4).

OCS01/OCS23/OCS45: ICP1/ICE1 correspond to output compare pins (OUT1/OUT3/OUT5).

Timer counter overflow interrupt

When the timer counter overflow interrupt request flag is set

The timer counter overflow generation flag in the timer counter control status register is set when the followings occur (TCCS: IVF=1)

- When an overflow ("FFFF$_H$"→ "0000$_H$") occurs in counting up of the free-run timer
- When the initialization by compare clear register is set to enable (TCCS: MODE=1) and an match between the setting value of the free-run timer and the value of the compare clear register occurs.
- When a compare match between the 16-bit free-run timer and compare clear register occurs for the number of times set by the interrupt mask selection bits (TCCS: MSI2 to MSI0)

When the timer counter overflow interrupt request occurs

If the timer counter overflow interrupt request is set to enable (TCCS: IVFE=1), the interrupt request is generated when the timer counter overflow generation flag is set to 1 (TCCS: IVF=1).

Input capture interrupt

When the valid edge (ICS: EG) set by the input capture pin is detected the interrupt operation is shown as follows:

- The count value of the free-run timer upon detection is stored in the input capture register.
- The valid edge detection flag in the control status register is set to 1 (ICS: ICP=1).
- When the input capture interrupt request output is set to enable (ICS: ICE=1), the interrupt request occurs.

Output compare interrupt

When a match between the count value of the free-run timer and the setting value of the compare register is detected the interrupt operation is shown as follows:

- The output compare match flag in the control register is set to 1 (OCS:IOP=1).
- When the output compare interrupt request is set to enable (OCS:IOE=1), the interrupt request occurs.

Interrupt of 16-Bit I/O Timer, DMA Transfer, and EI^{2}OS

Table 12-2 shows the relationship between the interrupt source, interrupt vector, and interrupt control register other than software interrupt.

Table 12-2. Interrupt Source, Interrupt Vector, and Interrupt Control Register

Interrupt source	EI^{2}OS clear	µDMAC channel number	Interrupt vector		Interrupt control register	
			Number	Address	Number	Address
Input capture (ch.0/ch.1) fetch*	❍	–	#24	FFFF9C$_H$	ICR06	0000B6$_H$
Output compare (ch.0/ch.1/ch.2) match	❍	–	#25	FFFF98$_H$	ICR07	0000B7$_H$
Output compare (ch.3/ch.4/ch.5) match	❍	–	#26	FFFF94$_H$		
16-bit free-run timer overflow/Compare clear	❍	9	#28	FFFF8C$_H$	ICR08	0000B8$_H$

❍: Interrupt request flag is cleared.

* : This interrupt source shares the interrupt source and interrupt number of other peripheral function. For details, see Table A-3.

Note:

If there are two interrupt sources in the same interrupt number, resource clears both interrupt request flags. Therefore, when one of two sources uses the EI^{2}OS/µDMAC function, other interrupt function cannot be used. The interrupt request enable bit of the relevant resource is set to “0” to execute the software polling processing.

Correspondence to DMA Transfer and EI^{2}OS Function

Input capture, free-run timer and output compare are supported only by EI^{2}OS function, not by DMA transfer function. When the EI^{2}OS function is used, it is necessary to disable other interrupt that shares the interrupt control register (ICR).

12.5 Operation of 16-Bit I/O timer

This section explains the operation and timing of the 16-bit I/O timer.

Operation and Timing of 16-Bit I/O Timer

The 16-bit I/O timer handles the operation and timing for the following items:

- Operation of free-run timer
- Operation of output compare
- Operation of input capture
- Timing of free-run timer
  - Count timing
  - Clear timing
- Timing of output compare
  - Interrupt timing
  - Change timing of the output pin
- Timing of input capture
  - Capture timing to the input signal

12.5.1 Operation of Free-Run Timer

This section explains the operation and timing of the free-run timer.

Operation of Free-Run Timer

The free-run timer starts counting at a counter value of "0000_H" after clearing reset operation. This counter value is used as a reference time for output compare and input capture.

The count value is cleared under the following conditions:

- Overflow occurs
- Compare match is found with the output compare value 0 (mode setting is required)
- SCLR bit of TCCS register is set to "1"
- TCDT register is set to "0000_H"
- Reset occurs

An interrupt occurs if an overflow is generated or compare results between the counter value of free-run timer and the value of compare register 0 match (a compare results match interrupt requires mode setting).

Figure 12-21 shows the timing chart of the counter cleared because of an overflow. Figure 12-22 shows the timing chart of the counter cleared because of a compare results match.

Figure 12-21. Timing Chart of Counter Cleared Because of Overflow

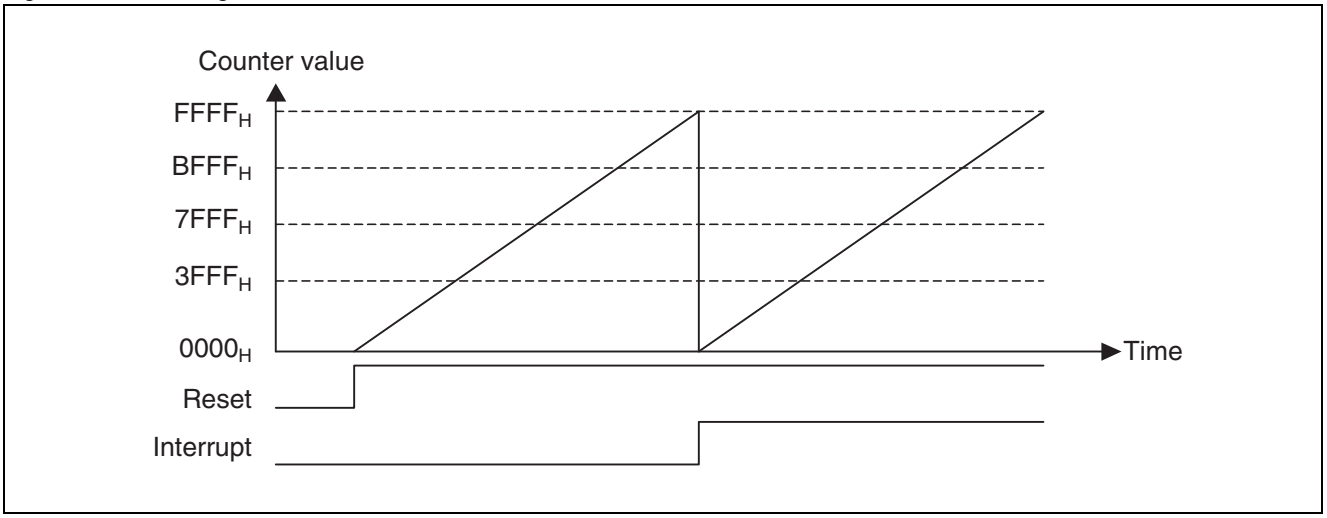
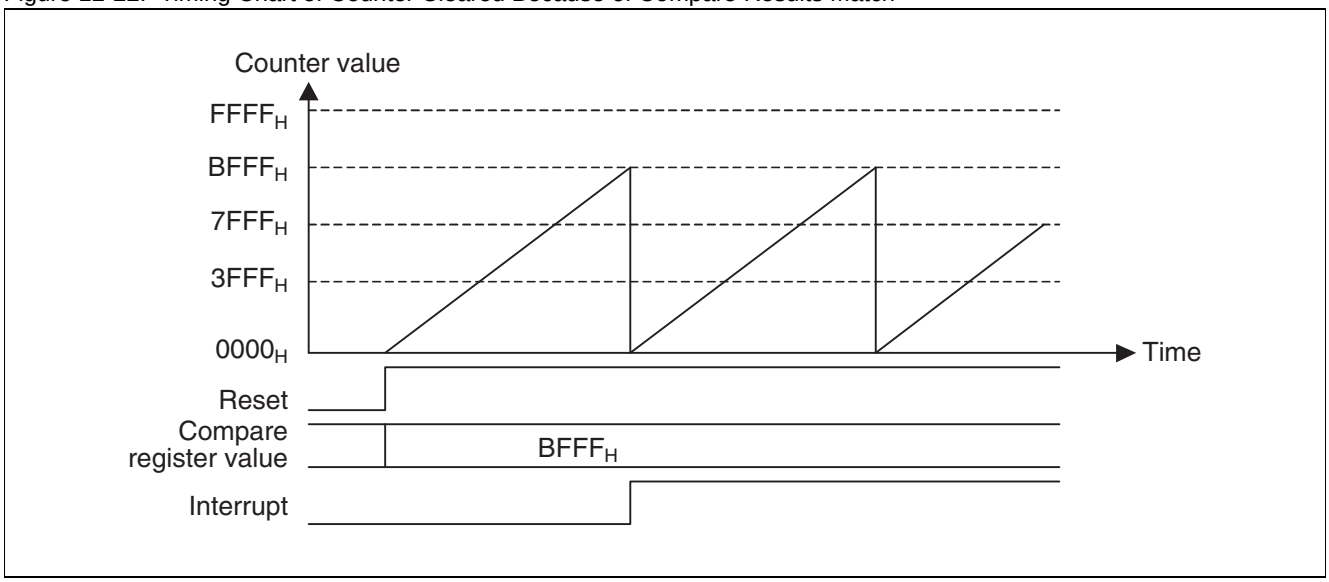


Figure 12-22. Timing Chart of Counter Cleared Because of Compare Results Match



12.5.2 Operation of Output Compare

Output compare compares the specified compare register value with the free-run timer value, and if they match, it issues an interrupt request and reverses the output level.

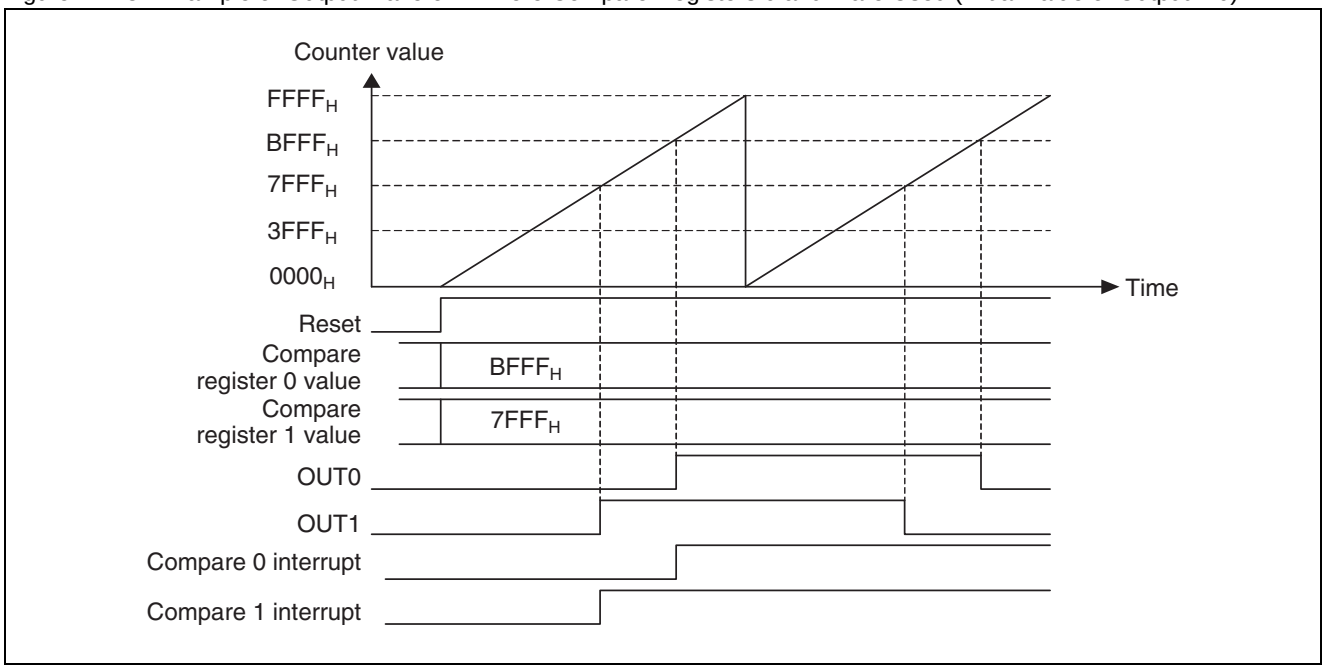
Examples of Output Waveform

Examples of output waveform are shown below.

Example of output waveform where compare registers 0 and 1 are used

Figure 12-23 shows an example of output waveform where the initial value of output is specified as “0”.

Figure 12-23. Example of Output Waveform where Compare Registers 0 and 1 are Used (Initial Value of Output = 0)

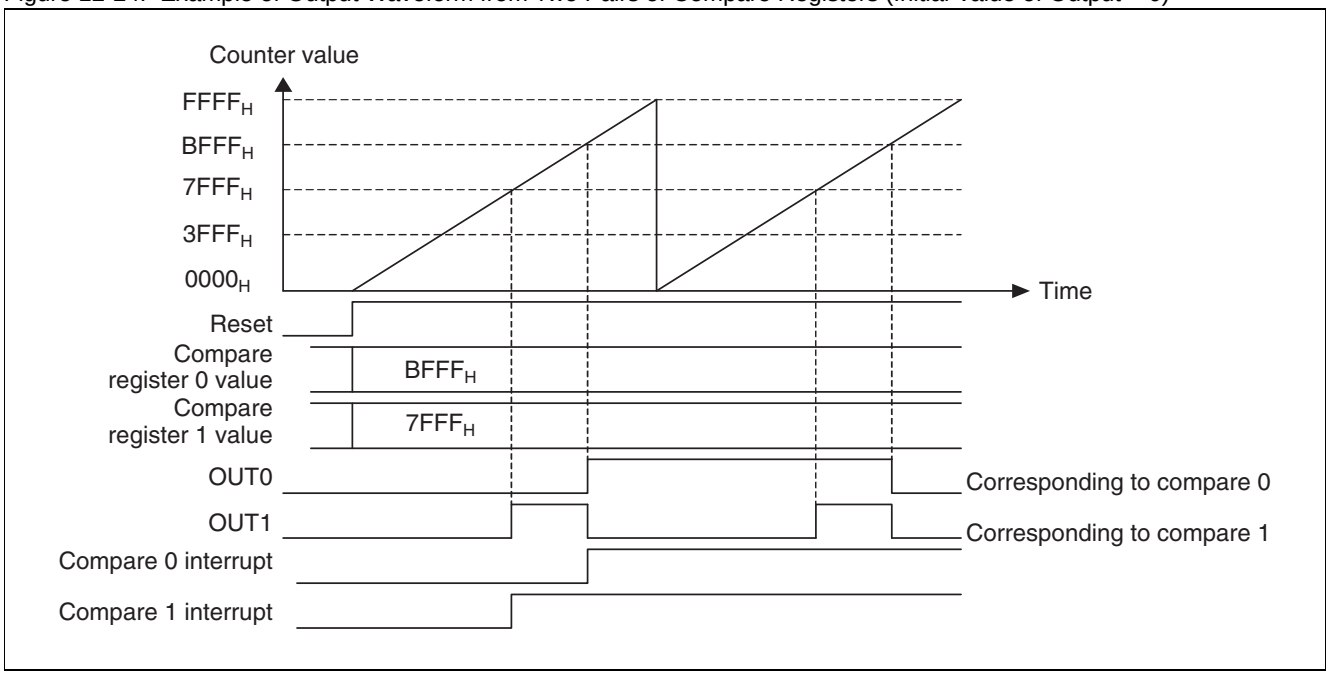


If CMOD = 1, two pairs of compare registers may be used to change the output level.

Example of output waveform from two pairs of compare registers

Figure 12-24 shows an example of output waveform where the initial value of output is specified as “0”.

Figure 12-24. Example of Output Waveform from Two Pairs of Compare Registers (Initial Value of Output = 0)



Note:

To rewrite the compare register, perform within the compare interrupt routine or compare operation disabled state. Be sure not to occur a compare result match and writing the compare register simultaneously.

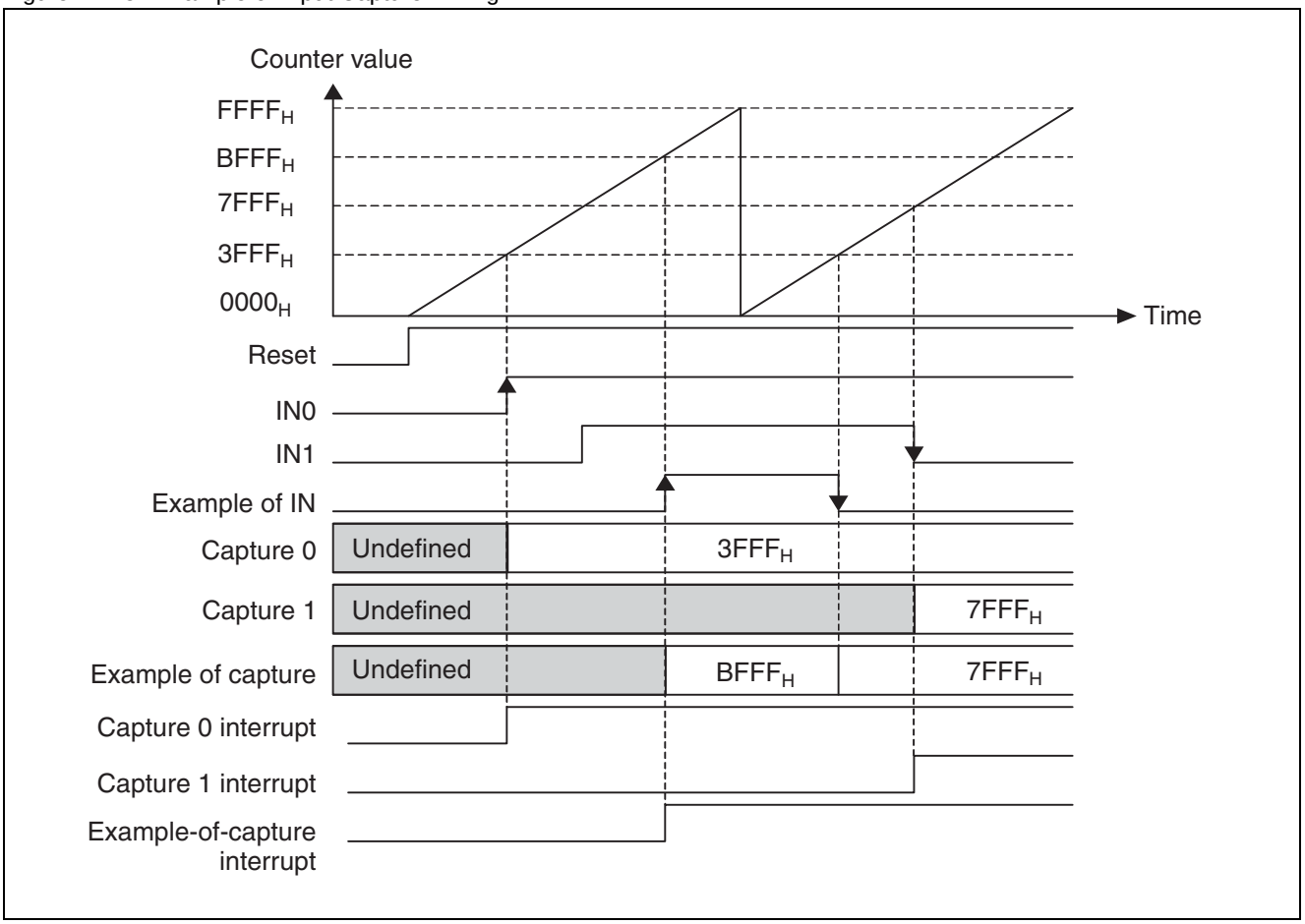
12.5.3 Operation of Input Capture

Input capture generates interrupt request by reading the free-run timer value into the capture register when the specified valid edge is detected.

Example of Input Capture Timing

Figure 12-25 shows an example of input capture timing.

Figure 12-25. Example of Input Capture Timing



12.5.4 Timing of Free-Run Timer

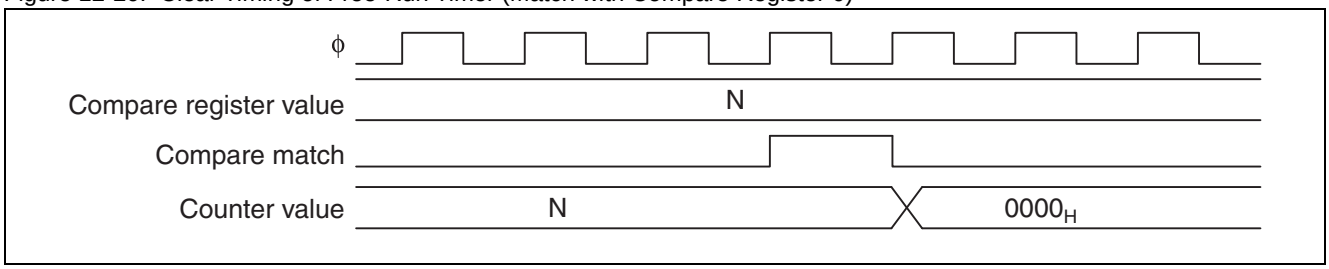
The free-run timer is incremented according to the timing of input clock.

Clear Timing of Free-Run Timer (Match with Compare Register 0)

The counter is cleared by a reset, clearing by software, or a match with compare register 0 and free-run timer. Counter clear by a reset or software occurs when a clear operation is performed. Counter clear because of a match with compare register 0 occurs in synch with count timing.

Figure 12-26 shows the clear timing of the free-run timer caused by a match with compare register 0.

Figure 12-26. Clear Timing of Free-Run Timer (Match with Compare Register 0)



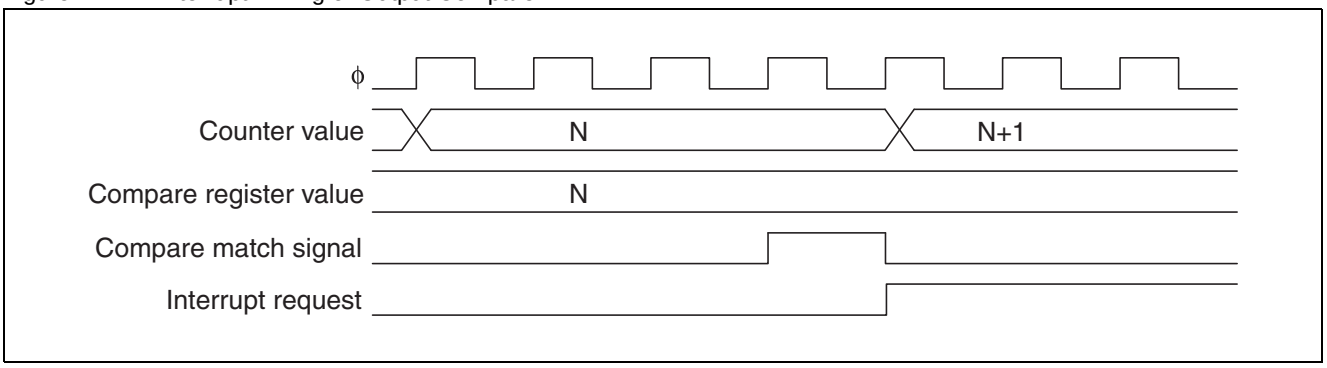
12.5.5 Timing of Output Compare

The output compare is used to issue compare match signals when the free-run timer and the compare register have matching values, to reverse the output value, and to generate interrupt requests. Output reverse timing at the compare match is in synch with the counter timing.

Interrupt Timing

Figure 12-27 shows the interrupt timing of output compare.

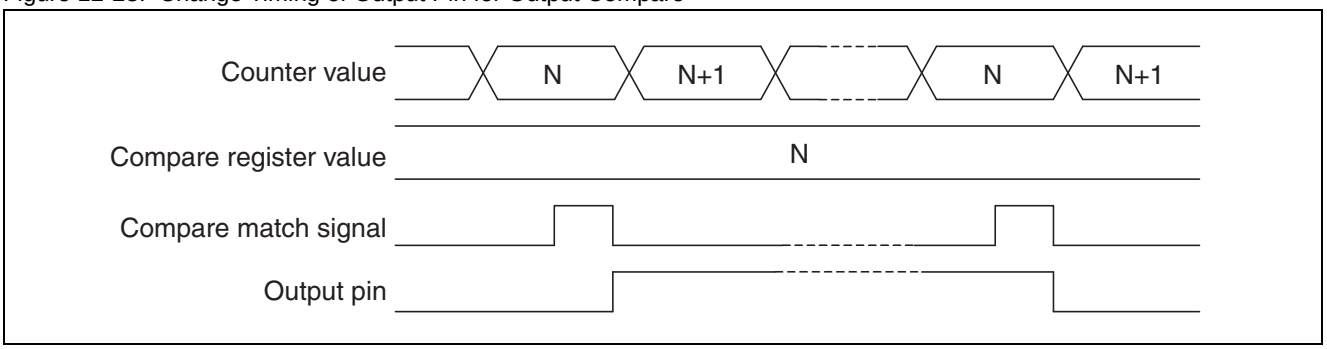
Figure 12-27. Interrupt Timing of Output Compare



Change Timing of Output Pin

Figure 12-28 shows the change timing of the output pin for output compare.

Figure 12-28. Change Timing of Output Pin for Output Compare



Note:

To rewrite the compare register, perform within the compare interrupt routine or compare operation disabled state. Be sure not to occur a compare result match and writing the compare register simultaneously.

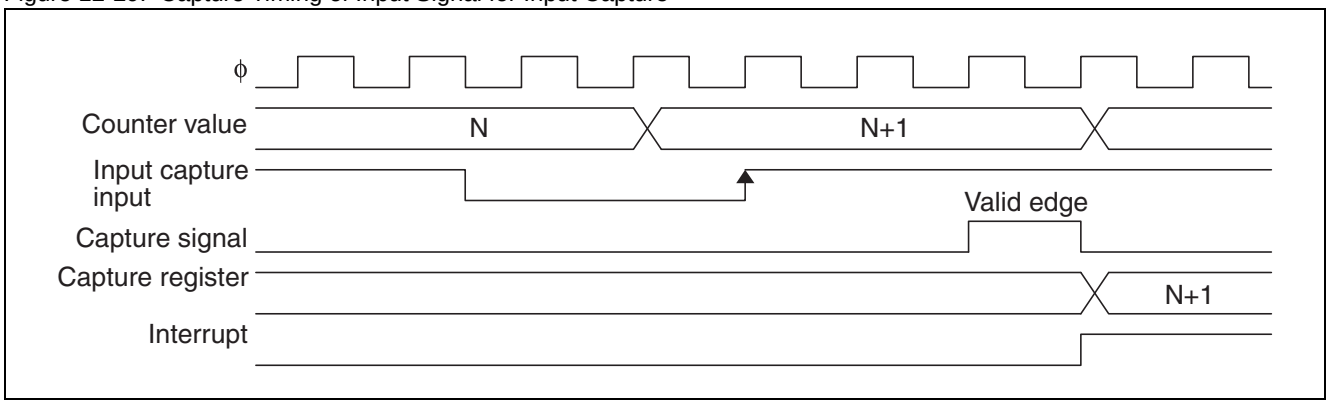
12.5.6 Timing of Input Capture

This section describes a capture timing of the input signal for input capture.

Capture Timing to Input Signal

Figure 12-29 shows the capture timing of input signal for input capture.

Figure 12-29. Capture Timing of Input Signal for Input Capture



13. 8/16-bit Up/Down Counter/Timer

This chapter provides an overview of the 8/16-bit up/down counter/timer, explains the configuration, configuration and functions of its registers, interrupt and its operation.

13.1 Overview of 8/16-Bit Up/Down Counter/Timer

13.2 Configuration of 8/16-Bit Up/Down Counter/Timer

13.3 Configuration and Functions of Registers for 8/16-Bit Up/Down Counter/Timer

13.4 Interrupt of 8/16-Bit Up/Down Counter/Timer

13.5 Operation of 8/16-Bit Up/Down Counter/Timer

13.1 Overview of 8/16-Bit Up/Down Counter/Timer

The 8/16-bit up/down counter/timer consists of six event input pins, two 8-bit up/down counter registers, two 8-bit reload/compare registers, and their control circuits.

Major Functions of 8/16-Bit Up/Down Counter/Timer

- 8-bit count register used for counting in a range of 0 to 255 (in the 16 bits x one operation mode, counting in a range of 0 to 65535 is possible).
- Four types of count modes can be selected for the count clock.
  - Timer mode
  - Up/down count mode
  - Phase difference count mode (multiplied by 2)
  - Phase difference count mode (multiplied by 4)
- In the timer mode, the count clock is selected from two types of internal clocks:
  - Divided by 2 (80ns at internal machine cycle f = 25 MHz)
  - Divided by 4 (320ns at internal machine cycle f = 25 MHz)
- In the up/down count mode, a detection edge of the external pin input signal may be selected:
  - Falling edge detection
  - Rising edge detection
  - Both falling and rising edges detection
  - Edge detection prohibited
- The phase difference count mode is suitable for counting encoder output such as motor, where encoder output from phases A, B, and Z is used as input, thereby facilitating high-precision counting of rotation angle and rotations.
- The ZIN pin is used to select from two types of functions:
  - Counter clear function
  - Gate function
- Compare and reload functions are provided, where each function or a combination of them are available. Starting both functions allows up/down counting with any width.
  - Compare function (an interrupt is issued during compare)
  - Compare function (an interrupt is issued and the counter is cleared during compare)
  - Reload function (an interrupt is issued and reloaded during compare)
  - Compare and reload functions (an interrupt is issued and the counter is cleared during compare, and an interrupt is issued and reloaded when an underflow occurs)
  - Compare and reload prohibited
- Interrupt generation is individually controlled during comparison, a reload (underflow), or an overflow and when the direction of the count changes.
- The count direction flag identifies the count direction of the last counter operation.

13.2 Configuration of 8/16-Bit Up/Down Counter/Timer

The 8-bit up/down counter/timer has two channels and consists of three event input pins, one 8-bit up/down count, and one 8-bit reload/compare register per channel. Also, one of two 8-bit up/down counter/timer channels can be used as the 16-bit up/down counter/timer. (When using as the 16-bit up/down counter/timer, the register of ch.0 is valid.)

Block Diagram of 8/16-Bit Up/Down Counter/Timer

Figure 13-1 and Figure 13-2 are block diagrams of the 8/16-bit up/down counter/timer.

Figure 13-1. Block Diagram of 8/16-Bit Up/Down Counter/Timer (Ch.0)

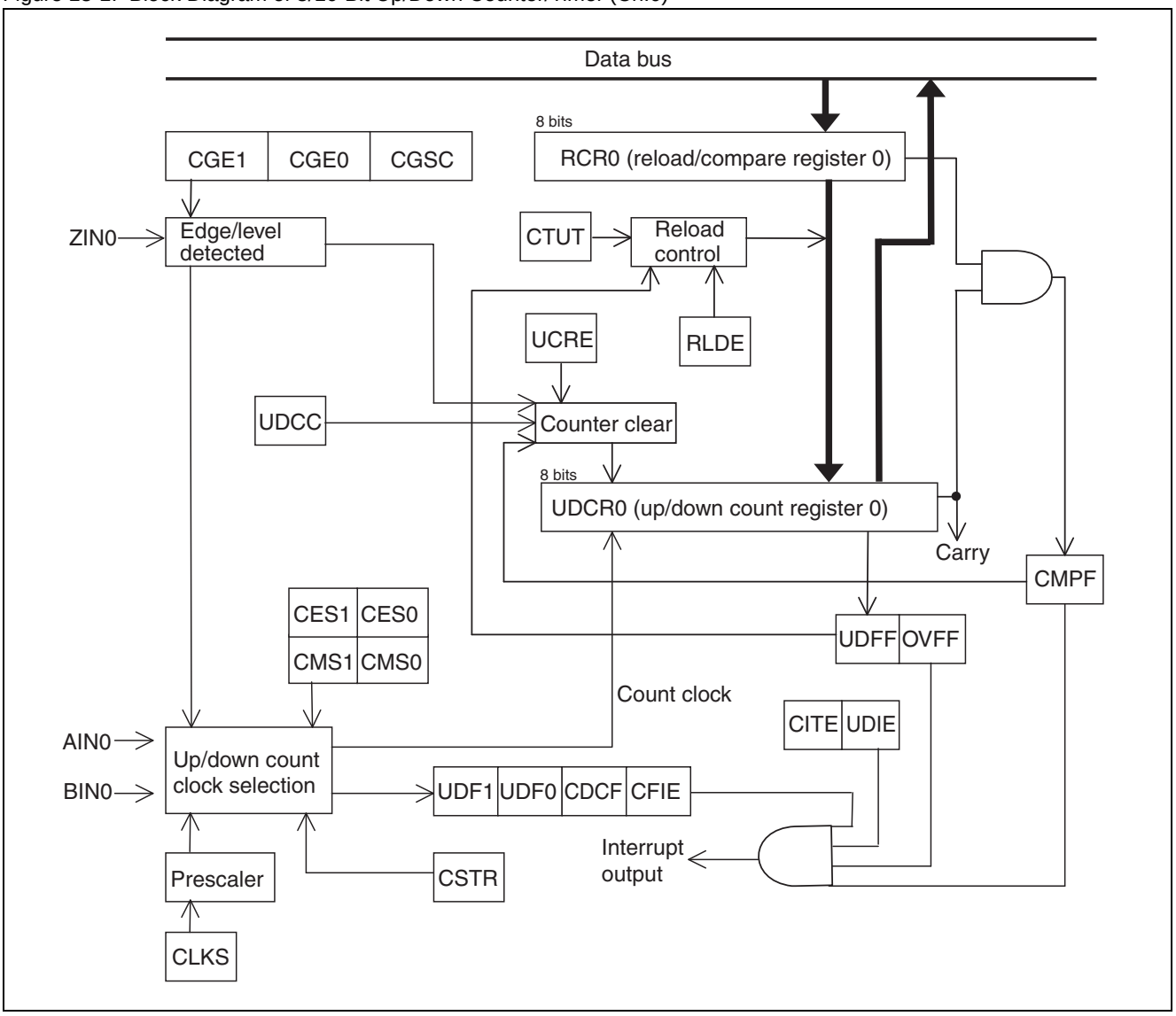
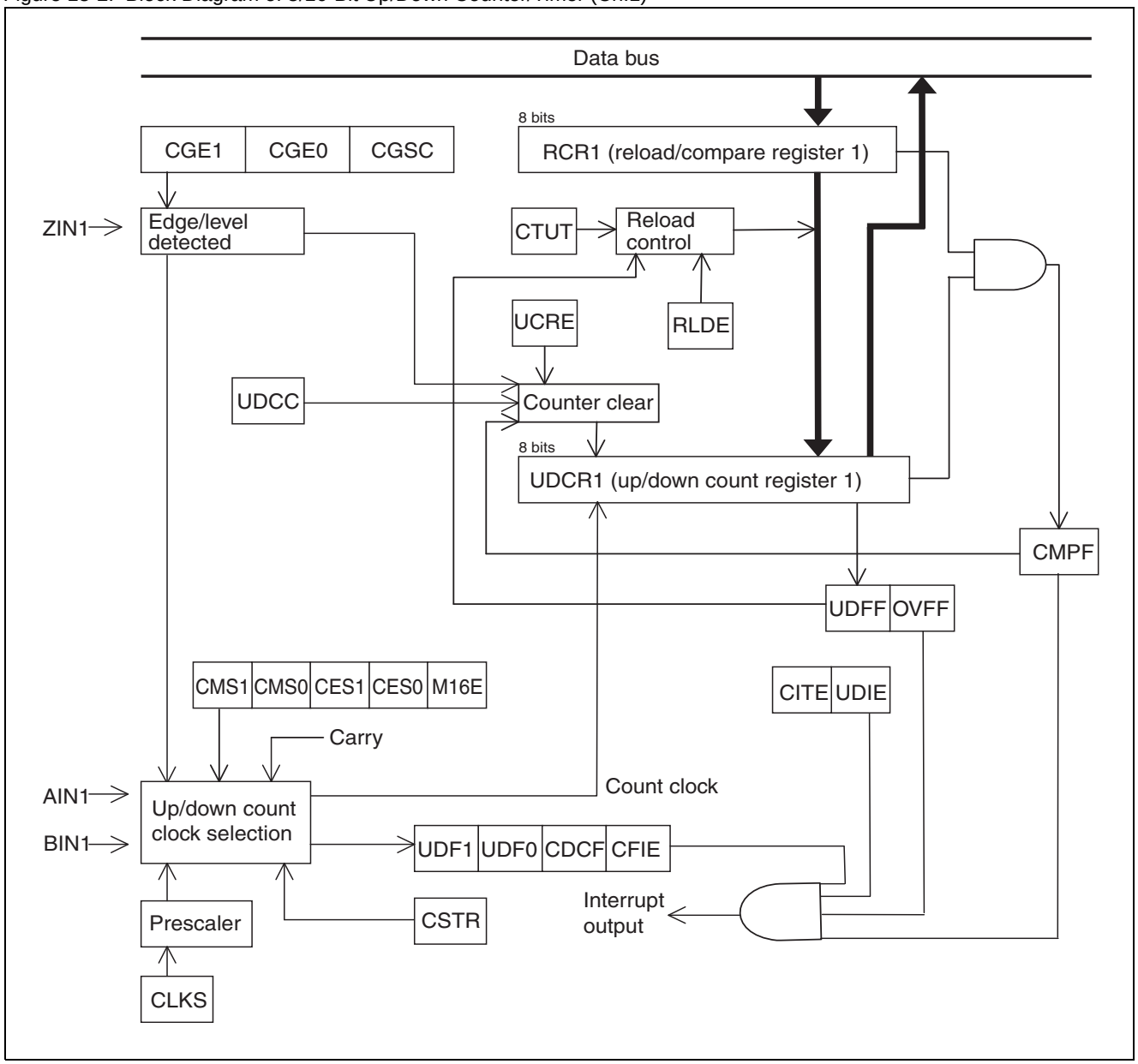


Figure 13-2. Block Diagram of 8/16-Bit Up/Down Counter/Timer (Ch.1)



Pin Related to 8/16-Bit Up/Down Counter/Timer

The pin related to the 8/16-bit up/down counter/timer has the AIN0/BIN0/ZIN0 and AIN1/BIN1/ZIN1 pins. AIN0/BIN0/ZIN0 is sharing a function as the general-purpose I/O port (AIN0/UO1/P31, BIN0/UCK1/P32, ZIN0/UI1/P30) and the input/output pin of the multi-function serial.

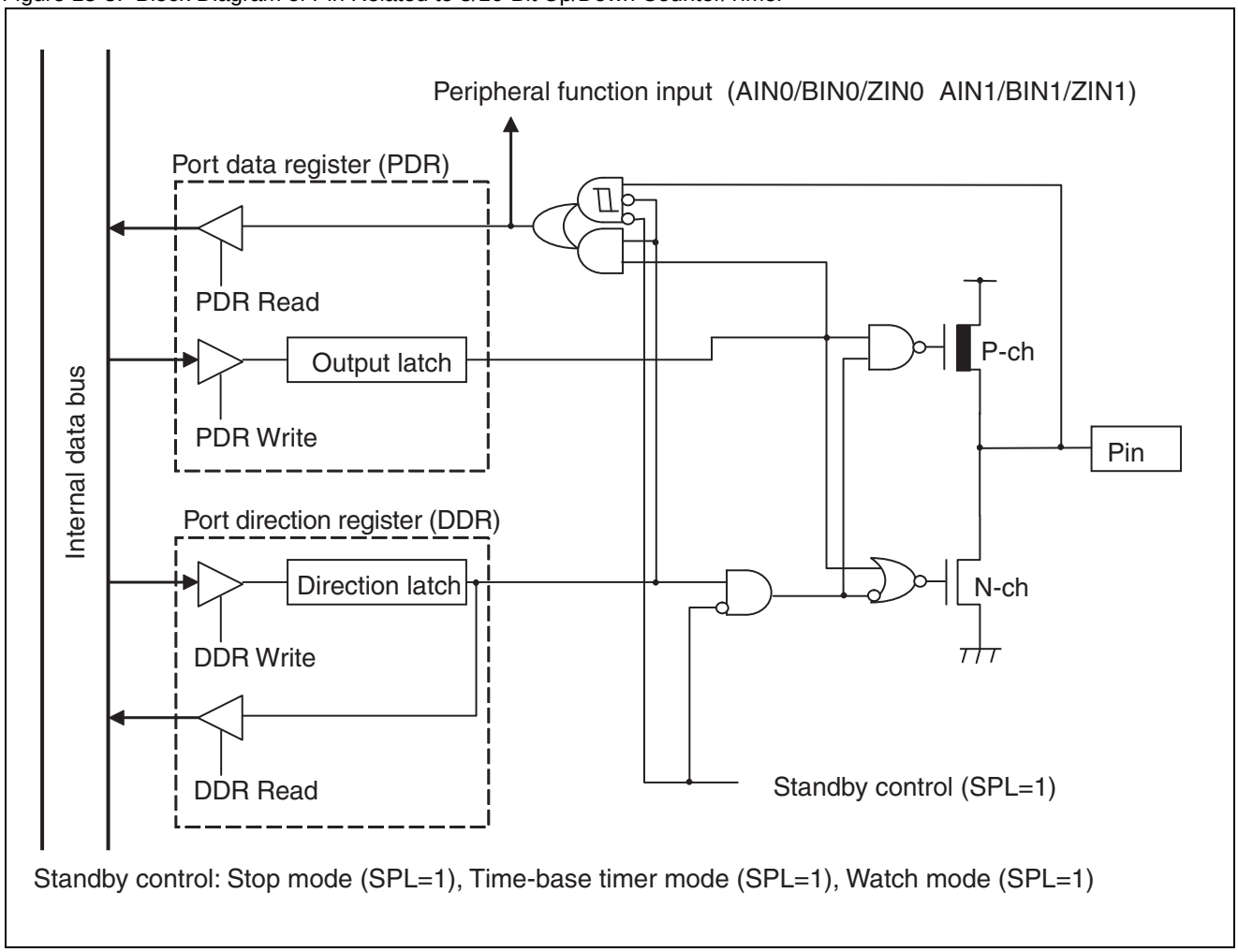
AIN1/BIN1/ZIN1 is sharing a function as the general-purpose I/O port (AIN1/IRQ8/P36, BIN1/IRQ9/P37, ZIN1/UCK2/P35) and the input/output pin of the multi-function serial or the DTP/external interrupt input pin.

Setting when using as AIN0/BIN0/ZIN0 and AIN1/BIN1/ZIN1 pins

In case of using AIN/BIN/ZIN as an input pin, pins AIN0/P31, BIN0/P32, ZIN0/P30, AIN1/P36, BIN1/P37 and ZIN1/P35 should be set as an input port (DDR3 bit0 to bit2, bit5, bit6, bit7 -> “0”) by port direction register.

Block Diagram of Pin Related to 8/16-Bit Up/Down Counter/Timer

Figure 13-3. Block Diagram of Pin Related to 8/16-Bit Up/Down Counter/Timer



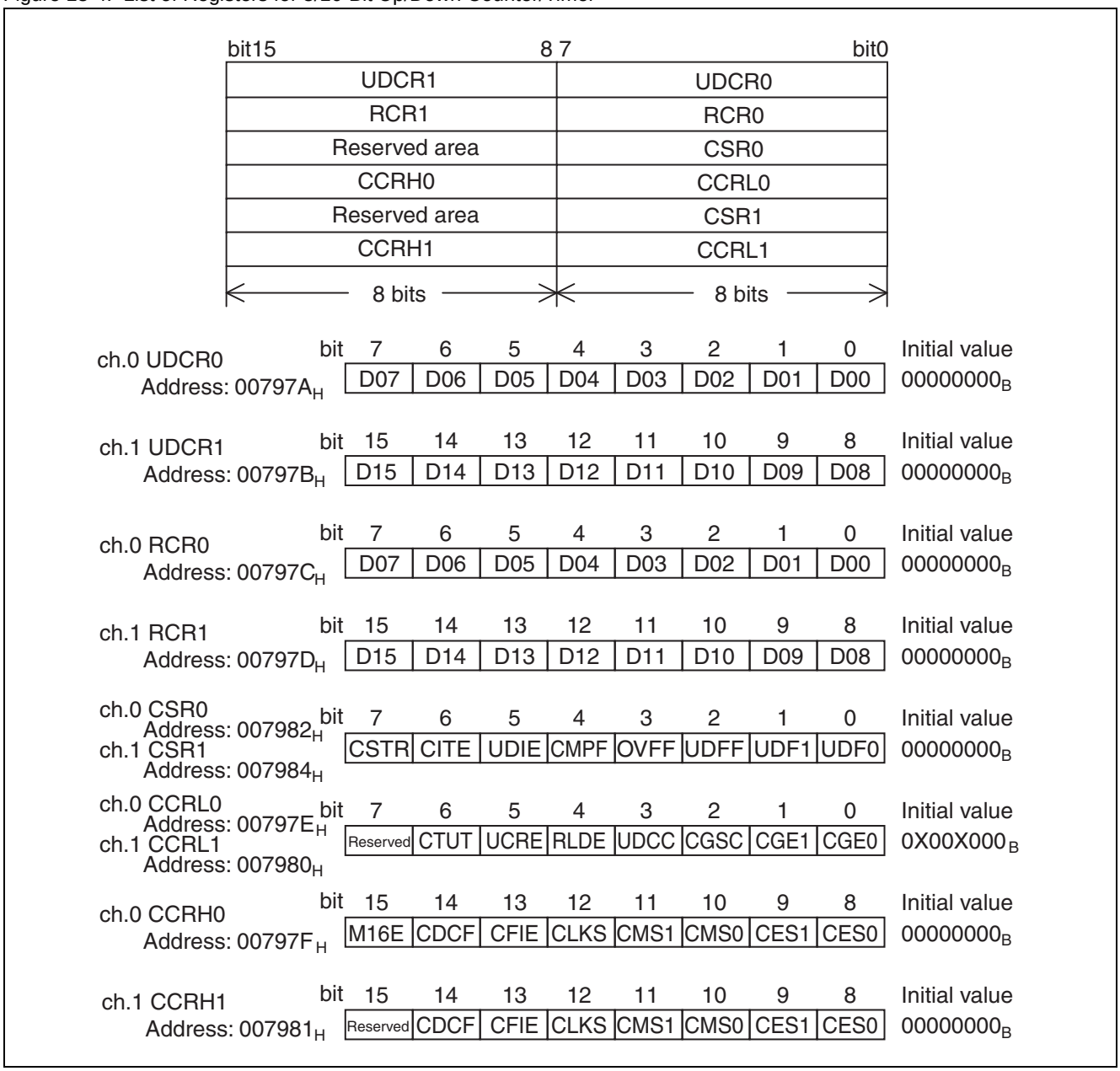
13.3 Configuration and Functions of Registers for 8/16-Bit Up/Down Counter/Timer

This section explains the configuration and function of the 8/16-bit up/down counter/timer registers.

List of 8/16-Bit Up/Down Counter/Timer Registers

Figure 13-4 shows a list of registers for the 8/16-bit up/down counter/timer.

Figure 13-4. List of Registers for 8/16-Bit Up/Down Counter/Timer



13.3.1 Counter Control Register (ch.0) Upper (CCRH0)

This section explains the configuration and functions of counter control register (ch.0) upper (CCRH0).

Counter Control Register (ch.0) Upper (CCRH0)

The bit configuration of counter control register (ch.0) upper (CCRH0) is shown below.

Figure 13-5. Bit Configuration of Counter Control Register (ch.0) Upper (CCRH0)

CCRH0	bit	15	14	13	12	11	10	9	8	Initial value
ch.0 Address: $00797F_H$		M16E	CDCF	CFIE	CLKS	CMS1	CMS0	CES1	CES0	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Counter control register (ch.0) upper (CCRH0) consists of bits that have the functions explained below.

[bit15] M16E (16-bit mode permit)

This bit is used to select (switch) an operation mode of 8 bits × 2 channels or 16 bits × 1 channel.

M16E	Setting 16-bit mode permit
0	8 bits × 2 channels operation mode (initial value)
1	16 bits × 1 channel operation mode

If this bit is rewritten after its start, the count value is not assured.

[bit14] CDCF (count direction reversal flag)

This bit is a flag that is set when the count direction is switched. It is set in the count start mode when the count direction is switched from either up to down or down to up.

The initialization (writing “0”) is only permitted.

Read-modify-write (RMW) instructions always read “1”.

CDCF	Direction reversal detection
0	No reversal of direction (initial value)
1	One or more reversals of direction

[bit13] CFIE (count direction reversal interrupt enable)

If CDCF is defined, this bit is used to control interrupt output to the CPU. An interrupt occurs if count direction changes even a single time in the count start mode when this bit is set to “1”.

CFIE	Direction reversal interrupt output
0	Direction reversal interrupt output prohibit (initial value)
1	Direction reversal interrupt output permit

[bit12] CLKS (built-in prescaler selection)

This bit is used to select the frequency of built-in prescaler in the selection of the timer mode.

It is only valid in the timer mode, and only decrementing (down count) is permitted.

CLKS	Internal clock selected
0	2 machine cycles (initial value)
1	8 machine cycles

If this bit is rewritten after its start, the count value is not assured.

[bit11, bit10] CMS1, CMS0 (count mode selection)

These bits are used to select the count mode.

CMS1	CMS0	Count mode
0	0	Timer mode [decremented] (initial value)
0	1	Up/down count mode
1	0	Phase difference count mode: frequency multiplied by 2
1	1	Phase difference count mode: frequency multiplied by 4

If this bit is rewritten after its start, the count value is not assured.

[bit9, bit8] CES1, CES0 (count clock edge selection)

These bits are used to select the detection edge of external pins AIN and BIN in the up/down count mode.

This setting is invalid in modes other than up/down count.

CES1	CES0	Selected edge
0	0	Edge detect prohibit (initial value)
0	1	Falling edge detect
1	0	Rising edge detect
1	1	Both rising and falling edges detected

If this bit is rewritten after its start, the count value is not assured.

13.3.2 Counter Control Register (ch.1) Upper (CCRH1)

This section explains the configuration and function of counter control register (ch.1) upper (CCRH1).

Counter Control Register (ch.1) Upper (CCRH1)

The bit configuration of the counter control register (ch.1) upper (CCRH1) is shown below.

Figure 13-6. Bit Configuration of Counter Control Register (ch.1) Upper (CCRH1)

CCRH1	bit	15	14	13	12	11	10	9	8	Initial value
ch.1 Address: 007981_H		Reserved	CDCF	CFIE	CLKS	CMS1	CMS0	CES1	CES0	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Counter control register (ch.1) upper (CCRH1) consists of bits that have the functions explained below.

[bit15] Reserved bit

This bit is a reserved bit. Be sure to write “0” to this bit.

[bit14] CDCF (count direction reversal flag)

This bit is set when the count direction changes. It is set in the count start mode when the count direction changes from up to down or from down to up.

The initialization (writing “0”) is only permitted.

Read-modify-write (RMW) instructions always read “1”.

CDCF	Direction reversal detection
0	No direction reversals (initial value)
1	One or more reversals of direction

[bit13] CFIE (count direction reversal interrupt enable)

If CDCF is defined, this bit is used to control interrupt output to the CPU. An interrupt occurs if count direction changes even a single time in the count start mode when this bit is set to “1”.

CFIE	Direction reversal interrupt output
0	Direction reversal interrupt output prohibit (initial value)
1	Direction reversal interrupt output permit

[bit12] CLKS (built-in prescaler selection)

This bit is used to select the frequency of built-in prescaler when the timer mode is selected.

This is only valid in the timer mode, where only decrementing is permitted.

CLKS	Selection internal clock
0	2 machine cycles (initial value)
1	8 machine cycles

If this bit is rewritten after its start, the count value is not assured.

[bit11, bit10] CMS1, CMS0 (count mode selection)

These bits are used to select the count mode.

CMS1	CMS0	Count mode
0	0	Timer mode [decremented] (initial value)
0	1	Up/down count mode
1	0	Phase difference count mode: frequency multiplied by 2
1	1	Phase difference count mode: frequency multiplied by 4

If this bit is rewritten after its start, the count value is not assured.

[bit9, bit8] CES1, CES0 (count clock edge selection)

These bits are used to select a detection edge for external pins AIN and BIN in the up/down count mode.

This setting is invalid in modes other than up/down count.

CES1	CES0	Selected edge
0	0	Edge detect prohibit (initial value)
0	1	Falling edge detect
1	0	Rising edge detect
1	1	Both rising and falling edges detected

If this bit is rewritten after its start, the count value is not assured.

13.3.3 Counter Control Register (ch.0/ch.1) Lower (CCRL0/CCRL1)

This section explains the configuration and function of counter control register (ch.0/ch.1) lower (CCRL0/CCRL1).

Counter Control Register (ch.0/ch.1) Lower (CCRL0/CCRL1)

The bit configuration of counter control register (ch.0/ch.1) lower (CCRL0/CCRL1) is shown below.

Figure 13-7. Bit Configuration of Counter Control Register (ch.0/ch.1) Lower (CCRL0/CCRL1)

	bit	7	6	5	4	3	2	1	0	Initial value
CCRL0 ch.0 Address: 00797E$_H$		Reserved	CTUT	UCRE	RLDE	UDCC	CGSC	CGE1	CGE0	0X00X000$_B$
CCRL1 ch.1 Address: 007980$_H$		R/W	W	R/W	R/W	W	R/W	R/W	R/W	

Counter control register (ch.0/ch.1) lower (CCRL0/CCRL1) consists of bits that have the functions explained below.

[bit7] Reserved bit

This bit is a reserved bit. Be sure to write “0” to this bit.

[bit6] CTUT (counter write)

This bit is used to control data transfers from RCR to UDCR.

If this bit is set to “1”, data is transferred from RCR to UDCR.

Writing “0” has no effect. Read value is always “0”.

Do not write this bit to “1”, during a count operation. (when the CSTR bit of CSR is "1")

[bit5] UCRE (UDCR clear enable)

This bit is used to control UDCR clear caused by compare.

This does not affect the UDCR clear function (such as caused by the ZIN pin setting) other than clear because of compare generation.

UCRE	Counter clear caused by compare
0	Counter clear prohibit (initial value)
1	Counter clear permit

[bit4] RLDE (rel8oad enable)

This bit is used to control the start of the reload function. The RCR value is transferred to UDCR if an underflow occurs when the reload function starts.

RLDE	Reload function
0	Reload function prohibit (initial value)
1	Reload function permit

[bit3] UDCC (UDCR clear)

This bit is used to clear UDCR. Writing “0” to this bit clears UDCR to “0000$_H$”.

Writing “1” has no effect. Read value is always “1”.

[bit2] CGSC (counter clear/gate selection)

This bit is used to select a function of external pin ZIN.

CGSC	ZIN function
0	Counter clear function (initial value)
1	Gate function

[bit1, bit0] CGE1, CGE0 (counter clear/gate edge selection)

These bits are used to select a detection edge/level for external pin ZIN.

CGE1	CGE0	For selecting the counter clear function	For selecting the gate function
0	0	Edge detect prohibited (initial value)	Level detect prohibited (count disable)
0	1	Falling edge	“L” level
1	0	Rising edge	“H” level
1	1	Setting is prohibited	Setting is prohibited

If this bit is rewritten after its start, the count value is not assured.

13.3.4 Counter Status Register 0/1 (CSR0/CSR1)

This section explains the configuration and function of counter status register 0/1 (CSR0/CSR1).

Counter Status Register 0/1 (CSR0/CSR1)

The bit configuration of the counter status register 0/1 (CSR0/CSR1) is shown below.

Figure 13-8. Bit Configuration of Counter Status Register 0/1(CSR0/CSR1)

CSR0 / CSR 1	bit	7	6	5	4	3	2	1	0	Initial value
ch.0 Address: 007982_H		CSTR	CITE	UDIE	CMPF	OVFF	UDFF	UDF1	UDF0	00000000_B
ch.1 Address: 007984_H		R/W	R/W	R/W	R/W	R/W	R/W	R	R	

Counter status register 0/1 (CSR0/CSR1) consists of bits that have the functions explained below.

[bit7] CSTR (count start)

This bit is used to control the UDCR count start/stop operation.

CSTR	Count start/stop operation
0	Count operation stop (initial value)
1	Count operation start

[bit6] CITE (compare interrupt output control)

This bit is used to control permit/prohibition of interrupt output to the CPU if CMPF is defined (if a compare occurs).

CITE	Permit/prohibit of compare interrupt output
0	Compare interrupt output prohibited (initial value)
1	Compare interrupt output permitted

[bit5] UDIE (overflow/underflow interrupt output control)

This bit is used to control the permit/prohibition of interrupt output to the CPU if OVFF/UDFF is defined (if overflow/underflow occurs).

UDIE	Permit/prohibit of overflow/underflow interrupt output
0	Overflow/underflow interrupt output prohibited (initial value)
1	Overflow/underflow interrupt output permitted

[bit4] CMPF (compare detect flag)

This bit is a flag indicating that the UDCR and RCR values match each other after a comparison.

The initialization (writing “0”) is only permitted.

Read-modify-write (RMW) instructions always read “1”.

CMPF	Match/no match at compare detection
0	No match in compare results (initial value)
1	Match in compare results

[bit3] OVFF (overflow detect flag)

This bit is a flag indicating an overflow occurred.

The initialization (writing “0”) is only permitted.

Read-modify-write (RMW) instructions always read “1”.

OVFF	Overflow occurrence
0	No overflow occurred (initial value)
1	Overflow occurred

[bit2] UDFF (underflow detect flag)

This bit is a flag indicating that an underflow occurs.

The initialization (writing “0”) is only permitted.

Read-modify-write (RMW) instructions always read “1”.

UDFF	Underflow occurrence
0	No underflow occurred (initial value)
1	Underflow occurred

[bit1, bit0] UDF1, UDF0 (up/down flag)

These bits are used to indicate the last count operation (up/down) performed.

Only reading is permitted but writing is not.

UDF1	UDF0	Detect edge
0	0	No input (initial value)
0	1	Decremented
1	0	Incremented
1	1	Simultaneous up/down occurred

13.3.5 Up/Down Count Register (ch.0/ch.1) (UDCR0/UDCR1)

This section explains the configuration and function of up/down count register (ch.0/ch.1) (UDCR0/UDCR1).

Up/Down Count Register (ch.0/ch.1) (UDCR0/UDCR1)

The bit configuration of the up/down count register (ch.0/ch.1) (UDCR0/1UDCR) is shown below.

Figure 13-9. Bit Configuration of Up/Down Count Register (ch.0/ch.1) (UDCR0/UDCR1)

	bit	15	14	13	12	11	10	9	8	Initial value
UDCR 1 ch.1 Address: 00797B$_H$		D15	D14	D13	D12	D11	D10	D09	D08	00000000$_B$
		R	R	R	R	R	R	R	R	
	bit	7	6	5	4	3	2	1	0	Initial value
UDCR 0 ch.0 Address: 00797A$_H$		D07	D06	D05	D04	D03	D02	D01	D00	00000000$_B$
		R	R	R	R	R	R	R	R	

This register is an 8-bit count register. With an internal prescaler or AIN/BIN pin input, an up/down count operation is performed.

It operates as a 16-bit count register in the 16-bit count mode. In this case, the high-order 8-bit setting of the control register is disabled.

This register cannot be written. To write to this register, be sure to write via RCR. The value to be written to this register must first be written to RCR, and then it is transferred from RCR value to this register by setting the CCRL: CTUT bit to “1” (reloading by software).

This register requires word access for reading.

13.3.6 Reload/Compare Register (ch.0/ch.1) (RCR0/RCR1)

This section explains the configuration and function of reload/compare register (ch.0/ch.1) (RCR0/RCR1).

Reload/Compare Register (ch.0/ch.1) (RCR0/RCR1)

The bit configuration of reload/compare register (ch.0/ch.1) (RCR0/RCR1) is shown below.

Figure 13-10. Bit Configuration of Reload/Compare Register (ch.0/ch.1) (RCR0/RCR1)

	bit	15	14	13	12	11	10	9	8	Initial value
RCR1 ch.1 Address: 00797D$_H$		D15	D14	D13	D12	D11	D10	D09	D08	00000000$_B$
		W	W	W	W	W	W	W	W	
	bit	7	6	5	4	3	2	1	0	Initial value
RCR0 ch.0 Address: 00797C$_H$		D07	D06	D05	D04	D03	D02	D01	D00	00000000$_B$
		W	W	W	W	W	W	W	W	

Reload/compare registers (ch.0/ch.1) (RCR0/RCR1) are used to specify a reload value and compare value. The reload value and compare value have the same value and, by starting the reload function and compare function, an up/down count is available between the 00_H and RCR values (16-bit operation mode: 0000_H to RCR value).

This register allows write-only operations but not read operations. Writing “1” to the CCR0/CCR1: CTUT bit transfers the register value to UDCR (reloading by software).

Write to this register with word access.

13.4 Interrupt of 8/16-Bit Up/Down Counter/Timer

The interrupt of the 8/16-bit up/down counter/timer occurs when the count direction is changed only once during count start, when an match of comparison result is detected, or when the overflow/underflow occurs.

The DMA transfer and extended intelligent I/O service (EI^{2}OS) cannot be activated for the interrupt of the 8/16-bit up/down counter/timer.

Interrupt of 8/16-Bit Up/Down Counter/Timer

Table 13-1 shows the interrupt control bit and interrupt source of the 8/16-bit up/down counter/timer.

Table 13-1. Interrupt of 8/16-Bit Up/Down Counter/Timer

	Count direction detection interrupt	Overflow/ underflow interrupt	Counter compare match interrupt
Interrupt request flag	CCRH0: CDCF (bit14) ch.0 CCRH1: CDCF (bit14) ch.1	CSR0: OVFF (bit3) ch.0 UDFF (bit2) CSR1: OVFF (bit3) ch.1 UDFF (bit2)	CSR0: CMPF (bit4) ch.0 CSR1: CMPF (bit4) ch.1
Interrupt request output enable bit	CCRH0: CFIE (bit13) ch.0 CCRH1: CFIE (bit13) ch.1	CSR0: UDIE (bit5) ch.0 CSR1: UDIE (bit5) ch.1	CSR0: CITE (bit6) ch.0 CSR1: CITE (bit6) ch.1
Interrupt generation source	Up/down counter direction detection	Overflow/underflow detection	Match between value of up/down counter and that of reload/ compare register

CCRH0/OCR0 correspond to up/down counter pins (AIN0/BIN0/ZIN0).
CCRH1/OCR1 correspond to up/down counter pins (AIN1/BIN1/ZIN1).

Count direction change interrupt

The operation for generating the count direction change interrupt is shown below.

- Bit14: CDCF flag of the counter control register (CCRH0/CCRH1) is set to “1”.
- While bit13: CFIE of the interrupt request (CCRH0/CCRH1) is enabled (“1”). When the count direction is changed only once during count start, the interrupt occurs.

Overflow/underflow interrupt

The operation for generating the overflow/underflow interrupt is shown below.

- Bit5: UDIE flag of the counter status register (CSR0/CSR1) is set to “1”.
- If bit3: OVFF or bit2: UDFF of the counter status register (CSR0/CSR1) is set to “1”, the interrupt request occurs.

Counter compare match interrupt

The operation for generating the compare interrupt is shown below.

- Bit6: CITE flag of the counter status register (CSR0/CSR1) is set to “1”.
- When a comparison result between the UDCR value and RCR value using bit4: CMPF of the counter status register (CSR0/CSR1) matches, the interrupt request occurs.

Interrupt of 8/16-Bit Up/Down Counter/Timer, DMA Transfer, and EI^{2}OS

Table 13-2 shows the relationship between the interrupt source, interrupt vector, and interrupt control register other than software interrupt.

Table 13-2. Interrupt Source, Interrupt Vector, and Interrupt Control Register

Interrupt source	EI^{2}OS clear	µDMAC channel number	Interrupt vector		Interrupt control register	
			Number	Address	Number	Address
8/16-bit up/down counter/timer* (ch.0, ch.1) Compare/underflow/overflow/reverse up/down	´	-	#23	FFFFA0$_H$	ICR06	0000B6$_H$

×: Interrupt request flag is not cleared.

❍: Interrupt request flag is cleared.

*: This interrupt source shares the interrupt source and interrupt number of other peripheral function. For details, see Table A-3.

Note: If there are two interrupt sources in the same interrupt number, resource clears both interrupt request flags. Therefore, when one of two sources uses the EI^{2}OS/µDMAC function, other interrupt function cannot be used. The interrupt request enable bit of the relevant resource is set to "0" to execute the software polling processing.

Correspondence to DMA Transfer and EI^{2}OS Function

The 8/16-bit up/down counter/timer does not correspond to the DMA transfer function, but the EI^{2}OS function. When the EI^{2}OS function is used, it is necessary to disable other interrupt that shares the interrupt control register (ICR).

13.5 Operation of 8/16-Bit Up/Down Counter/Timer

This section explains different count modes in the 8/16-bit up/down counter/timer and the operation of the reload/compare function.

Selection of Count Mode

The 8/16-bit up/down counter/timer has four types of count modes. These count modes are selected by CCRH: CMS1 and CMS0.

Table 13-3. Selection of Count Mode

CMS1	CMS0	Count mode
0	0	Timer mode (decremented)
0	1	Up/down count mode
1	0	Phase difference count mode: frequency multiplied by 2
1	1	Phase difference count mode: frequency multiplied by 4

Timer mode (decremented)

In the timer mode, output of the internal prescaler is decremented. The built-in prescaler enables selection of either 2 machine cycles or 8 machine cycles with CCRH: CLKS.

Up/down count mode

In the up/down count mode, counting the input of external pins AIN and BIN enables an up/down count. AIN pin input controls increments, and BIN pin input controls decrements.

Input of the AIN pin and BIN pin indicates an edge detection for input, detection edge can be selected by CCRH: CES1 and CES0.

Table 13-4. Selection of Edge Detection

CES1	CES0	Detect edge
0	0	Edge detect prohibit
0	1	Falling edge detect
1	0	Rising edge detect
1	1	Both falling and rising edges detected

Phase Difference Count Mode (at Frequency Multiplied by 2/Frequency Multiplied by 4)

In the phase difference count mode, to count the encoder phase difference between output signal phases A and B, the BIN pin input level is checked for counting if the AIN pin input edge is detected, and the AIN pin input level is checked for counting if the BIN pin input edge is detected.

In the modes at frequency multiplied by 2 and frequency multiplied by 4, the phase difference is checked between AIN and BIN pin inputs. If the AIN pin is advanced, it is incremented and, if the BIN pin is advanced, it is decremented.

In the mode at frequency multiplied by 2, at the timing of both the rising and falling edges of the BIN pin, counting is done as required by checking for the AIN pin value. Count operations in this case are as follows:

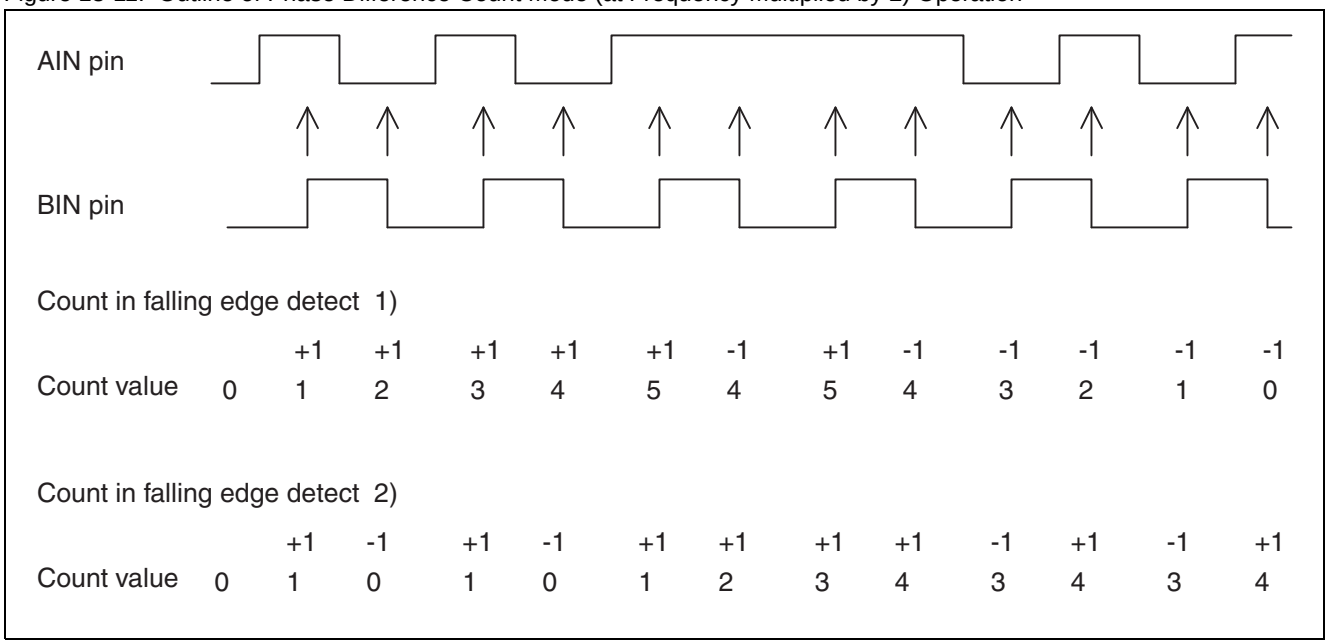
- Incremented if the AIN pin value detected at the rising edge of the BIN pin is “H”
- Decremented if the AIN pin value detected at the rising edge of the BIN pin is “L”

The AIN pin value detected at the falling edge of the BIN pin is selected from the following two types 1) and 2):

1). Decremented if the AIN pin value detected at the falling edge of BIN pin is “H”
Incremented if the AIN pin value detected at the falling edge of the BIN pin is “L”

2). Decremented if the AIN pin value detected at the falling edge of the BIN pin is “L”
Incremented if the AIN pin value detected at the falling edge of the BIN pin is “H”

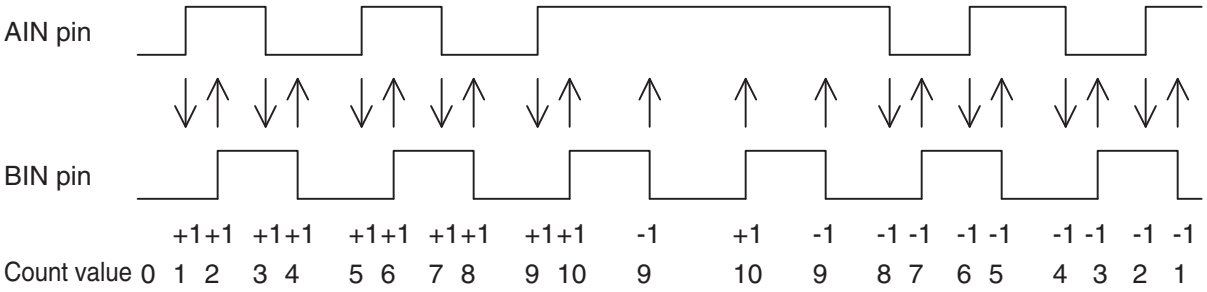
Figure 13-11. Outline of Phase Difference Count Mode (at Frequency Multiplied by 2) Operation



In the mode at frequency multiplied by 4, the AIN pin value is checked for counting at the timing of both the BIN pin rising and falling edges. BIN pin value is checked for counting at the timing of both the AIN pin rising and falling edges. Count operations for such cases are described below.

- Incremented if the AIN pin value detected at the rising edge of the BIN pin is “H”
- Decremented if the AIN pin value detected at the rising edge of the BIN pin is “L”
- Decremented if the AIN pin value detected at the falling edge of the BIN pin is “H”
- Incremented if the AIN pin value detected at the falling edge of the BIN pin is “L”
- Decremented if the BIN pin value detected at the rising edge of the AIN pin is “H”
- Incremented if the BIN pin value detected at the rising edge of the AIN pin is “L”
- Incremented if the BIN pin value detected at the falling edge of the AIN pin is “H”
- Decremented if the BIN pin value detected at the falling edge of the AIN pin is “L”

Figure 13-12. Outline of Phase Difference Count Mode (at Frequency Multiplied by 4) Operation



In counting the encoder output, the input condition must be arranged by defining the relationship between the phases and pins shown below. Thus, high-precision detection can enable the rotation angle, rotation count, and rotation direction to be measured.

- Inputting phase A to the AIN pin
- Inputting phase B to the BIN pin
- Inputting phase Z to the ZIN pin

If this count mode is selected, the selection of detection edge via CCRH: CES1/CES0 and CCRL: CGE1/CGE0 is disabled.

13.5.1 Reload/Compare Function

The 8/16-bit up/down counter/timer has reload and compare functions. These two functions may be mixed for processing.

Selection of Reload and Compare Functions

Table 13-5 shows an example of selecting reload and compare functions.

Table 13-5. Selection Example of Reload/Compare Function

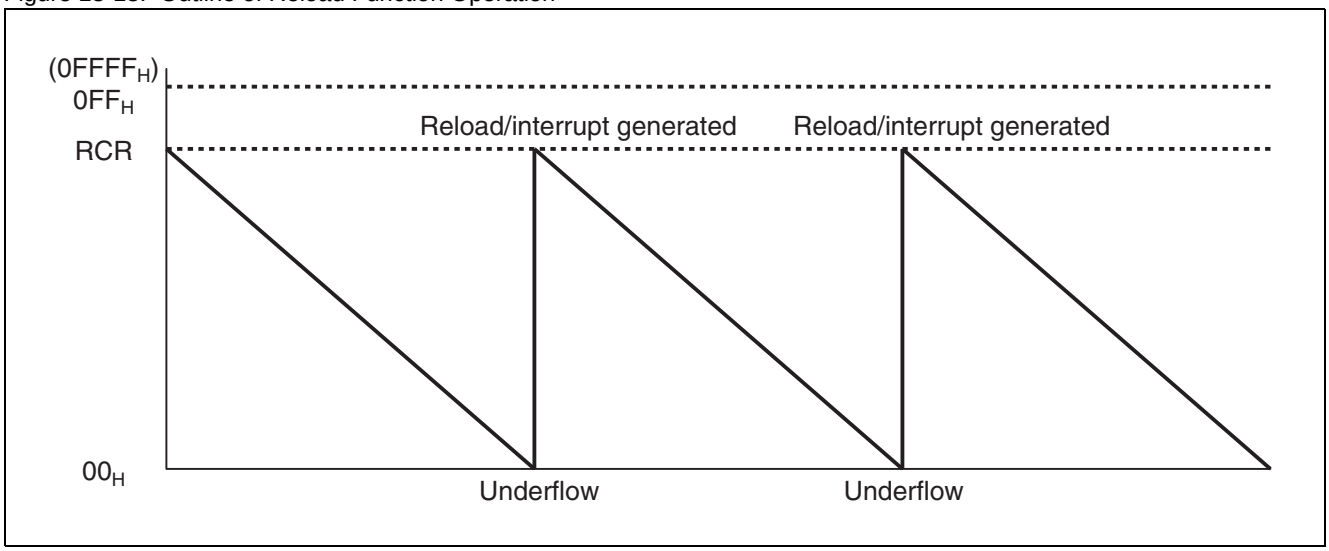
RLDE	UCRE	Reload and compare functions
00_B	00_B	Reload/compare prohibit (initial value)
01_B	01_B	Compare permit
10_B	10_B	Reload permit
11_B	11_B	Reload/compare permit

Reload Function

At the start of the reload function, the RCR value is transferred to UDCR at the timing of the down-count clock next to the clock in which an underflow occurs. In this example, UDFF is specified, and an interrupt request occurs.

In a mode where there is no down counting (decrement), the start of this function is disabled.

Figure 13-13. Outline of Reload Function Operation

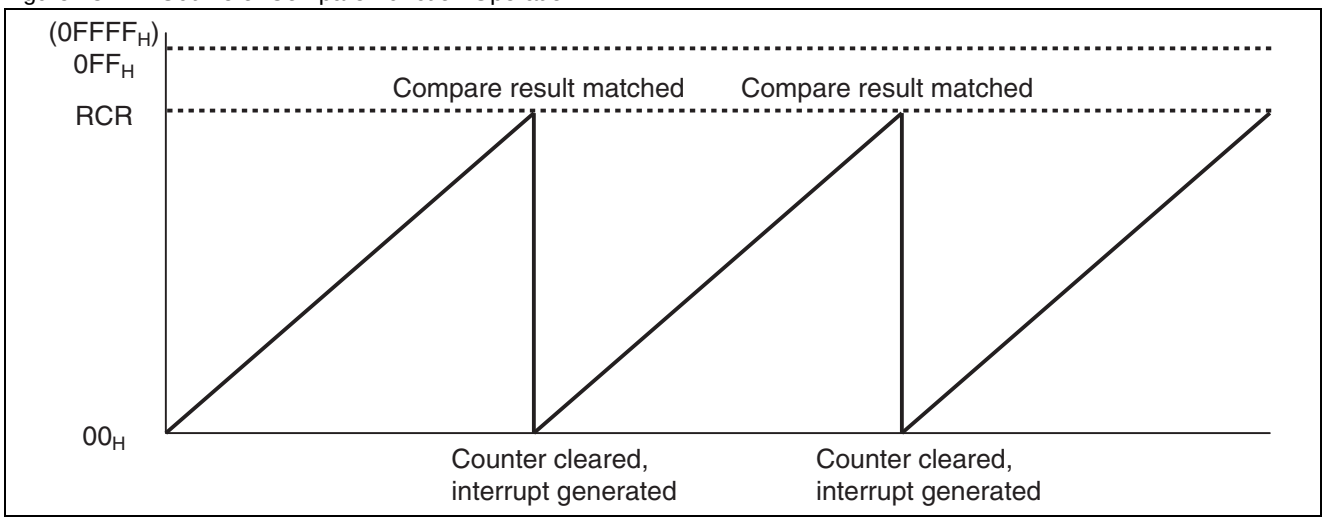


Compare Function

The compare function is available in any modes other than the timer mode. If RCR and UDCR values match at the start of the compare function, CMPF is specified and an interrupt request occurs. When the compare clear function starts, UDCR is cleared at the next timing of the incremented clock.

In a mode where there is no up counting, the start of this function is disabled.

Figure 13-14. Outline of Compare Function Operation

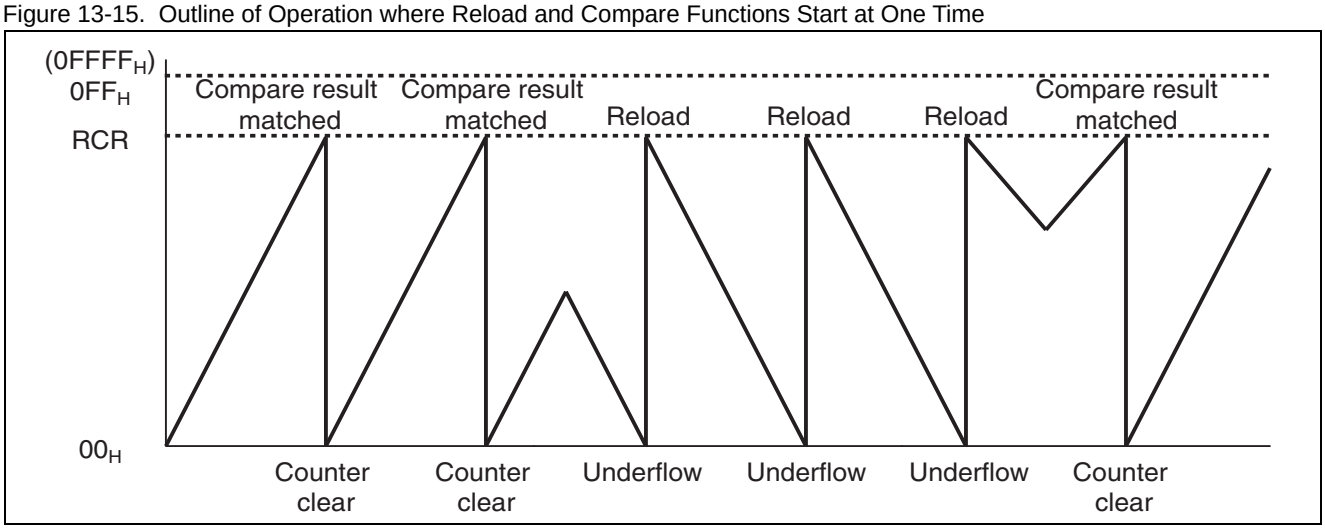


Up/Down Count at Any Width in Reload/Compare Function

When a reload/compare function starts, an up/down count is available at any width.

When an underflow occurs by the reload function, the RCR value is transferred to UDCR. The compare function clears UDCR if RCR and UDCR have matching values. Using both functions, an up/down count is performed in a range of 00_H to RCR.

Figure 13-15. Outline of Operation where Reload and Compare Functions Start at One Time



If compare result finds a match or a reload (underflow) occurs, an interrupt can be generated to CPU. These interrupt outputs are controlled so that they are enabled independently.

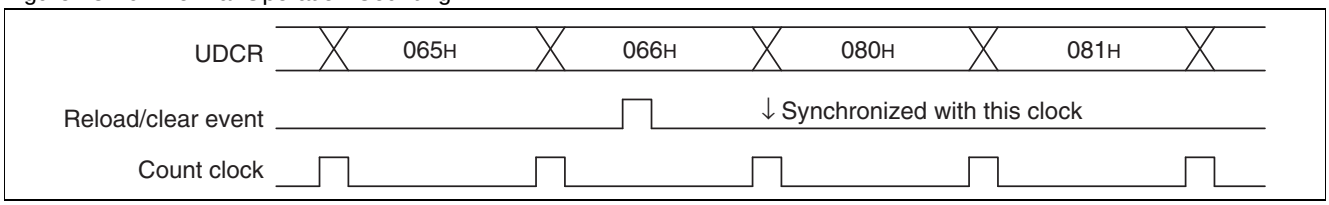
The timing of reload and clear operations for UDCR varies between the count start and stop modes.

Reloading (writing “1” to the CTUT bit) by software during counting is prohibited.

If reload or clear events are generated in a count operation

All updating operations of UDCR are in synch with the count clock. Figure 13-16 shows an example of reloading 080_H.

Figure 13-16. Normal Operation Counting

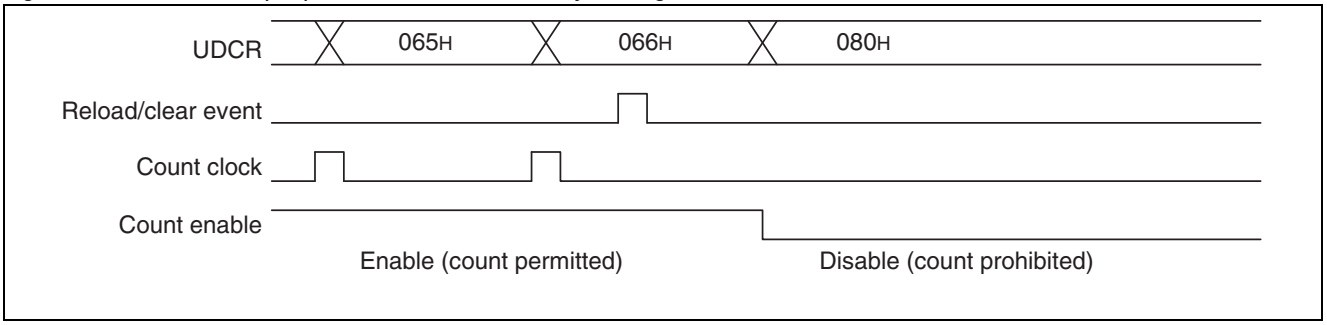


If reload or clear events are generated just before the count operation stops

If counting stops in the count clock synch wait mode (state where count input is held for synchronization), reload and clear operations to UDCR are performed when the stop occurs.

Figure 13-17 shows an example of reloading 080_H.

Figure 13-17. Count Stop Operation in Count Clock Synch Signal Wait Mode

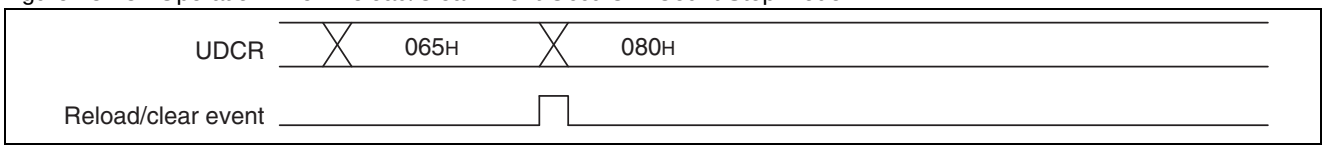


If reload and clear events are generated in the count stop mode

Update of UDCR is performed when an event occurs.

Figure 13-18 shows an example of reloading 080_H.

Figure 13-18. Operation when Reload/Clear Event Occurs in Count Stop Mode



If counter is cleared by the comparison result match

A clear operation caused by compare is performed if the UDCR and RCR values match and incrementing (up count) occurs. Even if the UDCR and RCR values match, no clear operation is performed if a down-count or count stop occurs subsequently.

A clear operation is performed at the above timing for all events other than reset input. Reload is also performed at the above timing in any event.

If clear and reload events occur at the same time, the clear event has priority.

13.5.2 Writing Data to Up/Down Count Register (UDCR)

Writing data directly to UDCR from a data bus is not permitted. This section includes procedures for writing any data to UDCR.

Writing Data to UDCR

Data can be written to UDCR with the following procedures:

1. Write the data to be written to UDCR to RCR first.
2. Writing “1” to CCRH: CTUT transfers the data from RCR to UDCR.

Perform the above procedure while counting is stopped (when the CSTR bit in the CSR is 0). If “1” is written to the CTUT bit by mistake during counting, the value of the RCR is transferred to the UDCR at the timing for a write.

Clearing the Counter

In addition to write “0000_H” to UDCR, the following procedures also clear the counter.

- Clearing with reset input (initialize)
- Clearing with an edge input from the ZIN pin
- Clearing by writing “0” to CCRL: UDCC
- Clearing with the compare function

Such clear operations are is performed regardless of the occurrence for count start/stop.

Count Clear/Gate Function

The ZIN pin is used as either a count clear or gate function by CCRH: CGSC.

If the count clear function starts, the counter is cleared by the edge input from the ZIN pin. CCRL: CGE1/CGE0 select the edge of the ZIN pin input signal where the counter is cleared.

When the gate function starts, the count is enabled or disabled depending on the level input from the ZIN pin. The level of the ZIN pin input signal used to enable the count selects using CCRL: CGE1/CGE0.

This function is available for all count modes.

Table 13-6. Selection of ZIN Pin Function

CGSC	ZIN pin function
0	Counter clear function
1	Gate function

Table 13-7. Count Clear/Gate Function

CGE1	CGE0	Counter clear function	Gate function
0	0	Detect prohibited	Detect prohibited
0	1	Rising edge	“L” level
1	0	Falling edge	“H” level
1	1	Setting is prohibited	Setting is prohibited

Count Direction Flag, Count Direction Reversal Flag

The count direction flags (UDF1, UDF0) indicate whether the last count operation was either an up-count or down-count when an up- or down-count was performed. By evaluating the count clock generated by input of both the AIN and BIN pins, a flag is updated at every count operation. If information on the current rotation direction is required for controlling the motor, it is identified by checking these flags.

This function is available in all count modes.

Table 13-8. Count Direction Flags

UDF1	UDF0	Count direction
0	0	No input [Initial value]
0	1	Down count (decrement)
1	0	Up count (increment)
1	1	Up/down simultaneously generated (no count operation performed)

The count direction reversal flag (CDCF) is set when the count direction is switched between counting up and counting down. When this flag is set, an interrupt is generated to the CPU. By referring to this interrupt and the count direction flags (UDF1, UDF0), changes of direction are identified. However, note that the direction indicated by the flag may be restored to the original one and the correct count direction reversal cannot be detected after one reversal of direction if the duration of the direction reversal is short and/or occurs repeatedly.

Table 13-9. Count Direction Reversal Flag

CDCF	Count direction reversal detection
0	No direction reversal [Initial value]
1	Direction reversal (one or more times)

Compare Detection Flag

The compare detection flag (CMPF) is set when the UDCR and RCR values become equal to each other during counting. This flag is also set at an increment/decrement match, a match upon occurrence of a reload event, or at a match upon count activation.

8/16-Bit Operation

This module can be used as an 8-bit up/down counter or 16-bit up/down counter. Writing “0” to the M16E bit in the CCR register selects 8-bit mode; writing “1” selects 16-bit mode.

Interrupt Generation Timing

Interrupt flag	Flag set interrupt	Reload	Clear
CDCF (count direction reversal flag)	At the count at which the count direction is changed, an interrupt occurs as soon as the flag is set.	–	–
CMPF (compare detect flag)	When incrementing, decrementing, or reload counting is started, an interrupt occurs as soon as the flag is set at a match between RCR and UDCR.	–	The UDCR is cleared (not during decrementing) at the increment timing that follows the match between RCR and UDCR.
OVFF (overflow detect flag)	At the increment timing that follows a count of "$FFFF_H$", an interrupt occurs as soon as the flag is set.	–	The UDCR is cleared at the count timing that follows a count of "$FFFF_H$".
UDFF (underflow detect flag)	At the decrement timing that follows a count of "0000_H", an interrupt occurs as soon as the flag is set.	The RCR value is transferred to the UDCR at the count timing that follows a count of "0000_H".	–

- Once an interrupt occurs, counting remains stopped until the interrupt flag is cleared.
- As the RCR value serves as both of the reload and compare values, the compare flag is set whenever reloading is executed.
- Clearing occurs when incrementing is performed after a compare match during decrementing with the clear function enabled.

Note

The count direction set immediately after a count reset is decrementing. Accordingly, incrementing performed immediately after a reset sets the CDCF bit to "1" to indicate that the count direction has been changed.

When the up count register (UDCR) reaches the full count, counting continues without a carry. Apparently, the up/down count register is cleared and counting continues.

14. PPG Timer

This chapter describes the operations of the PPG timer.

14.1 Overview of PPG Timer

14.2 Block Diagram of PPG Timer

14.3 Registers of PPG Timer

14.4 Interrupts of PPG Timer

14.5 Operational Description of PPG Timer

14.1 Overview of PPG Timer

The PPG timer consists of a prescaler, a 16-bit down counter, 16-bit data registers with a buffer for setting the cycle, 16-bit compare registers with a buffer for setting the duty, and a pin control unit.

This timer can output the pulse that is synchronized with a software trigger. The cycle and duty of the pulse to be output can be modified freely by rewriting two 16-bit register values.

Overview

PWM function

This function allows the user to output a pulse in a programmable manner while synchronizing it with the trigger and rewriting the register value.

Also, it can be used as a D/A converter by using the external circuit.

One-shot function

This function allows the user to output a single pulse by a trigger input.

Pin control

- A duty match sets the pin to “1” (Higher priority)
- A counter borrow resets the pin to “0”
- Fixed output value mode is available and can be selected to output all “L” (or all “H”) easily
- A polarity can be specified

16-bit down counter

- Any of four counter operation clocks can be selected.
  Four internal clocks: ϕ, ϕ/4, ϕ/16, ϕ/64 (ϕ: machine clock)

Interrupt requests

- Startup of the timer
- Occurrence of a counter borrow (cycle match)
- Occurrence of a duty match
- Occurrence of a counter borrow (cycle match) or a duty match
- A soft trigger can be used to set more than one channel to start up simultaneously and also to restart during the operation.

Simultaneous startup of multiple channels

Up to eight channels can be booted up at the same time.

14.2 Block Diagram of PPG Timer

This section shows block diagrams of the PPG timer.

Block Diagram

Figure 14-1. Block Diagram

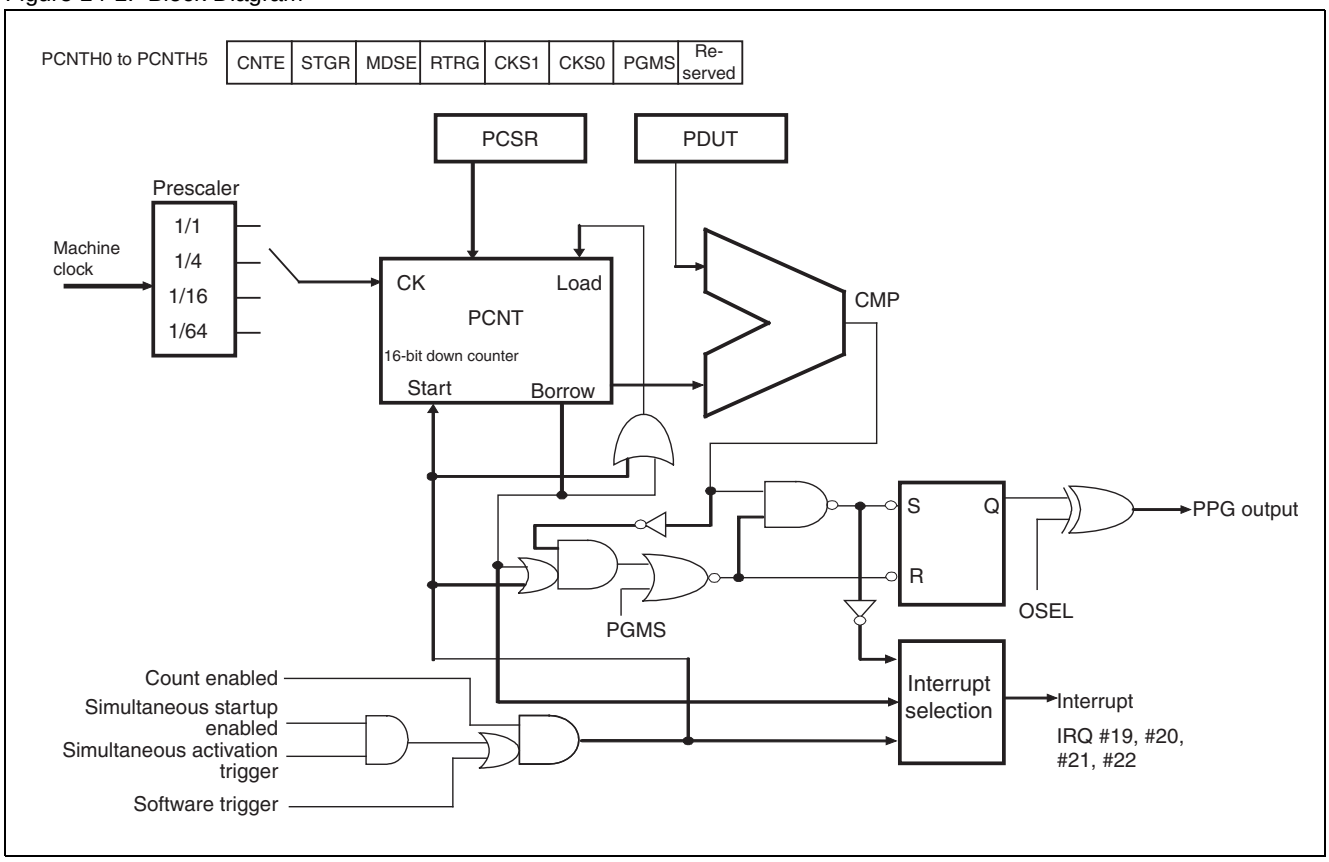
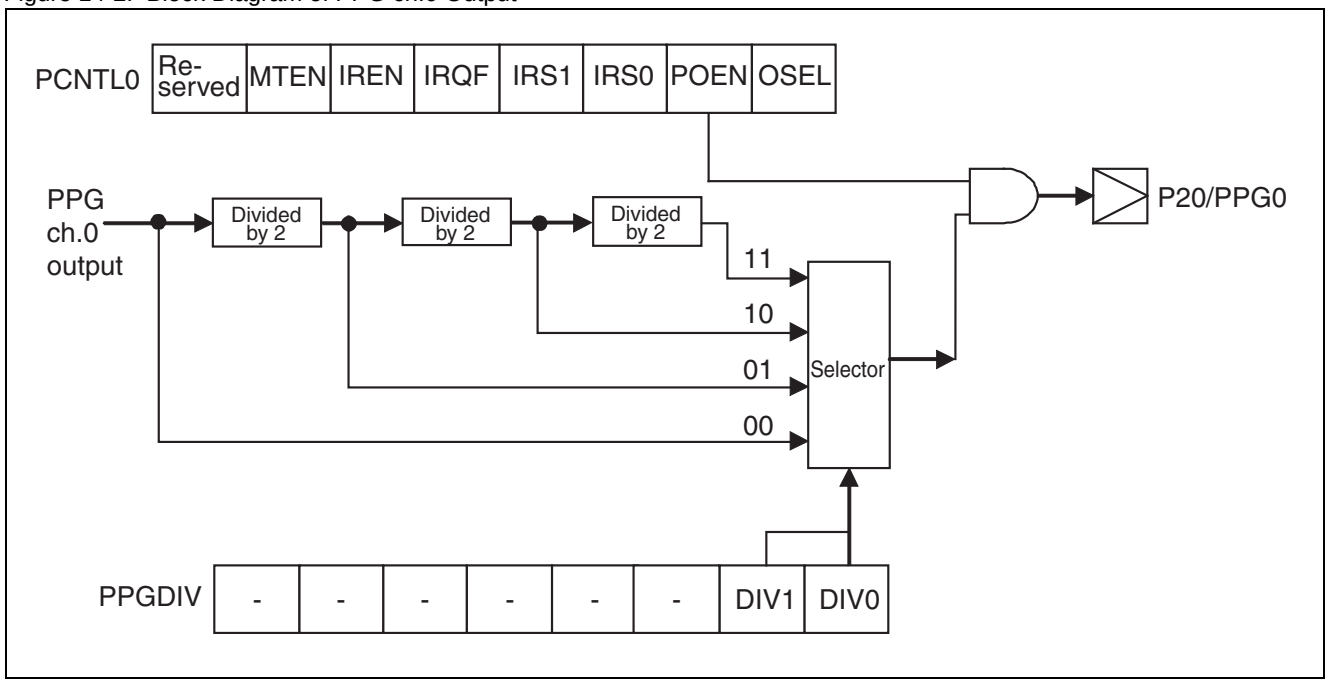


Figure 14-2. Block Diagram of PPG ch.0 Output



14.3 Registers of PPG Timer

This section explains the registers of PPG timer.

14.3.1List of Registers of PPG Timer

14.3.2Details of Registers of PPG Timer

14.3.1 List of Registers of PPG Timer

This section lists and describes the registers of the PPG timer.

List of Registers of PPG Time

Upper PPG control status register

Address: ch.0 007901_H
Address: ch.1 007903_H
Address: ch.2 007905_H
Address: ch.3 007907_H
Address: ch.4 007909_H
Address: ch.5 $00790B_H$
Address: ch.6 $00790D_H$
Address: ch.7 $00790F_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
CNTE	STGR	MDSE	RTRG	CKS1	CKS0	PGMS	Re-served	PCNTH0 to PCNTH7
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(1)	← Initial value

Lower PPG control status register

Address: ch.0 007900_H
Address: ch.1 007902_H
Address: ch.2 007904_H
Address: ch.3 007906_H
Address: ch.4 007908_H
Address: ch.5 $00790A_H$
Address: ch.6 $00790C_H$
Address: ch.7 $00790E_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Re-served	MTEN	IREN	IRQF	IRS1	IRS0	POEN	OSEL	PCNTL0 to PCNTL7
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

Upper PPG down counter register

Address: ch.0 007913_H
Address: ch.1 $00791B_H$
Address: ch.2 007923_H
Address: ch.3 $00792B_H$
Address: ch.4 007933_H
Address: ch.5 $00793B_H$
Address: ch.6 007943_H
Address: ch.7 $00794B_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
DC15	DC14	DC13	DC12	DC11	DC10	DC09	DC08	PDCRH0 to PDCRH7
R	R	R	R	R	R	R	R	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

Lower PPG down counter register

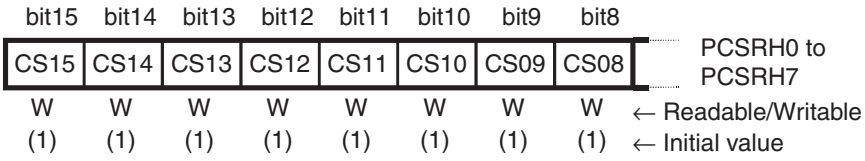
Address: ch.0 007912_H
Address: ch.1 $00791A_H$
Address: ch.2 007922_H
Address: ch.3 $00792A_H$
Address: ch.4 007932_H
Address: ch.5 $00793A_H$
Address: ch.6 007942_H
Address: ch.7 $00794A_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
DC07	DC06	DC05	DC04	DC03	DC02	DC01	DC00	PDCRL0 to PDCRL7
R	R	R	R	R	R	R	R	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

(Continued)

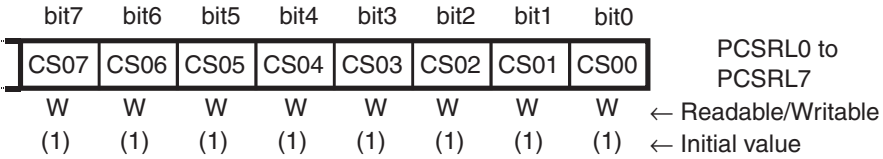
Upper PPG cycle setup register

Address: ch.0 007915H
Address: ch.1 00791DH
Address: ch.2 007925H
Address: ch.3 00792DH
Address: ch.4 007935H
Address: ch.5 00793DH
Address: ch.6 007945H
Address: ch.7 00794DH



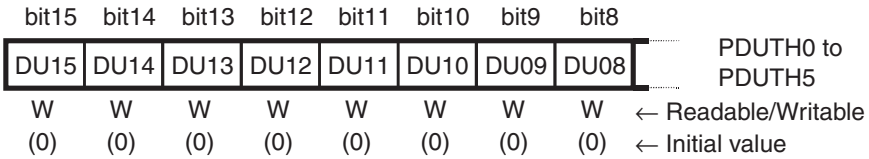
Lower PPG cycle setup register

Address: ch.0 007914H
Address: ch.1 00791CH
Address: ch.2 007924H
Address: ch.3 00792CH
Address: ch.4 007934H
Address: ch.5 00793CH
Address: ch.6 007944H
Address: ch.7 00794CH



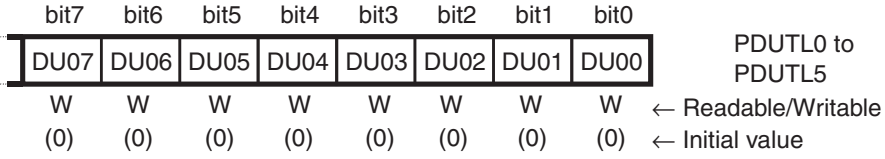
Upper PPG duty setup register

Address: ch.0 007917H
Address: ch.1 00791FH
Address: ch.2 007927H
Address: ch.3 00792FH
Address: ch.4 007937H
Address: ch.5 00793FH
Address: ch.6 007947H
Address: ch.7 00794FH



Lower PPG duty setup register

Address: ch.0 007916H
Address: ch.1 00791EH
Address: ch.2 007926H
Address: ch.3 00792EH
Address: ch.4 007936H
Address: ch.5 00793EH
Address: ch.6 007946H
Address: ch.7 00794EH



PPG0 output division setup register

Address: ch.0 007910H



(Continued)

(Continued)

PPG multi-channel simultaneous startup register

Address: 00799B$_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
MT7	MT6	MT5	MT4	MT3	MT2	MT1	MT0	PMSSR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

14.3.2 Details of Registers of PPG Timer

The following five registers function as PPG timer registers.

- PPG control status registers (PCNT0 to PCNT7)
- PPG down counter registers (PDCR0 to PDCR7)
- PPG cycle setup registers (PCSR0 to PCSR7)
- PPG duty setup registers (PDUT0 to PDUT7)
- PPG multi-channel simultaneous startup register (PMSSR)

PPG Control Status Register (PCNT)

Upper PPG control status register

Address: ch.0 007901H
Address: ch.1 007903H
Address: ch.2 007905H
Address: ch.3 007907H
Address: ch.4 007909H
Address: ch.5 00790BH
Address: ch.6 00790DH
Address: ch.7 00790FH

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
CNTE	STGR	MDSE	RTRG	CKS1	CKS0	PGMS	Re-served	PCNTH0 to PCNTH7
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(1)	← Initial value

Lower PPG control status register

Address: ch.0 007900H
Address: ch.1 007902H
Address: ch.2 007904H
Address: ch.3 007906H
Address: ch.4 007908H
Address: ch.5 00790AH
Address: ch.6 00790CH
Address: ch.7 00790EH

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Re-served	MTEN	IREN	IRQF	IRS1	IRS0	POEN	OSEL	PCNTL0 to PCNTL7
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

[bit15] CNTE: Timer enable bit

This bit is used to enable the operation of the 16-bit down counter.

0	Stops operation (initial value)
1	Enables operation

[bit14] STGR: Software trigger bit

A software trigger is applied when “1” is written to the STGR bit.

The STGR bit always reads “0”.

[bit13] MDSE: Mode selection bit

This bit is used to select either the PWM operation that generates pulses continuously or the one-shot operation that generates a single pulse. Do not attempt to rewrite this bit during operation.

0	PWM operation (initial value)
1	One-shot operation

[bit12] RTRG: Restart enable bit

This bit is used to enable the timer to be restarted by a trigger (software). Do not attempt to rewrite this bit during operation.

0	Disables restart (initial value)
1	Enables restart

[bit11, bit10] CKS1, CKS0: Count clock selection bits

These bits are used to select a count clock for the 16-bit down counter. Do not attempt to rewrite these bits during operation.

CKS1	CKS0	Cycle
0	0	ϕ (initial value)
0	1	ϕ/4
1	0	ϕ/16
1	1	ϕ/64

ϕ: Machine clock

[bit9] PGMS: PPG output mask selection bit

When “1” is written to the PGMS bit, the PPG output can be masked to “0” or “1”, regardless of the mode setting, cycle setting value, and duty setting value.

0	No output mask (initial value)
1	Output mask

PPG output when “1” is written to PGMS

Polarity	PPG output
Normal polarity	“L”
Inverted polarity	“H”

When outputting either all “H” during the normal polarity or all “L” during the inverted polarity, the above mask value will be inverted and then output if the same value is written to the cycle setup register and the duty setup register.

[bit8] Reserved bit

Always write “1” to this bit. Read value is always “1”.

[bit7] Reserved bit

Always write “0” to this bit. Read value is always “0”.

[bit6] MTEN: PPG multi-channel simultaneous startup enable bit

When “1” is written to the MTEN bit, that channel will become subject to multi-channel simultaneous startup. After the simultaneous startup, writing “0” to the MTEN bit has no effect on operation.

0	Disables simultaneous startup (initial value)
1	Enables simultaneous startup

[bit5] IREN: Interrupt request enable bit

This bit is used to enable interrupts of the PPG timer. An interrupt will occur, if the interrupt flag (bit4: IRQF) is set to “1” when the IREN bit is set to “1”.

0	Disables interrupts (initial value)
1	Enables interrupts

[bit4] IRQF: Interrupt request flag

The IRQF bit will be set to “1” if the interrupt source selected by the interrupt source selection bits (bit3, bit2: IRS1, IRS0) is generated. In this case, an interrupt request will be generated to the CPU, if the interrupt request enable bit (bit5: IREN) has been enabled.

The IRQF bit is readable and writable. It is cleared only by writing “0”; therefore writing “1” to this bit does not change its value. In a read-modify-write (RMW) instruction, the read value is “1”, regardless of the bit value.

If “0” is written at the same time as the bit is set to “1”, the operation of writing “0” has the higher priority.

0	No interrupt request (initial value)
1	Interrupt request

[bit3,8 bit2] IRS1, IRS0: Interrupt source selection bits

These bits are used to select an interrupt source that will set IRQF (bit4). Do not attempt to rewrite these bits during operation.

IRS1	IRS0	Edge selection
0	0	Software trigger or valid trigger input (initial value)
0	1	Counter borrow (cycle match)
1	0	Normal polarity PPG ↑ or inverted polarity PPG ↓ (duty match)
1	1	Counter borrow, normal polarity PPG ↑, or inverted polarity PPG ↓

[bit1] POEN: PPG output enable bit

A PPG output is generated from the pin when this bit is set to “1”.

0	General-purpose port (initial value)
1	PPG output pin

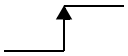
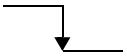
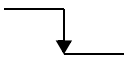
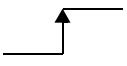
[bit0] OSEL: PPG output polarity specification bit

This bit is used to set the polarity for a PPG output.

0	Normal polarity (initial value)
1	Inverted polarity

The following options are available, when this bit is combined with PGMS (bit9).

PGMS	OSEL	PPG output
0	0	Normal polarity (initial value)
0	1	Inverted polarity
1	0	Output fixed to “L”
1	1	Output fixed to “H”

Polarity	Stop state	Duty match	Counter match
Normal polarity	"L" output		
Inverted polarity	"H" output		

PPG Down Counter Register (PDCR)

The PDCR register can be used to read the value of the 16-bit down counter.

Word access should be used for the PDCR register.

Upper PPG down counter register

Address: ch.0 007913_H
Address: ch.1 $00791B_H$
Address: ch.2 007923_H
Address: ch.3 $00792B_H$
Address: ch.4 007933_H
Address: ch.5 $00793B_H$
Address: ch.6 007943_H
Address: ch.7 $00794B_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
DC15	DC14	DC13	DC12	DC11	DC10	DC09	DC08	PDCRH0 to PDCRH7
R	R	R	R	R	R	R	R	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

Lower PPG down counter register

Address: ch.0 007912_H
Address: ch.1 $00791A_H$
Address: ch.2 007922_H
Address: ch.3 $00792A_H$
Address: ch.4 007932_H
Address: ch.5 $00793A_H$
Address: ch.6 007942_H
Address: ch.7 $00794A_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
DC07	DC06	DC05	DC04	DC03	DC02	DC01	DC00	PDCRL0 to PDCRL7
R	R	R	R	R	R	R	R	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

PPG Cycle Setup Register (PCSR)

PCSR is a register with a buffer to set a cycle. A transfer from the buffer is performed by either a counter borrow or startup.

Always perform a write operation on the duty setup register after writing to the cycle setup register, when initializing or rewriting the cycle setup register.

Word access should be used for the PCSR register. It can only be written, and the read value is undefined.

Upper PPG cycle setup register

Address: ch.0 007915_H
Address: ch.1 $00791D_H$
Address: ch.2 007925_H
Address: ch.3 $00792D_H$
Address: ch.4 007935_H
Address: ch.5 $00793D_H$
Address: ch.6 007945_H
Address: ch.7 $00794D_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
CS15	CS14	CS13	CS12	CS11	CS10	CS09	CS08	PCSRH0 to PCSRH7
W	W	W	W	W	W	W	W	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

Lower PPG cycle setup register

Address: ch.0 007914_H
Address: ch.1 $00791C_H$
Address: ch.2 007924_H
Address: ch.3 $00792C_H$
Address: ch.4 007934_H
Address: ch.5 $00793C_H$
Address: ch.6 007944_H
Address: ch.7 $00794C_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
CS07	CS06	CS05	CS04	CS03	CS02	CS01	CS00	PCSRL0 to PCSRL7
W	W	W	W	W	W	W	W	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	← Initial value

PPG Duty Setup Register (PDUT)

PDUT is a register with a buffer to set a duty. A transfer from the buffer is performed by either a counter borrow or startup.

When the cycle setup register and the duty setup register are set to the same value, all “H” is output during the normal polarity, and all “L” is output during the inverted polarity.

Do not set any value that will cause PCSR < PDUT. The PPG output is undefined. Word access should be used for the PDUT register. It can only be written and the read value is undefined.

Upper PPG duty setup register

Address: ch.0 007917H
Address: ch.1 00791FH
Address: ch.2 007927H
Address: ch.3 00792FH
Address: ch.4 007937H
Address: ch.5 00793FH
Address: ch.6 007947H
Address: ch.7 00794FH

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
DU15	DU14	DU13	DU12	DU11	DU10	DU09	DU08	PDUTH0 to PDUTH5
W	W	W	W	W	W	W	W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

Lower PPG duty setup register

Address: ch.0 007916H
Address: ch.1 00791EH
Address: ch.2 007926H
Address: ch.3 00792EH
Address: ch.4 007936H
Address: ch.5 00793EH
Address: ch.6 007946H
Address: ch.7 00794EH

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
DU07	DU06	DU05	DU04	DU03	DU02	DU01	DU00	PDUTL0 to PDUTL5
W	W	W	W	W	W	W	W	← Readable/Writable
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

PPG0 Output Division Setup Register (PPGDIV)

PPG0 output division setup register

Address: ch.0 007910H

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
-	-	-	-	-	-	DIV1	DIV0	PPGDIV
R	R	R	R	R	R	R/W	R/W	← Readable/Writable
(1)	(1)	(1)	(1)	(1)	(1)	(0)	(0)	← Initial value

[bit15 to bit10] Undefined bits

These are undefined bits.

Writing to the bits has no effect on operation.

The bits always reads “1”.

[bit9, bit8] DIV1, DIV0: Division setup bits

These bits are used to set a division ratio for the PPG ch.0 output.

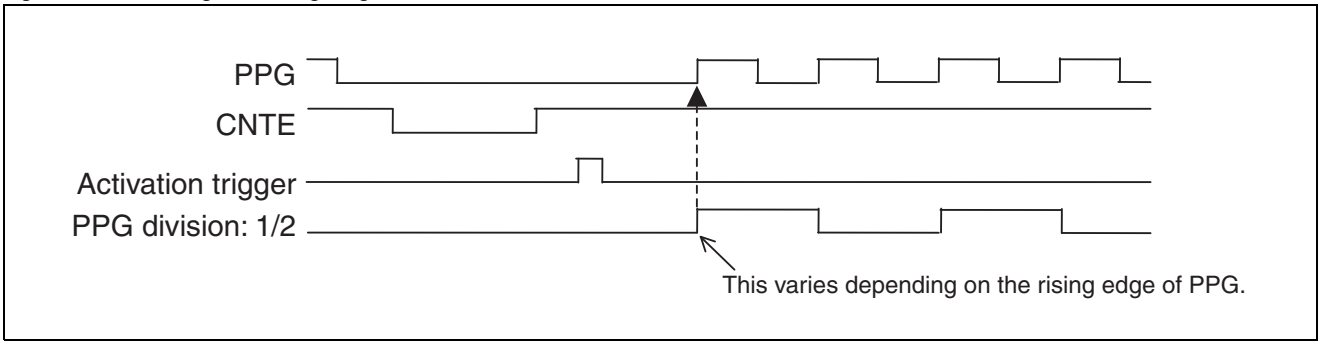
CS1	CS0	Division ratio
0	0	1/1 (initial value)
0	1	1/2
1	0	1/4
1	1	1/8

Notes:

The following limitations are applicable when the division ratio is set to 1/2, 1/4 or 1/8.

- The output waveform is fixed to a duty ratio of 50%.
- It is not allowed to set one-shot operation (PCNT:MDSE=1).
- It is not allowed to set the PPG output inversion function (PCNT:OSEL=1).
- It is not allowed to set the fixed PPG output status (PCNT:PGMS,OSEL=01_B,10_B,11_B).
- It is not allowed to perform setup when PCSR=PDUT.

Figure 14-3. Timing for Rising Edge of PPG Division



Note:

The PPG division varies depending on the timing for the rising edge of PPG.

PPG Multi-Channel Simultaneous Startup Register (PMSSR)

PPG multi-channel simultaneous startup register

Address: 00799B$_H$

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
	MT7	MT6	MT5	MT4	MT3	MT2	MT1	MT0	PMSSR
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Readable/Writable
	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	← Initial value

[bit7 to bit0] MT7 to MT0: Multi-channel simultaneous activation trigger bits

The MT7 to MT0 bits are activation trigger bits corresponding to PPG ch.7 to ch.0, respectively.

“1” can be written to each MT bit to simultaneously start up PPGs for the channels in which the MTEN and CNTE bits in the PCNT register are both set to “1”. The PPGs can be restarted by writing “1” to the MT bit during simultaneous startup, if the RTRG bit in the PCNT register is set to “1”.

Writing “0” to the MT bit has no effect at any time.

The bit always reads “0”.

Note:

A simultaneous stop function using software after simultaneous startup is not available for the PPG multi-channels. Therefore, “0” must be written to the CNTE bit of each channel to stop the operation.

14.4 Interrupts of PPG Timer

The PPG timer can generate an interrupt request by a startup of the timer, a counter borrow (cycle match) or a duty match. It also supports the extended intelligent I/O service (EI^{2}OS).

Interrupts of PPG Timer

Table 14-1 shows the interrupt control bits and interrupt sources of the PPG timer.

Table 14-1. Interrupt Control Bits and Interrupt Sources of PPG Timer

Interrupt source	Lower PPG control status register (PCNTL5 to PCNTL0)		
	Interrupt flag bit	Interrupt enable bit	Clearing of interrupt flag
■ Timer startup ■ Counter borrow (cycle match) ■ Duty match	IRQF	IREN	■ Writing “0” to IRQF bit ■ Clearing by EI^{2}OS ■ Reset

In the PPG timer, the interrupt flag bit is set to “1” by any of the interrupt sources listed in Table 14-1. In this case, an interrupt request will be output to the interrupt controller if the interrupt enable flag has been set to “1”.

PPG Timer Interrupts and EI^{2}OS

Table 14-2 shows the correspondence among the interrupt of the PPG timer, EI^{2}OS and μDMAC.

Table 14-2. Correspondence Among PPG Timer Interrupt, EI^{2}OS and μDMAC

Channel	Interrupt number	Interrupt control register		Vector table address			EI^{2}OS	μDMAC channel
		Register name	Address	Lower	Upper	Bank		
PPG timer 0	#19(13_H)	ICR04	$0000B4_H$	$FFFFB0_H$	$FFFFB1_H$	$FFFFB2_H$	❍	5
PPG timer 4								
PPG timer 1	#20(14_H)	ICR04	$0000B4_H$	$FFFFAC_H$	$FFFFAD_H$	$FFFFAE_H$	❍	6
PPG timer 5								
PPG timer 2	#21(15_H)	ICR05	$0000B5_H$	$FFFFA8_H$	$FFFFA9_H$	$FFFFAA_H$	❍	7
PPG timer 6								
PPG timer 3	#22(16_H)	ICR05	$0000B5_H$	$FFFFA4_H$	$FFFFA5_H$	$FFFFA6_H$	❍	8
PPG timer 7								

❍: Available

EI^{2}OS Function of PPG Timer

The PPG timer has a circuit supporting the EI^{2}OS function. Therefore, EI^{2}OS can be activated when the timer starts up, or a counter borrow (cycle match) or duty match occurs. Note, however, that in ICR06 and ICR07, EI^{2}OS is available only when no interrupt is used by any other peripheral functions that share the interrupt control register (ICR) or interrupt vector. For example, to use EI^{2}OS in PPG timer 0, it is necessary to disable as well as interrupts of the PPG timer 1.

14.5 Operational Description of PPG Timer

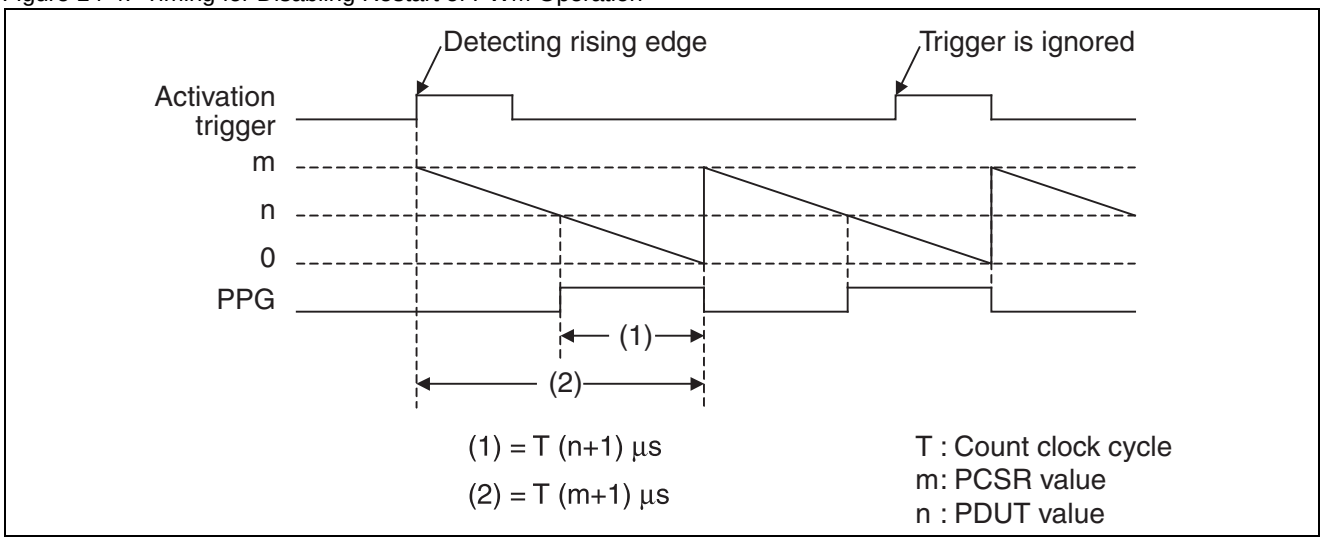
This section describes the operations of the PPG timer.

PWM Operation

PWM operation allows the user to generate pulses continuously once an activation trigger is detected. The output pulse cycle can be controlled by modifying the PCSR value. Similarly, the duty ratio can be controlled by modifying the PDUT value.

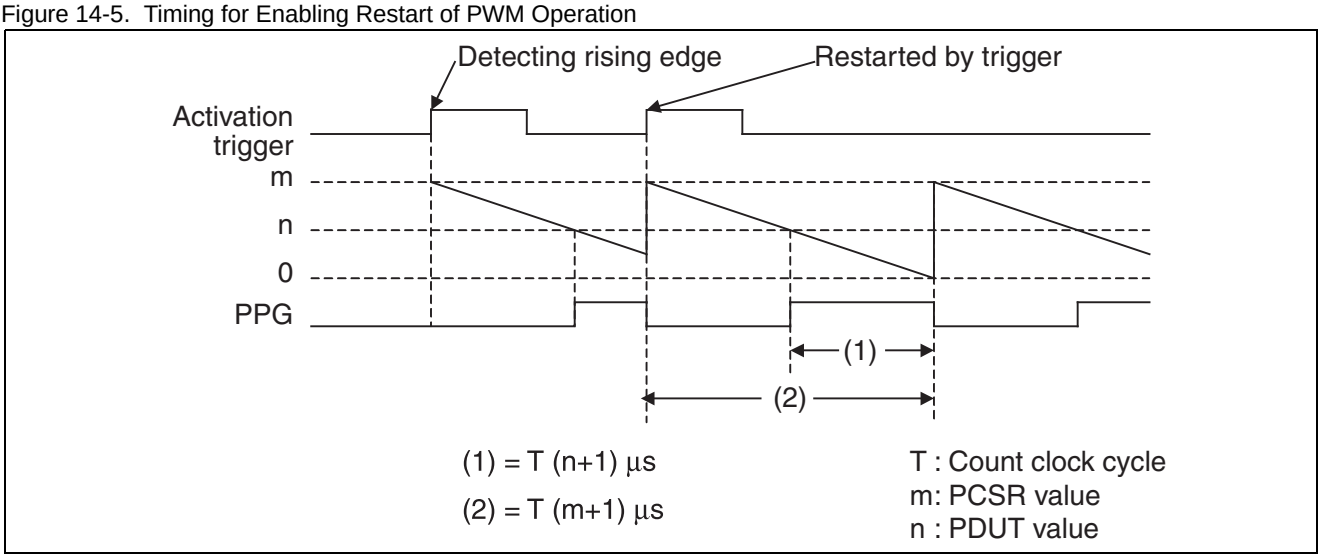
When Restart is Disabled

Figure 14-4. Timing for Disabling Restart of PWM Operation



When Restart is Enabled

Figure 14-5. Timing for Enabling Restart of PWM Operation



Notes:

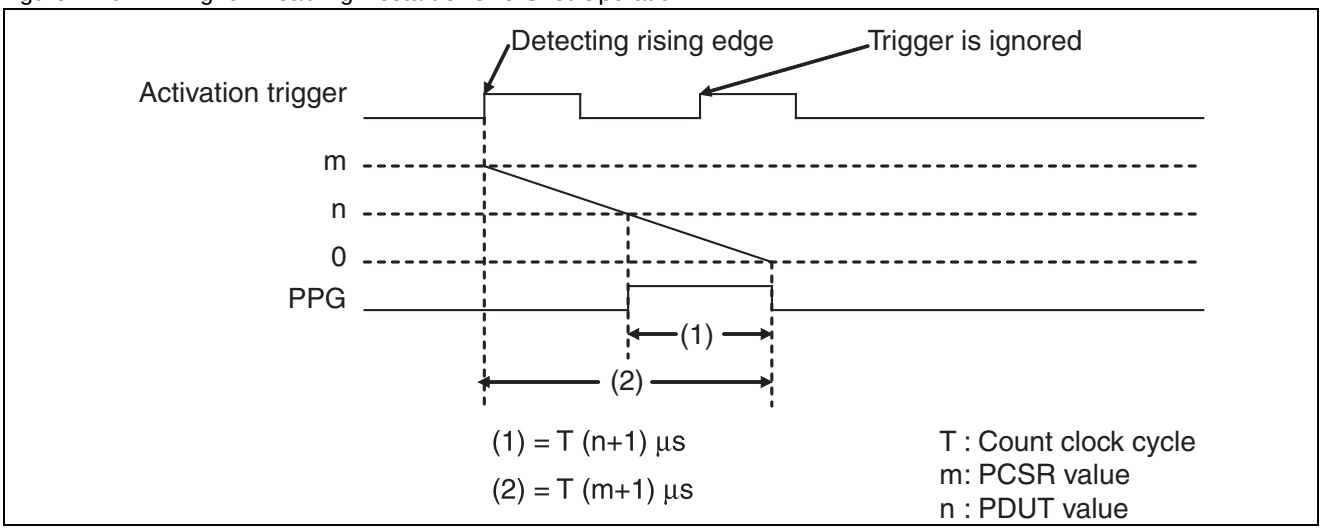
- Always write to PDUT after writing data to PCSR.
- For the PPG timer output to provide 100% duty, switch to a general-purpose port.

One-Shot Operation

One-shot operation allows the user to output a single pulse with any width using a trigger. When restart is enabled, the counter is reloaded once an activation trigger is detected during operation.

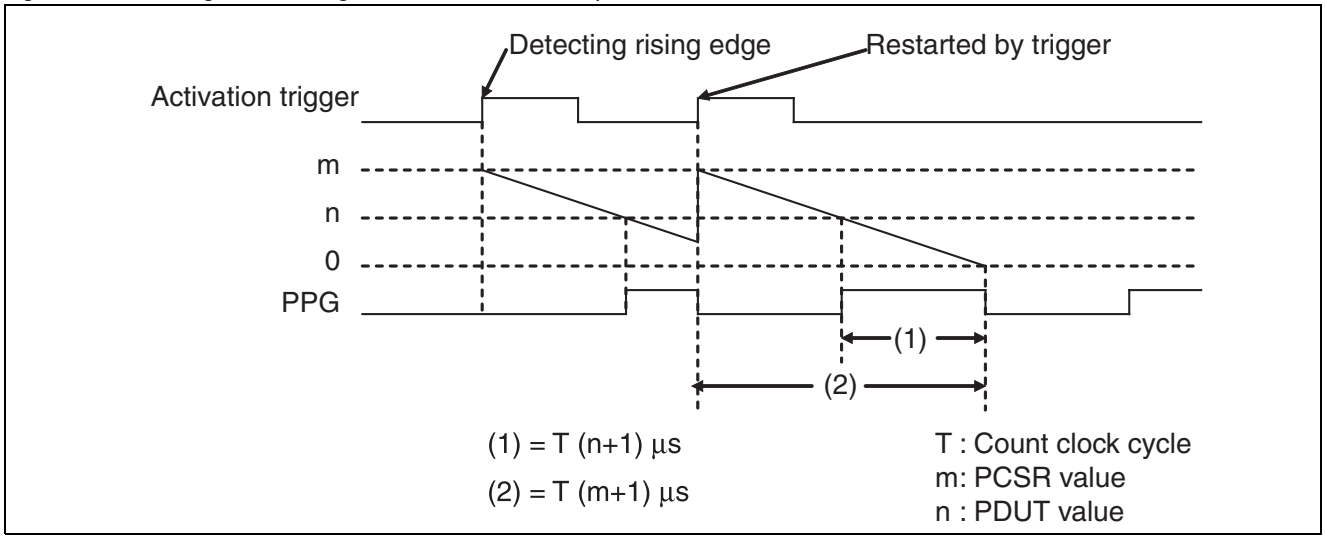
When Restart is Disabled

Figure 14-6. Timing for Disabling Restart of One-Shot Operation



When Restart is Enabled

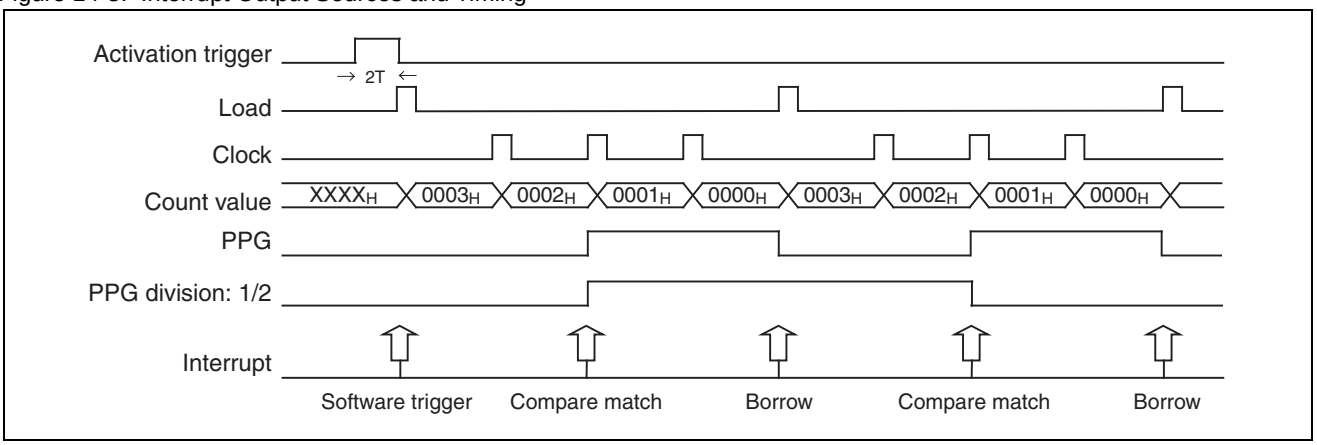
Figure 14-7. Timing for Enabling Restart of One-Shot Operation



Interrupt Sources and Timing

It takes 2T (T: system clock cycle) from when a activation trigger is applied and until the count value is loaded.

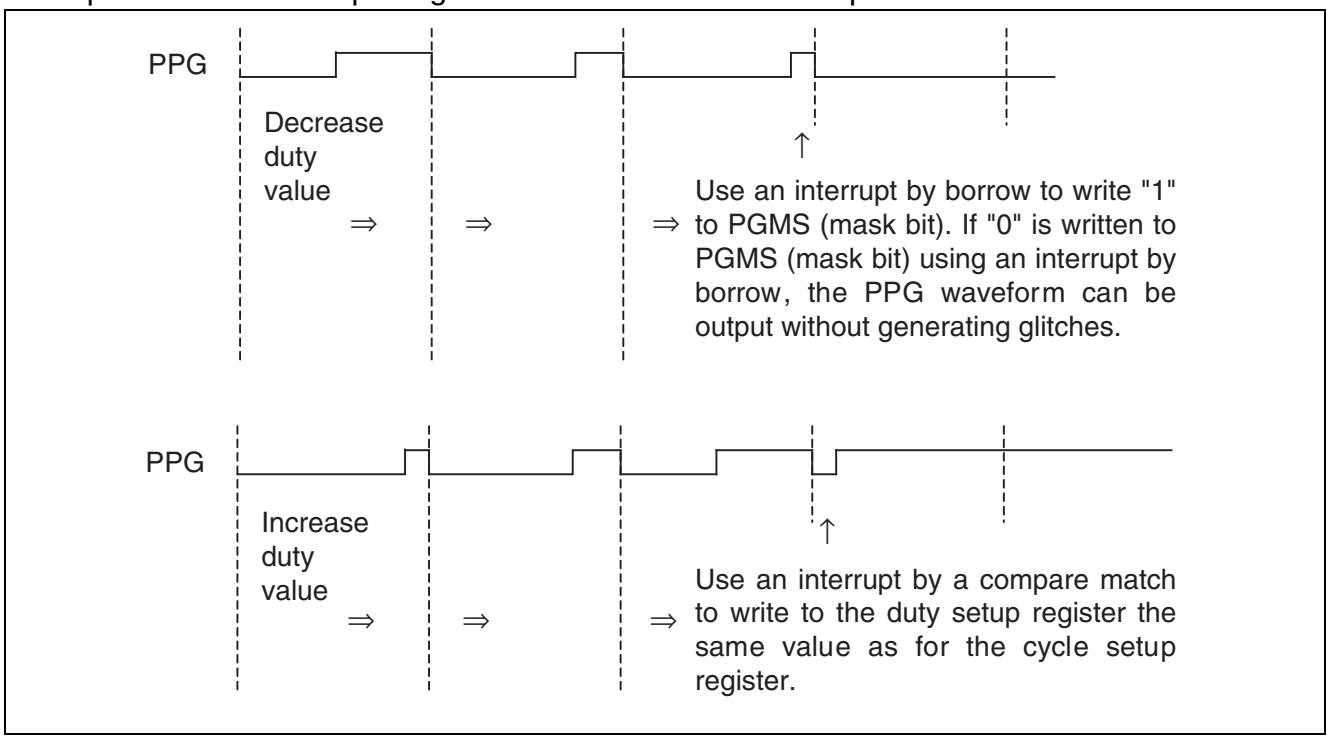
Figure 14-8. Interrupt Output Sources and Timing



Note:

An interrupt is generated at the timing of 1/1 division when 1/2, 1/4 or 1/8 division is used.

Example Method for Outputting all “L” or all “H” in PWM Output



15. Base Timer

This chapter describes the overview of base timer, configuration and function of its registers and operations of the base timer.

15.1 Overview of Base Timer

15.2 Block Diagram of Base Timer

15.3 Registers of Base Timer

15.4 Operations of Base Timer

15.5 Operation of 32-Bit Mode

15.6 Precautions when Using Base Timer

15.7 Interrupts of Base Timer

15.8 Functional Description of Base Timer

15.1 Overview of Base Timer

Only one of the following timer functions can be selected for the base timer by setting FMD2, FMD1 and FMD0 bits in the timer control register: 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section outlines the selectable timer functions.

Relationship between mode settings and timer functions

FMD2, FMD1, FMD0 bit setting	Function
000_B	Reset mode
001_B	16-bit PWM timer
010_B	16-bit PPG timer
011_B	16/32-bit reload timer
100_B	16/32-bit PWC timer

Reset mode

The macro for the base timer shall be in a reset state (each register set to its initial value) when this mode is selected. Select this mode before switching to a different timer function or T32-bit setting. Note, however, that a timer function or T32-bit setting can be selected without setting this mode after a reset,

16-bit PWM timer

This timer consists of a 16-bit down counter, 16-bit data register with a buffer for setting a cycle, 16-bit compare register with a buffer for setting a duty, and a pin control unit.

Cycle and duty-related data can be rewritten during timer operation, as the data is stored in the corresponding register with a buffer.

The count clock for the 16-bit down counter can be selected from 5 internal clocks (1/4/16/128/256 divisions of machine clock).

There are two selectable modes: one-shot mode in which a count is stopped when an underflow occurs, and continuous mode in which a count is repeated by reloading.

A software trigger is used for sartup of the counter.

16-bit PPG timer

This timer consists of a 16-bit down counter, 16-bit data register for setting the "H" width, 16-bit data register for setting the "L" width, and a pin control unit.

The count clock for the 16-bit down counter can be selected from 5 internal clocks (1/4/16/128/256 divisions of machine clock).

There are two selectable modes: one-shot mode in which a count is stopped when an underflow occurs, and continuous mode in which a count is repeated by reloading.

A software trigger is used for sartup of the counter.

16/32-bit reload timer

This timer consists of a 16-bit down counter, 16-bit reload register, and a pin control unit.

The count clock for the 16-bit down counter can be selected from 5 internal clocks (1/4/16/128/256 divisions of machine clock) and 3 external events (detection of rising edge, falling edge, and both edges).

There are two selectable modes: one-shot mode in which a count is stopped when an underflow occurs; and continuous mode in which a count is repeated by reloading.

A software trigger or one of the three external events (detection of rising edge, falling edge, or both edges) can be selected for startup of the counter.

16/32-bit PWC timer

This timer consists of a 16-bit up counter, input pins for measurement, and control registers.

An external pulse is entered to measure the time between any events.

The count clock to be used as the standard clock can be selected from 5 internal clocks (1/4/16/128/256 divisions of machine clock).

Each measurement mode: “H” pulse width (↑ to ↓) / “L” pulse width (↓ to ↑)
Rising edge cycle (↑ to ↑) / Falling edge cycle (↓ to ↓)
Measurement between edges (↑ or ↓ to ↓ or ↑)

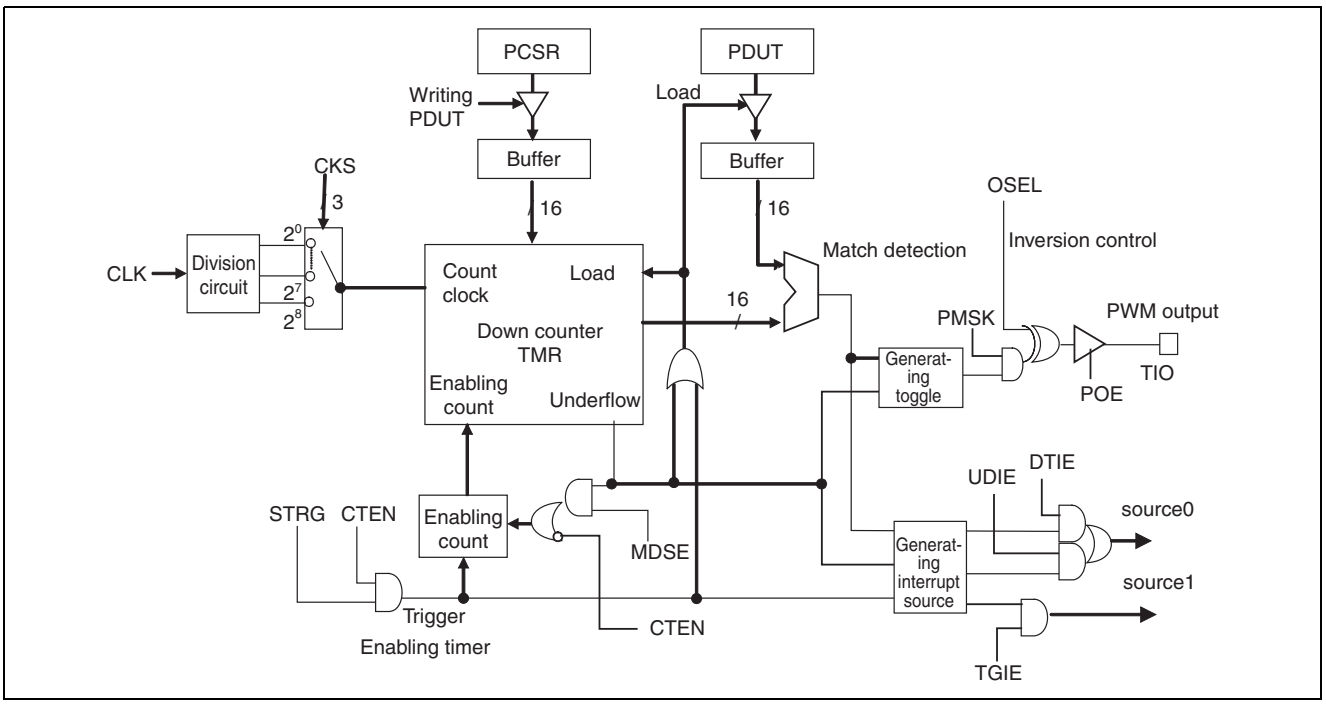
An interrupt request can be generated upon the completion of measurement.

Single measurement or continuous measurement can be selected.

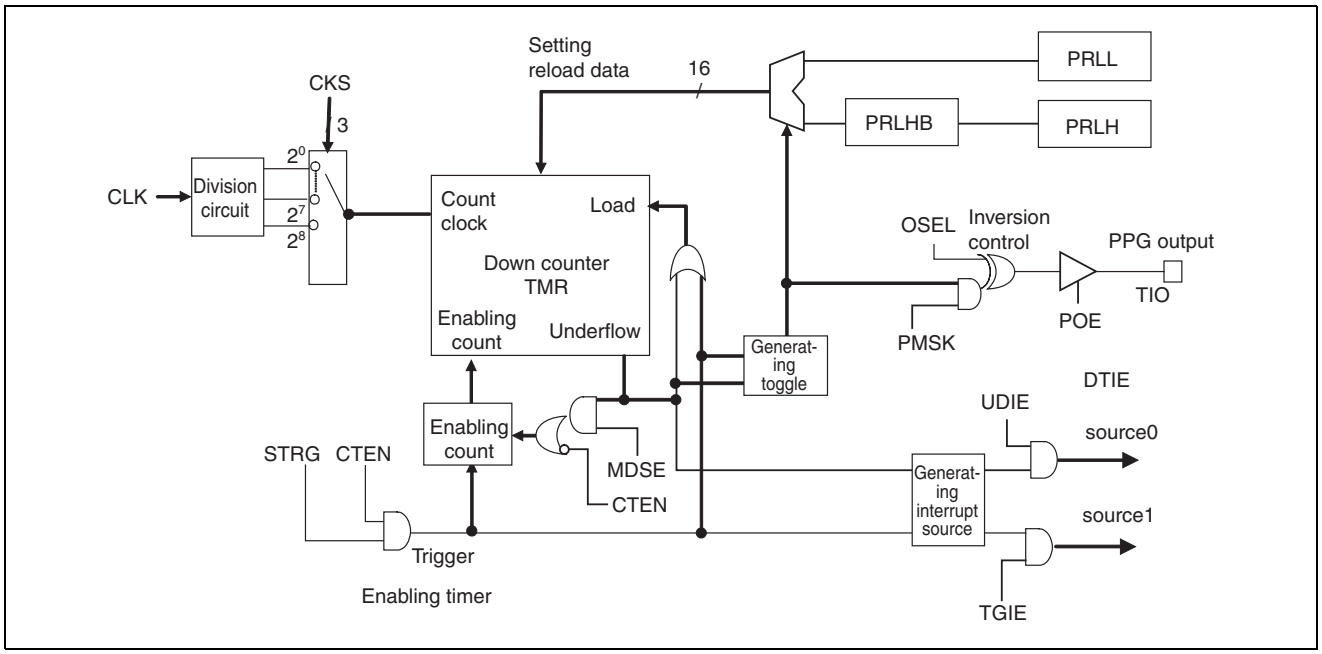
15.2 Block Diagram of Base Timer

This section shows a block diagram for each base timer mode.

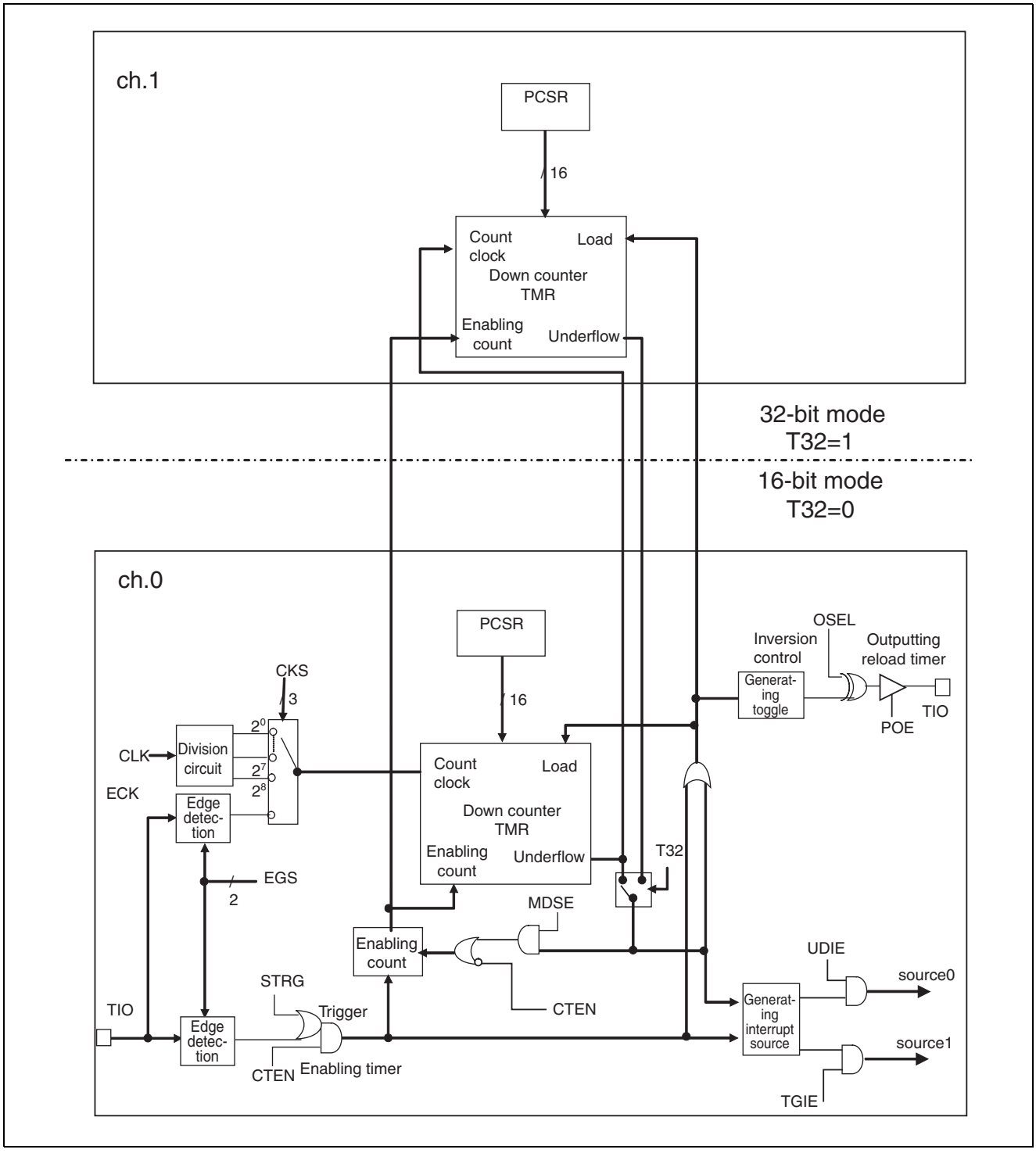
Block Diagram of 16-Bit PWM Timer



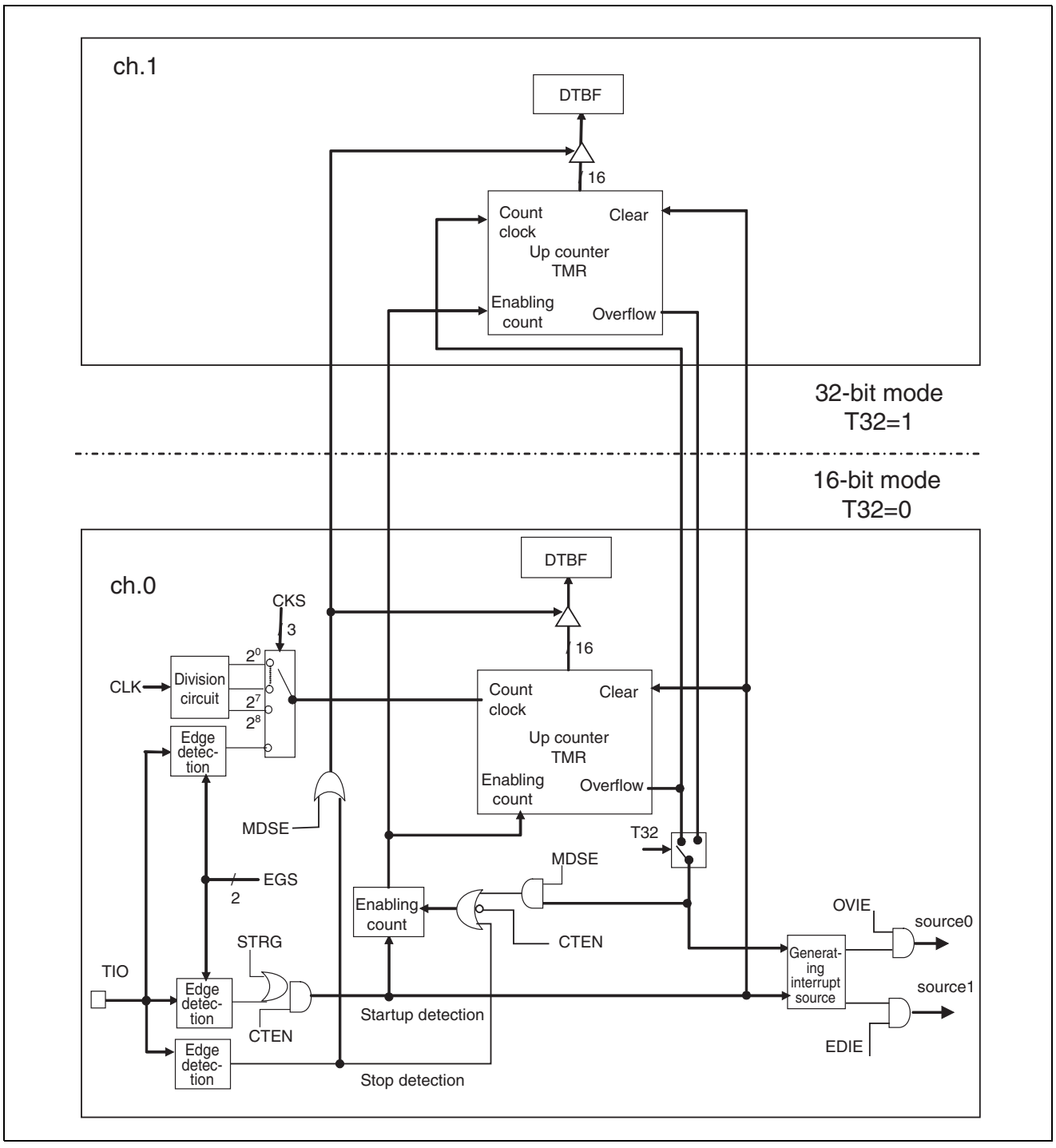
Block Diagram of 16-Bit PPG Timer



Block Diagram of 16/32-Bit Reload Timer (ch.1, ch.0)



Block Diagram of 16/32-Bit PWC (ch.1, ch.0)



15.3 Registers of Base Timer

This section lists the registers of the base timer and the bit configuration of each mode.

List of Base Timer Registers

Mode setting (FMD2, FMD1, FMD0)	Address		bit15bit8	bit7bit0
All modes	ch.0:007953$_H$ ch.1:00795d$_H$ ch.2:007967$_H$ ch.3:007971$_H$	ch.0:007952$_H$ ch.1:00795C$_H$ ch.2:007966$_H$ ch.3:007970$_H$	TMCR (Timer control register)	
All modes	-	ch.0:007954$_H$ ch.1:00795E$_H$ ch.2:007968$_H$ ch.3:007972$_H$	-	STC (Status control register)
001_B/010_B/011_B	ch.0:007957$_H$ ch.1:007961$_H$ ch.2:00796b$_H$ ch.3:007975$_H$	ch.0:007956$_H$ ch.1:007960$_H$ ch.2:00796A$_H$ ch.3:007974$_H$	TMR (Timer register)	
100_B			-	
001_B/011_B	ch.0:007959$_H$ ch.1:007963$_H$ ch.2:00796d$_H$ ch.3:007977$_H$	ch.0:007958$_H$ ch.1:007962$_H$ ch.2:00796C$_H$ ch.3:007976$_H$	PCSR (Cycle setup register)	
010_B			PRLL ("L" width setup reload register)	
100_B			-	
001_B	ch.0:00795b$_H$ ch.1:007965$_H$ ch.2:00796f$_H$ ch.3:007979$_H$	ch.0:00795A$_H$ ch.1:007964$_H$ ch.2:00796E$_H$ ch.3:007978$_H$	PDUT (Duty setup register)	
010_B			PRLH ("H" width setup reload register)	
011_B			-	
100_B			DTBF (Data buffer register)	

List for Bit Configuration of Each Mode

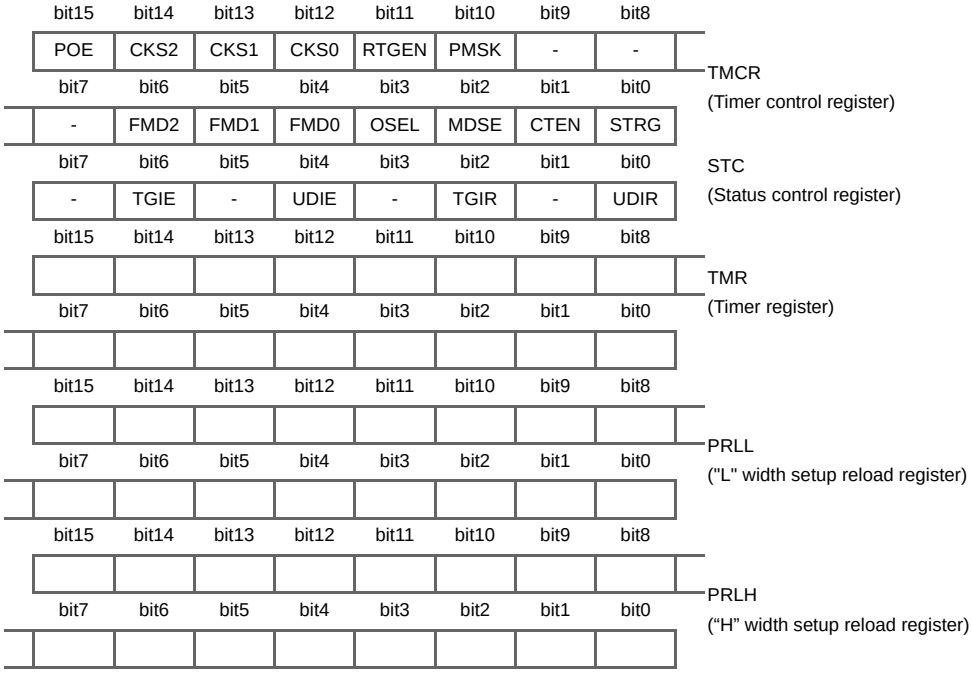
List of registers for when 16-bit PWM timer is selected

Mode setting FMD=001_B

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
POE	CKS2	CKS1	CKS0	RTGEN	PMSK	-	-	TMCR (Timer control register)
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
-	FMD2	FMD1	FMD0	OSEL	MDSE	CTEN	STRG	
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	STC (Status control register)
-	TGIE	DTIE	UDIE	-	TGIR	DTIR	UDIR	
bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
								TMR (Timer register)
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
								PCSR (Cycle setup register)
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
								PDUT (Duty setup register)
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

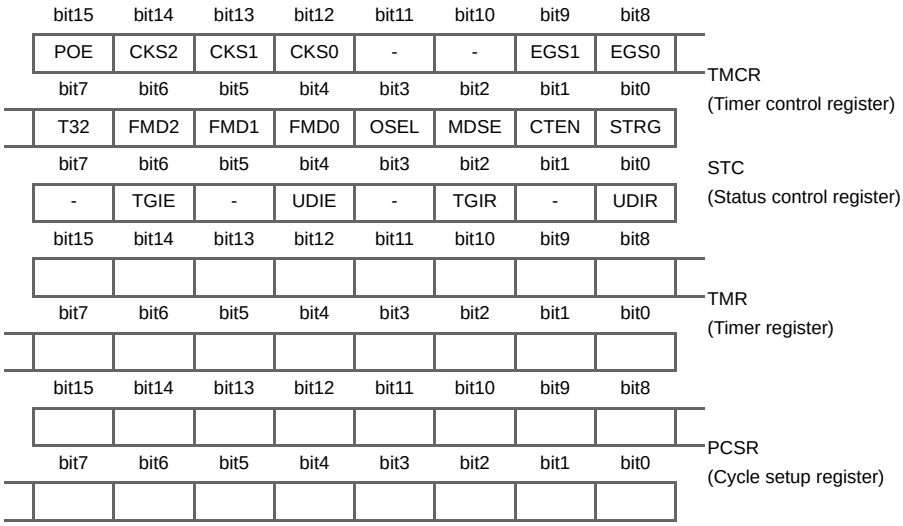
List of registers for when 16-bit PPG timer is selected

Mode setting FMD=010_B



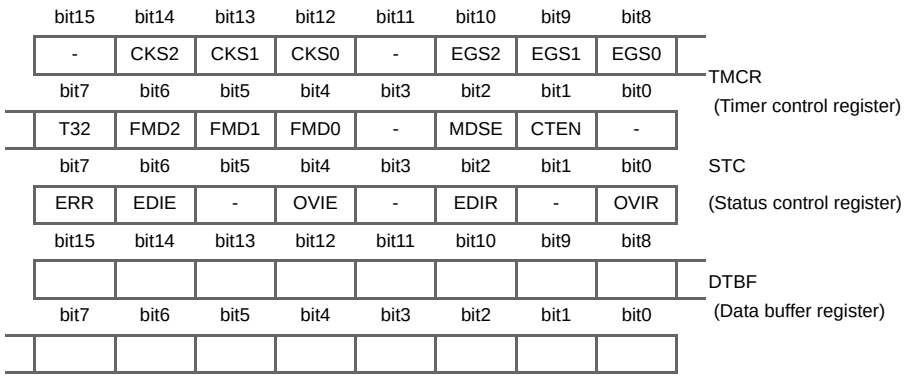
List of registers for when reload timer is selected

Mode setting FMD=011_B



List of registers for when PWC is selected

Mode setting FMD=100_B



15.4 Operations of Base Timer

This section describes the operations of the base timer.

Operation of Base Timer

Reset mode

The macro for the base timer shall be in a reset state (each register set to its initial value) when this mode is selected. Select this mode before switching to a different timer function or T32-bit setting. Note, however, that a timer function or T32-bit setting can be selected without setting this mode after a reset. If this mode is set in an even-numbered channel when selecting the 32-bit mode, the odd-numbered channel will also be reset at the same time. Therefore, the odd-numbered channel is not required to be set to reset mode.

16-bit PWM timer

The 16-bit PWM timer starts counting down the value of the cycle set by the activation trigger. In this case, set the output to the “L” level first. If the value of the 16-bit down counter matches the value set in the duty setup register, it will be inverted to the “H” level and then output. When an underflow eventually occurs in the counter, it will be inverted again back to the “L” level. As a result, a waveform with any cycle and duty will be generated.

16-bit PPG timer

The 16-bit PPG timer continues to count down until it reaches the value set in the “L” width setup reload register by trigger activation. First, set the output to the “L” level, and invert the output to the “H” level upon the occurrence of an underflow. Then, the timer will start counting down until it reaches the value set in the “H” width setup reload register, and invert the output to the “L” level upon the occurrence of the next underflow. As a result, a waveform with any “L” and “H” widths is generated.

16-bit reload timer

The 16-bit reload timer starts counting down until it reaches the value of the cycle set by trigger activation. The interrupt flag is set when an underflow occurs in the 16-bit down counter. Depending on the MDSE bit setting, the output level is either the toggle output that is inverted every time an underflow occurs, or the pulse output that is set to “H” at the beginning of a count and changes to “L” upon the occurrence of an underflow.

32-bit reload timer

In basic operation, this timer operates in the same way as the 16-bit reload timer. It, however, operates as a 32-bit reload timer when two channels (even- and odd-numbered channels) are used. In this case, the even-numbered channel operates the lower 16-bit timer, while the odd-numbered channel operates the upper 16-bit timer. Interrupts and output waveforms are controlled according to the settings of the even-numbered channel. To set a cycle, write to the upper register (odd-numbered channel) before writing to the lower register (even-numbered channel).

To read the timer value, read from the lower register (even-numbered channel) before reading from the upper register (odd-numbered channel).

16-bit PWC timer

The PWC timer activates the 16-bit up counter when a specified measurement start edge is entered, and it stops the counter when a measurement stop edge is detected. The count value achieved during this period is stored as the pulse width into the data buffer register.

32-bit PWC timer

In basic operation, this timer operates in the same way as the 16-bit PWC timer. It, however, operates as a 32-bit PWC timer when two channels (even- and odd-numbered channels) are used. In this case, the even-numbered channel performs the lower 16-bit portion of a count, while the odd-numbered channel performs the upper 16-bit portion of the count. Interrupts are controlled according to the settings of the even-numbered channel. To read the measurement or count value, read from the lower register (even-numbered channel) before reading from the upper register (odd-numbered channel).

15.5 Operation of 32-Bit Mode

The reload timer and PWC can perform a 32-bit mode operation, using two channels. This section describes the basic functions and operation of the 32-bit mode function.

32-bit mode function

In this function, two channels of the base timer are combined together to serve as a reload timer for 32-bit data, or to perform PWC operation using 32-bit data. The timer counter value can also be read during operation, as the counter value of the upper 16-bit timer (odd-numbered channel) is buffered when the counter value of the lower 16-bit timer (even-numbered channel) is read.

32-bit mode setting

Set FMD2, FMD1 and FMD0 bits in the TMCR register of the even-numbered channel to “000_B” to select reset mode first, and then select either the reload timer or PWC timer option to set the operation, like in the 16-bit mode. At this point, write “1” to the T32 bit in the TMCR register to select the 32-bit operation mode. Make sure to keep the T32 bit of the odd-numbered channel set to "0". It is also not required to set the reset mode. Next, if the reload timer option is to be selected, set the upper 16 bits of the 32-bit reload value in the cycle setup register of the odd-numbered channel, and then set the lower 16 bits of the reload value in the cycle setup register of the even-numbered channel.

Change any setting of both the channels, if necessary, when a count is in a stop state, as a transfer to the 32-bit operation mode is reflected immediately after data is written to the T32 bit.

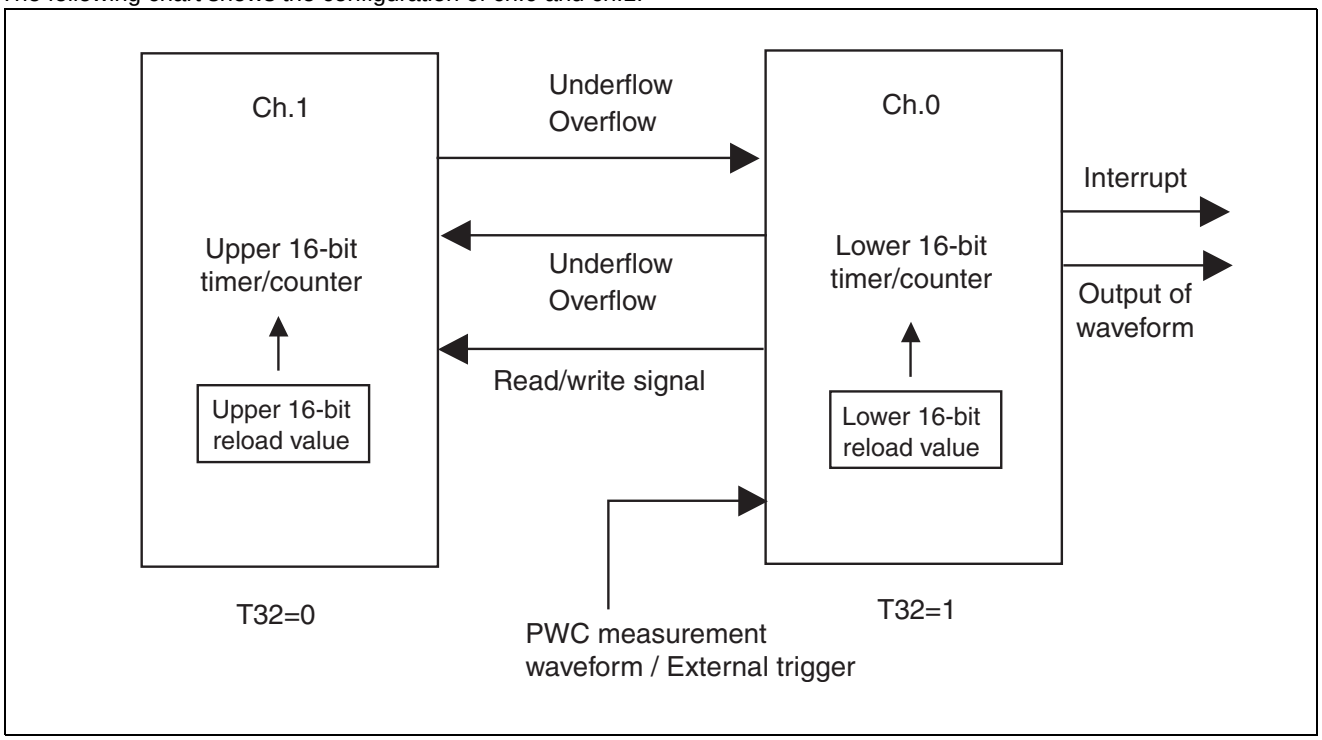
To transfer from the 32-bit mode to 16-bit mode, set the FMD2, FMD1 and FMD0 bits in the TMCR register of the even-numbered channel to “000_B” to select the reset mode, reset the status of both even- and odd-numbered channels, and then set the channels individually for 16-bit mode.

Operation of 32-bit mode

If the reload timer or PWC is activated by controlling the even-numbered channel after setting the 32-bit mode, the timer/counter of the even-numbered channel will perform the lower 16-bit operation, while the timer/counter of the odd-numbered channel will perform the upper 16-bit operation

As the 32-bit mode operates according to the settings of the even-numbered channel, the settings of the odd-numbered channel (except the cycle setup register in reload timer mode) are ignored. The settings of the even-numbered channel are also effective for timer activation, waveform output, and interrupt signals. (The odd-numbered channel is masked to “L”.)

The following chart shows the configuration of ch.0 and ch.1.



15.6 Precautions when Using Base Timer

This section describes precautions to be taken when using the base timer.

Common Precautions for Use of Timers

Precautions for setup by program

- The following bits in the TMCR register must not be rewritten during operation. They should always be rewritten either before startup or after stop.

  [bit14, bit13, bit12] CKS2, CKS1, CKS0: Clock selection bits
  [bit10, bit9, bit8] EGS2, EGS1, EGS0: Measurement edge selection bits
  [bit7] T32: 32-bit timer selection bit
  (when the reload timer or PWC function is selected)
  [bit6, bit5, bit4] FMD2, FMD1, FMD0: Timer function selection bits
  [bit2] MDSE: Measurement mode (single/continuous) selection bit

- All the registers of the base timer are initialized when the FMD2, FMD1 and FMD0 bits in the TMCR register are set to "000_B" to select reset mode. Therefore, all the registers must be set again.
- When the FMD2, FMD1 and FMD0 bits in the TMCR register are set to "000_B" to select reset mode, the settings of all the other bits are ignored and initialized.

Precautions when Using 16-Bit PWM/PPG/Reload Timer

Precautions for setup by program

- When the interrupt request flag is set at the same time as it is cleared, the flag setting operation has a higher priority; therefore the clearing operation is invalid.
- When the down counter is attempted to load and count at the same time, the loading has a higher priority.
- Set the timer function by the FMD2, FMD1 and FMD0 bits in the TMCR register before setting a cycle, duty, "H" width, and "L" width.
- In one-shot mode, the timer reloads the count value to restart when restart is detected upon the completion of a count.

Precautions when Using PWC Timer

Precautions for setup by program

- The counter is cleared when “1” is written to the activation enable bit (CTEN). Consequently, the data that existed in the counter before the activation is enabled becomes invalid.
- When PWC mode is set (FMD=100_B) at the same time as measurement start is set (CTEN=1) in system reset or reset mode, the timer may operate depending on the status of the measurement signal immediately before the event.
- When a measurement start edge is detected at the same time as restart is set in continuous measurement mode, a count immediately starts from “0001_H”.
- When restarting a count operation after it has started, the following may occur depending on the count timing.
  When a count restarts at the same time as a measurement stop edge in pulse width single measurement mode:

  While the count restarts and waits for a measurement start edge, the measurement stop flag (EDIR) is set.

  When a count restarts at the same time as a measurement stop edge in pulse width continuous measurement mode:

  While the count restarts and waits for a measurement start edge, the measurement stop flag (EDIR) is set and the current measurement result is transferred to DTBF.

  Control interrupts with special attention to the flag operation when restarting during operation, as described above.

Precautions when Using I/O Pins

The I/O pins of the base timer can be used as only either input or output for each channel.

- When used as input pin
  Set the DDR register for the corresponding pin to “0” before use. Also set the POE bit in the TMCR register to “0”.
- When used as output pin
  Set the DDR register for the corresponding pin to “1”, before use the trigger input edge selection bits (EGS1 and EGS0 bits in the TMCR register) to “00_B”, and the POE bit in the TMCR register.

15.7 Interrupts of Base Timer

The following table lists interrupt request flags, interrupt enable bits and interrupt sources for each base timer function.

Interrupt Control Bits and Interrupt Sources for Each Function

Table 15-1 shows the interrupt control bits and interrupt sources for each function.

Table 15-1. Interrupt Control Bits and Interrupt Sources for Each Mode

	Status control register (STC)			
	Interrupt request flag bit	**Interrupt request enable bit**	**Interrupt source**	**IRQ**
PWM timer function	UDIR: bit0	UDIE: bit4	Detection of underflow	IRQ0
	DTIR: bit1	DTIE: bit5	Detection of duty match	
	TGIR: bit2	TGIE: bit6	Detection of timer activation trigger	IRQ1
PPG timer function	UDIR: bit0	UDIE: bit4	Detection of underflow	IRQ0
	TGIR: bit2	TGIE: bit6	Detection of timer activation trigger	IRQ1
Reload timer function	UDIR: bit0	UDIE: bit4	Detection of underflow	IRQ0
	TGIR: bit2	TGIE: bit6	Detection of timer activation trigger	IRQ1
PWC timer function	OVIR: bit0	OVIE: bit4	Detection of overflow	IRQ0
	EDIR: bit2	EDIE: bit6	Detection of measurement completion	IRQ1

15.8 Functional Description of Base Timer

This section describes each functions of base timer.

Functions of Base Timer

- PWM timer function
- PPG timer function
- Reload timer function
- PWC timer function

15.8.1 PWM Timer Function

Only one of the following timer functions can be selected for the base timer by setting FMD2, FMD1 and FMD0 bits in the timer control register: 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section describes the timer function when PWM is selected.

Timer Control Register (TMCR) when PWM Timer is Selected

The timer control register (TMCR) is used to control the PWM timer. Take care that some bits cannot be rewritten during the operation of the PWM timer.

Figure 15-1. Timer Control Register (Upper Byte of TMCR)

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
POE	CKS2	CKS1	CKS0	RTGEN	PMSK	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: ch.0:007953H
ch.1:00795DH
ch.2:007967H
ch.3:007975H
Initial value:
00000000B (at reset)

PMSK	Pulse output mask bit
0	Normal output
1	Fixed at "L" output

RTGEN	Restart enable bit
0	Disables restart
1	Enables restart

CKS2	CKS1	CKS0	Count clock selection bits
0	0	0	ϕ
0	0	1	ϕ /4
0	1	0	ϕ /16
0	1	1	ϕ /128
1	0	0	ϕ /256
1	0	1	External clock (rising edge event)
1	1	0	External clock (falling edge event)
1	1	1	External clock (both edge event)

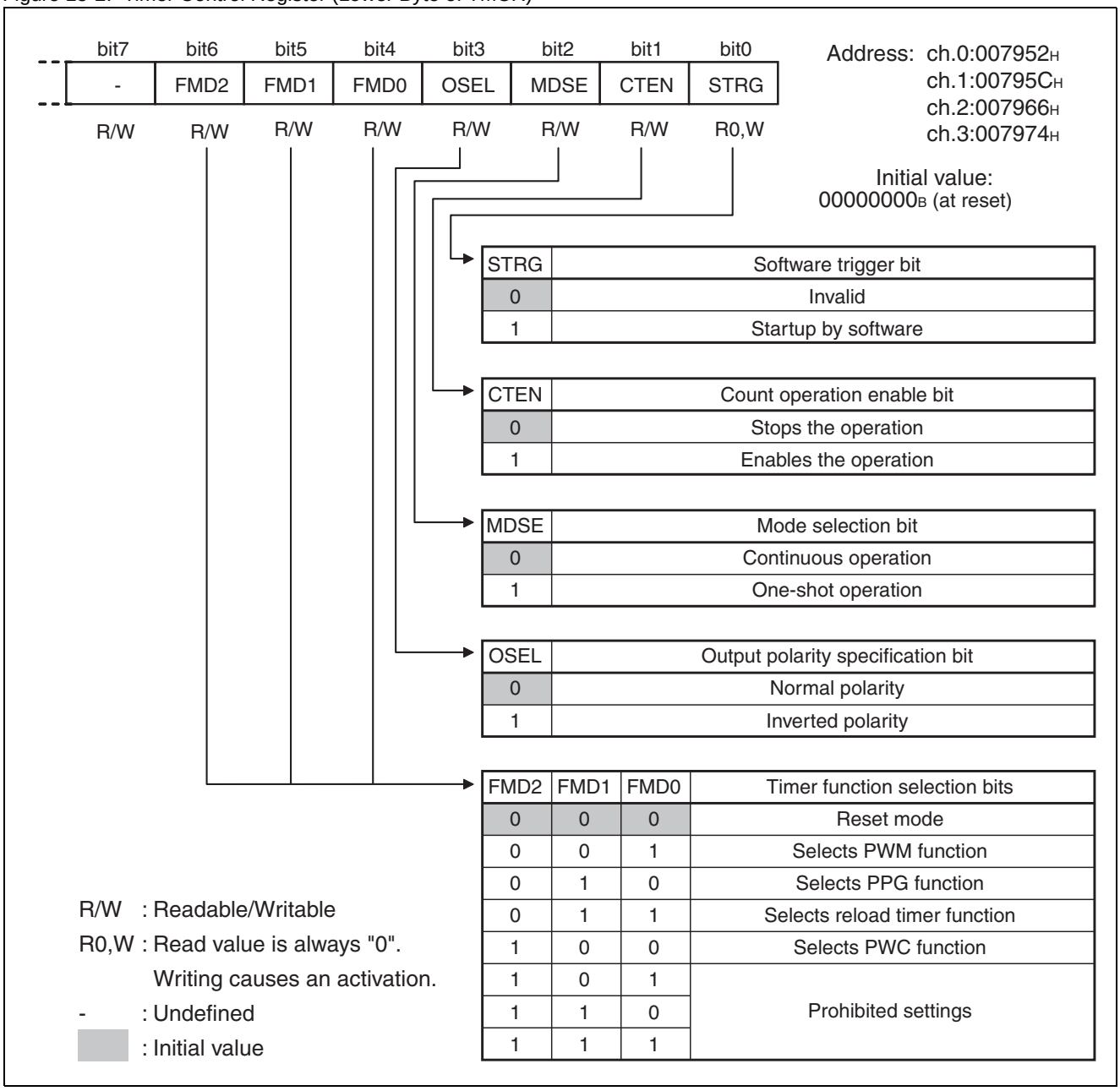
POE	Port output enable bit
0	Disables output
1	Enables output

R/W : Readable/Writable
: Initial value

Table 15-2. Timer Control Register (Upper Byte of TMCR)

Bit name		Function
bit15	POE: Port output enable bit	■ This bit is used to control the port output of PWM output waveform. ■ The TIO pin serves as a general-purpose port when this bit is set to “0”, and the same pin functions as an output pin for the PWM waveform when the bit is set to “1”.
bit14 to bit12	CKS2 to CKS0: Count clock selection bits	■ These bits are used to select a count clock for the 16-bit down counter. ■ Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS2 to CKS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.
bit11	RTGEN: Restart enable bit	This bit is used to enable restart by the input of a software trigger or a trigger.
bit10	PMSK: Pulse output mask bit	■ This bit is used to control the level of the PWM output waveform. ■ When this bit is set to “0”, the PWM waveform is output in its original form. ■ When this bit is set to “1”, PWM output is masked to “L” output, regardless of the value set as for the cycle or duty. **Note:** The output will be masked to “H” if PMSK is set to “1” when OSEL (bit3) has been set to invert the output.
bit9, bit8	Unused bits	■ The read value is “0”. ■ Write “0” to these bits.

Figure 15-2. Timer Control Register (Lower Byte of TMCR)



STRG	Software trigger bit
0	Invalid
1	Startup by software

CTEN	Count operation enable bit
0	Stops the operation
1	Enables the operation

MDSE	Mode selection bit
0	Continuous operation
1	One-shot operation

OSEL	Output polarity specification bit
0	Normal polarity
1	Inverted polarity

FMD2	FMD1	FMD0	Timer function selection bits
0	0	0	Reset mode
0	0	1	Selects PWM function
0	1	0	Selects PPG function
0	1	1	Selects reload timer function
1	0	0	Selects PWC function
1	0	1	Prohibited settings
1	1	0	
1	1	1	

Table 15-3. Timer Control Register (Lower Byte of TMCR)

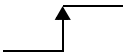
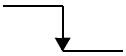
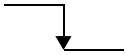
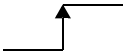
<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit7</td><td>Unused bit</td><td>■ The read value is “0”.
■ Write “0” to this bit.</td></tr>
<tr><td>bit6
to
bit4</td><td>FMD2 to FMD0:
Timer function selection bits</td><td>■ These bits are used to select a timer function.
■ The PWM function is selected when FMD2, FMD1 and FMD0 bits are set to “001_B”.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit3</td><td>OSEL:
Output polarity specification bit</td><td>This bit is used to set a polarity for the PWM output.
<table>
<tr><th>Polarity</th><th>After reset</th><th>Duty match</th><th>Underflow</th></tr>
<tr><td>Normal</td><td>“L” output</td><td></td><td></td></tr>
<tr><td>Inverted</td><td>“H” output</td><td></td><td></td></tr>
</table></td></tr>
<tr><td>bit2</td><td>MDSE:
Mode selection bit</td><td>■ This bit is used to select either the operation in which pulses are output continuously, or the one-shot operation in which only a single pulse is output.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit1</td><td>CTEN:
Count operation enable bit</td><td>■ This bit is used to enable the operation of the down counter.
■ The counter will be stopped if “0” is written to this bit when the counter operation has been enabled (CTEN=1).</td></tr>
<tr><td>bit0</td><td>STRG:
Software trigger bit</td><td>■ A software trigger will be applied if “1” is written to the STRG bit when the CTEN bit has been set to “1”.
Note:
A software trigger will also be applied if “1” is written to the CTEN and STRG bits at the same time.
■ The STRG bit always reads “0”.</td></tr>
</table>

Figure 15-3. Status Control Register (STC)

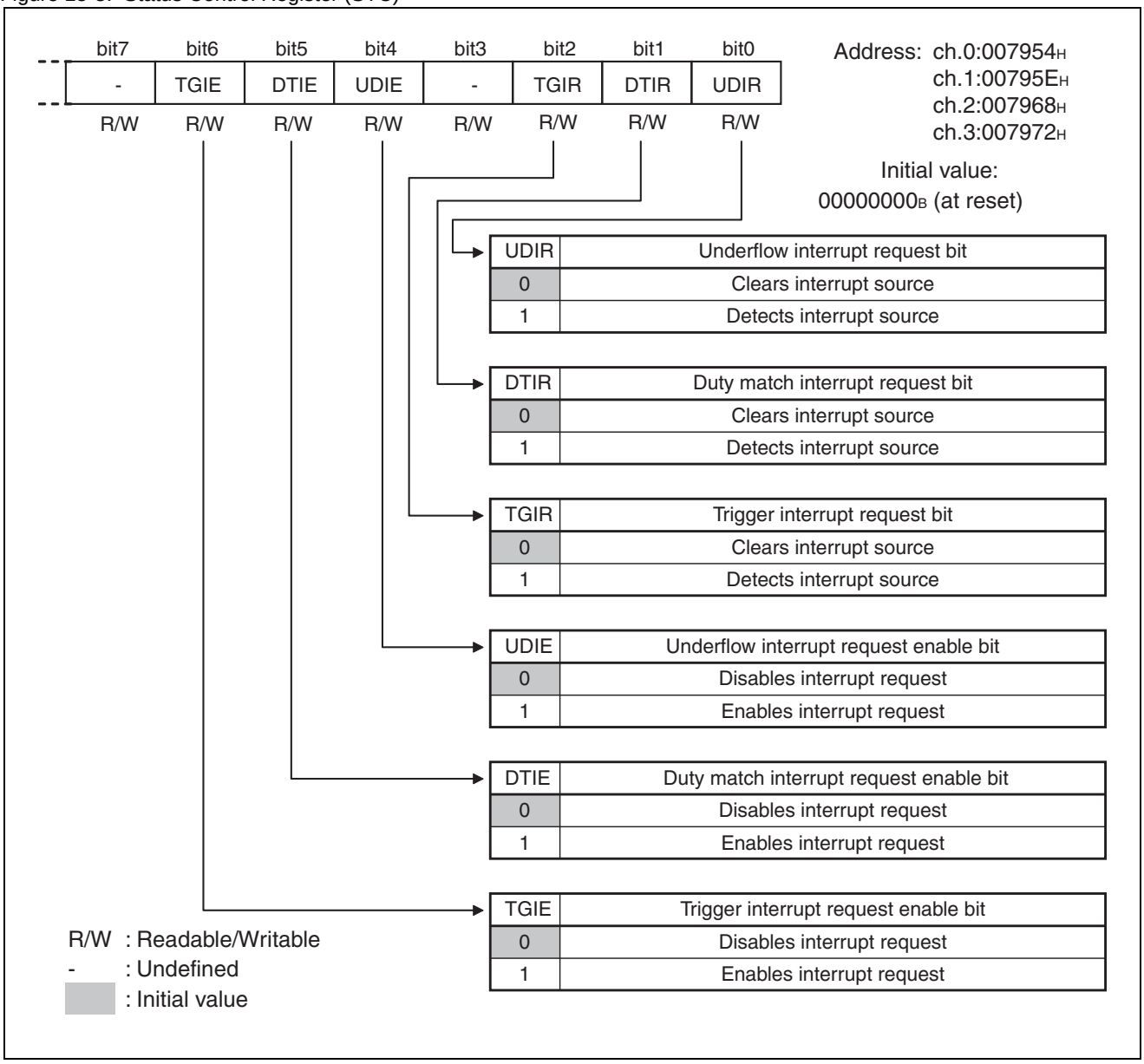


Table 15-4. Status Control Register (STC)

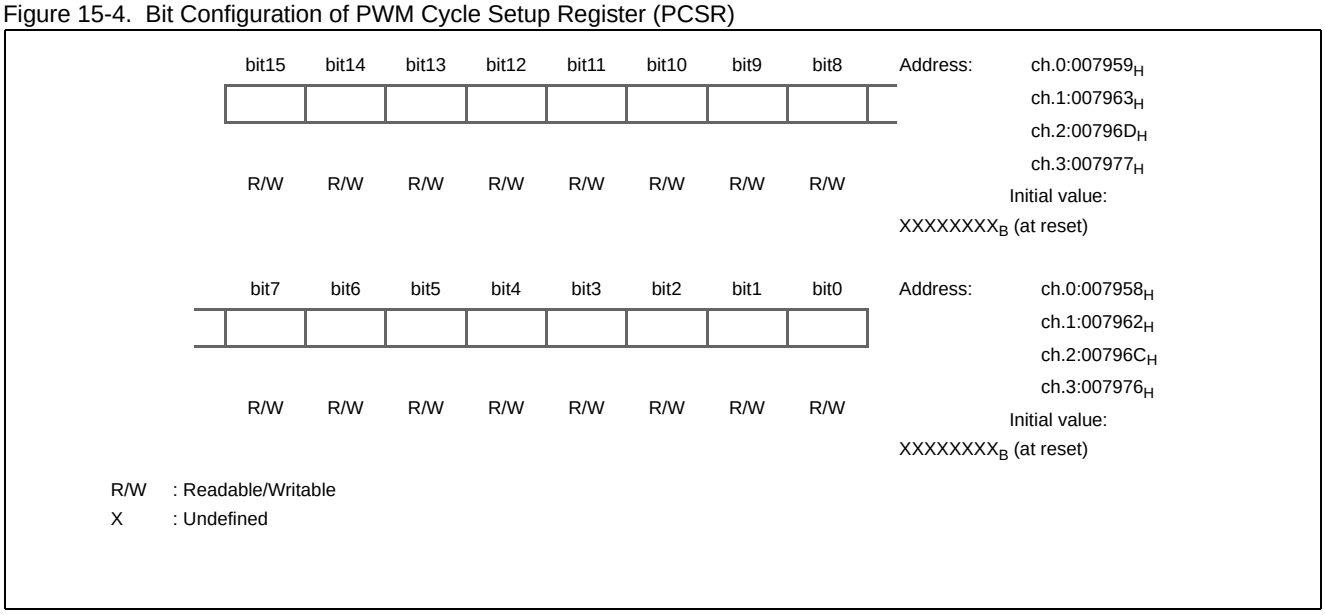
Bit name		Function
bit7	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit6	TGIE: Trigger interrupt request enable bit	■ This bit is used to control interrupt requests for TGIR (bit2). ■ An interrupt request will be generated to the CPU if the TGIR bit (bit2) is set when the TGIE bit is enabled.
bit5	DTIE: Duty match interrupt request enable bit	■ This bit is used to control interrupt requests for DTIR (bit1). ■ An interrupt request will be generated to the CPU if the DTIR bit (bit1) is set when the DTIE bit is enabled.
bit4	UDIE: Underflow interrupt request enable bit	■ This bit is used to control interrupt requests for UDIR (bit0). ■ An interrupt request will be generated to the CPU if the UDIR bit (bit0) is set when the UDIE bit is enabled.
bit3	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit2	TGIR: Trigger interrupt request bit	■ The TGIR bit is set to “1” when a software trigger is detected. ■ Writing “0” clears the TGIR bit. ■ Writing “1” to the TGIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.
bit1	DTIR: Duty match interrupt request bit	■ The DTIR bit is set to “1” when the count value matches the duty setting value. ■ Writing “0” clears the DTIR bit. ■ Writing “1” to the DTIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.
bit0	UDIR: Underflow interrupt request bit	■ The UDIR bit is set to “1” when the count value changes from 0000_H to $FFFF_H$, causing an underflow. ■ Writing “0” clears the UDIR bit. ■ Writing “1” to the UDIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.

PWM Cycle Setup Register (PCSR)

The PWM cycle setup register (PCSR) is a register with a buffer to set a cycle. A transfer to the timer register is performed upon startup and the occurrence of an underflow.

Figure 15-4 shows the bit configuration of the PWM cycle setup register (PCSR).

Figure 15-4. Bit Configuration of PWM Cycle Setup Register (PCSR)



This is a register with a buffer to set a cycle. A transfer to the timer register is performed upon startup and the occurrence of an underflow.

To initialize or rewrite the cycle setup register, always write to the duty setup register after writing to the cycle setup register.

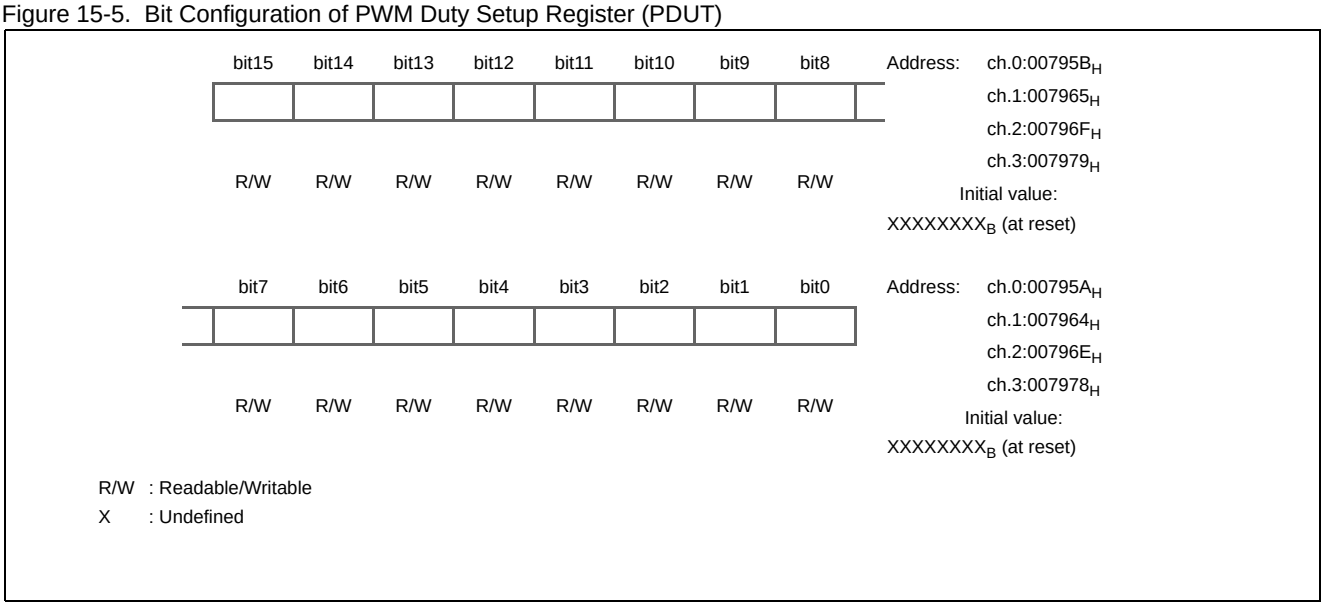
- The PCSR register should be accessed using 16-bit data.
- Set the PCSR register to select a cycle after setting the PWM function using the FMD2, FMD1 and FMD0 bits in the TMCR register.

PWM Duty Setup Register (PDUT)

The PWM duty setup register (PDUT) is a register with a buffer to set a duty. A transfer from the buffer is performed upon the occurrence of an underflow.

Figure 15-5 shows the bit configuration of the PWM duty setup register (PDUT).

Figure 15-5. Bit Configuration of PWM Duty Setup Register (PDUT)



This is a register with a buffer to set a duty. A transfer from the buffer is performed upon the occurrence of an underflow.

When the cycle setup register and the duty setup register are set to the same value, all “H” is output during the normal polarity, and all “L” is output during the inverted polarity.

Do not set any value that will cause PCSR < PDUT. The PWM output is undefined.

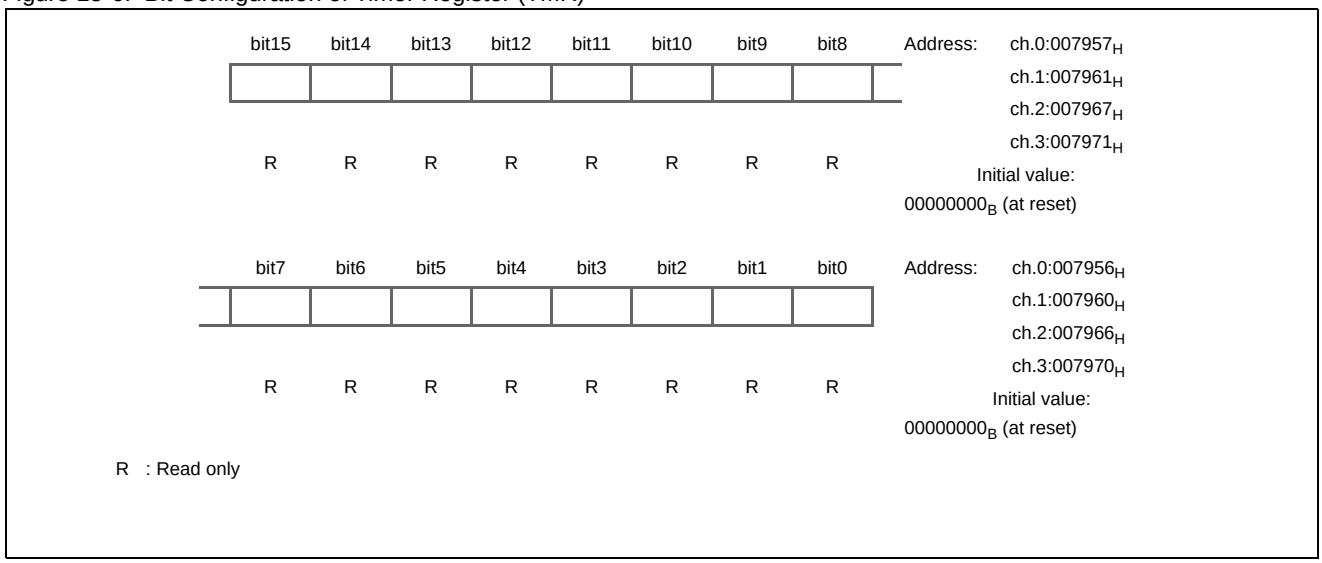
- The PDUT register should be accessed using 16-bit data.
- Set the PDUT register to select a duty ratio after setting the PWM function using the FMD2, FMD1 and FMD0 bits in the TMCR register.

Timer Register (TMR)

The timer register (TMR) can be used to read the value of the 16-bit down counter.

Figure 15-6 shows the bit configuration of the PWM timer register (TMR).

Figure 15-6. Bit Configuration of Timer Register (TMR)



This register can be used to read the value of the 16-bit down counter.

The TMR register should be accessed using 16-bit data.

Operation of 16-Bit PWM Timer

In PWM operation, the set cycle waveform can be output either only once or continuously upon the detection of a trigger.

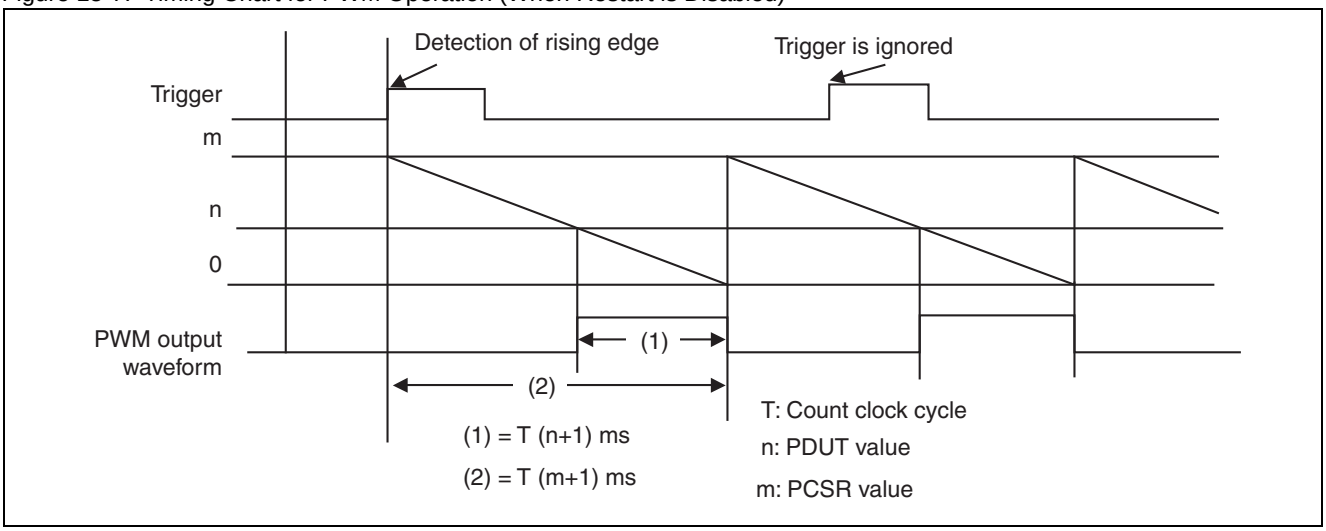
The cycle of output pulses can be controlled by modifying the PCSR value. Similarly, the duty ratio can be controlled by modifying the PDUT value.

Always write to PDUT after writing data to PCSR.

Continuous Operation

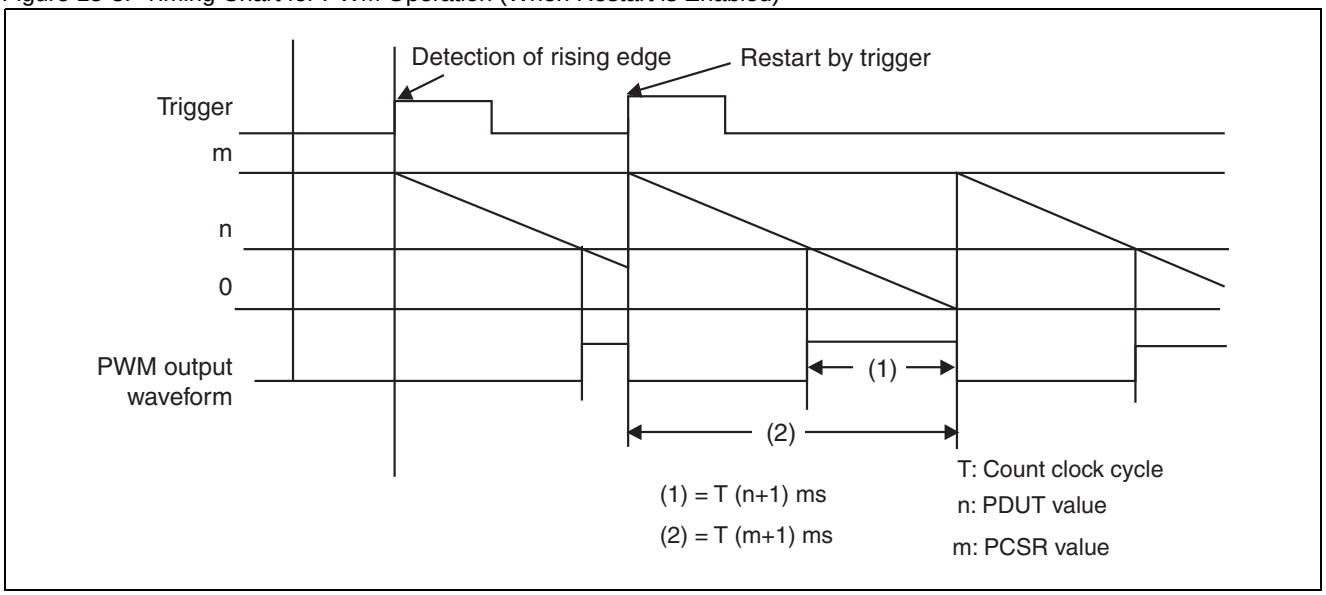
- When restart is disabled (RTGEN=0)

Figure 15-7. Timing Chart for PWM Operation (When Restart is Disabled)



- When restart is enabled (RTGEN=1)

Figure 15-8. Timing Chart for PWM Operation (When Restart is Enabled)

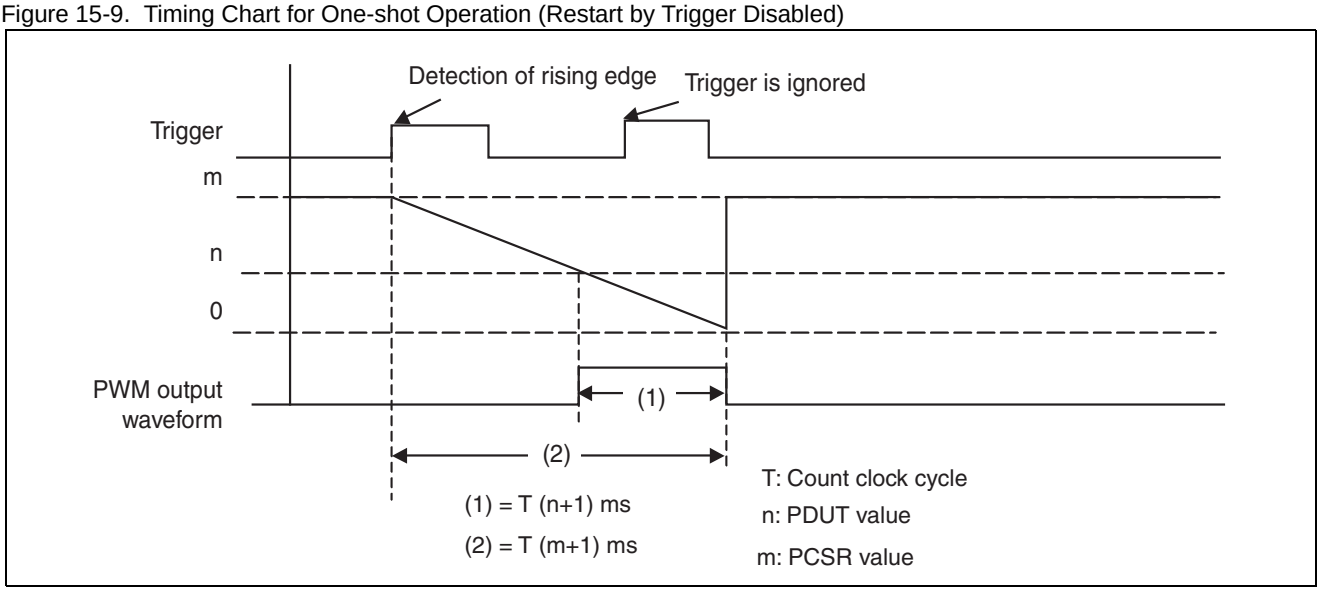


One-shot Operation

One-shot operation allows the user to output a single pulse with any width using a trigger. When restart is enabled, the counter is reloaded once an edge is detected during operation.

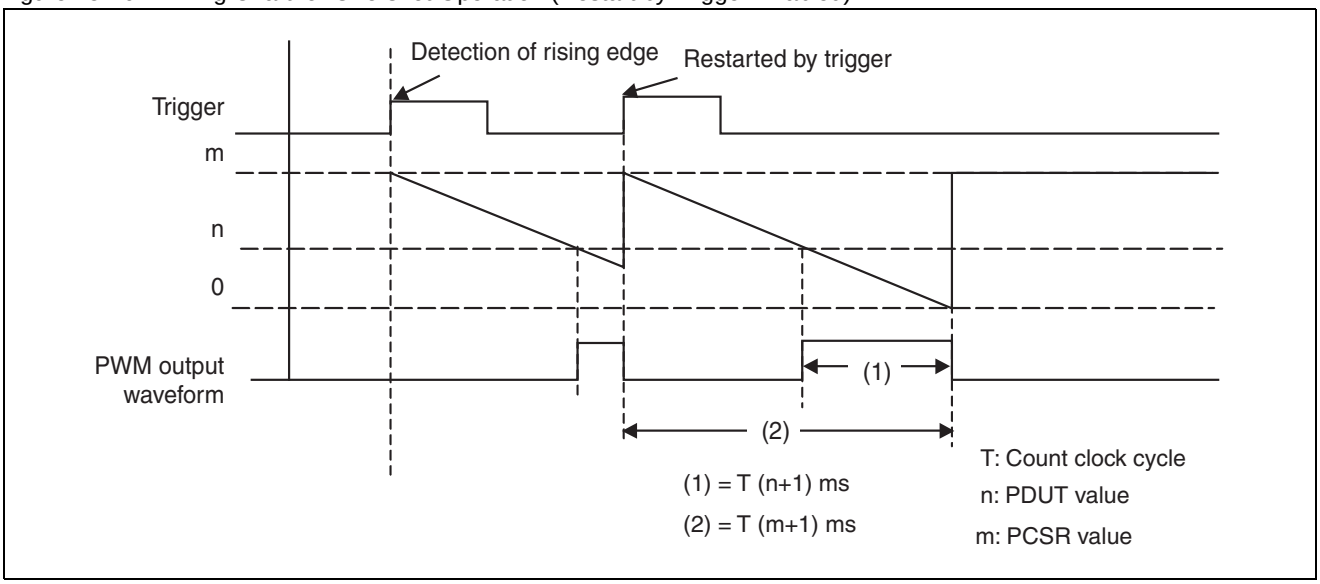
- When restart is disabled (RTGEN=0)

Figure 15-9. Timing Chart for One-shot Operation (Restart by Trigger Disabled)



- When restart is enabled (RTGEN=1)

Figure 15-10. Timing Chart for One-shot Operation (Restart by Trigger Enabled)

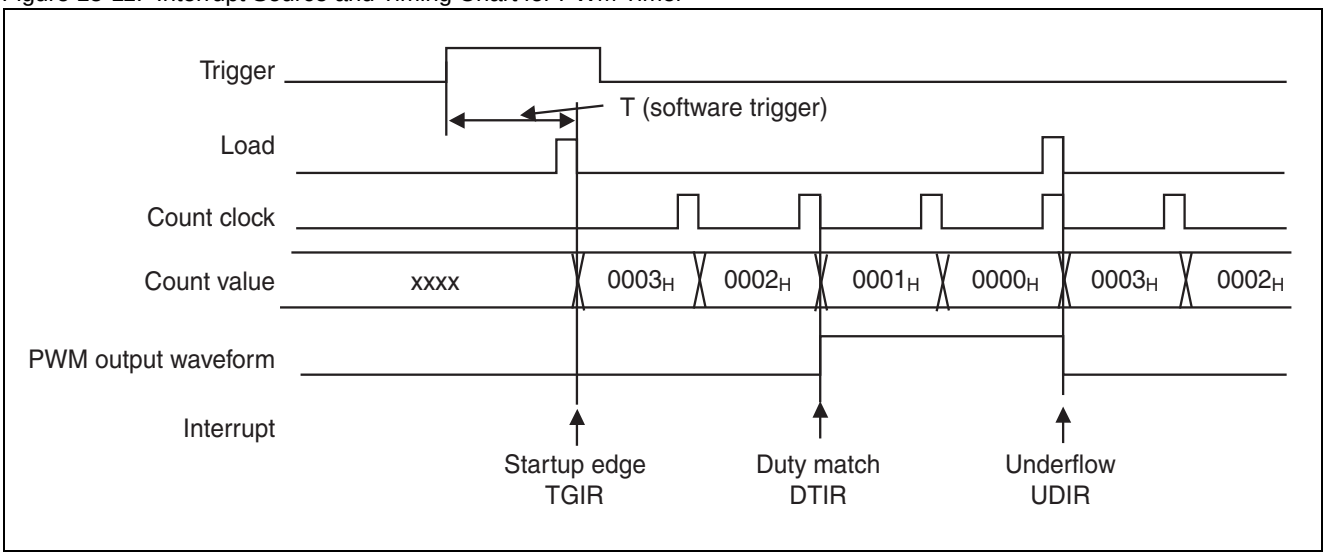


Interrupt Source and Timing Chart (PWM Output: Normal Polarity)

It takes T for a software trigger (T: machine cycle) from when the trigger is input until the counter value is loaded.

Figure 15-11 shows the interrupt source and timing chart for when the cycle setting value is “3” and the duty value is “1”.

Figure 15-11. Interrupt Source and Timing Chart for PWM Timer



How to Output All “L” or All “H” when Outputting PWM

Figure 15-12 shows how to output all “L” when outputting PWM, while Figure 15-13 shows how to output all “H”.

Figure 15-12. Example Method for Setting All PWM Output to “L” Level

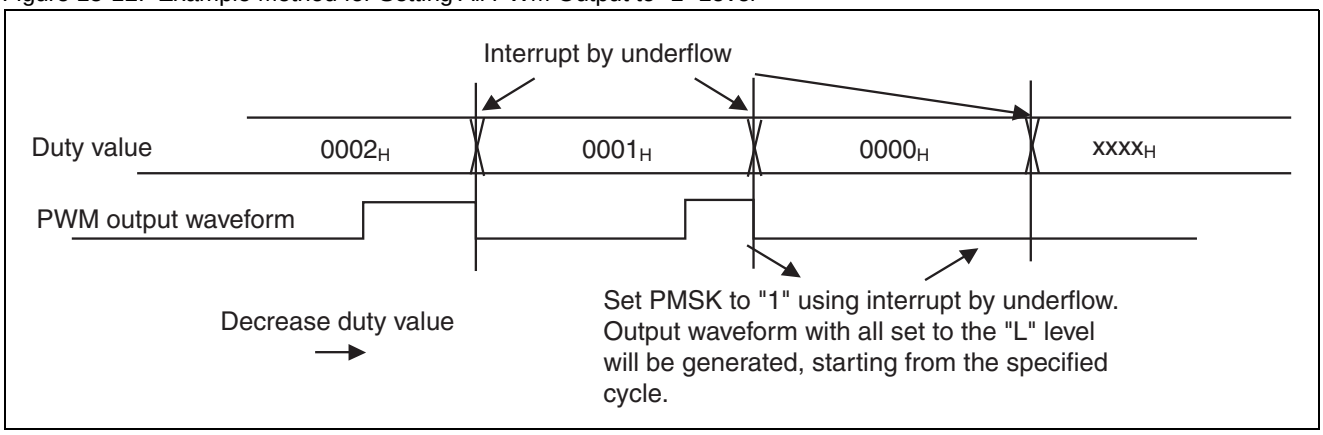
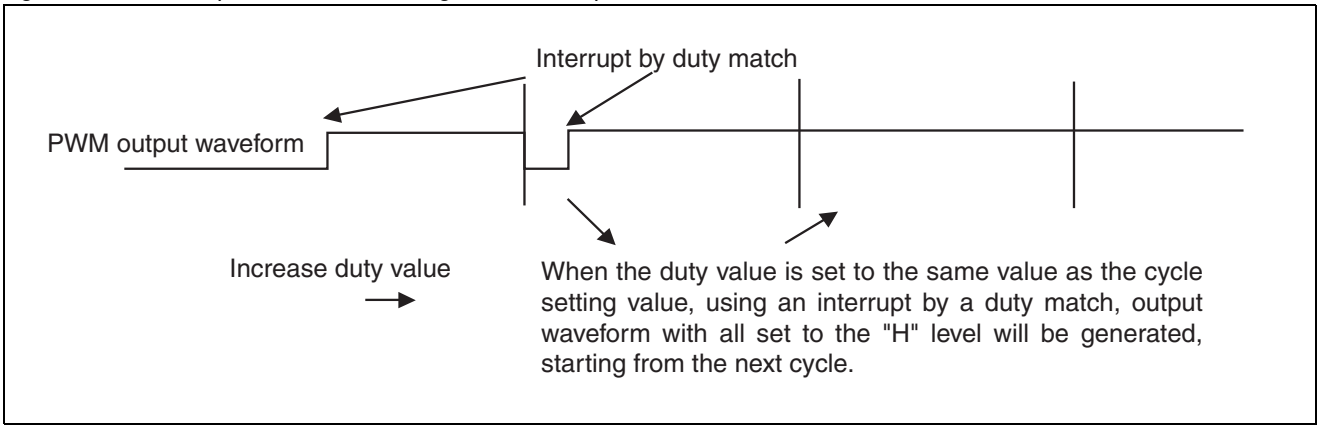


Figure 15-13. Example Method for Setting All PWM Output to “H” Level



15.8.2 PPG Timer Function

Only one of the following timer functions can be selected for the base timer by setting FMD2, FMD1 and FMD0 bits in the timer control register: 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section describes the timer function when PPG is selected.

Timer Control Register (TMCR) when PPG Timer is Selected

The timer control register (TMCR) is used to control the PPG timer. Take care that some bits cannot be rewritten during the operation of the PPG timer.

Figure 15-14. Timer Control Register (Upper Byte of TMCR)

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
POE	CKS2	CKS1	CKS0	RTGEN	PMSK	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Address: ch.0:007957H
ch.1:00795DH
ch.2:007967H
ch.3:007975H
Initial value:
00000000B(at reset)

PMSK	Pulse output mask bit
0	Normal output
1	Fixed at "L" output

RTGEN	Restart enable bit
0	Restart disabled
1	Restart enabled

CKS2	CKS1	CKS0	Count clock selection bits
0	0	0	ϕ
0	0	1	ϕ/4
0	1	0	ϕ/16
0	1	1	ϕ/128
1	0	0	ϕ/256
1	0	1	External clock (Rising edge event)
1	1	0	External clock (Falling edge event)
1	1	1	External clock (Both edges event)

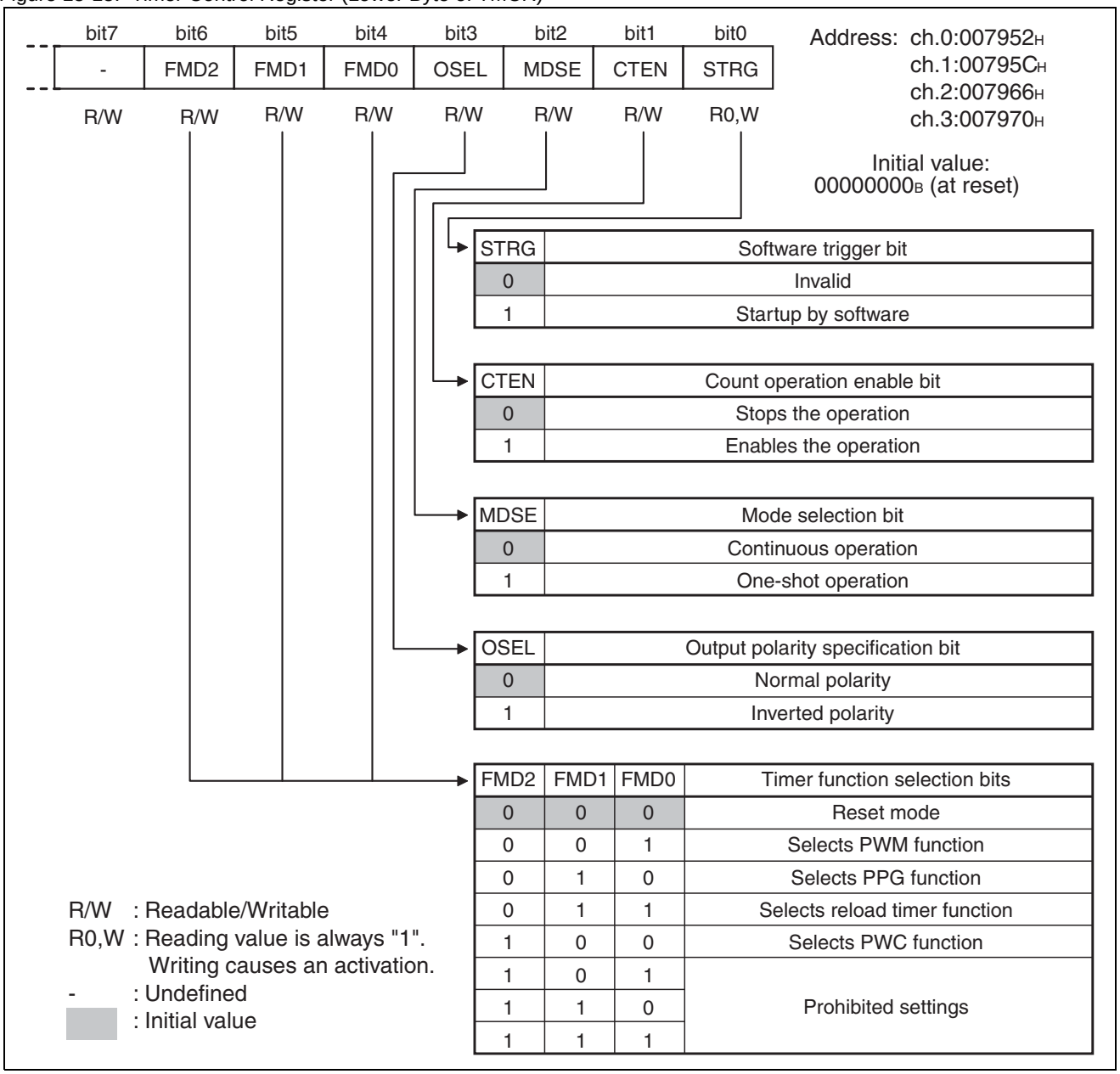
POE	Port output enable bit
0	Output disabled
1	Output enabled

R/W: Readable/Writable
: Initial value

Table 15-5. Timer Control Register (Upper Byte of TMCR)

Bit name		Function
bit15	POE: Port output enable bit	■ This bit is used to control the port output of PPG output waveform. ■ The TIO pin serves as a general-purpose port when this bit is set to “0”, and the same pin functions as an output pin for the PPG waveform when the bit is set to “1”.
bit14 to bit12	CKS2 to CKS0: Count clock selection bits	■ These bits are used to select a count clock for the 16-bit down counter. ■ Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS2 to CKS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.
bit11	RTGEN: Restart enable bit	This bit is used to enable restart by the input of a software trigger or a trigger.
bit10	PMSK: Pulse output mask bit	■ This bit is used to control the level of the PPG output waveform. ■ When this bit is set to “0”, the PPG waveform is output in its original form. ■ When this bit is set to “1”, PPG output is masked to “L” output, regardless of the “H”- and “L”-width setting values. **Note:** The output will be set to “H” if PMSK is set to “1” when OSEL (bit3) has been set to invert the output.
bit9, bit8	Unused bits	■ The read value is “0”. ■ Write “0” to these bits.

Figure 15-15. Timer Control Register (Lower Byte of TMCR)



STRG	Software trigger bit
0	Invalid
1	Startup by software

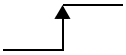
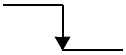
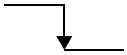
CTEN	Count operation enable bit
0	Stops the operation
1	Enables the operation

MDSE	Mode selection bit
0	Continuous operation
1	One-shot operation

OSEL	Output polarity specification bit
0	Normal polarity
1	Inverted polarity

FMD2	FMD1	FMD0	Timer function selection bits
0	0	0	Reset mode
0	0	1	Selects PWM function
0	1	0	Selects PPG function
0	1	1	Selects reload timer function
1	0	0	Selects PWC function
1	0	1	Prohibited settings
1	1	0	
1	1	1	

Table 15-6. Timer Control Register (Lower Byte of TMCR)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit7</td><td>Unused bit</td><td>■ The read value is “0”.
■ Write “0” to this bit.</td></tr>
<tr><td>bit6
to
bit4</td><td>FMD2 to FMD0:
Timer function selection bits</td><td>■ These bits are used to select a timer function.
■ The PPG function is selected when FMD2, FMD1 and FMD0 bits are set to “010_B”.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit3</td><td>OSEL:
Output polarity specification bit</td><td>This bit is used to set a polarity for the PPG output.
<table>
<tr><th>Polarity</th><th>After reset</th><th>Completion of “L” width count</th><th>Completion of “H” width count</th></tr>
<tr><td>Normal</td><td>“L” output</td><td></td><td></td></tr>
<tr><td>Inverted</td><td>“H” output</td><td></td><td></td></tr>
</table></td></tr>
<tr><td>bit2</td><td>MDSE:
Mode selection bit</td><td>■ This bit is used to select either the operation in which pulses are output continuously, or the one-shot operation in which only a single pulse is output.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit1</td><td>CTEN:
Count operation enable bit</td><td>■ This bit is used to enable the operation of the down counter.
■ The counter will be stopped if “0” is written to this bit when the counter operation has been enabled (CTEN=1).</td></tr>
<tr><td>bit0</td><td>STRG:
Software trigger bit</td><td>■ A software trigger will be applied if “1” is written to the STRG bit when the CTEN bit has been set to “1”.
Note:
A software trigger will also be applied if “1” is written to the CTEN and STRG bits at the same time.
■ The STRG bit always reads “0”.</td></tr>
</table>

Status Control Register (STC)

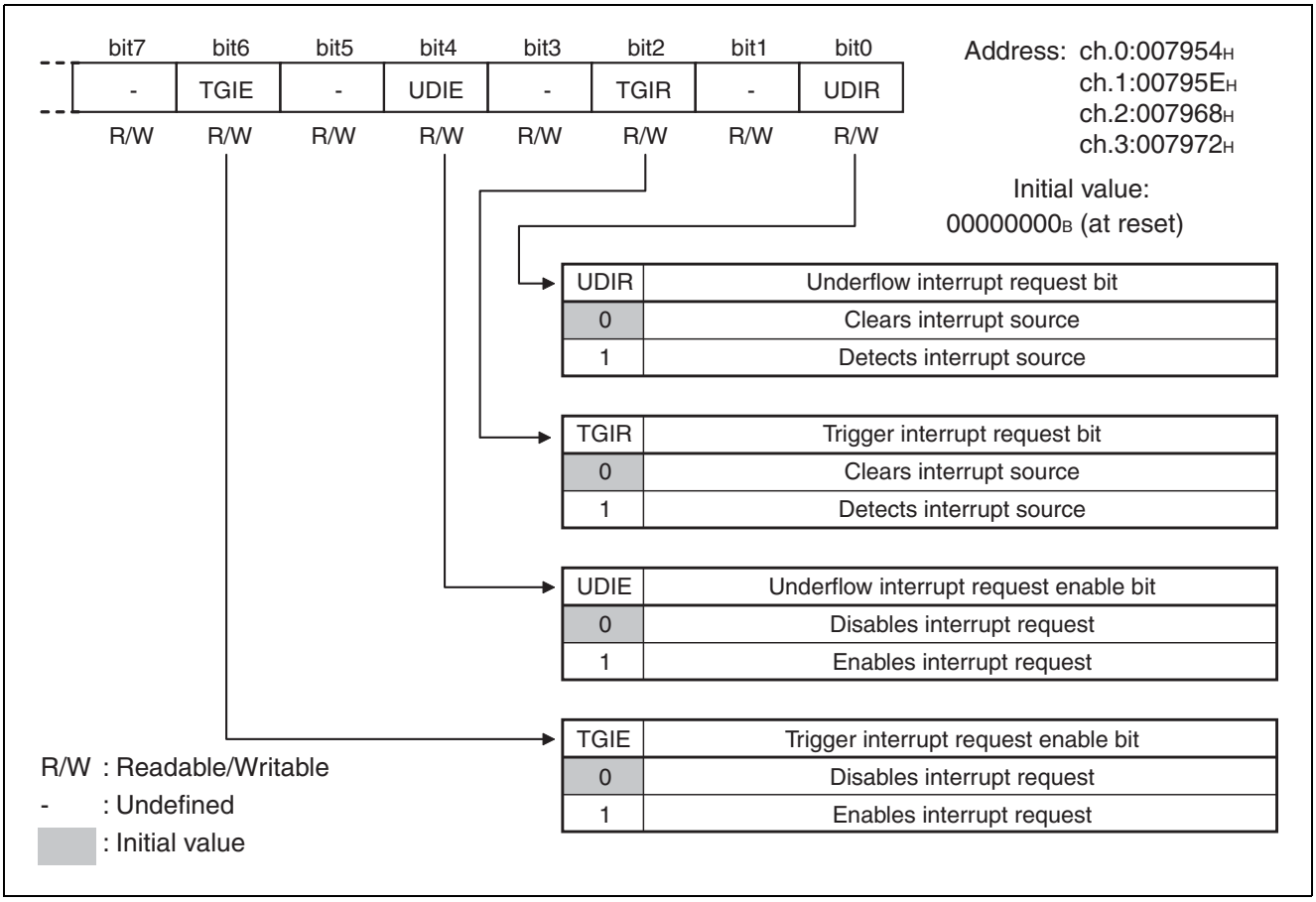


Table 15-7. Status Control Register (STC)

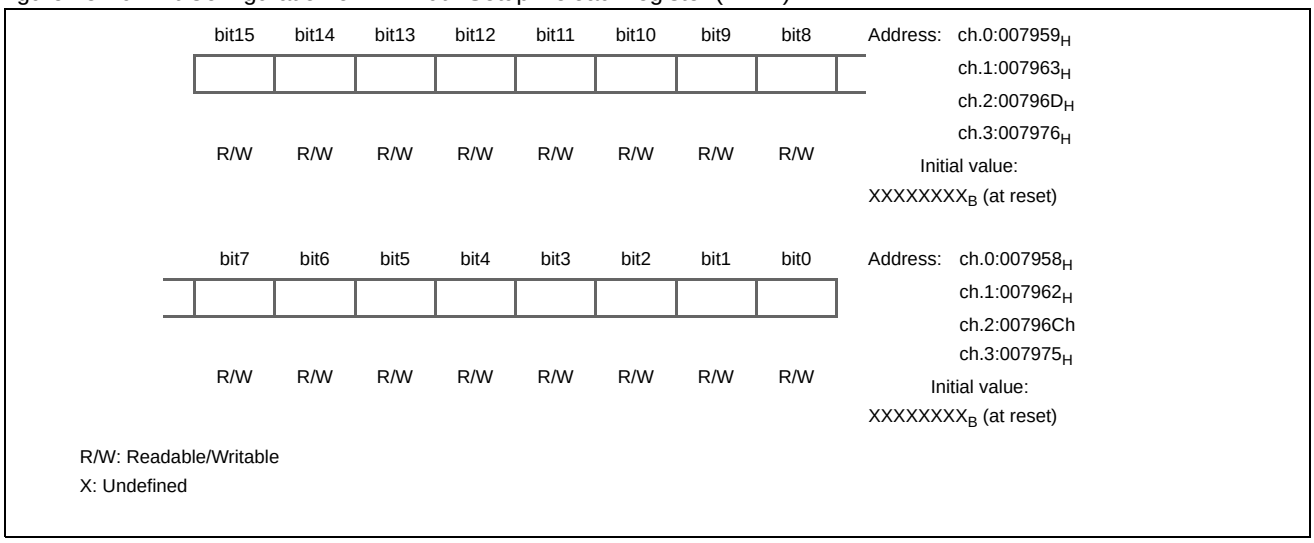
Bit name		Function
bit7	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit6	TGIE: Trigger interrupt request enable bit	■ This bit is used to control interrupt requests for TGIR (bit2). ■ An interrupt request will be generated to the CPU if the TGIR bit (bit2) is set when the TGIE bit is enabled.
bit5	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit4	UDIE: Underflow interrupt request enable bit	■ This bit is used to control interrupt requests for UDIR (bit0). ■ An interrupt request will be generated to the CPU if the UDIR bit (bit0) is set when the UDIE bit is enabled.
bit3	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit2	TGIR: Trigger interrupt request bit	■ The TGIR bit is set to “1” when a software trigger is detected. ■ Writing “0” clears the TGIR bit. ■ Writing “1” to the TGIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.
bit1	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit0	UDIR: Underflow interrupt request bit	■ The UDIR bit is set to “1” when the count value changes from 0000_H to $FFFF_H$, causing an underflow, during the count starting from the value set to the "H" width. ■ Writing “0” clears the UDIR bit. ■ Writing “1” to the UDIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.

“L” Width Setup Reload Register (PRLL)

The “L” width setup reload register (PRLL) is used to set the “L” width of the PPG output waveform. A transfer to the timer register is performed either upon the detection of an activation trigger or when an underflow occurs upon the completion of “H” width count.

Figure 15-16 shows the bit configuration of the “L” width setup reload register (PRLL).

Figure 15-16. Bit Configuration of “L” Width Setup Reload Register (PRLL)



This register is used to set the “L” width of the PPG output waveform. A transfer to the timer register is performed either upon the detection of an activation trigger or when an underflow occurs upon the completion of “H” width count.

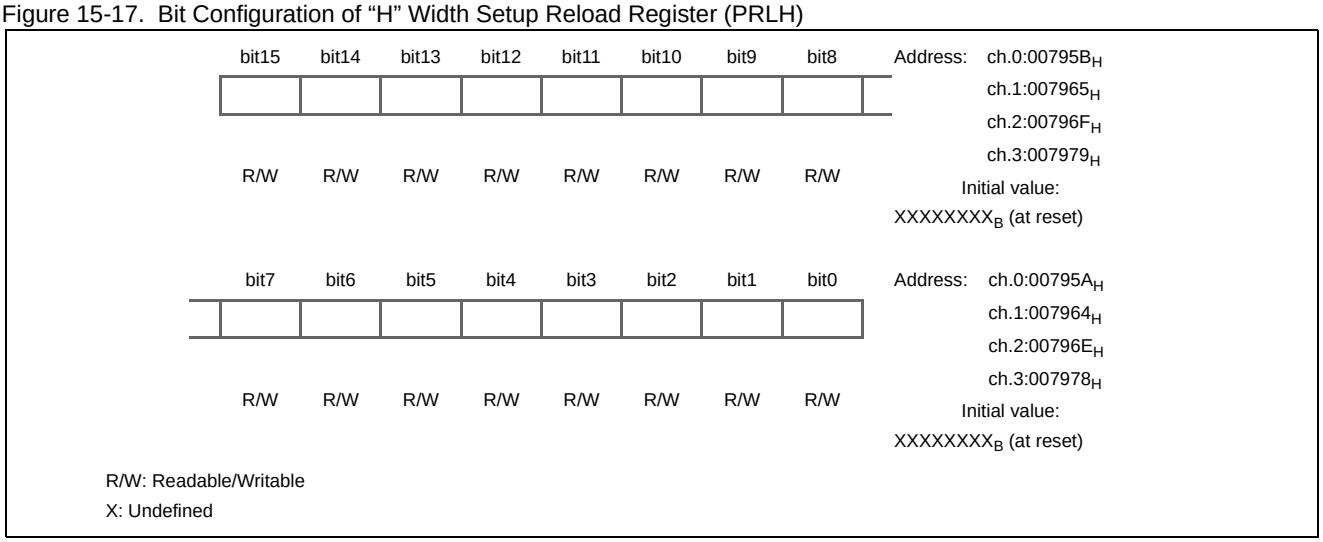
- The PRLL register should be accessed using 16-bit data.
- Set the PRLL register to the “L” width after setting the PPG function using the FMD2, FMD1 and FMD0 bits in the TMCR register.

“H” Width Setup Reload Register (PRLH)

The “H” width setup reload register (PRLH) is a register with a buffer to set the “H” width of the PPG output waveform. A transfer from PRLH to the buffer register is performed upon the detection of an activation trigger and when an underflow occurs upon the completion of “H” width count. A transfer from the buffer register to the timer register is performed when an underflow occurs upon the completion of “L” width count.

Figure 15-17 shows the bit configuration of the “H” width setup reload register (PRLH).

Figure 15-17. Bit Configuration of “H” Width Setup Reload Register (PRLH)



This register is used to set the “H” width of the PPG output waveform. A transfer from PRLH to the buffer register is performed upon the detection of an activation trigger and when an underflow occurs upon the completion of “H” width count. A transfer from the buffer register to the timer register is performed when an underflow occurs upon the completion of “L” width count.

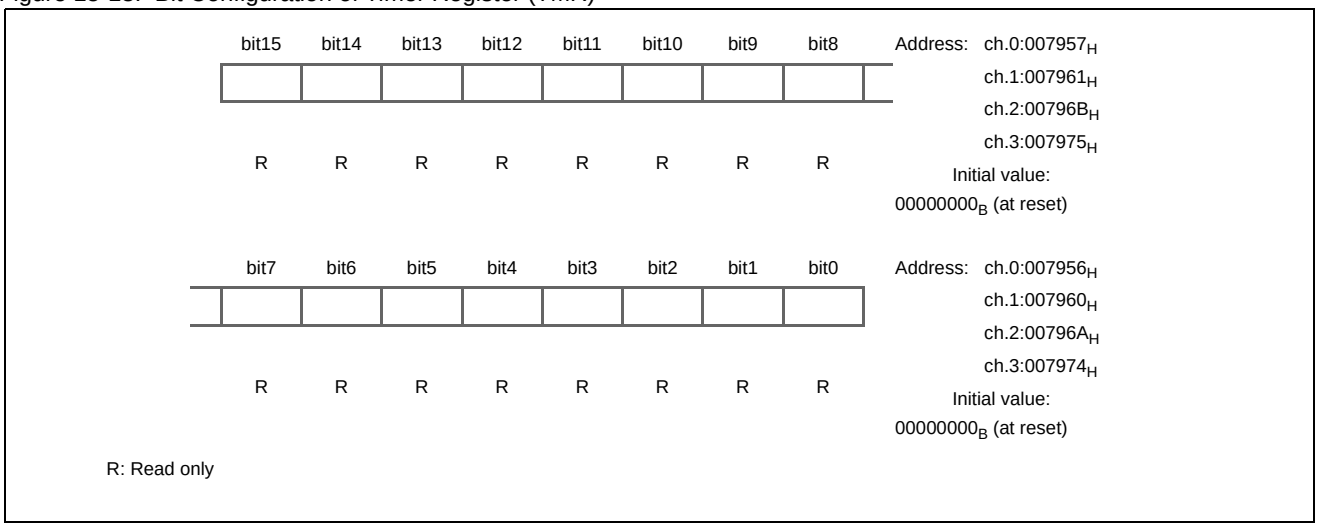
- The PRLH register should be accessed using 16-bit data.
- Set the PRLH register to the “H” width after setting the PPG function using the FMD2, FMD1 and FMD0 bits in the TMCR register.

Timer Register (TMR)

The timer register (TMR) can be used to read the value of the 16-bit down counter.

Figure 15-18 shows the bit configuration of the PPG timer register (TMR).

Figure 15-18. Bit Configuration of Timer Register (TMR)



This register can be used to read the value of the 16-bit down counter.

The TMR register should be accessed using 16-bit data.

Operation of 16-Bit PPG Timer

In PPG operation, any output pulse can be controlled by setting the “L” and “H” widths of the output pulse to the corresponding reload registers.

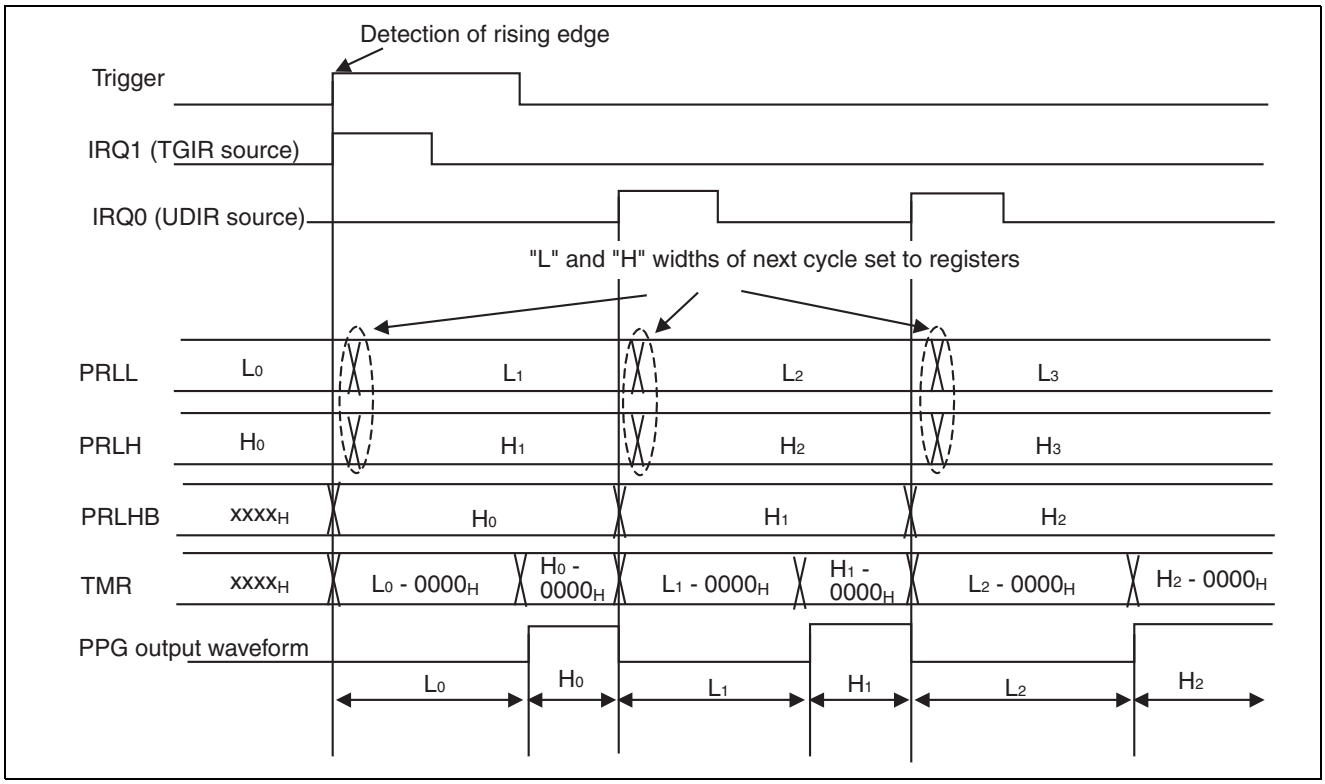
The timer has two 16-bit reload registers, one for setting the “L” width and the other for the “H” width, and one buffer for setting the "H" width. (PRLL, PRLH, PRLHB)

An activation trigger is used to load the value set in PRLL to the 16-bit down counter first, and then the value set in PRLH is transferred to PRLHB at the same time. With PPG output set to the “L” level, the timer counts down for each count clock. When an underflow is detected, the PRLHB value is reloaded to the counter and it counts down with the PPG output waveform inverted. When the next underflow is detected, the PPG output waveform is inverted, the value set in PRLL is reloaded to the counter, and the value set in PRLH is transferred to PRLHB.

This operation converts the output waveform to pulse output with “L” and “H” widths corresponding to each reload register value.

Timing for Writing to Reload Registers

Data should be written to the reload registers PRLL and PRLH upon the detection of an activation trigger, and during a period from when the underflow interrupt source (UDIR) is set until the next cycle starts. The data set at this point will be used in the next cycle. The data set in PRLL and PRLH is automatically transferred to TMR and PRLHB respectively when an activation trigger is detected and when an underflow occurs upon the completion of “H” width count. The data transferred to PRLHB is automatically reloaded to TMR when an underflow occurs upon the completion of “L” width count.

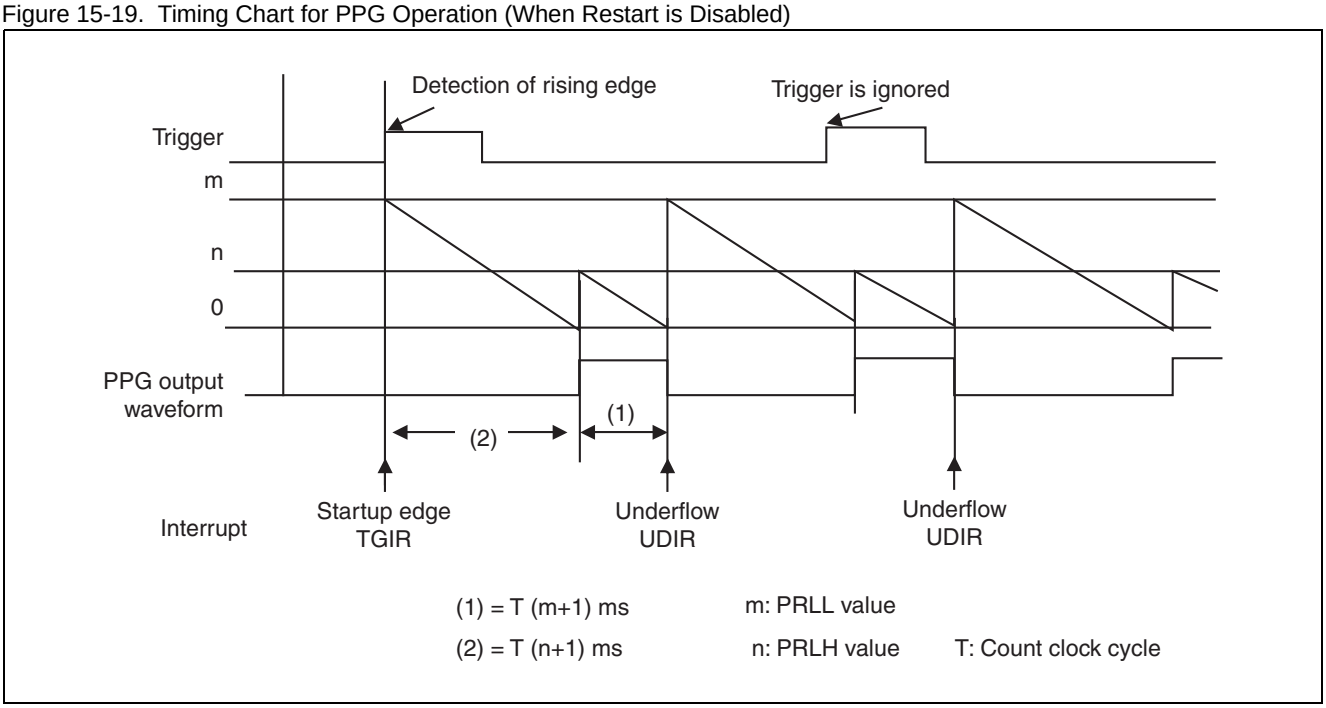


Continuous Operation

In continuous operation, any pulse can be output continuously when the “L” and “H” widths are updated at the same time as each interrupt source is set. The counter will be reloaded, if an edge is detected during operation when restart has been enabled.

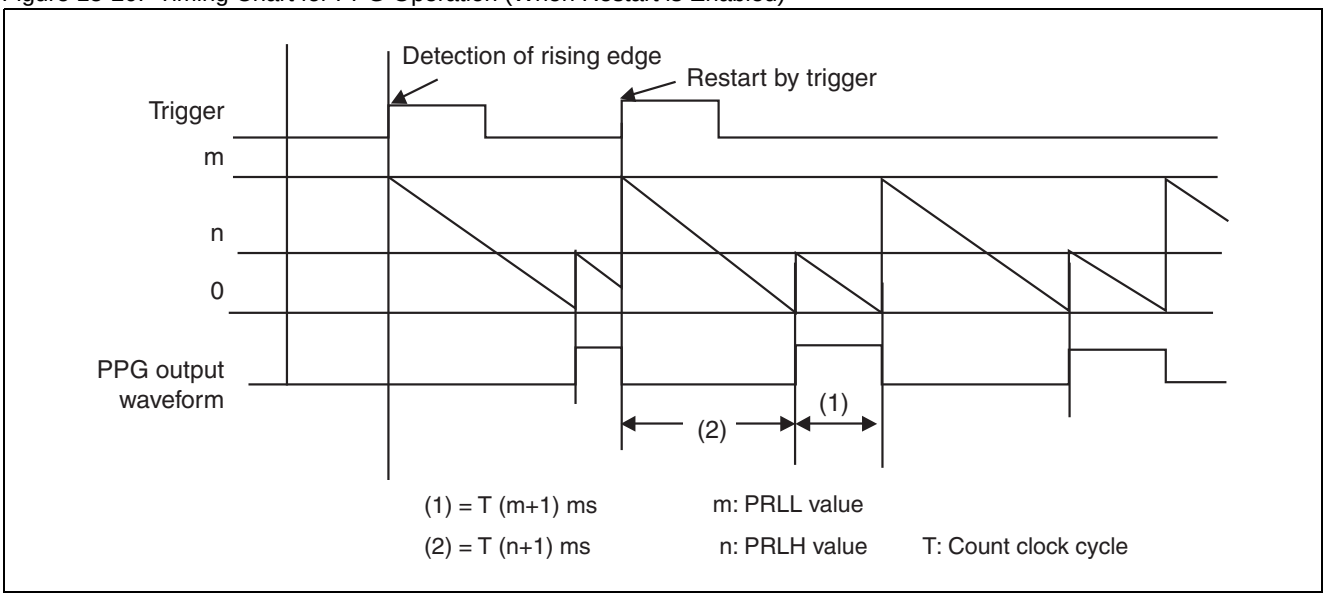
- When restart is disabled (RTGEN=0)

Figure 15-19. Timing Chart for PPG Operation (When Restart is Disabled)



- When restart is enabled (RTGEN=1)

Figure 15-20. Timing Chart for PPG Operation (When Restart is Enabled)

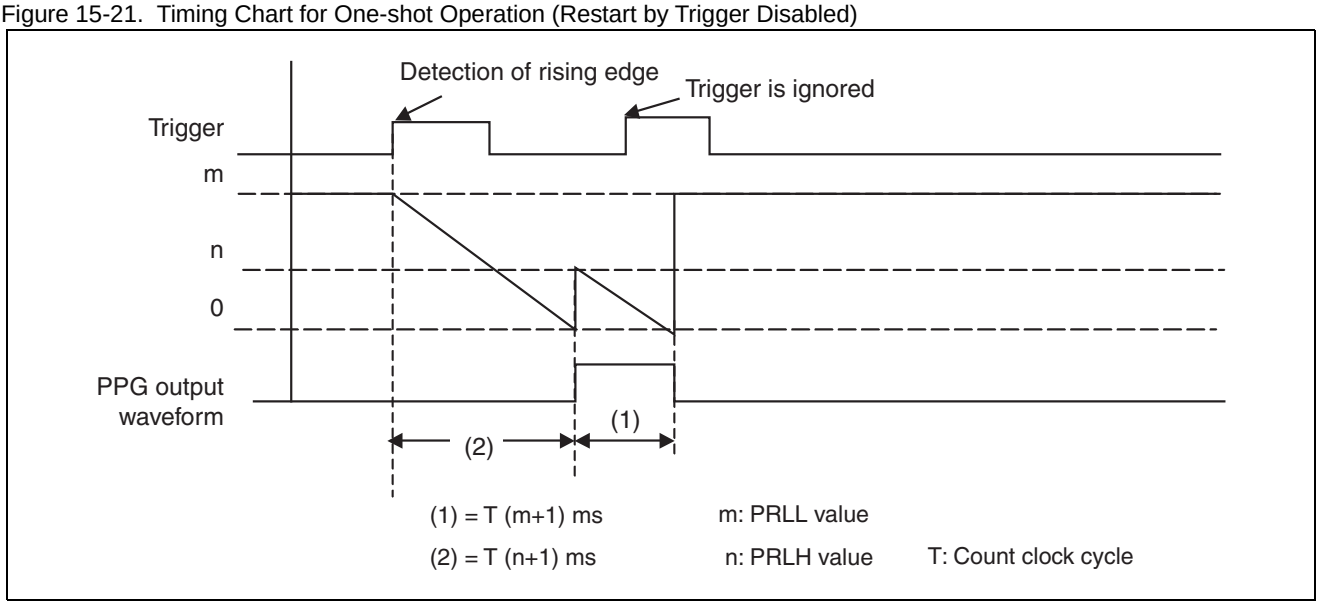


One-shot Operation

One-shot operation allows the user to output a single pulse with any width using a trigger. When restart is enabled, the counter is reloaded once an edge is detected during operation.

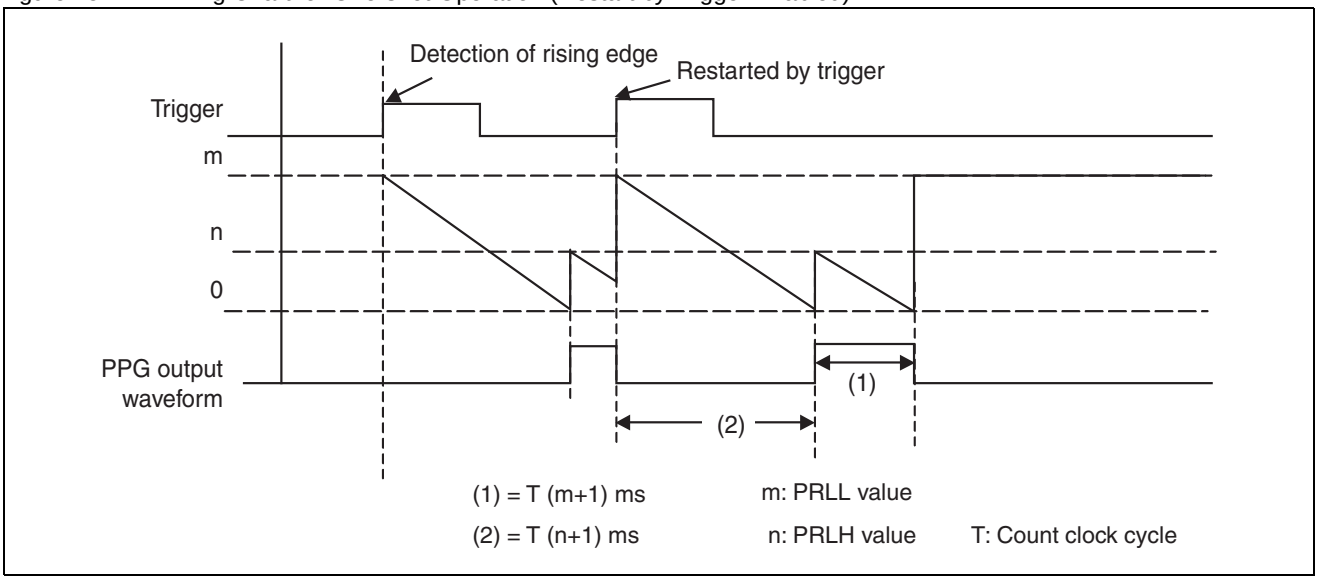
- When restart is disabled (RTGEN=0)

Figure 15-21. Timing Chart for One-shot Operation (Restart by Trigger Disabled)



- When restart is enabled (RTGEN=1)

Figure 15-22. Timing Chart for One-shot Operation (Restart by Trigger Enabled)



Relationship between Reload Value and Pulse Width

The pulse width to be output is the value written in the 16-bit reload register to which “1” has been added and multiplied by the count clock cycle. Therefore, the pulse width should be one cycle of the count clock, when the reload register value is “0000_H”. The pulse width should be 65536 cycles of the count clock, when the reload register value is “$FFFF_H$”. The following formula is used to calculate the pulse width.

PL = T × (L+1)

PH = T × (H+1)

PL : “L” pulse width

PH : “H” pulse width

T : Count clock cycle

L : PRLL value

H : PRLH value

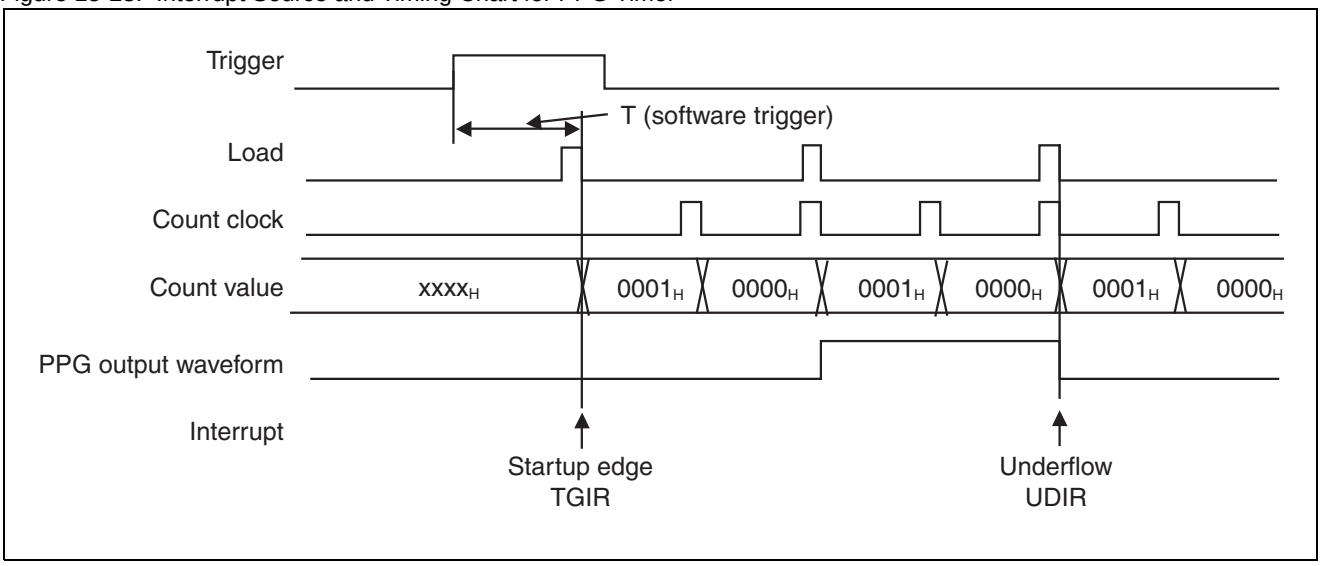
Interrupt Source and Timing Chart (PPG Output: Normal Polarity)

It takes T for a software trigger (T: machine cycle) from when the trigger is input until the counter value is loaded.

The interrupt source is set when a PPG activation trigger is detected and when an underflow is detected during “H”-level output.

Figure 15-23 shows the interrupt source and timing chart for when the setting value of the “L” width is “1” and the setting value of the “H” width is “1”.

Figure 15-23. Interrupt Source and Timing Chart for PPG Timer



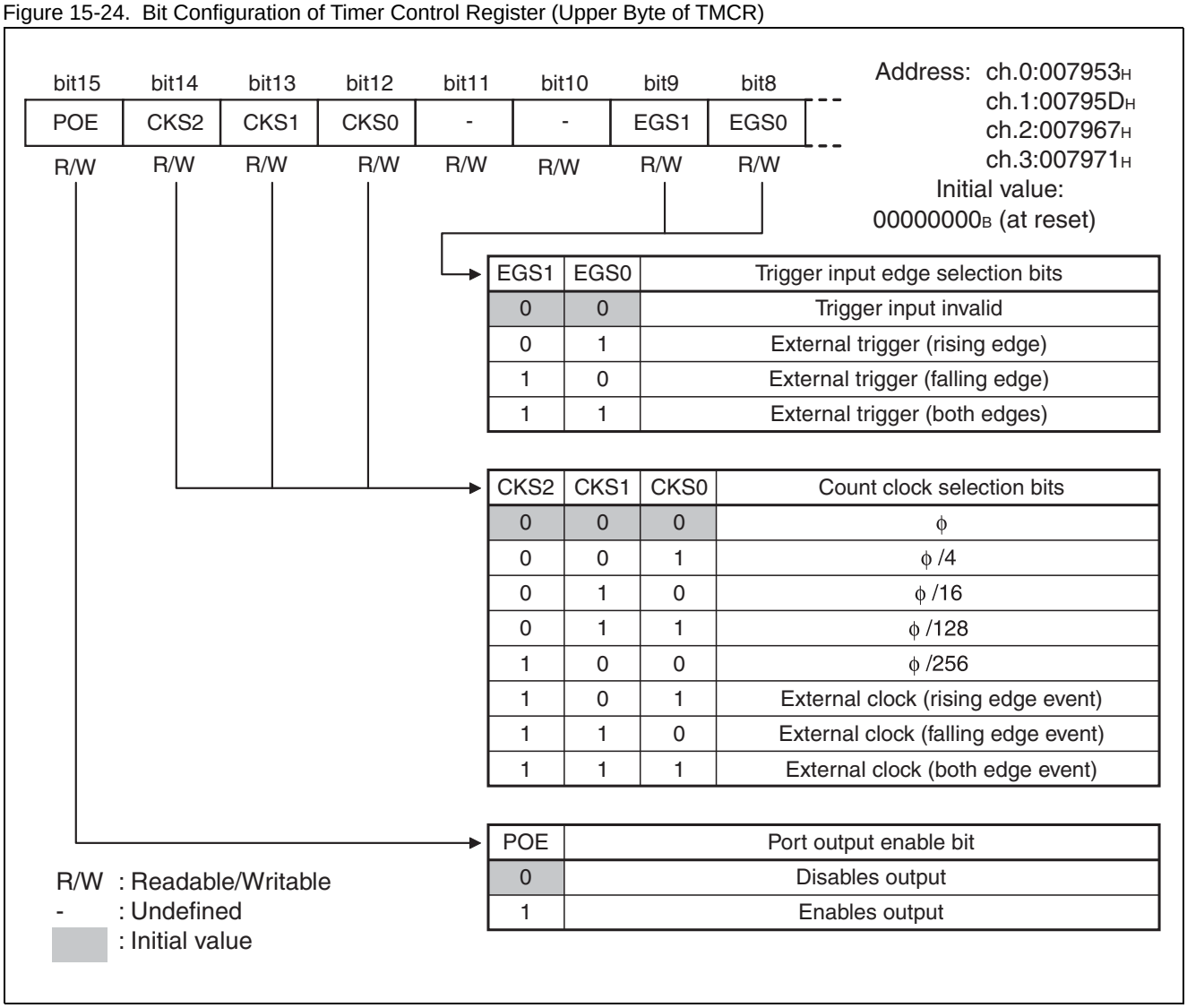
15.8.3 Reload Timer Function

Only one of the following timer functions can be selected for the base timer by setting FMD2, FMD1 and FMD0 bits in the timer control register: 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer. This section describes the timer function when the reload timer is selected.

Timer Control Register (TMCR) when Reload Timer is Selected

The timer control register (TMCR) is used to control the timer operation.

Figure 15-24. Bit Configuration of Timer Control Register (Upper Byte of TMCR)



EGS1	EGS0	Trigger input edge selection bits
0	0	Trigger input invalid
0	1	External trigger (rising edge)
1	0	External trigger (falling edge)
1	1	External trigger (both edges)

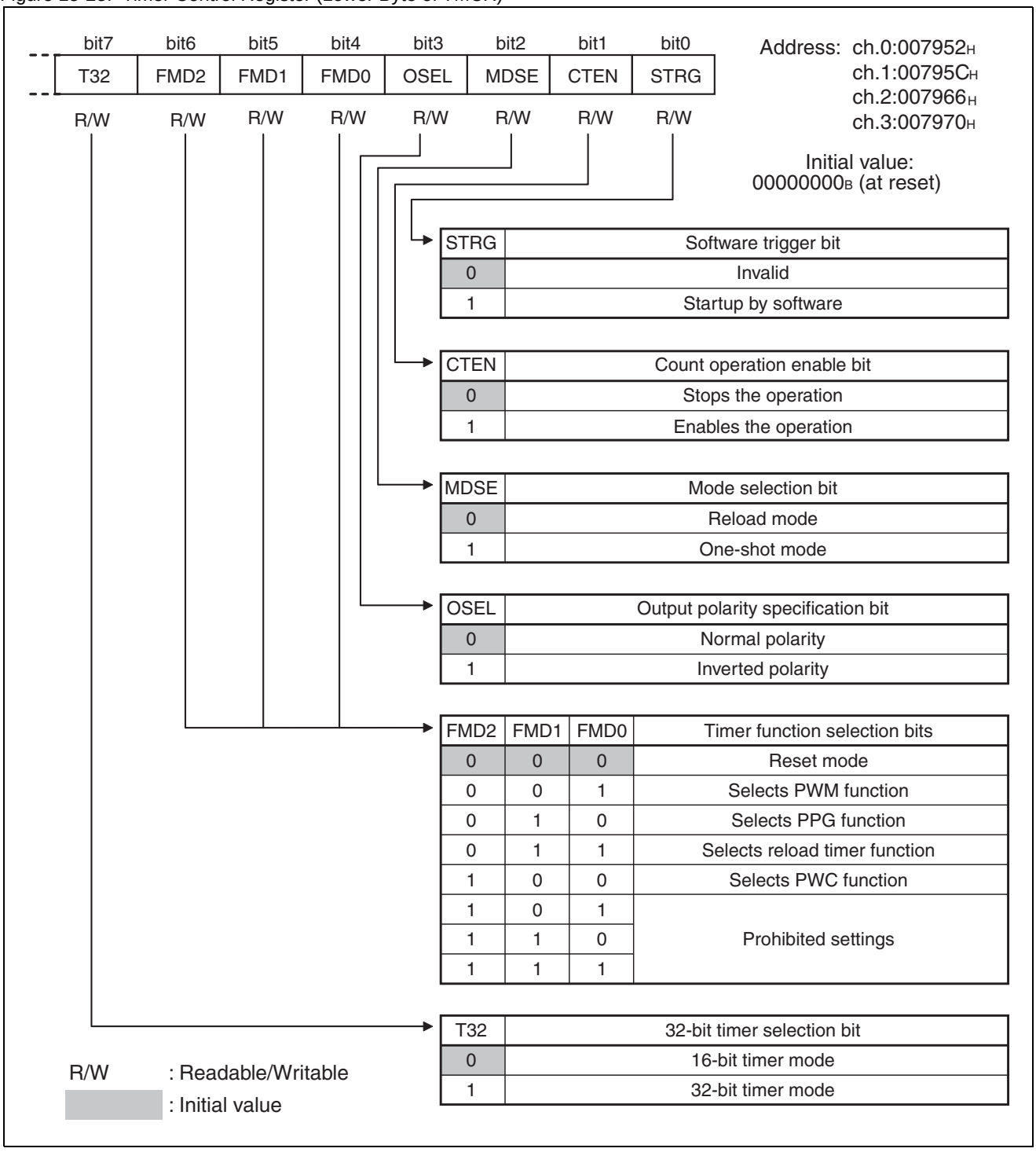
CKS2	CKS1	CKS0	Count clock selection bits
0	0	0	ϕ
0	0	1	ϕ /4
0	1	0	ϕ /16
0	1	1	ϕ /128
1	0	0	ϕ /256
1	0	1	External clock (rising edge event)
1	1	0	External clock (falling edge event)
1	1	1	External clock (both edge event)

POE	Port output enable bit
0	Disables output
1	Enables output

Table 15-8. Timer Control Register (Upper Byte of TMCR)

Bit name		Function
bit15	POE: Port output enable bit	■ This bit is used to control the port output of reload timer output waveform. ■ The TIO pin serves as a general-purpose port when this bit is set to “0”, and the same pin functions as an output pin for the reload timer waveform when the bit is set to “1”.
bit14 to bit12	CKS2 to CKS0: Count clock selection bits	■ These bits are used to select a count clock for the 16-bit down counter. ■ Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS2 to CKS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.
bit11, bit10	Unused bits	■ The read value is “0”. ■ Write “0” to these bits.
bit9, bit8	EGS1, EGS0: Trigger edge selection bits	■ These bits are used to select a valid edge for an input waveform as an external start source and set trigger conditions. ■ When these bits are set to the initial value or “00_B”, no valid edge is selected for the input waveform. Therefore, an external waveform cannot be used for startup. **Note**: When “1” is written to the STRG bit, a software trigger becomes valid, regardless of the settings of EGS1 and EGS0. ■ Modify EGS1 and EGS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.

Figure 15-25. Timer Control Register (Lower Byte of TMCR)



STRG	Software trigger bit
0	Invalid
1	Startup by software

CTEN	Count operation enable bit
0	Stops the operation
1	Enables the operation

MDSE	Mode selection bit
0	Reload mode
1	One-shot mode

OSEL	Output polarity specification bit
0	Normal polarity
1	Inverted polarity

FMD2	FMD1	FMD0	Timer function selection bits
0	0	0	Reset mode
0	0	1	Selects PWM function
0	1	0	Selects PPG function
0	1	1	Selects reload timer function
1	0	0	Selects PWC function
1	0	1	Prohibited settings
1	1	0	
1	1	1	

T32	32-bit timer selection bit
0	16-bit timer mode
1	32-bit timer mode

Table 15-9. Timer Control Register (Lower Byte of TMCR)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit7</td><td>T32:
32-bit timer selection bit</td><td>■ This bit is used to select the 32-bit timer function.
■ 32-bit timer mode will be selected if the T32 bit is set to “1” when the reload timer function has been selected by setting the FMD2, FMD1and FMD0 bits to “011_B”.
■ Modify this bit while the timer is stopped (CTEN=0). Note, however, that the bit can be modified at the same time as “1” is written to the CTEN bit.
→ See Section “15.5 Operation of 32-Bit Mode”.</td></tr>
<tr><td>bit6
to
bit4</td><td>FMD2 to FMD0:
Timer function selection bits</td><td>■ These bits are used to select a timer function.
■ The reload timer function is selected when FMD2, FMD1 and FMD0 bits are set to “011_B”.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit3</td><td>OSEL:
Output polarity specification bit</td><td>■ This bit is used to determine whether to output a level of timer output in its original form or inverted.
■ Combined with MDSE (bit2), this bit generates the following output waveforms.
<table>
<tr><th>MDSE</th><th>OSEL</th><th>Output waveform</th></tr>
<tr><td>0</td><td>0</td><td>Outputs “L” toggle when count starts</td></tr>
<tr><td>0</td><td>1</td><td>Outputs “H” toggle when count starts</td></tr>
<tr><td>1</td><td>0</td><td>“H” rectangular wave during count</td></tr>
<tr><td>1</td><td>1</td><td>“L” rectangular wave during count</td></tr>
</table></td></tr>
<tr><td>bit2</td><td>MDSE:
Mode selection bit</td><td>■ Reload mode is selected when the MDSE bit is set to “0”. When the count value changes from 0000_H to $FFFF_H$, causing an underflow, the reload register value is immediately loaded to the counter, which then continues the count operation.
■ One-shot mode is selected when the MDSE bit is set to “1”. The operation is stopped when the count value changes from 0000_H to $FFFF_H$, causing an underflow.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit1</td><td>CTEN:
Count operation enable bit</td><td>■ This bit is used to enable the operation of the down counter.
■ The counter will be stopped if “0” is written to this bit when the counter operation has been enabled (CTEN=1).</td></tr>
<tr><td>bit0</td><td>STRG:
Software trigger bit</td><td>■ A software trigger will be applied if “1” is written to the STRG bit when the CTEN bit has been set to “1”.
Note:
A software trigger will also be applied if “1” is written to the CTEN and STRG bits at the same time.
■ The STRG bit always reads “0”.
Note:
When “1” is written to the STRG bit, a software trigger becomes valid, regardless of the settings of EGS1 and EGS0.</td></tr>
</table>

Figure 15-26. Status Control Register (STC)

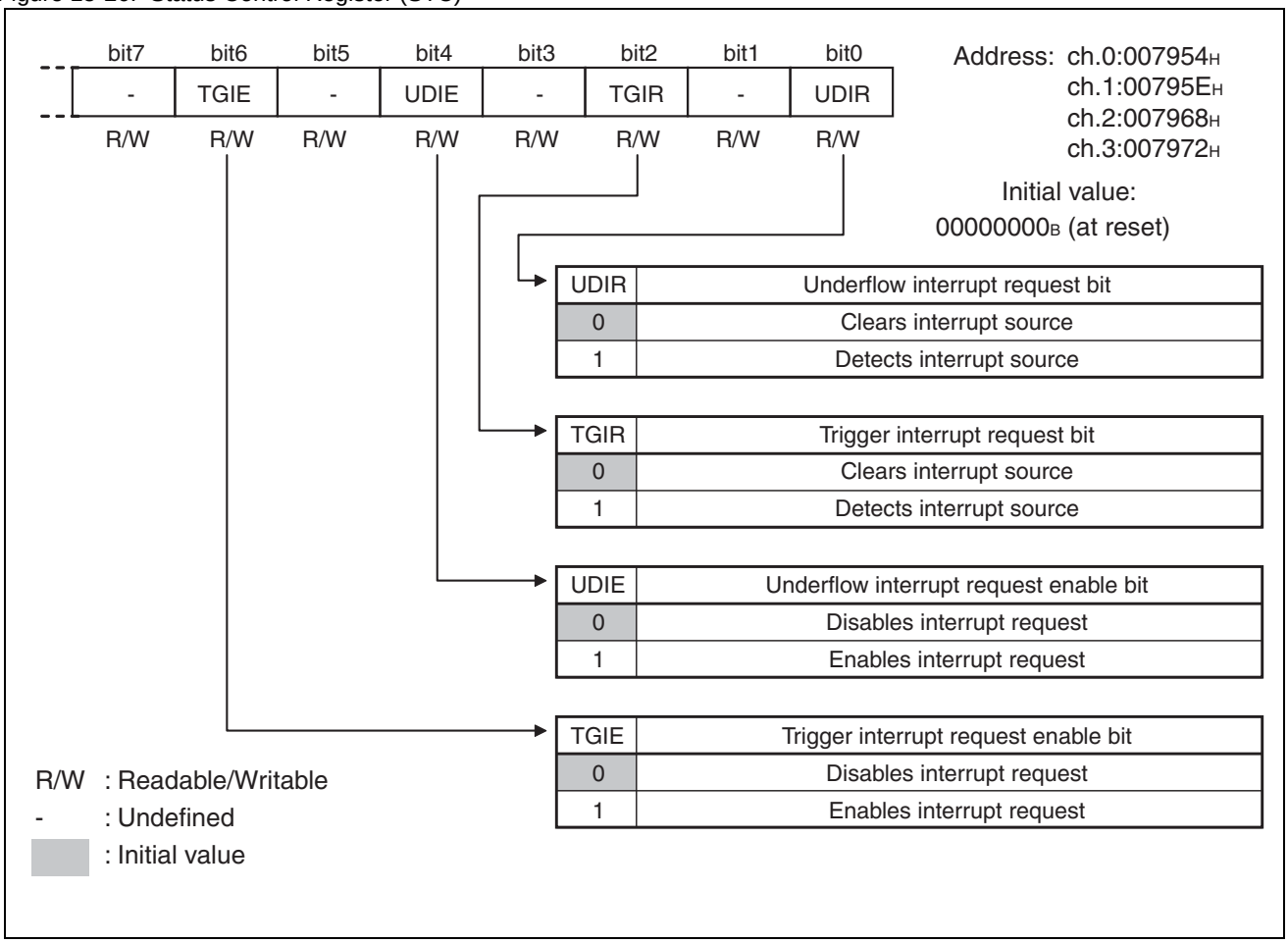


Table 15-10. Status Control Register (STC)

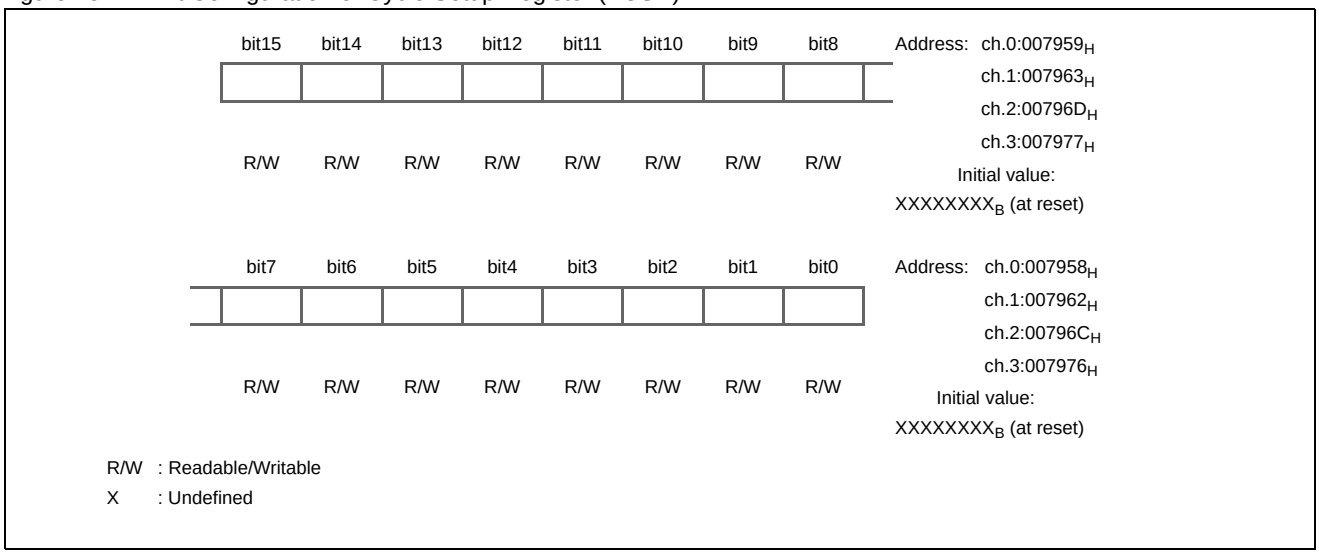
Bit name		Function
bit7	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit6	TGIE: Trigger interrupt request enable bit	■ This bit is used to control interrupt requests for TGIR (bit2). ■ An interrupt request will be generated to the CPU if the TGIR bit (bit2) is set when the TGIE bit is enabled.
bit5	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit4	UDIE: Underflow interrupt request enable bit	■ This bit is used to control interrupt requests for UDIR (bit0). ■ An interrupt request will be generated to the CPU if the UDIR bit (bit0) is set when the UDIE bit is enabled.
bit3	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit2	TGIR: Trigger interrupt request bit	■ The TGIR bit is set to “1” when the input of a software trigger or a trigger is detected. ■ Writing “0” clears the TGIR bit. ■ Writing “1” to the TGIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.
bit1	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit0	UDIR: Underflow interrupt request bit	■ The UDIR bit is set to “1” when the count value changes from 0000_H to $FFFF_H$, causing an underflow. ■ Writing “0” clears the UDIR bit. ■ Writing “1” to the UDIR bit has no effect on the bit value. ■ In read-modify-write (RMW) instructions, the read value is “1”, regardless of the bit value.

Cycle Setup Register (PCSR)

The cycle setup register (PCSR) is used to retain the initial count value. In 32-bit mode, it is set to the initial count value of the lower 16 bits for the even-numbered channel, and set to the initial count value of the upper 16 bits for the odd-numbered channel. The initial reset value is undefined. This register must always be accessed by a 16-bit data transfer instruction.

Figure 15-27 shows the bit configuration of the cycle setup register (PCSR).

Figure 15-27. Bit Configuration of Cycle Setup Register (PCSR)



This register is used to set a cycle. A transfer to the timer register is performed when an underflow occurs.

- The PCSR register should be accessed using 16-bit data.
- Set the PCSR register to select a cycle after setting the reload timer function using the FMD2, FMD1 and FMD0 bits in the TMCR register.
- To write data to the PCSR register in 32-bit mode, access the upper 16-bit data (data in the odd-numbered channel) before accessing the lower 16-bit data (data in the even-numbered channel).

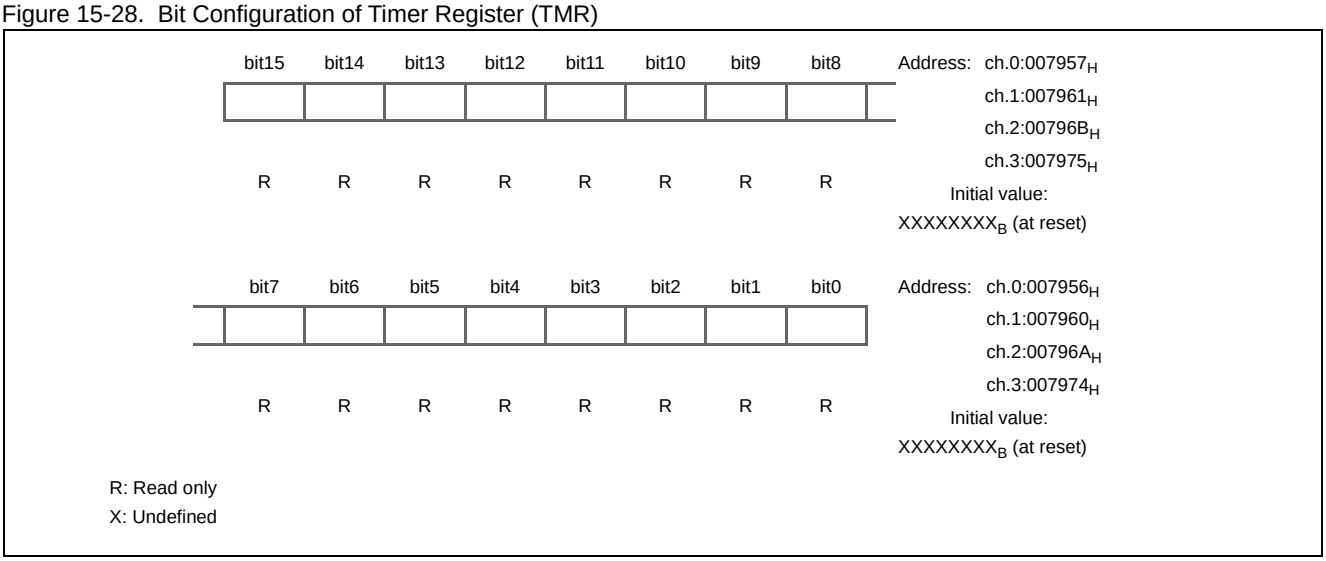
Timer Register (TMR)

The timer register (TMR) can be used to read the count value of the timer. In 32-bit mode, it is set to the count value of the lower 16 bits for the even-numbered channel, and set to the count value of the upper 16 bits for the odd-numbered channel. The initial value is undefined.

This register must always be read by a 16-bit data transfer instruction.

Figure 15-28 shows the bit configuration of the timer register (TMR).

Figure 15-28. Bit Configuration of Timer Register (TMR)



This register can be used to read the value of the 16-bit down counter.

- The TMR register should be accessed using 16-bit data.
- To read from the TMR register in 32-bit mode, read the lower 16-bit data (data in the even-numbered channel) before reading the upper 16-bit data (data in the odd-numbered channel).

Operation of 16-Bit Reload Timer

In reload timer operation, the timer counts down, starting from the value set in the cycle setup register, while being synchronized with the count clock. It stops counting when the count value reaches “0”, or continues its operation until the cycle setting is loaded automatically to stop the down-count operation.

Count Operation for when an Internal Clock is Selected

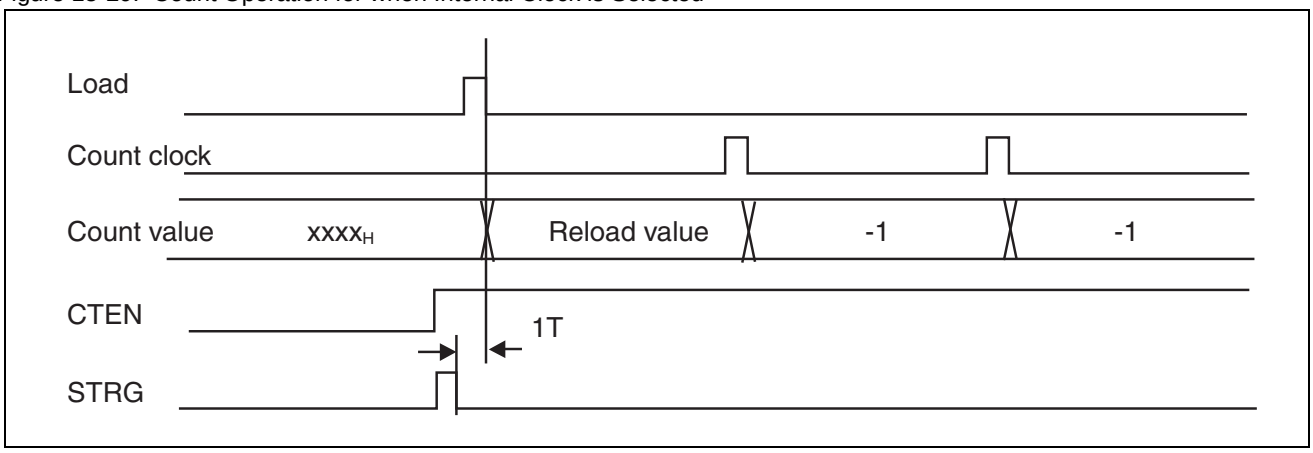
To start a count operation as soon as the count is enabled, write “1” to both the CTEN and STRG bits in the timer control register. As long as the timer is in a startup state (CTEN=1), trigger input by the STRG bit is always valid, regardless of which operation mode has been selected.

The value of the cycle setup register is loaded to the counter and a down-count operation starts, when the count operation is enabled and the timer is started up by a software trigger or an external trigger.

It takes 1T (T: machine cycle) from when a trigger to start the counter is set until the data in the cycle setup register is loaded to the counter.

Figure 15-29 shows how the counter is started up by a software trigger and how it operates.

Figure 15-29. Count Operation for when Internal Clock is Selected



Underflow Operation

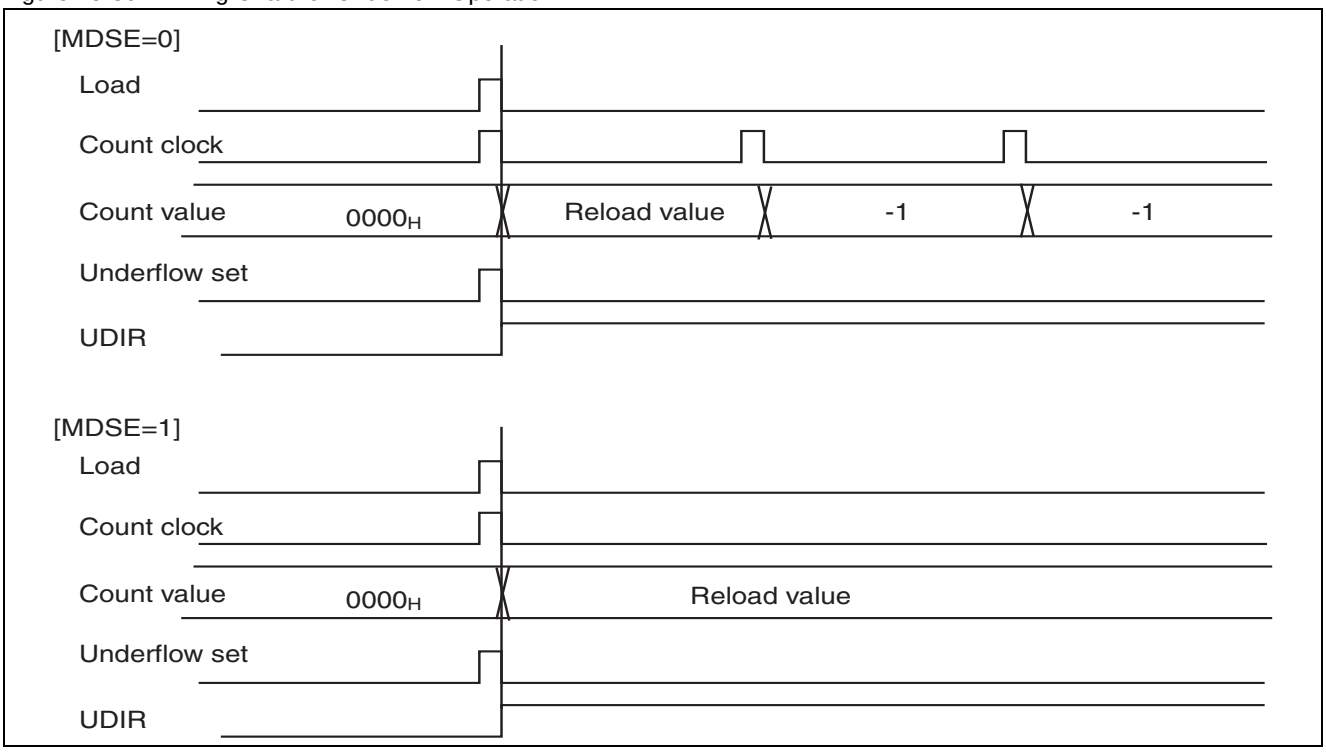
Underflow is an event in which the counter value changes from 0000_H to $FFFF_H$. Therefore, an underflow shall occur when a count reaches [the value set in the cycle setup register + 1].

The count operation will continue, if the MDSE bit in the timer control register (TMCR) contains “0” when the content of the cycle setup register (PCSR) is loaded to the counter upon the occurrence of an underflow. If the MDSE bit contains “1”, the operation will stop, retaining the loaded counter value.

The UDIR bit of status control register (STC) is set by underflow, and an interrupt request is generated when the UDIE bit is “1”.

Figure 15-30 shows a timing chart for underflow operation.

Figure 15-30. Timing Chart for Underflow Operation

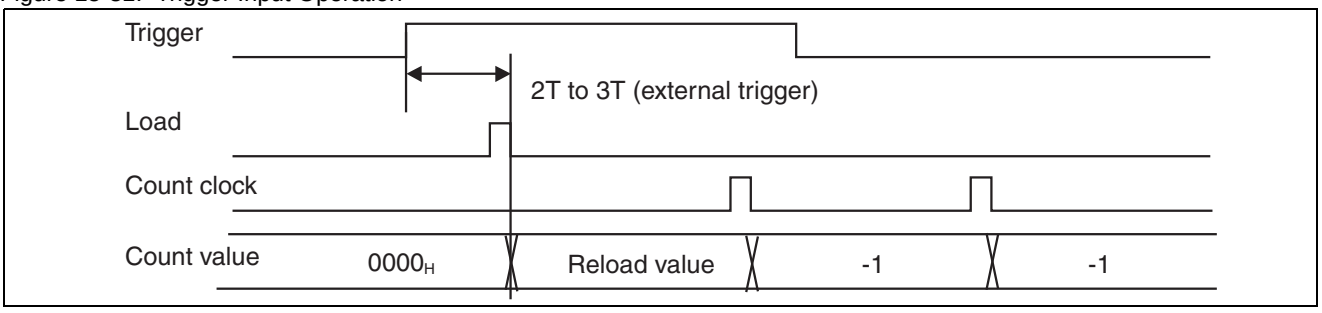


Operation of Input Pin Function

The TIO pin can be used as a trigger input. When a valid edge is input to the TIO pin, the content of the cycle setup register is loaded to the counter, which then starts a count operation. It takes 2T to 3T (T: machine cycle) from when a trigger is entered until the counter value is loaded.

Figure 15-31 shows the trigger input operation for when a rising edge is specified as the valid edge.

Figure 15-31. Trigger Input Operation

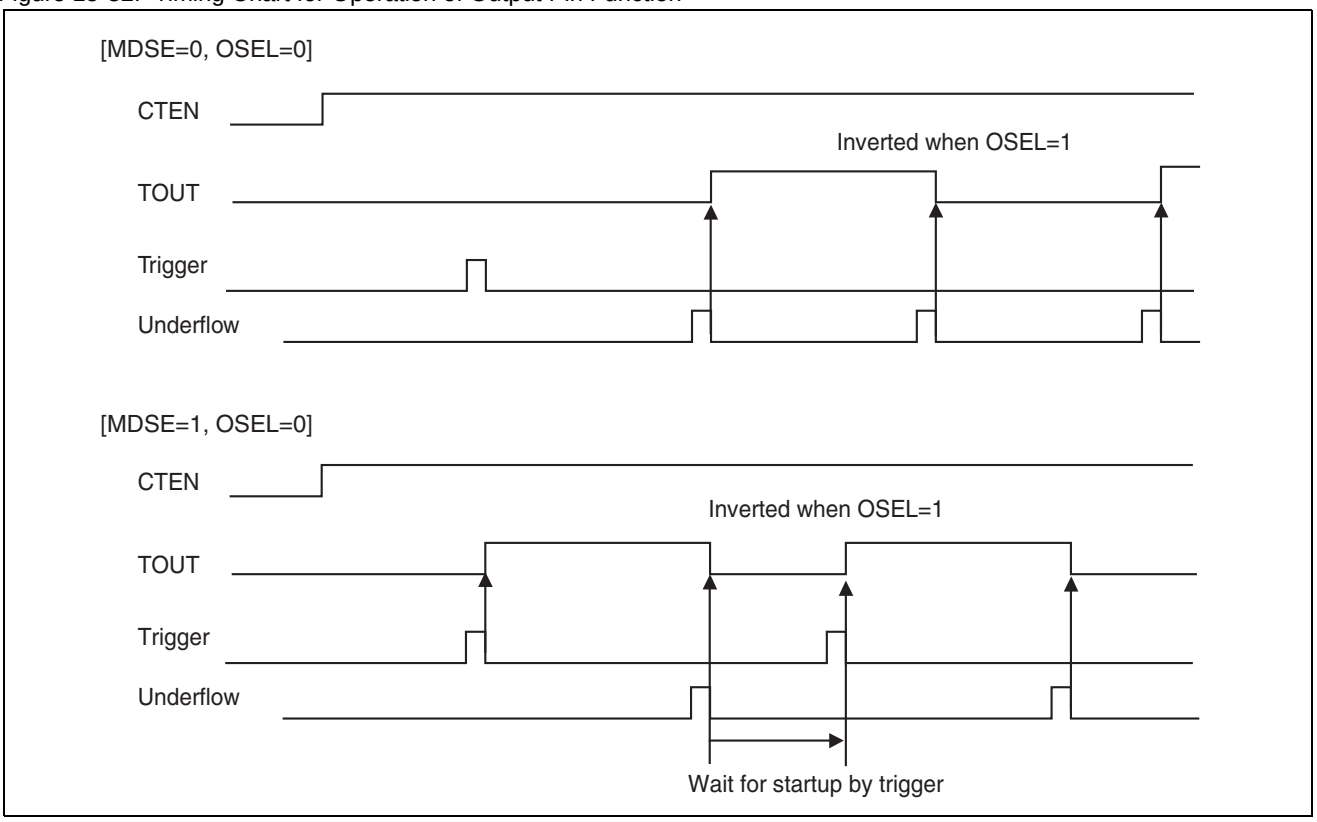


Operation of Output Pin Function

The TIO output pin serves as a toggle output which is inverted by underflow in reload mode, or as a pulse output which indicates that a count is in progress in one-shot mode. The OSEL bit in the timer control register (TMCR) can be used to set the output polarity. When OSEL is set to "0", the initial value is “0” in case of the toggle output, or “1” is output during a count in case of the one-shot pulse output. The output waveform is inverted when OSEL is set to “1”.

Figure 15-32 shows the timing chart for the operation of the output pin function.

Figure 15-32. Timing Chart for Operation of Output Pin Function



15.8.4 PWC Timer Function

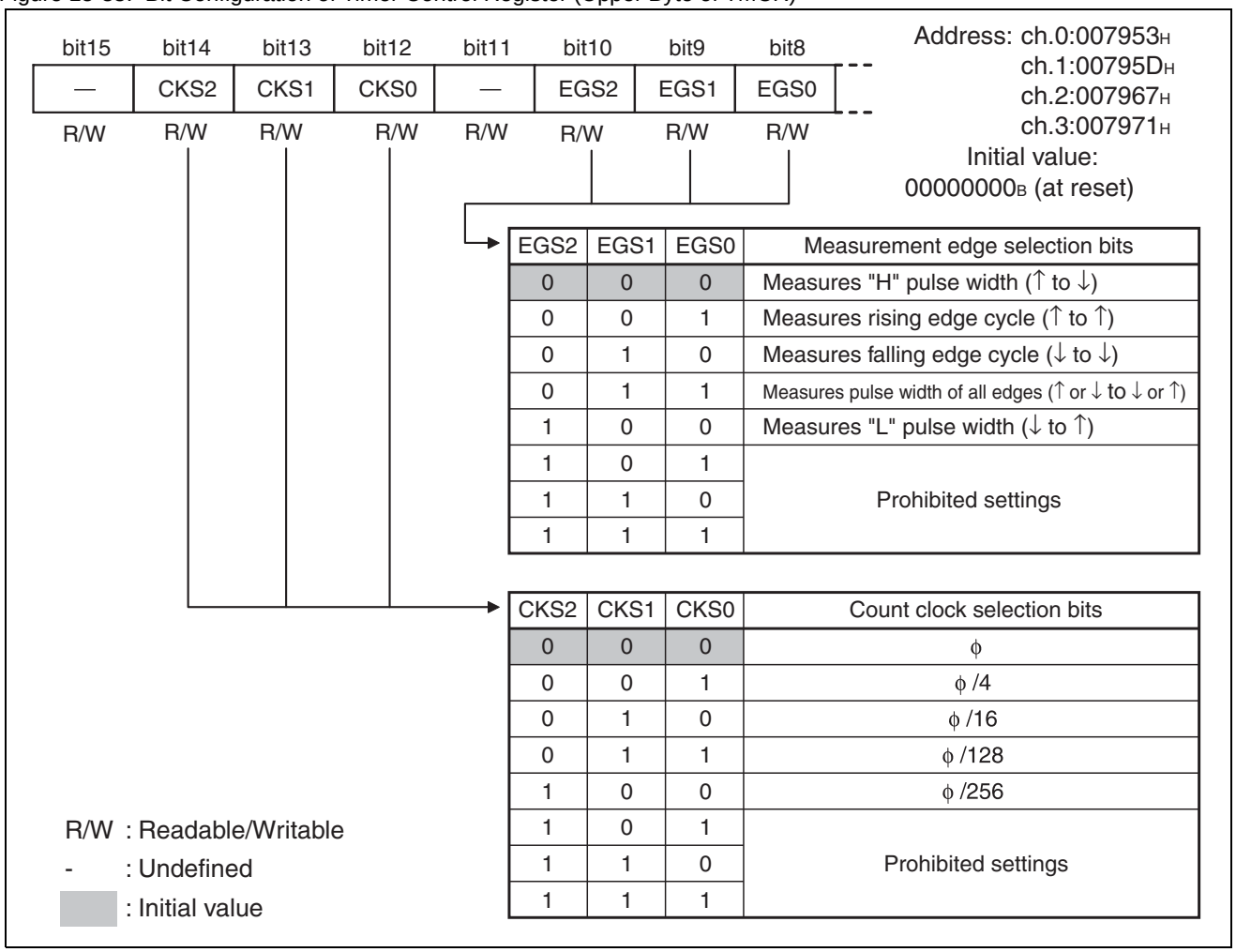
Only one of the following timer functions can be selected for the base timer by setting FMD2, FMD1 and FMD0 bits in the timer control register: 16-bit PWM timer, 16-bit PPG timer, 16/32-bit reload timer, and 16/32-bit PWC timer.

This section describes the timer function when PWC is selected.

Timer Control Register (TMCR) when PWC Timer is Selected

The timer control register (TMCR) is used to control the PWC timer.

Figure 15-33. Bit Configuration of Timer Control Register (Upper Byte of TMCR)



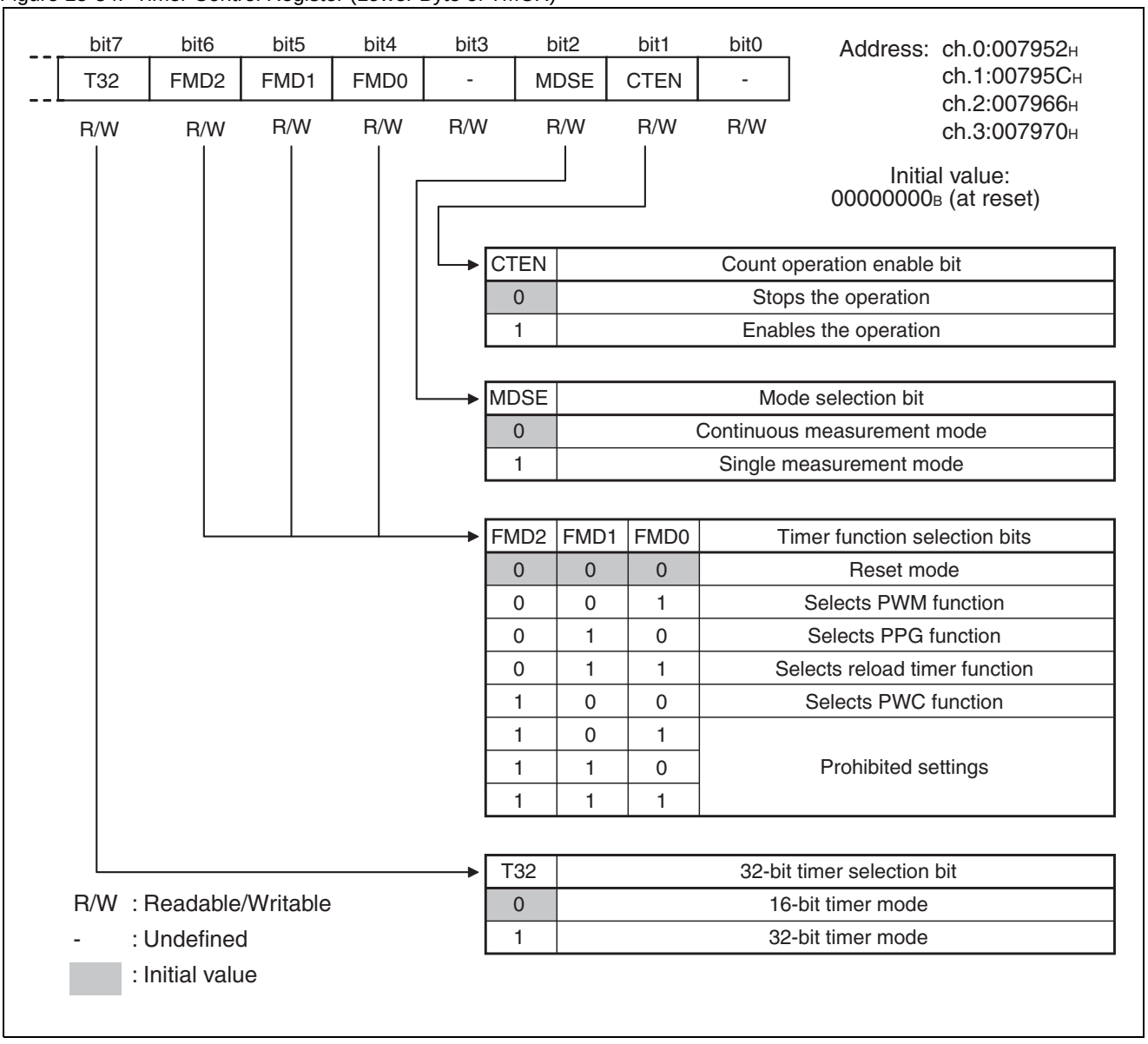
EGS2	EGS1	EGS0	Measurement edge selection bits
0	0	0	Measures "H" pulse width (↑ to ↓)
0	0	1	Measures rising edge cycle (↑ to ↑)
0	1	0	Measures falling edge cycle (↓ to ↓)
0	1	1	Measures pulse width of all edges (↑ or ↓ to ↓ or ↑)
1	0	0	Measures "L" pulse width (↓ to ↑)
1	0	1	Prohibited settings
1	1	0	
1	1	1	

CKS2	CKS1	CKS0	Count clock selection bits
0	0	0	ϕ
0	0	1	ϕ /4
0	1	0	ϕ /16
0	1	1	ϕ /128
1	0	0	ϕ /256
1	0	1	Prohibited settings
1	1	0	
1	1	1	

Table 15-11. Timer Control Register (Upper Byte of TMCR)

Bit name		Function
bit15	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit14 to bit12	CKS2 to CKS0: Count clock selection bits	■ These bits are used to select a count clock for the 16-bit up counter. ■ Any modification to the count clock is reflected immediately after the setting is changed. Therefore, modify CKS2 to CKS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.
bit11	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit10 to bit8	EGS2 to EGS0: Measurement edge selection bits	■ These bits are used to set measurement edge conditions. ■ Therefore, modify EGS2, EGS1 and EGS0 while a count is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as "1" is written to the CTEN bit.

Figure 15-34. Timer Control Register (Lower Byte of TMCR)



CTEN	Count operation enable bit
0	Stops the operation
1	Enables the operation

MDSE	Mode selection bit
0	Continuous measurement mode
1	Single measurement mode

FMD2	FMD1	FMD0	Timer function selection bits
0	0	0	Reset mode
0	0	1	Selects PWM function
0	1	0	Selects PPG function
0	1	1	Selects reload timer function
1	0	0	Selects PWC function
1	0	1	Prohibited settings
1	1	0	
1	1	1	

T32	32-bit timer selection bit
0	16-bit timer mode
1	32-bit timer mode

Table 15-12. Timer Control Register (Lower Byte of TMCR)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit7</td><td>T32:
32-bit timer selection bit</td><td>■ This bit is used to select the 32-bit timer function.
■ 32-bit PWC mode will be selected if the T32 bit is set to “1” when the PWC function has been selected by setting the FMD2, FMD1 and FMD0 bits to “100_B”.
■ Modify this bit while the timer is stopped (CTEN=0). Note, however, that the bit can be modified at the same time as “1” is written to the CTEN bit.
→ See Section “15.5 Operation of 32-Bit Mode”.</td></tr>
<tr><td>bit6
to
bit4</td><td>FMD2 to FMD0:
Timer function selection bits</td><td>■ These bits are used to select a timer function.
■ The PWC function is selected when the FMD2, FMD1 and FMD0 bits are set to “100_B”.
■ Modify the bits while the timer is stopped (CTEN=0). Note, however, that the bits can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit3</td><td>Unused bit</td><td>■ The read value is “0”.
■ Write “0” to this bit.</td></tr>
<tr><td>bit2</td><td>MDSE:
Mode selection bit</td><td>■ The measurement operation is selected, as shown below.
<table>
<tr><th>MDSE</th><th>Mode</th><th>Operation</th></tr>
<tr><td>0</td><td>Continuous measurement</td><td>Continuous measurement: buffer register enabled</td></tr>
<tr><td>1</td><td>Single measurement</td><td>Stops after one measurement session</td></tr>
</table>
■ Modify this bit while the timer is stopped (CTEN=0). Note, however, that the bit can be modified at the same time as “1” is written to the CTEN bit.</td></tr>
<tr><td>bit1</td><td>CTEN:
Count operation enable bit</td><td>■ This bit is used to enable the startup or restart of the up counter.
■ The counter will be restarted and cleared, then wait for an edge to start measurement, if “1” is written to this bit when the counter operation has been enabled (CTEN=1).
■ The counter will be stopped if “0” is written to this bit when the counter operation has been enabled (CTEN=1).</td></tr>
<tr><td>bit0</td><td>Unused bit</td><td>■ The read value is “0”.
■ Write “0” to this bit.</td></tr>
</table>

Figure 15-35. Status Control Register (STC)

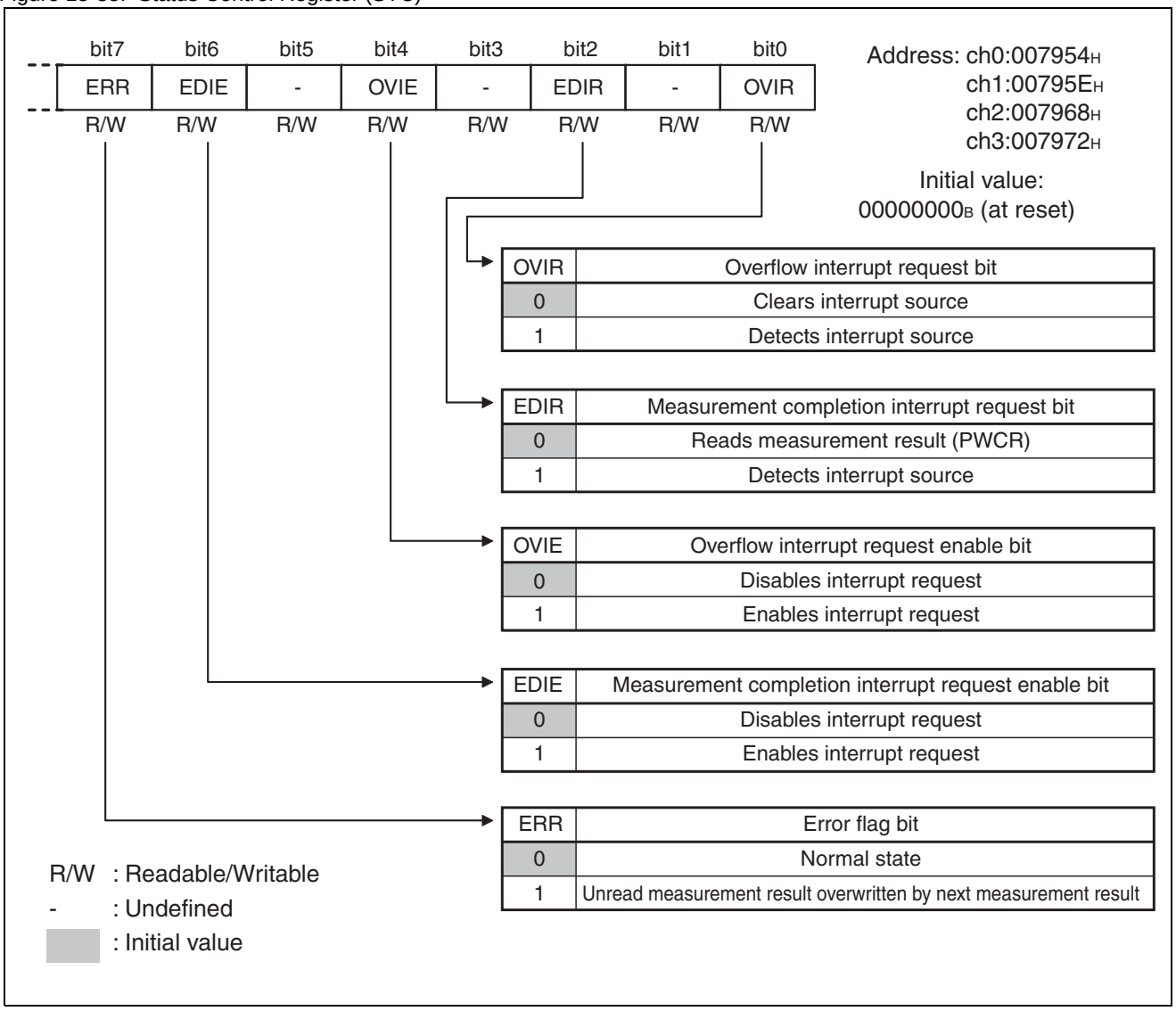


Table 15-13. Status Control Register (STC)

Bit name		Function
bit7	ERR: Error flag bit	■ This flag indicates that the next measurement has already been performed before the measurement result of the DTBF register is read in continuous measurement mode. In this case, the register value is updated with the new measurement result; therefore the immediately preceding result is lost. ■ The measurement continues regardless of the value of the ERR bit. ■ The ERR bit can only be read; therefore, writing to the bit has no effect on the bit value. ■ The ERR bit is cleared once the measurement result (DTBF) is read.
bit6	EDIE: Measurement completion interrupt request enable bit	■ This bit is used to control interrupt requests for EDIR (bit2). ■ An interrupt request will be generated to the CPU if the EDIR bit (bit2) is set when the EDIE bit is enabled.
bit5	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit4	OVIE: Overflow interrupt request enable bit	■ This bit is used to control interrupt requests for OVIR (bit0). ■ An interrupt request will be generated to the CPU if the OVIR bit (bit0) is set when the OVIE bit has been enabled.
bit3	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit2	EDIR: Measurement completion interrupt request bit	■ This bit indicates that measurement is completed and the flag is set to “1” upon the completion. ■ The EDIR bit is cleared once the measurement result (DTBF) is read. ■ The EDIR bit can only be read; therefore, writing to the bit has no effect on the bit value.
bit1	Unused bit	■ The read value is “0”. ■ Write “0” to this bit.
bit0	OVIR: Overflow interrupt request bit	■ The flag is set to “1” when the count value changes from $FFFF_H$ to 0000_H, causing an overflow. ■ Writing “0” clears the OVIR bit. ■ Writing “1” to the OVIR bit has no effect on the bit value. ■ In read-modify-writ (RMW) instructions, the read value is always “1”, regardless of the bit value.

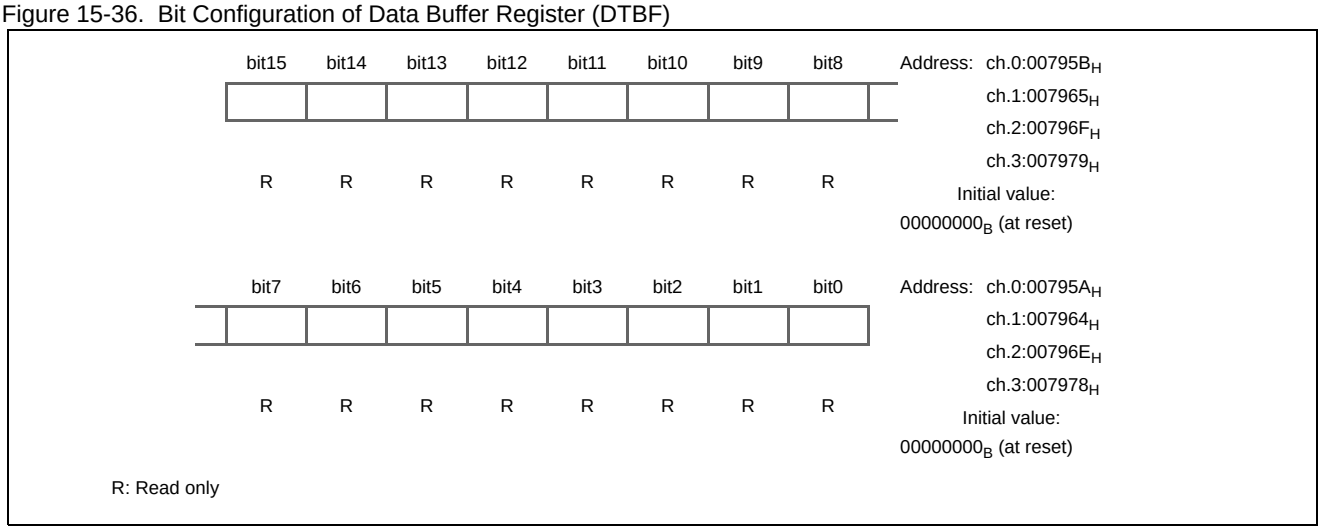
Data Buffer Register (DTBF)

The data buffer register (DTBF) can read the measurement value or count value of the PWC timer. In 32-bit mode, the lower 16 bits are read as the value for the even-numbered channel, and the upper 16 bits are read as the value for the odd-numbered channel.

This register must always be read using a 16-bit data transfer instruction.

Figure 15-36 shows the bit configuration of the data buffer register (DTBF).

Figure 15-36. Bit Configuration of Data Buffer Register (DTBF)



- The DTBF register can only be read in both continuous measurement mode and one-shot measurement mode. Writing to this register does not change the register value.
- In continuous measurement mode (TMCR bit3 MDSE=1), this register functions as a buffer register that retains the previous measurement result.
- In one-shot measurement mode (TMCR bit3 MDSE=0), the DTBF register is used to directly access the up counter. The count value can be read from this register even during a count. When the measurement is finished, the measurement result is held.
- The DTBF register should be accessed using 16-bit data.

Operation of PWC Timer

PWC timer has a pulse width measurement function in which five different count clocks are available and the counter can be used to measure the time and cycle between any events of input pulse. The following describes the basic features and operations of the pulse width measurement function.

Pulse Width Measurement Function

In this function, the counter is cleared to "0000_H" after startup and the count operation does not start until a specified measurement start edge is entered. When a measurement start edge is detected, the counter starts counting up from "0001_H". It stops counting when a measurement stop edge is detected. The count value between these two events is stored as a pulse width in the register.

An interrupt request can be generated upon the completion of measurement and when an overflow occurs.

This function operates upon the completion of measurement, as described below, depending on which measurement mode has been selected.

- Single measurement mode : Stops the operation
- Continuous measurement mode : Transfers the counter value to the buffer register, and stops the count until the next measurement start edge is entered.

Figure 15-37. Pulse Width Measurement Operation (Single Measurement Mode / Measuring "H" Width)

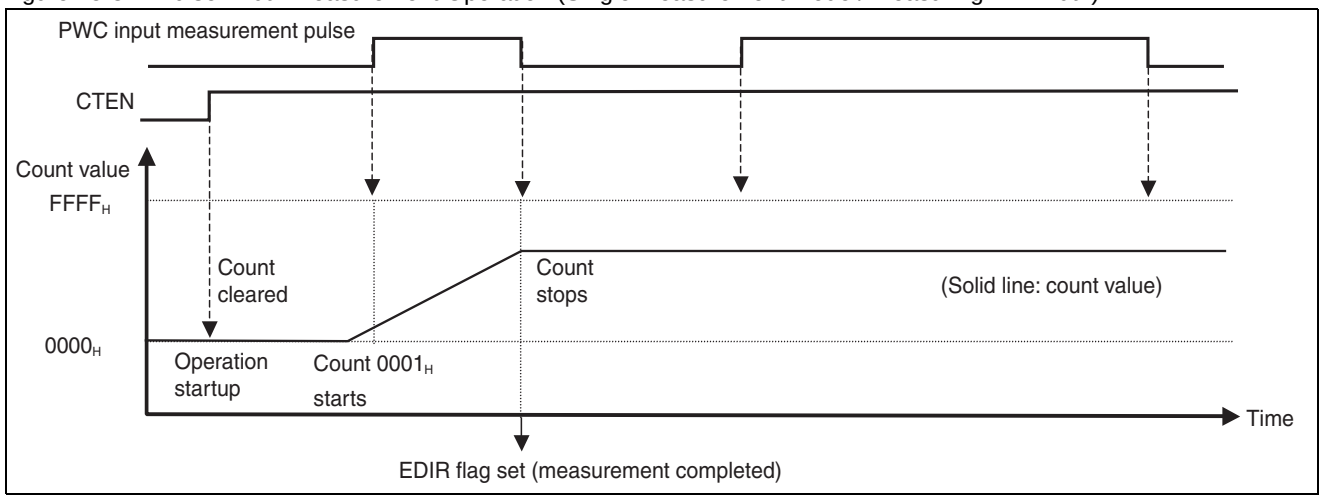
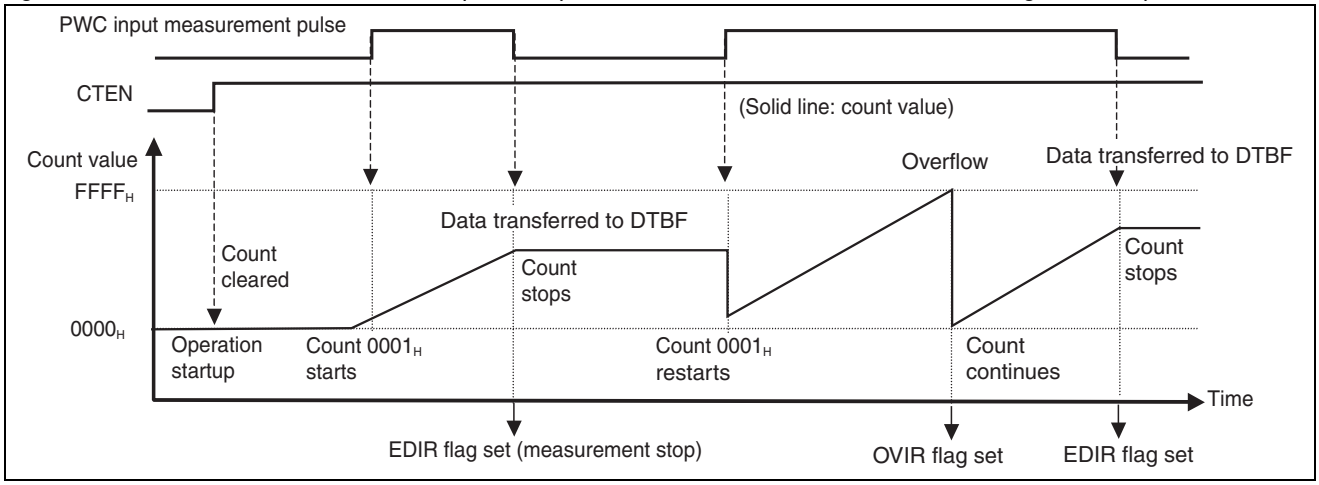


Figure 15-38. Pulse Width Measurement Operation (Continuous Measurement Mode / Measuring "H" Width)



Selecting Count Clock

The count clock of the counter can be selected from five internal clock sources, depending on the settings of bit6, bit5 and bit4 (CKS2, CKS1 and CKS0) in the TMCR register.

The selectable count clocks are as follows.

TMCR register CKS2, CKS1 and CKS0 bits	Selectable internal count clock
000_B	Machine clock [initial value]
001_B	Machine clock divided by 4
010_B	Machine clock divided by 16
011_B	Machine clock divided by 128
100_B	Machine clock divided by 256
101_B	Prohibited settings
110_B	
111_B	

The machine clock is selected as the initial value after a reset.

Note:

Make sure to select a count clock before starting up the counter.

Selecting Operation Mode

TMCR is used to select each operation/measurement mode.

Setting operation mode: TMCR bit10 to bit8: EGS2, EGS1, EGS0 (selection of measurement edge)

Setting measurement mode: TMCR bit2:MDSE (selection of single or continuous measurement)

The table below lists selectable operation modes.

Operation mode		MDSE	EGS2	EGS1	EGS0
↑ to ↓ Measuring “H” pulse width	Continuous measurement: buffer enabled [Initial value]	0	0	0	0
	Single measurement: buffer disabled	1	0	0	0
↑ to ↑ Measuring rising edge cycle	Continuous measurement: buffer enabled	0	0	0	1
	Single measurement: buffer disabled	1	0	0	1
↓ to ↓ Measuring falling edge cycle	Continuous measurement: buffer enabled	0	0	1	0
	Single measurement: buffer disabled	1	0	1	0
↑ or ↓ to ↑ or ↓ Measuring all edges	Continuous measurement: buffer enabled	0	1	1	1
	Single measurement: buffer disabled	1	1	1	1
↓ to ↑ Measuring “L” pulse width	Continuous measurement: buffer enabled	0	1	0	0
	Single measurement: buffer disabled	1	1	0	0
Prohibited settings		0	1	0	1
		1	1	0	1
		0	1	1	0
		1	1	1	0
		0	1	1	1
		1	1	1	1

The combination of the measurement of “H” pulse width and “single measurement mode” is selected as the initial value after reset.

Make sure to select the operation mode before starting up the counter.

Starting and Stopping Measurement of Pulse Width

The CTEN bit (bit1) in TMCR is used to start, restart or forcibly stop each operation.

Writing “1” to the CTEN bit starts or restarts the measurement of the pulse width, while writing “0” to the CTEN bit forces the measurement to stop.

Function	CTEN
Starting/restarting measurement of pulse width	1
Forcibly stopping measurement of pulse width	0

Operation after Startup

The operation in pulse width measurement mode after startup is as follows.

The count does not start until a measurement start edge is entered. Once a measurement start edge is detected, the 16-bit up counter starts counting from "0001_H".

Restart

"Restart" means performing startup during operation once again (Writing "1" to the CTEN bit again while it is set to "1"). The following operation is performed during restart.

The operation is not affected when the counter is waiting for a measurement start edge. During measurement, on the other hand, the counter is cleared to "0000_H" and then waits for the next measurement start edge. At this point, the measurement stop flag (EDIR) will be set and the measurement result will be transferred to DTBF in continuous measurement mode, if restart is performed at the same time as the detection of a measurement stop edge.

Stopping Operation

In single measurement mode, a count operation is stopped automatically when an overflow occurs in the counter or when measurement is completed. Therefore, no special precaution is required. However, the count operation must be forced to stop when continuous measurement mode has been selected, or when it needs to be stopped before automatically stopped.

Clearing the Counter and Its Initial Value

The 16-bit up counter is cleared to "0000_H", as shown below.

- At reset
- When "1" is written to the CTEN bit (bit1) in TMCR (including at restart)

The 16-bit up counter is initialized to "0001_H", upon the detection of a measurement start edge.

Details of Pulse Width Measurement Operation

- Single measurement and continuous measurement

There are two modes for pulse width measurement: performing it only once: or performing it continuously. Each mode is selected by the MDSE bit in TMCR (See "15.5 Operation of 32-Bit Mode"). The differences between the two modes are as follows.

Single measurement mode:

When the first measurement stop edge is entered, the counter stops counting, and the measurement stop flag (EDIR) in STC is set. No further measurement will be performed. Note, however, that the counter starts waiting for start of measurement when restart is performed at the same time.

Continuous measurement mode:

When a measurement stop edge is entered, the counter stops counting, the measurement stop flag (EDIR) in STC is set, and the counter remains stopped until the measurement start edge is entered again. The counter is initialized to "0001_H" and starts measurement, once the measurement start edge is entered again. The measurement result of the counter is transferred to DTBF when the measurement is completed.

Make sure to modify or select the measurement mode while the counter is stopped.

- Measurement result data

There are differences in the handling of measurement results and counter values as well as the function of DTBF between single and continuous measurement modes. The differences of the measurement results between both modes are as follows.

Single measurement mode:

The count value of the current measurement is obtained when DTBF is read during operation.

The measurement result data is obtained when DTBF is read after the measurement is completed.

Continuous measurement mode:

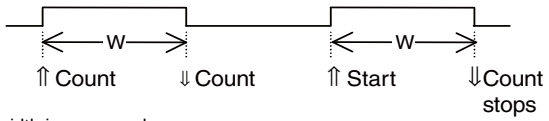
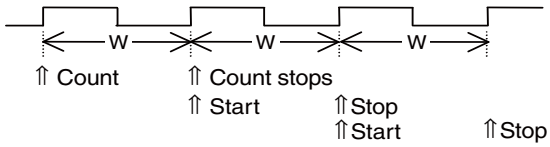
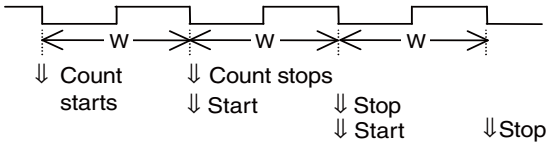
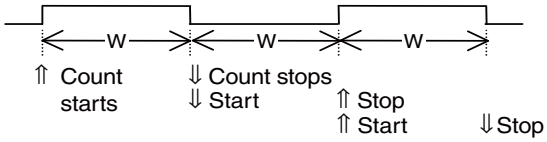
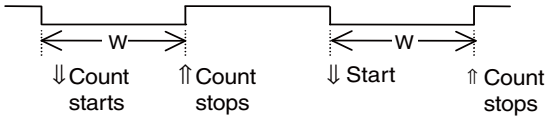
The measurement result in the counter is transferred to DTBF when the measurement is completed.

When DTBF is read, the immediately preceding measurement result is obtained and the previous measurement result is retained even during measurement operation. The count value of the current measurement is not read.

In continuous measurement mode, the previous measurement result will be deleted by the new measurement result, if the next measurement is completed before the measurement result is read. In this case, the error flag (ERR) in STC is set. The error flag (ERR) is cleared automatically when DTBF is read.

■ Measurement mode and count operation

The measurement mode can be selected from five different options, depending on which part of the input pulse should be measured. This is explained below.

Measurement mode	EGS2, EGS1, EGS0	What is measured (W: pulse width to be measured)
Measuring “H” pulse width	000_B	 The “H” cycle width is measured. Count (measurement) starts upon the detection of a rising edge. Count (measurement) stops upon the detection of a falling edge.
Measuring rising edge cycle	001_B	 The rising edge cycle is measured. Count (measurement) starts upon the detection of a rising edge. Count (measurement) stops upon the detection of a rising edge.
Measuring falling edge cycle	010_B	 The falling edge cycle is measured. Count (measurement) starts upon the detection of a falling edge. Count (measurement) stops upon the detection of a falling edge.
Measuring pulse width of all edges	011_B	 The width of the edges that are input continuously is measured. Count (measurement) starts upon the detection of an edge. Count (measurement) stops upon the detection of an edge.
Measuring “L” pulse width	100_B	 The “L” cycle width is measured. Count (measurement) starts upon the detection of a falling edge. Count (measurement) stops upon the detection of a rising edge.

In any measurement mode, the counter is cleared to “0000_H” at the startup of measurement. After that, the counter does not perform a count operation until a measurement start edge is entered. Once a measurement start edge is entered, the counter starts and continues to count up for each count clock until a measurement stop edge is entered.

A stop edge also serves as the next measurement start edge, when the pulse width of all edges or the cycle is measured in continuous measurement mode.

How to calculate pulse width / cycle

The measurement pulse width and cycle can be calculated as shown below, using the measurement result data obtained in DTBF after the measurement is completed.

$T_W = n \times t$ [µs]

T_W: Measurement pulse width / cycle [µs]
n : Measurement result data in DTBF
t: Count clock cycle [µs]

Generating interrupt requests

Two types of interrupt requests can be generated.

Interrupt request by counter overflow

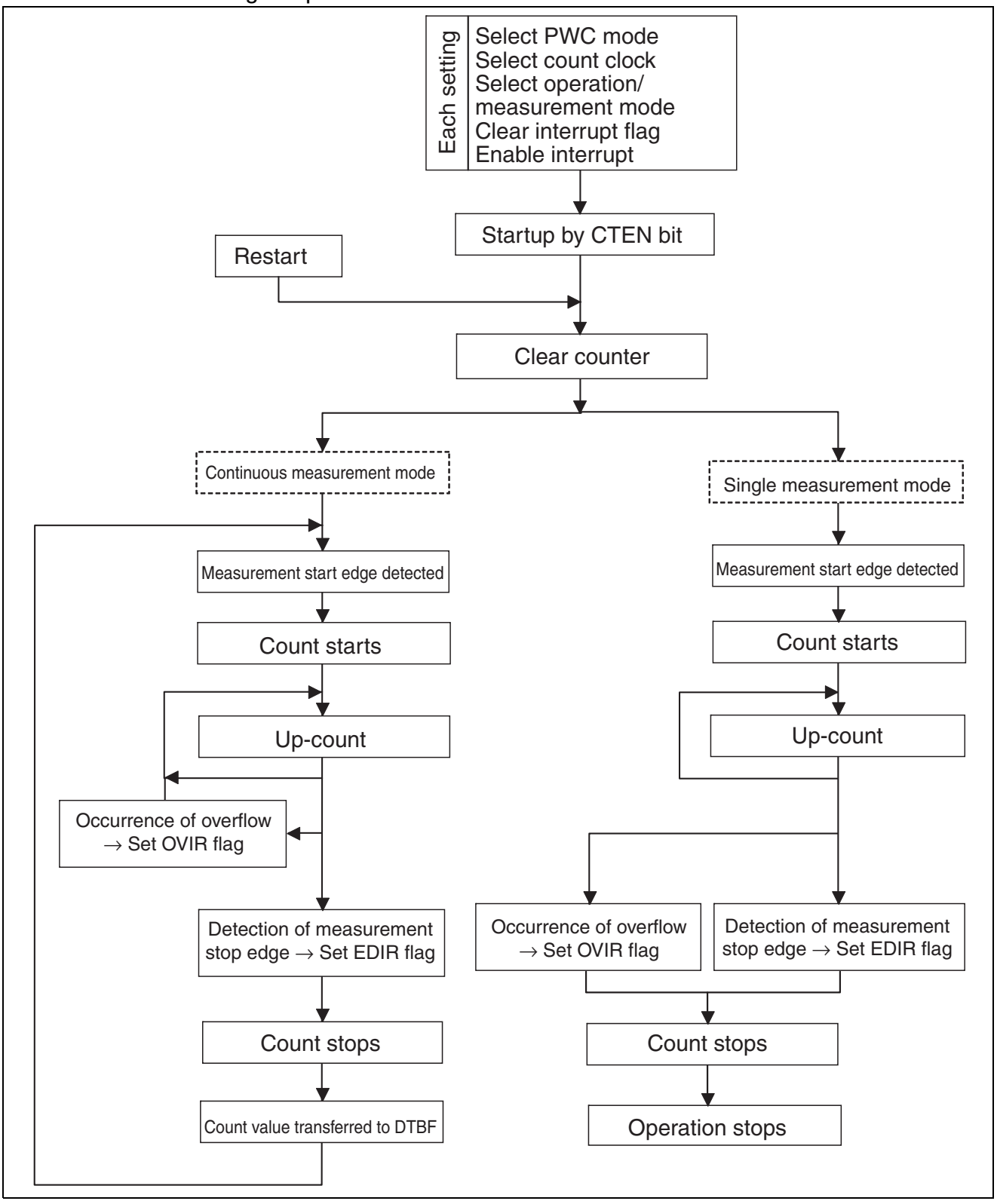
If an overflow occurs due to up-count operation during measurement, the overflow flag (OVIR) will be set. In this case, an interrupt request will be generated if interrupt requests by overflow have been enabled.

Interrupt request by completion of measurement

When a measurement stop edge is detected, the measurement stop flag (EDIR) in STC is set. In this case, an interrupt request will be generated if interrupt requests by completion of measurement have been enabled.

The measurement stop flag (EDIR) is cleared automatically once the measurement result DTBF is read.

Flowchart for measuring the pulse width



16. DTP/External Interrupts

This chapter provides an overview of the DTP/external interrupt unit, explains configuration and functions of its registers and its operation, and shows the precautions on use.

16.1 Overview of DTP/External Interrupt Unit

16.2 Configuration and Functions of DTP/External Interrupt Unit Registers

16.3 DTP/External Interrupt

16.4 Operations of DTP/External Interrupt Circuit

16.5 Precautions on Use of DTP/External Interrupt Unit

16.1 Overview of DTP/External Interrupt Unit

The DTP (Data Transfer Peripheral) unit is a peripheral control section located between the peripheral units outside the device and the F^{2}MC-16LX CPU. It is used to receive DMA request or interrupt requests from the external peripheral device and report such requests to the F^{2}MC-16LX CPU to start μDMAC, EI^{2}OS, or interrupt handling.

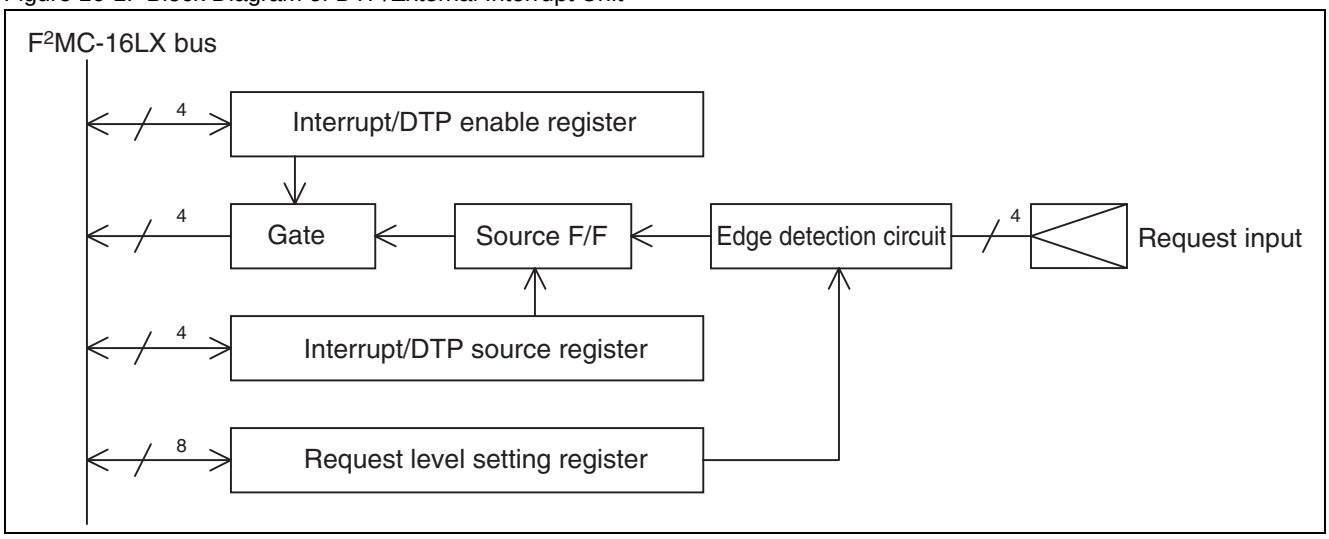
Overview of DTP/External Interrupt Unit

For μDMAC or EI^{2}OS, the request level can be selected from two types, “H” and “L”. For external interrupt requests, it can be selected from four types: rising edge, falling edge, “H” and “L” signals.

Block Diagram of DTP/External Interrupt Unit

Figure 16-1 shows a block diagram of the DTP/external interrupt.

Figure 16-1. Block Diagram of DTP/External Interrupt Unit



Pin Related to DTP/External Interrupt

Pins related to external interrupt pin are IRQ0 to IRQ23 pins and function as input port. IRQ0 to IRQ23 pins are sharing the function with general-purpose input/output ports (P00/IRQ0 to P07/IRQ7, P36/AIN1/IRQ8, P37/BIN1/IRQ9, P71/AN17/IRQ10, P72/AN18/IRQ11, P73/AN19/IRQ12, P74/UI5/IRQ13, P76/UCK5/IRQ14, P80/UI6/IRQ15, P82/UCK6/IRQ16, P83/IRQ17, P87/ADTG/IRQ18, PA0/IRQ19, PA1/IRQ20, PA2/IRQ21, PA3/IRQ22 and P53/IRQ23), external interrupt input pin, analog input pin of A/D, trigger input of A/D or multi-function serial input/output pin.

Notes:

The external interrupt pins shared with external bus pins such as P00/AD00/D00/IRQ0 to P07/AD07/D07/IRQ7, P36/A06/AIN1/IRQ8, P37/A07/BIN1/IRQ9, and P53/$\overline{\text{WRH}}$/IRQ23 cannot be used as external interrupt pins in the following cases:

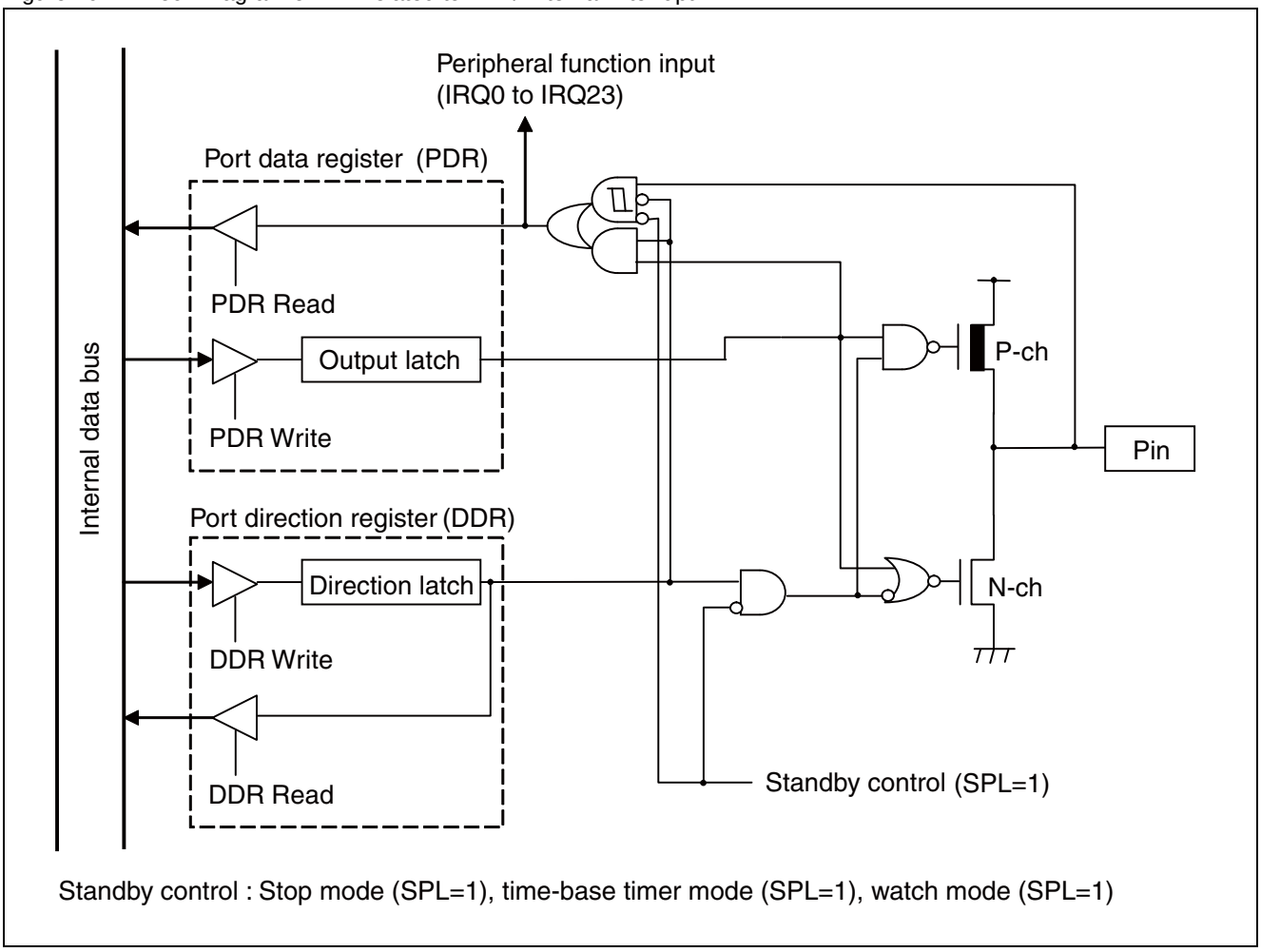
- P00/AD00/D00/IRQ0 to P07/AD07/D07/IRQ7, P53/$\overline{\text{WRH}}$/IRQ23
  In the external bus mode, these pins cannot be used as external interrupt pins.
- P53/$\overline{\text{WRH}}$/IRQ23
  In the external bus mode, this pin cannot be used as an external interrupt pin even if it is switched to the I/O port by the WRE bit of bus control signal selection register (EPCR).
- P36/A06/AIN1/IRQ8, P37/A07/BIN1/IRQ9
  In the external bus mode, these pins cannot be used as external interrupt pins when transiting to the time base timer mode, the watch mode, and the stop mode.

Setting when using as IRQ0 to IRQ23 pins

When the P00/IRQ0 to P07/IRQ7, P36/IRQ8, P37/IRQ9, P71/IRQ10, P72/IRQ11, P73/IRQ12, P74/IRQ13, P76/IRQ14, P80/IRQ15, P82/IRQ16, P83/IRQ17, P87/IRQ18, PA0/IRQ19, PA1/IRQ20, PA2/IRQ21, PA3/IRQ22 and P53/IRQ23 pins are used as an input pin, be sure to set the port direction register to the input port (DDR0: bit0 to bit7, DDR3 bit6, bit7, DDR7 bit6, bit7, bit4 to bit1, bit6, bit7, DDR8 bit0, bit2, bit7, DDRA bit0 to bit3 and DDR5 bit3 → “0”).

Block Diagram of Pin Related to DTP/External Interrupt

Figure 16-2. Block Diagram of Pin Related to DTP/External Interrupt



16.2 Configuration and Functions of DTP/External Interrupt Unit Registers

This section describes the configuration and functions of the registers used in the DTP/external interrupt unit.

List of Registers for DTP/External Interrupt Unit

Figure 16-3 shows a list of the registers for the DTP/external interrupt unit.

Figure 16-3. List of DTP/External Interrupt Unit Registers

	bit 7	6	5	4	3	2	1	0	
Address: 0000E0$_H$	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	Interrupt/DTP enable register (ENIR0)
	bit 15	14	13	12	11	10	9	8	
Address: 0000E1$_H$	ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	Interrupt/DTP source register (EIRR0)
	bit 7	6	5	4	3	2	1	0	
Address: 0000E2$_H$	LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	Request level setting register (ELVR0)
	bit 15	14	13	12	11	10	9	8	
Address: 0000E3$_H$	LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	Request level setting register (ELVR0)
	bit 7	6	5	4	3	2	1	0	
Address: 0000E4$_H$	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	Interrupt/DTP enable register (ENIR1)
	bit 15	14	13	12	11	10	9	8	
Address: 0000E5$_H$	ER15	ER14	ER13	ER12	ER11	ER10	ER9	ER8	Interrupt/DTP source register (EIRR1)
	bit 7	6	5	4	3	2	1	0	
Address: 0000E6$_H$	LB11	LA11	LB10	LA10	LB9	LA9	LB8	LA8	Request level setting register (ELVR1)
	bit 15	14	13	12	11	10	9	8	
Address: 0000E7$_H$	LB15	LA15	LB14	LA14	LB13	LA13	LB12	LA12	Request level setting register (ELVR1)
	bit 7	6	5	4	3	2	1	0	
Address: 0000E8$_H$	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16	Interrupt/DTP enable register (ENIR2)
	bit 15	14	13	12	11	10	9	8	
Address: 0000E9$_H$	ER23	ER22	ER21	ER20	ER19	ER18	ER17	ER16	Interrupt/DTP source register (EIRR2)
	bit 7	6	5	4	3	2	1	0	
Address: 0000EA$_H$	LB19	LA19	LB18	LA18	LB17	LA17	LB16	LA16	Request level setting register (ELVR2)
	bit 15	14	13	12	11	10	9	8	
Address: 0000EB$_H$	LB23	LA23	LB22	LA22	LB21	LA21	LB20	LA20	Request level setting register (ELVR2)

Interrupt/DTP Enable Register (ENIR)

The bit configuration of the interrupt/DTP enable register (ENIR) is shown below.

	bit	7	6	5	4	3	2	1	0	Initial value
ENIR0 Address: $0000E0_H$		EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ENIR1 Address: $0000E4_H$	bit	7	6	5	4	3	2	1	0	Initial value
		EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ENIR2 Address: $0000E8_H$	bit	7	6	5	4	3	2	1	0	Initial value
		EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The interrupt/DTP enable register (ENIR) enables or disables an external interrupt/DTP request for an external interrupt/DTP channel.

If the interrupt/DTP enable bits (ENIR:EN) and the interrupt/DTP request flag bits (EIRR:EN) are all set to “1”, an interrupt request for the corresponding interrupt/DTP pin is generated. Signal inputs to this register are not interrupted during standby mode.

Note:

Clear the corresponding DTP/external interrupt factor bit (EIRR:ER) just before enabling the DTP/external interrupts (ENIR:EN = 1).

Interrupt/DTP Source Register (EIRR)

The bit configuration of the interrupt/DTP source register (EIRR) is shown below.

	bit	15	14	13	12	11	10	9	8	Initial value
EIRR0 Address: $0000E1_H$		ER7	ER6	ER5	ER4	ER3	ER2	ER1	ER0	XXXXXXXX_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EIRR1 Address: $0000E5_H$	bit	15	14	13	12	11	10	9	8	Initial value
		ER15	ER14	ER13	ER12	ER11	ER10	ER9	ER8	XXXXXXXX_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EIRR2 Address: $0000E9_H$	bit	15	14	13	12	11	10	9	8	Initial value
		ER23	ER22	ER21	ER20	ER19	ER18	ER17	ER16	XXXXXXXX_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The interrupt/DTP source register (EIRR) is set to “1” if the edge or level signal set in the detection condition selection bits of the request level setting register (ELVR:LB, LA) is input to the external interrupt pin.

If the register is set to “1”, an interrupt request for the corresponding interrupt/DTP channel is generated when the interrupt/DTP request enable bits (ENIR:EN) are set to “1”.

If the register is set to “0”, it is cleared.

If the register is set to “1”, the interrupt request status is not affected.

Notes:

- Reading by read-modify-write (RMW) instructions always read “1”. If multiple external interrupt request outputs are enabled (ENIR: EN7 to EN0=1), only the bits for which the CPU accepts an interrupt (bits for which “1” was set in EN7 to EN0) are cleared to “0”. No other bits must be cleared unconditionally.
- The value of the DTP/external interrupt factor bit (EIRR:ER) is available only when the corresponding DTP/external interrupt enable bit (ENIR:EN) is set to “1”. When the DTP/external interrupt is not enabled (ENIR:EN = 0), the DTP/external interrupt factor bit may be set regardless of whether the DTP/external interrupt factor exists or not.
- Clear the corresponding DTP/external interrupt factor bit (EIRR:ER) just before enabling the DTP/external interrupts (ENIR:EN = 1).

Request Level Setting Register (ELVR)

The bit configuration of the request level setting register (ELVR) is shown below.

	bit	7	6	5	4	3	2	1	0	Initial value
ELVR0		LB3	LA3	LB2	LA2	LB1	LA1	LB0	LA0	00000000_B
Address: $0000E2_H$		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ELVR0	bit	15	14	13	12	11	10	9	8	Initial value
Address: $0000E3_H$		LB7	LA7	LB6	LA6	LB5	LA5	LB4	LA4	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ELVR1	bit	7	6	5	4	3	2	1	0	Initial value
Address: $0000E6_H$		LB11	LA11	LB10	LA10	LB9	LA9	LB8	LA8	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ELVR1	bit	15	14	13	12	11	10	9	8	Initial value
Address: $0000E7_H$		LB15	LA15	LB14	LA14	LB13	LA13	LB12	LA12	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ELVR2	bit	7	6	5	4	3	2	1	0	Initial value
Address: $0000EA_H$		LB19	LA19	LB18	LA18	LB17	LA17	LB16	LA16	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
ELVR2	bit	15	14	13	12	11	10	9	8	Initial value
Address: $0000EB_H$		LB23	LA23	LB22	LA22	LB21	LA21	LB20	LA20	00000000_B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

The request level setting register (ELVR) is used to select a request detection level. Two bits are assigned for each pin, as shown in Table 16-1. If the setting for a request input indicates a level, the corresponding level will be set again when it is cleared, provided the input is active.

Table 16-1. ELVR Assignment (LA0 to LA23, LB0 to LB23)

LBx	LAx	Operation
0	0	Request by “L” level
0	1	Request by “H” level
1	0	Request by rising edge
1	1	Request by falling edge

16.3 DTP/External Interrupt

The interrupt related to the DTP/external interrupt occurs when the edge or level input to input pin is detected.
The DTP/external interrupt can activate the DMA transfer and extended intelligent I/O service (EI^{2}OS).

DTP/External Interrupt

The interrupt control bit and interrupt source of the DTP/external interrupt is shown in the following table.

	External interrupt When ISE of ICR = 0	DTP interrupt when ISE of ICR = 1
Interrupt request flag	EIRR:ER (bit8 to bit15)	EIRR:ER (bit8 to bit15)
Interrupt request output enable bit	ENIR:EN (bit0 to bit7)	ENIR:EN (bit0 to bit7)
Interrupt generation source	Detect external interrupt	Detect external interrupt

Setting procedure

To use the DTP/external interrupt, set each register by using the following procedures:

1. Set the interrupt request enable bit corresponding to the DTP/external interrupt channel to be used to “0” (ENIR:EN).
2. Use the detection condition select bit corresponding to the DTP/external interrupt channel to be used to set the edge or level to be detected (ELVR:LA/LB).
3. Set the interrupt request flag corresponding to the DTP/external interrupt channel to be used to “0” (EIRR:ER).
4. Set the corresponding interrupt request enable bit to “1” (ENIR:EN).

Notes:

- When setting the registers for the DTP/external interrupt, the external interrupt request must be disabled (ENIR:EN=0).
- When enabling the DTP/external interrupt (ENIR:EN=1), the corresponding DTP/external interrupt request flag bit must be cleared in advance (EIRR:ER=0). These actions prevent the mistaken interrupt request from occurring when setting the register.

Selecting of DTP function or external interrupt function

Whether the DTP function or the external interrupt function is executed depends on the setting of the EI^{2}OS enable bit in the corresponding interrupt control register (ICR:ISE) or that of the DMA enable register (DER:EN).

Setting the SE bit to “1” enables extended intelligent I/O service (EI^{2}OS); setting the EN bit to “1” enables DMA transfer.

If the ISE and EN bits are set to “0”, the EI^{2}OS and DMA transfer are disabled and the external interrupt function is executed.

Notes:

- All interrupt requests assigned to one interrupt control register have the same interrupt level (IL2 to IL0).
- If two or more interrupt requests are assigned to one interrupt control register and EI^{2}OS is started for any of them, other interrupt requests cannot be used.

DTP/External Interrupt, DMA Transfer, and EI^{2}OS

Table 16-2 shows the relationship between the interrupt source, interrupt vector, and interrupt control register other than software interrupt.

Table 16-2. Interrupt Source, Interrupt Vector, and Interrupt Control Register

Interrupt source	EI^{2}OS clear	µDMAC channel number	Interrupt vector		Interrupt control register	
			Num-ber	Address	Number	Address
INT0 (IRQ0/IRQ1)	❍	0	#11	$FFFFD0_H$	ICR00	$0000B0_H$
INT1 (IRQ2 to IRQ7)	❍	-	#12	$FFFFCC_H$		
INT2 (IRQ8 to IRQ15)	❍	-	#13	$FFFFC8_H$	ICR01	$0000B1_H$
INT3 (IRQ16 to IRQ23)	❍	-	#14	$FFFFC4_H$		

❍ :Interrupt request flag is cleared.

Note: If there are two interrupt sources in the same interrupt number, resource clears both interrupt request flags. Therefore, when one of two sources uses the EI^{2}OS/µDMAC function, the other interrupt function cannot use. The interrupt request enable bit of the relevant resource is set to “0” to execute the software polling processing.

Correspondence to DMA Transfer and EI^{2}OS Function

DTP/external interrupt supports EI^{2}OS function, and each of ch.0/ch.1 supports DMA transfer function. When the DMA or EI^{2}OS function is used, it is necessary to disable other interrupt that shares the interrupt control register (ICR).

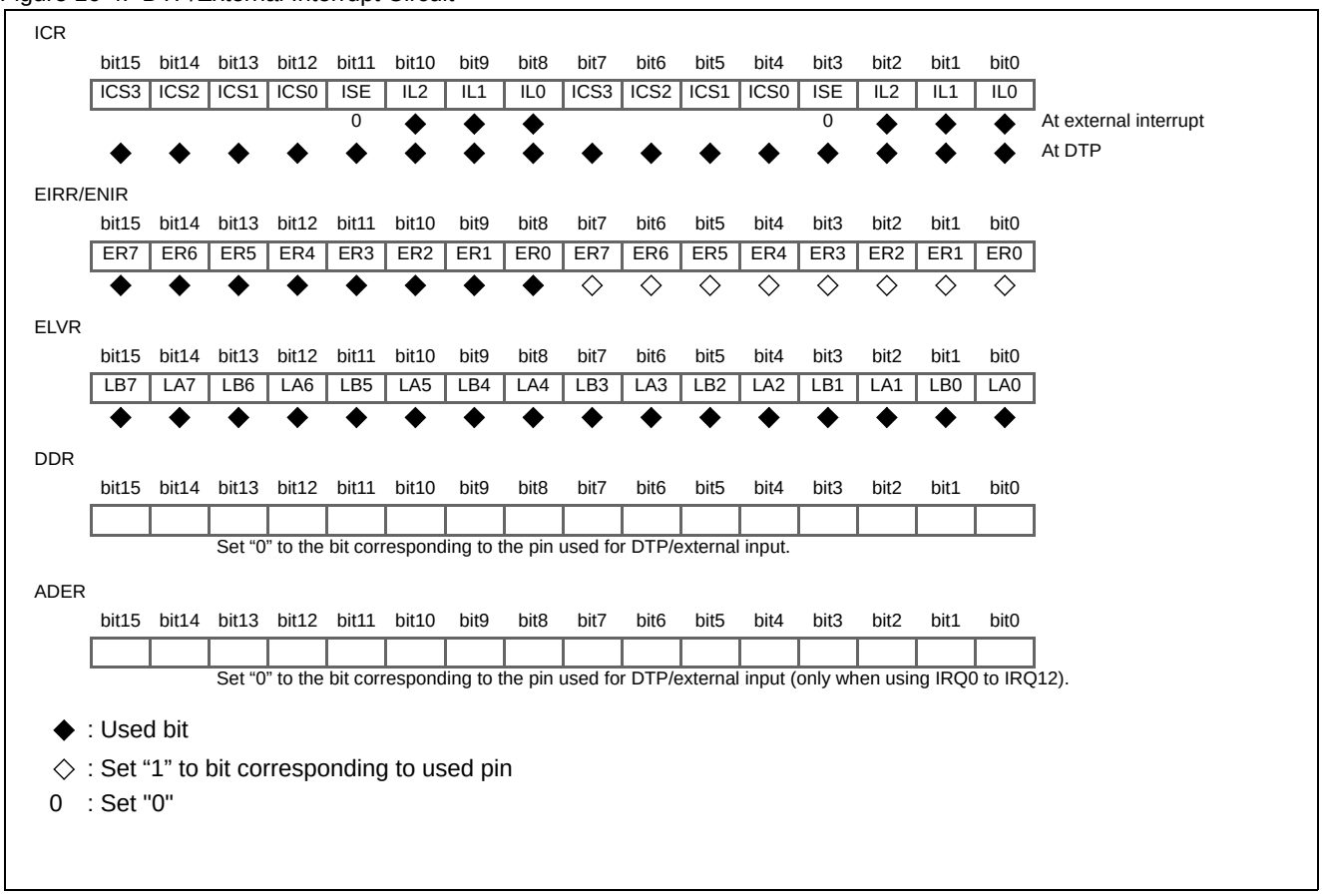
16.4 Operations of DTP/External Interrupt Circuit

The DTP/external interrupt circuit has external interrupt and DTP functions. This section describes the settings for each function and its operation.

Setting DTP/External Interrupt Circuit

The settings in Figure 16-4 are required for the DTP/external interrupt circuit to work.

Figure 16-4. DTP/External Interrupt Circuit



Take the following steps to set the registers of the DTP/external interrupt circuit:

1. Use the relevant bit in the interrupt/DTP enable register (ENIR) to disable interrupts.
2. Set the relevant bit in the request level setting register (ELVR).
3. Clear the relevant bit in the interrupt/DTP source register (EIRR).
4. Use the relevant bit in the interrupt/DTP enable register (ENIR) to enable interrupts.

When setting the registers of the DTP/external interrupt circuit, disable the external interrupt request output first (ENIR: EN23 to EN0 = 0). Before enabling the external interrupt request output (ENIR: EN23 to EN0 = 1), clear the corresponding interrupt request flag (EIRR: ER23 to ER0 = 0).

This prevents an interrupt request from occurring accidentally during register setting.

Switching between the external interrupt function and DTP function

Whether the external interrupt function or DTP function is executed depends on the setting of the EI^{2}OS enable bit in the corresponding interrupt control register (ICR: ISE) and the DMA enable register setting (DER: EN).

Setting the SE bit to “1” enables extended intelligent I/O service (EI^{2}OS); setting the EN bit to “1” enables DMA transfer.

Setting the ISE and EN bit to “0” disables EI^{2}OS and DMA transfer, causing the external interrupt function to be executed.

Note:

If two or more interrupt requests are assigned to one ICR register, all the interrupt requests have a common interrupt level (IL2 to IL0). When EI^{2}OS is used for one interrupt request, in principle, it cannot be used for any other interrupt request.

Operation of DTP/External Interrupt

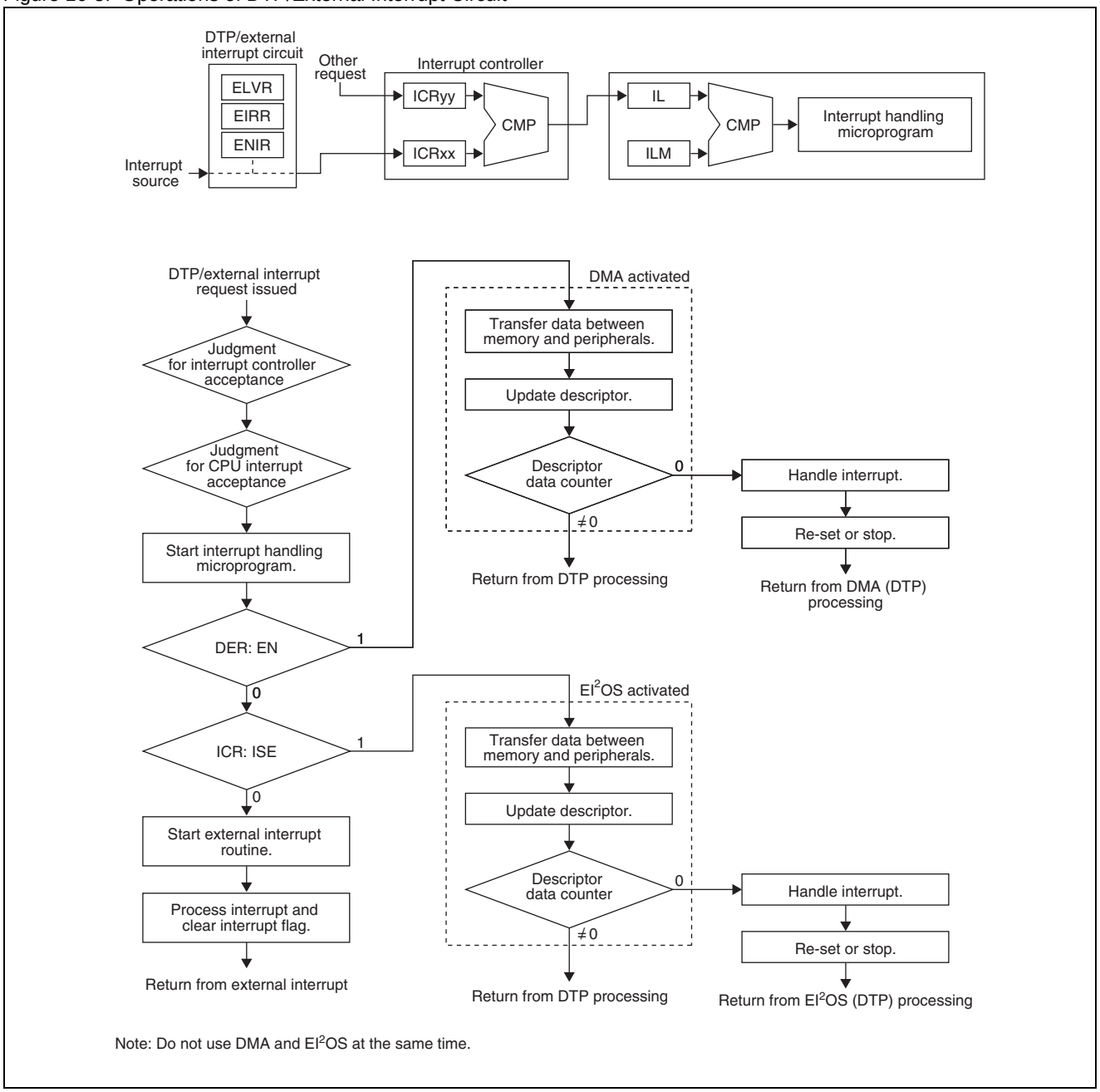
Table 16-3 shows the control bits and interrupt source of the DTP/external interrupt circuit.

Table 16-3. Control Bits and Interrupt Source of DTP/External Interrupt Circuit

	DTP/external interrupt circuit
Interrupt request flag bits	EIRR: ER23 to ER0
Interrupt request enable bits	ENIR: EN23 to EN0
Interrupt source	Input of effective edge/level to the INT7 to INT0 pins

When a DTP/external interrupt request is output to the interrupt controller, interrupt servicing is executed if the EI^{2}OS enable bit in the interrupt control register (ICR: ISE) and the DMA enable register (DER: EN) contain “0”. If either of the bits is “1”, extended intelligent I/O service (EI^{2}OS) or DMA transfer is executed.

Figure 16-5. Operations of DTP/External Interrupt Circuit



Note: Do not use DMA and EI^{2}OS at the same time.

16.4.1 External Interrupt Function

The DTP/external interrupt circuit has the external interrupt function for generating an interrupt request upon input of a selected signal level to the DTP/ external interrupt pin.

External Interrupt Function

When the signal (edge or level) selected in the request level setting register (ELVR) is detected at a DTP/external interrupt pin, the ER23 to ER0 bits in the DTP/interrupt source register (EIRR) are set to “1”. If the interrupt request enable bit in the DTP/interrupt enable register has been set (ENIR: EN23 to EN0 = 1) to enable interrupts, an interrupt request is issued to the interrupt controller. The interrupt controller checks the interrupt levels (ICR: IL2 to IL0) of the interrupt requests from other peripheral resources and their interrupt priorities in case of simultaneous occurrence. The CPU compares the interrupt level with the interrupt level mask register (PS: ILM2 to ILM0) and checks the interrupt enable bit (PS: CCR: I). As soon as the interrupt request is accepted by the CPU, the CPU internally services the interrupt (using the microprogram) and causes a branch to the interrupt service routine.

The interrupt service routine must write “0” to the corresponding interrupt request flag bit to clear the interrupt request.

Note:

The ER bit is set to “1” as a DTP/external interrupt trigger event occurs irrespective of the setting of the EN bit.

When the interrupt routine is activated, clear the bit as the trigger event. The interrupt routine does not return control with the ER bit containing “1”. In this case, be careful not to clear any other flag bit unconditionally.

16.4.2 DTP Function

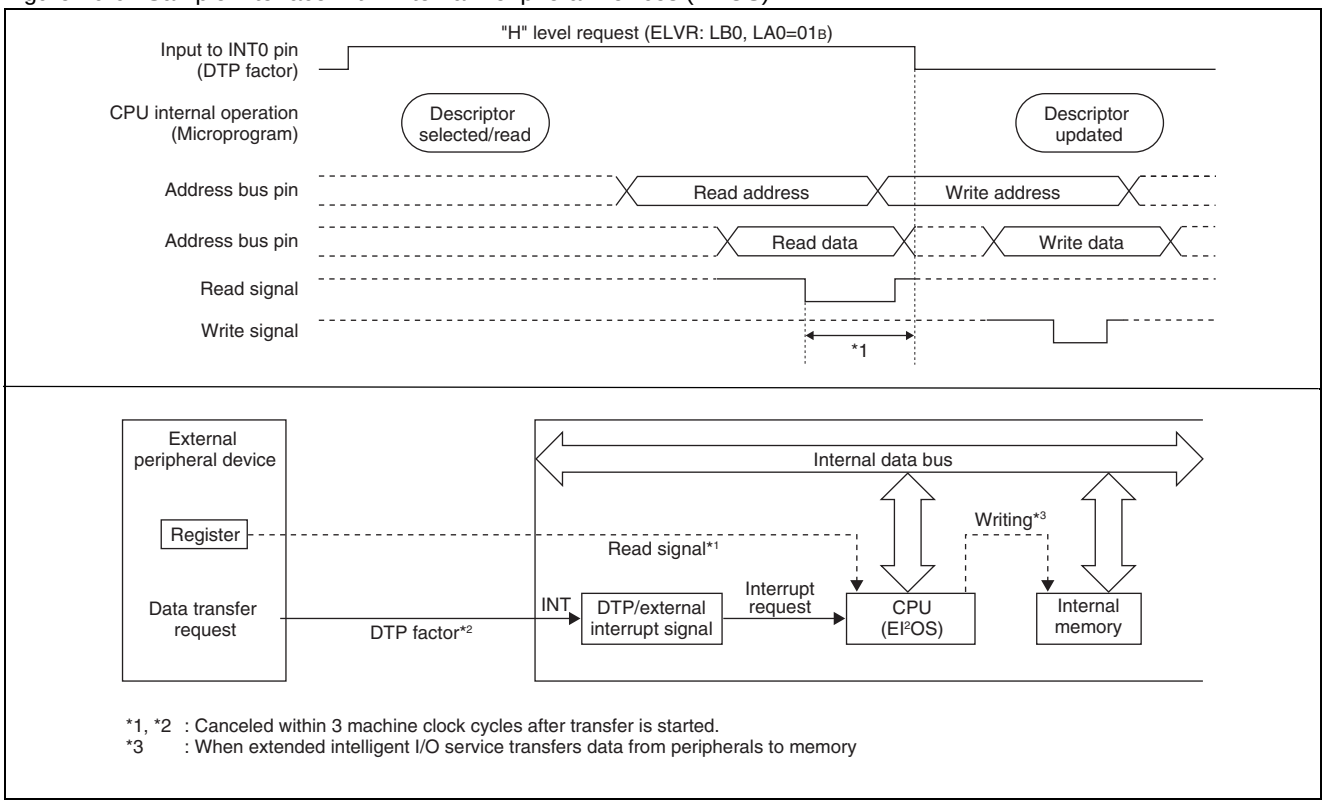
The DTP/external interrupt circuit has the DTP function that detects signals from external peripheral devices at the DTP/external interrupt pins and activates DMA transfer or extended intelligent I/O service.

Operations of DTP Function

The DTP function activates DMA transfer or extended intelligent I/O service (EI^{2}OS) upon detection of the signal level set by the detection level setting register of the DTP/external interrupt function.

- When DMA transfer has been enabled (DER: EN = 1), DMA is activated to start data transfer upon acceptance of an interrupt request.
- When EI^{2}OS has been enabled (ICR: ISE = 1), EI^{2}OS is activated to start data transfer upon acceptance of the interrupt request.
- Upon completion of 1 data transfer, the descriptor is updated and the DTP/external interrupt request flag bit is cleared to get ready for the next request from a DTP/external interrupt pin.
- A branch to interrupt handling takes place when transfer by DMA/EI^{2}OS is completed.

Figure 16-6. Sample Interface with External Peripheral Devices (EI^{2}OS)



16.5 Precautions on Use of DTP/External Interrupt Unit

This section shows precautions on use of the DTP/external interrupt unit.

Conditions for External Connection of Peripheral Devices

For support by the DTP unit, external peripheral devices must be able to automatically clear a request after successful data transfer. If a transfer request fails to be withdrawn within three machine cycles after the transfer operation starts, the DTP unit will proceed as if a new transfer request had been generated.

Operation Procedures for DTP/External Interrupt Unit

Set the values of registers in the DTP/external interrupt unit as follows:

1. Set the general-purpose I/O port that serves as the pin for an external interrupt input to the input port.
2. Set the bits for the enable register (ENIR) to “disable”.
3. Set the bits of the request level setting register (ELVR).
4. Clear the bits in the source register (EIRR).
5. Set the bits for the enable register (ENIR) to “enable”.

Steps 4 and 5 allow simultaneous writing by word-length specification.

To set the contents of DTP/external interrupt unit registers, first disable the enable register. Before enabling the enable register again, clear the source register in order to avoid accidental generation of an interrupt source in register setting and interrupt enabled state.

External Interrupt Request Level

- If edge request has been selected for the request level, the period of the minimum pulse width described in data sheet or more is required for detecting edge.
- When the request input level is a level setting, the required pulse width is the period of the minimum pulse width described in data sheet or more. Even if the interrupt source register is cleared, in addition, the interrupt request to the interrupt controller keeps being generated as long as the interrupt input pin maintains the active level.
- If level setting has been selected for the request input level, note that an external request that has been input remains active with respect to the interrupt controller even if it is later withdrawn, since the interrupt controller contains an internal source retention circuit. To withdraw a request with respect to the interrupt controller, the source retention circuit must be cleared.

Figure 16-7. Clearing the Source Retention Circuit when Setting the Level

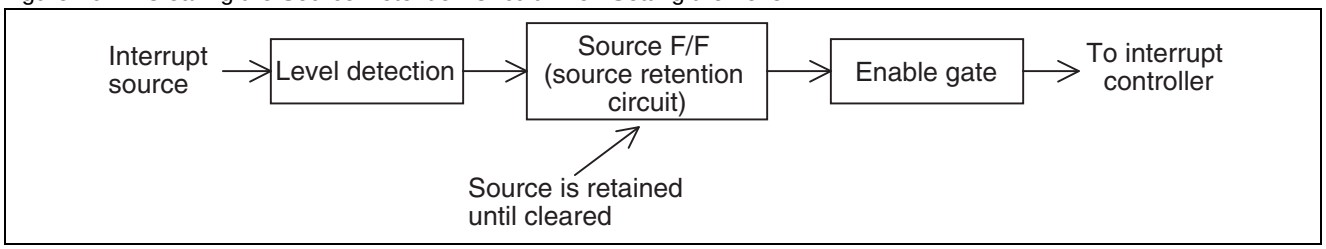
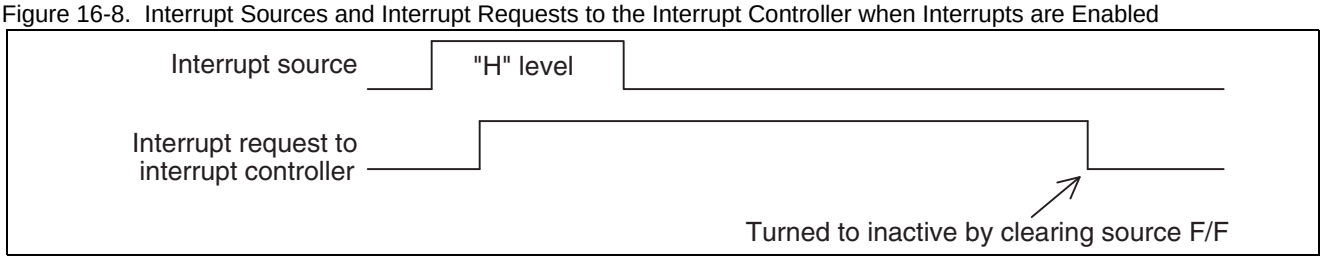


Figure 16-8. Interrupt Sources and Interrupt Requests to the Interrupt Controller when Interrupts are Enabled



17. 8/10-Bit A/D Converter

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

17.1 Overview of the 8/10-Bit A/D Converter

17.2 Configuration of the 8/10-Bit A/D Converter

17.3 Pins of 8/10-Bit A/D Converter

17.4 Registers of 8/10-Bit A/D Converter

17.5 Interrupts of 8/10-Bit A/D Converter

17.6 Operation of the 8/10-Bit A/D Converter

17.7 Precautions When Using the 8/10-Bit A/D Converter

17.1 Overview of the 8/10-Bit A/D Converter

The 8/10-bit A/D converter uses the RC successive approximation method to convert analog input voltages to a 10-bit or 8-bit digital value. The input signal can be selected from the 20 channels of analog input pin. Three different triggers are available to invoke conversion: the software trigger, base timer 0 output trigger, and external pin trigger input.

Functions of 8/10-Bit A/D Converter

Converts the analog voltage (input voltage) input to an analog input pin to a digital value (A/D conversion).

The 8/10-bit A/D converter has the following functions and features.

- Minimum conversion time is 3μs (for a 32MHz machine clock, including sampling time)
- Minimum sampling time is 1.2μs
- The RC successive approximation method with a sample and hold circuit is used for conversion.
- 10-bit or 8-bit resolution can be selected.
- The analog input pin can be selected from 20 channels by program.
- When A/D conversion completes, an interrupt request can be issued and EI^{2}OS invoked.
- The conversion data protection function is active if interrupts are enabled. This ensures no data is lost when performing continuous conversion.
- The conversion activation source can be selected. The available triggers are software, the output of base timer 0 (rising edge), and an external trigger input (falling edge).

Table 17-1 lists the three conversion modes.

Table 17-1. Conversion Modes for the 8/10-Bit A/D Converter

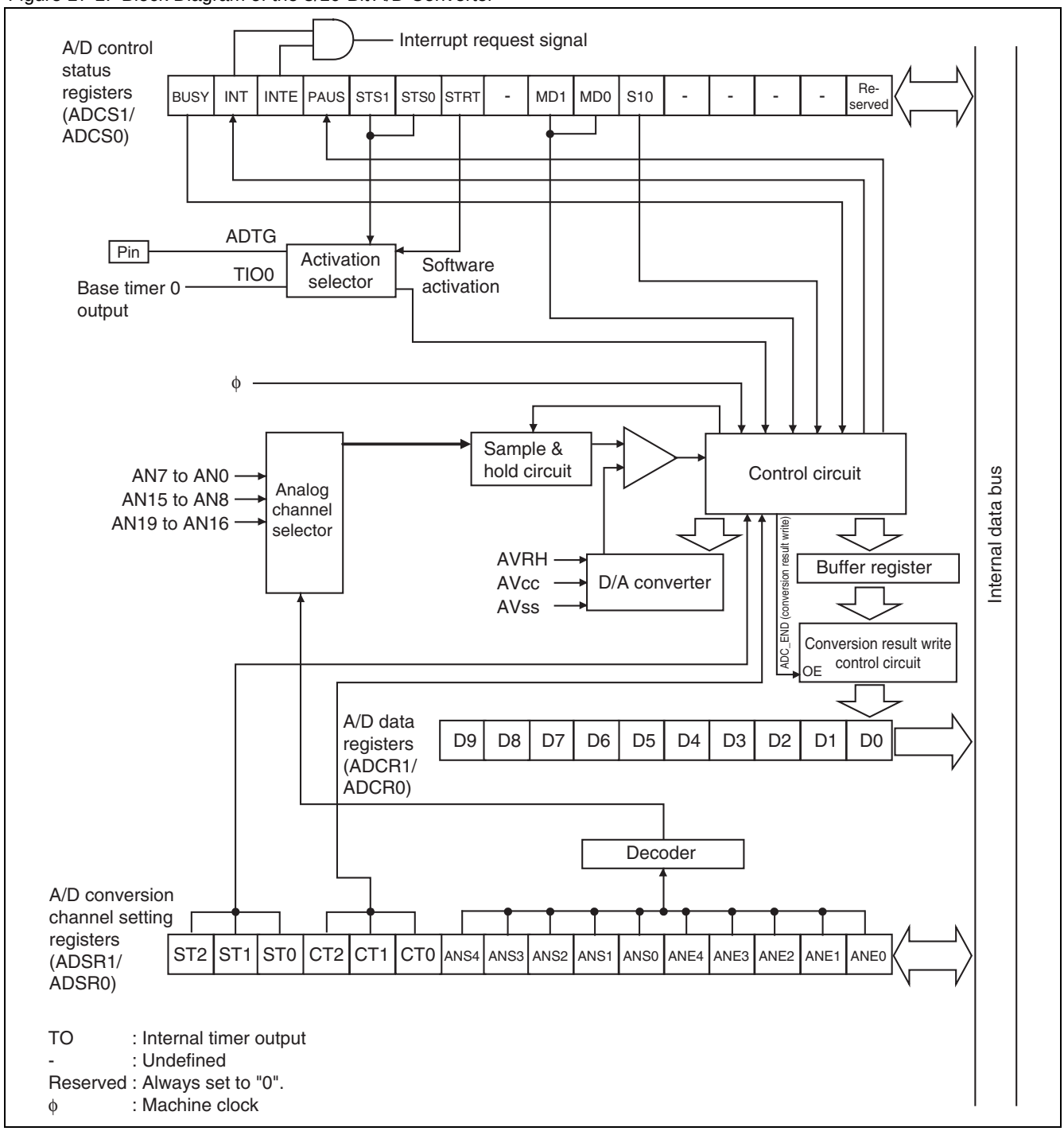
Conversion Mode	Descriptions
Single-shot conversion mode	Performs successive A/D conversions from the start channel to the end channel. The A/D conversion function halts after the A/D conversion for the end channel is performed.
Continuous conversion mode	Performs successive A/D conversions from the start channel to the end channel. After the A/D conversion for the end channel is performed, A/D conversion operation continues again from the start channel.
Stop conversion mode	Performs A/D conversion for each channel in turn, pausing after each conversion. After the A/D conversion for the end channel is performed, operation repeats again from the start channel.

17.2 Configuration of the 8/10-Bit A/D Converter

The 8/10-bit A/D converter consists of the following blocks.

Block Diagram of the 8/10-Bit A/D Converter

Figure 17-1. Block Diagram of the 8/10-Bit A/D Converter



A/D Control Status Register (ADCS)

This register is used for the software activation, activation selection, conversion mode selection, enabling or disabling the interrupt request, checking the interrupt request status, and to indicating when conversion is in progress or paused.

Buffer Register

Temporarily stores the A/D conversion result.

A/D Conversion Channel Setting Register (ADSR)

This register selects the A/D conversion channel by specifying the start and end channels from the 20 available channels. The register also sets the compare time and sampling time.

A/D Data Register (ADCR)

The A/D conversion result is temporarily saved in the buffer register and then transferred to this register when A/D conversion is completed and the value is finalized. The A/D conversion result can be read from this register.

Decoder

This circuit selects the analog input pin to use based on the ANE0 to ANE4 and ANS0 to ANS4 bit settings in the A/D conversion channel setting register (ADSR).

Analog Channel Selector

This circuit selects which of the 20 analog input pins to use.

Sample & Hold Circuit

This circuit holds the input voltage selected by the analog channel selector. Sampling and holding the input voltage at the time when A/D conversion is triggered prevents the result from being influenced by any variation in the input voltage during A/D conversion (during comparison).

D/A Converter

Generates the reference voltage used to compare with the input voltage captured by the sample & hold circuit.

Comparator

Compares the output voltage of the D/A converter with the input voltage captured by the sample & hold circuit to determine which is larger.

Control Circuit

Determines the A/D conversion result based on the comparison signal from the comparator. When the conversion result is finalized, the value is stored in the A/D data register via the conversion result write control circuit. If interrupt requests are enabled, an interrupt is generated.

Conversion Result Write Control Circuit

When A/D conversion completes and the conversion result is finalized, this circuit transfers the result data temporarily stored in the buffer register to the A/D data register.

17.3 Pins of 8/10-Bit A/D Converter

This section describes the pins of 8/10-bit A/D converter.

Pins of 8/10-Bit A/D Converter

The A/D converter pins are shared with general-purpose ports. Table 17-2 lists the pin functions, I/O type, and various settings when used by the 8/10-bit A/D converter.

Table 17-2. Pins of 8/10-Bit A/D Converter

Function	Pin Name	Pin Function	I/O Type	Pull-Up Setting	Standby Control	Required I/O Port Setting When Using Pin
ch.0	P60/AN0	Port 6 I/O, analog input	CMOS output/ CMOS hysteresis input or analog input	None	None	Set port 6 as an input (DDR6: bit0 to bit7 = 0), and set as analog inputs (ADER0: bit8 to bit15 = 1)
ch.1	P61/AN1					
ch.2	P62/AN2					
ch.3	P63/AN3					
ch.4	P64/AN4					
ch.5	P65/AN5					
ch.6	P66/AN6					
ch.7	P67/AN7					
ch.8	P90/CS0/AN8	Port 9 I/O, analog input				Set port 9 as an input (DDR9: bit8 to bit15 = 0), and set as analog inputs (ADER1: bit0 to bit7 = 1)
ch.9	P91/CS1/AN9					
ch.10	P92/CS2/AN10					
ch.11	P93/CS3/AN11					
ch.12	P94/AN12					
ch.13	P95/(UI3*)/AN13					
ch.14	P96/(UO3*)/AN14					
ch.15	P97/(UCK3*)/AN15					
ch.16	P70/PWM1P2/AN16	Port 7 I/O, analog input				Set port 7 as an input (DDR7: bit8 to bit11 = 0), and set as analog inputs (ADER2: bit8 to bit11 = 1)
ch.17	P71/(UI4*)/AN17					
ch.18	P72/(UO4*)/AN18					
ch.19	P73/(UCK4*)/AN19					

17.4 Registers of 8/10-Bit A/D Converter

This section lists the registers of the 8/10-bit A/D converter.

Register List of 8/10-Bit A/D Converter

Figure 17-2. Register List of 8/10-Bit A/D Converter

address	bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$001B_H$	ADER0															
$001D_H/001C_H$	ADER2								ADER1							
$0035_H/0034_H$	ADCSH								ADCSL							
$0037_H/0036_H$	ADCRH								ADCRL							
$0039_H/0038_H$	ADSRH								ADSRL							

Refer to Section “8.2.3 Other Registers” for explanation of ADER0, ADER1, and ADER2.

17.4.1 A/D Control Status Register (Upper) (ADCSH)

The functions of the A/D control status register (upper) (ADCSH) are the software activation, activation selection, interrupt request enable and disable, and checking whether conversion is in progress or paused.

A/D Control Status Register (Upper) (ADCSH)

Figure 17-3. A/D Control Status Register (Upper) (ADCSH)

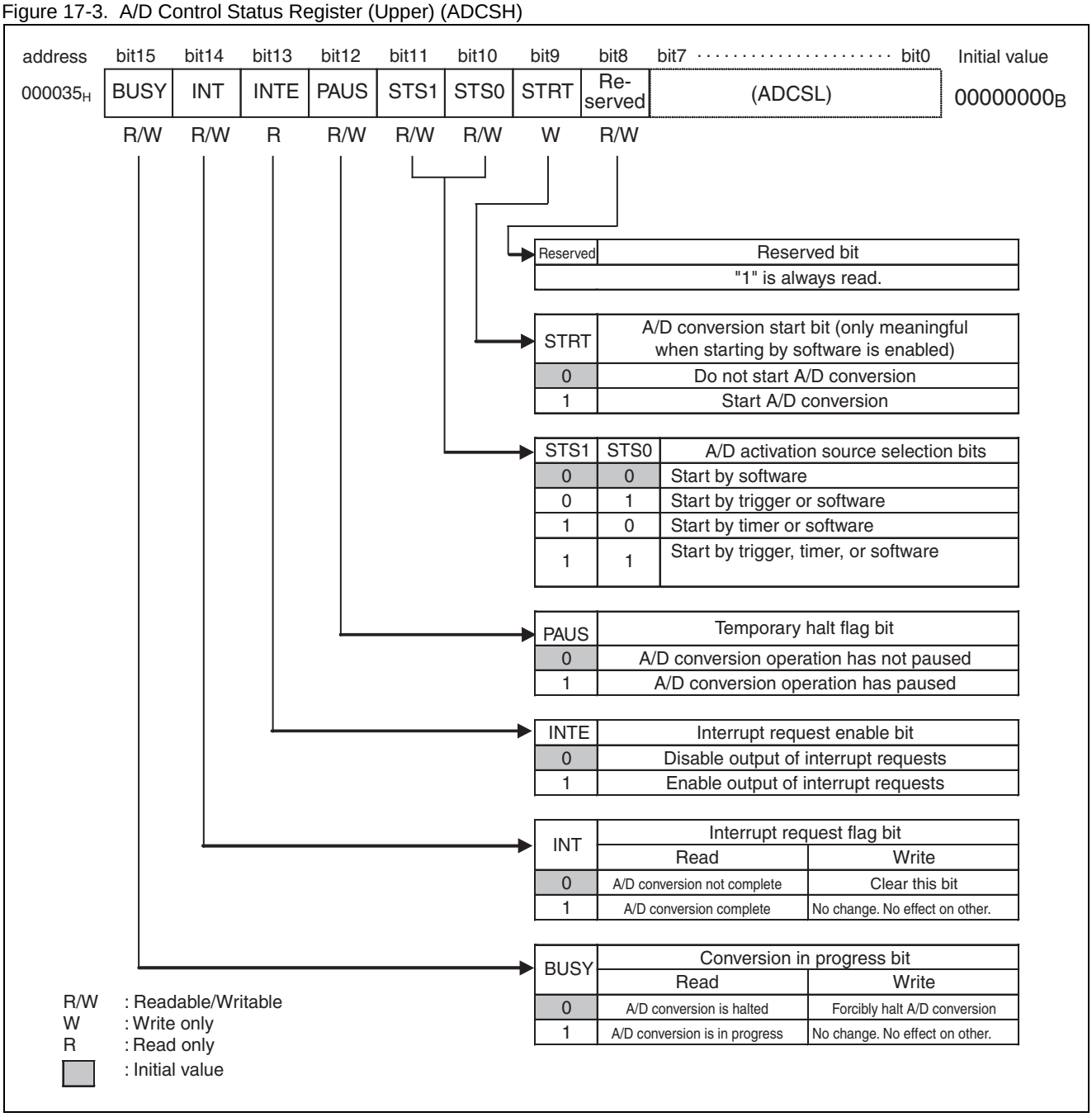


Table 17-3. Function of Each Bit of the A/D Control Status Register (Upper) (ADCSH) (Sheet 1 of 2)

Bit name		Function
bit15	BUSY: Conversion in progress bit	■ This bit indicates when the A/D converter is operating. ■ On reading the BUSY bit, “0” indicates that A/D conversion is halted and “1” indicates that conversion is in progress. ■ Writing “0” to the BUSY bit forcibly halts the A/D conversion. Writing "1" does not change the bit value and has no effect on the other. ■ The bit is read as “1” by read-modify-write (RMW) instructions. **Note:** Do not attempt to forcibly terminate (BUSY=0) and activate (by software (STRT=1), external trigger, or timer) the A/D converter at the same time.
bit14	INT: Interrupt request flag bit	■ The INT bit is set to “1” when the A/D conversion sets data to the A/D data register. ■ If the INT bit and the interrupt request enable bit (ADCSH:INTE) are both “1”, an interrupt request is generated. If EI^{2}OS is enabled, EI^{2}OS starts. ■ The INT bit is cleared by writing “0” to it. ■ Writing “1” does not change the bit value and has no effect on the other. ■ The INT bit is cleared automatically when transfer of A/D conversion data by EI^{2}OS completes ■ The bit is read as “1” by read-modify-write (RMW) instructions. ■ If the bit is set to “1” at the same time as writing “0”, writing “0” has priority.
bit13	INTE: Interrupt request enable bit	■ This bit enables and disables output of interrupt to the CPU. ■ If the INTE bit and the interrupt request flag bit (ADCSH:INT) are both “1”, an interrupt request is generated. ■ Set this bit to “1” when using EI^{2}OS.

Table 17-3. Function of Each Bit of the A/D Control Status Register (Upper) (ADCSH) (Sheet 2 of 2)

Bit name		Function
bit12	PAUS: Pause flag bit	■ The PAUS bit indicates that the A/D conversion protection function is operating. The value is only meaningful when interrupt requests are enabled (ADCS:INTE=1) ■ The bit is set to “1” when the A/D conversion protection function is operating. ■ This bit is set by writing “1” to it and cleared by writing “0”. ■ When interrupt requests are enabled (ADCS:INTE=1) and A/D conversion started, the interrupt request flag bit (ADCS:INT) is set after an A/D conversion completes and an interrupt request is output. If the next A/D conversion completes before the interrupt request flag bit (ADCS:INT) is cleared, A/D conversion is paused to prevent the previous data from being overwritten (A/D conversion protection function). When A/D conversion is paused, the PAUS bit is set to “1”. ■ When the interrupt request flag bit (ADCS:INT) is cleared, the 8/10-bit A/D converter releases the pause and A/D conversion operation restarts. ■ Writing “0” to the interrupt request flag bit (ADCS:INT) clears the flag. If EI^{2}OS/μDMAC has been setup to transfer the A/D conversion result from the A/D data register (ADCR), the interrupt request flag bit (ADCS:INT) is cleared by EI^{2}OS/μDMAC when transfer of the A/D conversion result completes. **Notes:** ■ See “17.6.2 A/D Conversion Data Protection Function” for details of the A/D conversion data protection function. ■ This bit is not cleared automatically when the pause is released. Write “0” to this bit to clear it.
bit11, bit10	STS1, STS0: A/D activation source selection bits	■ Selects the activation source for A/D conversion. ■ If more than one activation source is selected, the first activation source to occur starts conversion. **Notes:** ■ As the activation source setting changes immediately after writing to the bits, ensure that the newly selected activation source is not active if you write to these bits while A/D conversion is in operation. ■ The timer trigger is the output of base timer 0.
bit9	STRT: A/D conversion start bit	■ This bit starts A/D conversion by software. ■ Writing “1” to the STRT bit starts A/D conversion. ■ Restarting using the STRT bit has no effect in other than single-shot conversion mode 1 (MD1/MD0=00_B) ■ This bit always reads “0”. **Note:** ■ If a forcible halt (BUSY=0) and software activation (STRT=1) are set simultaneously, the forcible halt has priority.
bit8	Reserved bit	■ “1” is always read. Writing has no effect on the operation. ■ Always write “0” to this bit.

17.4.2 A/D Control Status Register (Lower) (ADCSL)

The function of the A/D control status register (lower) (ADCSL) is used to select the conversion mode.

A/D Control Status Register (Lower) (ADCSL)

Figure 17-4. A/D Control Status Register (Lower) (ADCSL)

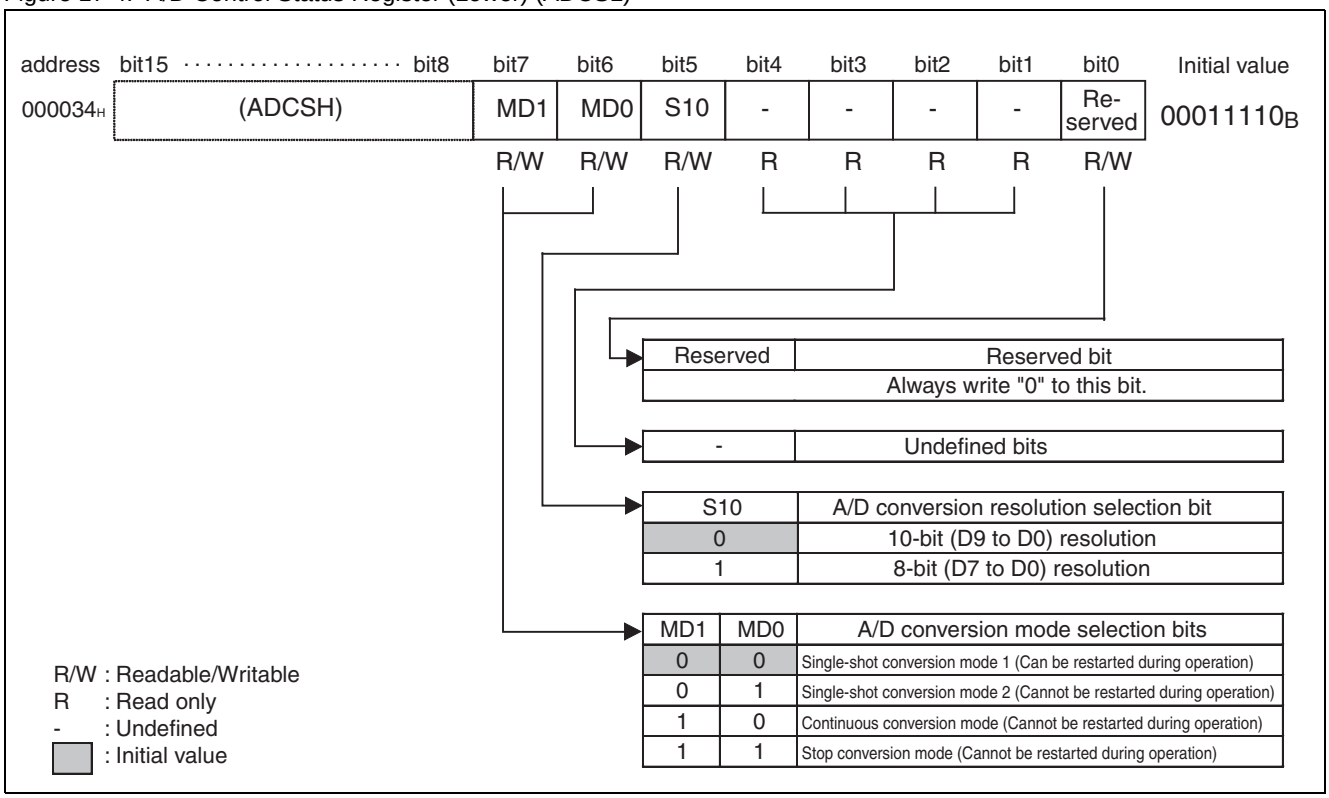


Table 17-4. Function of Each Bit of the A/D Control Status Register (Lower) (ADCSL) (Sheet 1 of 2)

Bit name		Function
bit7, bit6	MD1,MD0: A/D conversion mode selection bits	■ These bits select the conversion mode for the A/D conversion function. ■ The 2-bit value in MD1 and MD0 select one of single-shot conversion mode 1, single-shot conversion mode 2, continuous conversion mode, or stop conversion mode. ■ The meaning of each mode is as follows. **Single-shot conversion mode 1:** ■ The A/D conversion is consecutively performed through analog inputs from the start channel (ADCR:ANS4 to ANS0) to the end channel (ADCR:ANE4 to ANE0). ■ A/D conversion halts after conversion completes for the end channel. ■ Conversion can be restarted while still in progress. Even though the activation trigger is input when A/D conversion is paused with the A/D conversion data protection function working, however, it is not restarted but the channel selector value is initialized. **Single-shot conversion mode 2:** ■ The A/D conversion is consecutively performed through analog inputs from the start channel (ADCR:ANS4 to ANS0) to the end channel (ADCR:ANE4 to ANE0). ■ A/D conversion halts after conversion completes for the end channel. ■ A/D conversion cannot be restarted while in progress. **Continuous conversion mode:** ■ The A/D conversion is consecutively performed through analog inputs from the start channel (ADCR:ANS4 to ANS0) to the end channel (ADCR:ANE4 to ANE0). ■ After conversion completes for the end channel, operation returns to the analog input of the start channel and repeats. ■ To terminate A/D conversion, write “0” to the A/D conversion in progress bit in the A/D control status register (ADCS:BUSY). ■ A/D conversion cannot be restarted while in progress. **Stop conversion mode:** ■ A/D conversion starts from the start channel (ADCR:ANS4 to ANS0). A/D conversion pauses after conversion completes for that channel. When the activation trigger input occurs while A/D conversion is paused, A/D conversion is performed for the next channel. ■ A/D conversion operation pauses as usual after conversion completes for the end channel. When the activation trigger input occurs while A/D conversion is paused, operation returns to the analog input of the start channel and repeats. ■ To terminate A/D conversion, write “0” to the A/D conversion in progress bit in the A/D control status register (ADCS:BUSY). ■ A/D conversion cannot be restarted while in progress. **Note:** When it is stated that operation cannot be restarted for the single-shot, continuous, and stop conversion modes, this applies to the external trigger and software trigger. The conversion mode should be changed in the stopped state existing prior to A/D conversion.

Table 17-4. Function of Each Bit of the A/D Control Status Register (Lower) (ADCSL) (Sheet 2 of 2)

Bit name		Function
bit5	S10: A/D conversion resolution selection bit	■ This bit selects the resolution of A/D conversion. ■ Writing “0” to the S10 bit selects 10-bit resolution and writing "1" selects 8-bit resolution. **Note:** The data bits used are different depending on the resolution.
bit4 to bit1	Undefined bits	■ The read value is fixed at “1”. ■ Writing has no effect on the operation.
bit0	Reserved bit	Always set this bit to “1”.

Notes:

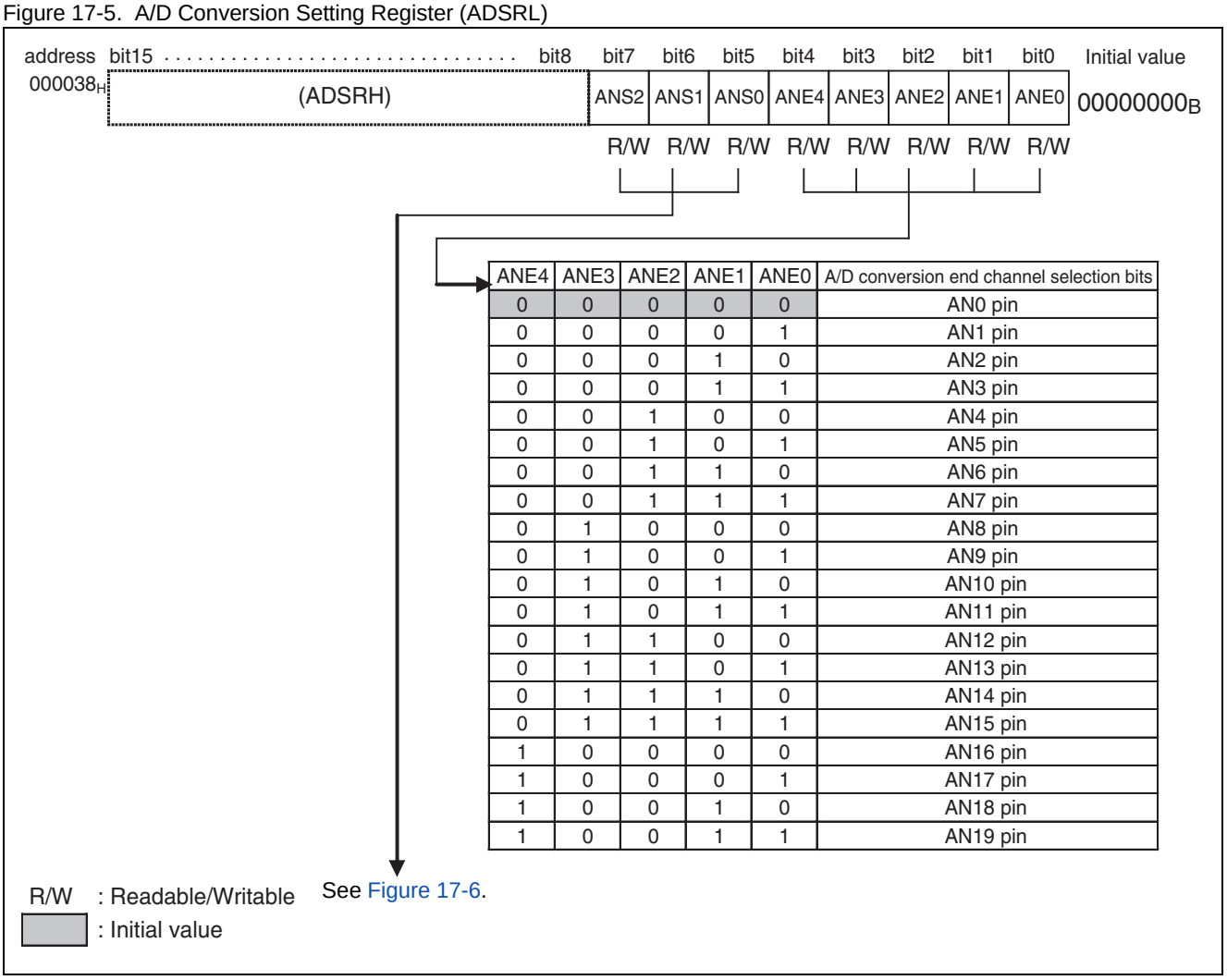
- Only change the value of the S10 bit while A/D operation is halted prior to starting conversion. The contents of ADCR will become undefined if changed after conversion.
- If A/D conversion is restarted at the same time as conversion completes in single-shot conversion mode 1, the A/D conversion in progress will be cancelled and conversion will restart.

17.4.3 A/D Conversion Channel Setting Register (ADSR)

The function of the A/D conversion channel setting register (ADSR) is used to select the A/D conversion channels and conversion time setting.

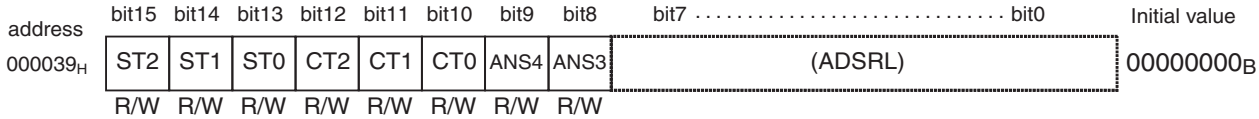
A/D Conversion Setting Register (ADSR)

Figure 17-5. A/D Conversion Setting Register (ADSRL)



ANE4	ANE3	ANE2	ANE1	ANE0	A/D conversion end channel selection bits
0	0	0	0	0	AN0 pin
0	0	0	0	1	AN1 pin
0	0	0	1	0	AN2 pin
0	0	0	1	1	AN3 pin
0	0	1	0	0	AN4 pin
0	0	1	0	1	AN5 pin
0	0	1	1	0	AN6 pin
0	0	1	1	1	AN7 pin
0	1	0	0	0	AN8 pin
0	1	0	0	1	AN9 pin
0	1	0	1	0	AN10 pin
0	1	0	1	1	AN11 pin
0	1	1	0	0	AN12 pin
0	1	1	0	1	AN13 pin
0	1	1	1	0	AN14 pin
0	1	1	1	1	AN15 pin
1	0	0	0	0	AN16 pin
1	0	0	0	1	AN17 pin
1	0	0	1	0	AN18 pin
1	0	0	1	1	AN19 pin

Figure 17-6. A/D Conversion Setting Register (ADSRH)



ANS4	ANS3	ANS2	ANS1	ANS0	A/D conversion start channel selection bits: Channel number	Read during conversion	Read during pause in stop conversion mode
0	0	0	0	0	AN0 pin	Number of channel currently being converted	Number of channel converted previously
0	0	0	0	1	AN1 pin		
0	0	0	1	0	AN2 pin		
0	0	0	1	1	AN3 pin		
0	0	1	0	0	AN4 pin		
0	0	1	0	1	AN5 pin		
0	0	1	1	0	AN6 pin		
0	0	1	1	1	AN7 pin		
0	1	0	0	0	AN8 pin		
0	1	0	0	1	AN9 pin		
0	1	0	1	0	AN10 pin		
0	1	0	1	1	AN11 pin		
0	1	1	0	0	AN12 pin		
0	1	1	0	1	AN13 pin		
0	1	1	1	0	AN14 pin		
0	1	1	1	1	AN15 pin		
1	0	0	0	0	AN16 pin		
1	0	0	0	1	AN17 pin		
1	0	0	1	0	AN18 pin		
1	0	0	1	1	AN19 pin		

CT2	CT1	CT0	Compare time selection bits	Internal operating frequency: 33MHz	25MHz	16MHz
0	0	0	22 machine cycles	Prohibited setting		
0	0	1	33 machine cycles	Prohibited setting		2.06 µs
0	1	0	44 machine cycles	Prohibited setting	Prohibited setting	2.75 µs
0	1	1	66 machine cycles	2.00 µs	2.64 µs	4.12 µs
1	0	0	88 machine cycles	2.67 µs	3.52 µs	5.49 µs
1	0	1	132 machine cycles	4.00 µs	5.28 µs	8.24 µs
1	1	0	176 machine cycles	5.33 µs	7.04 µs	10.98 µs
1	1	1	264 machine cycles	8.00 µs	10.56 µs	16.47 µs

ST2	ST1	ST0	Sampling time selection bits	Internal operating frequency: 33MHz	25MHz	16MHz
0	0	0	4 machine cycles	Prohibited setting		
0	0	1	6 machine cycles	Prohibited setting		
0	1	0	8 machine cycles	Prohibited setting		
0	1	1	12 machine cycles	Prohibited setting		
1	0	0	24 machine cycles	Prohibited setting		1.50 µs
1	0	1	36 machine cycles	Prohibited setting	1.44 µs	2.25 µs
1	1	0	48 machine cycles	1.45 µs	1.92 µs	3.00 µs
1	1	1	128 machine cycles	3.88 µs	5.12 µs	8.00 µs

R/W : Readable/Writable
: Initial value

Table 17-5. Function of Each Bit of the A/D Conversion Setting Register (ADSR)

Bit name		Function
bit15 to bit13	ST2 to ST0: Sampling time selection bits	■ These bits select the sampling time for A/D conversion. ■ After A/D starts, the analog input is obtained for the time specified by the ST2, ST1, and ST0 bits. **Notes:** ■ The correct analog conversion value may not be obtained if a prohibited setting is selected. ■ Do not set the sampling time during A/D conversion.
bit12 to bit10	CT2 to CT0: Compare time selection bits	■ These bits select the compare time for A/D conversion. ■ After the analog input is obtained (after the sampling time elapses), the conversion result data is finalized after the time specified in the CT2, CT1, and CT0 bits has elapsed and the value is stored in bits 9 to 0 of the ADCR register. **Notes:** ■ The correct analog conversion value may not be obtained if a prohibited setting is selected. ■ Do not set the compare time during A/D conversion.
bit9 to bit5	ANS4 to ANS0: A/D conversion start channel selection bits	■ These bits are used to set the start channel for A/D conversion and to verify the number of the channel currently being converted. ■ When A/D conversion starts, the channel specified by these bits is converted first. ■ During A/D conversion, the number of the channel currently being converted can be read. When conversion has paused in stop conversion mode, the number of the previously converted channel can be read. **Notes:** ■ Word access should be done when writing to this bit. If byte-write is used, A/D conversion starts unintentionally. ■ Do not use a read-modify-write (RMW) instruction to set the A/D conversion mode setting bits (MD1, MD0) or A/D conversion end bits (ANE4 to ANE0) after setting the A/D conversion start channel selection bits (ANS4 to ANS0). As reading the A/D conversion start channel selection bits (ANS4 to ANS0) prior to A/D conversion starting returns the last channel to be converted, the value of the bits may be updated unintentionally. ■ Do not set the A/D conversion start channel bits (ANS4 to ANS0) during A/D conversion. ■ As analog input channels AN31 to AN20 do not exist, their A/D conversion results are invalid. ■ Word access should be done when writing to this bit. If byte-write is used, A/D conversion starts at an unintentionally channel.
bit4 to bit0	ANE4 to ANE0: A/D conversion end channel selection bits	■ These bits specify the end channel for A/D conversion. ■ After A/D conversion starts, conversion continues up to the channel specified by these bits. ■ If the same channel number as ANS4 to ANS0 is set, conversion is performed for that channel only. Also, when continuous conversion or stop conversion mode is set, operation returns to the start channel specified in ANS4 to ANS0 after conversion completes for the channel specified by these bits. **Notes:** ■ Do not set the A/D conversion end channel bits during A/D conversion. ■ As analog input channels AN31 to AN20 do not exist, their A/D conversion results are invalid.

17.4.4 A/D Data Register (ADCRH/ADCRL)

The A/D data register (ADCRH/ADCRL) stores the A/D conversion result.

A/D Data Register (ADCRH/ADCRL)

Figure 17-7. A/D Data Register (ADCRH/ADCRL)

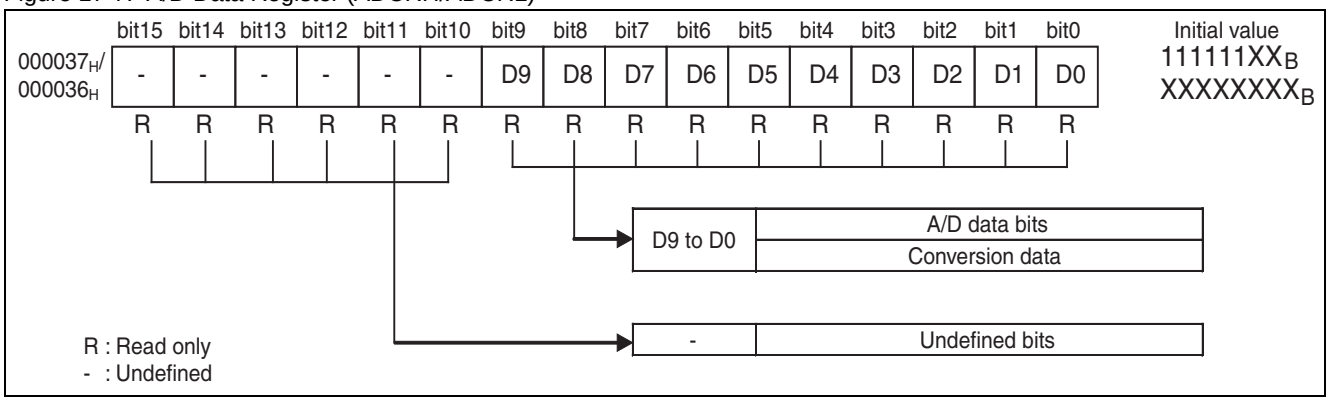


Table 17-6. Function of the A/D Data Register (ADCRH/ADCRL)

Bit name		Function
bit15 to bit10	Undefined bits	“1” is always read.
bit9 to bit0	D9 to D0: A/D data bits	These bits hold the result of A/D conversion. When the resolution is set to 10 bits (ADCS: SA10=0) Conversion data is stored in 10 bits from D9 to D0. When the resolution is set to 8 bits (ADCS: S10=1) Conversion data is stored in 8 bits from D7 to D0. The value read from D9 and D8 at this time is "1". **Notes:** ■ Writing to this register is prohibited. ■ To read the result of conversion from the A/D conversion data bits (D9 to D0), use a word instruction (MOVW).

17.5 Interrupts of 8/10-Bit A/D Converter

The 8/10-bit A/D converter can generate an interrupt request when it sets data in the A/D data register after an A/D conversion. The unit also supports the extended intelligent I/O service (EI^{2}OS).

Table 17-7. Function of the A/D Data Register (ADCRH/ADCRL) at Interrupt Request

Bit name		Function
bit15 to bit10	Undefined bits	■ The read value is fixed to “1”. ■ Writing has no effect on the operation.
bit9 to bit0	D9 to D0	■ Stores the A/D conversion result. The register is update after each conversion. ■ The register always holds the result of the most recent conversion. ■ The initial value is undefined. **Notes:** ■ Writing to this register is prohibited. ■ A conversion data protection function is provided. (See “17.6.2 A/D Conversion Data Protection Function” for details.)

Interrupts of 8/10-Bit A/D Converter

Table 17-8 describes the interrupt control bit and interrupt source for the 8/10-bit A/D converter.

Table 17-8. Interrupt Control Bit and Interrupt Source for the 8/10-Bit A/D Converter

Interrupt Source	A/D Control Status Register (Upper) (ADCSH)		
	Interrupt Flag Bit	Interrupt Enable Bit	Clear Interrupt Flag
A/D conversion result written to the A/D data register	INT	INTE	■ Write “0” to the INT bit ■ Reset ■ EI^{2}OS starts

The 8/10-bit A/D converter sets the interrupt flag bit to “1” when the interrupt source described in Table 17-8 occurs. If the interrupt enable bit is also “1” at this time, an interrupt request is output to the interrupt controller.

Interrupts of 8/10-Bit A/D Converter, EI^{2}OS, and μDMAC

Table 17-9. Interrupts of 8/10-Bit A/D Converter, EI^{2}OS, and μDMAC

Channel	Interrupt No.	Interrupt Control Register		Vector Table Address			EI^{2}OS	μDMAC Channel No.
		Register Name	Address	Lower	Upper	Bank		
A/D converter	#27($1B_H$)	ICR08	$0000B8_H$	$FFFF8C_H$	$FFFF8D_H$	$FFFF8E_H$	❍	–

❍: Available

EI^{2}OS Function for the 8/10-Bit A/D Converter

The 8/10-bit A/D converter can use the EI^{2}OS function to transfer A/D conversion results to memory. In this case, the conversion data protection function operates and A/D conversion pauses until after the A/D conversion data is transferred to memory and the INT bit is cleared. This prevents any loss of data. (See “17.6.2 A/D Conversion Data Protection Function”).

17.6 Operation of the 8/10-Bit A/D Converter

The 8/10-bit A/D converter has three different types of conversion mode: single-shot conversion mode, continuous conversion mode, and stop conversion mode. This section describes the operation in each mode.

Operation in Single-Shot Conversion Mode

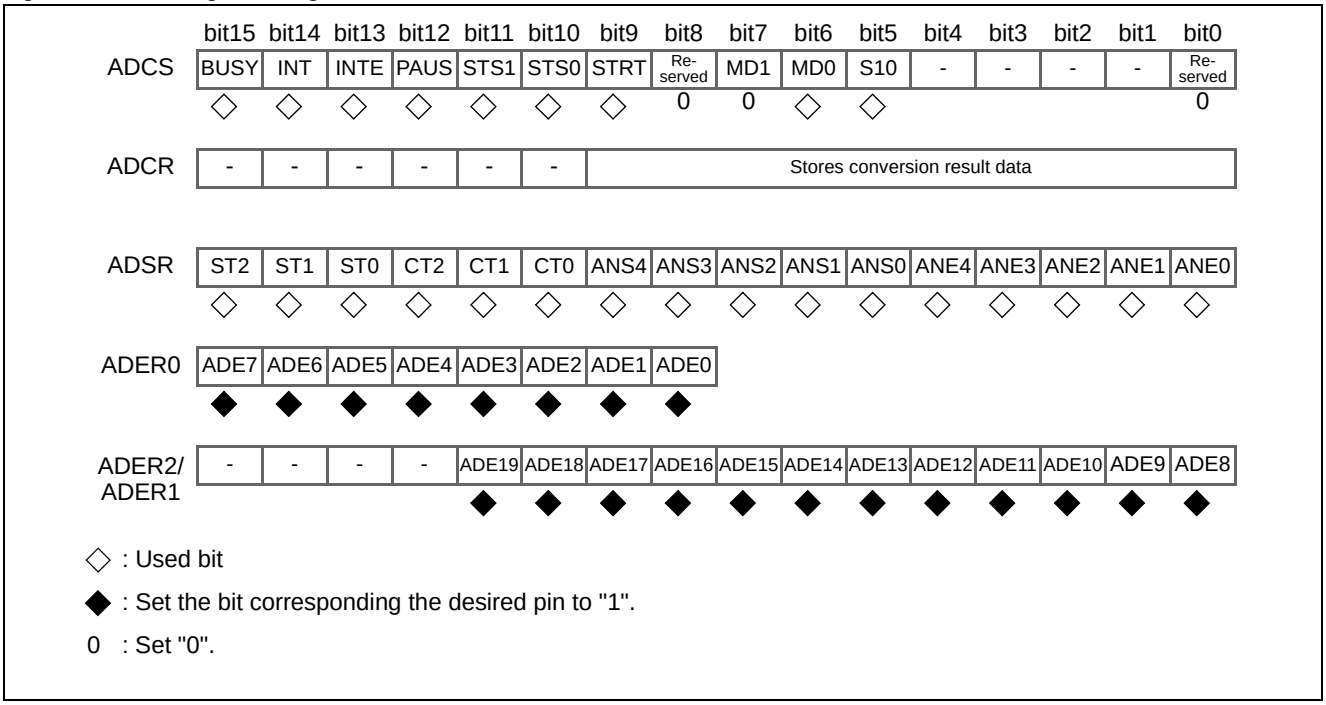
In single-shot conversion mode, conversion is performed successively for the analog inputs specified by the ANS and ANE bits, and then operation halts after conversion completes for the end channel specified in the ANE bits. If the same channel is specified as both the start and end channels (ANS = ANE), conversion is performed for one channel only specified by the ANS bits. The settings shown in Figure 17-8 are required to use single-shot conversion mode.

Restarting operation during A/D conversion is permitted if the A/D conversion mode selection bits (MD1, MD0) are set to "00_B". Even though the activation trigger is input when A/D conversion is paused with the A/D conversion data protection function working, however, it is not restarted but the channel selector value is initialized.

Operation cannot be restarted during A/D conversion if the A/D conversion mode selection bits (MD1, MD0) are set to "01_B".

To abort A/D conversion, write "0" to the A/D conversion busy flag bit in the A/D control status register (ADCS:BUSY).

Figure 17-8. Settings for Single-Shot Conversion Mode



The following is an example of the conversion sequence in single-shot conversion mode.

ANS=00000_B, ANE=00011_B: AN0 → AN1 → AN2 → AN3 → End

ANS=00011_B, ANE=00011_B: AN3 → End

[Restart]

To restart A/D conversion in progress or being paused, abort it first. Take the following steps:

1) Clear the A/D conversion busy flag (ADCS:BUSY).

2) Clear the interrupt request flag bit (ADCS:INT).

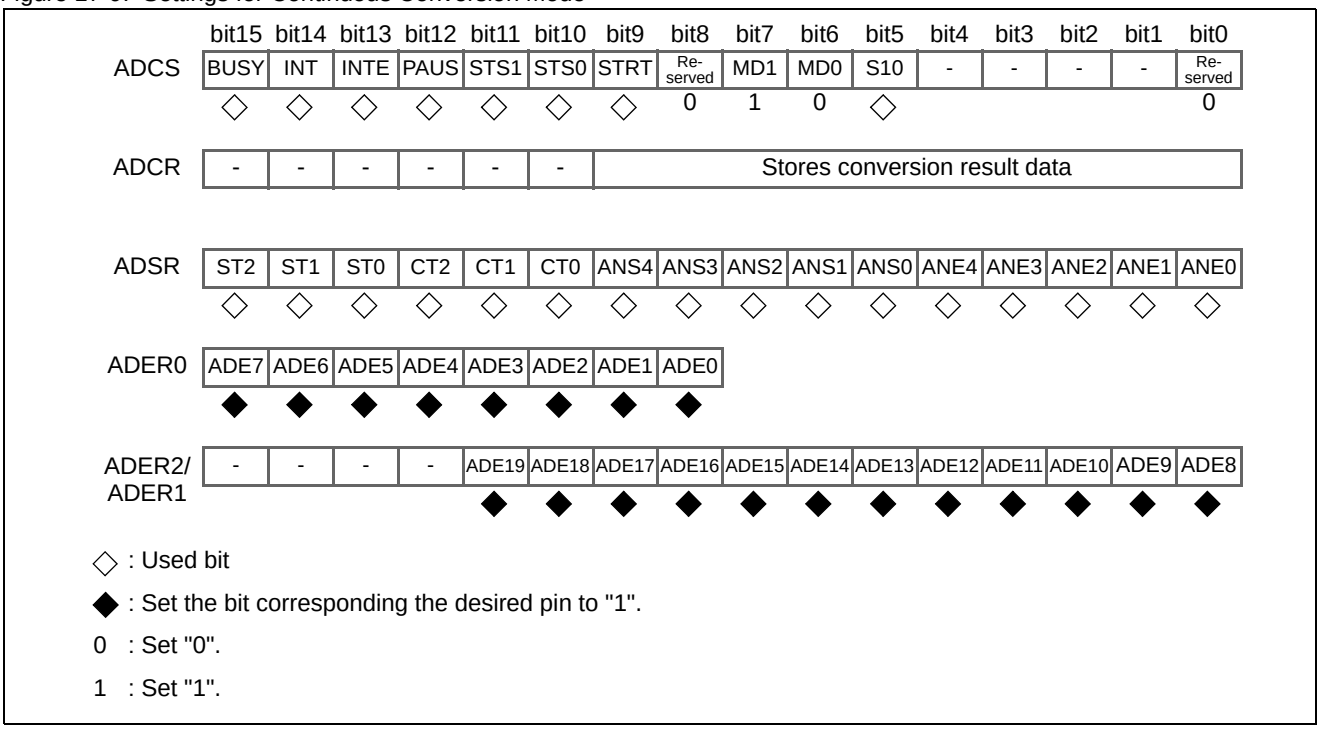
3) Set the A/D conversion software start bit (ADCS:STRT).

Operation in Continuous Conversion Mode

In continuous conversion mode, conversion is performed successively for the analog inputs specified by the ANS and ANE bits. After conversion completes for the end channel specified in the ANE bits, operation returns to the analog input specified by the ANS bits and A/D conversion continues. If the same channel is specified as both the start and end channels (ANS = ANE), conversion is performed repeatedly for that channel only specified by the ANS bits. The settings shown in Figure 17-9 are required to use continuous conversion mode.

To abort A/D conversion, write “0” to the A/D conversion busy flag bit in the A/D control status register (ADCS:BUSY).

Figure 17-9. Settings for Continuous Conversion Mode



The following is an example of the conversion sequence in continuous conversion mode.

ANS=00000_B, ANE=00011_B: AN0 → AN1 → AN2 → AN3 → AN0 → Repeat

ANS=00011_B, ANE=00011_B: AN3 → AN3 → Repeat

[Restart]

To restart A/D conversion in progress or being paused, abort it first. Take the following steps:

1) Clear the A/D conversion busy flag (ADCS:BUSY).

2) Clear the interrupt request flag bit (ADCS:INT).

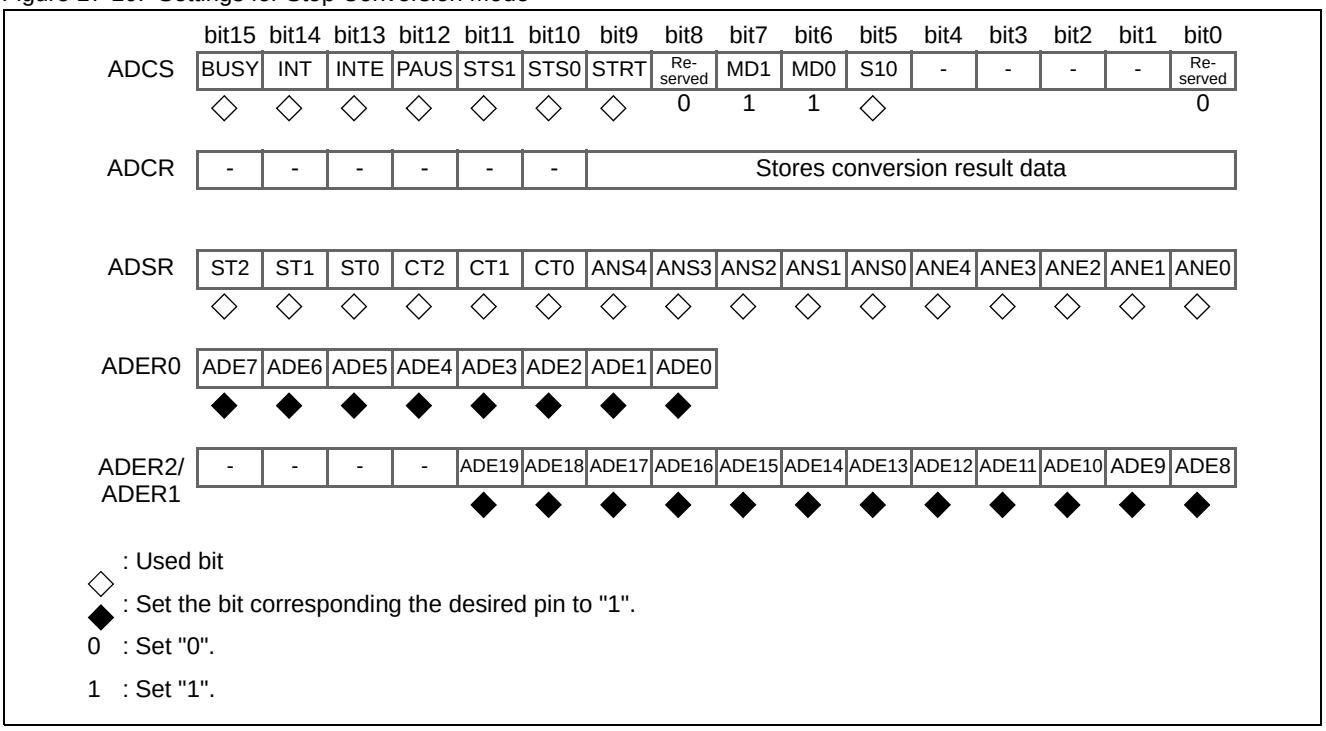
3) Set the A/D conversion software start bit (ADCS:STRT).

Operation in Stop Conversion Mode

In stop conversion mode, conversion is performed one channel at a time for the analog inputs specified by the ANS and ANE bits, pausing after each conversion. After conversion completes for the end channel specified in the ANE bits, the same operation (convert and pause) is repeated, starting again from the analog input specified by the ANS bits. If the same channel is specified as both the start and end channels (ANS = ANE), conversion is performed repeatedly for that channel only specified by the ANS bits. The activation source specified by the STS1 and STS0 bits is used to restart conversion when paused. The settings shown in Figure 17-10 are required to use stop conversion mode.

To abort A/D conversion, write “0” to the A/D conversion busy flag bit in the A/D control status register (ADCS:BUSY).

Figure 17-10. Settings for Stop Conversion Mode



The following is an example of the conversion sequence in stop conversion mode.

ANS=00000_B, ANE=00011_B: AN0 → Pause → AN1 → Pause → AN2 → Pause → AN3 → Pause → AN0 → Repeat

ANS=00011_B, ANE=00011_B: AN3 → Pause → AN3 → Pause → Repeat

[Restart]

To restart A/D conversion in progress or being paused, abort it first. Take the following steps:

1) Clear the A/D conversion busy flag (ADCS:BUSY).

2) Clear the interrupt request flag bit (ADCS:INT).

3) Set the A/D conversion software start bit (ADCS:STRT).

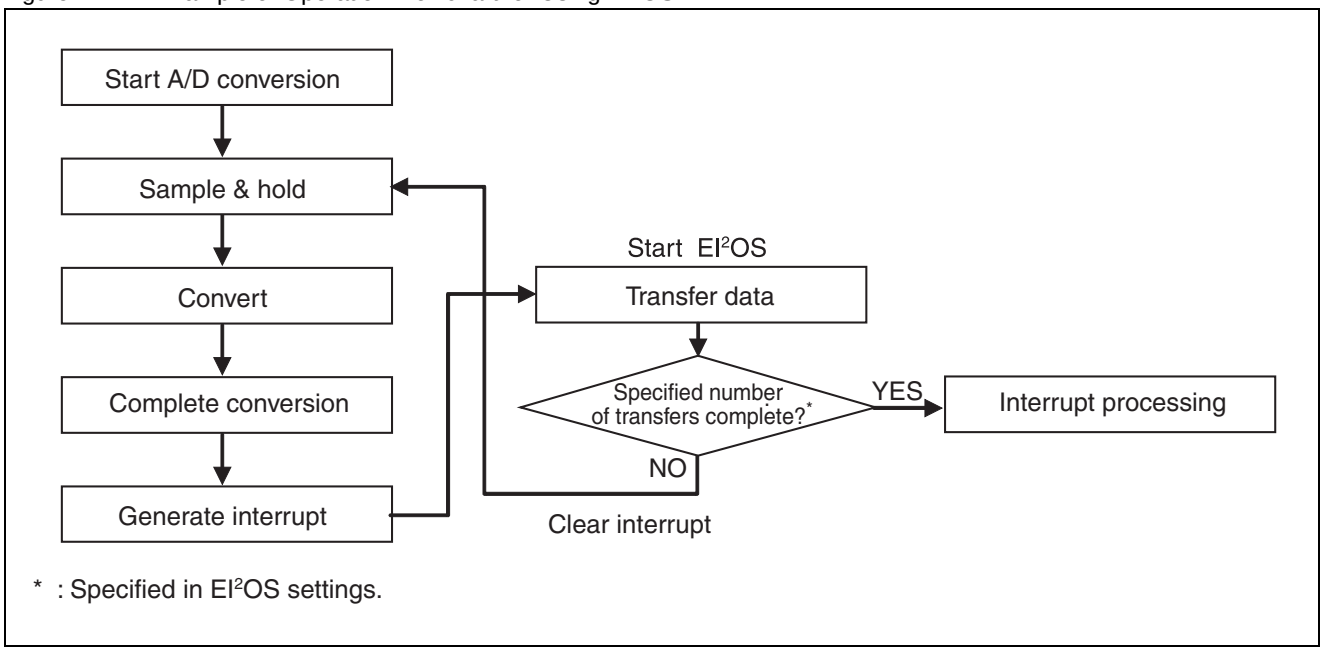
17.6.1 Conversion Operation Using EI^{2}OS

The 8/10-bit A/D converter can use EI^{2}OS to transfer the A/D conversion results to memory.

Conversion Operation Using EI^{2}OS

Figure 17-11 shows the operation flowchart when using EI^{2}OS.

Figure 17-11. Example of Operation Flowchart for Using EI^{2}OS



When using EI^{2}OS, the conversion data protection function allows multiple data values to be transferred to memory reliably without any data loss, even when using continuous conversion.

17.6.2 A/D Conversion Data Protection Function

The conversion data protection function is active when A/D conversion is performed with interrupts enabled.

Explanation of the A/D Conversion Data Protection Function

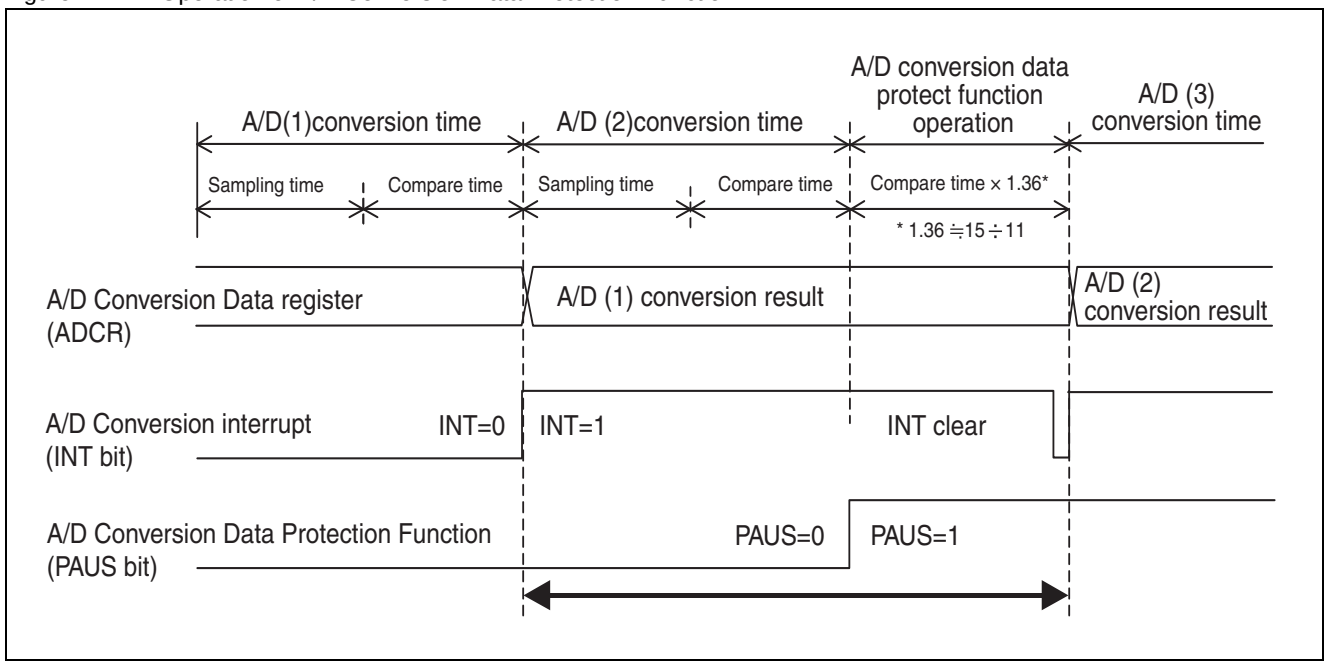
The A/D conversion data protection function prevents data from being lost during A/D conversion.

The 8/10-bit A/D converter has a single A/D data register (ADCR1/0) to store the conversion data and a single buffer register to store the next conversion result. When A/D conversion completes, the converted data is first stored in the buffer register and then transferred to the A/D data register.

The A/D converter operates as follows, depending upon whether the A/D conversion data protection function is on or off.

- The data protection function is turned off when the interrupt request enable bit (ADCS:INTE) is set to "0". If a number of consecutive conversions are performed with the data protection function off, the conversion result is saved in the A/D data register after each conversion. (That is, the register always contains the most recent conversion result.)
- Setting the interrupt request enable bit (ADCS:INTE) to “1” enables the data protection function.
  If A/D conversion is executed continuously in this state, the interrupt request flag bit (ADCS:INT) is set to “1” upon completion of the first conversion. When the next conversion ends up with INT = “1”, the 8/10-bit A/D converter is “paused” immediately before the successive approximation circuit transfers the conversion result to the A/D data register, thereby preventing conversion data from being overwritten. At this time, the pause flag bit in the A/D control status register (ADCS: PAUS) is set to “1”. Clearing the interrupt request flag bit (ADCS:INT) to "0" while A/D conversion is being paused causes the successive approximation circuit to transfer stored data to the A/D data register (See Figure 17-12).

Figure 17-12. Operation of A/D Conversion Data Protection Function



Using the A/D Conversion Data Protection Function

The following describes how to use the A/D conversion data protection function for the cases when EI^{2}OS is and is not used respectively.

Using the A/D Conversion Data Protection Function When EI^{2}OS is not Used

- Set the interrupt request enable bit in the A/D control status register (ADCS:INTE) to “1” and start A/D conversion.
- When A/D conversion completes, the conversion result is stored in the buffer register and then transferred to the A/D data register (ADCR1/ADCR0). At this time, the interrupt request flag bit in the A/D control status register (ADCS:INT) is set to “1”.
- If the interrupt request flag bit (ADCS:INT) is still set to “1” when the next A/D conversion completes, A/D conversion operation is paused before transferring the contents of the buffer register to the A/D data register to prevent the previous value from being overwritten. Also, the pause flag bit in the A/D control status register (ADCS:PAUS) is set to “1”.
- As interrupt requests are enabled in the A/D control status register (ADCS:INTE = 1), an interrupt request is generated when the interrupt request flag bit is set. When the interrupt routine clears the interrupt request flag bit after reading the A/D data register, the pause is released and the value in the buffer register is transferred to the A/D data register. If performing continuous conversion, A/D conversion restarts. Note, however, that the pause flag bit (ADCS:PAUS) is not automatically cleared to “0”. To clear this bit, write “0” to the bit.

Notes:

- The A/D conversion data protection function only operates if interrupt requests are enabled. Always set the interrupt request output enable bit in the A/D control status register (ADCS:INTE) to “1”. If interrupt requests are disabled while conversion is paused (ADCS:INTE = 0), A/D conversion restarts and the data in the A/D data register may be overwritten.
- When performing continuous conversion, always write your program in such a way that it reads the data from the A/D data register before clearing the interrupt request flag bit (ADCS:INT). If the interrupt request flag bit (ADCS:INT) is cleared before reading the data from the A/D data register while A/D conversion is paused, the previous data value will be lost and reading will return the next conversion data value.

Using the A/D Conversion Data Protection Function When EI^{2}OS is Used

Enable use of EI^{2}OS, set the interrupt request enable bit in the A/D control status register (ADCS:INTE) to “1”, and start A/D conversion.

When A/D conversion completes, the conversion result is stored in the buffer register and then transferred to the A/D data register (ADCR). At this time, the interrupt request flag bit in the A/D control status register (ADCS:INT) is set to “1” and EI^{2}OS is started. If the next A/D conversion completes before the previous result is transferred from the A/D data register to memory, A/D conversion operation is paused before transferring the contents of the buffer register to the A/D data register to prevent the previous value from being overwritten. Also, the pause flag bit in the A/D control status register (ADCS:PAUS) is set to “1”.

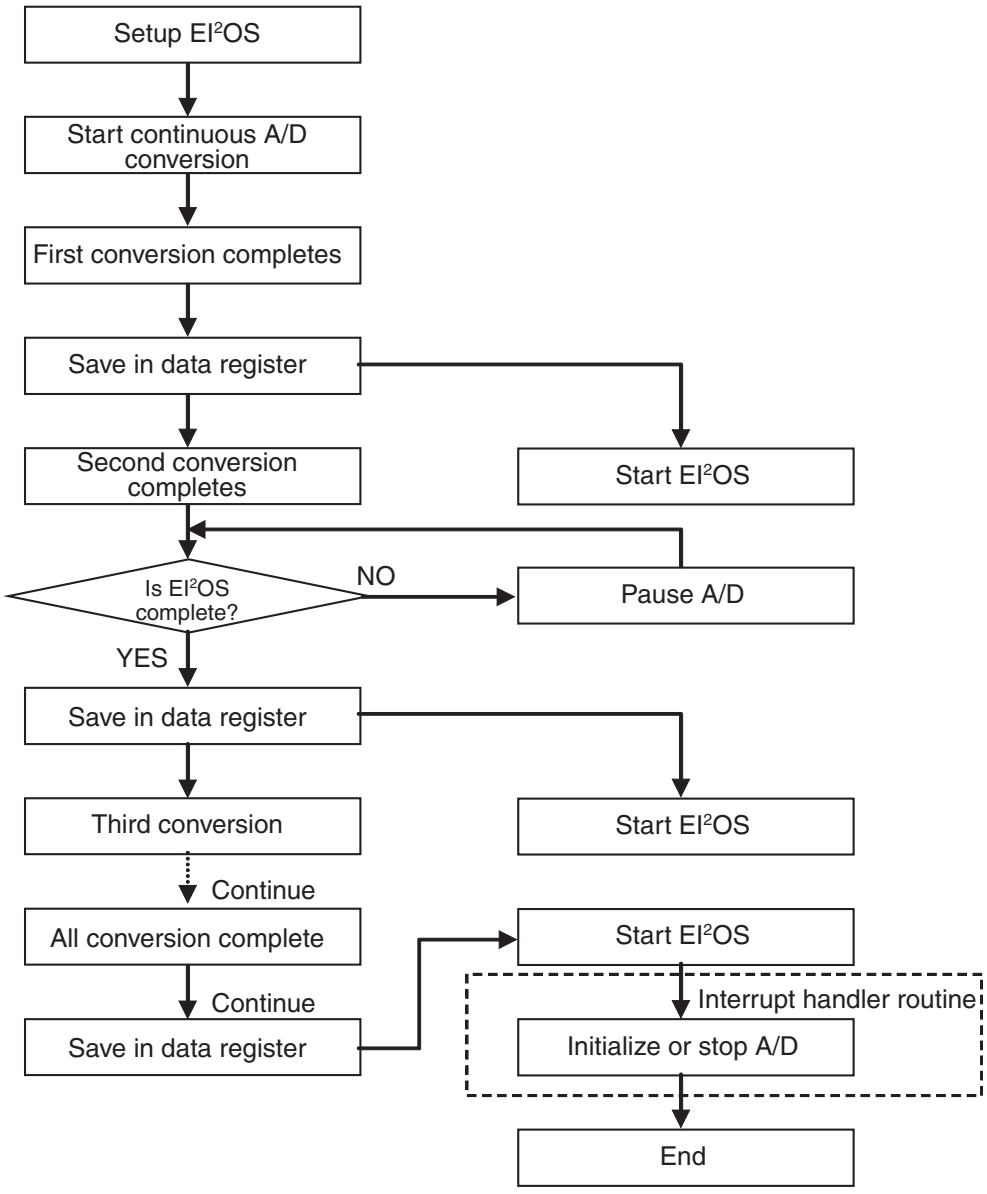
When the EI^{2}OS transfer completes, the interrupt request flag bit (ADCS:INT) is cleared to “0”, the pause is released, and the data from the buffer register is transferred to the A/D data register. If performing continuous conversion, A/D conversion restarts. Note, however, that the pause flag bit (ADCS:PAUS) is not automatically cleared to “0”. To clear this bit, write “0” to the bit.

Notes:

- The A/D conversion data protection function only operates if interrupt requests are enabled. Always set the interrupt request output enable bit in the A/D control status register (ADCS:INTE) to “1”.
- If using the EI^{2}OS function to transfer the A/D conversion results to memory, do not clear the interrupt request flag bit (ADCS:INT = 0). Clearing the interrupt request flag may cause the data in the A/D data register to be modified while transfer is still in progress.
- If using the EI^{2}OS function to transfer the A/D conversion results to memory, do not disable interrupt requests. If interrupt requests are disabled (ADCS:INTE=0) while conversion is paused, A/D conversion will restart and the data in the A/D data register may be modified.

Figure 17-13 shows the flowchart of the data protection function when EI^{2}OS is used.

Figure 17-13. Flowchart of Data Protection Function When EI^{2}OS Used



Note: The flowchart for the case when the A/D converter is stopped is omitted.

17.7 Precautions When Using the 8/10-Bit A/D Converter

This section describes points to note when using the 8/10-bit A/D converter.

Precautions When Using the 8/10-Bit A/D Converter

Analog input pins

The A/D inputs share pins with the port 6, port 7, and port 9 I/O pins and are selected using the port 6, port 7, and port 9 direction registers (DDR6, DDR7, and DDR9) and the analog input enable register (ADER). To use a pin as an analog input, set the corresponding bit in the DDR6, DDR7, or DDR9 register to “0” to set the port as an input and set analog input mode (ADEx = 1) in the ADER register to fix the state of the port input gate. Note that an input leak current will flow in the gate if an intermediate level signal is input while the pin is set to port input mode (ADEx = 0).

Sequence for turning on A/D converter power supply and analog inputs

Ensure that the A/D converter power supply (AV_{CC}) and analog inputs (AN0 to AN19) are not turned on until after (or at the same time) as the digital power supply (V_{CC}). Similarly, when turning off the power, turn off the digital power supply (V_{CC}) at the same time or after turning off the A/D converter power supply and analog inputs.

Do not let AVRH to exceed AV_{CC} when the power is turned on or off.

A/D converter power supply voltage

To prevent latch-up, ensure that the A/D converter power supply (AV_{CC}) does not exceed the digital power supply (V_{CC}) voltage.

18. Multi-function Serial Interface

This chapter describes an overview of the multi-function serial interface and the configuration of the registers. For details, refer to sections 18.4 through 18.7 explaining each mode.

18.1 Overview of the Multi-function Serial Interface

18.2 Pins of the Multi-function Serial Interface

18.3 Registers of the Multi-function Serial Interface

18.4 UART (Asynchronous Serial Interface)

18.5 CSIO (Clock Synchronous Serial Interface)

18.6 LIN-UART (LIN Communication Control UART)

18.7 I^2C Interface

18.1 Overview of the Multi-function Serial Interface

The multi-function serial interface supports asynchronous serial interface, synchronous serial interface, LIN function and I^2C interface by setting its registers.

Functions of the UART (Asynchronous Serial Interface)

		Functions
1	Data	■ Full-duplex double buffer ■ Transmission/reception
2	Serial input	After oversampling three times, the reception value is determined by a majority of the sampling values.
3	Transfer format	Asynchronous
4	Baud rate	■ Dedicated baud rate generator (15-bit reload counter configuration) ■ An external clock input can be adjusted using the reload counter.
5	Data length	5 bits to 9 bits (for normal mode), 7 bits, 8 bits (for multiprocessor mode)
6	Signal format	NRZ (Non Return to Zero), inverted NRZ
7	Start bit detection	■ Synchronization with start bit falling edge (for NRZ format) ■ Synchronization with start bit rising edge (for inverted NRZ format)
8	Reception error detection	■ Framing error ■ Overrun error ■ Parity error *
9	Interrupt requests	■ Reception interrupts (Reception completion, framing error, overrun error, parity error *) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Both transmission and reception have extended intelligent I/O service (EI^2OS) and DMA functions.
10	Master/slave type communication function (Multiprocessor mode)	Communication between 1 (master) and n (slave) is enabled. (Both master and slave systems are supported.)

*: A parity error occurs only in normal mode.

Functions of the CSIO (Clock Synchronous Serial Interface)

		Functions
1	Data buffer	■ Full-duplex double buffer ■ Transmission/reception
2	Transfer format	■ Clock synchronous (without start bit/stop bit) ■ Master/slave function ■ SPI compliant (for both master/slave supported)
3	Baud rate	■ Dedicated baud rate generator (consisting of a 15-bit reload counter, in master operation) ■ External clock input enabled (in slave operation)
4	Data length	5 bits to 9 bits variable
5	Reception error detection	Overrun error
6	Interrupt requests	■ Reception interrupts (Reception completion, overrun error) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Both transmission and reception have extended intelligent I/O service (EI^{2}OS) and DMA transfer support functions.
7	Synchronous mode	Master or slave function
8	Pin access	A serial data output pin can be set to “1”.

Functions of the LIN-UART (LIN Communication Control UART)

		Functions
1	Data buffer	■ Full-duplex double buffer ■ Transmission/reception
2	Serial input	After oversampling three times using machine clock, the reception value is determined by a majority of the sampling values.
3	Transfer mode	Asynchronous
4	Baud rate	■ Dedicated baud rate generator (consisting of a 15-bit reload counter) ■ An external clock can be adjusted using the reload counter.
5	Data length	8 bits
6	Signal format	NRZ (Non Return to Zero)
7	Start bit detection	Synchronization with start bit falling edge
8	Reception error detection	■ Framing error ■ Overrun error
9	Interrupt requests	■ Reception interrupts (Reception completion, framing error, overrun error) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Status interrupt (LIN synch break detected) ■ Interrupt request to ICU (LIN synch field detected: LSYN) ■ Both transmission and reception have extended intelligent I/O service (EI^{2}OS) and DMA functions.
10	LIN bus options	■ LIN protocol Revision 2.0 supported ■ Master device operation ■ Slave device operation ■ LIN Synch break generation (variable length between 13 bits and 16 bits) ■ Synch Delimiter generation (variable length between 1 bit and 4 bits) ■ LIN Synch break detection ■ Detection of start/stop edge of LIN synch field connecting to an input capture

Functions of the I²C -UART (I²C Communication Control UART)

		Functions
1	Data buffer	■ Full-duplex double buffer ■ Transmission/reception
2	I²C bus functions	■ Master/slave transmission/reception function ■ Arbitration function ■ Clock synchronization function ■ Transmission direction detection function ■ Function to generate and detect a repeated start condition ■ Bus error detection function ■ General call addressing function ■ 7-bit addressing as master and slave ■ Interrupts can be generated during transmission and when a bus error occurs. ■ 10-bit addressing function can be supported by program.

18.2 Pins of the Multi-function Serial Interface

This section lists and details the pins, interrupt sources, and registers of the Multi-function Serial Interface.

Pins the Multi-function Serial Interface

The Multi-function Serial Interface pins also serve as general-purpose ports. Table 18-1 lists the pin functions, I/O formats,and settings required to use Multi-function Serial Interface.

Table 18-1. Pins of Multi-function Serial Interface

Pin name	Pin function	I/O type	Pull-up select	Standby control	Setting to use pin
P84/UI0 P30/UI1 P33/UI2 P42/UI3 P95/(UI3) P45/UI4 P71/(UI4) P74/UI5 P80/UI6	Port I/O, serial data input	CMOS output, CMOS hysteresis I^2C Characteristics CMOS hysteresis (*1)	None	Yes	Sets to input port (DDR: corresponding bit = 0)
P85/UO0 P31/UO1 P34/UO2 P43/UO3 P96/(UO3) P46/UO4 P72/(UO4) P75/UO5 P81/UO6	Port I/O, serial data output				Set to output enable mode (SMRn:SOE = 1)
P86/UCK0 P32/UCK1 P35/UCK2 P44/UCK3 P97/(UCK3) P47/UCK4 P73/(UCK4) P76/UCK5 P82/UCK6	Port I/O, Serial clock input/output				Set as an input port when a clock is inputted (DDR: corresponding bit = 0)
					Set to output enable mode when a clock is outputted (SMRn: SCKE = 1)

*1:Selection of input level is possible according to Serial input level selection register(ILSR0, ILSR1, ILSR2). Please refer to “8.2 Registers for I/O Port”.

18.3 Registers of the Multi-function Serial Interface

A register list of the multi-function serial interface is shown below.

Register List of the Multi-function Serial Interface

Table 18-2. Register List of the Multi-function Serial Interface

	Address		bit15bit8	bit7bit0
Mode 0 to 3*	ch.0:000021$_H$ ch.1:00002B$_H$	ch.0:000020$_H$ ch.1:00002A$_H$	SCR (Serial control register)	SMR (Serial mode register)
Mode 4*	ch.2:00003F$_H$ ch.3:000049$_H$ ch.4:000053$_H$ ch.5:00005D$_H$ ch.6:007791$_H$	ch.2:00003E$_H$ ch.3:000048$_H$ ch.4:000052$_H$ ch.5:00005C$_H$ ch.6:007990$_H$	IBCR (I^2C bus control register)	
Mode 0 to 3*	ch.0:000023$_H$ ch.1:00002D$_H$	ch.0:000022$_H$ ch.1:00002C$_H$	SSR (Serial status register)	ESCR (Extended communication control register)
Mode 4*	ch.2:000041$_H$ ch.3:00004B$_H$ ch.4:000055$_H$ ch.5:00005F$_H$ ch.6:007993$_H$	ch.2:000040$_H$ ch.3:00004A$_H$ ch.4:000054$_H$ ch.5:00005E$_H$ ch.6:007992$_H$		IBSR (I^2C bus status register)
Mode 0 to 2*	ch.0:000025$_H$ ch.1:00002F$_H$	ch.0:000024$_H$ ch.2:00002E$_H$	RDR1/TDR1 (Reception/transmission data register 1)	RDR0/TDR0 (Reception/transmission data register 0)
Mode 3 to 4*	ch.2:000043$_H$ ch.3:00004D$_H$ ch.4:000057$_H$ ch.5:000061$_H$ ch.6:007995$_H$	ch.3:000042$_H$ ch.4:00004C$_H$ ch.5:000056$_H$ ch.6:000060$_H$ ch.7:007994$_H$	-	
Mode 0 to 4*	ch.0:000027$_H$ ch.1:000031$_H$ ch.2:000045$_H$ ch.3:00004F$_H$ ch.4:000059$_H$ ch.5:000063$_H$ ch.6:007997$_H$	ch.0:000026$_H$ ch.1:000030$_H$ ch.2:000044$_H$ ch.3:00004E$_H$ ch.4:000058$_H$ ch.5:000062$_H$ ch.6:007996$_H$	BGR1 (Baud rate generator register 1)	BGR0 (Baud rate generator register 0)
Mode 0 to 3*	ch.0:000029$_H$ ch.1:000033$_H$	ch.0:000028$_H$ ch.1:000032$_H$	-	-
Mode 4*	ch.2:000047$_H$ ch.3:000051$_H$ ch.4:00005B$_H$ ch.5:000065$_H$ ch.6:007999$_H$	ch.2:000046$_H$ ch.3:000050$_H$ ch.4:00005A$_H$ ch.5:000064$_H$ ch.6:007998$_H$	ISMK (7-bit slave address mask register)	ISBA (7-bit slave address register)

*: Mode 0 indicates asynchronous normal mode, mode 1 indicates asynchronous multiprocessor mode, mode 2 indicates clock synchronous mode, mode 3 indicates LIN mode, and mode 4 indicates I^2C mode. Each mode is determined by setting MD2, MD1 and MD0 bits of the serial mode register.

Table 18-3. Bit Assignment of the Registers of the Multi-function Serial Interface

Register	Mode	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	0/1	UPCL	-	-	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	Reser-ved	SBL	BDS	SCKE	SOE
	2		MS	SPI										SCINV			
	3			LBR										SBL	-		
IBCR/SMR	4	MSS	ACT/ SCC	ACKE	WSEL	CNDE	INTE	BER	INT					RIE	TIE	ITST1	ITST0
SSR/ESCR	0/1	REC	-	PE	FRE	ORE	RDRF	TDRE	TBI	-	-	INV	PEN	P	L2	L1	L0
	2			-	-					SOP		-	-	-			
	3			LBD	FRE					-			LBIE	LBL1	LBL0	DEL1	DEL0
SSR/IBSR	4		TSET	-	-				-	FBT	RACK	RSA	TRX	AL	RSC	SPC	BB
TDR (RDR)	0 to 2	-							D8 (AD)	D7	D6	D5	D4	D3	D2	D1	D0
	3 to 4								-								
BGR0, BGR1	0/1/3	EXT	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	2/4	-															
ISMK/ISBA	0 to 3	-								-							
	4	EN	SM6	SM5	SM4	SM3	SM2	SM1	SM0	SAEN	SA6	SA5	SA4	SA3	SA2	SA1	SA0

Operation Modes

The multi-function serial interface operates in five different modes. The mode is determined by MD2, MD1, and MD0 of the serial mode register (SMR).

Table 18-4. Operation Modes of the Multi-function Serial Interface

MD2	MD1	MD0	Mode
0	0	0	Operation mode 0 (Asynchronous normal mode)
0	0	1	Operation mode 1 (Asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (Clock synchronous mode)
0	1	1	Operation mode 3 (LIN mode)
1	0	0	Operation mode 4 (I^2C mode)

Note: Settings other than the above are prohibited.

18.4 UART (Asynchronous Serial Interface)

This section describes the UART functions supported by operation modes 0 and 1 among multi-function serial interface functions.

UART (Asynchronous Serial Interface)

Overview of the UART (Asynchronous Serial Interface)

Registers of UART (Asynchronous Serial Interface)

- Serial Control Register (SCR)
- Serial Mode Register (SMR)
- Serial Status Register (SSR)
- Extended Communication Control Register (ESCR)
- Reception Data Register/Transmission Data Register (RDR/TDR)
- Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

Interrupts of the UART

- Reception Interrupt Generation and the Flag Set Timing
- Transmission Interrupt Generation and the Flag Set Timing

Operation of the UART

Dedicated Baud Rate Generator

Baud Rate Setting

Setting Procedure and the Program Flow for Operation Mode 0 (Asynchronous Normal Mode)

Setting Procedure and the Program Flow for Operation Mode 1 (Asynchronous Multiprocessor Mode)

Functions of the UART (Asynchronous Serial Interface)

The UART (asynchronous serial interface) is a general-purpose serial data communication interface to perform asynchronous communication (start-stop synchronization) with external devices. It supports bidirectional communication function (normal mode), and master/slave type communication function (multiprocessor mode: both master/slave supported).

		Functions
1	Data	■ Full-duplex double buffer ■ Transmission/reception
2	Serial input	After oversampling three times, the reception value is determined by a majority of the sampling values.
3	Transfer format	Asynchronous
4	Baud rate	■ Dedicated baud rate generator (15-bit reload counter configuration) ■ An external clock input can be adjusted using the reload counter.
5	Data length	■ 5 bits to 9 bits (for normal mode), 7 bits, 8 bits (for multiprocessor mode)
6	Signal format	NRZ (Non Return to Zero), inverted NRZ
7	Start bit detection	■ Synchronization with start bit falling edge (for NRZ format) ■ Synchronization with start bit rising edge (for inverted NRZ format)
8	Reception error detection	■ Framing error ■ Overrun error ■ Parity error *
9	Interrupt requests	■ Reception interrupts (Reception completion, framing error, overrun error, parity error *) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Both transmission and reception have extended intelligent I/O service (EI^{2}OS) and DMA functions.
10	Master/slave type communication function (Multiprocessor mode)	Communication between 1 (master) and n (slave) is enabled. (Both master and slave systems are supported.)

*: A parity error occurs only in normal mode.

18.4.1 Registers of UART (Asynchronous Serial Interface)

Registers of UART (Asynchronous Serial Interface) are as follows.

Register List of the UART (Asynchronous Serial Interface)

Figure 18-5 lists a register list of the UART (asynchronous serial interface).

Table 18-5. Register List of the UART (Asynchronous Serial Interface)

	Address		bit15 bit8	bit7 bit0
UART	ch.0:000021$_H$ ch.1:00002B$_H$ ch.2:00003F$_H$ ch.3:000049$_H$ ch.4:000053$_H$ ch.5:00005D$_H$ ch.6:007791$_H$	ch.0:000020$_H$ ch.1:00002A$_H$ ch.2:00003E$_H$ ch.3:000048$_H$ ch.4:000052$_H$ ch.5:00005C$_H$ ch.6:007990$_H$	SCR (Serial control register)	SMR (Serial mode register)
	ch.0:000023$_H$ ch.1:00002D$_H$ ch.2:000041$_H$ ch.3:00004B$_H$ ch.4:000055$_H$ ch.5:00005F$_H$ ch.6:007993$_H$	ch.0:000022$_H$ ch.1:00002C$_H$ ch.2:000040$_H$ ch.3:00004A$_H$ ch.4:000054$_H$ ch.5:00005E$_H$ ch.6:007992$_H$	SSR (Serial status register)	ESCR (Extended communication control register)
	ch.0:000025$_H$ ch.1:00002F$_H$ ch.2:000043$_H$ ch.3:00004D$_H$ ch.4:000057$_H$ ch.5:000061$_H$ ch.6:007995$_H$	ch.0:000024$_H$ ch.2:00002E$_H$ ch.3:000042$_H$ ch.4:00004C$_H$ ch.5:000056$_H$ ch.6:000060$_H$ ch.7:007994$_H$	RDR1/TDR1 (Reception/transmission data register 1)	RDR0/TDR0 (Reception/transmission data register 0)
	ch.0:000027$_H$ ch.1:000031$_H$ ch.2:000045$_H$ ch.3:00004F$_H$ ch.4:000059$_H$ ch.5:000063$_H$ ch.6:007997$_H$	ch.0:000026$_H$ ch.1:000030$_H$ ch.2:000044$_H$ ch.3:00004E$_H$ ch.4:000058$_H$ ch.5:000062$_H$ ch.6:007996$_H$	BGR1 (Baud rate generator register 1)	BGR0 (Baud rate generator register 0)
	ch.0:000029$_H$ ch.1:000033$_H$ ch.2:000047$_H$ ch.3:000051$_H$ ch.4:00005B$_H$ ch.5:000065$_H$ ch.6:007999$_H$	ch.0:000028$_H$ ch.1:000032$_H$ ch.2:000046$_H$ ch.3:000050$_H$ ch.4:00005A$_H$ ch.5:000064$_H$ ch.6:007998$_H$	-	-

Table 18-6. Bit Assignment of the UART (Asynchronous Serial Interface)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	-	-	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	Reser-ved	SBL	BDS	SCKE	SOE
SSR/ESCR	REC	-	PE	FRE	ORE	RDRF	TDRE	TBI	-	-	INV	PEN	P	L2	L1	L0
TDR (RDR)	-							D8 (AD)	D7	D6	D5	D4	D3	D2	D1	D0
BGR0, BGR1	EXT	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0

Operation Modes

The UART (asynchronous serial interface) operates in two different modes. The mode is determined by MD2, MD1 and MD0 of the serial mode register (SMR).

Table 18-7. Operation Modes of the UART (Asynchronous Serial Interface)

Operation mode	MD2	MD1	MD0	Type
0	0	0	0	UART0 (asynchronous normal mode)
1	0	0	1	UART1 (asynchronous multiprocessor mode)

Serial Control Register (SCR)

Figure 18-1 shows the bit configuration of the serial control register (SCR) and Table 18-8 shows the functions of each bit.

The serial control register (SCR) enables/disables transmission/reception, transmission/reception interrupts, and transmission bus idle interrupts and performs UART reset.

Figure 18-1. Bit Configuration of the Serial Control Register (SCR)

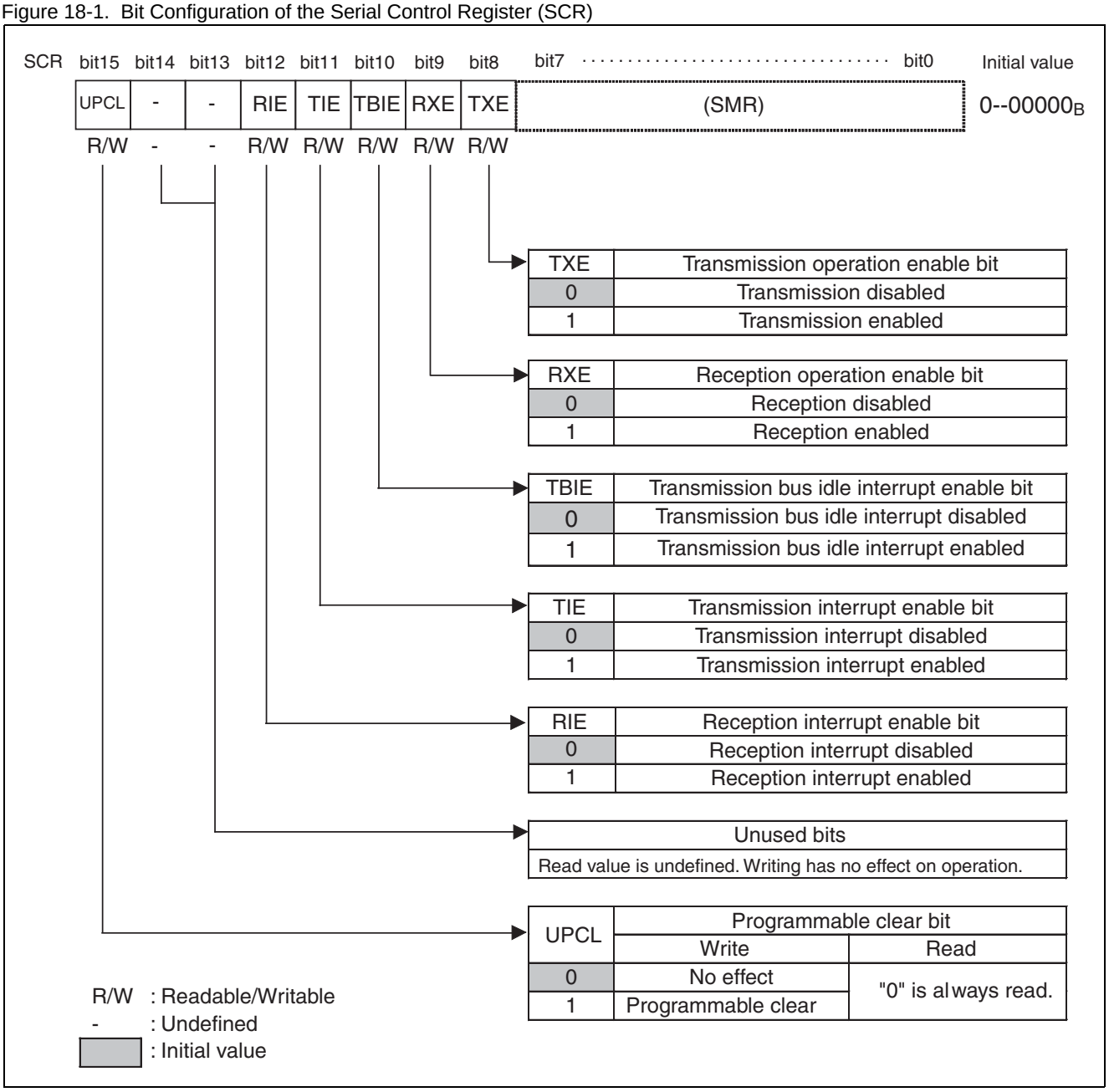


Table 18-8. Function Description of Each Bit of the Serial Control Register (SCR)

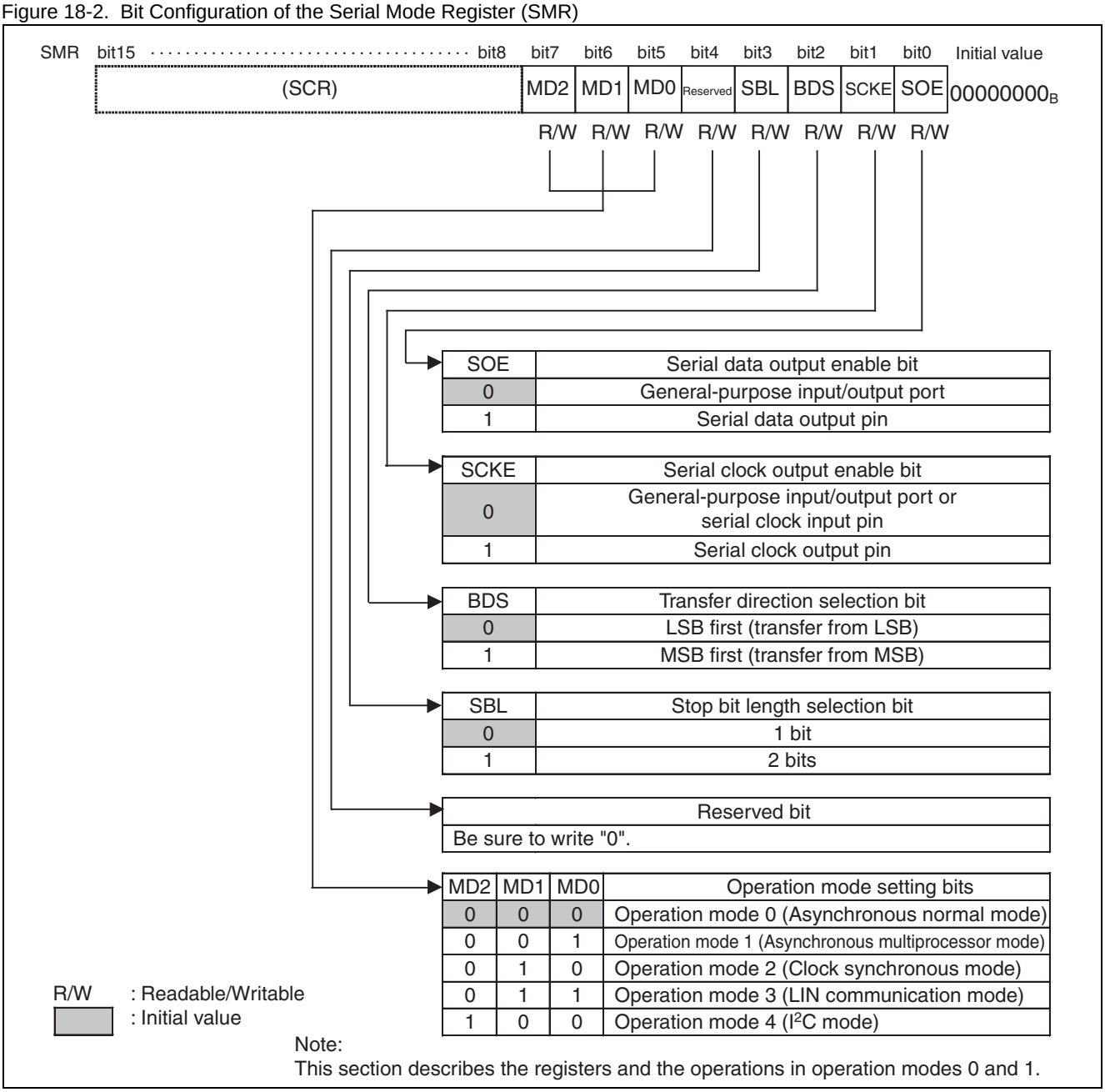
Bit name		Function
bit15	UPCL: Programmable clear bit	This bit initializes the internal state of the UART. When “1” is set: ■ UART is reset directly (software reset). However, the register settings are retained. If a register is in transmission/reception state at that time, it will be disconnected immediately. ■ The baud rate generator reloads the setting values in BGR0, BGR1 registers and restarts. ■ All transmission/reception interrupt sources (PE, FRE, ORE, RDRF, TDRE, and TBI) are initialized to (000011_B). When “0” is set: No effect “0” is always read when read. **Note:** After interrupt disabled is set, execute a programmable clear.
bit14, bit13	Unused bits	When read: Values are undefined. When write: No effect
bit12	RIE: Reception interrupt enable bit	■ This bit enables/disables the output of reception interrupt requests to the CPU. ■ If RIE bit and reception data flag bit (RDRF) are “1”, or if any one of the error flag bits (PE, ORE, FRE) is “1”, a reception interrupt request is output.
bit11	TIE: Transmission interrupt enable bit	■ This bit enables/disables the output of transmission interrupt requests to the CPU. ■ If TIE bit and TDRE bit are “1”, a transmission interrupt request is output.
bit10	TBIE: Transmission bus idle interrupt enable bit	■ This bit enables/disables the output of transmission bus idle interrupt requests to the CPU. ■ If TBIE bit and TBI bit are “1”, a transmission bus idle interrupt request is output.
bit9	RXE: Reception operation enable bit	■ This bit enables/disables the reception operation of the UART. ■ If “0” is set: Reception operation is disabled. ■ If “1” is set: Reception operation is enabled. **Notes:** ■ Even if the reception operation is enabled (RXE=1), it does not start until a start bit falling edge (when NRZ format (INV=0) is set) is input. (When inverted NRZ format (INV=1) is set, the reception operation does not start until a rising edge is input.) ■ If the reception operation is disabled (RXE=0) during reception, the operation is stopped immediately.
bit8	TXE: Transmission operation enable bit	This bit enables/disables the transmission operation of the UART. ■ If “0” is set: Transmission operation is disabled. ■ If “1” is set: Transmission operation is enabled. **Note:** If the transmission operation is disabled (TXE=0) during transmission, the operation is stopped immediately.

Serial Mode Register (SMR)

Figure 18-2 shows the bit configuration of the serial mode register (SMR) and Table 18-9 shows the functions of each bit.

The serial mode register (SMR) sets the operation mode, selects transfer direction, data length and stop bit length, and enables/disables the outputs to the serial data and serial clock pins.

Figure 18-2. Bit Configuration of the Serial Mode Register (SMR)



SOE	Serial data output enable bit
0	General-purpose input/output port
1	Serial data output pin

SCKE	Serial clock output enable bit
0	General-purpose input/output port or serial clock input pin
1	Serial clock output pin

BDS	Transfer direction selection bit
0	LSB first (transfer from LSB)
1	MSB first (transfer from MSB)

SBL	Stop bit length selection bit
0	1 bit
1	2 bits

Reserved bit
Be sure to write "0".

MD2	MD1	MD0	Operation mode setting bits
0	0	0	Operation mode 0 (Asynchronous normal mode)
0	0	1	Operation mode 1 (Asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (Clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^2C mode)

Table 18-9. Function Description of Each Bit of the Serial Mode Register (SMR)

Bit name		Function
bit7 to bit5	MD2 to MD0: Operation mode setting bits	These bits set the operation mode for the asynchronous serial interface. "000_B": Operation mode 0 (asynchronous normal mode) is set. "001_B": Operation mode 1 (asynchronous multiprocessor mode) is set. "010_B": Operation mode 2 (clock synchronous mode) is set. "011_B": Operation mode 3 (LIN communication mode) is set. "100_B": Operation mode 4 (I^2C mode) is set. This section describes the registers and the operations in operation mode 0 (asynchronous normal mode) and in operation mode 1 (asynchronous multiprocessor mode). **Notes:** ■ Settings other than the above are prohibited. ■ To switch the operation mode, do so after executing a programmable clear (SCR:UPCL=1). ■ Set each register after setting the operation mode.
bit4	Reserved bit	Be sure to write "0".
bit3	SBL: Stop bit length selection bit	This bit sets the bit length of the stop bit (frame end mark of transmission data). When "0" is set: Stop bit length is set to 1 bit. When "1" is set: Stop bit length is set to 2 bits. **Notes:** ■ For reception, only the first bit of the stop bit is always detected. ■ Set this bit when the transmission is disabled (TXE=0).
bit2	BDS: Transfer direction selection bit	This bit selects whether to transfer the transfer serial data from the least significant bit side first (LSB first, BDS=0) or to transfer it from the most significant bit side first (MSB first, BDS=1). **Note:** Set this bit when the transmission and reception operations are disabled (TXE=RXE=0).
bit1	SCKE: Serial clock output enable bit	This bit controls input/output ports of the serial clock. When "0" is set: UCK pin becomes a general-purpose input/output port or a serial clock input pin. When "1" is set: UCK pin becomes a serial clock output pin and outputs the clock. **Note:** When the UCK pin is used as a serial clock input (SCKE=0), set a general-purpose input/output port to an input port. Also, select the external clock (BGR:EXT=1) using the external clock selection bit. **Reference:** If the UCK pin is set to serial clock output (SCKE=1), it functions as a serial clock output pin regardless of the general-purpose input/output port (DDR) settings.
bit0	SOE: Serial data output enable bit	This bit enables/disables the output of the serial data. When "0" is set: UO pin becomes a general-purpose input/output port. When "1" is set: UO pin becomes a serial data output pin (UO). **Reference:** When this bit is set to serial data output (SOE=1), the UO pin functions as a UO pin regardless of the general-purpose input/output port (DDR) settings.

Serial Status Register (SSR)

The serial status register (SSR) checks transmission/reception statuses and reception error flags, and clears the flags.

Figure 18-3 shows the bit configuration of the serial status register (SSR) and Table 18-10 shows the functions of each bit.

Figure 18-3. Bit Configuration of the Serial Status Register (SSR)

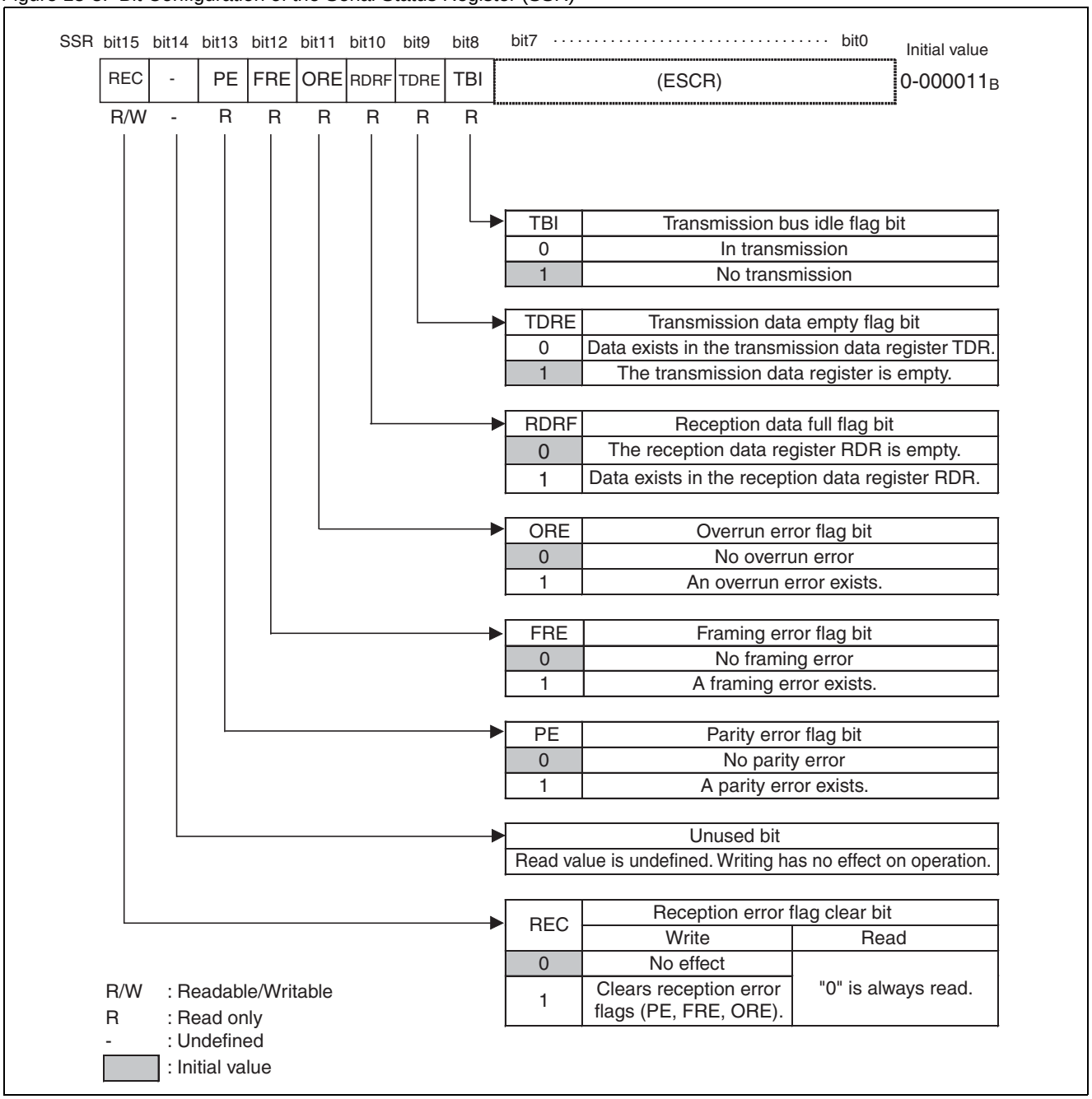


Table 18-10. Function Description of Each Bit of the Serial Status Register (SSR)

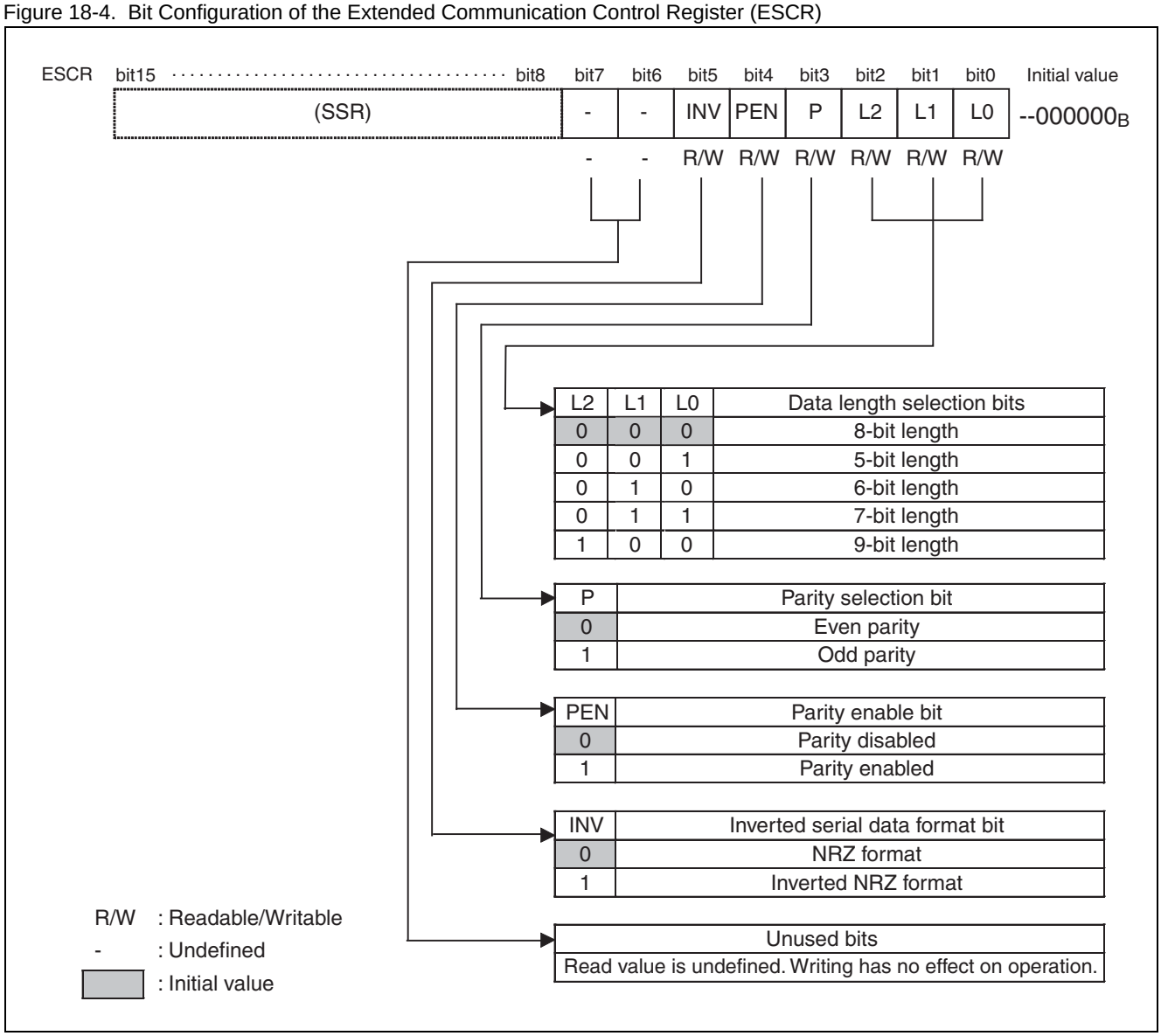
Bit name		Function
bit15	REC: Reception error flag clear bit	This bit clears PE, FRE, and ORE flags of the serial status register (SSR). ■ When “1” is written, error flags are cleared. ■ When “0” is written, this writing has no effect. When read, “0” is always read.
bit14	Unused bit	When read: The value is undefined. When write: No effect
bit13	PE: Parity error flag bit (operable only in operation mode 0)	■ If SMR: PEN=1, this bit is set to “1” when a parity error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the PE bit and RIE bit in the SCR are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit12	FRE: Framing error flag bit	■ This bit is set to “1” when a framing error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the FRE bit and RIE bit are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit11	ORE: Overrun error flag bit	■ This bit is set to “1” when an overrun error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the ORE bit and RIE bit are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit10	RDRF: Reception data full flag bit	■ This flag indicates the status of the reception data register (RDR). ■ This bit is set to “1” when reception data is loaded into the RDR and is cleared to “0” when the reception data register (RDR) is read. ■ A reception interrupt request is output when the RDRF bit and RIE bit are “1”.
bit9	TDRE: Transmission data empty flag bit	■ This flag indicates the status of the transmission data register (TDR). ■ This bit becomes “0” when transmission data is written to TDR, indicating that valid data exists in the TDR. This bit becomes “1” when the data is loaded into the transmission shift register and transmission starts, indicating that valid data does not exist in the TDR. ■ A transmission interrupt request is output when the TDRE bit and TIE bit are “1”. ■ The TDRE bit becomes “1” when the UPCL bit in the serial control register (SCR) is set to “1”.
bit8	TBI: Transmission bus idle flag bit	■ This bit indicates that the UART is not in transmission operations. ■ This bit becomes “0” when transmission data is written to the transmission data register (TDR). ■ This bit becomes “1” when the transmission data register is empty (TDRE=1) and no transmission operation is performed. ■ The TBI bit becomes “1” when the UPCL bit of the serial control register (SCR) is set to “1”. ■ A transmission interrupt request is output when this bit is “1” and the transmission bus idle interrupt is being enabled (SCR:TBIE=1).

Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) sets the transmission/reception data length, enables/disables a parity bit, selects a parity bit, and sets the inverted serial data format.

Figure 18-4 shows the bit configuration of the extended communication control register (ESCR) and Table 18-11 shows the functions of each bit.

Figure 18-4. Bit Configuration of the Extended Communication Control Register (ESCR)



L2	L1	L0	Data length selection bits
0	0	0	8-bit length
0	0	1	5-bit length
0	1	0	6-bit length
0	1	1	7-bit length
1	0	0	9-bit length

P	Parity selection bit
0	Even parity
1	Odd parity

PEN	Parity enable bit
0	Parity disabled
1	Parity enabled

INV	Inverted serial data format bit
0	NRZ format
1	Inverted NRZ format

Unused bits
Read value is undefined. Writing has no effect on operation.

Table 18-11. Function Description of Each Bit of the Extended Communication Control Register (ESCR)

Bit name		Function
bit7, bit6	Unused bits	When read: The value is undefined. When write: No effect
bit5	INV: Inverted serial data format bit	This bit selects the serial data format either NRZ format or inverted NRZ format.
bit4	PEN: Parity enable bit (operable only in operation mode 0)	This bit sets whether to add a parity bit (during reception) and to detect it (during transmission). ■ When “0” is set: A parity bit is not added. ■ When “1” is set: A parity bit is added. **Note:** In operation mode 1, this bit is fixed to "0" internally.
bit3	P: Parity selection bit (operable only in operation mode 0)	When parity is provided (ESCR: PEN=1), this bit sets either an odd parity “1” or even parity “0”. ■ When “0” is set: An even parity is set. ■ When “1” is set: An odd parity is set.
bit2 to bit0	L2 to L0: Data length selection bits	These bits select the data length for transmission/reception data. ■ When “000_B” is set: The data length is set to 8 bits. ■ When “001_B” is set: The data length is set to 5 bits. ■ When “010_B” is set: The data length is set to 6 bits. ■ When “011_B” is set: The data length is set to 7 bits. ■ When “100_B” is set: The data length is set to 9 bits. **Notes:** ■ Settings other than the above are prohibited. ■ In operation mode 1, set the data length to 7 bits or 8 bits. Other settings are prohibited.

Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data register and transmission data register are located at the same address. When reading the address, it functions as a reception data register, and when writing, it functions as a transmission data register.

Figure 18-5 shows the bit configuration of the reception data register (RDR).

Figure 18-5. Bit Configuration of the Reception Data Register (RDR)

Reception Data Register (RDR)

bit15	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
	D8	D7	D6	D5	D4	D3	D2	D1	D0	000000000_B
	R	R	R	R	R	R	R	R	R	

R: Read only

The reception data register (RDR) is a 9-bit data buffer register for serial data reception.

- A serial data signal transmitted to the serial input pin (UI pin) is converted in the shift register and stored in the reception data register (RDR).
- Depending on the data length, “0” is entered to the upper bits as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	0	X	X	X	X	X	X	X	X
7 bits	0	0	X	X	X	X	X	X	X
6 bits	0	0	0	X	X	X	X	X	X
5 bits	0	0	0	0	X	X	X	X	X

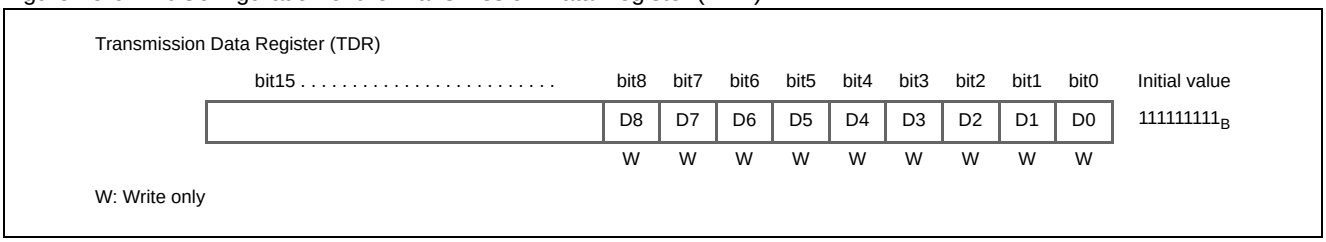
(X indicates a reception data bit)

- When reception data is stored into the reception data register (RDR), the reception data full flag bit (SSR:RDRF) is set to “1”. If reception interrupt is being enabled (SSR:RIE=1), a reception interrupt request is generated.
- Read the reception data register (RDR) while the reception data full flag bit (SSR:RDRF) is “1”. The reception data full flag bit (SSR:RDRF) is cleared to “0” automatically when the reception data register (RDR) is read.
- When a reception error occurs (any one of SSR:PE, ORE and FRE is “1”), data in the reception data register (RDR) becomes invalid.
- In operation mode 1 (multiprocessor mode), the operation becomes 7-bit or 8-bit length operation, and a received AD bit is stored into D8 bit.
- In the case of 9-bit length transfer, and in operation mode 1, reading of the RDR is performed with 16-bit access.

Transmission Data Register (TDR)

Figure 18-6 shows the bit configuration of the transmission data register.

Figure 18-6. Bit Configuration of the Transmission Data Register (TDR)



The transmission data register (TDR) is a 9-bit data buffer register for serial data transmission.

- When data to be transmitted is written into the transmission data register (TDR) while transmission operations are enabled (SCR:TXE=1), the transmission data is transferred to the shift register for transmission, then converted into serial data, and transmitted from the serial data output pin (UO pin).
- Depending on the data length, data becomes invalid from the upper bits as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	Invalid	X	X	X	X	X	X	X	X
7 bits	Invalid	Invalid	X	X	X	X	X	X	X
6 bits	Invalid	Invalid	Invalid	X	X	X	X	X	X
5 bits	Invalid	Invalid	Invalid	Invalid	X	X	X	X	X

- The transmission data empty flag (SSR:TDRE) is cleared to “0” when transmission data is written into the transmission data register (TDR).
- The transmission data empty flag (SSR:TDRE) is set to “1” when transmission data is transferred to the shift register for transmission and transmission starts.
- When the transmission data empty flag (SSR:TDRE) is “1”, transmission data can be written. If transmission interrupt is being enabled, a transmission interrupt is generated. Write transmission data at the time of transmission interrupt generation or while the transmission data empty flag (SSR:TDRE) is “1”.
- When the transmission data empty flag (SSR:TDRE) is “0”, transmission data cannot be written.
- In operation mode 1 (multiprocessor mode), the operation becomes 7-bit or 8-bit length operation, and an AD bit transmission is performed by writing to D8 bit.
- In the case of 9-bit length transfer, and in operation mode 1, writing to the TDR is performed with 16-bit access.

Note:

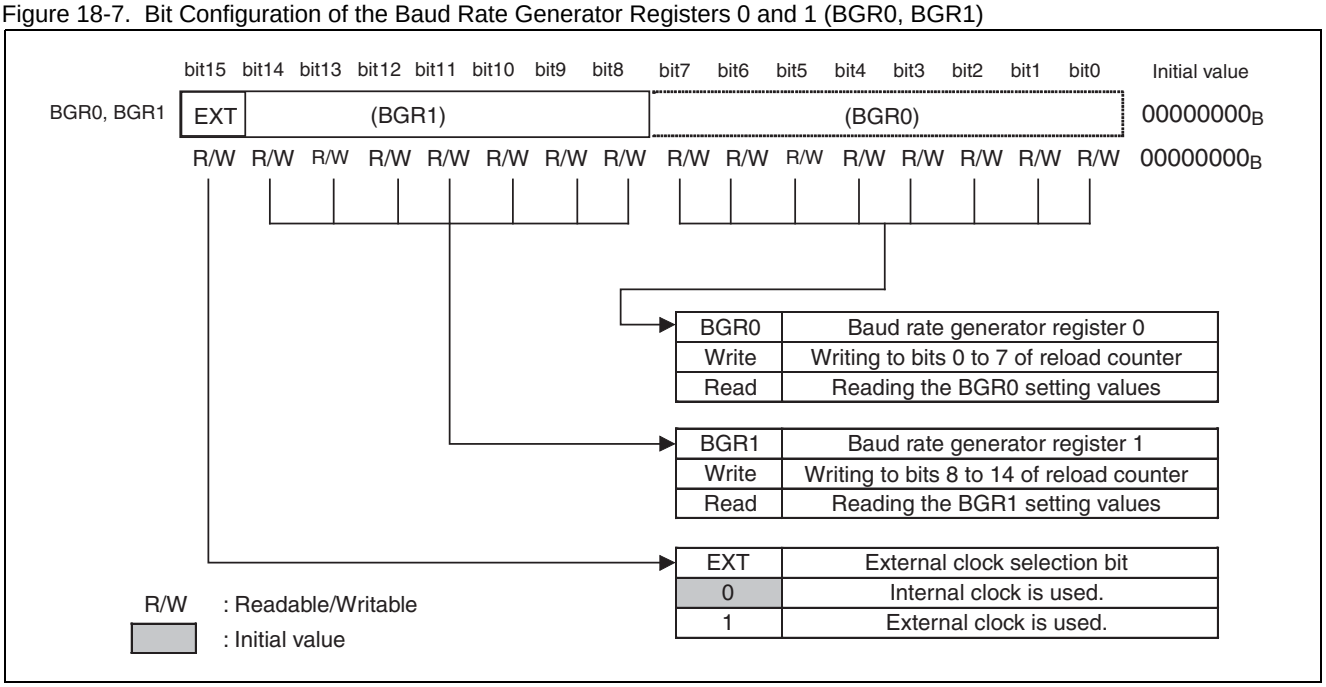
The transmission data register is a write only register, and the reception data register is a read only register. Since the transmission/reception registers are located at the same address, the write value and the read value are different. Therefore, the read-modify-write (RMW) instruction such as INC/DEC instructions cannot be used.

Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

The baud rate generator registers 0, 1 (BGR0, BGR1) set the division ratio for the serial clock. They can also select an external clock as a clock source for the reload counter.

Figure 18-7 shows the bit configuration of the baud rate generator registers 0 and 1 (BGR0, BGR1).

Figure 18-7. Bit Configuration of the Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)



- The baud rate generator registers set the division ratio of the serial clock.
- The BGR1 corresponds to the upper bits and BGR0 corresponds to the lower bits, and writing of reload values for counting and reading of the BGR0, BGR1 setting values are allowed.
- When writing reload values to the baud rate generator registers (BGR0, BGR1), the reload counter starts counting.
- Bit15 is the EXT bit and it selects an internal clock or an external clock used as a clock source for the reload counter. If EXT=0 is set, it selects an internal clock. If EXT=1 is set, it selects an external clock.

Notes:

- When writing into the baud rate generator registers (BGR0, BGR1), use 16-bit access.
- When a setting value in the baud rate generator registers (BGR0, BGR1) is changed, a new setting value is reloaded after the counter value becomes "0000_H". Therefore, if you want to enable a new setting value immediately, execute a programmable clear (UPCL) after changing the BGR0, BGR1 setting value.
- If a reload value is an even number, the "L" width of the reception serial clock becomes longer than the "H" width by one machine clock cycle. If it is an odd number, the "H" width and "L" width of the serial clock become the same.
- Set BGR0, BGR1 to 4 or larger value. Data, however, may not be received properly depending on the baud rate error and the reload value setting.
- To switch to the external clock setting (EXT=1) while the baud rate generator is operating, first write 0 into the baud rate generator register 0 and 1 (BGR0, BGR1), execute a programmable clear (UPCL), and then set the external clock (EXT=1).

18.4.2 Interrupts of the UART

The UART uses both transmission and reception interrupts. An interrupt request can be generated by one of the following sources:

- When reception data is set into the reception data register (RDR), or when a reception error occurs
- When transmission data is transferred from the transmission data register (TDR) to the shift register for transmission and transmission starts
- Transmission bus idle (no transmission operation)

Interrupts of the UART

The interrupt control bits and the interrupt sources of the UART are shown in Table 18-12.

Table 18-12. Interrupt Control Bits and Interrupt Sources of the UART

Interrupt type	Interrupt request flag bit	Flag register	Operation mode		Interrupt source	Interrupt source enable bit	How to clear the interrupt request flag
			0	1			
Reception	RDRF	SSR	❍	❍	1-byte reception	SCR:RIE	Read the reception data (RDR).
	ORE	SSR	❍	❍	Overrun error		Write “1” into the reception error flag clear bit (SSR:REC).
	FRE	SSR	❍	❍	Framing error		
	PE	SSR	❍	×	Parity error		
Trans-mission	TDRE	SSR	❍	❍	Transmission register empty	SCR:TIE	Write into the transmission data (TDR).
	TBI	SSR	❍	❍	No transmission operation	SCR:TBIE	Write into the transmission data (TDR).

Note:

Wait until TDRE bit becomes “0” before setting TIE bit to “1”.

Reception Interrupt Generation and the Flag Set Timing

Interrupts during reception are caused by completion of reception (SSR:RDRF) and occurrence of a reception error (SSR:PE, ORE, FRE).

Reception data is stored in the reception data register (RDR) when the first stop bit is detected. When the reception is completed (SSR:RDRF=1) or when a reception error occurs (SSR:PE, ORE, FRE=1), each flag is set. If reception interrupt is being enabled (SSR:RIE=1) at that time, then a reception interrupt is generated.

Note:

If any reception error occurs, data in the reception data register (RDR) becomes invalid.

Figure 18-8. RDRF (Reception Data Full) Flag Bit Set Timing

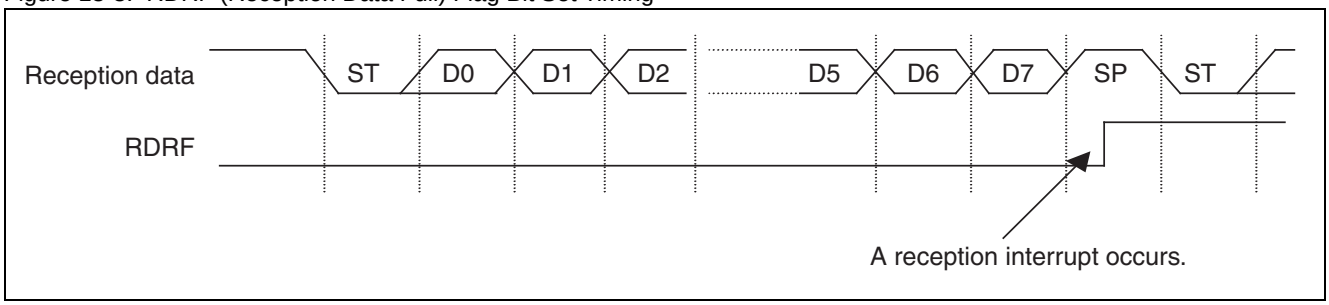
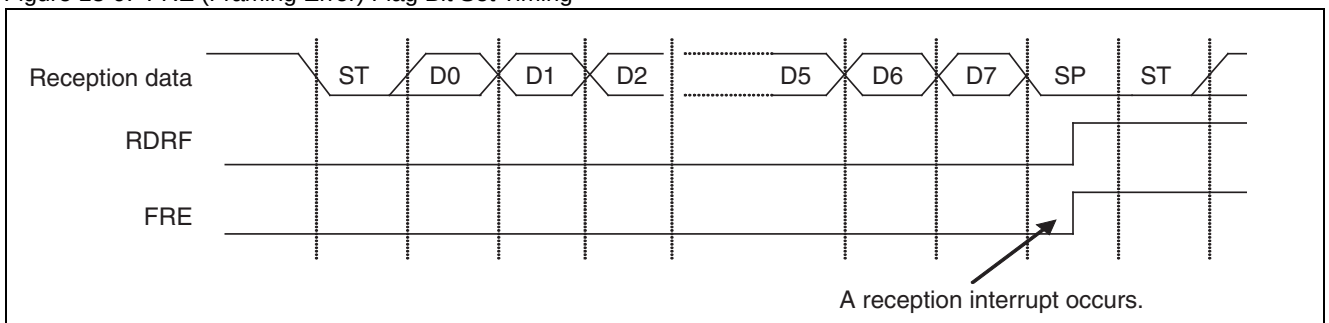
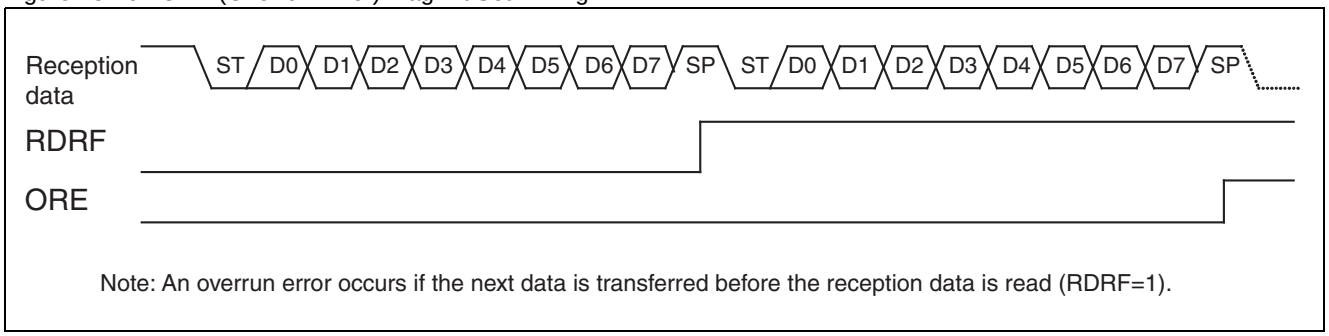


Figure 18-9. FRE (Framing Error) Flag Bit Set Timing



Notes: • When the first stop bit is at "L" level, a framing error occurs.
• Although "1" is set in RDRF and data is received even when a framing error occurs, the reception data is invalid.

Figure 18-10. ORE (Overrun Error) Flag Bit Set Timing



Note: An overrun error occurs if the next data is transferred before the reception data is read (RDRF=1).

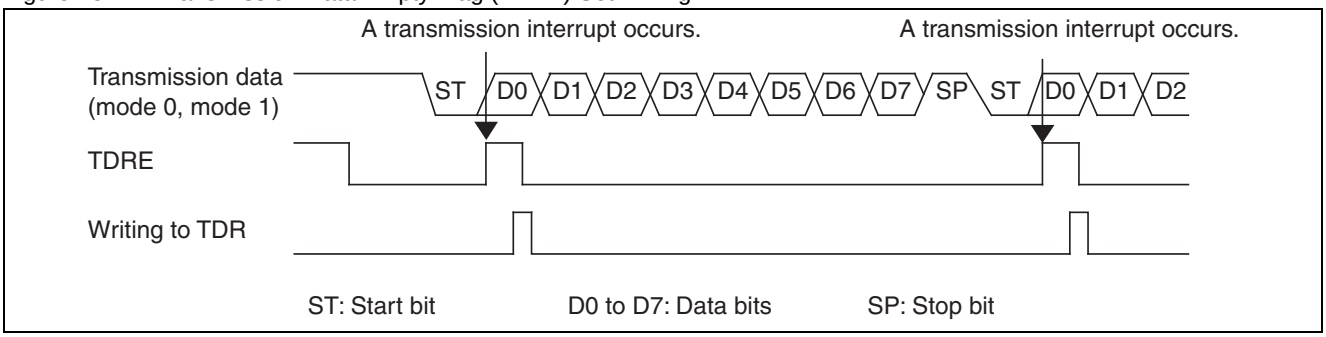
Transmission Interrupt Generation and the Flag Set Timing

Interrupts during transmission are caused when transmission data is transferred from the transmission data register (TDR) to the shift register for transmission (SSR:TDRE=1) and transmission starts, and when no transmission operation is running (SSR:TBI=1).

Transmission data empty flag (TDRE) set timing

When data written to the transmission data register (TDR) is transferred to the transmission shift register, the state becomes ready for the next data to be written (SSR:TDRE=1). If transmission interrupt is being enabled (SCR:TIE=1) at that time, then a transmission interrupt is generated. Since the TDRE bit is a read only bit, writing data into the transmission data register (TDR) clears the bit to “0”.

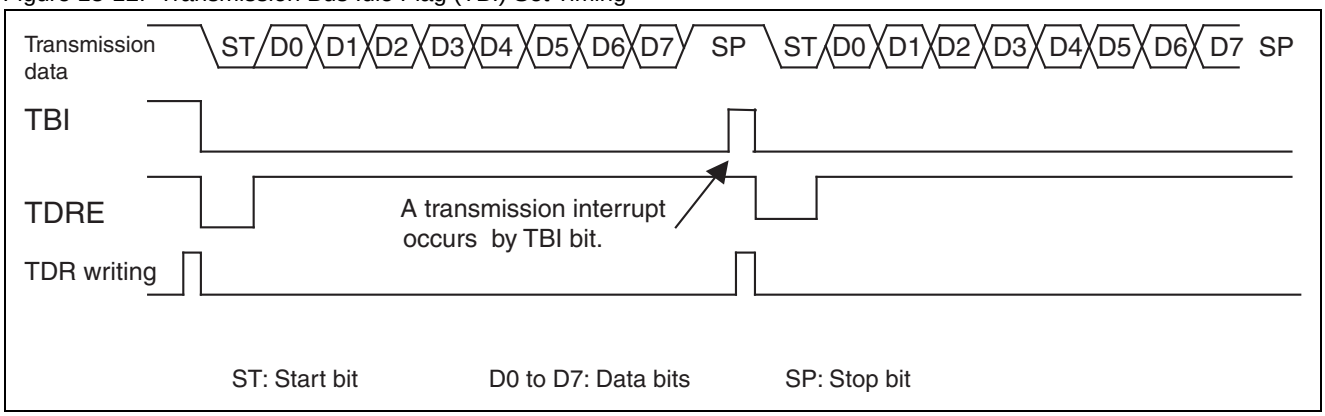
Figure 18-11. Transmission Data Empty Flag (TDRE) Set Timing



Transmission bus idle flag (TBI) set timing

When the transmission data register is empty (TDRE=1) and no transmission operation is running, SSR:TBI bit is set to “1”. If transmission bus idle interrupt is being enabled (SCR:TBIE=1) at that time, then a transmission interrupt is generated. When transmission data is set in the transmission data register (TDR), the TBI bit and the transmission interrupt request are cleared.

Figure 18-12. Transmission Bus Idle Flag (TBI) Set Timing



18.4.3 Operation of the UART

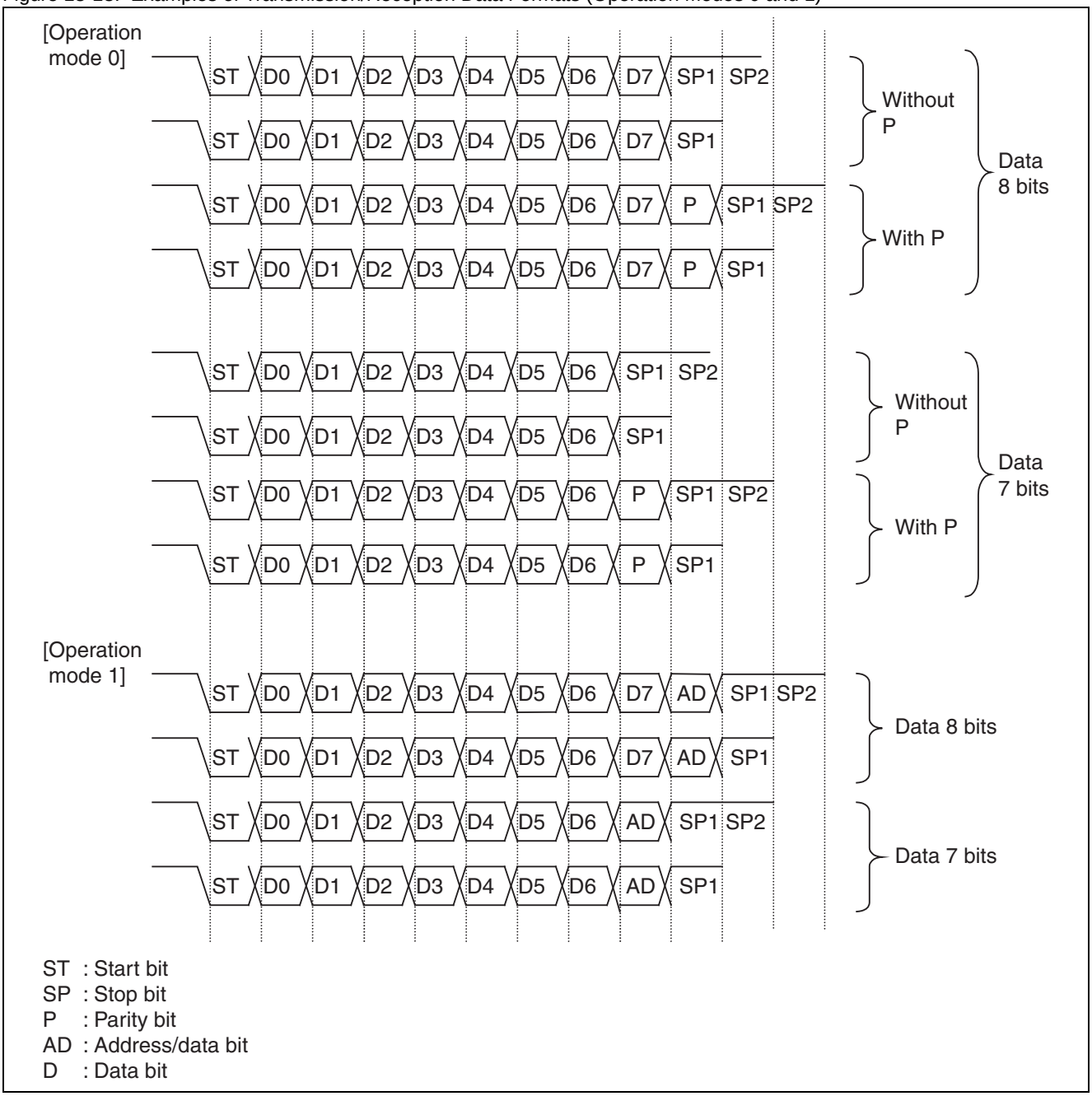
The UART operates with bidirectional serial asynchronous communication in mode 0 and with master/slave multiprocessor communication in mode 1.

Transmission/Reception Data Formats

- Transmission/reception data always starts with a start bit, and after a transmission/reception operation of a specified data bit length is performed, it ends with a stop bit consisting of at least one bit.
- Data transfer direction (LSB first or MSB first) is determined by BDS bit in the serial mode register (SMR). When using parities, a parity bit is always set between the last data bit and the first stop bit.
- In operation mode 0 (normal mode), data with/without a parity can be selected.
- In operation mode 1 (multiprocessor mode), an AD bit is added instead of a parity.

Figure 18-13 shows transmission/reception data formats in operation modes 0 and 1.

Figure 18-13. Examples of Transmission/Reception Data Formats (Operation Modes 0 and 1)



Notes:

- Figure 18-13 shows formats with the data length set to 7 bits and 8 bits. (The data length can be set to 5 bits to 9 bits in operation mode 0.)
- If the BDS bit of the serial mode register (SMR) is set to “1” (MSB first), bits are processed in the order from D7, D6, D5 ... D1, D0 (P).
- If the data length is set to X bits length, the lower X bits in the reception/transmission data register (RDR/TDR) become valid.

Transmission Operation

- When the transmission data empty flag bit (TDRE) of the serial status register (SSR) is “1”, transmission data can be written to the transmission data register (TDR).
- When transmission data is written to the transmission data register (TDR), the transmission data empty flag bit (TDRE) becomes “0”.
- When the transmission operation enable bit of the serial control register (SCR:TXE) is set to “1”, transmission data is loaded into the transmission shift register and then transmission starts from a start bit in order.
- When transmission is started, the transmission data empty flag bit (TDRE) is reset to “1”. If transmission interrupt is being enabled (SCR:TIE=1) at that time, a transmission interrupt is generated. The next transmission data can be written to the transmission data register in the interrupt process.

Note:

Since the initial value of the transmission data empty flag bit (SSR:TDRE) is “1”, a transmission interrupt will occur immediately after enabling transmission interrupt (SCR:TIE).

Reception Operation

- When reception operation is enabled (SCR:RXE=1), a reception operation is performed.
- When a start bit is detected, one frame of data is received in accordance with the data format set in the extended communication control register (ESCR:PEN, P, L2, L1, L0) and the serial mode register (SMR:BDS).
- After one frame reception is completed, the reception data full flag bit (SSR:RDRF) is set to “1”. If reception interrupt is being enabled (SCR:RIE=1) at that time, a reception interrupt is generated.
- When reading reception data after one frame data reception is completed, and check the error flag status in the serial status register (SSR). If any reception error is detected, perform error processing.
- The reception data full flag bit (SSR:RDRF) is cleared to “0” after reception data is read.

Note:

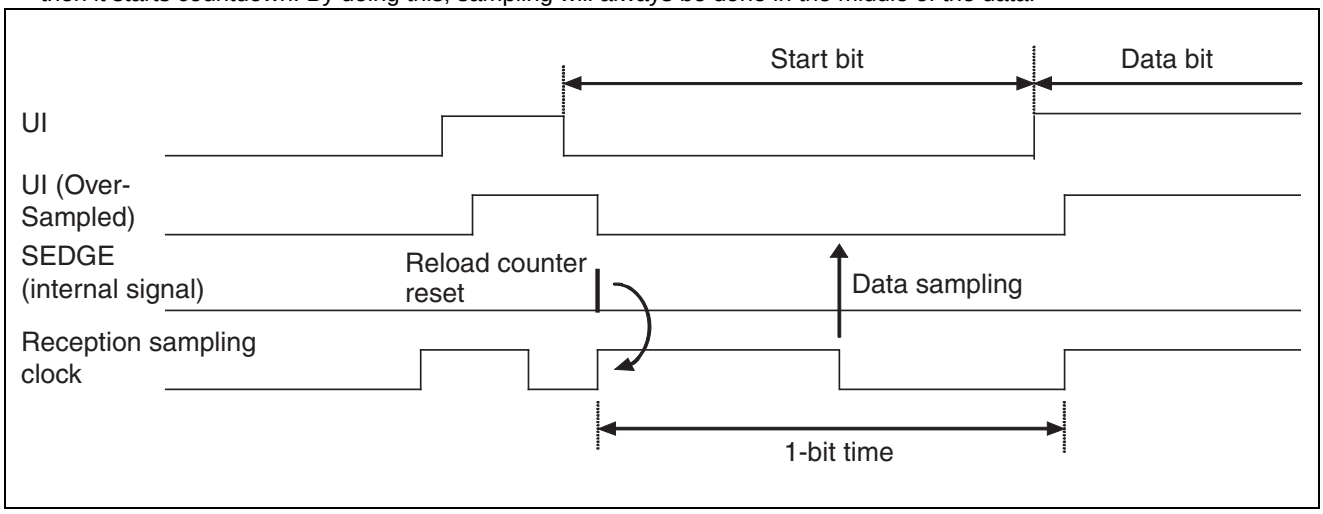
Data in the reception data register (RDR) becomes valid if the reception data register full flag bit (SSR:RDRF) is set to “1” and no reception error occurs (SSR:PE, ORE, FRE=0).

Clock Selection

- An internal clock or an external clock can be used.
- When using an external clock, set SMR:EXT=1. In this case, an external clock frequency is divided by the baud rate generator.

Start Bit Detection

- In asynchronous mode, a start bit is recognized at a UI signal falling edge.
- Therefore, a reception operation will not start until a UI signal falling edge is input even if reception operation is being enabled (SCR:RXE=1).
- When a start bit falling edge is detected, a reception reload counter in the baud rate generator is reset and reloaded, and then it starts countdown. By doing this, sampling will always be done in the middle of the data.



Stop Bit

- Either 1 bit or 2 bits length can be selected.
- The reception data full flag bit (SSR:RDRF) is set to “1” when the first stop bit is detected.

Error Detection

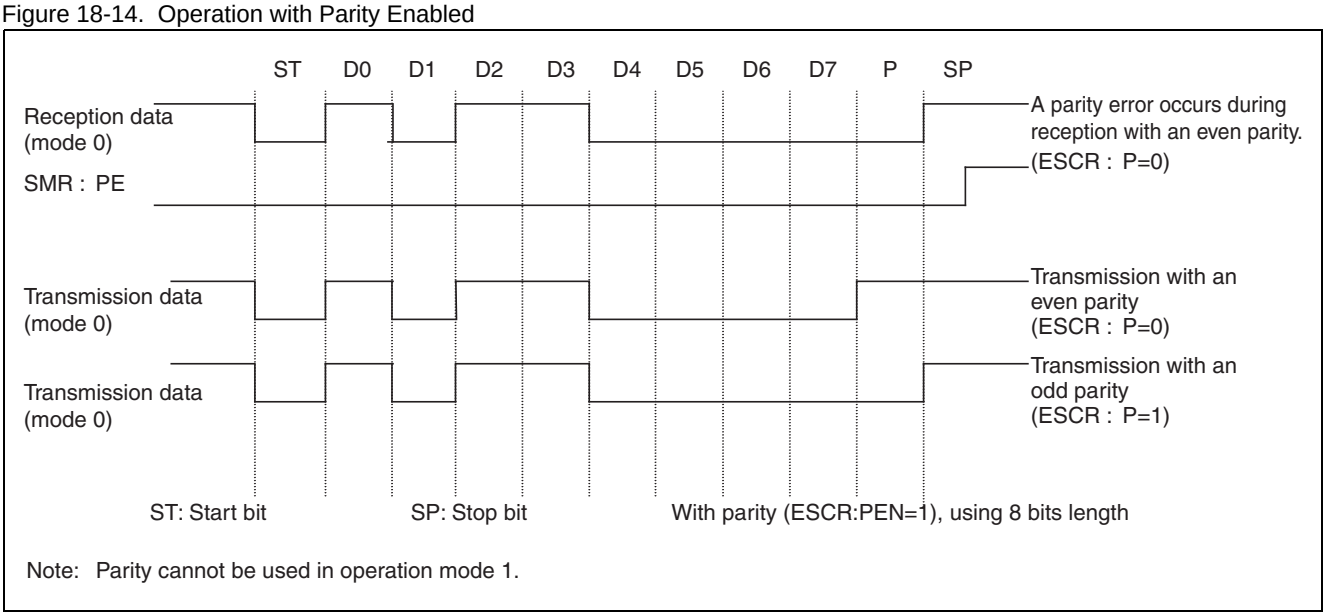
- In operation mode 0, a parity error, overrun error and frame error can be detected.
- In operation mode 1, an overrun error and frame error can be detected. A parity error cannot be detected.

Parity Bit

- Parity bit addition can be set only in operation mode 0. With/without parity can be set using the parity enable bit (ESCR:PEN) and even parity/odd parity can be set using the parity selection bit (ESCR:P).
- Parity cannot be used in operation mode 1.

Figure 18-14 shows a transmission/reception data when parity is enabled.

Figure 18-14. Operation with Parity Enabled



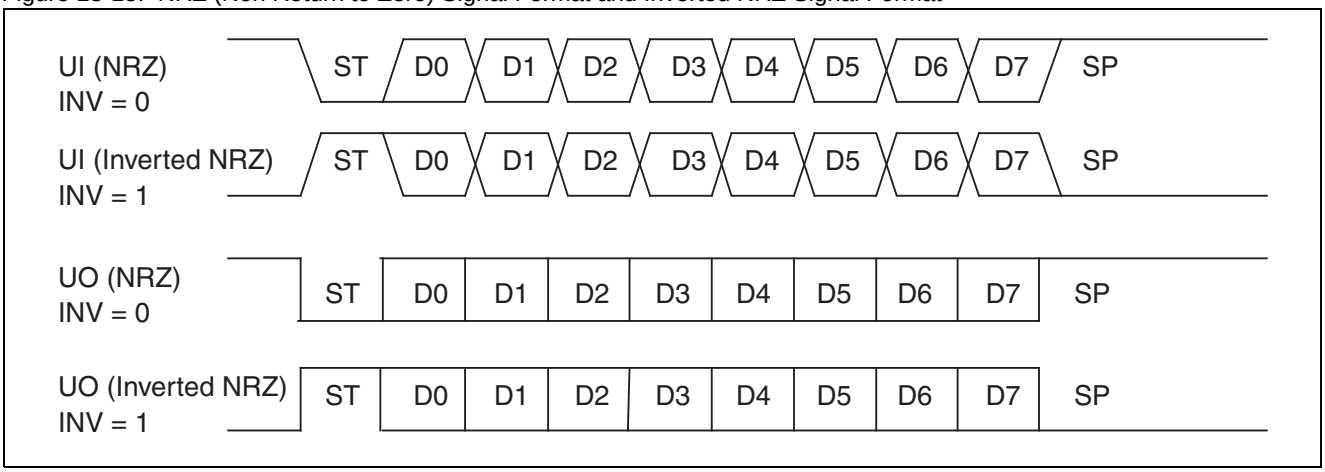
Note: Parity cannot be used in operation mode 1.

Data Signal Format

Either NRZ (Non Return to Zero) signal format (ESCR:INV=0) or inverted NRZ signal format (ESCR:INV=1) can be selected by setting the INV bit of the extended communication control register.

Figure 18-15 shows NRZ signal format and inverted NRZ signal format.

Figure 18-15. NRZ (Non Return to Zero) Signal Format and Inverted NRZ Signal Format



Data Transfer Format

LSB first or MSB first method can be selected for the data bit transfer format.

18.4.4 Dedicated Baud Rate Generator

One of the followings can be selected for the UART transmission/reception clock source:

- Dedicated baud rate generator (reload counter)
- External clock input to a baud rate generator (reload counter)

UART Baud Rate Selection

One of the following two types of baud rate can be selected:

Baud rate determined by dividing an internal clock using the dedicated baud rate generator (reload counter)

UART has two internal reload counters and each of them corresponds to transmission/reception serial clock. The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 1 and 0 (BGR0, BGR1).

The reload counter divides an internal clock by a setting value.

Select to use internal clock (SMR:EXT=0) for the clock source setting.

Baud rate determined by dividing an external clock using the dedicated baud rate generator (reload counter)

An external clock is used as a clock source for the reload counter.

The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 1 and 0 (BGR0, BGR1).

The reload counter divides an external clock by a setting value.

Select to use external clock and baud rate generator clock (SMR:EXT=1) for the clock source setting.

This mode is designed to be used in the case where an oscillator of special frequency is used by dividing it.

Notes:

- Set the external clock (EXT=1) while the reload counters are being stopped (BGR0, BGR1=0000_H).
- If external clock is set (EXT=1), two or more machine clock cycles are required for the “H” width and “L” width of the external clock.

Baud Rate Setting

The following describes the baud rate setting. It also shows a calculation result of a serial clock frequency.

Calculation of Baud Rate

Two 15-bit reload counters are set using the baud rate generator registers 1 and 0 (BGR0, BGR1).

Calculation formula of baud rate is shown below.

1. Reload value

$V = \phi/b-1$

V: Reload value b: Baud rate ϕ: Machine clock, external clock frequency

2. Example of the calculation

When the machine clock is set to 16MHz, an internal clock is used, and the baud rate is set to 19200 bps, the reload value can be calculated as follows:

Reload value:

$V = (16 \times 1000000)/19200 -1 = 832$

Therefore, the baud rate is:

$b = (16 \times 1000000)/(832 +1) = 19208$ bps

3. Baud rate error

The baud rate error can be calculated using the following formula:

Error (%) = (calculated value -desired value)/desired value × 100

(Example) When the machine clock is set to 20MHz and the desired baud rate is set to 153600 bps:

Reload value = $(20 \times 1000000)/153600 -1 = 129$

Baud rate (calculated value) = $(20 \times 1000000)/(129 + 1) = 153846$ (bps)

Error (%) = $(153846 -153600)/153600 \times 100 = 0.16$ (%)

Notes:

- The reload counter will be stopped when the reload value is set to “0”.
- If the reload value is an even number, the “L” width of the reception serial clock becomes longer than the “H” width by one machine clock cycle. If it is an odd number, the “H” width and “L” width of the serial clock become the same.
- Set the reload value to 4 or larger. Data, however, may not be received properly depending on the baud rate error and the reload value setting.

Reload Values and Baud Rates Corresponding to Each Machine Clock Frequency

Table 18-13. Reload Values and Baud Rates

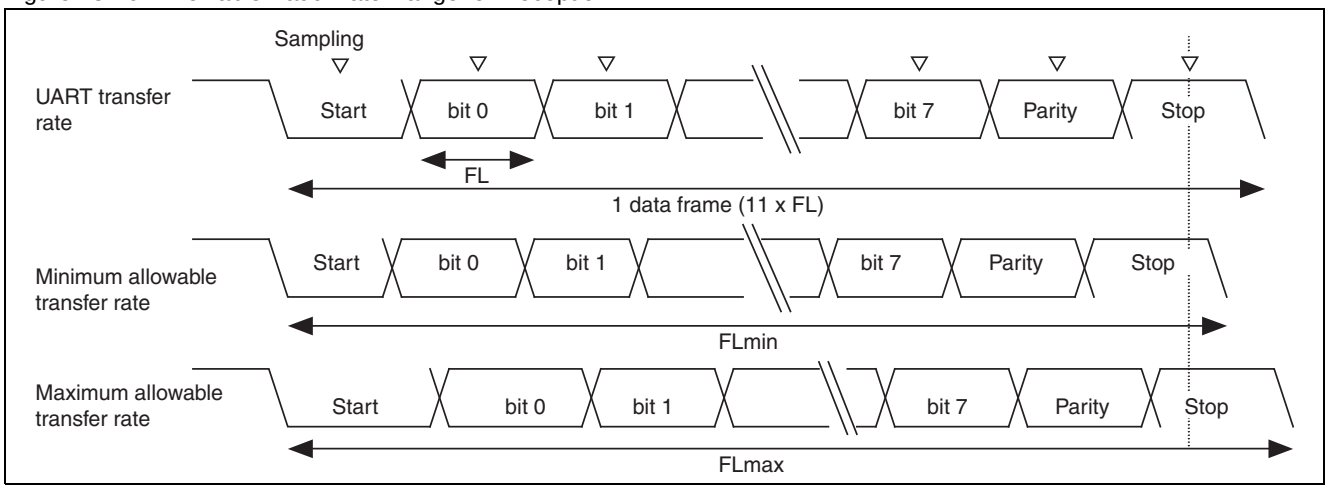
Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
4M	-	-	-	-	-	0	4	0	5	0	7	0
2.5M	-	-	-	0	-	-	-	-	-	-	-	-
2M	-	0	4	0	7	0	9	0	11	0	15	0
1M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	-0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	-	-	103	-0.16	-	-
153600	51	-0.16	64	-0.16	103	-0.16	129	-0.16	155	-0.16	207	-0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	68	-0.64	86	0.22	138	0.08	173	0.22	207	-0.16	277	0.08
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	311	-0.16	416	0.08
57600	138	0.08	173	0.22	277	0.08	346	-0.16	416	0.08	555	0.08
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	624	0	832	-0.04
28800	277	0.08	346	< 0.01	554	-0.01	693	-0.06	832	-0.03	1110	-0.01
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1249	0	1666	0.02
10417	767	< 0.01	959	< 0.01	1535	< 0.01	1919	< 0.01	2303	< 0.01	3071	< 0.01
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	2499	0	3332	-0.01
7200	1110	< 0.01	1388	< 0.01	2221	< 0.01	2777	< 0.01	3332	< 0.01	4443	-0.01
4800	1666	0.02	2082	-0.02	3332	< 0.01	4166	< 0.01	4999	0	6666	< 0.01
2400	3332	< 0.01	4166	< 0.01	6666	< 0.01	8332	< 0.01	9999	0	13332	< -0.01
1200	6666	< 0.01	8334	0.02	13332	< 0.01	16666	< 0.01	19999	0	26666	< 0.01
600	13332	< 0.01	16666	< 0.01	26666	< 0.01	-	-	-	-	-	-
300	26666	26666	< 0.01	-	-	-	-	-	-	-	-	-

- Value : Setting value in BGR0, BGR1 register (decimal)
- ERR : Baud rate error (%)

Allowable Baud Rate Range for Reception

The allowable baud rate error range at the transmission destination during reception is shown below. The baud rate error during reception must be set to within the allowable error range using the calculation formula shown below.

Figure 18-16. Allowable Baud Rate Range for Reception



As shown in Figure 18-16, after a start bit is detected, the sampling timing of reception data is determined by a counter set in BGR0, BGR1 register. If the last data (stop bit) is reached by this sampling timing, the data can be received properly.

If this is applied to an 11-bit reception logically, the rates can be calculated as follows:

If a sampling timing margin is set to two machine clocks (ϕ), the minimum allowable transfer rate (FLmin) can be calculated as follows:

$FLmin = (11bits \times (V + 1) - (V + 1)/2 + 3)/\phi = (21V + 27)/2\phi$ (s)

V: Reload value ϕ: Machine clock

Therefore, the receivable maximum baud rate (BGmax) at the transmission destination can be calculated as follows:

$BGmax = 11/FLmin = 22\phi/(21V + 27)$ (bps) V: Reload value ϕ: Machine clock

In a similar way, the maximum allowable transfer rate (FLmax) can be calculated as follows:

$FLmax = (11bits \times (V + 1) + (V + 1)/2 - 3)/\phi = (23V + 17)/2\phi$ (s)

V: Reload value ϕ: Machine clock

Thus, the receivable minimum baud rate (BGmin) at the transmission destination can be calculated as follows:

$BGmin = 11/FLmax = 22\phi/(23V + 17)$ (bps)

V: Reload value ϕ: Machine clock

The following allowable baud rate errors between UART and the transmission destination are calculated using the above calculation formulas for minimum/maximum baud rate values.

Reload value (V)	Maximum allowable baud rate error	Minimum allowable baud rate error
3	0%	0
10	+2.98%	-2.81%
50	+4.37%	-4.02%
100	+4.56%	-4.18%
200	+4.66%	-4.26%
32767	+4.76%	-4.35%

Note:

The reception accuracy depends on the number of bit in one frame, machine clock, and the reload value. The higher the machine clock and the division ratio, the higher the reception accuracy will be.

External Clock

If “1” is written to the EXT bit of the baud rate generator register (BGR), an external clock is divided by the baud rate generator.

Note:

An external clock signal synchronizes with an internal clock in UART. Therefore, if an external clock that cannot synchronize is used, the operation will become unstable.

Function of Reload Counter

Two types of reload counter, a transmission reload counter and a reception reload counter, are available and each of them functions as a dedicated baud rate generator. It consists of a 15-bit register corresponding to reload values and it generates transmission/reception clock from an external clock or an internal clock.

Count Start

The reload counter starts counting when a reload value is written into the baud rate generator registers (BGR0, BGR1).

Restart

The reload counter restarts under the following conditions:

- For both transmission/reception reload counters
  Programmable reset (SCR:UPCL bit)
- For reception reload counter
  Detection of a start bit falling edge in asynchronous mode

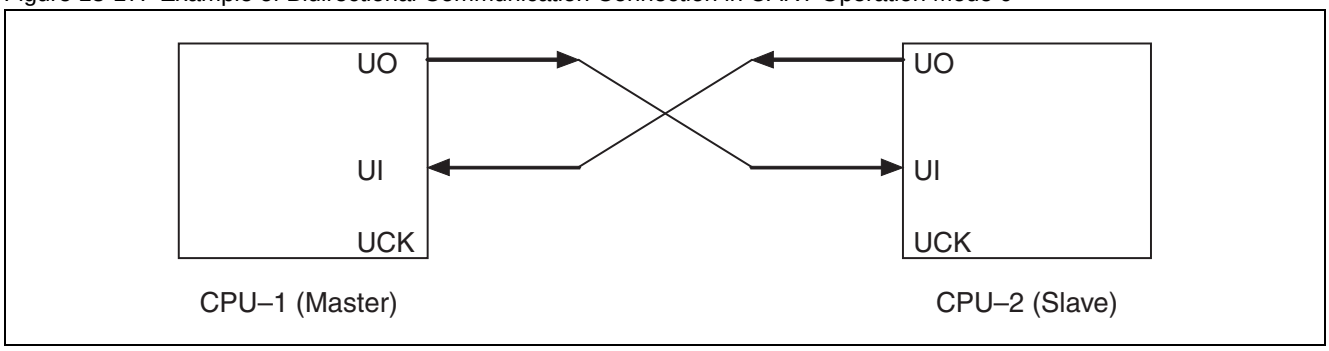
18.4.5 Setting Procedure and the Program Flow for Operation Mode 0 (Asynchronous Normal Mode)

Operation mode 0 allows asynchronous serial bidirectional communication.

Connection Between CPUs

In operation mode 0 (normal mode), select bidirectional communication. Connect two CPUs to each other as shown in Figure 18-17.

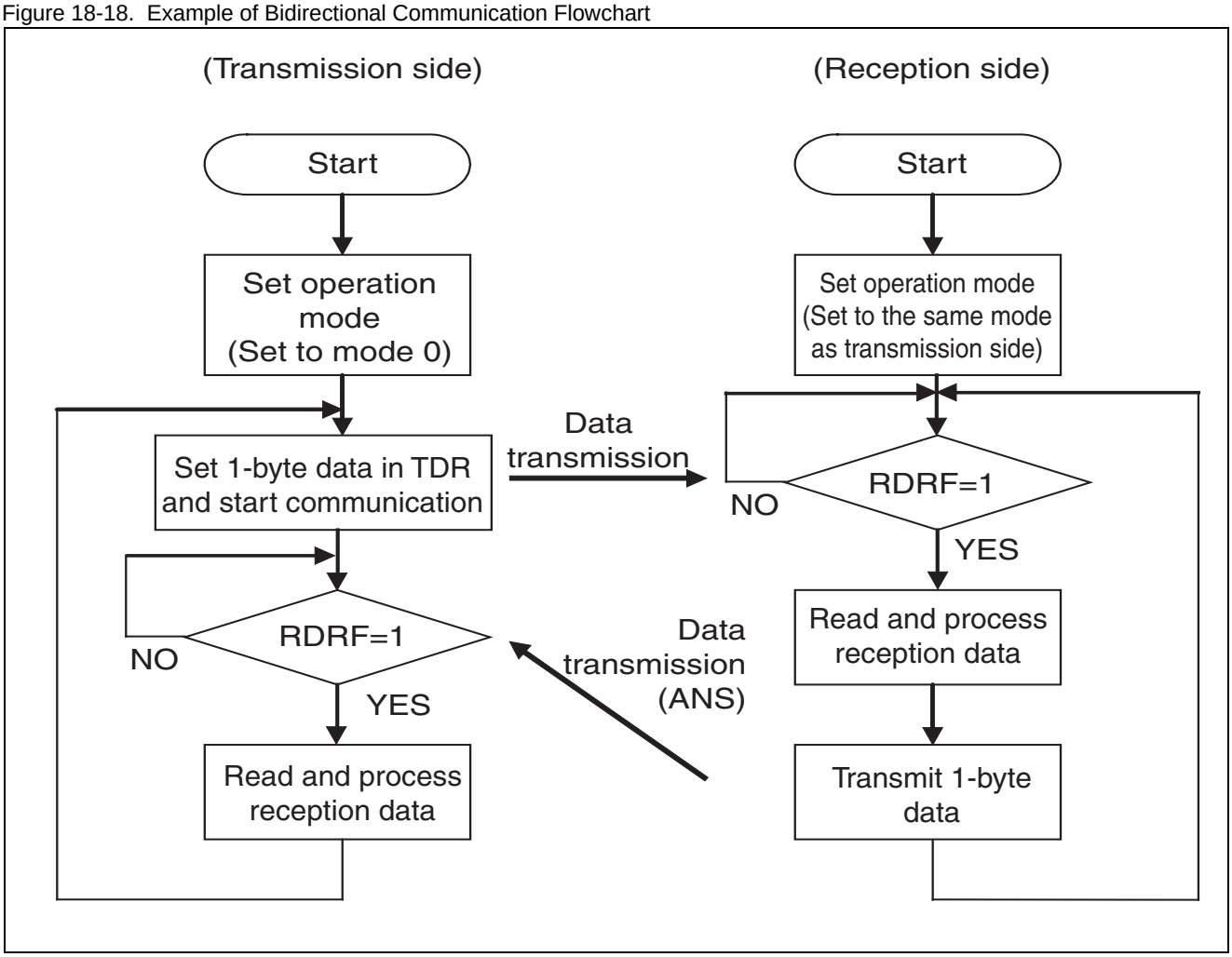
Figure 18-17. Example of Bidirectional Communication Connection in UART Operation Mode 0



Flowchart

Bidirectional communication flowchart

Figure 18-18. Example of Bidirectional Communication Flowchart



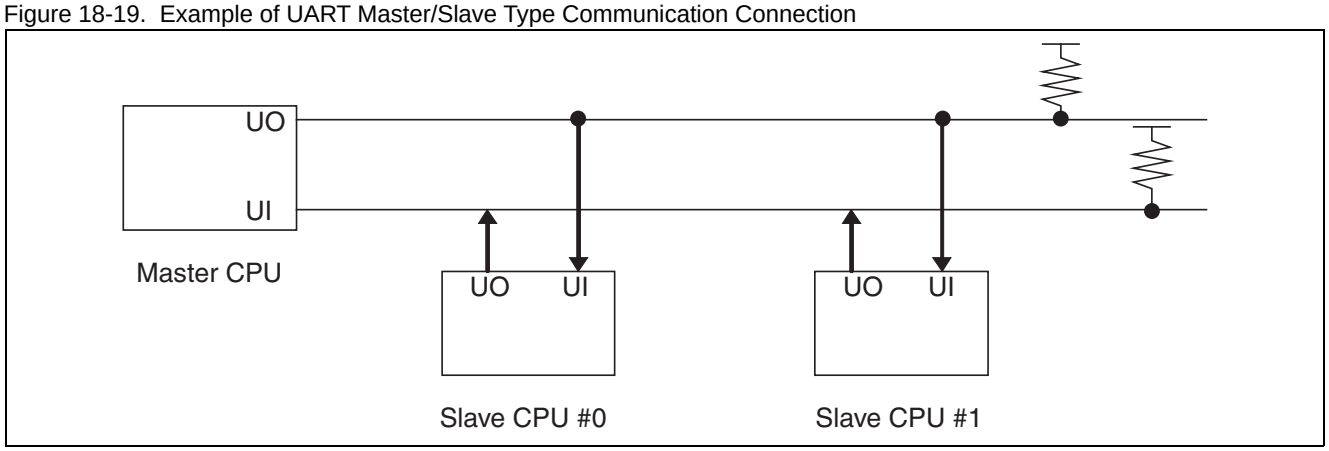
18.4.6 Setting Procedure and the Program Flow for Operation Mode 1 (Asynchronous Multiprocessor Mode)

Operation mode 1(multiprocessor mode) allows communication between multiple CPUs using master/slave connection. It can be used as master/slave.

Connection Between CPUs

In master/slave type communication, a communication system is configured by connecting one master CPU and multiple slave CPUs with two common communication lines as shown in the Figure. The UART can be used in master or slave.

Figure 18-19. Example of UART Master/Slave Type Communication Connection



Function Selection

For master/slave type communication, select the operation mode and data transfer format as shown in Table 18-14.

Table 18-14. Function Selection for Master/Slave Type Communication

	Operation mode		Data	Parity	Stop bit	Bit direction
	Master CPU	Slave CPU				
Address transmission/ reception	Mode 1 (AD bit transmission)	Mode 1 (AD bit reception)	AD = 1 + 7-bit or 8-bit address	None	1 bit or 2 bits	LSB first or MSB first
Data transmission/reception			AD = 0 + 7-bit or 8-bit data			

Note:

Use word access for transmission/reception data (TDR/RDR) in operation mode 1.

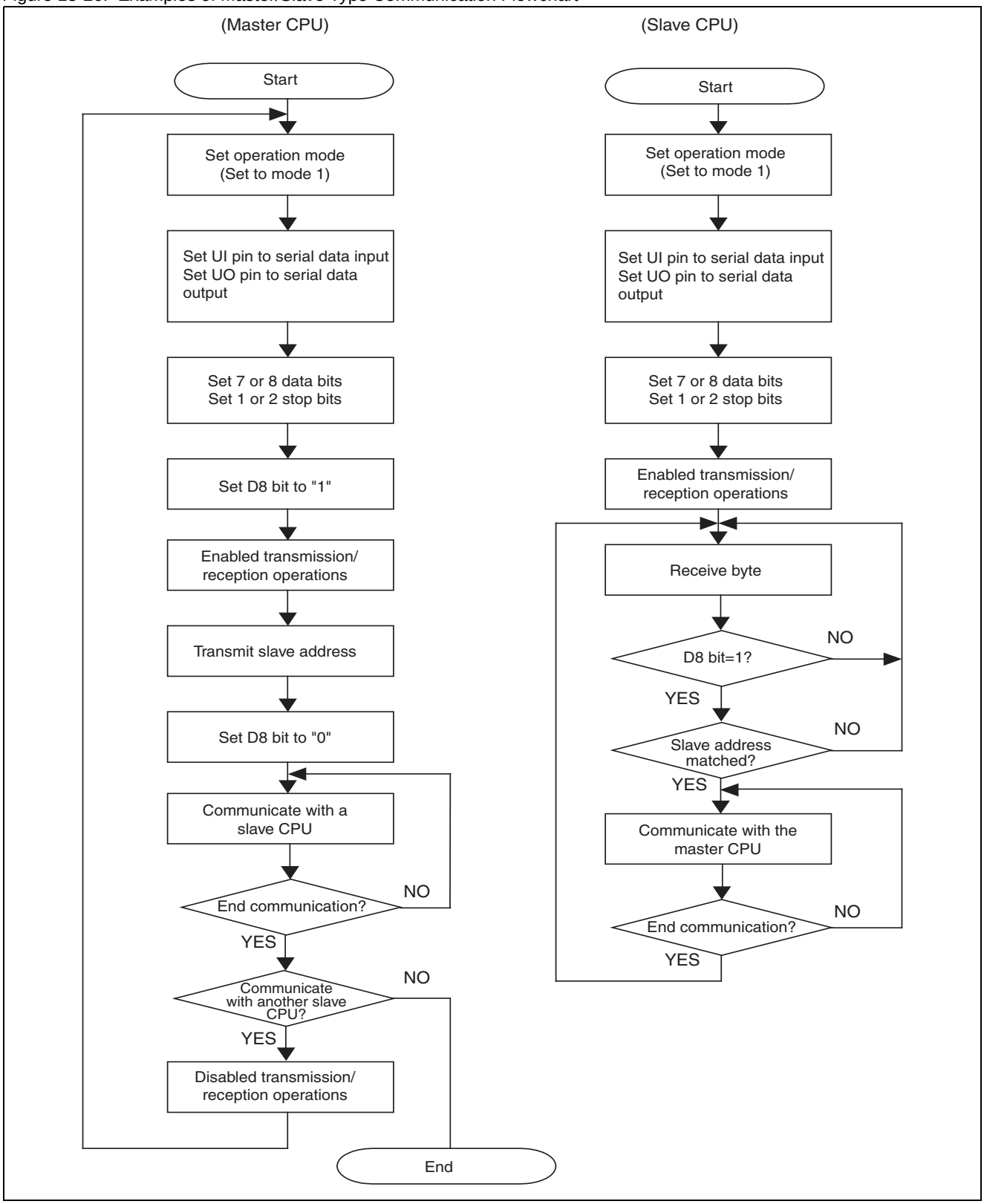
Communication procedure

Communication is initiated by transmitting address data by the master CPU. The address data is the data having "1" in D8 bit and it selects a slave CPU to have communication. Each slave CPU determines the address data using program and if the address is matched with the assigned address, the slave CPU starts the communication (normal data) with the master CPU.

Figure 18-20 shows examples of master/slave type communication flowchart.

Flowchart

Figure 18-20. Examples of Master/Slave Type Communication Flowchart



18.5 CSIO (Clock Synchronous Serial Interface)

This section describes the CSIO functions supported by operation mode 2 among multi-function serial interface functions.

CSIO (Clock Synchronous Serial Interface)

Overview of the CSIO (Clock Synchronous Serial Interface)

Registers of the CSIO (Clock Synchronous Serial Interface)

- Serial Control Register (SCR)
- Serial Mode Register (SMR)
- Serial Status Register (SSR)
- Extended Communication Control Register (ESCR)
- Reception Data Register/Transmission Data Register (RDR/TDR)
- Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

Interrupts of the CSIO (Clock Synchronous Serial Interface)

- Reception Interrupt Generation and the Flag Set Timing
- Transmission data empty flag (TDRE) set timing
- Operation of the CSIO (Clock Synchronous Serial Interface)

Dedicated Baud Rate Generator

- Baud Rate Setting
- Setting Procedure and the Program Flow for CSIO (Clock Synchronous Serial Interface)

18.5.1 Overview of the CSIO (Clock Synchronous Serial Interface)

The CSIO (clock synchronous serial interface) is a general-purpose serial data communication interface to perform synchronous communication with external devices. (SPI compliant)

Functions of the CSIO (Clock Synchronous Serial Interface)

		Functions
1	Data buffer	■ Full-duplex double buffer ■ Transmission/reception
2	Transfer format	■ Clock synchronous (without start bit/stop bit) ■ Master/slave function ■ SPI compliant (for both master/slave supported)
3	Baud rate	■ Dedicated baud rate generator (consisting of a 15-bit reload counter, in master operation) ■ External clock input enabled (in slave operation)
4	Data length	5 bits to 9 bits variable
5	Reception error detection	Overrun error
6	Interrupt requests	■ Reception interrupts (Reception completion, overrun error) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Both transmission and reception have extended intelligent I/O service (EI^{2}OS) and DMA transfer support functions.
7	Synchronous mode	Master or slave function
8	Pin access	A serial data output pin can be set to “1”.

18.5.2 Registers of the CSIO (Clock Synchronous Serial Interface)

A register list of CSIO (clock synchronous serial interface) is shown below.

Register List of the CSIO (Clock Synchronous Serial Interface)

Table 18-15. Register List of the CSIO (Clock Synchronous Serial Interface)

	Address		bit15 bit8	bit7 bit0
CSIO	ch.0:000021$_H$ ch.1:00002B$_H$ ch.2:00003F$_H$ ch.3:000049$_H$ ch.4:000053$_H$ ch.5:00005D$_H$ ch.6:007791$_H$	ch.0:000020$_H$ ch.1:00002A$_H$ ch.2:00003E$_H$ ch.3:000048$_H$ ch.4:000052$_H$ ch.5:00005C$_H$ ch.6:007990$_H$	SCR (Serial control register)	SMR (Serial mode register)
	ch.0:000023$_H$ ch.1:00002D$_H$ ch.2:000041$_H$ ch.3:00004B$_H$ ch.4:000055$_H$ ch.5:00005F$_H$ ch.6:007993$_H$	ch.0:000022$_H$ ch.1:00002C$_H$ ch.2:000040$_H$ ch.3:00004A$_H$ ch.4:000054$_H$ ch.5:00005E$_H$ ch.6:007992$_H$	SSR (Serial status register)	ESCR (Extended communication control register)
	ch.0:000025$_H$ ch.1:00002F$_H$ ch.2:000043$_H$ ch.3:00004D$_H$ ch.4:000057$_H$ ch.5:000061$_H$ ch.6:007995$_H$	ch.0:000024$_H$ ch.2:00002E$_H$ ch.3:000042$_H$ ch.4:00004C$_H$ ch.5:000056$_H$ ch.6:000060$_H$ ch.7:007994$_H$	RDR1/TDR1 (Reception/transmission data register 1)	RDR0/TDR0 (Reception/transmission data register 0)
	ch.0:000027$_H$ ch.1:000031$_H$ ch.2:000045$_H$ ch.3:00004F$_H$ ch.4:000059$_H$ ch.5:000063$_H$ ch.6:007997$_H$	ch.0:000026$_H$ ch.1:000030$_H$ ch.2:000044$_H$ ch.3:00004E$_H$ ch.4:000058$_H$ ch.5:000062$_H$ ch.6:007996$_H$	BGR1 (Baud rate generator register 1)	BGR0 (Baud rate generator register 0)
	ch.0:000029$_H$ ch.1:000033$_H$ ch.2:000047$_H$ ch.3:000051$_H$ ch.4:00005B$_H$ ch.5:000065$_H$ ch.6:007999$_H$	ch.0:000028$_H$ ch.1:000032$_H$ ch.2:000046$_H$ ch.3:000050$_H$ ch.4:00005A$_H$ ch.5:000064$_H$ ch.6:007998$_H$	-	-

Table 18-16. Bit Assignment of the CSIO (Clock Synchronous Serial Interface)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	Reser-ved	SCINV	BDS	SCKE	SOE
SSR/ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	-	-	L2	L1	L0
TDR/RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0

Serial Control Register (SCR)

The serial control register (SCR) enables/disables transmission/reception interrupts, transmission idle interrupts, and transmission/reception operations. It can also perform setting for connecting to SPI and CSIO reset.

Figure 18-21 shows the bit configuration of the serial control register (SCR) and Table 18-17 shows the functions of each bit.

Figure 18-21. Bit Configuration of the Serial Control Register (SCR)

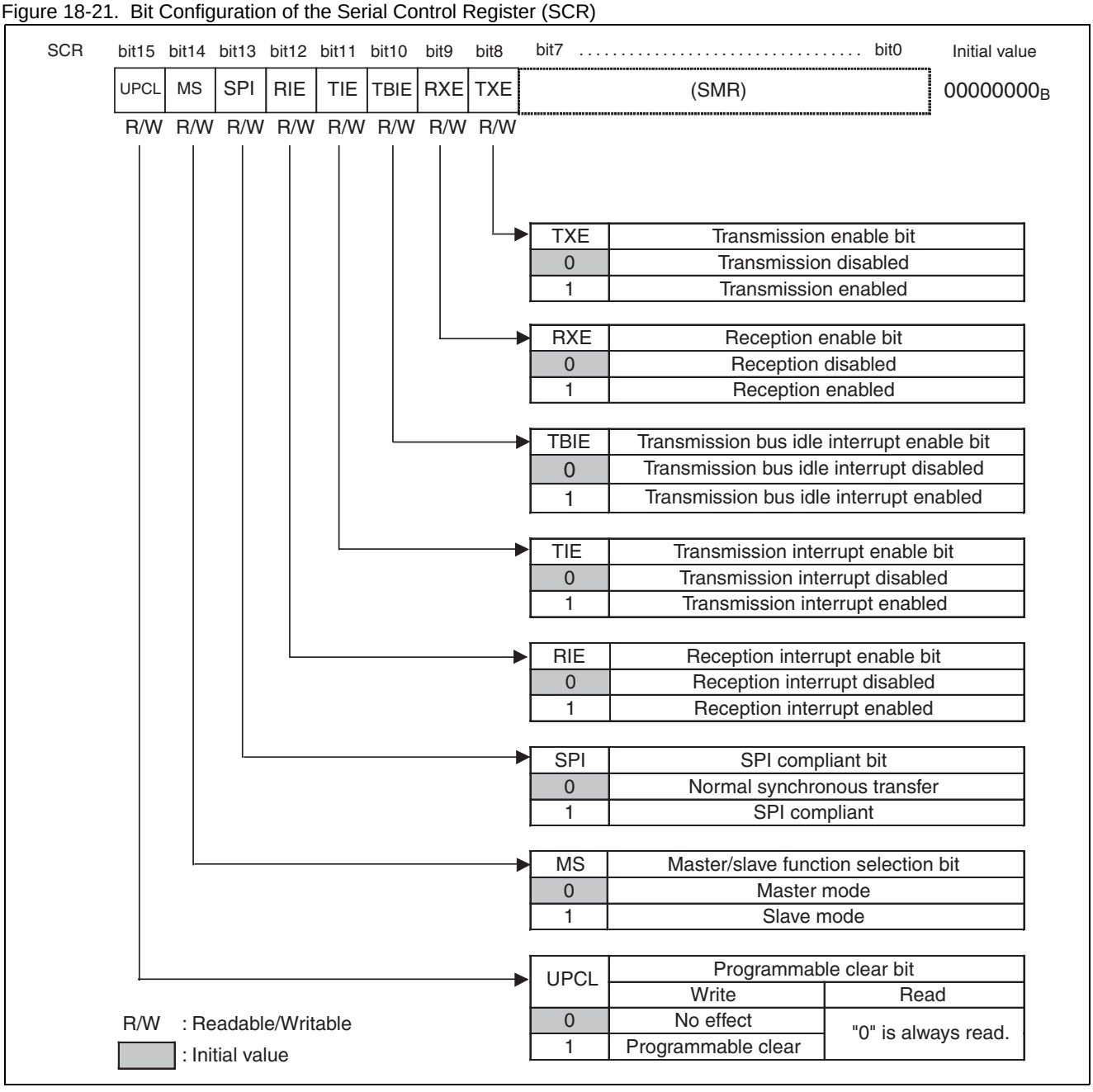


Table 18-17. Function Description of Each Bit of the Serial Control Register (SCR)

Bit name		Function
bit15	UPCL: Programmable clear bit	This bit initializes the internal state of the CSIO. When “1” is set: ■ CSIO is reset directly (software reset). However, the register settings are retained. If a register is in transmission/reception state at that time, it will be disconnected immediately. ■ The baud rate generator reloads the setting values in BGR0, BGR1 registers and restarts. ■ All transmission/reception interrupt sources (TDRE, TBI, RDRF and ORE) are initialized. When “0” is set: It has no effect on operations. “0” is always read when read. **Note:** After interrupt disabled is set, execute a programmable clear.
bit14	MS: Master/slave function selection bit	This bit selects master or slave mode. When “0” is set: Master mode is set. When “1” is set: Slave mode is set. **Note:** When slave mode is selected, an external clock is input directly if SMR:SCKE=0.
bit13	SPI: SPI compliant bit	This bit enables SPI compliant communication. When “0” is set: Normal synchronous communication is performed. When “1” is set: Communication complies with SPI.
bit12	RIE: Reception interrupt enable bit	■ This bit enables/disables the output of reception interrupt requests to the CPU. ■ If RIE bit and reception data flag bit (RDRF) are “1”, or if an error flag bit (ORE) is “1”, a reception interrupt request is output.
bit11	TIE: Transmission interrupt enable bit	■ This bit enables/disables the output of transmission interrupt requests to the CPU. ■ If TIE bit and TDRE bit are “1”, a transmission interrupt request is output.
bit10	TBIE: Transmission bus idle interrupt enable bit	■ This bit enables/disables the output of transmission bus idle interrupt requests to the CPU. ■ If TBIE bit and TBI bit are “1”, a transmission bus idle interrupt request is output.
bit9	RXE: Reception operation enable bit	This bit enables/disables the reception operation of the CSIO. ■ If “0” is set: Data frame reception operation is disabled. ■ If “1” is set: Data frame reception operation is enabled. **Note:** If the reception operation is disabled (RXE=0) during reception, the operation is stopped immediately.
bit8	TXE: Transmission operation enable bit	This bit enables/disables the transmission operation of the CSIO. ■ If “0” is set: Data frame transmission operation is disabled. ■ If “1” is set: Data frame transmission operation is enabled. **Note:** If the transmission operation is disabled (TXE=0) during transmission, the operation is stopped immediately.

Serial Mode Register (SMR)

The serial mode register (SMR) sets the operation mode, selects transfer direction, data length and serial clock reversal, and enables/disables the outputs to the serial data and serial clock pins.

Figure 18-22 shows the bit configuration of the serial mode register (SMR) and Table 18-18 shows the functions of each bit.

Figure 18-22. Bit Configuration of the Serial Mode Register (SMR)

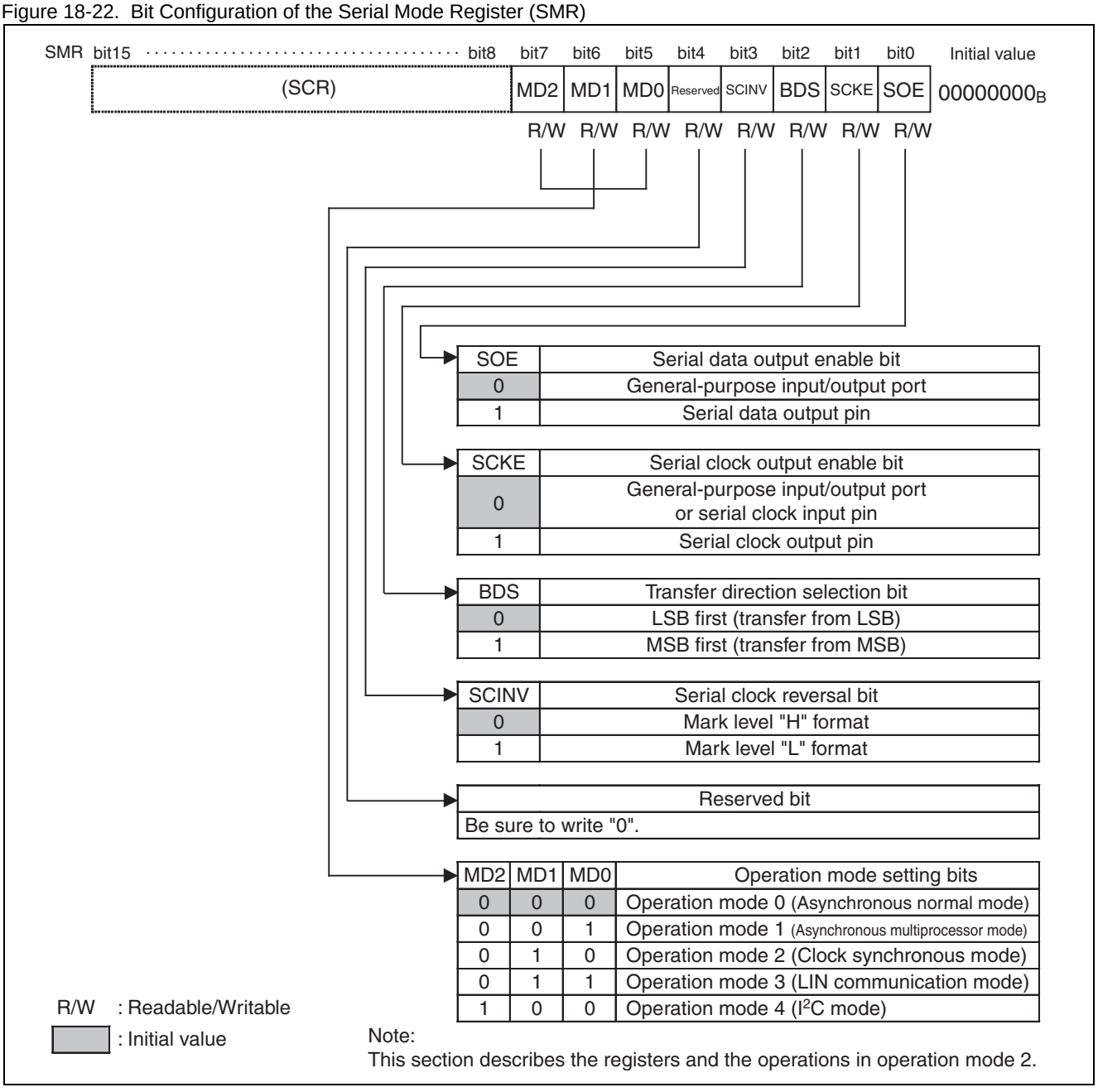


Table 18-18. Function Description of Each Bit of the Serial Mode Register (SMR)

Bit name		Function
bit7 to bit5	MD2 to MD0: Operation mode setting bits	These bits set the operation mode. "000_B": Operation mode 0 (asynchronous normal mode) is set. "001_B": Operation mode 1 (asynchronous multiprocessor mode) is set. "010_B": Operation mode 2 (clock synchronous mode) is set. "011_B": Operation mode 3 (LIN communication mode) is set. "100_B": Operation mode 4 (I^2C mode) is set. This section describes the registers and the operations in operation mode 2 (clock synchronous mode). **Notes:** ■ Settings other than the above are prohibited. ■ To switch the operation mode, do so after executing a programmable clear (SCR:UPCL=1). ■ Set each register after setting the operation mode.
bit4	Reserved bit	Be sure to write "0".
bit3	SCINV: Serial clock reversal bit	This bit reverses the serial clock format. **When "0" is set:** ■ Mark level of the serial clock output becomes "H". ■ Transmission data is output in synchronization with a serial clock falling edge in normal transfer, and with a serial clock rising edge in SPI transfer. ■ Reception data is sampled at a serial clock rising edge in normal transfer, and at a serial clock falling edge in SPI transfer. **When "1" is set:** ■ Mark level of the serial clock output becomes "L". ■ Transmission data is output in synchronization with a serial clock rising edge in normal transfer, and with a serial clock falling edge in SPI transfer. ■ Reception data is sampled at a serial clock falling edge in normal transfer, and at a serial clock rising edge in SPI transfer. **Note:** Set this bit when the transmission and reception operations are disabled (TXE=RXE=0).
bit2	BDS: Transfer direction selection bit	This bit selects whether to transfer the transfer serial data from the least significant bit side first (LSB first, BDS=0) or to transfer it from the most significant bit side first (MSB first, BDS=1). Note: Set this bit when the transmission and reception operations are disabled (TXE=RXE=0).
bit1	SCKE: Serial clock output enable bit	This bit controls input/output ports of the serial clock. When "0" is set: UCK pin becomes a general-purpose input/output port or a serial clock input pin. When "1" is set: UCK pin becomes a serial clock output pin and outputs the clock. **Note:** When the UCK pin is used as a serial clock input (SCKE=0), set a general-purpose input/output port to an input port. **Reference:** If the UCK pin is set to serial clock output (SCKE=1), it functions as a serial clock output pin regardless of the general-purpose input/output port (DDR) settings.
bit0	SOE: Serial data output enable bit	This bit enables/disables the output of the serial data. When "0" is set: UO pin becomes a general-purpose input/output port. When "1" is set: UO pin becomes a serial data output pin (UO). **Reference:** When this bit is set to serial data output (SOE=1), the UO pin functions as a UO pin regardless of the general-purpose input/output port (DDR) settings.

Serial Status Register (SSR)

The serial status register (SSR) checks transmission/reception statuses and reception error flags, and clears the flags.

Figure 18-23 shows the bit configuration of the serial status register (SSR) and Table 18-19 shows the functions of each bit.

Figure 18-23. Bit Configuration of the Serial Status Register (SSR)

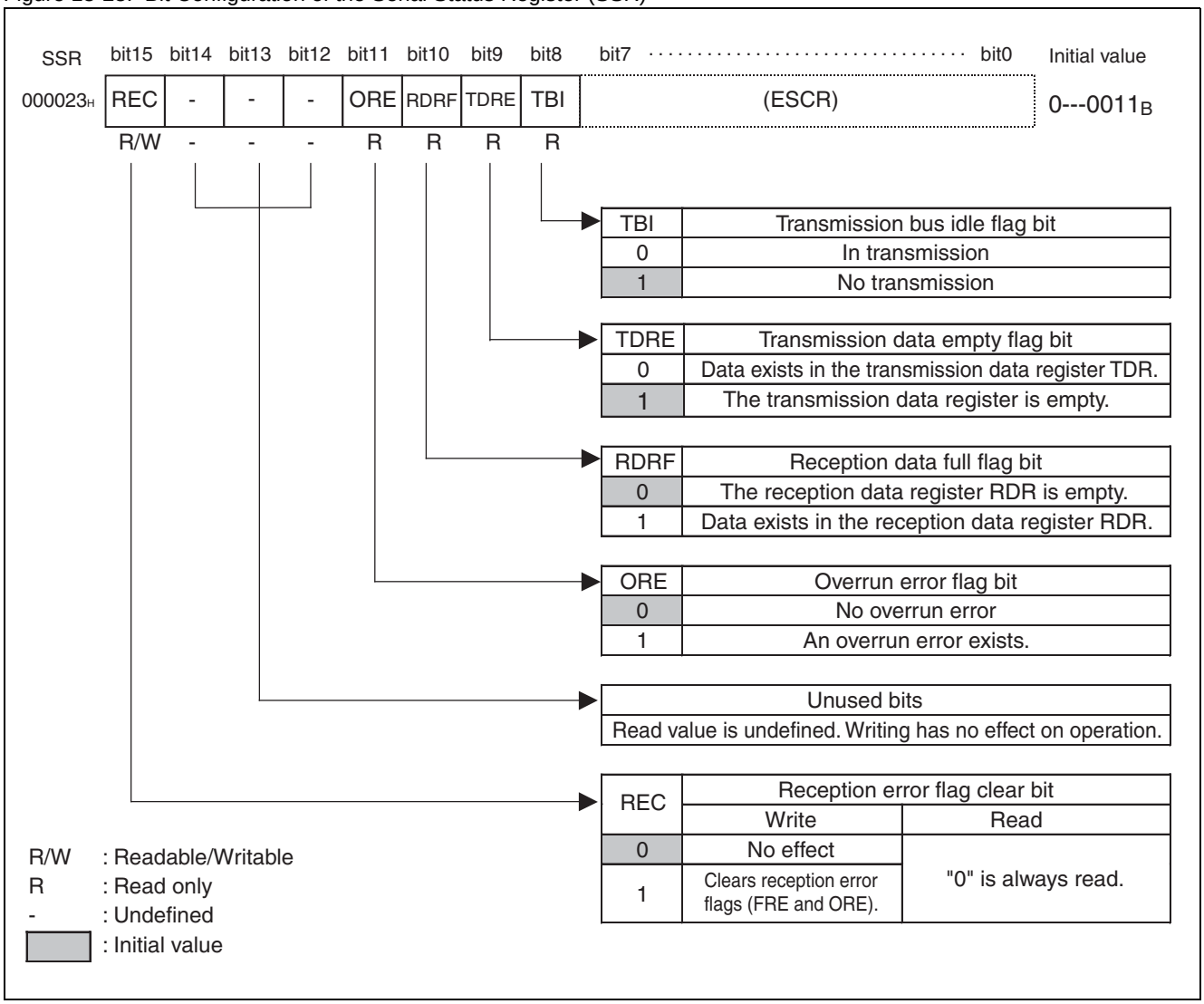


Table 18-19. Function Description of Each Bit of the Serial Status Register (SSR)

Bit name		Function
bit15	REC: Reception error flag clear bit	This bit clears ORE flag of the serial status register (SSR). ■ When “1” is written, the error flag is cleared. ■ When “0” is written, this writing has no effect. When read, “0” is always read.
bit14 to bit12	Unused bits	When read: The value is undefined. When write: No effect
bit11	ORE: Overrun error flag bit	■ This bit is set to “1” when an overrun error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the ORE bit and RIE bit are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit10	RDRF: Reception data full flag bit	■ This flag indicates the status of the reception data register (RDR). ■ This bit is set to “1” when reception data is loaded into the RDR and is cleared to “0” when the reception data register (RDR) is read. ■ A reception interrupt request is output when the RDRF bit and RIE bit are “1”.
bit9	TDRE: Transmission data empty flag bit	■ This flag indicates the status of the transmission data register (TDR). ■ This bit becomes “0” when transmission data is written to TDR, indicating that valid data exists in the TDR. This bit becomes “1” when the data is loaded into the transmission shift register and transmission starts, indicating that valid data does not exist in the TDR. ■ A transmission interrupt request is output when the TDRE bit and TIE bit are “1”. ■ The TDRE bit becomes “1” when the UPCL bit in the serial control register (SCR) is set to “1”.
bit8	TBI: Transmission bus idle flag bit	■ This bit indicates that the CSIO is not in transmission operations. ■ This bit becomes “0” when data is written to the transmission data register (TDR). ■ This bit becomes “1” when the transmission data register (TDR) is empty (TDRE=1) and no transmission operation is performed. ■ The TDRE bit becomes “1” when the UPCL bit of the serial control register (SCR) is set to “1”. ■ A transmission interrupt request is output when this bit is "1" and the transmission bus idle interrupt is being enabled (SCR:TBIE=1).

Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) can set the transmission/reception data length and set the serial output fixed to “H”.

Figure 18-24 shows the bit configuration of the extended communication control register (ESCR) and Table 18-20 shows the functions of each bit.

Figure 18-24. Bit Configuration of the Extended Communication Control Register (ESCR)

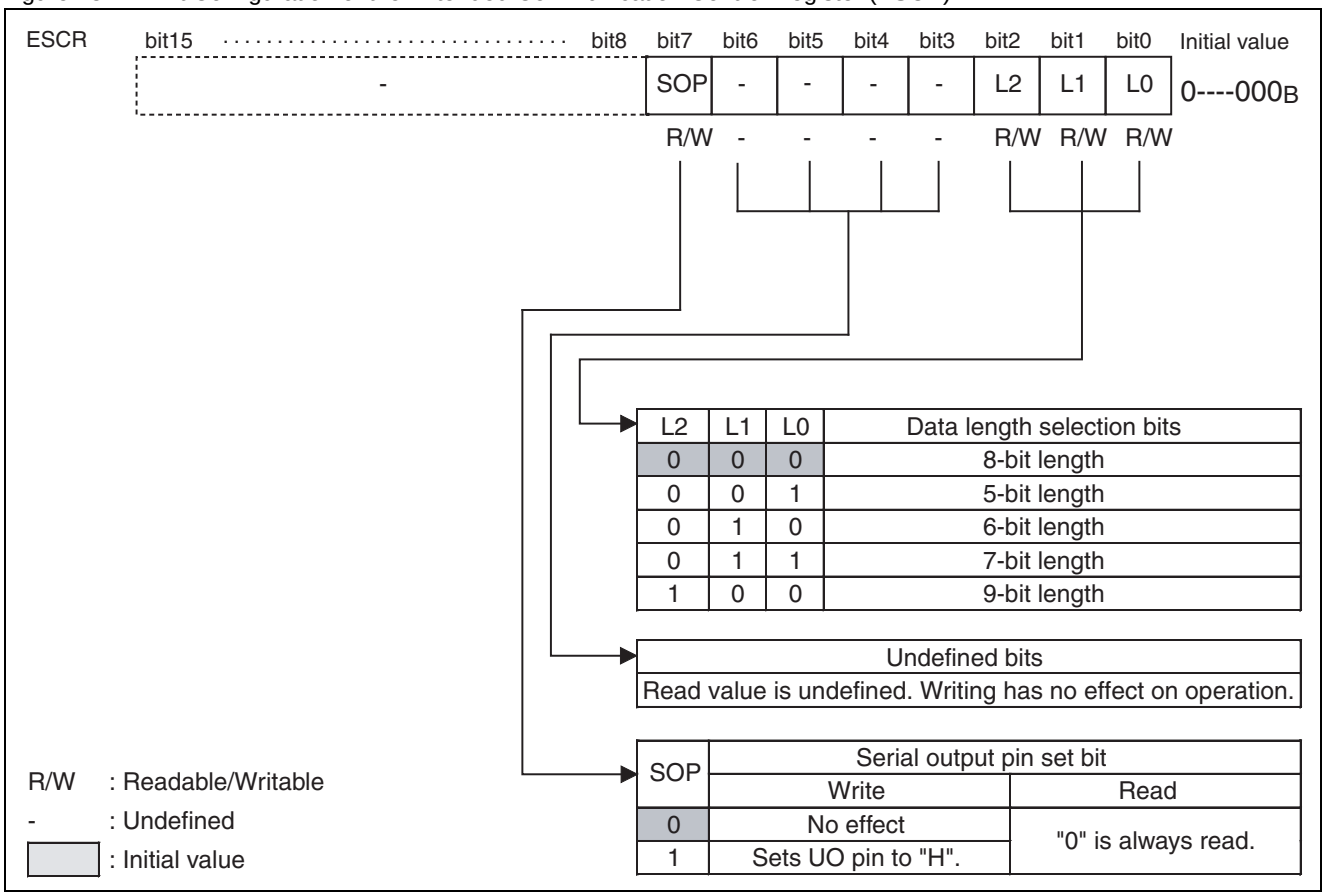


Table 18-20. Function Description of Each Bit of the Extended Communication Control Register (ESCR)

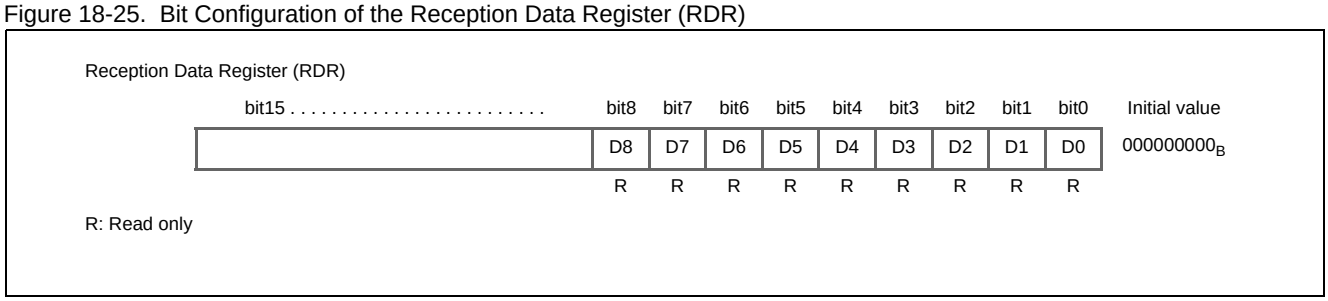
Bit name		Function
bit7	SOP: Serial output pin set bit	■ This bit sets the serial output pin to “H”. When “1” is written to this bit, it sets UO pin to “H”, but there is no need to write “0” to it afterward. ■ When read, “0” is always read. **Notes:** ■ Do not set this bit during serial data transmission. ■ A set value of this bit is effective only for the TXE bit of serial control register (SCR) is “0”.
bit6 to bit3	Unused bits	When read: The value is undefined. When write: No effect
bit2 to bit0	L2 to L0: Data length selection bits	These bits select the data length for transmission/reception data. ■ When “000_B” is set: The data length is set to 8 bits. ■ When “001_B” is set: The data length is set to 5 bits. ■ When “010_B” is set: The data length is set to 6 bits. ■ When “011_B” is set: The data length is set to 7 bits. ■ When “100_B” is set: The data length is set to 9 bits. **Note:** Settings other than the above are prohibited.

Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data register and transmission data register are located at the same address. When reading the address, it functions as a reception data register, and when writing, it functions as a transmission data register.

Figure 18-25 shows the bit configuration of the reception data register (RDR).

Figure 18-25. Bit Configuration of the Reception Data Register (RDR)



The reception data register (RDR) is a 9-bit data buffer register for serial data reception.

- A serial data signal transmitted to the serial input pin (UI pin) is converted in the shift register and stored in the reception data register (RDR).
- Depending on the data length, “0” is entered to the upper bits as follows.

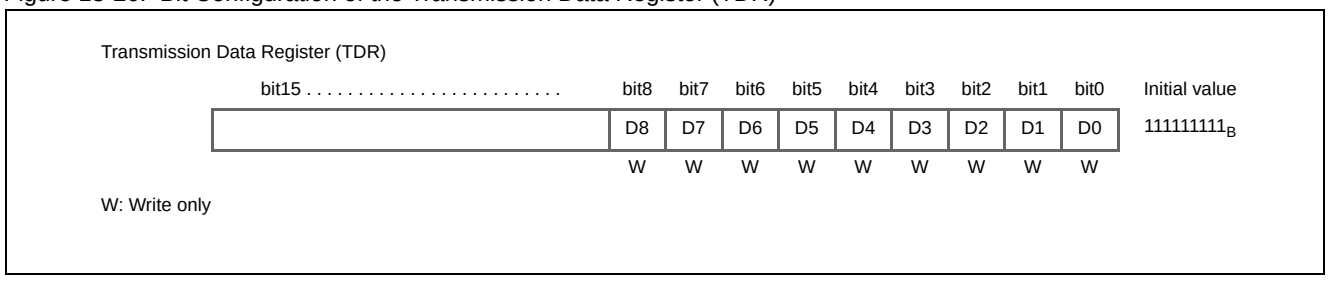
Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	0	X	X	X	X	X	X	X	X
7 bits	0	0	X	X	X	X	X	X	X
6 bits	0	0	0	X	X	X	X	X	X
5 bits	0	0	0	0	X	X	X	X	X

X: Reception data

- When reception data is stored into the reception data register (RDR), the reception data full flag bit (SSR:RDRF) is set to “1”. If reception interrupt is being enabled (SSR:RIE=1), a reception interrupt request is generated.
- Read the reception data register (RDR) while the reception data full flag bit (SSR:RDRF) is “1”. The reception data full flag bit (SSR:RDRF) is cleared to “0” automatically when the reception data register (RDR) is read.
- When a reception error occurs (SSR:ORE), data in the reception data register (RDR) becomes invalid.
- In the case of 9-bit length transfer, reading of the RDR is performed with 16-bit access.

Figure 18-26 shows the bit configuration of the transmission data register.

Figure 18-26. Bit Configuration of the Transmission Data Register (TDR)



The transmission data register (TDR) is a 9-bit data buffer register for serial data transmission.

- When data to be transmitted is written into the transmission data register (TDR) while transmission operations are enabled (SCR:TXE=1), the transmission data is transferred to the shift register for transmission, then converted into serial data, and transmitted from the serial data output pin (UO pin).

- Depending on the data length, data becomes invalid from the upper bits as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	Invalid	X	X	X	X	X	X	X	X
7 bits	Invalid	Invalid	X	X	X	X	X	X	X
6 bits	Invalid	Invalid	Invalid	X	X	X	X	X	X
5 bits	Invalid	Invalid	Invalid	Invalid	X	X	X	X	X

X: Transmission data

- The transmission data empty flag (SSR:TDRE) is cleared to "0" when transmission data is written into the transmission data register (TDR).
- The transmission data empty flag (SSR:TDRE) is set to "1" when transmission data is transferred to the shift register for transmission and transmission starts.
- When the transmission data empty flag (SSR:TDRE) is "1", the next data for transmission can be written. If transmission interrupt is being enabled, a transmission interrupt is generated. Write the next transmission data at the time of transmission interrupt generation or while the transmission data empty flag (SSR:TDRE) is "1".
- When the transmission data empty flag (SSR:TDRE) is "0", transmission data cannot be written into the transmission data register (TDR).
- In the case of 9-bit length transfer, writing to the TDR is performed with 16-bit access.

Note:

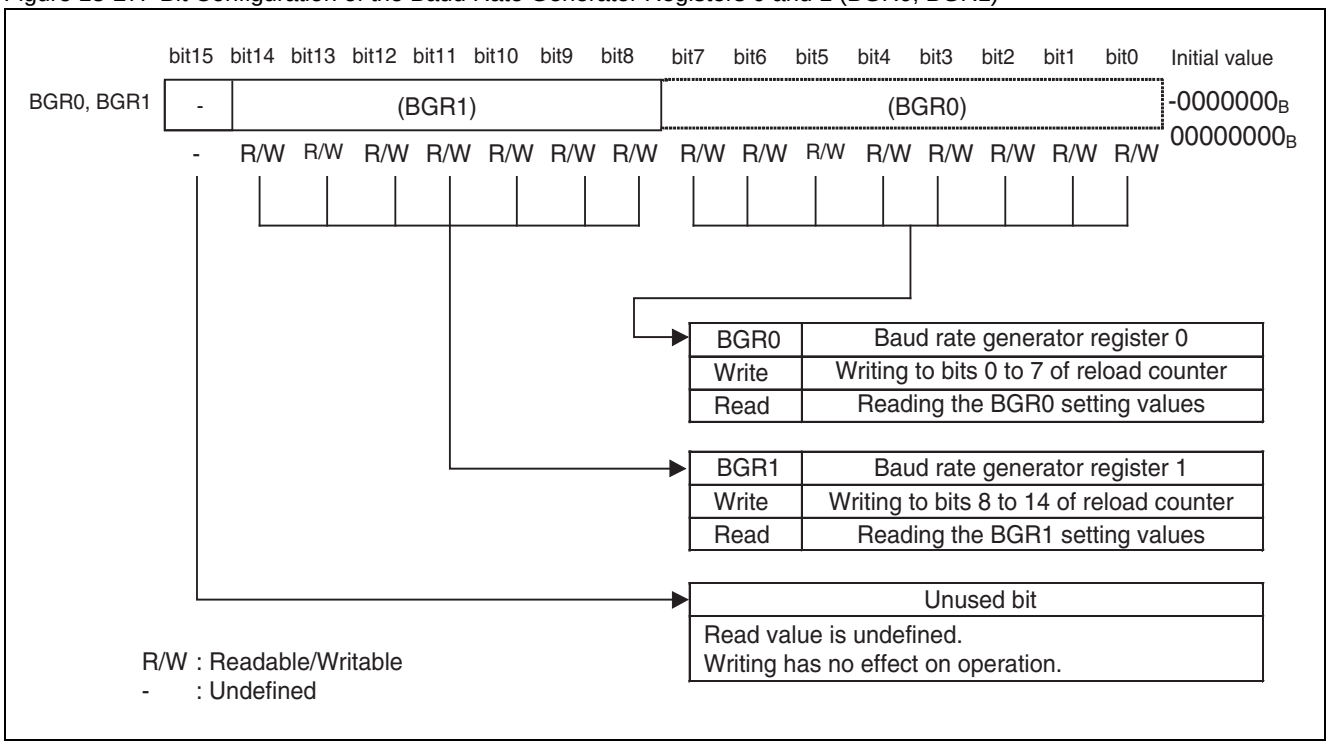
The transmission data register is a write only register, and the reception data register is a read only register. Since the transmission/reception data registers are located at the same address, the write value and the read value are different. Therefore, the read-modify-write (RMW) instruction such as INC/DEC instructions cannot be used.

Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

The baud rate generator registers 0 and 1 (BGR0, BGR1) set the division ratio for the serial clock.

Figure 18-27 shows the bit configuration of the baud rate generator registers 0 and 1(BGR0, BGR1).

Figure 18-27. Bit Configuration of the Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)



- Set values in the baud rate generator registers 0 and 1 (BGR0, BGR1).
- The BGR1 corresponds to the upper bits and BGR0 corresponds to the lower bits, and writing of reload values for counting and reading of the BGR0, BGR1 setting values are allowed.
- When writing reload values to the baud rate generator registers 0 and 1 (BGR0, BGR1), the reload counter starts counting.

Notes:

- When writing into the baud rate generator registers 0 and 1 (BGR0, BGR1), use 16-bit access.
- If a reload value is an even number, the “H” width and “L” width of the serial clock are set by setting the SCINV bit as follows. If it is an odd number, the “H” width and “L” width of the serial clock become the same.
  When SCINV=0, the “H” width of the serial clock becomes longer by one machine clock cycle.
  When SCINV=1, the “L” width of the serial clock becomes longer by one machine clock cycle.
- Set a reload value to 3 or larger.
- When a setting value in the baud rate generator registers 0 and 1 (BGR0, BGR1) is changed, a new setting value is reloaded after the counter value becomes “0000_H”. Therefore, if you want to enable a new setting value immediately, execute a CSIO reset (UPCL) after changing the BGR0, BGR1 setting value.

18.5.3 Interrupts of the CSIO (Clock Synchronous Serial Interface)

The CSIO (clock synchronous serial interface) uses both transmission and reception interrupts. An interrupt request can be generated by one of the following sources:

- When reception data is set into the reception data register (RDR), or when a reception error occurs
- When transmission data is transferred from the transmission data register (TDR) to the shift register for transmission and transmission starts
- Transmission bus idle (no transmission operation)

Interrupts of the CSIO

The interrupt control bits and the interrupt sources of the CSIO are shown in Table 18-21.

Table 18-21. Interrupt Control Bits and Interrupt Sources of the CSIO

Interrupt type	Interrupt request flag bit	Flag register	Interrupt source	Interrupt source enable bit	How to clear the interrupt request flag
Reception	RDRF	SSR	1-byte reception	SCR:RIE	Read the reception data (RDR).
	ORE	SSR	Overrun error		Write “1” into the reception error flag clear bit (SSR:REC).
Trans-mission	TDRE	SSR	Transmission register empty	SCR:TIE	Write into the transmission data (TDR).
	TBI	SSR	No transmission operation	SCR:TBIE	Write into the transmission data (TDR).

Note: Wait until TDRE bit becomes “0” before setting TIE bit to **“1”**.

Reception Interrupt Generation and the Flag Set Timing

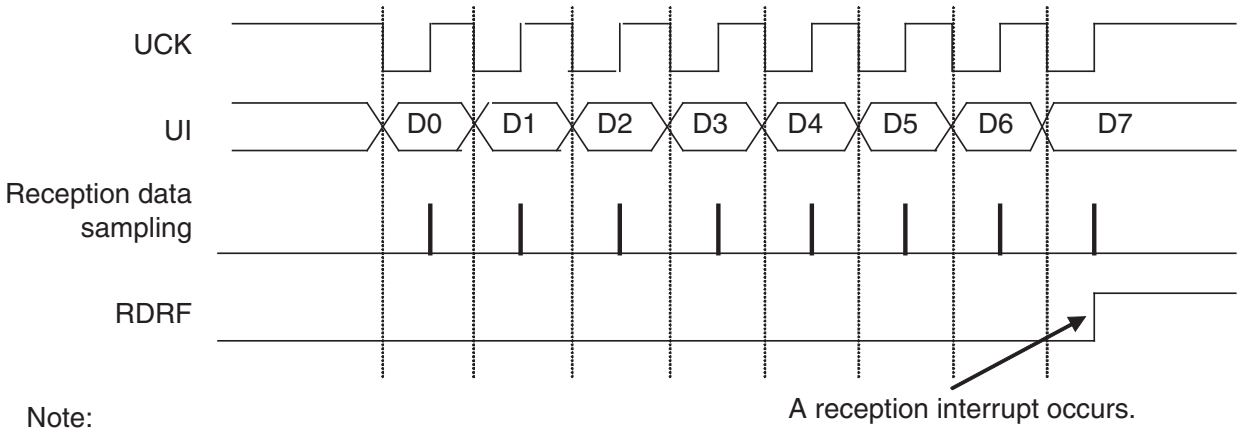
Interrupts during reception are caused by completion of reception (SSR:RDRF) and occurrence of a reception error (SSR:ORE).

Reception data is stored in the reception data register (RDR) when the last data bit is detected. When the reception is completed (SSR:RDRF=1) or when a reception error occurs (SSR:ORE=1), each flag is set. If reception interrupt is being enabled (SSR:RIE=1) at that time, then a reception interrupt is generated.

Note:

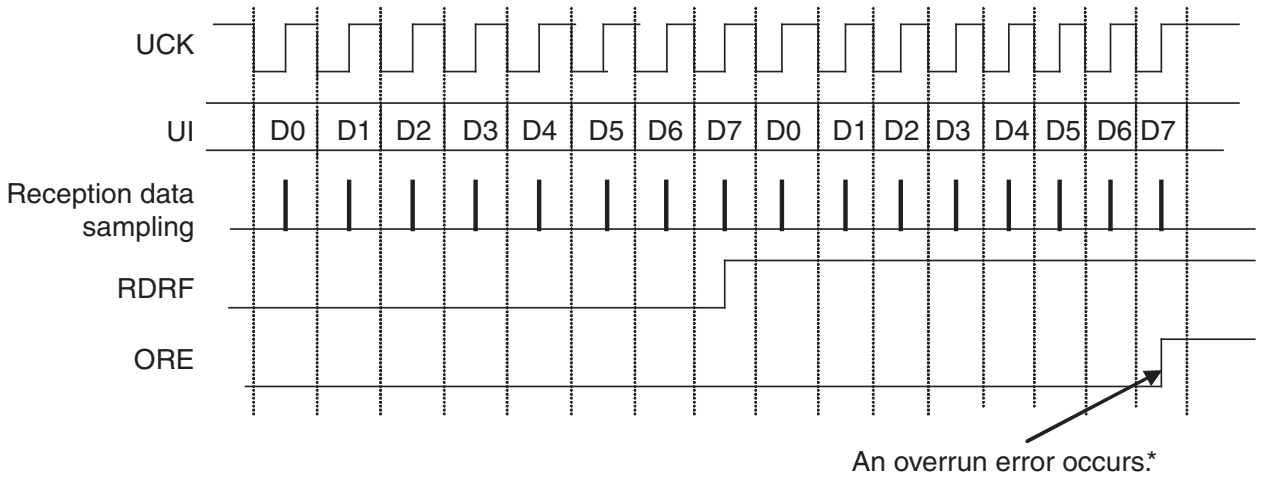
If any reception error occurs, data in the reception data register (RDR) becomes invalid.

Figure 18-28. Reception Operation and the Flag Set Timing



Note:
Figure shows the timings under the following conditions:
SCR: MS=1, SPI=0
ESCR: L2 to L0=000_B
SMR: SCINV=0, BDS=0, SCKE=0, SOE=0

Figure 18-29. ORE (Overrun Error) Flag Set Timing



*: An overrun error occurs if the next data is transferred before the reception data is read (RDRF=1).

Note:
Figure shows the timings under the following conditions:
SCR: MS=1, SPI=0
ESCR: L2 to L0=000_B
SMR: SCINV=0, BDS=0, SCKE=0, SOE=0

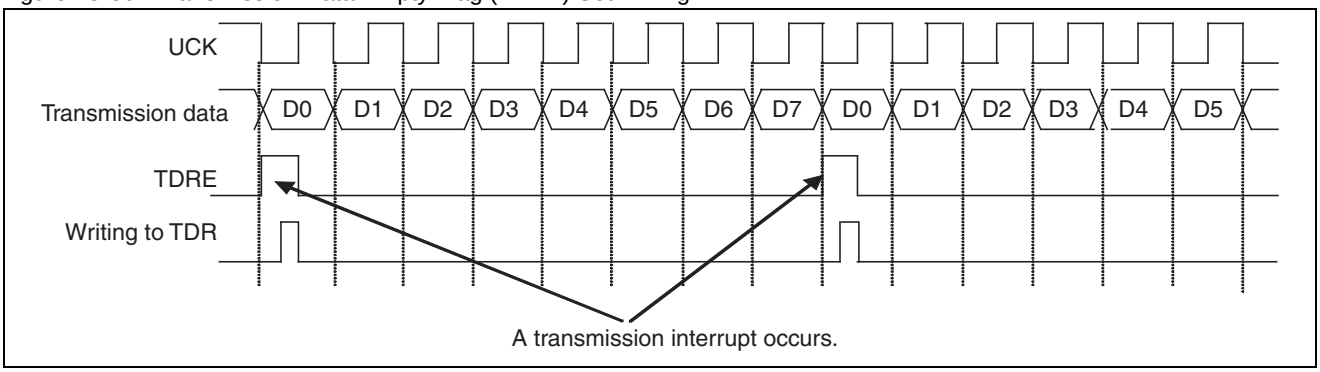
Transmission Interrupt Generation and the Flag Set Timing

Interrupts during transmission are caused when the transmission data is transferred from the transmission data register (TDR) to the shift register for transmission (SSR:TDRE=1) and transmission starts, and when no transmission operation is running (SSR:TBI=1).

Transmission data empty flag (TDRE) set timing

When data written to the transmission data register (TDR) is transferred to the transmission shift register, the state becomes ready for the next data to be written (SSR:TDRE=1). If transmission interrupt is being enabled (SCR:TIE=1) at that time, then a transmission interrupt is generated. Since the TDRE bit is a read only bit, writing data into the transmission data register (TDR) clears the bit to “0”.

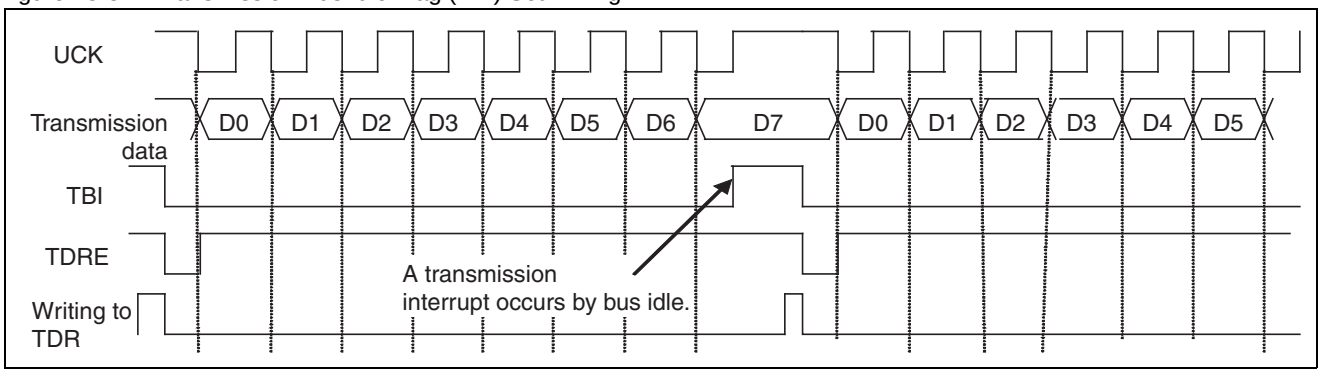
Figure 18-30. Transmission Data Empty Flag (TDRE) Set Timing



Transmission bus idle flag (TBI) set timing

When the transmission data register is empty (TDRE=1) and no transmission operation is running, SSR:TBI bit is set to “1”. If transmission bus idle interrupt is being enabled (SCR:TBIE=1) at that time, then a transmission interrupt is generated. When transmission data is set in the transmission data register (TDR), the TBI bit and the transmission interrupt request are cleared.

Figure 18-31. Transmission Bus Idle Flag (TBI) Set Timing



Operation of the CSIO (Clock Synchronous Serial Interface)

The transfer format is clock synchronous format.

(1) Normal Transfer (I)

Features:

	Item	Description
1	Serial clock (UCK) mark level	“H”
2	Transmission data output timing	UCK falling edge
3	Reception data sampling	UCK rising edge
4	Data length	5 bits to 9 bits

Register settings

The setting values of the registers required for normal transfer (I) are shown below.

Table 18-22. Register Settings for Normal Transfer (I)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	0	*	*	*	*	*	0	1	0	0	0	*	1/0	1/0
SSR/ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	-	-	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	-	-	*	*	*
TDR/RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
	-							*	*	*	*	*	*	*	*	*
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1: “1” setting

0: “0” setting

*: Setting determined by users

Note:

The setting value for the above bits with (1/0) is different for a master operation and a slave operation. Set the value as follows:

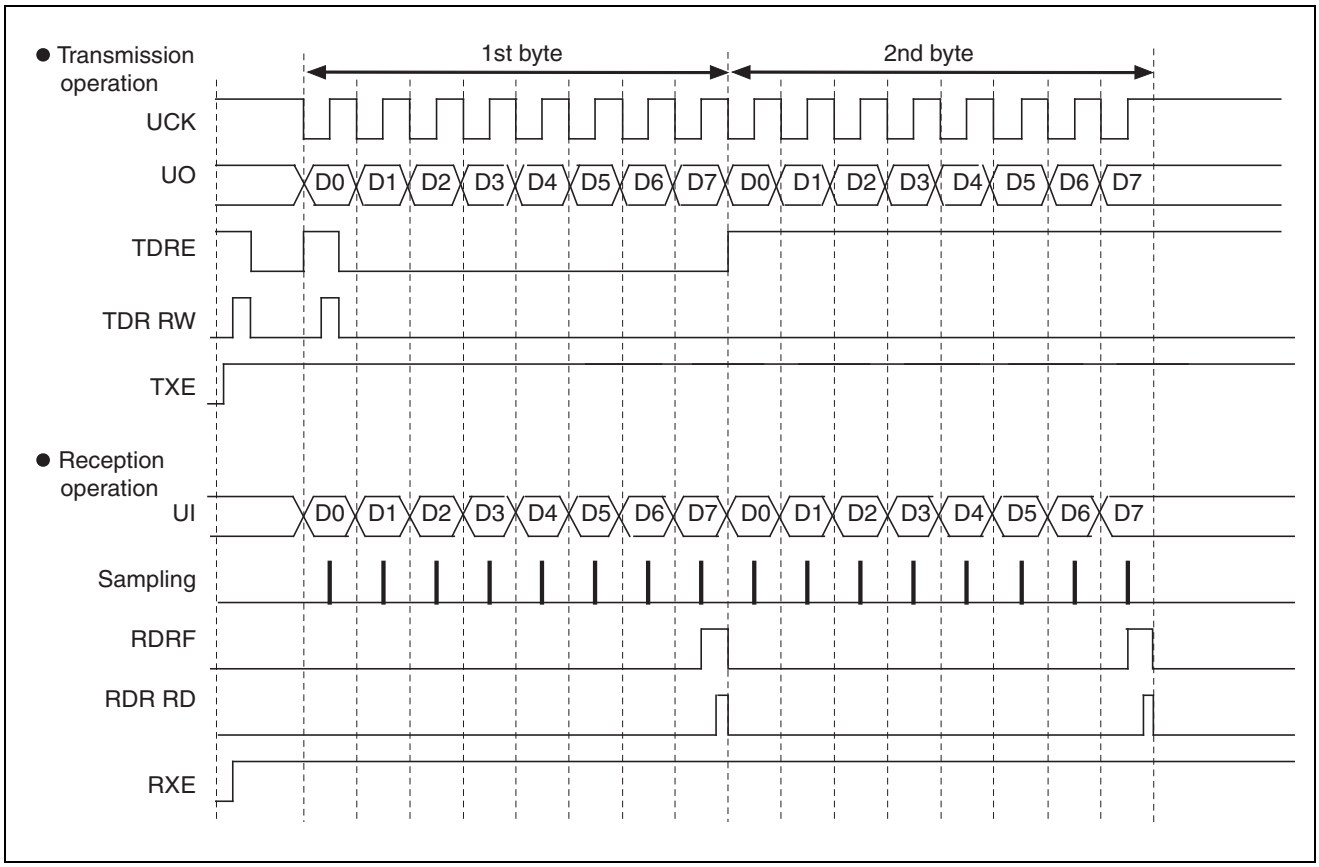
For master transmission: SCR:MS=0, SMR:SCKE=1, SOE=1

For master reception: SCR:MS=0, SMR:SCKE=1, SOE=0

For slave transmission: SCR:MS=1, SMR:SCKE=0, SOE=1

For slave reception: SCR:MS=1, SMR:SCKE=0, SOE=0

Timing chart for normal transfer (I)



Operation descriptions

[1] Master operation (Set SCR:MS=0, SMR:SCKE=1)

- Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), and disabling the reception operation (SCR:RXE=0), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a falling edge of the serial clock (UCK) output.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When dummy data is written to TDR after disabling the serial data output (SMR:SOE=0) and enabling the transmission operation (SCR:TXE=1) and the reception operation (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock (UCK) output.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1).
   The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

Note:

When performing only reception operation, write dummy data into TDR to output the serial clock (UCK).

[2] Slave operation (Set SCR:MS=1, SMR:SCKE=0)

- Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a falling edge of the serial clock (UCK) input.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When the serial data output is disabled (SMR:SOE=0) and the reception operation is enabled (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock (UCK) input.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1).
   The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

(2) Normal Transfer (II)

Features

	Item	Description
1	Serial clock (UCK) mark level	“L”
2	Transmission data output timing	UCK rising edge
3	Reception data sampling	UCK falling edge
4	Data length	5 bits to 9 bits

Register settings

The setting values of the registers required for normal transfer (II) are shown below.

Table 18-23. Register Settings for Normal Transfer (II)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	0	*	*	*	*	*	0	1	0	0	1	*	1/0	1/0
SSR/ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	-	-	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	-	-	*	*	*
TDR/RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
	-							*	*	*	*	*	*	*	*	*
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1: “1” setting

0: “0” setting

*: Setting determined by users

Note:

The setting value for the above bits with (1/0) is different for a master operation and a slave operation. Set the value as follows:

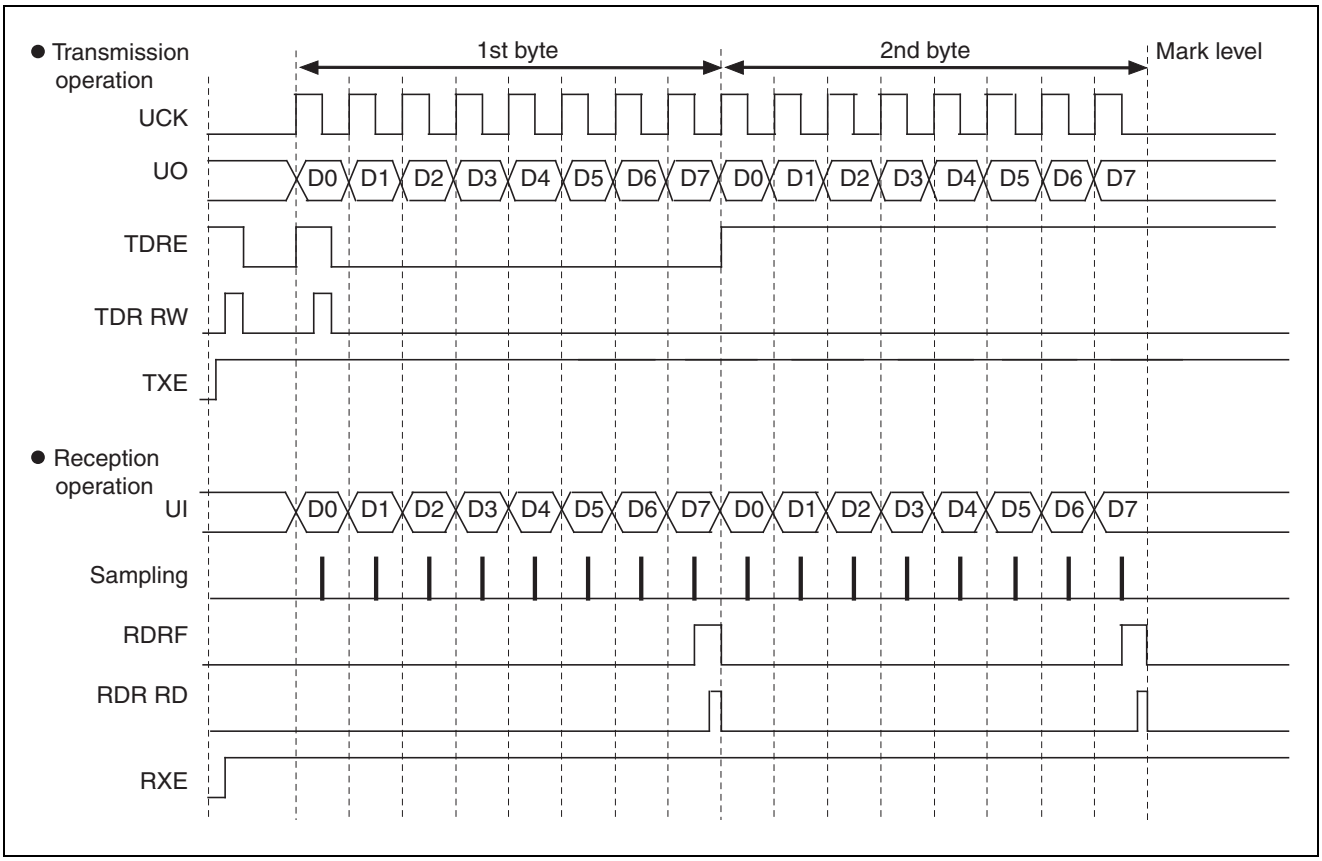
For master transmission: SCR:MS=0, SMR:SCKE=1, SOE=1

For master reception: SCR:MS=0, SMR:SCKE=1, SOE=0

For slave transmission: SCR:MS=1, SMR:SCKE=0, SOE=1

For slave reception: SCRMS=1, SMR:SCKE=0, SOE=0

Timing chart for normal transfer (II)



Operation descriptions

[1] Master operation (Set SCR:MS=0, SMR:SCKE=1)

■ Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), and disabling the reception operation (SCR:RXE=0), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a rising edge of the serial clock (UCK) output.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

■ Reception operation

1. When dummy data is written to TDR after disabling the serial data output (SMR:SOE=0) and enabling the transmission operation (SCR:TXE=1) and the reception operation (SCR:RXE=1), reception data is sampled at a falling edge of the serial clock (UCK) output.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1).
   The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

Note:

When performing only reception operation, write dummy data into TDR to output the serial clock (UCK).

[2] Slave operation (Set SCR:MS=1, SMR:SCKE=0)

- Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a rising edge of the serial clock (UCK) input.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When the serial data output is disabled (SMR:SOE=0) and the reception operation is enabled (SCR:RXE=1), reception data is sampled at a falling edge of the serial clock (UCK) input.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1).
   The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

(3) SPI Transfer (I)

Features

	Item	Description
1	Serial clock (UCK) mark level	“H”
2	Transmission data output timing	UCK rising edge
3	Reception data sampling	UCK falling edge
4	Data length	5 to 9 bits

Register settings

The setting values of the registers required for SPI transfer (I) are shown below.

Table 18-24. Register Setting for SPI Transfer (I)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	1	*	*	*	*	*	0	1	0	0	0	*	1/0	1/0
SSR/ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	-	-	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	-	-	*	*	*
TDR/RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
	-							*	*	*	*	*	*	*	*	*
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1: “1” setting

0: “0” setting

*: Setting determined by users

Note:

The setting value for the above bits with (1/0) is different for a master operation and a slave operation. Set the value as follows:

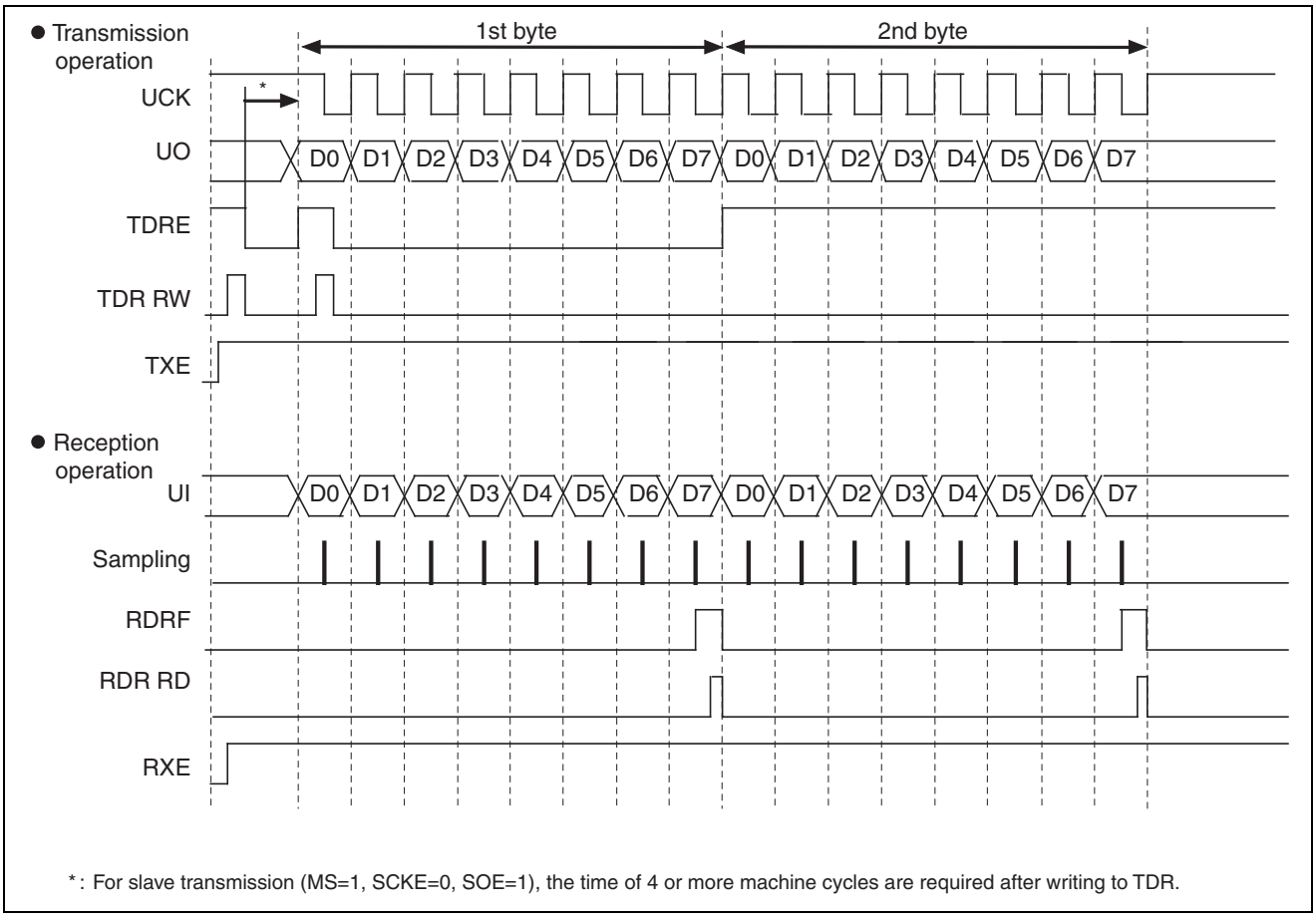
For master transmission: SCR:MS=0, SMR:SCKE=1, SOE=1

For master reception: SCR:MS=0, SMR:SCKE=1, SOE=0

For slave transmission: SCR:MS=1, SMR:SCKE=0, SOE=1

For slave reception: SCR:MS=1, SMR:SCKE=0, SOE=0

Timing chart for SPI transfer (I)



Operation descriptions

[1] Master operation (Set SCR:MS=0, SMR:SCKE=1)

- Transmission operation

1. hen transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), and disabling the reception operation (SCR:RXE=0), SSR:TDRE=0 is set and the first bit is output. Then the transmission data is output in synchronization with a rising edge of the serial clock (UCK) output.
2. SSR:TDRE=1 is set half cycle before the first serial clock falling edge, and a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When dummy data is written to TDR after disabling the serial data output (SMR:SOE=0) and enabling the transmission operation (SCR:TXE=1) and the reception operation (SCR:RXE=1), reception data is sampled at a falling edge of the serial clock (UCK) output.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1). The reception data (RDR) can be read at this time.

Note:

When performing only reception operation, write dummy data into TDR to output the serial clock (UCK).

[2] Slave operation (Set SCR:MS=1, SMR:SCKE=0)

- Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), SSR:TDRE=0 is set and the first bit is output. Then the transmission data is output in synchronization with a rising edge of the serial clock (UCK) output.
2. SSR:TDRE=1 is set half cycle before the first serial clock falling edge, and a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When the serial data output is disabled (SMR:SOE=0) and the reception operation is enabled (SCR:RXE=1), reception data is sampled at a falling edge of the serial clock (UCK) input.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1). The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

(4) SPI Transfer (II)

Features

	Item	Description
1	Serial clock (UCK) mark level	“L”
2	Transmission data output timing	UCK falling edge
3	Reception data sampling	UCK rising edge
4	Data length	5 bits to 9 bits

Register settings

The setting values of the registers required for SPI transfer (II) are shown below.

Table 18-25. Register Setting for SPI Transfer (II)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	1	*	*	*	*	*	0	1	0	0	1	*	1/0	1/0
SSR/ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	-	-	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	-	-	*	*	*
TDR/RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
	-							*	*	*	*	*	*	*	*	*
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1: “1” setting

0: “0” setting

*: Setting determined by users

Note:

The setting value for the above bits with (1/0) is different for a master operation and a slave operation. Set the value as follows:

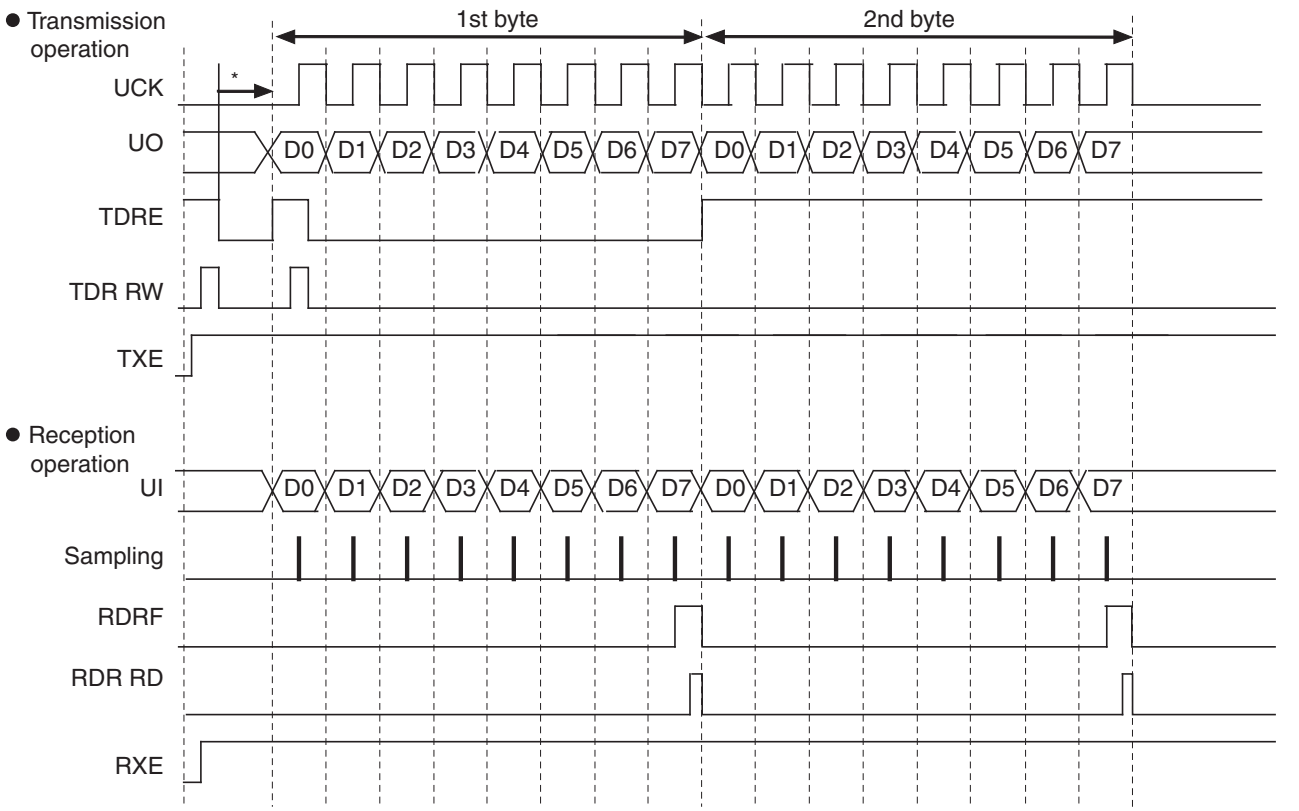
For master transmission: SCR:MS=0, SMR:SCKE=1, SOE=1

For master reception: SCR:MS=0, SMR:SCKE=1, SOE=0

For slave transmission: SCR:MS=1, SMR:SCKE=0, SOE=1

For slave reception: SCR:MS=1, SMR:SCKE=0, SOE=0

Timing chart for SPI transfer (II)



* : For slave transmission (MS=1, SCKE=0, SOE=1), the time of 4 or more machine cycles are required after writing to TDR.

Operation descriptions

[1] Master operation (Set SCR:MS=0, SMR:SCKE=1)

■ Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), and disabling the reception operation (SCR:RXE=0), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a falling edge of the serial clock (UCK) output.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

■ Reception operation

1. When dummy data is written to TDR after disabling the serial data output (SMR:SOE=0) and enabling the transmission operation (SCR:TXE=1) and the reception operation (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock (UCK) output.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1). The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to "0".

Note:

When performing only reception operation, write dummy data into TDR to output the serial clock (UCK).

[2] Slave operation (Set SCR:MS=1, SMR:SCKE=0)

- Transmission operation

1. When transmission data is written to TDR after enabling the serial data output (SMR:SOE=1) and the transmission operation (SCR:TXE=1), SSR:TDRE=0 is set and then the transmission data is output in synchronization with a falling edge of the serial clock (UCK) output.
2. When the first bit of the transmission data is output, SSR:TDRE=1 is set and then a transmission interrupt request is output if transmission interrupt is being enabled (SCR:TIE=1). The second byte of the transmission data can be written at this time.

- Reception operation

1. When the serial data output is disabled (SMR:SOE=0) and the reception operation is enabled (SCR:RXE=1), reception data is sampled at a rising edge of the serial clock (UCK) input.
2. When the last bit is received, SSR:RDRF=1 is set and then a reception interrupt request is output if reception interrupt is being enabled (SCR:RIE=1).
   The reception data (RDR) can be read at this time.
3. If the reception data (RDR) is read, SSR:RDRF is cleared to “0”.

18.5.4 Dedicated Baud Rate Generator

The dedicated baud rate generator operates only in master operation.

CSIO (Clock Synchronous Serial Interface) Baud Rate Selection

The dedicated baud rate generator setting is different for a master operation and a slave operation.

[1] Master operation

The baud rate is selected by dividing an internal clock using the dedicated baud rate generator.

- The CSIO has two internal reload counters and each of them corresponds to transmission/reception serial clock. The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 1 and 0 (BGR0, BGR1).
- The reload counter divides an internal clock by a setting value.

[2] Slave operation

The dedicated baud rate generator does not operate in slave operation (SCR:MS=1).
(An external clock input from the clock input pin UCK is used directly.)

Baud Rate Setting

The following describes the baud rate setting. It also shows a calculation result of a serial clock frequency.

Calculation of Baud Rate

Two 15-bit reload counters are set using the baud rate generator registers 1 and 0 (BGR0, BGR1).

Calculation formula of baud rate is shown below.

1. Reload value

$V = \phi/b - 1$

V: Reload value b: Baud rate ϕ: Machine clock, external clock frequency

2. Example of the calculation

When the machine clock is set to 16MHz, an internal clock is used, and the baud rate is set to 19200 bps, the reload value can be calculated as follows:

Reload value:

$V = (16 \times 1000000)/19200 - 1 = 832$

Therefore, the baud rate is:

$b = (16 \times 1000000)/(832 + 1) = 19208$ bps

3. Baud rate error

The baud rate error can be calculated using the following formula:

Error (%) = (calculated value -desired value)/desired value × 100

(Example) When the machine clock is set to 20MHz and the desired baud rate is set to 153600 bps:

Reload value = $(20 \times 1000000)/153600 - 1 = 129$

Baud rate (calculated value) = $(20 \times 1000000)/(129 + 1) = 153846$ (bps)

Error (%) = (153846 -153600)/153600 × 100 = 0.16 (%)

Notes:

- The reload counter will be stopped when the reload value is set to “0”.
- If a reload value is an even number, the “H” width and “L” width of the serial clock are set by setting the SCINV bit as follows. If it is an odd number, the “H” width and “L” width of the serial clock become the same.
  When SCINV=0, the “H” width of the serial clock becomes longer by one machine clock cycle.
  When SCINV=1, the “L” width of the serial clock becomes longer by one machine clock cycle.
- Set a reload value to 3 or larger.

Reload Values and Baud Rates Corresponding to Each Machine Clock Frequency

Table 18-26. Reload Values and Baud Rates

Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	-	-	-	-	-	-	-	-	-	-	3	0
6M	-	-	-	-	-	-	-	-	3	0	-	-
5M	-	-	-	-	-	-	3	0	-	-	-	-
4M	-	-	-	-	3	0	4	0	5	0	7	0
2.5M	-	-	3	0	-	-	-	-	-	-	-	-
2M	3	0	4	0	7	0	9	0	11	0	15	0
1M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	-0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	-	-	103	-0.16	-	-
153600	51	-0.16	64	-0.16	103	-0.16	129	-0.16	155	-0.16	207	-0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	68	-0.64	86	0.22	138	0.08	173	0.22	207	-0.16	277	0.08
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	311	-0.16	416	0.08
57600	138	0.08	173	0.22	277	0.08	346	-0.16	416	0.08	555	0.08
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	624	0	832	-0.04
28800	277	0.08	346	< 0.01	554	-0.01	693	-0.06	832	-0.03	1110	-0.01
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1249	0	1666	0.02
10417	767	< 0.01	959	< 0.01	1535	< 0.01	1919	< 0.01	2303	< 0.01	3071	< 0.01
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	2499	0	3332	-0.01
7200	1110	< 0.01	1388	< 0.01	2221	< 0.01	2777	< 0.01	3332	< 0.01	4443	-0.01
4800	1666	0.02	2082	-0.02	3332	< 0.01	4166	< 0.01	4999	0	6666	< 0.01
2400	3332	< 0.01	4166	< 0.01	6666	< 0.01	8332	< 0.01	9999	0	13332	< -0.01
1200	6666	< 0.01	8334	0.02	13332	< 0.01	16666	< 0.01	19999	0	26666	< 0.01
600	13332	< 0.01	16666	< 0.01	26666	< 0.01	-	-	-	-	-	-
300	26666	26666	< 0.01	-	-	-	-	-	-	-	-	-

- Value : Setting value in BGR0, BGR1 register (decimal)
- ERR : Baud rate error (%)

Function of Reload Counter

Two types of reload counter, a transmission reload counter and a reception reload counter, are available and each of them functions as a dedicated baud rate generator. It consists of a 15-bit register corresponding to reload values and it generates transmission/reception clock from an internal clock.

Count Start

The reload counter starts counting when a reload value is written into the baud rate generator registers 0 and 1 (BGR0, BGR1).

Restart

The reload counter restarts under the following condition:

- For both transmission/reception reload counters
  Programmable reset (SCR:UPCL bit)

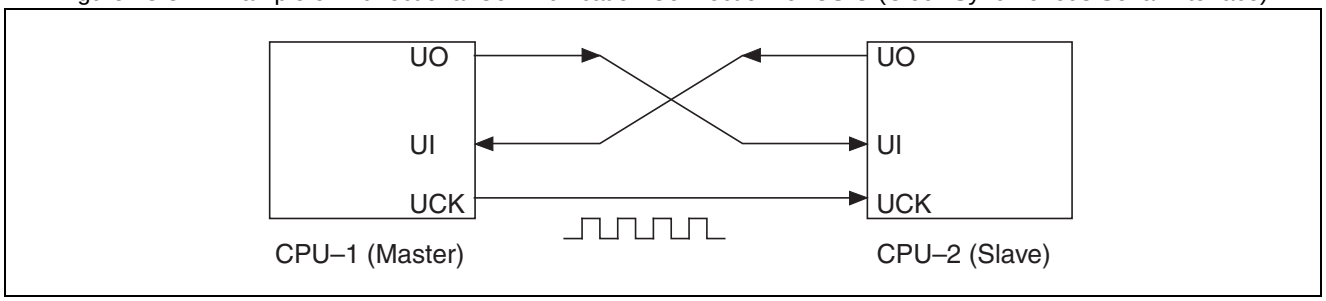
Setting Procedure and the Program Flow for CSIO (Clock Synchronous Serial Interface)

The CSIO (clock synchronous serial interface) allows serial bidirectional synchronous communication.

Connection between CPUs

For CSIO (clock synchronous serial interface), select bidirectional communication. Connect two CPUs to each other as shown in Figure 18-32.

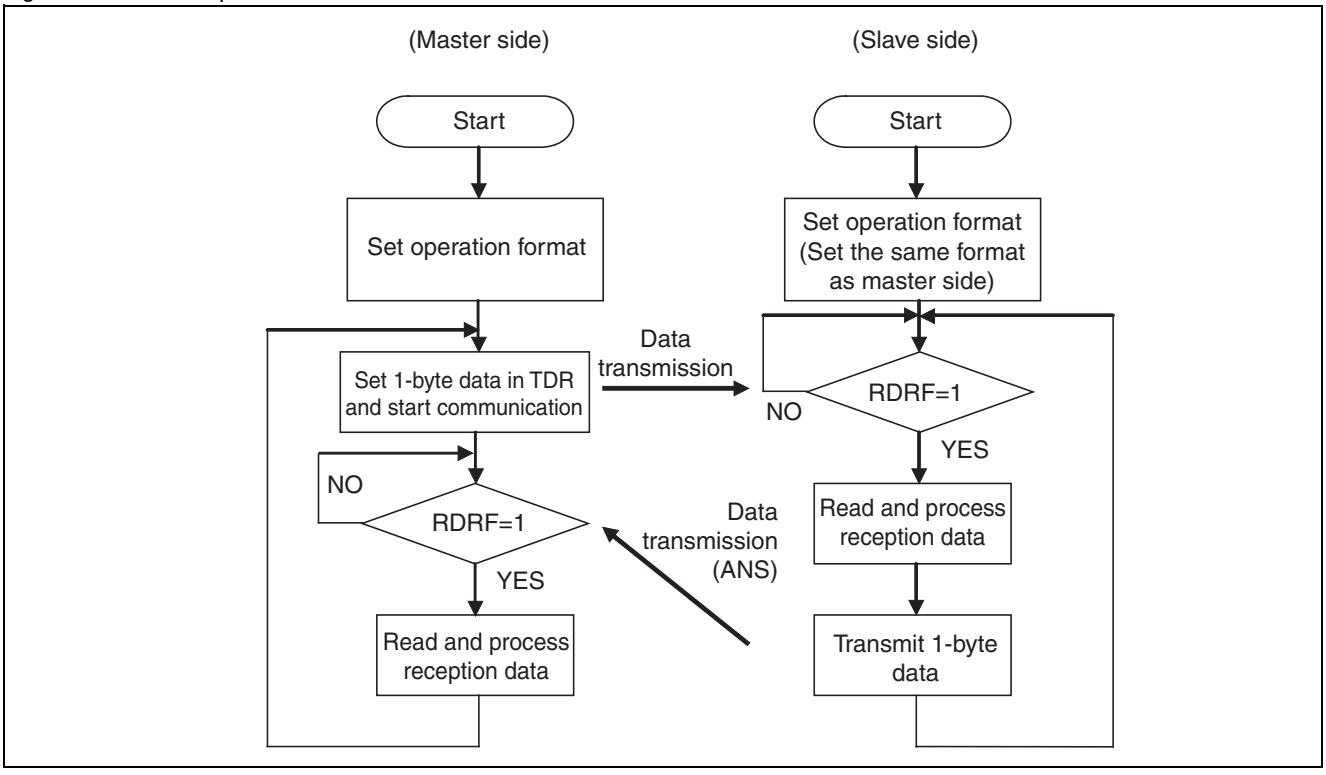
Figure 18-32. Example of Bidirectional Communication Connection for CSIO (Clock Synchronous Serial Interface)



Flowchart

- Bidirectional communication flowchart

Figure 18-33. Example of Bidirectional Communication Flowchart



18.6 LIN-UART (LIN Communication Control UART)

This section describes the LIN communication functions supported by operation mode 3 among multi-function serial interface functions.

LIN-UART (LIN Communication Control UART)

Overview of the LIN-UART (LIN Communication Control UART)

Registers of the LIN-UART

- Serial Control Register (SCR)
- Serial Mode Register (SMR)
- Serial Status Register (SSR)
- Extended Communication Control Register (ESCR)
- Reception Data Register/Transmission Data Register (RDR/TDR)
- Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

Interrupts of the LIN-UART

- Reception Interrupt Generation and the Flag Set Timing
- Transmission Interrupt Generation and the Flag Set Timing

Dedicated Baud Rate Generator

Baud Rate Setting

Operation of the LIN-UART

Setting Procedure and the Program Flow for Operation Mode 3 (LIN Communication Mode)

18.6.1 Overview of the LIN-UART (LIN Communication Control UART)

The LIN-UART (LIN communication control UART) supports special functions to support LIN bus.

Functions of the LIN-UART (LIN Communication Control UART)

		Functions
1	Data buffer	■ Full-duplex double buffer ■ Transmission/reception
2	Serial input	After oversampling three times using machine clock, the reception value is determined by a majority of the sampling values.
3	Transfer mode	Asynchronous
4	Baud rate	■ Dedicated baud rate generator (consisting of a 15-bit reload counter) ■ An external clock can be adjusted using the reload counter.
5	Data length	8 bits
6	Signal format	NRZ (Non Return to Zero)
7	Start bit detection	Synchronization with start bit falling edge
8	Reception error detection	■ Framing error ■ Overrun error
9	Interrupt requests	■ Reception interrupts (Reception completion, framing error, overrun error) ■ Transmission interrupts (Transmission data empty, transmission bus idle) ■ Status interrupt (LIN synch break detection) ■ Interrupt request to ICU (LIN synch field detection: LSYN) ■ Both transmission and reception have extended intelligent I/O service (EI^{2}OS) and DMA functions.
10	LIN bus options	■ LIN protocol Revision 2.0 supported ■ Master device operation ■ Slave device operation ■ LIN Synch break generation (variable length between 13 bits and 16 bits) ■ Synch Delimiter generation (variable length between 1 bit and 4 bits) ■ LIN Synch break detection ■ Detection of start/stop edge of LIN synch field connecting to an input capture

18.6.2 Registers of the LIN-UART

A register list of the LIN-UART is shown below.

Register List of the LIN-UART

Table 18-27. Register List of the LIN-UART

	Address		bit15 bit8	bit7 bit0
LIN-UART	ch.0:000021$_H$ ch.1:00002B$_H$ ch.2:00003F$_H$ ch.3:000049$_H$ ch.4:000053$_H$ ch.5:00005D$_H$ ch.6:007791$_H$	ch.0:000020$_H$ ch.1:00002A$_H$ ch.2:00003E$_H$ ch.3:000048$_H$ ch.4:000052$_H$ ch.5:00005C$_H$ ch.6:007990$_H$	SCR (Serial control register)	SMR (Serial mode register)
	ch.0:000023$_H$ ch.1:00002D$_H$ ch.2:000041$_H$ ch.3:00004B$_H$ ch.4:000055$_H$ ch.5:00005F$_H$ ch.6:007993$_H$	ch.0:000022$_H$ ch.1:00002C$_H$ ch.2:000040$_H$ ch.3:00004A$_H$ ch.4:000054$_H$ ch.5:00005E$_H$ ch.6:007992$_H$	SSR (Serial status register)	ESCR (Extended communication control register)
	ch.0:000025$_H$ ch.1:00002F$_H$ ch.2:000043$_H$ ch.3:00004D$_H$ ch.4:000057$_H$ ch.5:000061$_H$ ch.6:007995$_H$	ch.0:000024$_H$ ch.2:00002E$_H$ ch.3:000042$_H$ ch.4:00004C$_H$ ch.5:000056$_H$ ch.6:000060$_H$ ch.7:007994$_H$	-	RDR/TDR (Reception/transmission data register)
	ch.0:000027$_H$ ch.1:000031$_H$ ch.2:000045$_H$ ch.3:00004F$_H$ ch.4:000059$_H$ ch.5:000063$_H$ ch.6:007997$_H$	ch.0:000026$_H$ ch.1:000030$_H$ ch.2:000044$_H$ ch.3:00004E$_H$ ch.4:000058$_H$ ch.5:000062$_H$ ch.6:007996$_H$	BGR1 (Baud rate generator register 1)	BGR0 (Baud rate generator register 0)
	ch.0:000029$_H$ ch.1:000033$_H$ ch.2:000047$_H$ ch.3:000051$_H$ ch.4:00005B$_H$ ch.5:000065$_H$ ch.6:007999$_H$	ch.0:000028$_H$ ch.1:000032$_H$ ch.2:000046$_H$ ch.3:000050$_H$ ch.4:00005A$_H$ ch.5:000064$_H$ ch.6:007998$_H$	-	-

Table 18-28. Bit Assignment of the LIN-UART

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/SMR	UPCL	MS	LBR	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	Reser-ved	SBL	-	SCKE	SOE
SSR/ESCR	REC	-	LBD	FRE	ORE	RDRF	TDRE	TBI	-	-	-	LBIE	LBL1	LBL0	DEL1	DEL0
TDR/RDR	-								D7	D6	D5	D4	D3	D2	D1	D0
BGR1/BGR0	EXT	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0

Serial Control Register (SCR)

The serial control register (SCR) enables/disables transmission/reception interrupts, transmission idle interrupts, and transmission/reception operations. It also generates LIN Synch Break and has LIN-UART reset setting.

Figure 18-34 shows the bit configuration of the serial control register (SCR) and Table 18-29 shows the functions of each bit.

Figure 18-34. Bit Configuration of the Serial Control Register (SCR)

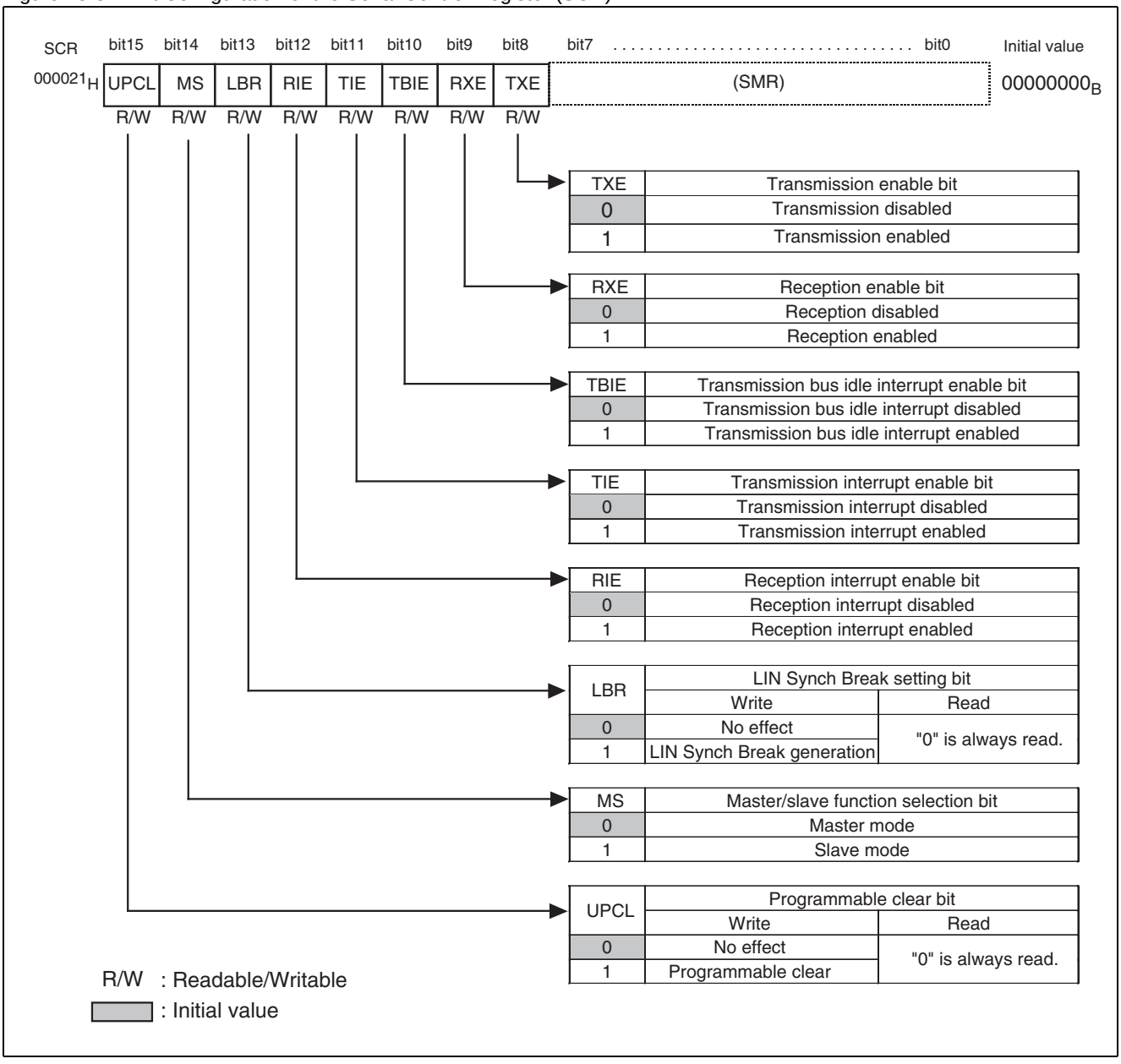


Table 18-29. Function Description of Each Bit of the Serial Control Register (SCR)

Bit name		Function
bit15	UPCL: Programmable clear bit	This bit initializes the internal state of the LIN-UART. When “1” is set: ■ The LIN-UART is reset directly (software reset). However, the register settings are retained. If a register is in transmission/reception state at that time, it will be disconnected immediately. ■ The baud rate generator reloads the setting values in BGR0, BGR1 registers and restarts. ■ All transmission/reception interrupt sources (TDRE, TBI, RDRF, FRE, ORE, and LBD) are initialized. ■ When “0” is set: No effect “0” is always read when read. **Note:** After interrupt disabled is set, execute a programmable clear.
bit14	MS: Master/slave function selection bit	This bit selects master or slave mode. When “0” is set: Master mode is set. When “1” is set: Slave mode is set.
bit13	LBR: LIN Synch Break setting bit (operable only in master operation)	When “1” is set in this bit, it generates LIN Synch Break and LIN Synch delimiter having the length set in the ESCR:LBL1/LBL0 and DEL1/DEL0 bits. When write: “0” is written: No effect “1” is written: It generates LIN Synch Break. When read: “0” is always read. **Note:** This bit operates only in master operation.
bit12	RIE: Reception interrupt enable bit	■ This bit enables/disables the output of reception interrupt requests to the CPU. ■ If RIE bit and reception data flag bit (SSR:RDRF) are “1”, or if any one of the error flag bits (SSR:LER, FRE, ORE) is “1”, a reception interrupt request is output.
bit11	TIE: Transmission interrupt enable bit	■ This bit enables/disables the output of transmission interrupt requests to the CPU. ■ If TIE bit and SSR:TDRE bit are “1”, a transmission interrupt request is output.
bit10	TBIE: Transmission bus idle interrupt enable bit	■ This bit enables/disables the output of transmission bus idle interrupt requests to the CPU. ■ If TBIE bit and SSR:TBI bit are “1”, a transmission bus idle interrupt request is output.
bit9	RXE: Reception operation enable bit	This bit enables/disables the reception operation of the LIN-UART. ■ If “0” is set: Data frame reception operation is disabled. ■ If “1” is set: Data frame reception operation is enabled. **Notes:** ■ Even if the reception operation is enabled (RXE=1), it does not start until a start bit falling edge is input. ■ In master operation, even if the reception operation is enabled (RXE=1), the data is not received during LIN Synch Break transmission. ■ If the reception operation is disabled (RXE=0) during reception, the operation is stopped immediately.
bit8	TXE: Transmission operation enable bit	This bit enables/disables the transmission operation of the LIN-UART. ■ If “0” is set: Data frame transmission operation is disabled. ■ If “1” is set: Data frame transmission operation is enabled. **Note:** If the transmission operation is disabled (TXE=0) during transmission, the operation is stopped immediately.

Serial Mode Register (SMR)

The serial mode register (SMR) sets the operation mode, selects transfer direction, data length and stop bit length, and enables/disables the outputs to the serial data and serial clock pins.

Figure 18-35 shows the bit configuration of the serial mode register (SMR) and Table 18-30 shows the functions of each bit.

Figure 18-35. Bit Configuration of the Serial Mode Register (SMR)

SMR	bit15 ····· bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
	(SCR)	MD2	MD1	MD0	Reserved	SBL	-	SCKE	SOE	$00000\text{-}00_B$
		R/W	R/W	R/W	R/W	R/W	-	R/W	R/W	

SOE	Serial data output enable bit
0	General-purpose input/output port
1	Serial data output pin

SCKE	Serial clock output enable bit
0	General-purpose input/output port or serial clock input pin
1	Serial clock output pin

Unused bit
Read value is undefined. Writing has no effect on operation.

SBL	Stop bit length selection bit
0	1 bit
1	2 bits

Reserved bit
Be sure to write "0".

MD2	MD1	MD0	Operation mode setting bits
0	0	0	Operation mode 0 (Asynchronous normal mode)
0	0	1	Operation mode 1 (Asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (Clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^2C mode)

R/W : Readable/Writable
\- : Undefined
: Initial value

Note:
This section describes the registers and the operations in operation mode 3.

Table 18-30. Function Description of Each Bit of the Serial Mode Register (SMR)

Bit name		Function
bit7 to bit5	MD2 to MD0: Operation mode setting bits	These bits set the operation mode. "000_B": Operation mode 0 (asynchronous normal mode) is set. "001_B": Operation mode 1 (asynchronous multiprocessor mode) is set. "010_B': Operation mode 2 (clock synchronous mode) is set. "011_B': Operation mode 3 (LIN communication mode) is set. "100_B": Operation mode 4 (I^2C mode) is set. This section describes the registers and the operations in operation mode 3 (LIN communication mode). **Notes:** ■ Settings other than the above are prohibited. ■ To switch the operation mode, do so after executing a programmable clear (SCR:UPCL=1). ■ Set each register after setting the operation mode.
bit4	Reserved bit	Be sure to write "0".
bit3	SBL: Stop bit length selection bit	This bit sets the bit length of the stop bit (frame end mark of transmission data). When "0" is set: Stop bit length is set to 1 bit. When "1" is set: Stop bit length is set to 2 bits. **Notes:** ■ For reception, only the first bit of the stop bit is always detected. ■ Set this bit when the transmission is disabled (TXE=0).
bit2	Unused bit	When read: The value is undefined. When write: It has no effect.
bit1	SCKE: Serial clock output enable bit	This bit controls input/output ports of the serial clock. When "0" is set: UCK pin becomes a general-purpose input/output port or a serial clock input pin. When "1" is set: UCK pin becomes a serial clock output pin and outputs the clock. **Note:** When the UCK pin is used as a serial clock input (SCKE=0), set a general-purpose input/output port to an input port. Also, select the external clock (BGR:EXT=1) using the external clock selection bit. **Reference:** If the UCK pin is set to serial clock output (SCKE=1), it functions as a serial clock output pin regardless of the general-purpose input/output port (DDR) settings.
bit0	SOE: Serial data output enable bit	This bit enables/disables the output of the serial data. When "0" is set: UO pin becomes a general-purpose input/output port. When "1" is set: UO pin becomes a serial data output pin (UO). **Reference:** When this bit is set to serial data output (SOE=1), the UO pin functions as a UO pin regardless of the general-purpose input/output port (DDR) settings.

Serial Status Register (SSR)

The serial status register (SSR) checks transmission/reception statuses and reception error flags, detects LIN Synch break, and clears the reception error flags.

Figure 18-36 shows the bit configuration of the serial status register (SSR) and Table 18-31 shows the functions of each bit.

Figure 18-36. Bit Configuration of the Serial Status Register (SSR)

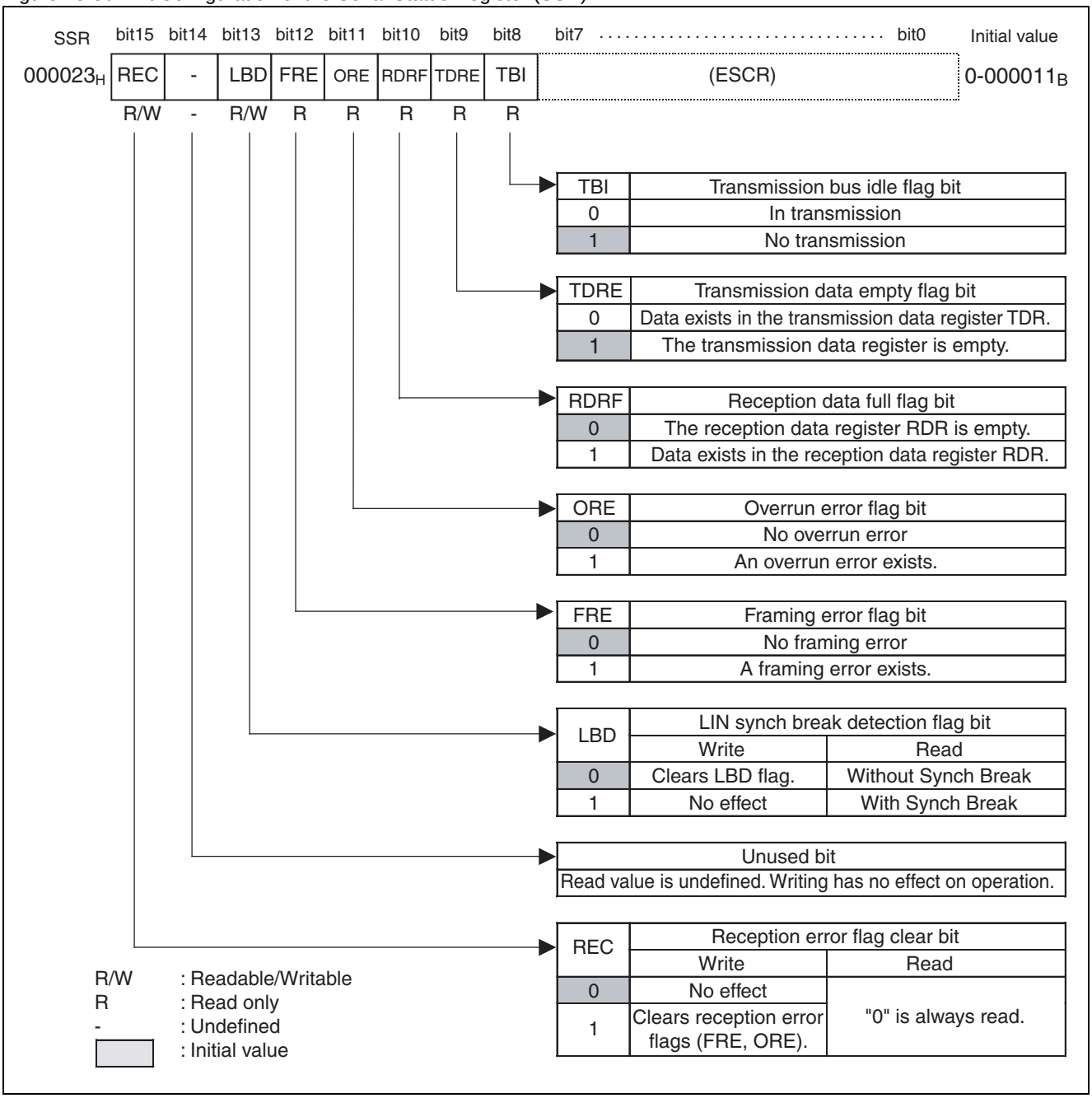


Table 18-31. Function Description of Each Bit of the Serial Status Register (SSR)

Bit name		Function
bit15	REC: Reception error flag clear bit	This bit clears FRE and ORE flags of the serial status register (SSR). ■ When “1” is written, error flags are cleared. ■ When “0” is written, this writing has no effect. When read, “0” is always read.
bit14	Unused bit	When read: The value is undefined. When write: No effect
bit13	LBD: LIN Synch Break detection flag bit (operable only in slave operation)	This bit indicates LIN Synch Break detection. When “0” is input to the serial input (UI) for 11-bit width or over, the LBD bit is set to “1”. If “1” is set in the LIN Synch Break interrupt enable bit (LBIE) at this time, then a status interrupt is generated. (When reading) When “1”: LIN Synch Break has been detected. When “0”: LIN Synch Break has not been detected. (When writing) When “0” is written: Clears LBD bit. When “1” is written: No effect When a read-modify-write (RMW) instruction is issued, “1” is read. **Notes:** ■ This function is operable only in slave operation. ■ When a read-modify-write (RMW) instruction is issued, “1” is read.
bit12	FRE: Framing error flag bit	■ This bit is set to “1” when a framing error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the FRE bit and RIE bit are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit11	ORE: Overrun error flag bit	■ This bit is set to “1” when an overrun error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the ORE bit and RIE bit are “1”. ■ Data in the reception data register (RDR) is invalid when this flag is set.
bit10	RDRF: Reception data full flag bit	■ This flag indicates the status of the reception data register (RDR). ■ This bit is set to “1” when reception data is loaded into the RDR and is cleared to “0” when the reception data register (RDR) is read. ■ A reception interrupt request is output when the RDRF bit and RIE bit are “1”.
bit9	TDRE: Transmission data empty flag bit	■ This flag indicates the status of the transmission data register (TDR). ■ This bit becomes “0” when transmission data is written to TDR, indicating that valid data exists in the TDR. This bit becomes “1” when the data is loaded into the transmission shift register and transmission starts, indicating that valid data does not exist in the TDR. ■ A transmission interrupt request is output when the TDRE bit and TIE bit are “1”. ■ The TDRE bit becomes “1” when the UPCL bit in the serial control register (SCR) is set to “1”.
bit8	TBI: Transmission bus idle flag bit	■ This bit indicates that the LIN-UART is not in transmission operations. ■ This bit becomes “0” when transmission data is written to the transmission data register (TDR). ■ This bit becomes “0” when LIN Synch Break is set (LBR=1). ■ This bit becomes “1” when the transmission data register (TDR) is empty (TDRE=1) and no transmission operation is performed. ■ This bit becomes “1” when LIN Synch Break is set (LBR=1). ■ This bit becomes “1” when the transmission data register is empty after LIN Synch Break transmission is completed. ■ A transmission interrupt request is output when this bit is “1” and the transmission bus idle interrupt is being enabled (SCR:TBIE=1).

Extended Communication Control Register (ESCR)

The extended communication control register (ESCR) enables/disables LIN Synch Break interrupts, detects LIN Synch Break, and sets LIN Synch Break length and Synch delimiter length.

Figure 18-37 shows the bit configuration of the extended communication control register (ESCR) and Table 18-32 shows the functions of each bit.

Figure 18-37. Bit Configuration of the Extended Communication Control Register (ESCR)

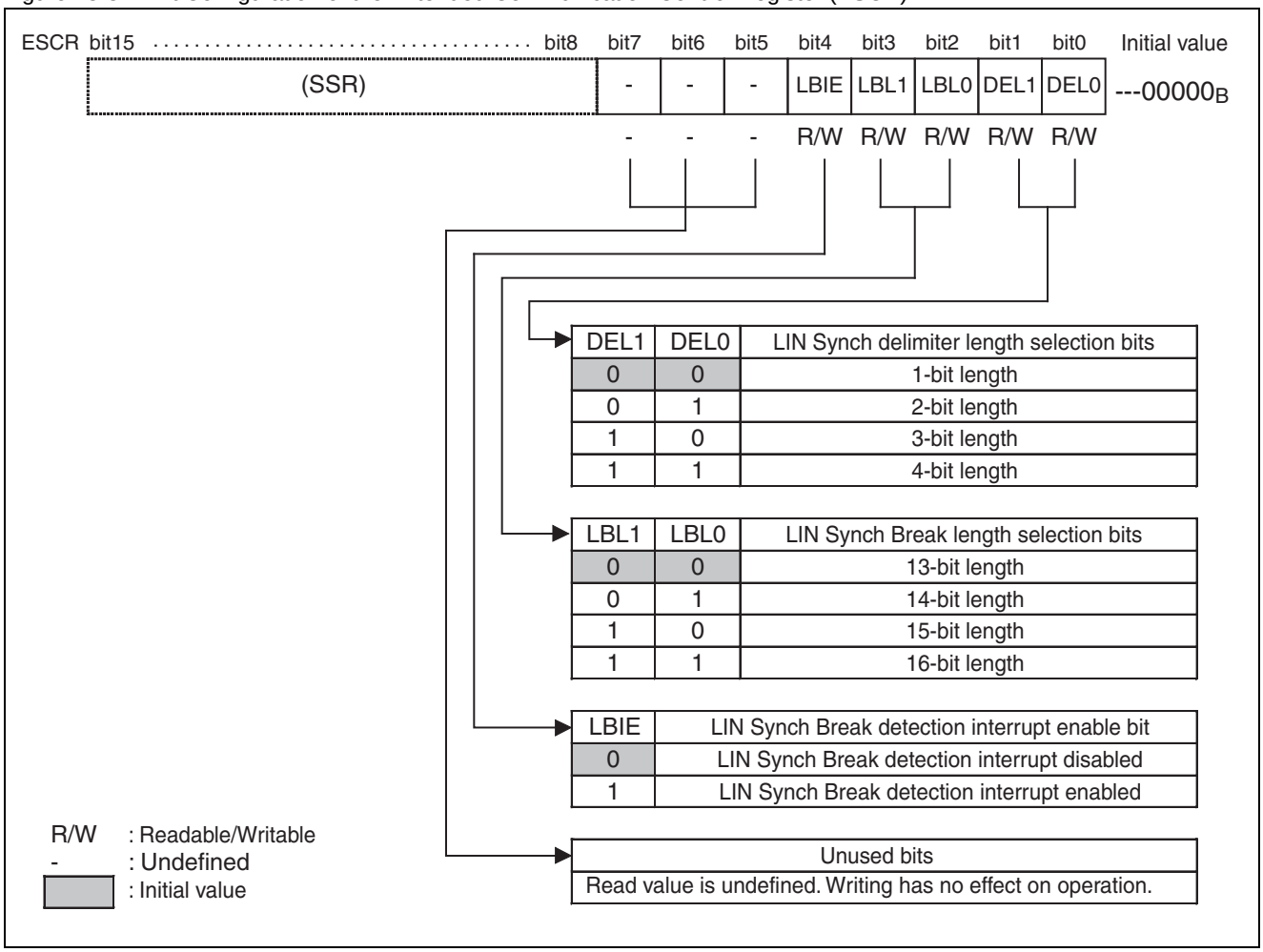


Table 18-32. Function Description of Each Bit of the Extended Communication Control Register (ESCR)

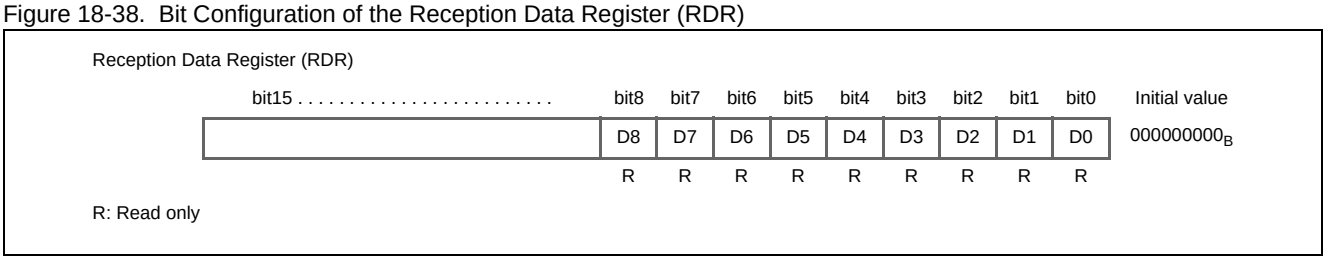
Bit name		Function
bit7 to bit5	Unused bits	When read: The value is undefined. When write: No effect
bit4	LBIE: LIN Synch Break detection interrupt enable bit	This bit enables/disables LIN Synch Break detection interrupts. When the LIN Synch Break detection flag (LBD) is "1", a reception interrupt is generated if the interrupt is enabled (LBIE=1).
bit3, bit2	LBL1/LBL0: LIN synch break length selection bits (operable only in master operation)	■ These bits set LIN Synch Break generation time in bits. ■ Set them before setting the LBR bit of the serial control register (SCR) to "1" (LIN Synch Break transmission). ■ In slave operation, LIN Synch Break detection timing always occurs at the 11th bit regardless of the setting value of these bits. **Note:** This function is operable only in master operation.
bit1, bit0	DEL1/DEL0: LIN synch delimiter length selection bits (operable only in master operation)	■ These bits set LIN Synch delimiter length in bits. ■ Set them before setting the LBR bit of the serial control register (SCR) to "1" (LIN Synch Break transmission). **Note:** This function is operable only in master operation.

Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data register and transmission data register are located at the same address. When reading the address, it functions as a reception data register, and when writing, it functions as a transmission data register.

Figure 18-38 shows the bit configuration of the reception data register (RDR).

Figure 18-38. Bit Configuration of the Reception Data Register (RDR)

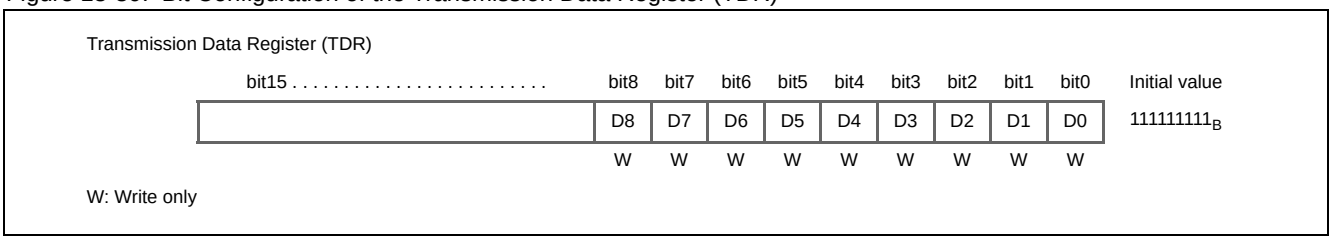


The reception data register (RDR) is a data buffer register for serial data reception.

- A serial data signal transmitted to the serial input pin (UI pin) is converted in the shift register and stored in the reception data register (RDR).
- When reception data is stored into the reception data register (RDR), the reception data full flag bit (SSR:RDRF) is set to “1”. If reception interrupt is being enabled (SSR:RIE=1), a reception interrupt request is generated.
- Read the reception data register (RDR) while the reception data full flag bit (SSR:RDRF) is “1”. The reception data full flag bit (SSR:RDRF) is cleared to “0” automatically when the reception data register (RDR) is read.
- A reception error occurs (any one of SSR:ORE and FRE is “1”), data in the reception data register (RDR) becomes invalid.

Figure 18-39 shows the bit configuration of the transmission data register.

Figure 18-39. Bit Configuration of the Transmission Data Register (TDR)



The transmission data register (TDR) is a data buffer register for serial data transmission.

- When data to be transmitted is written into the transmission data register (TDR) while transmission operations are enabled (SCR:TXE=1), the transmission data is transferred to the shift register for transmission, then converted into serial data, and transmitted from the serial data output pin (UO pin).
- The transmission data empty flag (SSR:TDRE) is cleared to “0” when transmission data is written into the transmission data register (TDR).
- The transmission data empty flag (SSR:TDRE) is set to “1” when transmission data is transferred to the shift register for transmission and transmission starts.
- When the transmission data empty flag (SSR:TDRE) is “1”, the next transmission data can be written. If transmission interrupt is being enabled, a transmission interrupt is generated. Write the next transmission data at the time of transmission interrupt generation or while the transmission data empty flag (SSR:TDRE) is “1”.
- When the transmission data empty flag (SSR:TDRE) is “0”, transmission data cannot be written into the transmission data register (TDR).

Note:

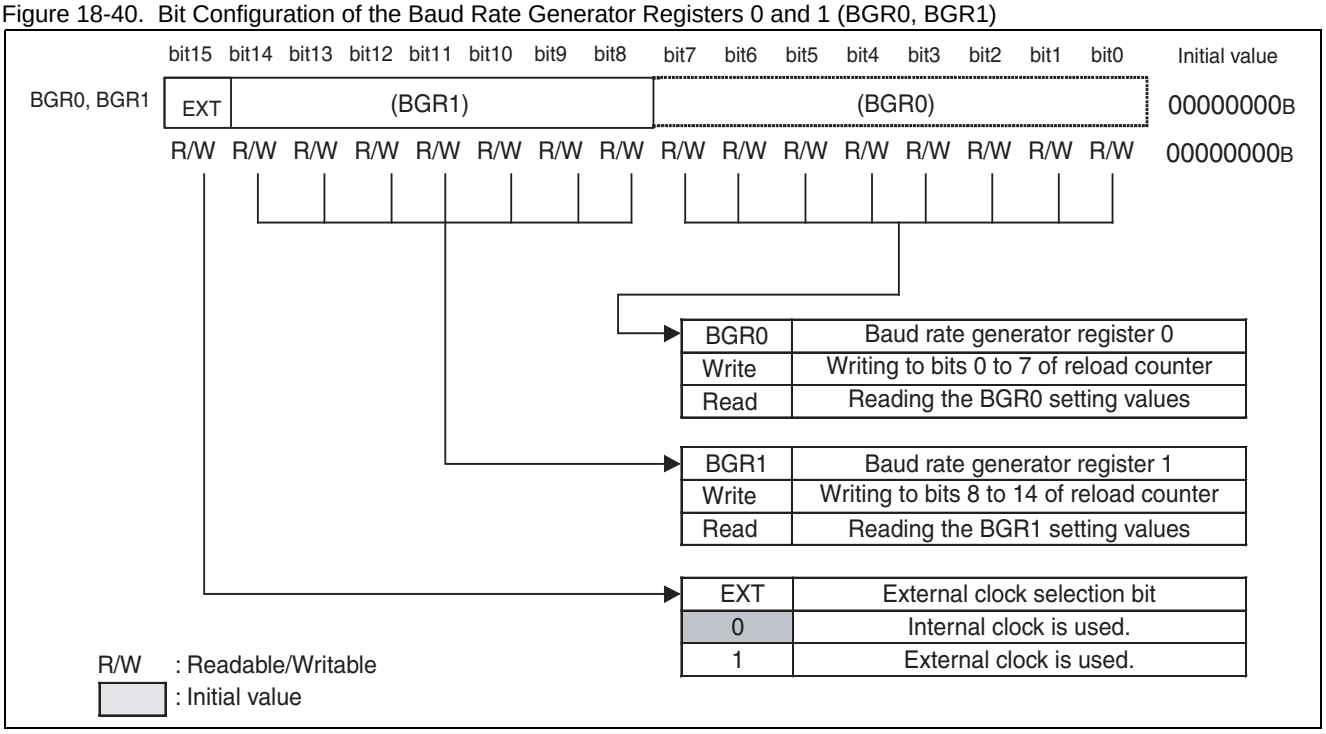
The transmission data register is a write only register, and the reception data register is a read only register. Since the transmission/reception data registers are located at the same address, the write value and the read value are different. Therefore, the read-modify-write (RMW) instruction such as INC/DEC instructions cannot be used.

Baud Rate Generator Registers 0, 1 (BGR0, BGR1)

The baud rate generator registers 0 and 1 (BGR0, BGR1) set the division ratio for the serial clock. They can also select an external clock as a clock source for the reload counter.

Figure 18-40 shows the bit configuration of the baud rate generator registers 0 and 1 (BGR0, BGR1).

Figure 18-40. Bit Configuration of the Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)



- The baud rate generator registers set the division ratio of the serial clock.
- The BGR1 corresponds to the upper bits and BGR0 corresponds to the lower bits, and writing of reload values for counting and reading of the BGR0, BGR1 setting values are allowed.
- When writing reload values to the baud rate generator registers 0 and 1 (BGR0, BGR1), the reload counter starts counting.
- Bit 15 is the EXT bit and it selects an internal clock or an external clock used as a clock source for the reload counter. If EXT=0 is set, it selects an internal clock. If EXT=1 is set, it selects an external clock.

Notes:

- When writing into the baud rate generator registers 0 and 1 (BGR0, BGR1), use 16-bit access.
- When a setting value in the baud rate generator registers 0 and 1 (BGR0, BGR1) is changed, a new setting value is reloaded after the counter value becomes "0000_H". Therefore, if you want to enable a new setting value immediately, execute a programmable clear (UPCL) after changing the BGR0, BGR1 setting value.
- If a reload value is an even number, the "L" width of the serial clock becomes longer than the "H" width by one machine clock cycle. If it is an odd number, the "H" width and "L" width of the serial clock become the same.
- Set a reload value to 3 or larger. Data, however, may not be received properly depending on the baud rate error and the reload value setting.
- To switch to the external clock setting (EXT=1) while the baud rate generator is operating, first write 0 into the baud rate generators 0 and 1 (BGR0, BGR1), execute a programmable clear (UPCL), and then set the external clock (EXT=1).

18.6.3 Interrupts of the LIN-UART

The LIN-UART uses both transmission and reception interrupts. An interrupt request can be generated by one of the following sources:

- When reception data is set into the reception data register (RDR), or when a reception error occurs
- When transmission data is transferred from the transmission data register (TDR) to the shift register for transmission and transmission starts
- Transmission bus idle (no transmission operation)
- LIN Synch Break detection

Interrupts of the LIN-UART

The interrupt control bits and the interrupt sources of the LIN-UART are shown in Table 18-33.

Table 18-33. Interrupt Control Bits and Interrupt Sources of the LIN-UART

Interrupt type	Interrupt request flag bit	Flag register	Interrupt source	Interrupt source enable bit	How to clear the interrupt request flag
Reception	RDRF	SSR	1-byte reception	SCR:RIE	Read the reception data (RDR).
	ORE	SSR	Overrun error		Write “1” into the reception error flag clear bit (SSR:REC).
	FRE	SSR	Framing error		
Transmission	TDRE	SSR	Transmission register empty	SCR:TIE	Write into the transmission data (TDR).
	TBI	SSR	No transmission operation	SCR:TBIE	Write into the transmission data (TDR).
Status	LBD	SSR	LIN Synch Break detection	ESCR:LBIE	Write “0” into the SSR:LBD bit.
Input capture	ICP0	ICS0	1st falling edge in LIN Synch Field	ICS0:ICE0	Disable ICP0.
	ICP0	ICS0	5th falling edge in LIN Synch Field		

Note:

Wait until TDRE bit becomes “0” before setting TIE bit to “1”.

Reception Interrupt Generation and the Flag Set Timing

Interrupts during reception are caused by completion of reception (SSR:RDRF), occurrence of a reception error (SSR:ORE, FRE), and LIN Synch Break detection.

Reception data is stored in the reception data register (RDR) when the first stop bit is detected. When the reception is completed (SSR:RDRF=1) or when a reception error occurs (SSR:ORE, FRE=1), each flag is set. If reception interrupt is being enabled (SSR:RIE=1) at that time, then a reception interrupt is generated.

Note:

If any reception error occurs, data in the reception data register (RDR) becomes invalid.

Figure 18-41. RDRF (Reception Data Full) Flag Bit Set Timing

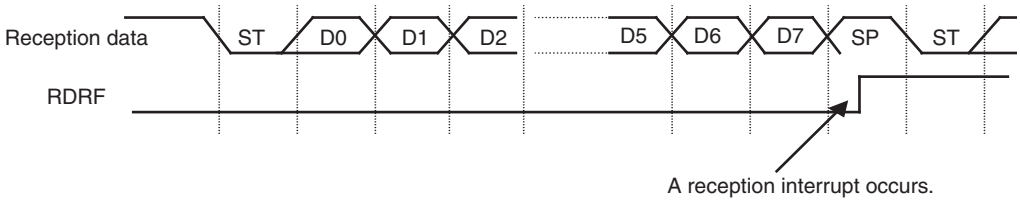
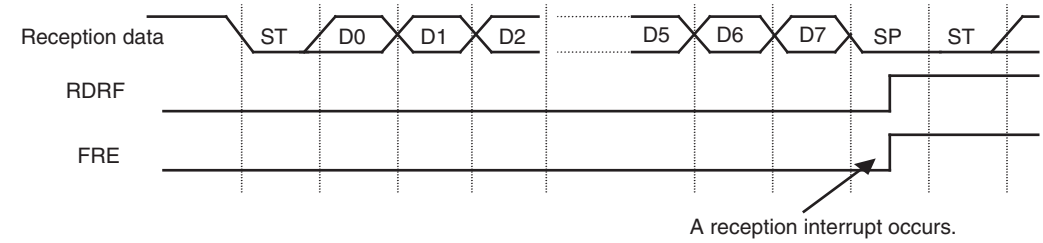
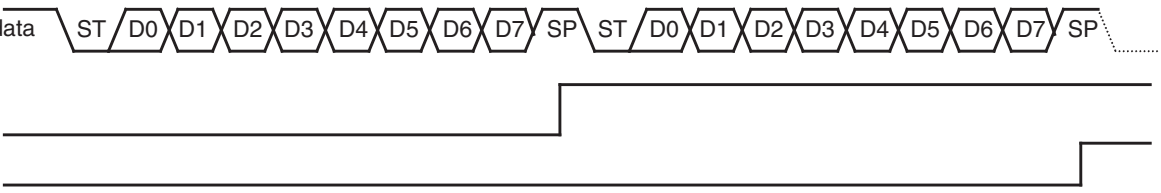


Figure 18-42. FRE (Framing Error) Flag Bit Set Timing



Notes:
- When the first stop bit is at "L" level, a framing error occurs.
- Although "1" is set in RDRF and data is received even when a framing error occurs, the reception data is invalid.

Figure 18-43. ORE (Overrun Error) Flag Bit Set Timing

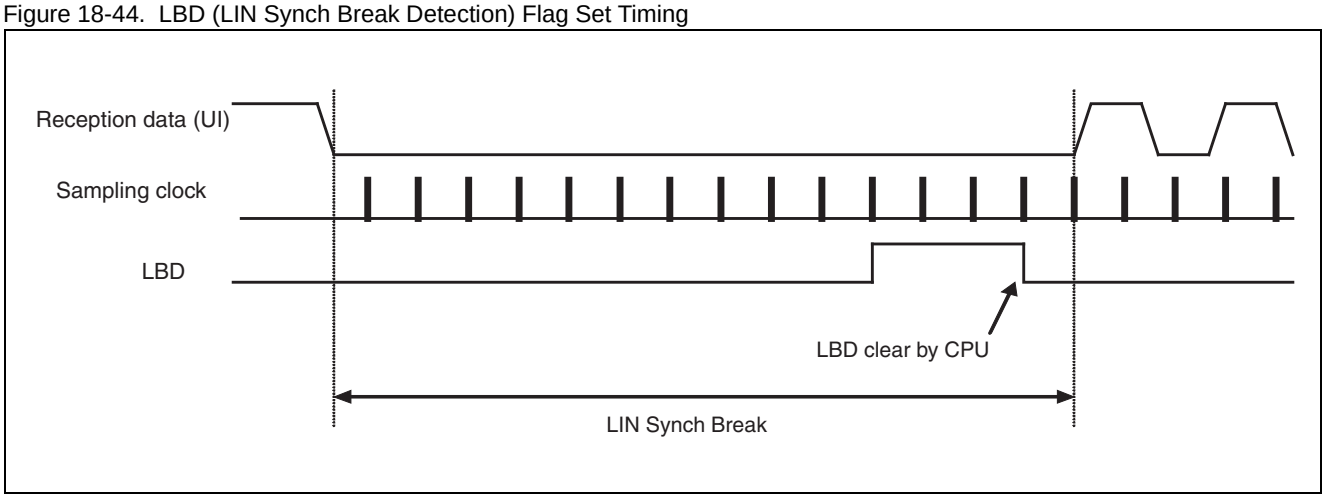


Note:
An overrun error occurs if the next data is transferred before the reception data is read (RDRF=1).

LIN Synch Break Detection Flag (LBD) Set Timing

When “0” is input to the serial input (UI) for 11-bit width or over during slave operation (SCR:MS=1), the LBD bit is set to “1”. A reception interrupt is generated if LIN Synch Break interrupt is being enabled (ESCR:LBIE=1) at this time.

Figure 18-44. LBD (LIN Synch Break Detection) Flag Set Timing



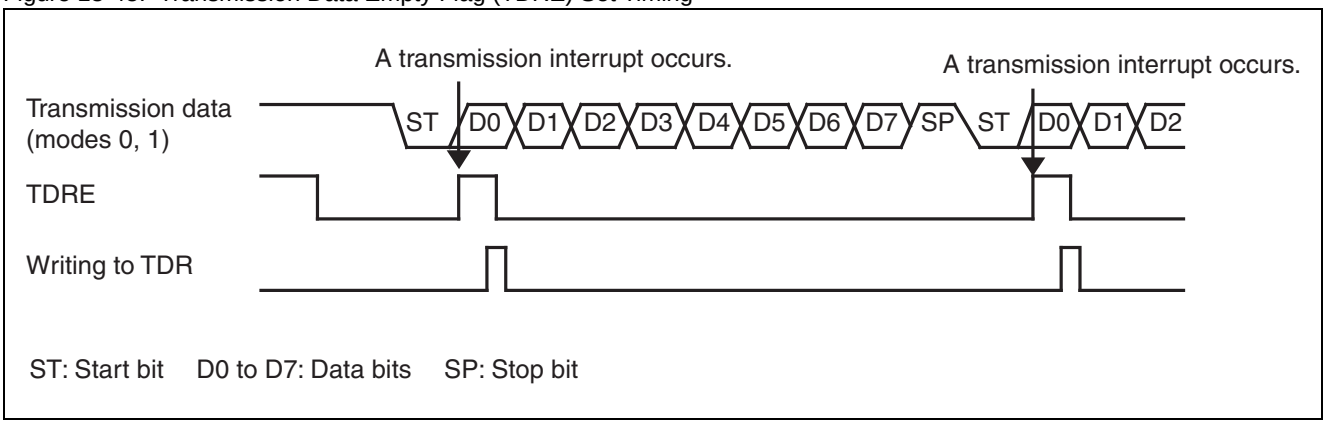
Transmission Interrupt Generation and the Flag Set Timing

Interrupts during transmission are caused when transmission data is transferred from the transmission data register (TDR) to the shift register for transmission (SSR:TDRE=1) and transmission starts, and when no transmission operation is running (SSR:TBI=1).

Transmission data empty flag (TDRE) set timing

When data written to the transmission data register (TDR) is transferred to the transmission shift register, the state becomes ready for the next data to be written (SSR:TDRE=1). If transmission interrupt is being enabled (SCR:TIE=1) at that time, then a transmission interrupt is generated. Since TDRE bit is a read only bit, writing data into the transmission data register (TDR) clears the bit to “0”.

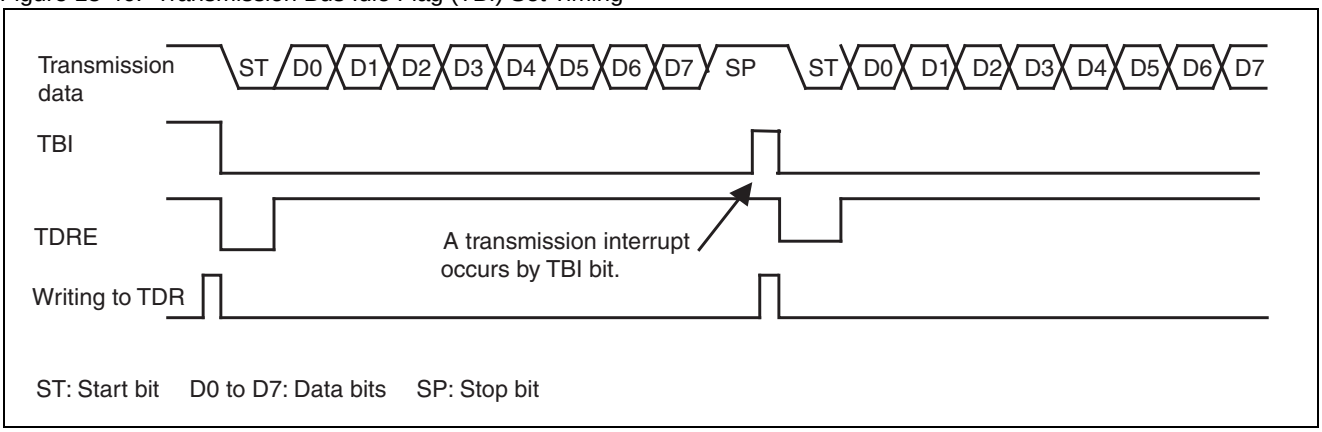
Figure 18-45. Transmission Data Empty Flag (TDRE) Set Timing



Transmission bus idle flag (TBI) set timing

When the transmission data register is empty (TDRE=1) and no transmission operation is running, SSR:TBI bit is set to “1”. If transmission bus idle interrupt is being enabled (SCR:TBIE=1) at that time, then a transmission interrupt is generated. When transmission data is set in the transmission data register (TDR), the TBI bit and the transmission interrupt request are cleared.

Figure 18-46. Transmission Bus Idle Flag (TBI) Set Timing



18.6.4 Dedicated Baud Rate Generator

One of the followings can be selected for the LIN-UART transmission/reception clock source:

- **Dedicated baud rate generator (reload counter)**
- **External clock input to a baud rate generator (reload counter)**

LIN-UART Baud Rate

One of the following two types of baud rate can be selected:

Baud rate determined by dividing an internal clock using the dedicated baud rate generator (reload counter)

The LIN-UART has two internal reload counters and each of them corresponds to transmission/reception serial clock. The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 0 and 1 (BGR0, BGR1).

The reload counter divides an internal clock by a setting value.

Select to use internal clock (SMR:EXT=0) for the clock source setting.

Baud rate determined by dividing an external clock using the dedicated baud rate generator (reload counter)

An external clock is used as a clock source for the reload counter.

The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 0 and 1 (BGR0, BGR1).

The reload counter divides an external clock by a setting value.

Select to use external clock and baud rate generator clock (SMR:EXT=1) for the clock source setting.

This mode is designed to be used in the case where an oscillator of special frequency is used by dividing it.

Notes:

- Set the external clock (EXT=1) while the reload counters are being stopped (BGR0, BGR1=0000_H).
- If external clock is set (EXT=1), two or more machine clock cycles are required for the “H” width and “L” width of the external clock.

Baud Rate Setting

The following describes the baud rate setting. It also shows a calculation result of a serial clock frequency.

Calculation of Baud Rate

Two 15-bit reload counters are set using the baud rate generator registers 1 and 0 (BGR0, BGR1).

Calculation formula of baud rate is shown below.

1. Reload value

$V = \phi/b - 1$

V: Reload value b: Baud rate ϕ: Machine clock, external clock frequency

2. Example of the calculation

When the machine clock is set to 16MHz, an internal clock is used, and the baud rate is set to 19200 bps, the reload value can be calculated as follows:

Reload value:

$V = (16 \times 1000000)/19200 - 1 = 832$

Therefore, the baud rate is:

$b = (16 \times 1000000)/(832 + 1) = 19208$ bps

3. Baud rate error

The baud rate error can be calculated using the following formula:

Error (%) = (calculated value -desired value)/desired value x 100

(Example) When the machine clock is set to 20MHz and the desired baud rate is set to 153600 bps:

Reload value = $(20 \times 1000000)/153600 - 1 = 129$

Baud rate (calculated value) = $(20 \times 1000000)/(129 + 1) = 153846$ (bps)

Error (%) = $(153846 - 153600)/153600 \times 100 = 0.16$ (%)

Notes:

- The reload counter will be stopped when the reload value is set to “0”.
- If the reload value is an even number, the “L” width of the serial clock becomes longer than the “H” width by one machine clock cycle. If it is an odd number, the “H” width and “L” width of the serial clock become the same.
- Set the reload value to 3 or larger. Data, however, may not be received properly depending on the baud rate error and the reload value setting.

Reload Values and Baud Rates Corresponding to Each Machine Clock Frequency

Table 18-34. Reload Values and Baud Rates

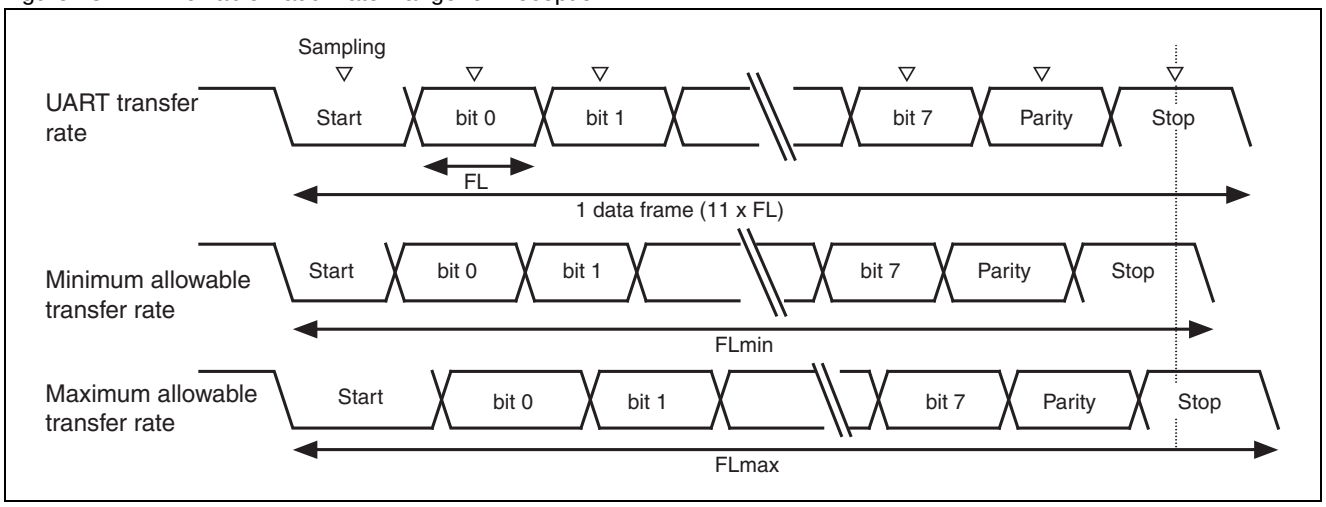
Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz		32MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	-	-	-	-	-	-	-	-	-	-	-	0
6M	-	-	-	-	-	-	-	-	-	-	-	-
5M	-	-	-	-	-	-	-	-	-	-	-	-
4M	-	-	-	-	-	-	4	0	5	0	7	0
2.5M	-	-	3	0	-	-	-	-	-	-	-	-
2M	3	0	4	0	7	0	9	0	11	0	15	0
1M	7	0	9	0	15	0	19	0	23	0	31	0
500000	15	0	19	0	31	0	39	0	47	0	63	0
460800	-	-	-	-	-	-	-	-	51	-0.16	-	-
250000	31	0	39	0	63	0	79	0	95	0	127	0
230400	-	-	-	-	-	-	-	-	103	-0.16	-	-
153600	51	-0.16	64	-0.16	103	-0.16	129	-0.16	155	-0.16	207	-0.16
125000	63	0	79	0	127	0	159	0	191	0	255	0
115200	68	-0.64	86	0.22	138	0.08	173	0.22	207	-0.16	277	0.08
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	311	-0.16	416	0.08
57600	138	0.08	173	0.22	277	0.08	346	-0.16	416	0.08	555	0.08
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	624	0	832	-0.04
28800	277	0.08	346	< 0.01	554	-0.01	693	-0.06	832	-0.03	1110	-0.01
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1249	0	1666	0.02
10417	767	< 0.01	959	< 0.01	1535	< 0.01	1919	< 0.01	2303	< 0.01	3071	< 0.01
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	2499	0	3332	-0.01
7200	1110	< 0.01	1388	< 0.01	2221	< 0.01	2777	< 0.01	3332	< 0.01	4443	-0.01
4800	1666	0.02	2082	-0.02	3332	< 0.01	4166	< 0.01	4999	0	6666	< 0.01
2400	3332	< 0.01	4166	< 0.01	6666	< 0.01	8332	< 0.01	9999	0	13332	< -0.01
1200	6666	< 0.01	8334	0.02	13332	< 0.01	16666	< 0.01	19999	0	26666	< 0.01
600	13332	< 0.01	16666	< 0.01	26666	< 0.01	-	-	-	-	-	-
300	26666	26666	< 0.01	-	-	-	-	-	-	-	-	-

- Value : Setting value in BGR0, BGR1 register
- ERR : Baud rate error (%)

Allowable Baud Rate Range for Reception

The allowable baud rate error range at the transmission destination during reception is shown below. The baud rate error during reception must be set to within the allowable error range using the calculation formula shown below.

Figure 18-47. Allowable Baud Rate Range for Reception



As shown in Figure 18-37, after a start bit is detected, the sampling timing of reception data is determined by a counter set in BGR0, BGR1 register. If the last data (stop bit) is reached by this sampling timing, the data can be received properly.

If it is applied to an 11-bit reception logically, the rates can be calculated as follows:

If a sampling timing margin is set to one machine clock (ϕ), the minimum allowable transfer rate (FLmin) can be calculated as follows:

$FLmin = (11bits \times (V + 1) - (V + 1)/2 + 2)/\phi = (21V + 25)/2\phi$ (s)

V: Reload value ϕ: Machine clock

Therefore, the receivable maximum baud rate (BGmax) at the transmission destination can be calculated as follows:

$BGmax = 11/FLmin = 22\phi/(21V + 25)$ (bps) V: Reload value ϕ: Machine clock

In a similar way, the maximum allowable transfer rate (FLmax) can be calculated as follows:

$FLmax = (11bits \times (V + 1) + (V + 1)/2 - 2)/\phi = (23V + 19)/2\phi$ (s)

V: Reload value ϕ: Machine clock

Thus, the receivable minimum baud rate (BGmin) at the transmission destination can be calculated as follows:

$BGmin = 11/FLmax = 22\phi/(23V + 19)$ (bps) V: Reload value ϕ: Machine clock

The following allowable baud rate errors between UART and the transmission destination are calculated using the above calculation formulas for minimum/maximum baud rate values.

Reload value (V)	Maximum allowable baud rate error	Minimum allowable baud rate error
3	0%	0
10	+2.98%	-2.81%
50	+4.37%	-4.02%
100	+4.56%	-4.18%
200	+4.66%	-4.26%
32767	+4.76%	-4.35%

Note:

The reception accuracy depends on the number of bit in one frame, machine clock, and the reload value. The higher the machine clock and the division ratio, the higher the reception accuracy will be.

External Clock

If “1” is written to the EXT bit of the baud rate generator register (BGR), an external clock is divided by the baud rate generator.

Note:

An external clock signal synchronizes with an internal clock in UART. Therefore, if an external clock that cannot synchronize is used, the operation will become unstable.

Function of Reload Counter

Two types of reload counter, a transmission reload counter and a reception reload counter, are available and each of them functions as a dedicated baud rate generator. It consists of a 15-bit register corresponding to reload values and it generates transmission/reception clock from an external clock or an internal clock.

Count Start

The reload counter starts counting when a reload value is written into the baud rate generator registers 0 and 1 (BGR0, BGR1).

Restart

The reload counter restarts under the following conditions:

- For both transmission/reception reload counters
  Programmable reset (SCR:UPCL bit)
- For reception reload counter
  Detection of a start bit falling edge in asynchronous mode

18.6.5 Operation of the LIN-UART

The LIN-UART operates with master/slave bidirectional LIN communication.

Operation of the LIN-UART (Master Device Operation)

Device selection

To operate as a master device, set SCR:MS bit to “0”.

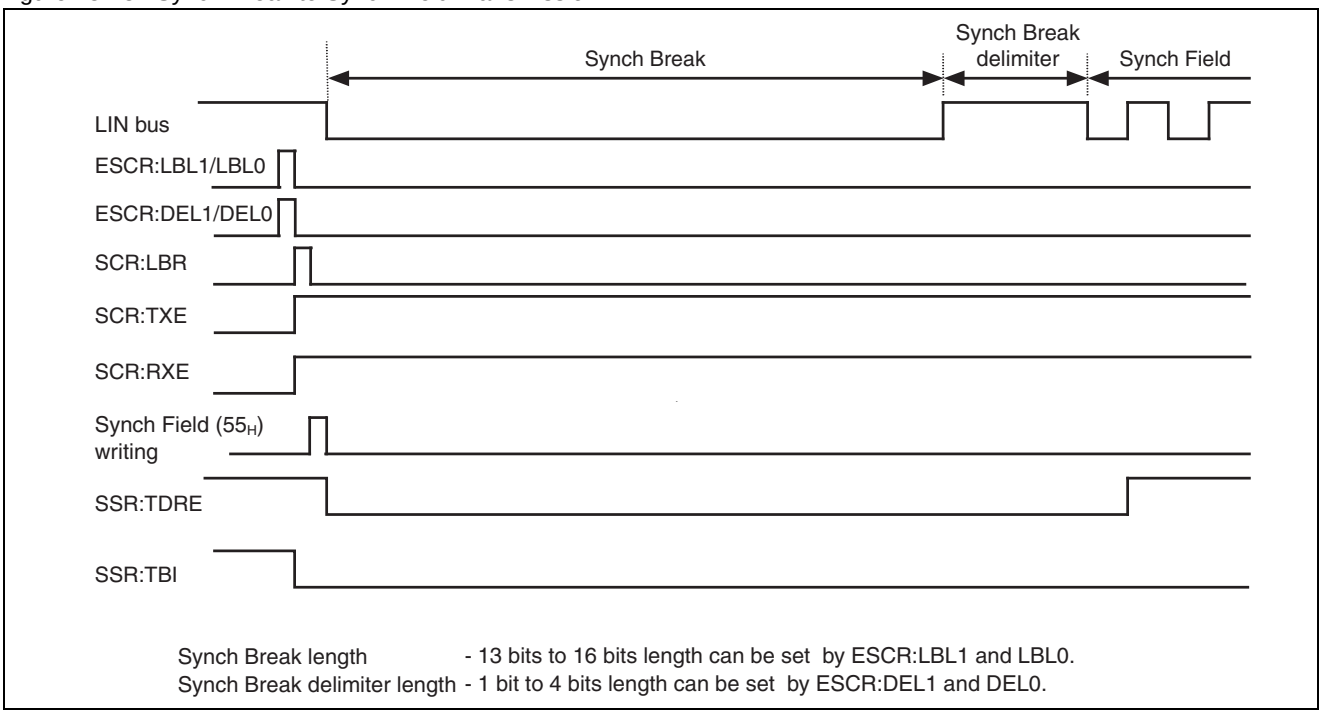
Synch Break transmission to Synch Field transmission

- Select Synch Break length (ESCR:LBL1, LBL0) and Synch Break delimiter length (ESCR:DEL1, DEL0).
- When transmission operation is enabled (SCR:TXE=1) and the SCR:LBR bit (LIN Synch Break setting bit) is set to "1", Synch Break is transmitted.
- Synch Field is transmitted by writing 55_H to the transmission data register (TDR).

Notes:

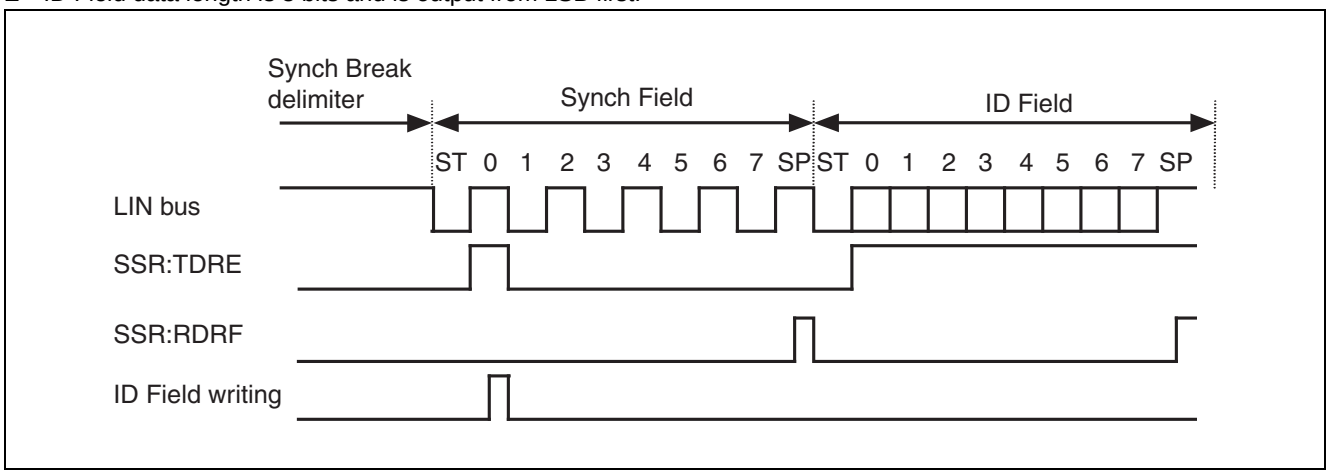
- After setting the SCR:LBR bit (LIN Synch Break setting bit) to “1”, set the transmission data register (TDR) to 55_H.
- Even if the SCR:RXE bit (reception operation enable bit) is set to “1”, no reception operation is performed during Synch Break.

Figure 18-48. Synch Break to Synch Field Transmission



Synch Field transmission to ID Field transmission

- When the first bit of Synch Field (55_H) is transmitted, SSR:TDRE (transmission data empty) bit is set to “1”. If transmission interrupt is being enabled (SCR:TIE=1) at this time, then a transmission interrupt is generated.
- When a transmission interrupt occurs, ID Field can be written to the transmission data register (TDR).
- When a reception interrupt occurs, transmission data is compared with reception data to check whether any errors occurred.
- ID Field data length is 8 bits and is output from LSB first.



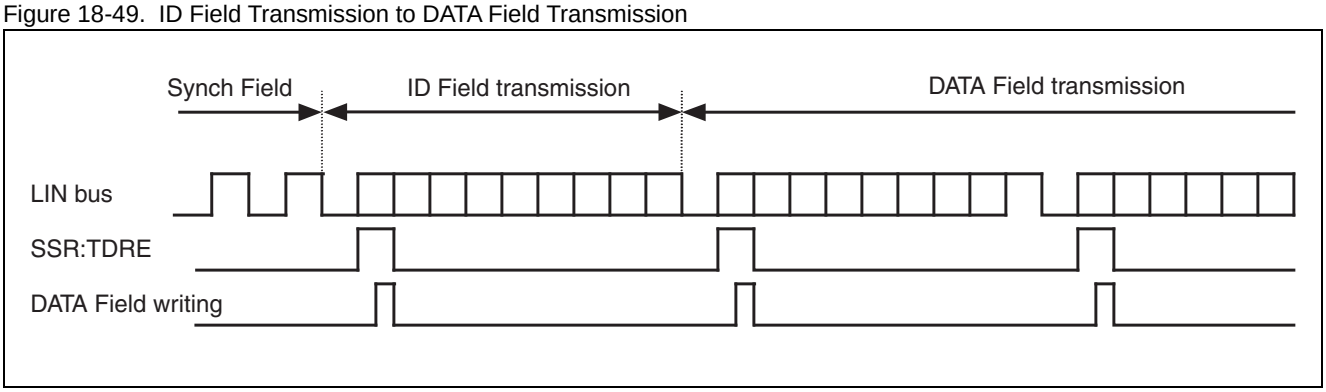
ID Field transmission to DATA Field transmission/reception

- Select whether to transmit DATA Field to a slave device or to receive it.

(To transmit DATA Field)

- When the first bit of ID Field is transmitted, SSR:TDRE=1 is set. DATA Field writing is enabled at this time.

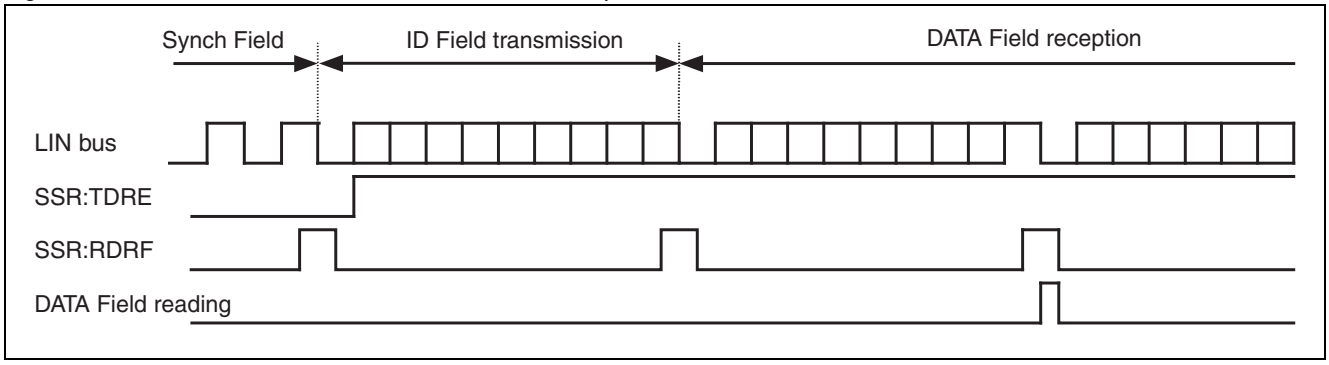
Figure 18-49. ID Field Transmission to DATA Field Transmission



(To receive DATA Field)

- When the first bit of ID Field is transmitted, SSR:TDRE=1 is set, however, do not write transmission data. In addition, set to disable transmission interrupt (SCR:TIE=0).
- When DATA Field is received, SSR:RDRF is set to “1”. A reception interrupt occurs if reception interrupt is being enabled (SSR:RIE=1) at this time.

Figure 18-50. ID Field Transmission to DATA Field Reception



Master device operation timing chart

Figure 18-51. LIN Bus Timing (DATA Field Transmission)

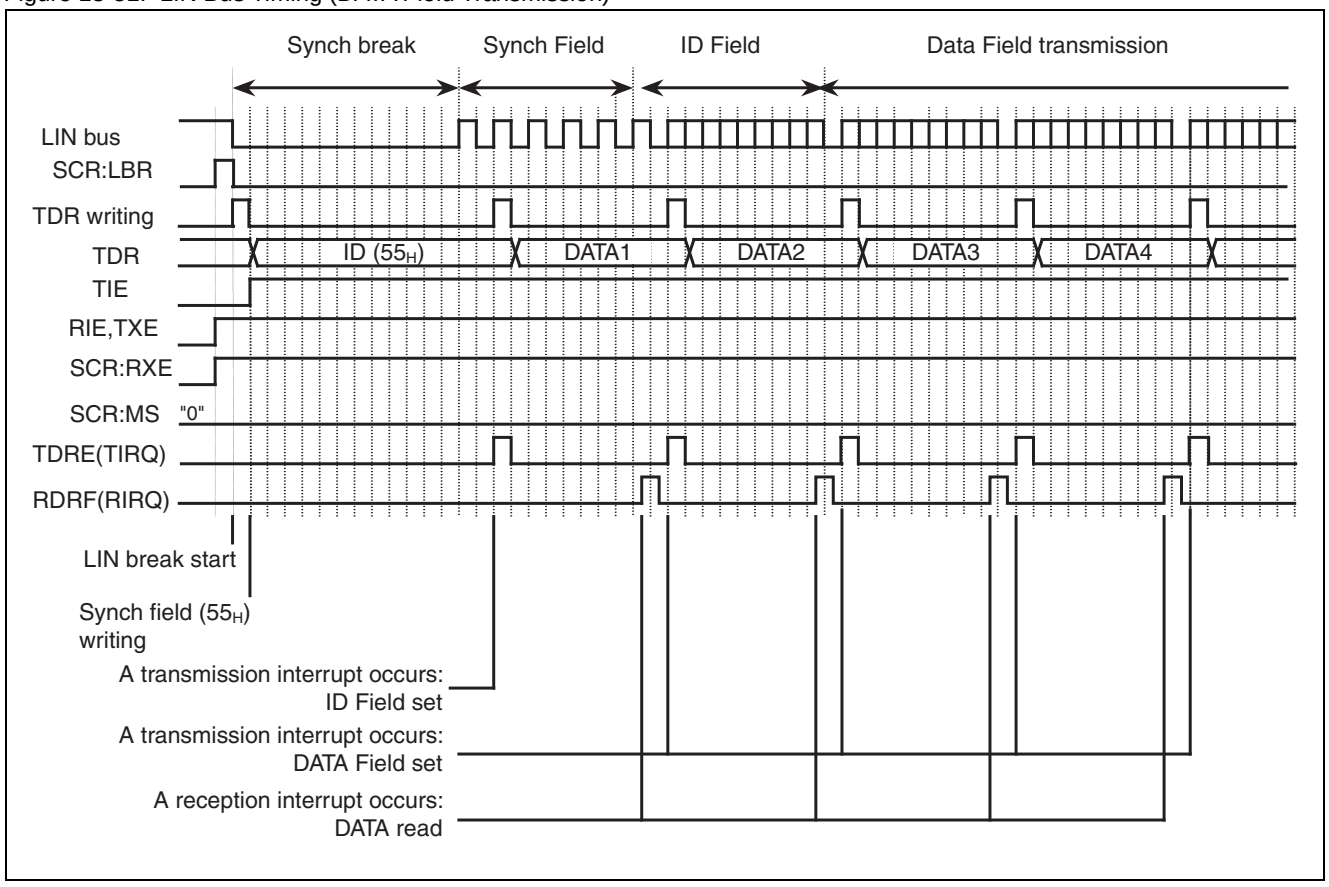
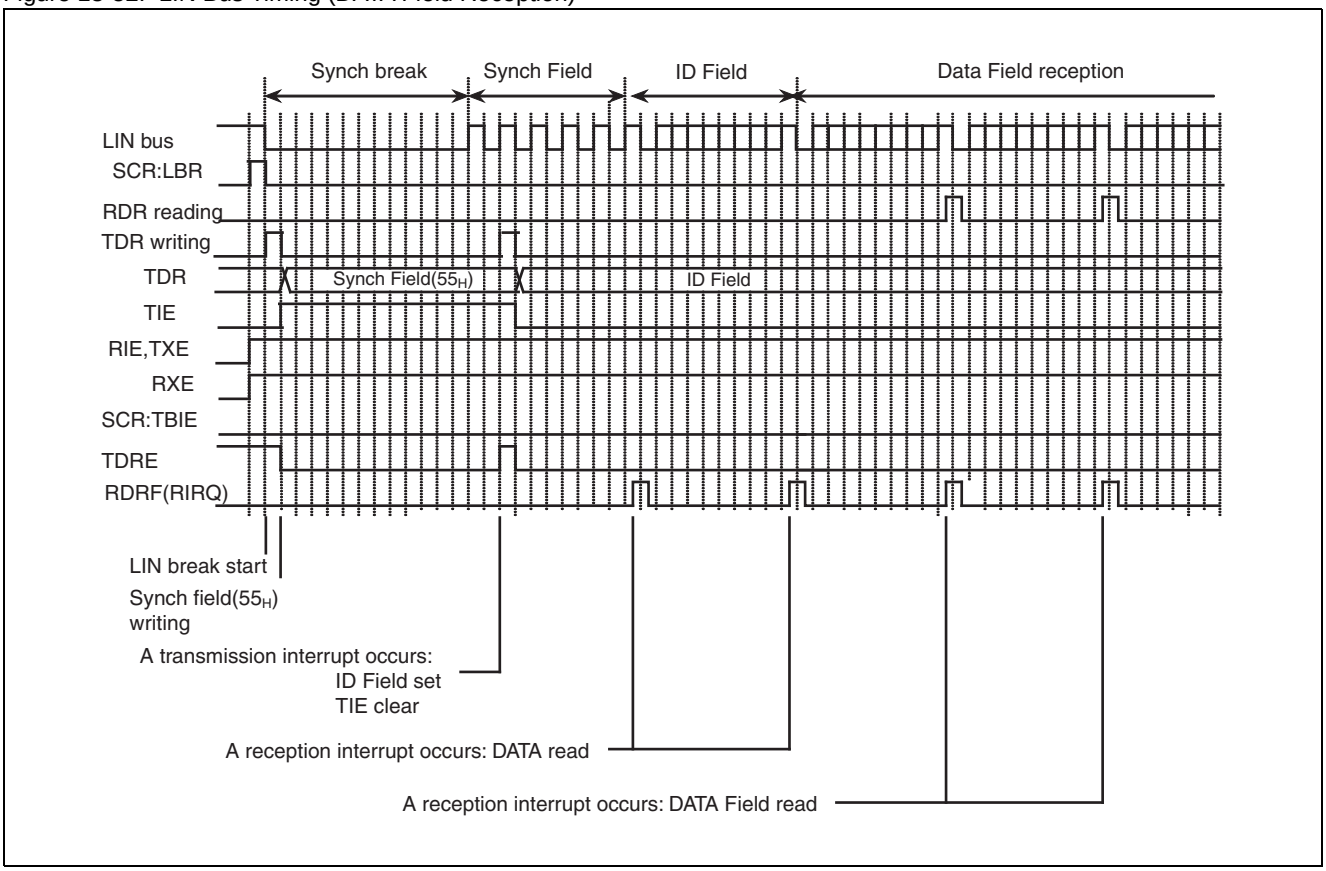


Figure 18-52. LIN Bus Timing (DATA Field Reception)



Operation of the LIN-UART (Slave Device Operation)

Device selection

To operate as a slave device, set SCR:MS bit to “1”.

Synch Break Field reception to Synch Field reception

(1). When Synch Break is input, the Synch Break is detected (SSR:LBD=1) at the 11th bit. A reception interrupt occurs if ESCR:LBIE bit is being set to “1” at this time.

(2). Now, enable ICU interrupt and set to both edge detection.

(3). When the LIN-UART detects the first falling edge in Synch field, it sets the internal signal (LSYN) input to the ICU to “H” and starts the ICU. This internal signal (LSYN) becomes “L” at the 5th falling edge.

(4). “H” time for the internal signal (LSYN) to be input to the ICU is set to the value of the baud rate multiplied by 8. The baud rate setting value is as follows:

If a free-run timer has not been overflowed:

$$\text{BGR value} = (b - a) \times Fe/(8 \times \phi) - 1$$

If a free-run timer has been overflowed:

$$\text{BGR value} = (Max + b - a) \times Fe/(8 \times \phi) - 1$$

Max :Maximum value for free-run timer

a :ICU data register value after the first interrupt

b :ICU data register value after the second interrupt

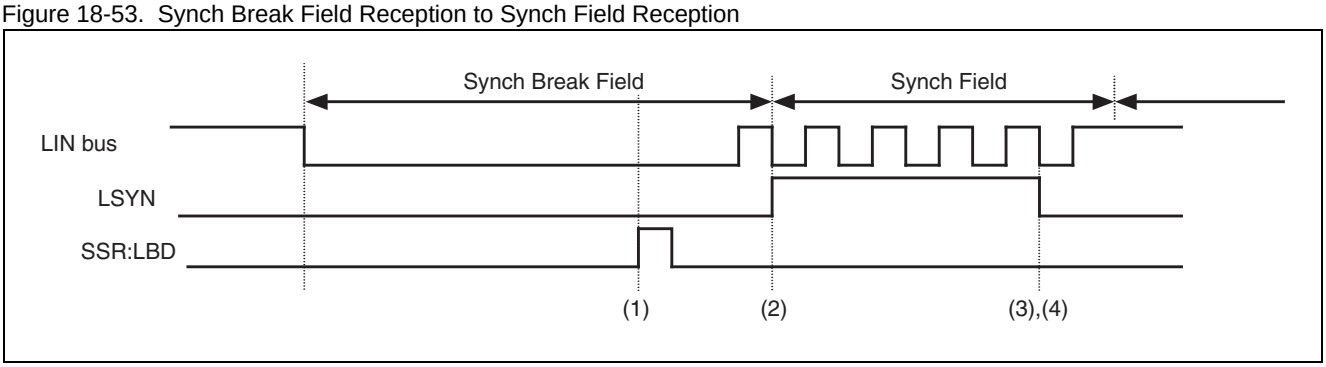
ϕ :Machine clock frequency (MHz)

Fe :External clock frequency (MHz). When an internal clock is used (EXT=0), calculate it as Fe=ϕ.

Note:

Disable reception (SCR:RXE=0) during Synch Break and Synch Field.

Figure 18-53. Synch Break Field Reception to Synch Field Reception



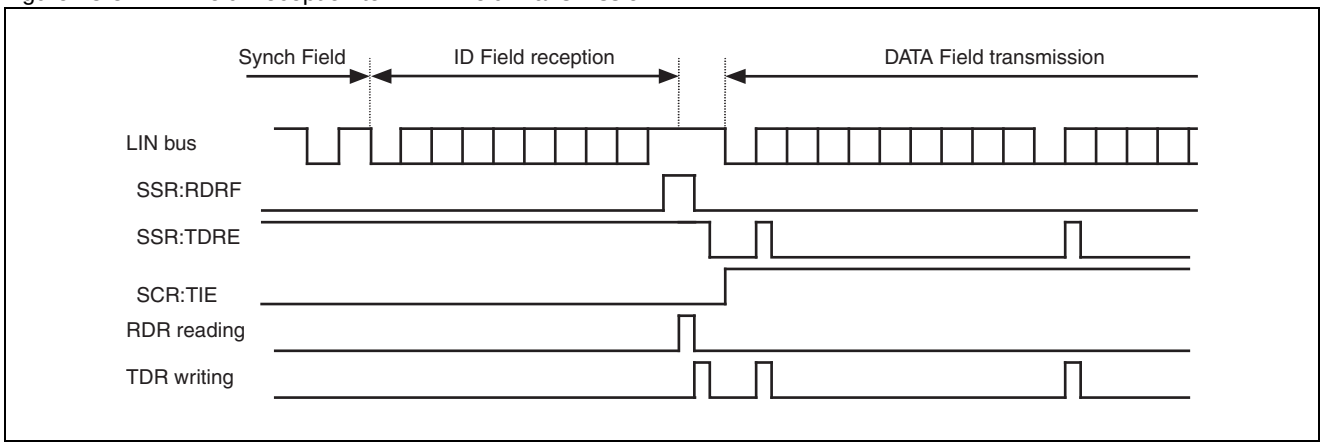
ID Field reception to DATA Field transmission/reception

- Select whether to transmit DATA Field to a master device or to receive it after receiving ID Field.

(To transmit DATA Field)

- Write data to the transmission data register (TDR) after receiving ID Field. Set transmission interrupt to enabled (SCR:TIE=1) at this time.

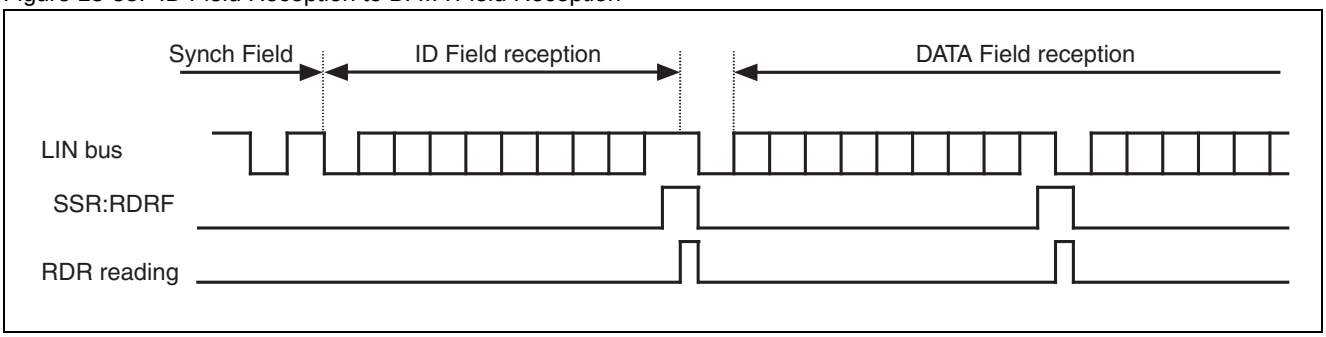
Figure 18-54. ID Field Reception to DATA Field Transmission



(To receive DATA Field)

- SSR:RDRF is set to “1” for each DATA Field reception. A reception interrupt occurs if reception interrupt is being enabled (SCR:RDRF=1) at this time.

Figure 18-55. ID Field Reception to DATA Field Reception



Slave device operation timing chart

Figure 18-56. LIN Bus Timing (DATA Field Transmission)

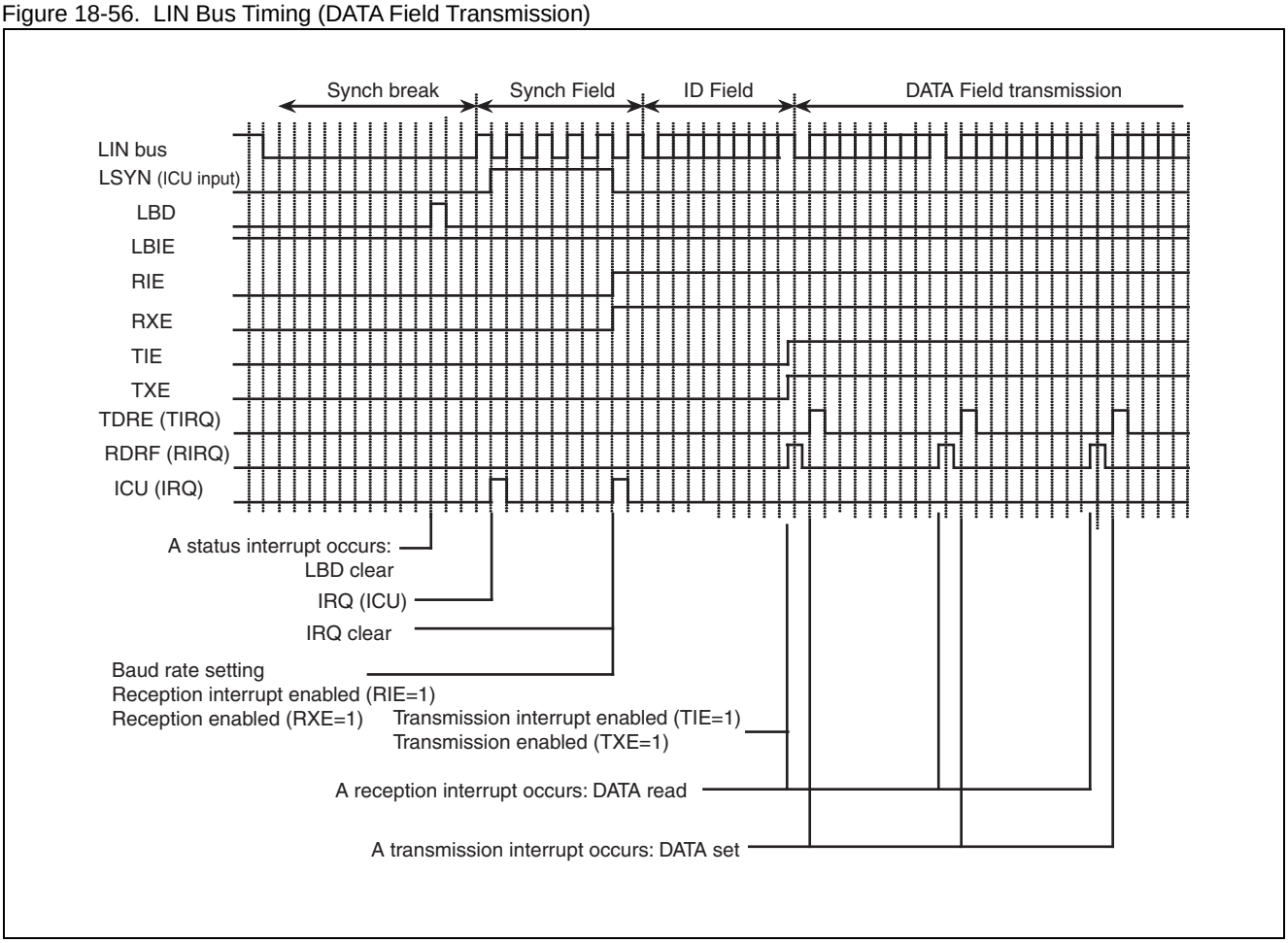
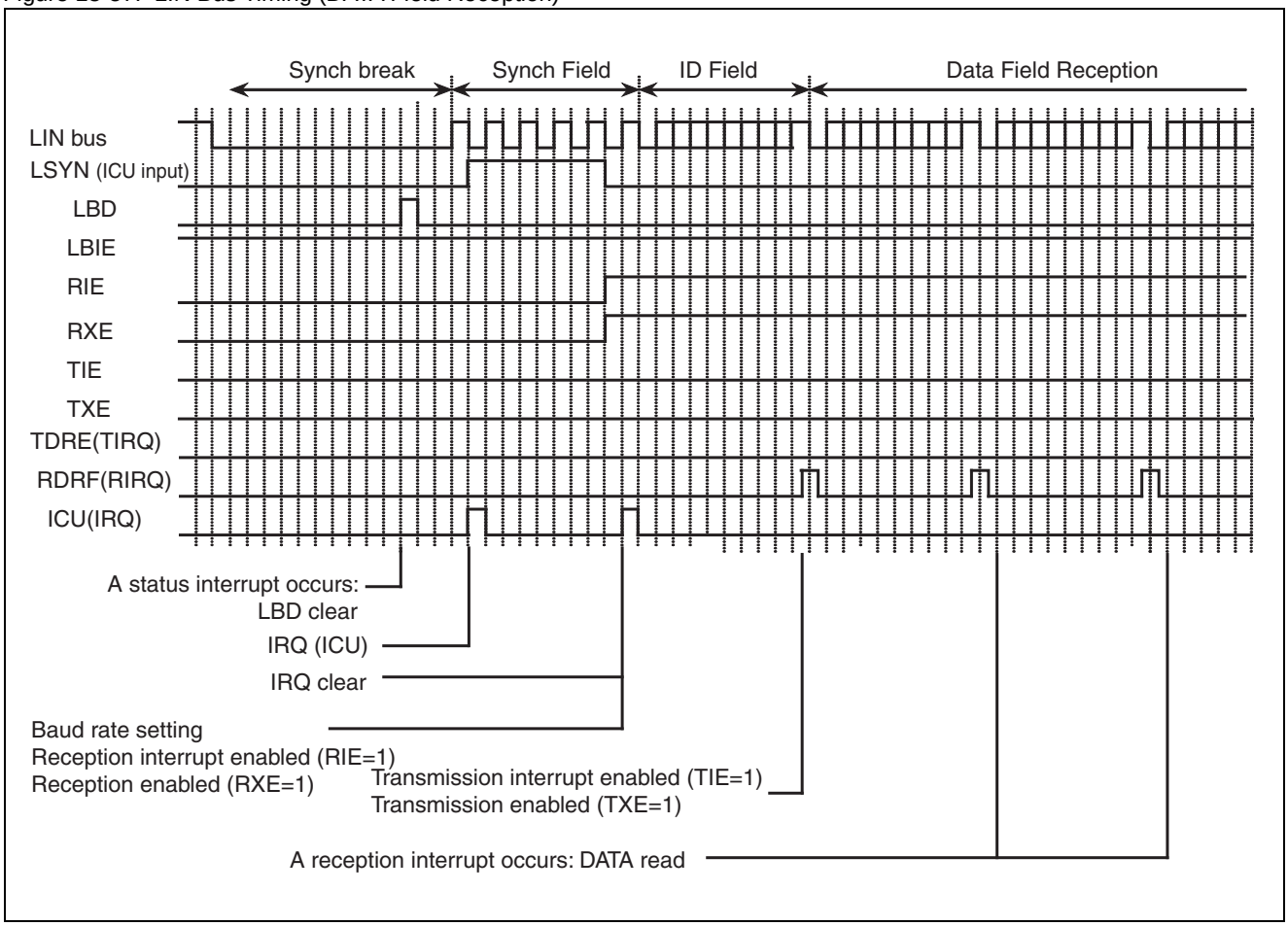


Figure 18-57. LIN Bus Timing (DATA Field Reception)



18.6.6 Setting Procedure and the Program Flow for Operation Mode 3 (LIN Communication Mode)

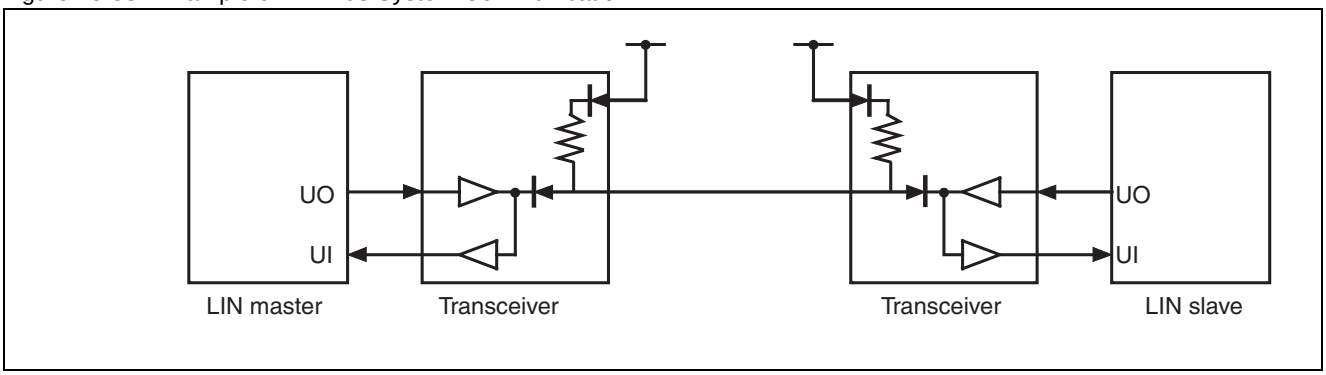
Operation mode 3 (LIN communication mode) can be used in LIN master system or in LIN slave system.

Register Settings

Connection between CPUs

Figure 18-58 shows an example of the LIN bus system communication. The multi-function serial interface can operate as LIN master or LIN slave.

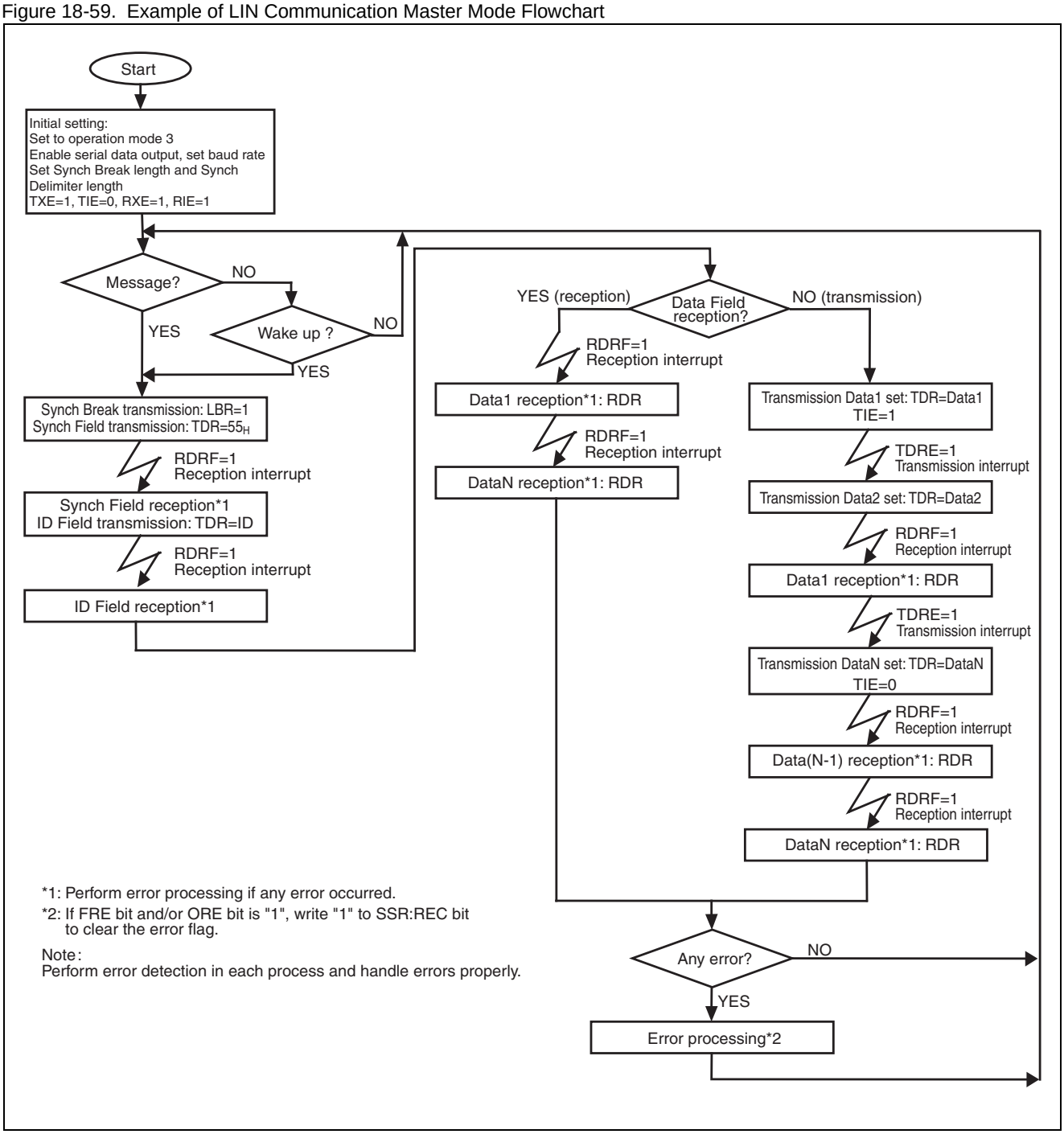
Figure 18-58. Example of LIN Bus System Communication



Flowchart Examples

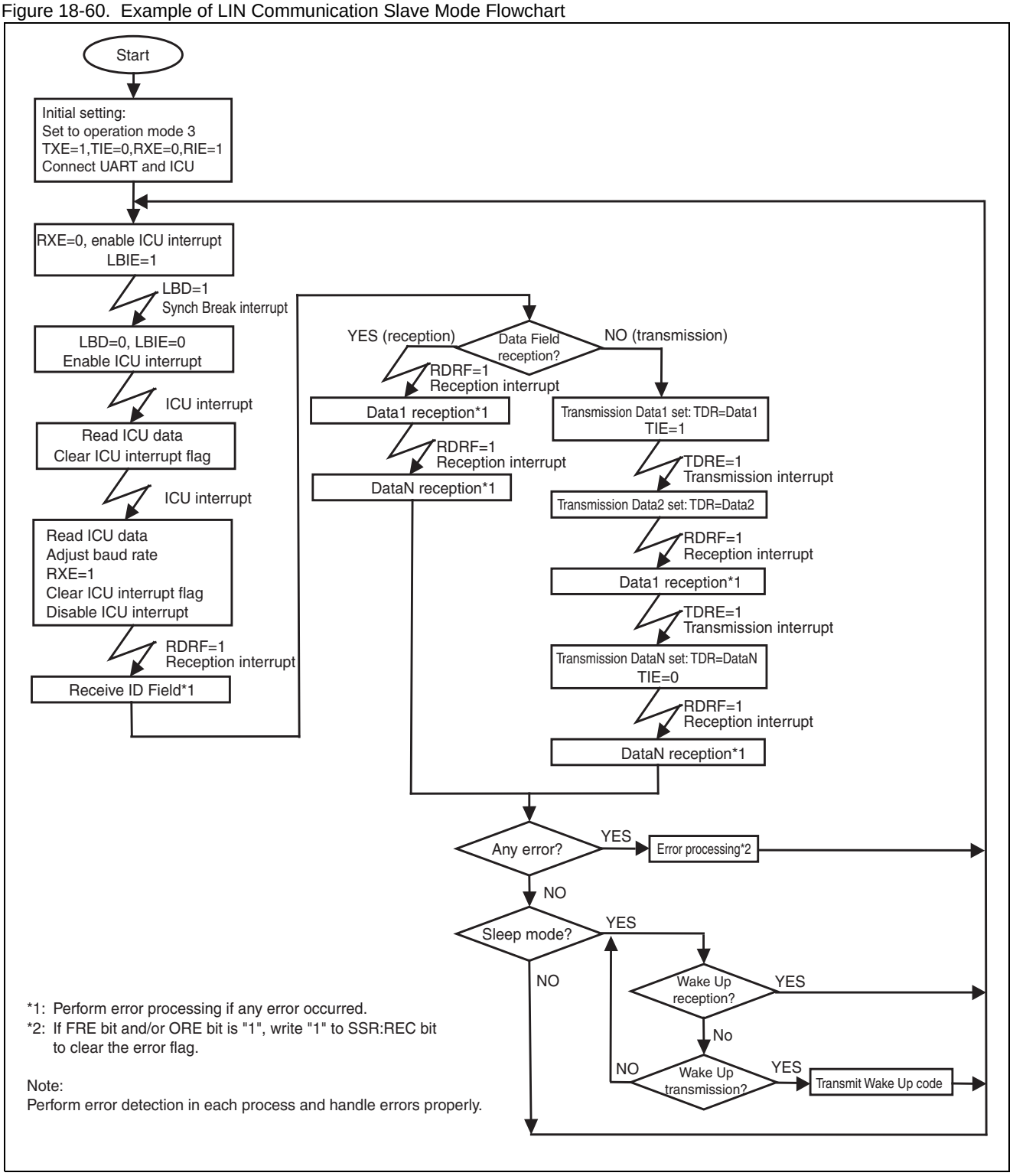
1. Master

Figure 18-59. Example of LIN Communication Master Mode Flowchart



2. Slave

Figure 18-60. Example of LIN Communication Slave Mode Flowchart



18.7 I²C Interface

This section describes the I²C interface supported by operation mode 4 among multi-function serial interface functions.

I²C Interface

Overview of the I²C Interface

Registers of the I²C Interface

- I²C Bus Control Register (IBCR)
- Serial Mode Register (SMR)
- I²C Bus Status Register (IBSR)
- Serial Status Register (SSR)
- Reception Data Register/Transmission Data Register (RDR/TDR)
- 7-Bit Slave Address Mask Register (ISMK)
- 7-Bit Slave Address Register (ISBA)
- Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)

Interrupts of the I²C Interface

- Master Mode
- Slave Mode
- Bus Error

Dedicated Baud Rate Generator

I²C Flowchart Examples

18.7.1 Overview of the I²C Interface

The I²C interface supports inter IC bus and functions as a master/slave device on the I²C bus.

Functions of the I²C Interface

The I²C interface has following functions

- Master/slave transmission/reception function
- Arbitration function
- Clock synchronization function
- Transmission direction detection function
- Function that generates and detects a repeated START condition
- Bus error detection function
- General call addressing function
- 7-bit addressing as master and slave
- Interrupts can be generated during transmission and when a bus error occurs.
- 10-bit addressing function can be supported by program.

18.7.2 Registers of the I²C Interface

A register list of the I²C interface is shown below.

Register List of the I²C Interface

Table 18-35. Register List of the I²C Interface

	Address		bit15bit8	bit7bit0
I²C	ch.0:000021H ch.1:00002BH ch.2:00003FH ch.3:000049H ch.4:000053H ch.5:00005DH ch.6:007791H	ch.0:000020H ch.1:00002AH ch.2:00003EH ch.3:000048H ch.4:000052H ch.5:00005CH ch.6:007990H	IBCR (I²C bus control register)	SMR (Serial mode register)
	ch.0:000023H ch.1:00002DH ch.2:000041H ch.3:00004BH ch.4:000055H ch.5:00005FH ch.6:007993H	ch.0:000022H ch.1:00002CH ch.2:000040H ch.3:00004AH ch.4:000054H ch.5:00005EH ch.6:007992H	SSR (Serial status register)	IBSR (I²C bus status register)
	ch.0:000025H ch.1:00002FH ch.2:000043H ch.3:00004DH ch.4:000057H ch.5:000061H ch.6:007995H	ch.0:000024H ch.1:00002EH ch.2:000042H ch.3:00004CH ch.4:000056H ch.5:000060H ch.6:007994H	-	RDR/TDR (Reception/transmission data register)
	ch.0:000027H ch.1:000031H ch.2:000045H ch.3:00004FH ch.4:000059H ch.5:000063H ch.6:007997H	ch.0:000026H ch.1:000030H ch.2:000044H ch.3:00004EH ch.4:000058H ch.5:000062H ch.6:007996H	BGR1 (Baud rate generator register 1)	BGR0 (Baud rate generator register 0)
	ch.0:000029H ch.1:000033H ch.2:000047H ch.3:000051H ch4.:00005BH ch.5:000065H ch.6:007999H	ch.0:000028H ch.1:000032H ch.2:000046H ch.3:000050H ch.4:00005AH ch.5:000064H ch.6:007998H	ISMK (7-bit slave address mask register)	ISBA (7-bit slave address register)

Table 18-36. Bit Assignment of the I^{2}C Interface

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IBCR/SMR	MSS	ACT/ SCC	ACKE	WSEL	CNDE	INTE	BER	INT	MD2	MD1	MD0	Reser-ved	RIE	TIE	Reser-ved	Reser-ved
SSR/IBSR	REC	TSET	-	-	ORE	RDRF	TDRE	-	FBT	RACK	RSA	TRX	AL	RSC	SPC	BB
RDR/TDR	-	-	-	-	-	-	-	-	D7	D6	D5	D4	D3	D2	D1	D0
BGR0, BGR1	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
ISMK/ISBA	EN	SM6	SM5	SM4	SM3	SM2	SM1	SM0	SAEN	SA6	SA5	SA4	SA3	SA2	SA1	SA0

I²C Bus Control Register (IBCR)

The I²C bus control register (IBCR) selects master/slave mode, generates repeated start conditions, enables acknowledges and interrupts, and indicates the interrupt flags.

Figure 18-61 shows the bit configuration of the I²C bus control register (IBCR) and Table 18-37 shows the functions of each bit.

Figure 18-61. Bit Configuration of the I²C Bus Control Register (IBCR)

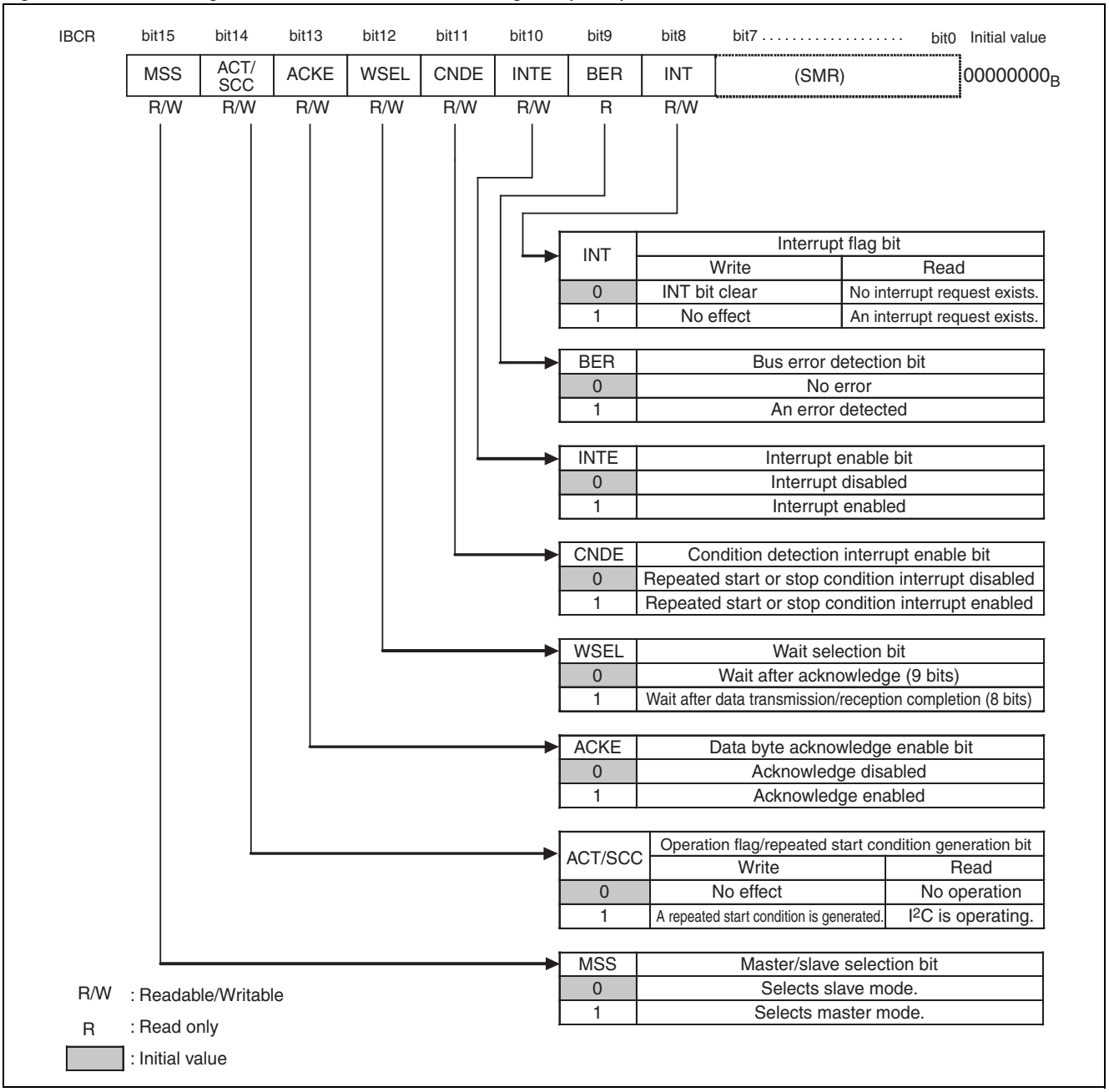


Table 18-37. Function Description of Each Bit of the I²C Bus Control Register (IBCR) (Sheet 1 of 3)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit15</td><td>MSS:
Master/slave selection bit</td><td>
■ Master mode is selected when this bit is set to “1” while I²C bus is in the idle state (EN=1, BB=0).
■ While the BB bit of the IBSR register is “1”, when this bit is set to “1”, it waits to generate a start condition until the BB bit becomes “0”. If the slave addresses matches during the wait and it operates as a slave, this bit becomes “0” and the AL bit of the IBSR register becomes “1”.
■ While the interrupt flag (INT) is “1” during master operation (MSS=1, ACT=1), when “0” is written to this bit, a stop condition is generated.
The MSS bit is cleared by the following conditions:
1. I²C interface disabled (EN bit=0)
2. Arbitration lost occurrence
3. Bus error detection (BER bit=1)
4. “0” writing to the MSS bit when INT=1
The relationship between the MSS bit and the ACT bit is shown below:
<table>
<tr><th>MSS bit</th><th>ACT bit</th><th>State</th></tr>
<tr><td>0</td><td>0</td><td>Idle</td></tr>
<tr><td>0</td><td>1</td><td>Slave addresses are matched or ACK response* to reserved address is made, and slave operation is running. (Slave mode)</td></tr>
<tr><td>1</td><td>0</td><td>Master operation standby</td></tr>
<tr><td>1</td><td>1</td><td>Master operation is running. (Master mode)</td></tr>
</table>
*: ACK response: indicates that the SDA of the I²C bus in acknowledge period is “L”.
Notes:
■ If changing the MSS bit to “0” from “1”, do so while the MSS bit =1 and INT bit =1 are being set. When writing “0” to the MSS bit while the ACT bit is “1”, the INT bit is also cleared to “0”.
■ During master operation, even if the MSS bit is written to “0”, “1” is read from it while the ACT bit is being “1”.
</td></tr>
</table>

Table 18-37. Function Description of Each Bit of the I²C Bus Control Register (IBCR) (Sheet 2 of 3)

Bit name		Function
bit14	ACT/SCC: Operation flag/repeated start condition generation bit	This bit differs its meaning between read and write access. **Read**: ACT bit **Write**: SCC bit The ACT bit indicates that the operation is in master mode or in slave mode. **ACT bit set conditions:** ■ When a start condition is output to the I²C bus. (Master mode) ■ When the slave address and the address transmitted from the master are matched. (Slave mode) ■ When a reserved address is detected and an acknowledge response is made to it. (Slave mode when MSS=0) **ACT bit reset conditions:** <Master mode> ■ Stop condition detection ■ Arbitration lost detection ■ Bus error detection ■ I²C interface disabled (EN bit=0) <Slave mode> ■ (Repeated) start condition detection ■ Stop condition detection ■ No acknowledge response made under a reserved address detected condition (RSA=1) ■ I²C interface disabled (EN bit=0) ■ Bus error occurrence (BER bit=1) In master mode, a repeated start is executed by writing “1” to this bit. Writing “0” is invalid. **Notes:** ■ Write “1” to the SCC bit during a master mode interrupt (MSS=1, ACT=1, INT=1). When writing “1” to the SCC bit while the ACT bit is “1”, the INT bit is cleared to “0”. ■ In slave mode (MSS=0, ACT=1), writing “1” to this bit is disabled. ■ When writing “1” to the SCC bit and “0” to the MSS bit, the MSS bit has a higher priority. ■ When reading by a read-modify-write (RMW) instruction, the SCC bit is read.
bit13	ACKE: Data byte acknowledge enable bit	■ When this bit is set to “1”, “L” is output at the acknowledge timing. ■ To change this bit when ACT=1, do so while the INT bit is “1”. This bit becomes invalid under the following conditions: ■ Acknowledge (automatic generation) generated to the address field other than reserved addresses ■ During data transmission (RSA=0, TRX=1, FBT=0)
bit12	WSEL: Wait selection bit	■ This bit selects to generate an interrupt (INT=1) before or after acknowledge to make the I²C bus wait. ■ The WSEL bit becomes invalid under the following conditions: ❒ Interrupt occurrence (INT=1) to the first byte* ❒ Reserved address detection (FBT=1, RSA=1) *: First byte: means the data after a (repeated) start condition.
bit11	CNDE: Condition detection interrupt enable bit	In master mode or slave mode (ACT=1), this bit enables an interrupt generation when a stop condition or a repeated start condition is detected. An interrupt is generated when the RSC or SPC bit of the IBSR register is “1” and this bit is “1”.
bit10	INTE: Interrupt enable bit	In master mode or slave mode, this bit enables an interrupt generation (INT=1) for data transmission/reception and for a bus error.

Table 18-37. Function Description of Each Bit of the I²C Bus Control Register (IBCR) (Sheet 3 of 3)

Bit name		Function
bit9	BER: Bus error detection bit	This bit indicates that an error is detected on the I²C bus. BER bit set conditions: ■ Detection of a start condition or a stop condition during the first byte* transfer ■ Detection of a (repeated) start condition or a stop condition in the 2nd to 9th (acknowledge) bit of the data in the second byte or later. BER bit reset conditions: ■ Writing “0” to the INT bit when BER=1 ■ I²C interface disabled (EN=0) *: First byte: means the data after a (repeated) start condition. **Note:** Check this bit when the interrupt flag (INT bit) becomes “1”. When this bit is “1”, indicating that the transmission/reception operation was not performed successfully, perform a retransmission or other necessary processes.
bit8	INT: Interrupt flag bit	In master mode or slave mode, this flag is sets to “1” after the 8th or 9th bit of a data transmission/reception (ACK) or when a bus error occurs. In the case except a bus error, the SCL is set to “L” when the INT bit becomes “1”, and the “L” state of the SCL is cancelled when the INT bit becomes “0”. **INT bit set conditions:** <8th bit> ■ When a reserved address is detected in the first byte. ■ When arbitration lost is detected in the second byte or later if WSEL is “1”. ■ When the TDRE bit is “1” in the second byte or later if WSEL is “1” in master operation. ■ When the TDRE bit is “1” in the second byte or later if WSEL is “1” in slave operation. ■ When the TDRE bit is “1” in the second byte or later if WSEL is “1” during slave transmission. <9th bit> ■ When arbitration lost is detected in the first byte. ■ When NACK is received if the setting other than stop condition output setting (writing “0” to the MSS bit in master operation) is being set. ■ When a reserved address is not detected in the first byte but the TDRE bit is “1” in the transmission direction (TRX=1) of the master mode or slave mode. ■ When a reserved address is not detected in the first byte but the TDRE bit is “1” in the reception direction (TRX=0) of the master mode or slave mode. ■ When arbitration lost is detected in the second byte or later if WSEL=0. ■ When the TDRE bit is “1” in the second byte or later if WSEL=0 in master operation. ■ When the TDRE bit is “1” in the second byte or later if WSEL=0 during slave transmission. ■ During slave reception if WSEL=0. However, when the slave reception is performed in the first byte where a reserved address is detected, no interrupt will be generated at the 9th bit. <Other> ■ Detection of a bus error INT bit reset conditions: ■ Writing “0” to the INT bit ■ Writing “0” to the MSS bit while the INT bit and the ACT bit are “1”. ■ Writing “1” to the SCC bit while the INT bit and the ACT bit are “1”. Writing “1” to the INT bit is invalid. **Notes:** ■ When setting the EN bit to “0”, the RDRF bit and the INT bit may become “1” depending on the reception timing. In such case, read reception data to clear the INT bit. ■ When reading by a read-modify-write (RMW) instruction, “1” is read.

Serial Mode Register (SMR)

The serial mode register (SMR) sets operation mode and enables/disables transmission/reception interrupts.

Figure 18-62 shows the bit configuration of the serial mode register (SMR) and Table 18-38 shows the functions of each bit.

Figure 18-62. Bit Configuration of the Serial Mode Register (SMR)

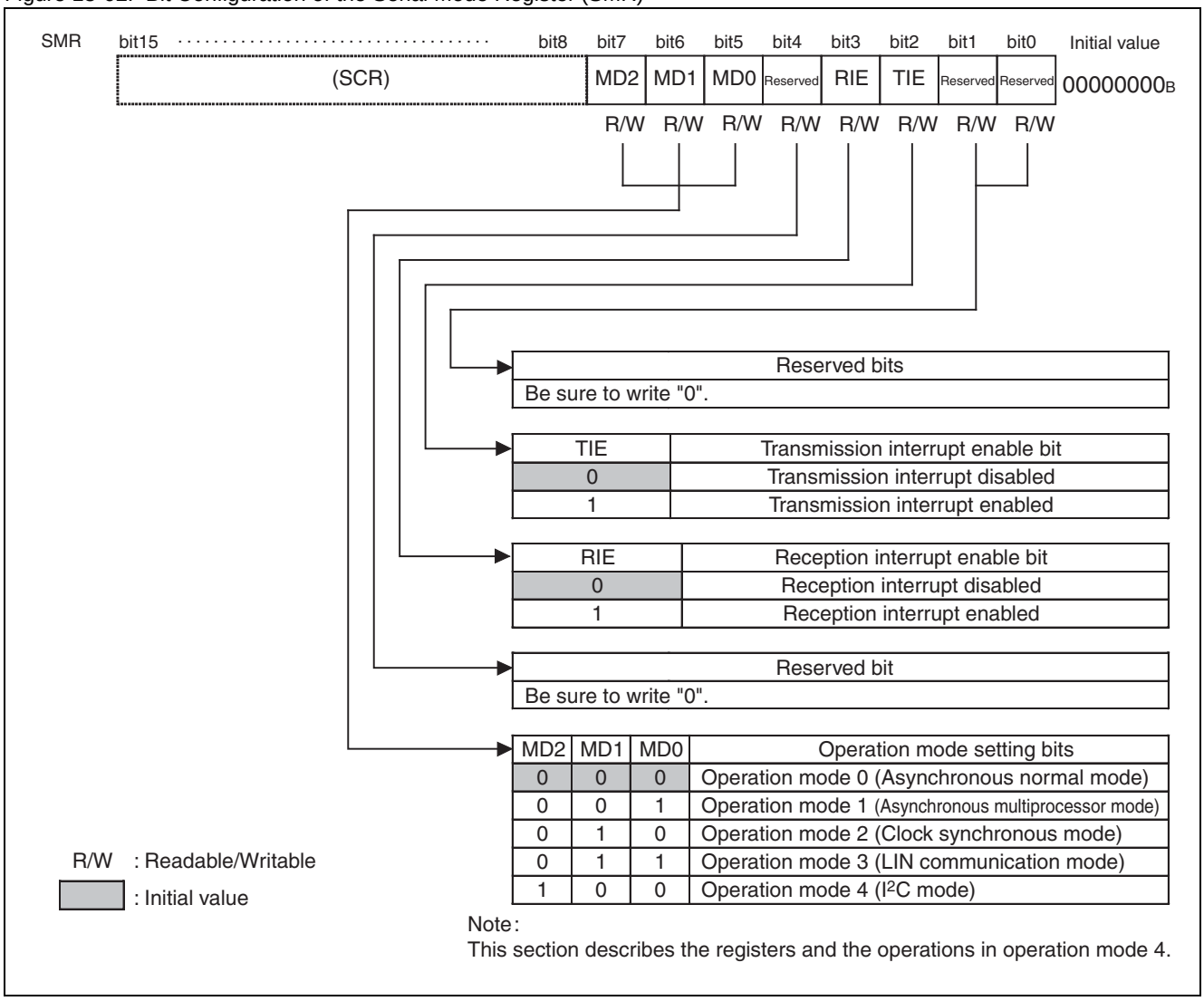


Table 18-38. Function Description of Each Bit of the Serial Mode Register (SMR)

Bit name		Function
bit7 to bit5	MD2 to MD0: Operation mode setting bits	These bits set the operation mode. "000_B": Operation mode 0 (asynchronous normal mode) is set. "001_B": Operation mode 1 (asynchronous multiprocessor mode) is set. "010_B": Operation mode 2 (clock synchronous mode) is set. "011_B": Operation mode 3 (LIN communication mode) is set. "100_B": Operation mode 4 (I^2C mode) is set. This section describes the registers and the operations in operation mode 4 (I^2C mode). **Notes:** ■ Settings other than the above are prohibited. ■ To switch the operation mode, do so after disabling I^2C (ISMK:EN=0). ■ Set each register after setting the operation mode.
bit4	Reserved bit	Be sure to write "0".
bit3	RIE: Reception interrupt enable bit	■ This bit enables/disables the output of reception interrupt requests to the CPU. ■ If RIE bit and reception data flag bit (RDRF) are "1", or if the error flag bit (ORE) is "1", a reception interrupt request is output. **Note:** When receiving data using the INT bit of the I^2C bus control register (IBCR), set this bit to "0".
bit2	TIE: Transmission interrupt enable bit	■ This bit enables/disables the output of transmission interrupt requests to the CPU. ■ If TIE bit and TDRE bit are "1", a transmission interrupt request is output. **Note:** When transmitting data using the INT bit of the I^2C bus control register (IBCR), set this bit to "0".
bit1, bit0	Reserved bits	Be sure to write "0".

I²C Bus Status Register (IBSR)

The I²C bus status register (IBSR) indicates a detection of a repeated start, acknowledge, data direction, arbitration lost, stop condition, I²C bus state, and bus error.

Figure 18-63 shows the bit configuration of the I²C bus status register (IBSR) and Table 18-39 shows the function of each bit.

Figure 18-63. Bit Configuration of the I²C Bus Status Register (IBSR)

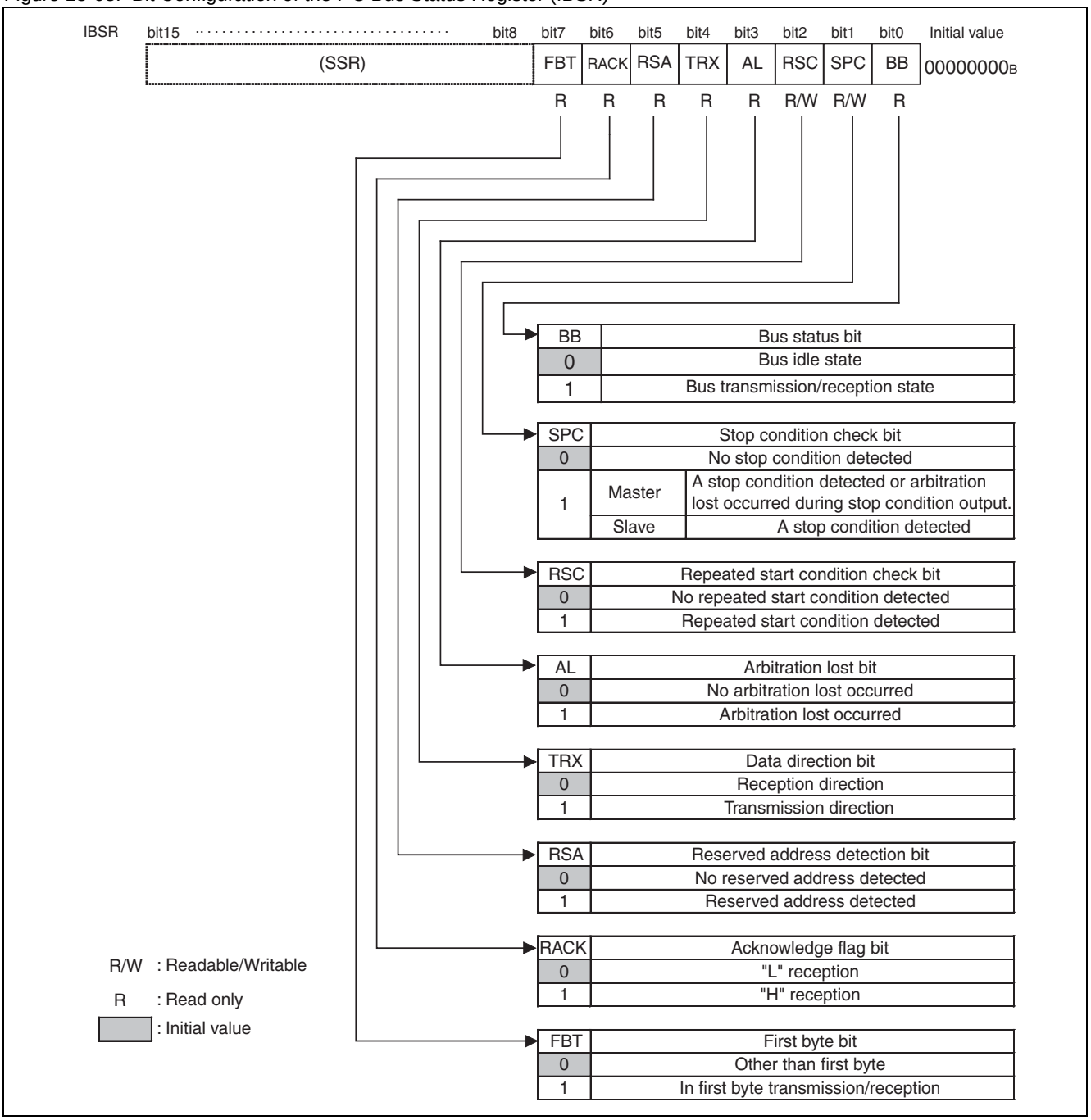


Table 18-39. Function Description of Each Bit of the I²C Bus Status Register (IBSR) (Sheet 1 of 2)

Bit name		Function
bit7	FBT: First byte bit	This bit indicates the first byte. **FBT bit set condition:** ■ Detection of a (repeated) start condition **FBT bit clear conditions:** ■ Transmission/reception of the 2nd byte ■ Detection of a stop condition ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1)
bit6	RACK: Acknowledge flag bit	This bit indicates acknowledge is received in the first byte or is received in master mode or slave mode. **RACK bit update conditions:** ■ Acknowledge in the first byte ■ Acknowledge of data in master mode or slave mode **RACK bit clear conditions (RACK bit=0):** ■ Detection of a (repeated) start condition ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1)
bit5	RSA: Reserved address detection bit	This bit indicates a detection of a reserved address. **RSA bit set conditions (RSA=1):** ■ The first byte is ($0000xxxx_B$) or ($1111xxxx_B$). An “x” indicates “0” or “1”. **RSA bit reset conditions (RSA=0):** ■ Detection of a (repeated) start condition ■ Detection of a stop condition ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1) If the RSA bit becomes “1” in the first byte, the interrupt flag (INT) is set to “1” to set the SCL to “L” at the SCL falling in the 8th bit of the first byte. To read reception data and operate as a slave, set the ACKE to “1” at this time to clear the interrupt flag (INT) to “0”. Then, if the TRX bit is “0”, it receives data as a slave. Set the ACKE bit to “0” not to receive data in the middle of the process. No data will be received after this setting. **Note:** If the ACKE is set to “0” in data transfer operation, the ACKE cannot be set to “1” until a stop condition or a repeated start condition is detected.
bit4	TRX: Data direction bit	This bit indicates data direction. **TRX bit set conditions:** ■ When a (repeated) start condition is transmitted in master mode. ■ When the 8th bit of the first byte is “1” in slave mode (transmission direction as slave). **TRX bit reset conditions:** ■ When arbitration lost is occurred (AL=1). ■ When the 8th bit of the first byte is “0” in slave mode (reception direction as slave). ■ When the 8th bit of the first byte is “1” in master mode (reception direction as master). ■ Detection of a stop condition ■ Detection of a (repeated) start condition in mode other than master mode ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1)

Table 18-39. Function Description of Each Bit of the I²C Bus Status Register (IBSR) (Sheet 2 of 2)

Bit name		Function
bit3	AL: Arbitration lost bit	This bit indicates arbitration lost. **AL bit set conditions:** ■ When, in master mode, the data output and the one received are different. ■ When it is operating as a slave even if the MSS bit is being set to “1”. ■ When a repeated start condition is detected at the 1st bit of the data in the second byte or later in master mode. ■ When a stop condition is detected at the 1st bit of the data in the second byte or later in master mode. ■ When trying to generate a repeated start condition but cannot in master mode. ■ When trying to generate a stop condition but cannot in master mode. **AL bit reset conditions:** ■ Writing “1” to the MSS bit ■ Writing “0” to the INT bit ■ Writing to “0” to the SPC bit when AL bit=1 and SPC bit=1. ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1)
bit2	RSC: Repeated start condition check bit	This bit indicates a detection of a repeated start condition in master mode or in slave mode. **RSC bit set condition:** ■ When a repeated start condition is detected after acknowledge in slave or master mode operation. **RSC bit reset conditions:** ■ Writing “0” to the RSC bit ■ Writing “1” to the MSS bit ■ I²C interface disabled (EN bit=0) Writing “1” to this bit is invalid. **Notes:** ■ If no acknowledge response is made while a reception operation is running in slave mode after detecting reserved address, slave mode ends. Therefore, this bit will not be set to “1” even if a repeated start condition is detected next. ■ When reading by a read-modify-write (RMW) instruction, “1” is read.
bit1	SPC: Stop condition check bit	This bit indicates a detection of a stop condition in master mode or in slave mode. **SPC bit set conditions:** ■ When a stop condition is detected in slave or master mode operation. ■ In master mode, when arbitration lost occurs in the stop condition generating operation. **SPC bit reset conditions:** ■ Writing “0” to this bit ■ Writing “1” to the MSS bit ■ I²C interface disabled (EN bit=0) Writing “1” to this bit is invalid. **Notes:** ■ If no acknowledge response is made while a reception operation is running in slave mode after detecting reserved address, slave mode ends. Therefore, this bit will not be set to “1” even if a stop condition is detected next. ■ When reading by a read-modify-write (RMW) instruction, “1” is read.
bit0	BB: Bus status bit	This bit indicates the bus status. **BB bit set condition:** ■ When “L” is detected in the SDA or SCL of the I²C bus. **BB bit reset conditions:** ■ Detection of a stop condition ■ I²C interface disabled (EN bit=0) ■ Detection of a bus error (BER bit=1)

Serial Status Register (SSR)

The serial status register (SSR) checks the transmission/reception status.

Figure 18-64 shows the bit configuration of the serial status register (SSR) and Table 18-40 shows the functions of each bit.

Figure 18-64. Bit Configuration of the Serial Status Register (SSR)

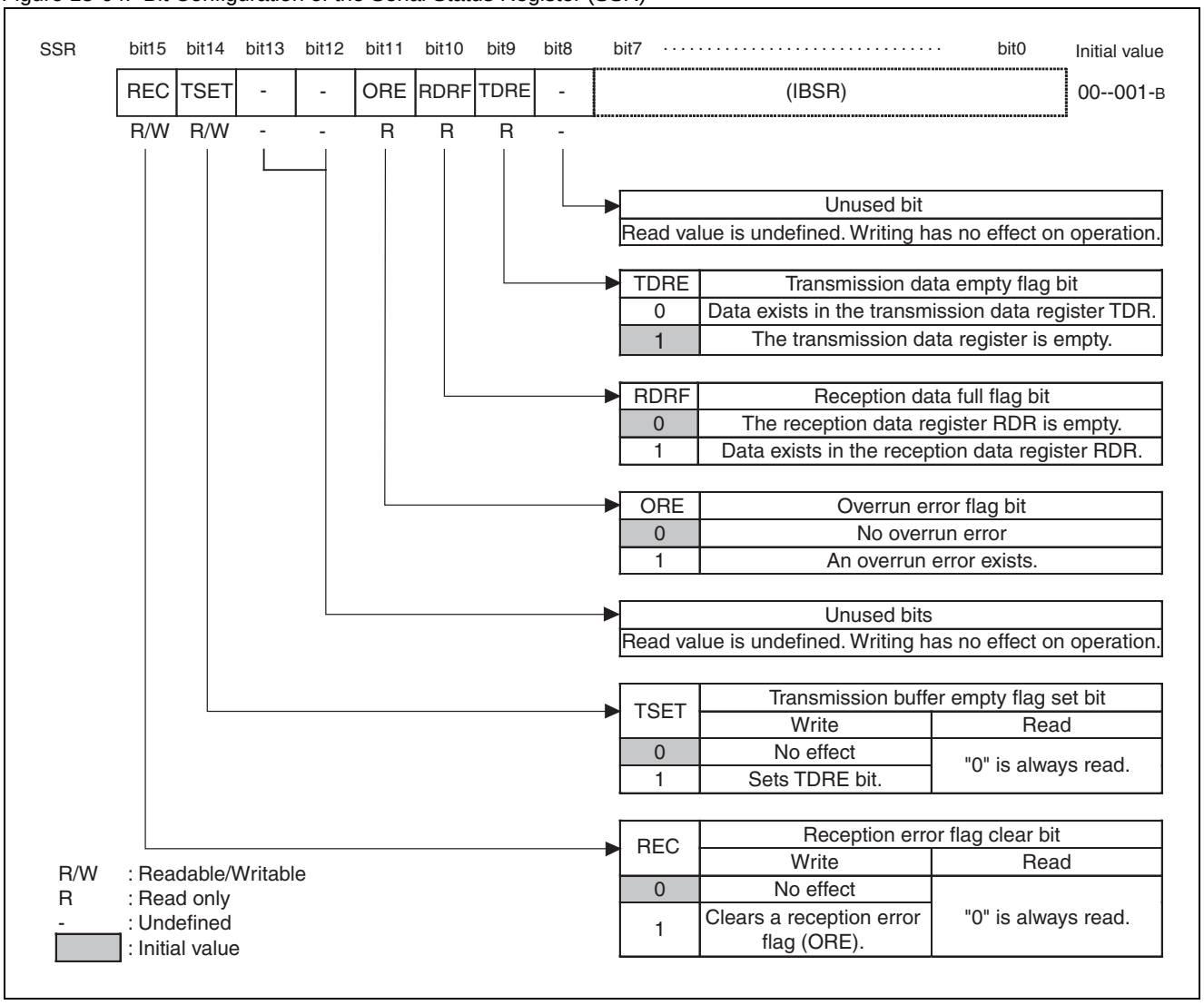


Table 18-40. Function Description of Each Bit of the Serial Status Register (SSR)

Bit name		Function
bit15	REC: Reception error flag clear bit	This bit clears ORE bit of the serial status register (SSR). ■ When “1” is written, the ORE bit is cleared. ■ When “0” is written, this writing has no effect. When read, “0” is always read.
bit14	TSET: Transmission buffer empty flag set bit	This bit sets the TDRE bit of the serial status register (SSR). ■ When “1” is written, the TDRE bit is set. ■ When “0” is written, this writing has no effect. When read, “0” is always read.
bit13, bit12	Unused bits	When read: The value is undefined. When write: No effect
bit11	ORE: Overrun error flag bit	■ This bit is set to “1” when an overrun error occurs during reception and is cleared when “1” is written to the REC bit of the serial status register (SSR). ■ A reception interrupt request is output when the ORE bit and RIE bit are “1”. ■ The reception data register (RDR) is invalid when this flag is set.
bit10	RDRF: Reception data full flag bit	■ This flag indicates the status of the reception data register (RDR). ■ A reception interrupt request is output when the RIE bit and the reception data flag bit (RDRF) are “1”. ■ This bit is set to “1” when reception data is loaded into the RDR and is cleared to “0” when the reception data register (RDR) is read. ■ This bit is set at the SCL falling timing in the 8th bit of data. ■ This bit is also set by NACK response*. *: NACK response: indicates that the SDA of the I^2C bus in acknowledge period is “H”.
bit9	TDRE: Transmission data empty flag bit	■ This flag indicates the status of the transmission data register (TDR). ■ A transmission interrupt request is output when the TIE bit and TDRE bit are “1”. ■ This bit becomes “0” when transmission data is written to the TDR, indicating that valid data exists in the TDR. This bit becomes “1” when the data is loaded into the transmission shift register and transmission starts, indicating that valid data does not exist in the TDR. ■ This bit is set when “1” is written to the TSET bit of the serial status register (SSR). Use this bit to set the TDRE bit to “1” in the case where arbitration lost or a bus error, etc. is detected.
bit8	Unused bit	When read: The value is undefined. When write: No effect

Reception Data Register/Transmission Data Register (RDR/TDR)

The reception data register and transmission data register are located at the same address. When reading the address, it functions as a reception data register, and when writing, it functions as a transmission data register.

Figure 18-65 shows the bit configuration of the reception data register (RDR).

Figure 18-65. Bit Configuration of the Reception Data Register (RDR)

Reception Data Register (RDR)

bit15.....bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
	D7	D6	D5	D4	D3	D2	D1	D0	00000000_B
	R	R	R	R	R	R	R	R	

R: Read only

The reception data register (RDR) is a data buffer register for serial data reception.

- A serial data signal transmitted to the serial data line (SDA pin) is converted in the shift register and stored in the reception data register (RDR).
- When the first byte* is received, the lowest bit (RDR:D0) becomes a data direction bit.
- When reception data is stored into the reception data register (RDR), the reception data full flag bit (SSR:RDRF) is set to “1”.
- The reception data full flag bit (SSR:RDRF) is cleared to "0" automatically when the reception data register (RDR) is read.

*: First byte: means the data after a (repeated) start condition.

Figure 18-66 shows the bit configuration of the transmission data register.

Figure 18-66. Bit Configuration of the Transmission Data Register (TDR)

Transmission Data Register (TDR)

bit15.....bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
	D7	D6	D5	D4	D3	D2	D1	D0	11111111_B
	W	W	W	W	W	W	W	W	

W: Write only

The transmission data register (TDR) is a data buffer register for serial data transmission.

- Values in the transmission data register (TDR) are output from MSB first to the serial data line (SDA pin).
- When the first byte is transmitted, the lowest bit (TDR:D0) becomes a data direction bit.
- The transmission data empty flag (SSR:TDRE) is cleared to “0” when transmission data is written into the transmission data register (TDR).
- The transmission data empty flag (SSR:TDRE) is set to “1” when transmission data is transferred to the shift register for transmission.
- Write the next transmission data under the following conditions:
- When the interrupt flag (INT bit) is “1”.
- No bus error is occurring (BER bit=0).
- Acknowledge is ACK response (“0” reception as acknowledge).
- When the data empty flag (SSR:TDRE) is “0”, transmission data cannot be written into the transmission data register (TDR).

Note:

The transmission data register is a write only register, and the reception data register is a read only register. Since the transmission/reception data registers are located at the same address, the write value and the read value are different. Therefore, the read-modify-write (RMW) instruction such as INC/DEC instructions cannot be used.

7-Bit Slave Address Mask Register (ISMK)

The 7-bit slave address mask register (ISMK) compares or sets each bit of the slave address.

Figure 18-67 shows the bit configuration of the 7-bit slave address register (ISMK) and Table 18-41 shows the functions of each bit.

Figure 18-67. Bit Configuration of the 7-Bit Slave Mask Register (ISMK)

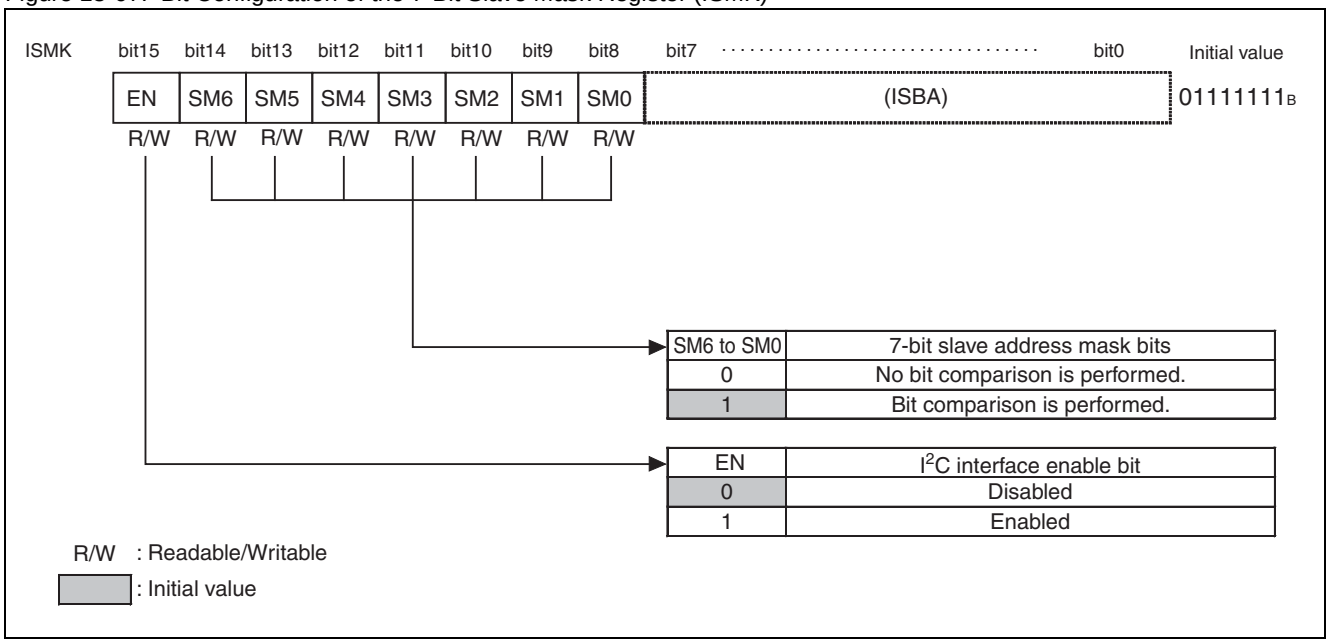


Table 18-41. Function Description of Each Bit of the 7-Bit Slave Mask Register (ISMK)

Bit name		Function
bit15	EN: I^2C interface enable bit	This bit enables/disables I^2C interface operations. When “0” is set: I^2C interface operations are disabled. When “1” is set: I^2C interface operations are enabled. **Notes:** ■ This bit is not cleared to “0” even when the BER bit of the IBSR register is set to “1”. ■ Set the baud rate generator when this bit is “0”. ■ Set the 7-bit slave address and the 7-bit slave mask register when this bit is “0”. ■ When the EN bit is set to “0” while transmitting, pulses can be generated in the SDA/SCL of the I^2C bus.
bit14 to bit8	SM6 to SM0: Slave address mask bits	These bits set whether to compare or not the 7-bit slave address and received address. Bit with “1” setting: The bit is used for comparison. Bit with “0” setting: The bit is handled as if matched. **Note:** Set this register when the EN bit is “0”.

7-Bit Slave Address Register (ISBA)

The 7-bit slave address register (ISBA) sets the slave address.

Figure 18-68 shows the bit configuration of the 7-bit slave address register (ISBA) and Table 18-42 shows the functions of each bit.

Figure 18-68. Bit Configuration of the 7-Bit Slave Address Register (ISBA)

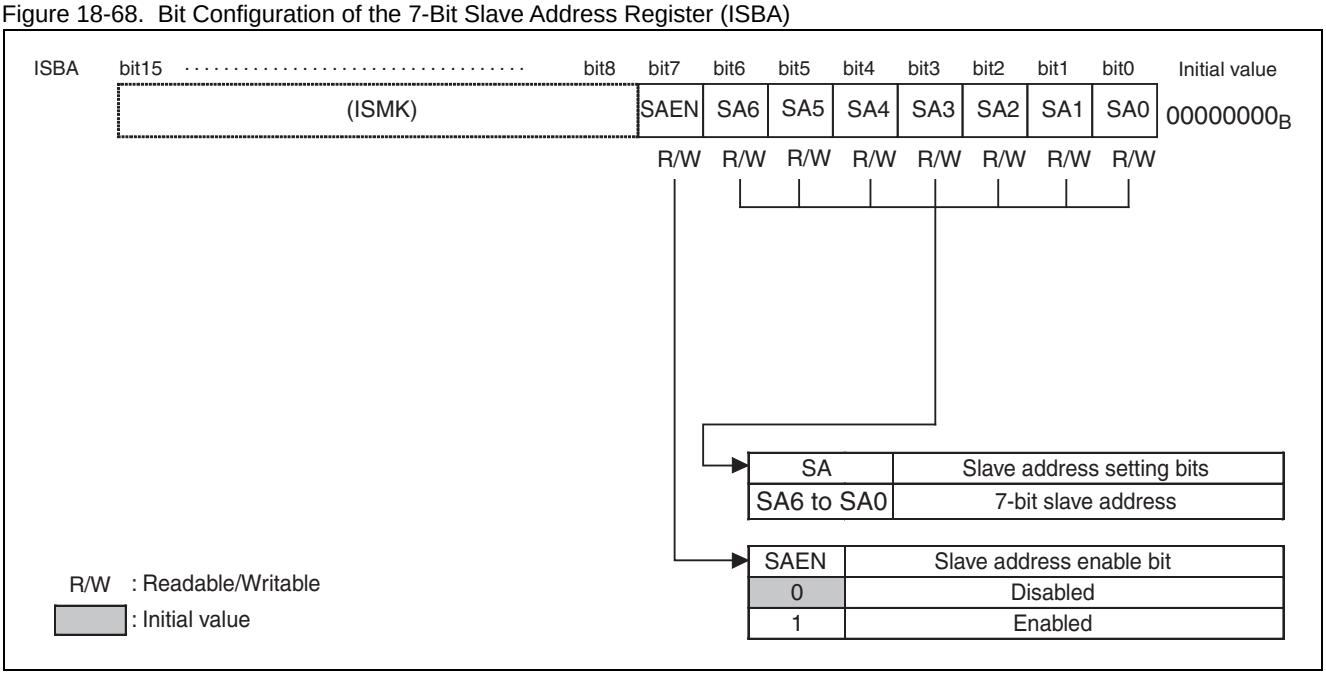


Table 18-42. Function Description of Each Bit of the 7-Bit Slave Address Register (ISBA)

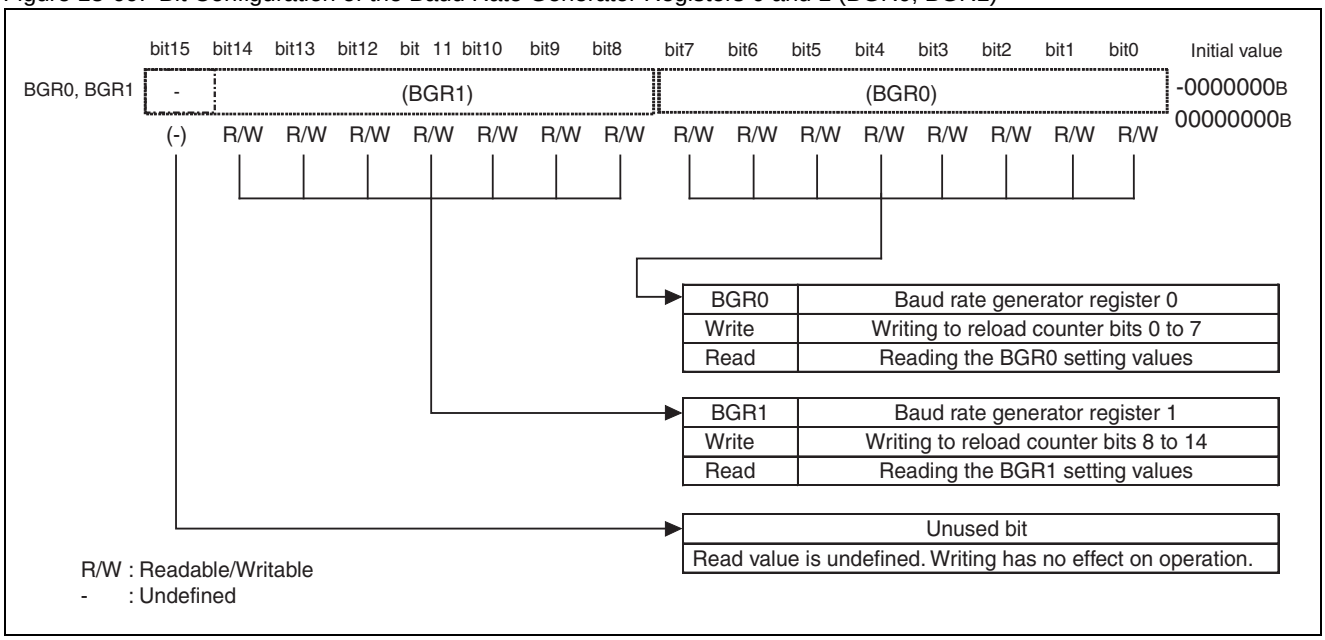
Bit name		Function
bit7	SAEN: Slave address enable bit	This bit is the slave address detection enable bit. When “0” is set: No slave address detection When “1” is set: ISBA and ISMK are set, and the received first byte is compared.
bit6 to bit0	SA6 to SA0: Slave address setting bits	■ If the slave address detection is enabled (SAEN=1), the 7-bit data received after detecting a (repeated) start condition is compared with this register. If all the bits are matched, the device operates as slave mode and outputs ACK. The received slave address is set in this register at that time. (If SAEN=0, no ACK is output.) ■ The address bit set with “0” in the ISMK register is not used for the comparison. **Notes:** ■ Do not set a reserved address. ■ Set this register when the EN bit of the ISMK register is “0”.

Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)

The baud rate generator registers 0 and 1 (BGR0, BGR1) set the division ratio for the serial clock.

Figure 18-69 shows the bit configuration of the baud rate generator registers 0 and 1 (BGR0, BGR1).

Figure 18-69. Bit Configuration of the Baud Rate Generator Registers 0 and 1 (BGR0, BGR1)



The baud rate generator registers set the division ratio of the serial clock.

The BGR1 corresponds to the upper bits and BGR0 corresponds to the lower bits, and writing of reload values for counting and reading of the BGR0, BGR1 setting values are allowed.

When writing reload values to the baud rate generator registers 0 and 1 (BGR0, BGR1), the reload counter starts counting.

Notes:

- When writing into the baud rate generator registers 0 and 1 (BGR0, BGR1), use 16-bit access.
- Set the baud rate generator register when the EN bit of the ISMK register is “0”.
- Set the baud rate regardless of whether the device is in master mode or slave mode.
- In operation mode 4 (I^2C mode), use the machine clock at 8MHz or higher and set the baud rate generator not to exceed 400 kbps.

18.7.3 Interrupts of the I2C Interface

The I2C interface can generate an interrupt request of by one of the following sources:

- Completion of the first byte transmission and reception/completion of data transmission and reception
- Stop condition
- Repeated start condition

Interrupts of the I2C Interface

The interrupt control bits and the interrupt sources of the I2C interface are shown in Table 18-43.

Table 18-43. Interrupt Control Bits and Interrupt Sources of the I2C Interface

Interrupt type	Interrupt request flag bit	Flag register	Interrupt source	Interrupt source enable bit	How to clear the interrupt request flag
Status	INT	IBCR	First byte transmission/reception completion	IBCR:INTE	Write “0” to the interrupt flag bit (IBCR:INT).
			Data transmission/reception completion		
			Bus error detection		
			Arbitration lost detection		
			Reserved address detection		
			NACK reception		
	SPC	IBSR	Stop condition	IBCR:CNDE	Write “0” to SPC.
	RSC		Repeated start detection		Write “0” to RSC.
Reception	RDRF	SSR	Reserved address reception	SMR:FRIE	Read the reception data (RDR).
			Data reception completion		
	ORE	SSR	Overrun error		Write “1” to the reception error flag bit (SSR:REC).
Transmission	TDRE	SSR	Transmission register is empty.	SMR:FTIE	Write to the transmission data (TDR).
			Write "1" to the transmission buffer empty flag set bit (SSR:TSET)		

*:If the normal data has been transmitted/received and if TDRE is "0", no interrupt will be generated. This is because the DMA transfer is supported. To generate the INT flag during data transmission/reception, the TDRE bit must be set be "1" before the INT flag set timing.

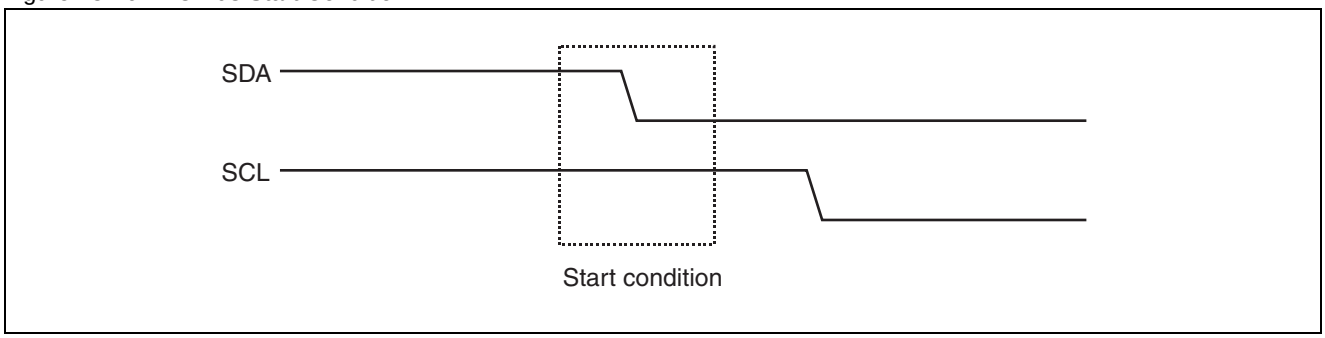
Operation of the I²C Interface Communication

The I²C interface performs communication using two bidirectional bus lines that consist of one serial data line (SDA) and one serial clock line (SCL).

I²C Bus Start Condition

An I²C bus start condition is shown below.

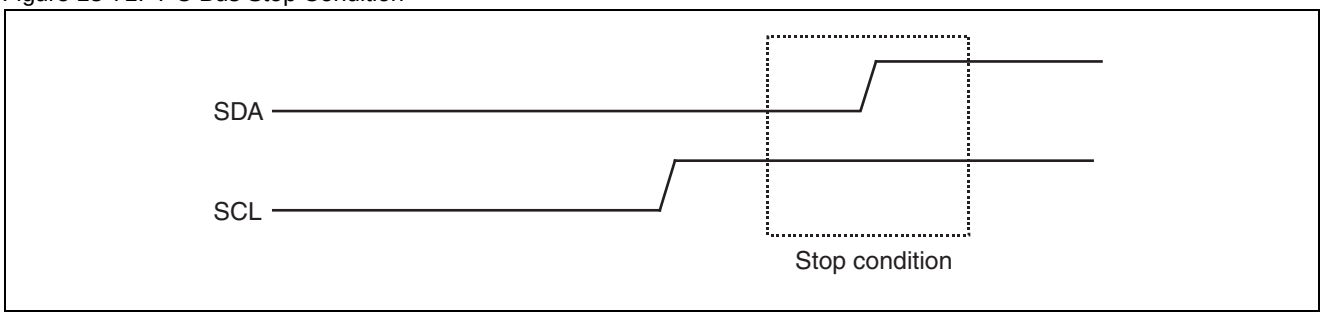
Figure 18-70. I²C Bus Start Condition



I²C Bus Stop Condition

An I²C bus stop condition is shown below.

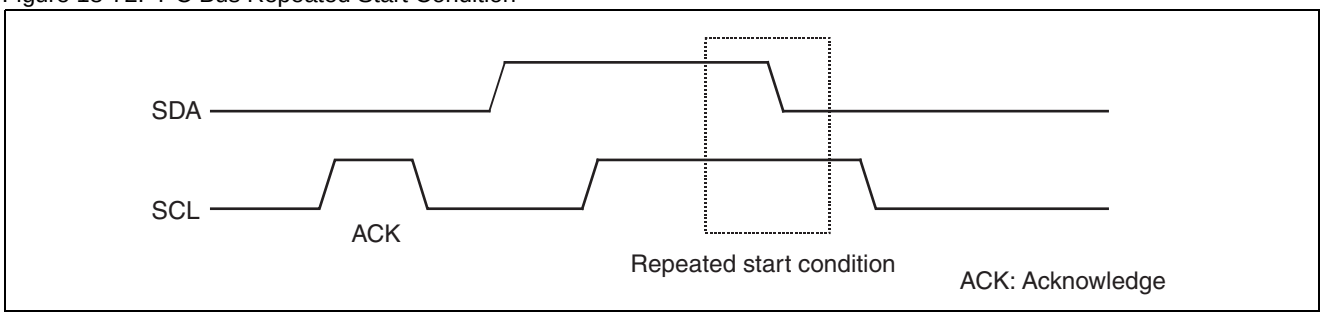
Figure 18-71. I²C Bus Stop Condition



I²C Bus Repeated Start Condition

An I²C bus repeated start condition is shown below.

Figure 18-72. I²C Bus Repeated Start Condition



Master Mode

Master mode generates a start condition on the I^{2}C bus and outputs the clock to the I^{2}C bus. If the MSS bit of the IBCR register is set to “1” when the I^{2}C bus is in idle state (SCL = “H”, SDA = “H”), it enters master mode and the ACT bit of the IBCR register becomes “1”.

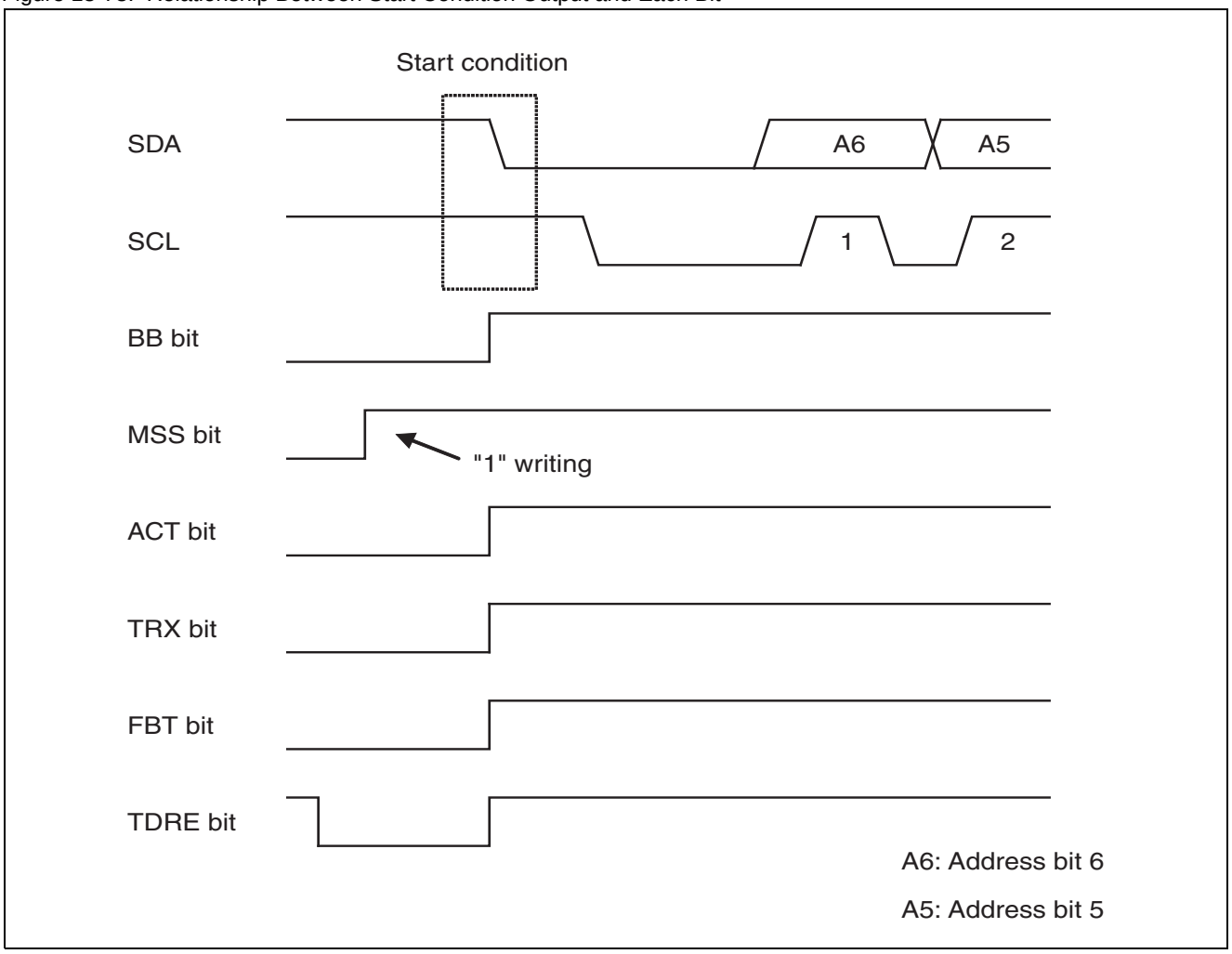
Start Condition Generation

A start condition is output under the following conditions:

- Writing “1” to the MSS bit when SDA= “H”, SCL= “H”, EN=1 and BB=0.

When a start condition is output to the I^{2}C bus, the ACT bit is set to “1”. Then, the BB bit is set to “1” when the start condition is received, indicating that the I^{2}C bus is being used for communication. (See Figure 18-73).

Figure 18-73. Relationship Between Start Condition Output and Each Bit



Note:

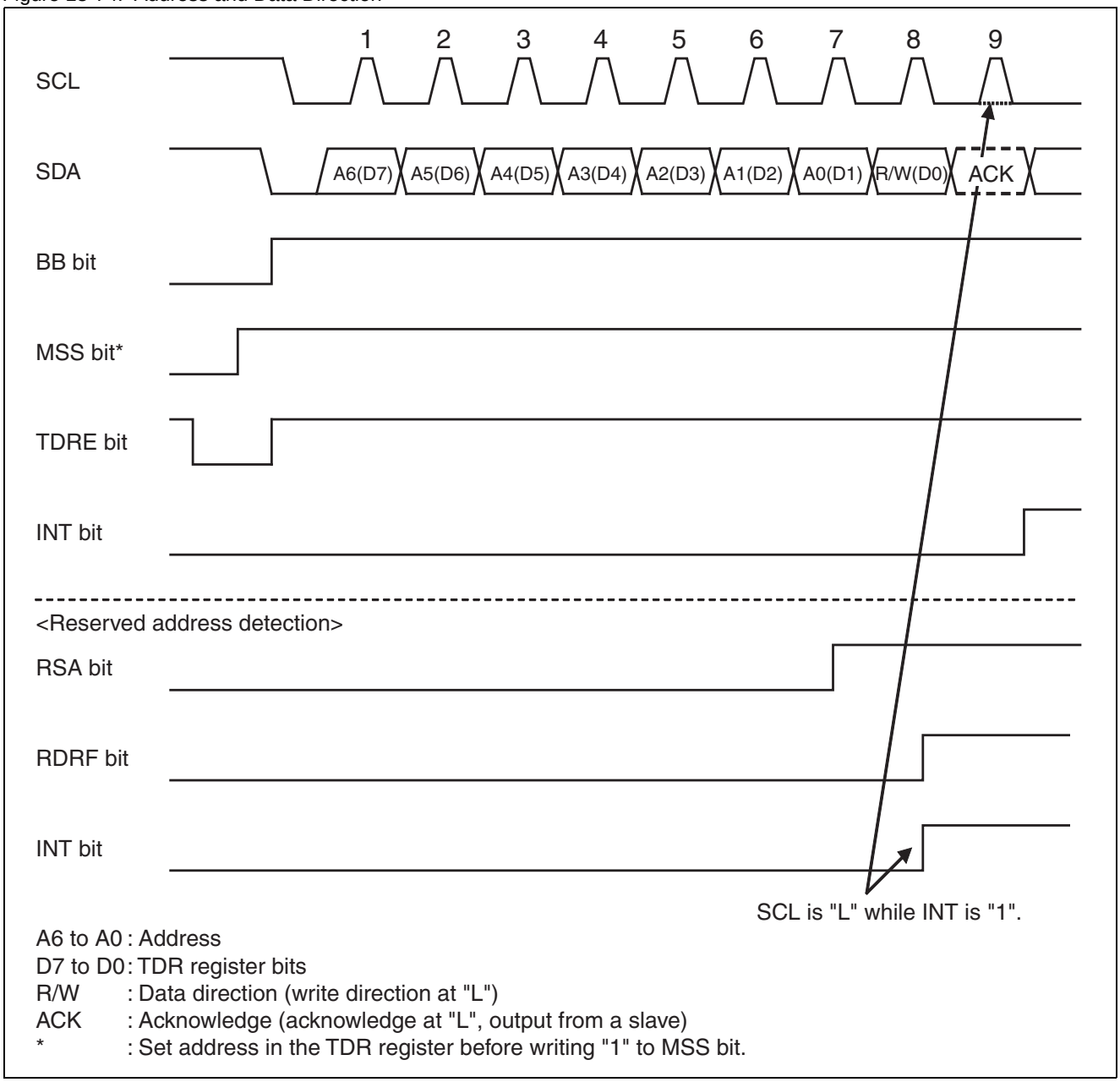
In operation mode 4 (I^{2}C mode), use the machine clock at 8MHz or higher and set the baud rate generator not to exceed 400 kbps.

Slave Address Output

When a start condition is output, data set in the TDR register is output from bit7 as an address. bit0 is used as a data direction bit (R/W), and when the data direction bit (R/W) is “0”, it indicates the data direction is write direction (master -> slave). Set address in the TDR register before setting MSS=1 or SCC=1.

Figure 18-74 shows an output timing of address and data direction.

Figure 18-74. Address and Data Direction



A6 to A0 : Address
D7 to D0 : TDR register bits
R/W : Data direction (write direction at "L")
ACK : Acknowledge (acknowledge at "L", output from a slave)
* : Set address in the TDR register before writing "1" to MSS bit.

Acknowledge Reception by the First Byte Transmission

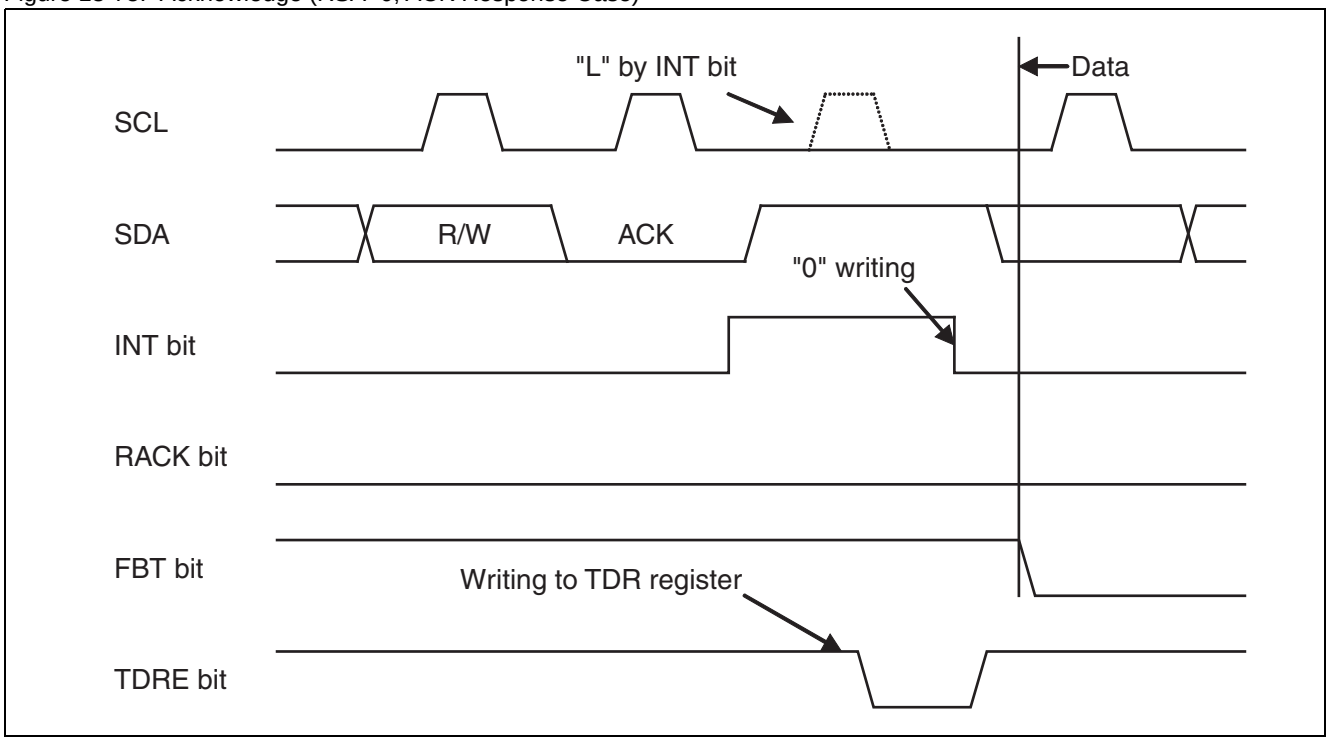
When the data direction bit (R/W) is output, the I^2C interface receives acknowledge from a slave.

Table 18-44. Operation after Acknowledge Reception (RSA Bit=0)

Data direction bit (R/W)	Operation immediately after acknowledge reception	
	When acknowledge is ACK	When acknowledge is NACK
0	If the TDRE bit is “1”, set the INT bit to “1” to wait. If the TDRE bit is “0”, keep the INT bit at “0” and no wait.	Set the INT bit to “1” and wait.
1		

- When the RSA bit is “0”, if the TDRE bit is “1” after acknowledge reception, set the interrupt flag (INT) to “1” and keep the SCL at “L” to wait. To cancel the wait, write “0” to the interrupt flag to set it to “0”. If the TDRE bit is “0”, when ACK is received, the clock is generated in the SCL without setting the interrupt flag to “1”.
- When the RSA bit is “1”, after reserved address reception (before acknowledge), set the interrupt flag (INT) to “1” and keep the SCL at “L” to wait. After the RDR register is read, set the ACKE bit and transmission data, and write “0” to the interrupt flag to set it to “0” to cancel the wait.
- A received acknowledge is set in the RACK bit. The RACK bit is checked during wait, and if it is NACK, “0” is written to the MSS bit or “1” is written to the SCC bit to generate a stop condition or a repeated start condition. The INT bit is automatically cleared to “0” at this time.

Figure 18-75. Acknowledge (RSA=0, ACK Response Case)



Wait operation for address is performed:

- After acknowledge reception when the RSA bit is “0”.
- Before acknowledge reception when the RSA bit is “1”.

The WSEL setting has no effect on the wait operation.

Figure 18-76. Acknowledge (RSA=0, NACK Response Case)

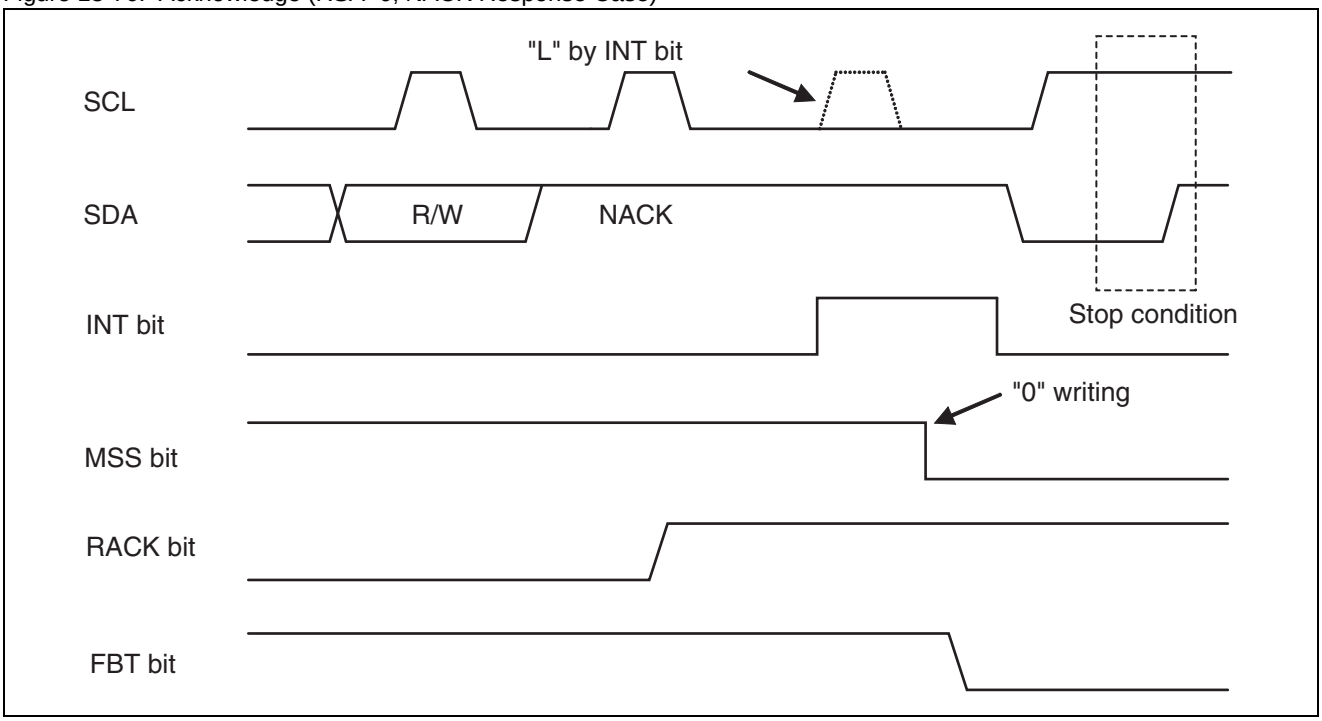


Figure 18-77. Acknowledge (RSA=1, ACK Response Case)

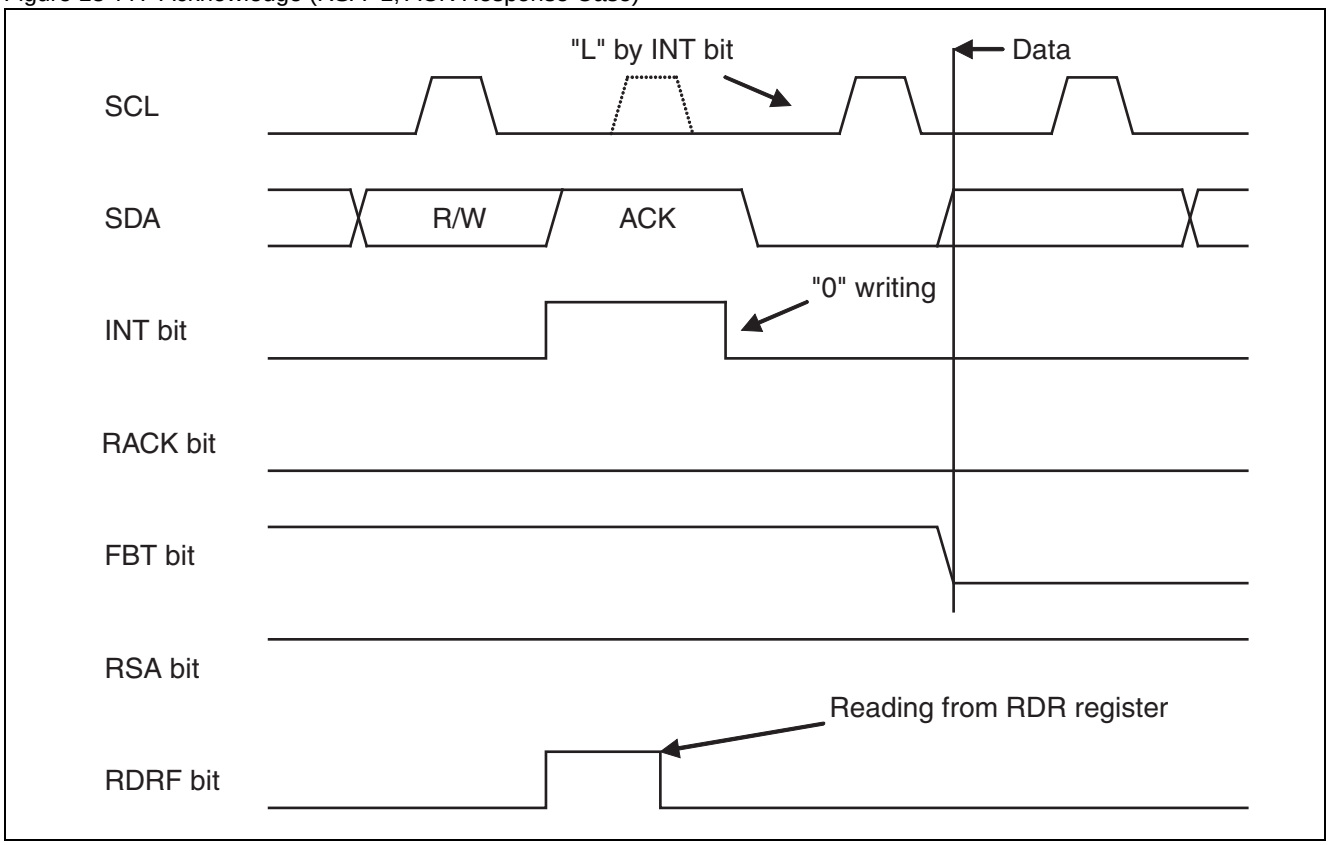
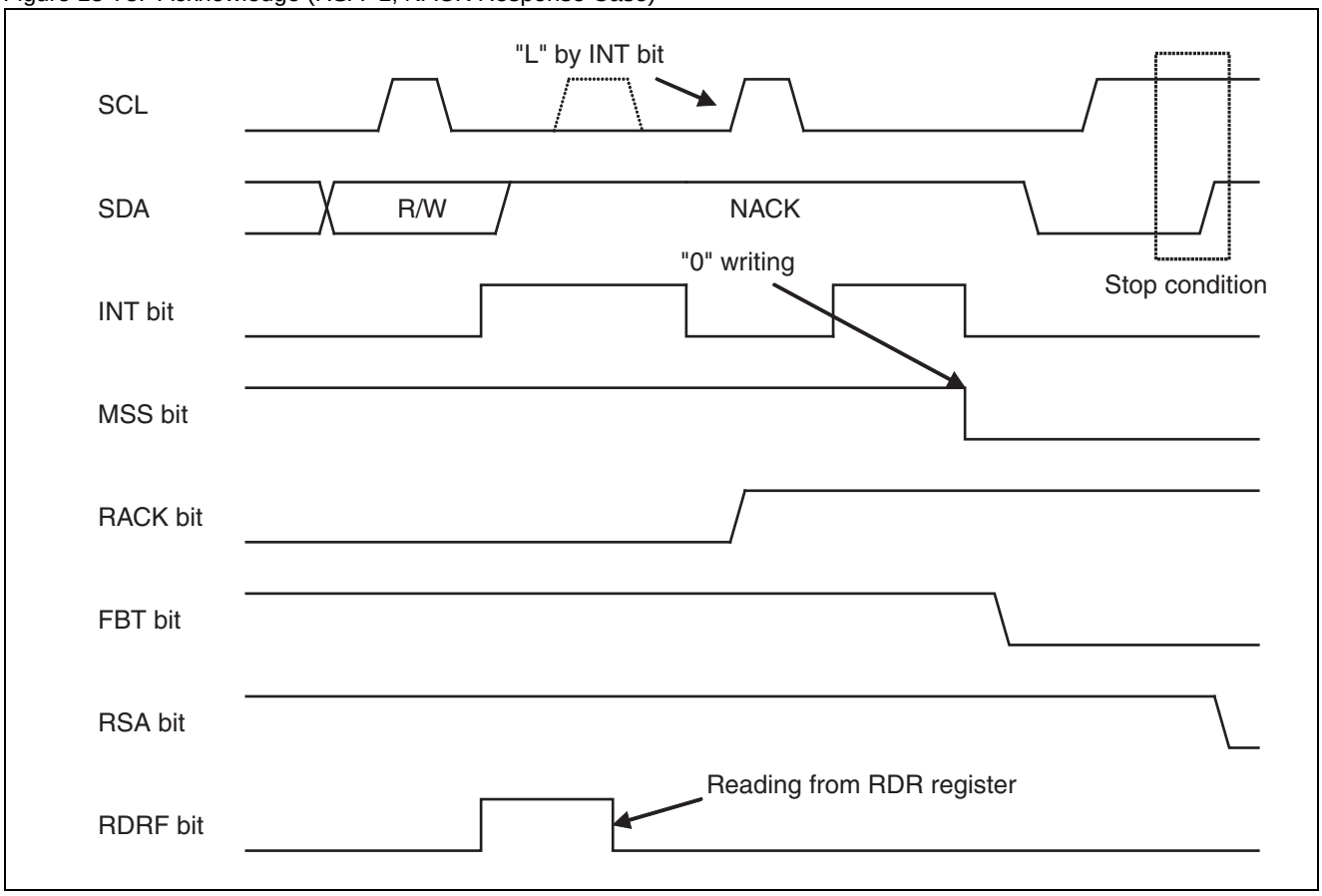


Figure 18-78. Acknowledge (RSA=1, NACK Response Case)



Data Transmission by Master

When the data direction bit (R/W) is “0”, data is transmitted from the master. An ACK or NACK response is returned from the slave for each 1-byte transmission.

Location where wait is generated is different depending on the WSEL bit setting.

Table 18-45. WSEL Bit for Master Data Transmission

WSEL bit	Operation
0	From the second byte, when the TDRE bit is “1” or after acknowledge is generated by arbitration lost detection, the interrupt flag (INT) is set to “1” and the SCL is set to “L” to set wait state.
1	From the second byte, when the TDRE bit is “1” or after the master transmits 1-byte data by arbitration lost detection, the interrupt flag (INT) is set to “1” and the SCL is set to “L” to set wait state.

When NACK is received under a condition other than the stop condition setting (IBCR:MSS=0, ACT=1), however, the interrupt flag (INT) is set after acknowledge regardless of the WSEL setting.

An example procedure to transmit data to a slave is shown below:

- Transmission to an address other a reserved address:

(1). The slave address (including data direction bit) is set in the TDR register and “1” is written to the MSS bit.

(2). After slave address transmission, ACK is received and the interrupt flag (INT) becomes “1”.

(3). The data to be transmitted is written in the TDR register.

(4). The WSEL bit is updated at the same time as “0” is written to the interrupt flag (INT) to cancel the I^2C bus wait.

(5). After transmitting 1 byte, the interrupt flag is set to “1” to make the I^2C bus wait after receiving acknowledge when WSEL=0, or immediately after transmitting 1 byte when WSEL=1. Steps (2) to (4) are repeated until the specified number of data is transmitted. When WSEL=1, however, if NACK is received after wait is cancelled, an interrupt is generated again to make the bus wait after receiving acknowledge.

(6). The MSS bit is set to “0” or the SCC bit is set to “1” to generate a stop condition or a repeated start condition.

- Transmission to a reserved address:

(1). A reserved address is set in the TDR register as slave address and “1” is written to the MSS bit.

(2). After slave address transmission, the interrupt flag (INT) becomes “1”.

(3). The RDR register is read to check the reserved address. *

(4). The data to be transmitted is written in the TDR register.

(5). The WSEL bit is updated at the same time as “0” is written to the interrupt flag (INT) to cancel the I^2C bus wait.

(6). After transmitting 1 byte, the interrupt flag is set to “1” to make the I^2C bus wait after receiving acknowledge when WSEL=0, or immediately after transmitting 1 byte when WSEL=1. Steps (4) to (6) are repeated until the specified number of data is transmitted. When WSEL=1, however, if NACK is received after wait is cancelled, an interrupt is generated again to make the bus wait after receiving acknowledge.

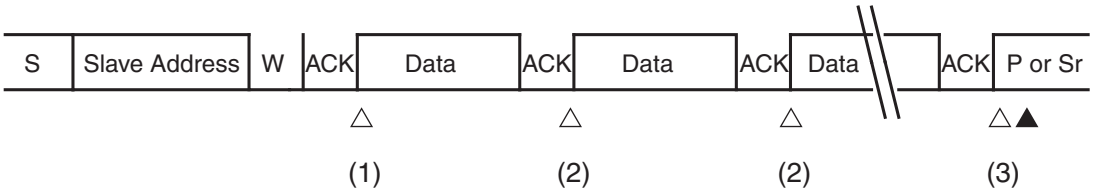
(7). The MSS bit is set to “0” or the SCC bit is set to “1” to generate a stop condition or a repeated start condition.

*: When the reserved address is a general call address in multi-master mode, it is necessary to confirm whether the device will operate as the master or slave for the next data by setting the ACKE and WSEL bits to “1”, if the device may operate as the slave due to the generation of an arbitration lost condition.

Notes:

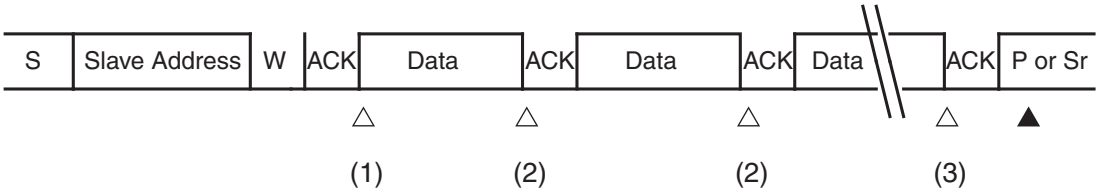
- To change the IBCR register during transmission/reception operation, do so when the interrupt flag (INT) is “1”.
- If the WSEL bit is changed, it will be used for an interrupt flag (INT) generation condition for the next data.
- When the TDRE is “1” in a data transmission operation, if transmission data is written in the TDR register and an ACK response is detected, the interrupt flag (INT) is not set to “1” and the written data is directly transmitted.

Figure 18-79. Master Interrupt 1 (WSEL=0, RSA=0)



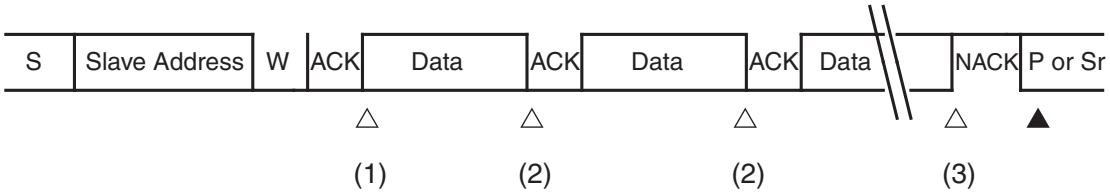
S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by 1-byte transmission + acknowledge reception.
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-80. Master Transmission Interrupt 2 (WSEL=1, RSA=0, ACK Response)



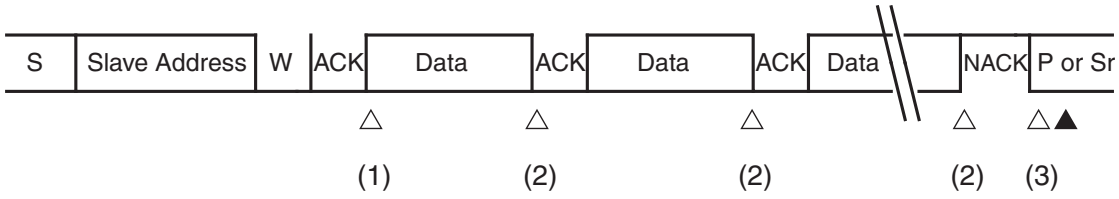
S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by 1-byte transmission
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-81. Master Transmission Interrupt 3 (WSEL=1, RSA=0, NACK Response)



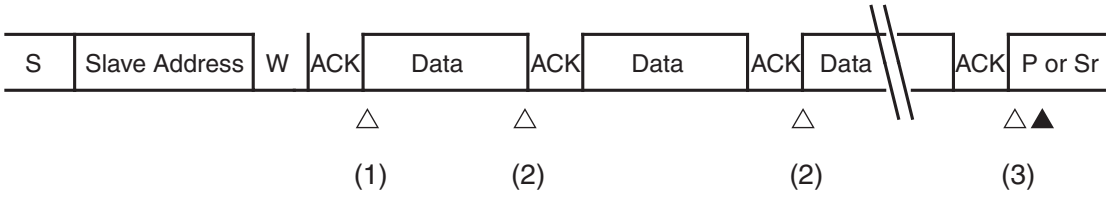
S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by 1-byte transmission
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-82. Master Transmission Interrupt 4 (WSEL=1, RSA=0, NACK Response in the Middle)



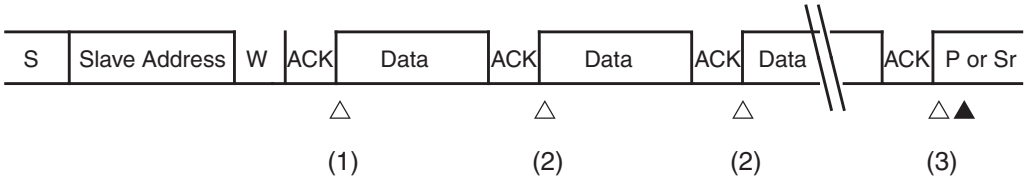
S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by NACK response.
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-83. Master Transmission Interrupt 5 (WSEL=1->"0", RSA=0, ACK Response)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by 1-byte transmission + acknowledge reception.
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-84. Master Interrupt 6 (WSEL=0, RSA=1)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address (reserved address) transmission + direction bit transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(2) An interrupt is generated by 1-byte transmission + acknowledge reception.
INT=0 writing after writing transmission data to TDR register
(3) An interrupt is generated by 1-byte transmission + acknowledge reception.
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Data Reception by Master

When the data direction bit (R/W) is “1”, data transmitted from a slave is received.

If the TDRE bit is “1”, the master generates a wait (INT=1, RDRF=1) for each 1-byte reception, and returns ACK or NACK response using the ACKE bit setting in the IBCR register by following the WSEL bit. When the TDRE bit is “0”, if ACK response is being set in the ACKE bit of the IBCR register, no wait will be generated (INT=0) and the next data is received, and if NACK response is being set, wait will be generated (INT=1).

Wait states generated by interrupts are as Table 18-46:

Table 18-46. WSEL Bit for Master Data Reception

WSEL bit	Operation
0	From the second byte, when the TDRE bit is “1” and after acknowledge, the interrupt flag (INT) is set to “1” and the SCL is set to “L” to set wait state.
1	From the second byte, when the TDRE bit is “1” and after the master receives 1-byte data, the interrupt flag (INT) is set to “1” and the SCL is set to “L” to set wait state.

An example procedure to receive data from a slave is shown below:

(1). The slave address (including data direction bit) is set in the TDR register and “1” is written to the MSS bit.

(2). After slave address transmission, ACK is received and the interrupt flag (INT) becomes “1”.

(3). The WSEL bit is updated at the same time as “0” is written to the interrupt flag (INT) to cancel the I^2C bus wait.

(4). After receiving 1 byte, the interrupt flag is set to “1” to make the I^2C bus wait after transmitting acknowledge when WSEL=0, or immediately after receiving 1 byte when WSEL=1. Steps (2) to (4) are repeated until the specified number of data is received.

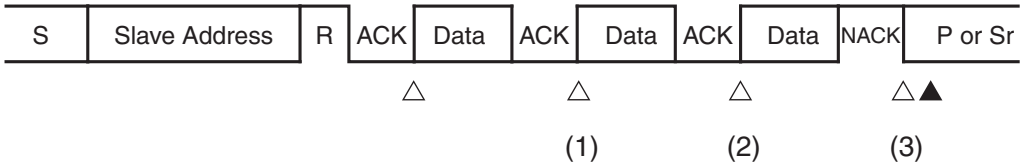
(5). After receiving the last data, NACK is outputted and the MSS bit is set to “0” or the SCC bit is set to “1” to generate a stop condition or a repeated start condition.

Notes:

When the TDRE is “0”, acknowledge is output following the ACKE bit setting even if an overrun error occurs, and the next process is executed.

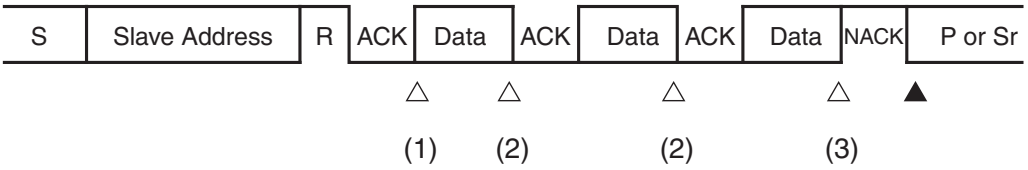
- To change the IBCR register during transmission/reception operation, do so when the interrupt flag (INT) is “1”.
- During master reception operation, write dummy data to the TDR register, and if the TDRE bit is “0” at the timing when the interrupt flag (INT) becomes “1”, the next data is received while the interrupt flag (INT) is kept to “0”.

Figure 18-85. Master Reception Interrupt 1 (WSEL=0, RSA=0)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
An interrupt is generated by writing INT=0.
(2) An interrupt is generated by 1-byte reception + acknowledge transmission.
ACKE=0 setting and INT = 0 writing after reading the received data
(3) An interrupt is generated by 1-byte reception + acknowledge transmission.
MSS=0 or MSS=1, SCC=1 setting
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Figure 18-86. Master Reception Interrupt 2 (WSEL=1, RSA=0)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated by slave address transmission + direction bit transmission + acknowledge reception.
Writing INT=0 to clear the interrupt to "0"
(2) An interrupt is generated by 1-byte reception.
INT=0 writing after reading the received data.
(3) An interrupt is generated by 1-byte reception.
After reading the received data, ACKE=0 setting and then MSS=0 or MSS=1, SCC=1 setting.
Note: TDRE bit is "1" when the interrupt flag (INT) occurs.

Arbitration Lost

If a data collision with data transmitted from another master occurs and so the master receives data that is different from the one transmitted by the master, the master determines it as arbitration lost and sets the MSS bit to “0” and the AL bit to “1” to enable the operation as slave mode.

The AL bit can be cleared to “0” using one of the following conditions:

- Writing “1” to MSS bit
- Writing “0” to INT bit
- Writing “0” to SPC bit when AL bit=1 and SPC bit=1
- I^2C interface disabled (EN bit=0)

When arbitration lost occurs, the interrupt flag (INT) is set to “1” by following the WSEL setting to set the SCL of the I^2C bus to “L”.

Wait in Master Mode

If the MSS bit is set to “1” when the BB bit is “1”, the device waits for master mode while the BB bit is “1” if it does not operate as slave mode. It transmits a start condition after the BB bit becomes “0”. To check whether the device is waiting for the master mode or not, use the MSS bit and the ACT bit (wait state if MSS=1 and ACT=0). If it is operated as slave mode after setting the MSS bit to “1”, set the AL bit to “1”, the MSS bit to “0”, and the ACT bit to “1”.

Slave Mode

If a (repeated) start condition is detected, and the combination of the ISBA register and the ISMK register and the received address are matched, the device returns ACK and operates as slave mode.

Slave Address Match Detection

When a (repeated) start condition is detected, 7 bits of the next data is received as address. For the bit with “1” setting in the ISMK register, the ISBA register and each bit of the received address are compared with each other, and ACK is output if matched.

Table 18-47. Operation Immediately After Outputting Acknowledge to Slave Address

Data direction bit (R/W)	Operation immediately after acknowledge	
	When acknowledge is ACK	When acknowledge is NACK
0	If the TDRE bit is “1”, set the INT bit to “1” to wait. If the TDRE bit is “0”, keep the INT bit to “0” and no wait.	Keep the INT bit to “0” and no wait.
1		

- Reserved address detection
  If the first byte and a reserved address (“$0000xxxx_B$” or “$1111xxxx_B$”) are matched, after receiving the 8th bit of the data, the INT bit is set to “1” to make the I^2C bus wait. The reception data is read at this time and if you want to operate it as a slave, set the ACKE to “1” to clear the INT bit. Then, the device operates as a slave. If the ACKE is set to “0”, it will not operate as a slave after outputting acknowledge.

Data Direction Bit

After receiving address, a data direction bit that specifies data transmission/reception is received. When this bit is "0", it indicates that data is transmitted from the master, so that means the slave receives the data.

Reception by Slave

If the slave addresses are matched and when the data direction bit is “0”, it indicates that reception by slave mode is performed. An example procedure to receive data in slave mode is shown below:

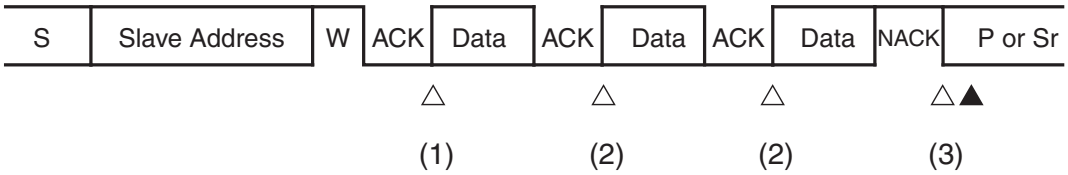
(1). After ACK transmission, the interrupt flag (INT) is set to “1” to make the I^2C bus wait. An interrupt is determined by matching with the slave address using the MSS bit, ACT bit and FBT bit, and the ACKE bit is set to “1” and the interrupt flag (INT) is set to “0” to cancel the I^2C bus wait. (See Table 18-47).

(2). After 1-byte data reception, the interrupt flag (INT) is set to “1” by following the WSEL setting to make the I^2C bus wait.

(3). The reception data is read from the RDR register and the interrupt flag (INT) is set to “0” after setting the ACKE bit to cancel the I^2C bus wait.

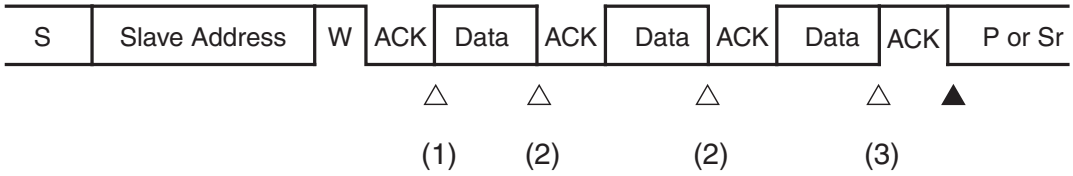
(4). The steps 2 and 3 are repeated until a stop condition or a repeated start condition is detected.

Figure 18-87. Slave Reception Interrupt 1 (WSEL=0, RSA=0)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated when slave address is matched and ACK is output.
ACKE=1, INT=0 writing
(2) An interrupt is generated by 1-byte reception + ACK response.
INT=0 writing after reading the reception data from the reception buffer
(3) An interrupt is generated by 1-byte reception + NACK response.
INT=0 writing after reading the reception data from the reception buffer

Figure 18-88. Slave Reception Interrupt 2 (WSEL=1, RSA=0)



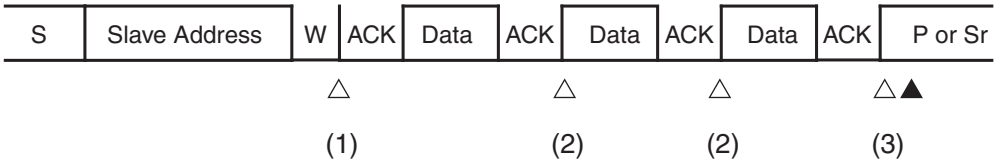
S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated when slave address is matched and ACK is output.
ACKE=1, INT=0 writing
(2) An interrupt is generated by 1-byte reception + ACK response.
INT=0 writing after reading the reception data from the reception buffer
(3) An interrupt is generated by 1-byte reception + NACK response.
INT=0 writing after reading the reception data from the reception buffer

Figure 18-89. Slave Reception Interrupt 3 (WSEL=1, RSA=0)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated when slave address is matched and ACK is output.
ACKE=1, INT=0 writing
(2) An interrupt is generated by 1-byte reception + ACK response.
INT=0 writing after reading the reception data from the reception buffer
(3) An interrupt is generated by NACK response.
INT=0 writing

Figure 18-90. Slave Reception Interrupt 6 (WSEL=0, RSA=1)



S : Start condition
W : Data direction bit (write direction)
P : Stop condition
Sr : Repeated start condition
△ : Interrupt by INTE=1
▲ : Interrupt by CNDE=1
(1) An interrupt is generated when reserved address ("$0000xxxx_B$" or "$1111xxxx_B$") is matched.
Reception data reading and ACKE=1, INT=0 writing
(2) An interrupt is generated by 1-byte reception + acknowledge output.
INT=0 writing
(3) An interrupt is generated by 1-byte reception + acknowledge output.
Interrupted by INT=0 writing

Transmission by Slave

If the slave addresses are matched and when the data direction bit is “1”, it indicates that transmission by a slave is performed. After 1-byte transmission or acknowledge response, the interrupt flag (INT) is set to “1” by the WSEL setting to generate wait. (See Table 18-47).

The acknowledge output from a master can be checked using the RACK bit. This bit indicates whether data has not been received by a master properly or data reception has been completed when NACK is returned from the master. If NACK is detected when WSEL=1, an interrupt is generated to make the bus wait.

Bus Error

A stop condition and a (repeated) start condition detected on the I^2C bus during data transmission/reception are handled as bus errors.

Bus Error Generation Condition

The BER bit is set to “1” by a bus error under the following conditions:

- Detection of a (repeated) start condition or stop condition in the first byte transfer
- Detection of a (repeated) start condition or stop condition in the 2nd bit to 9th (acknowledge) bit of the data

Bus Error Operation

When the interrupt flag (INT) becomes “1” by transmission/reception, check the BER bit. If the BER bit is “1”, perform error processing. The BER bit can be cleared by writing “0” to the INT bit.

Although the INT bit is set to “1” by a bus error, the SCL on the I^2C bus will be set to “L” and so the bus will not enter the wait state.

18.7.4 Dedicated Baud Rate Generator

The dedicated baud rate generator sets the serial clock frequency.

Baud Rate Selection

Baud rate determined by dividing an internal clock using the dedicated baud rate generator (reload counter)

The I^2C interface has two internal reload counters and each of them corresponds to transmission/reception serial clock. The baud rate can be selected by setting a 15-bit reload value in the baud rate generator registers 1 and 0 (BGR0, BGR1).

The reload counter divides an internal clock by a setting value.

Calculation of Baud Rate

Two 15-bit reload counters are set using the baud rate generator registers 1 and 0 (BGR0, BGR1).

Calculation formula of baud rate is shown below.

1. Reload value

$V = \phi/b - 1$

V: Reload value b: Baud rate ϕ: Machine clock, external clock frequency

Since the baud rate, however, may not be generated as it is being set depending on the SCL rising time on the I^2C bus, adjustment of the reload value may be needed.

2. Example of the calculation

When the machine clock is set to 16MHz, an internal clock is used, and the baud rate is set to 19200 bps, the reload value can be calculated as follows:

Reload value

$V = (16 \times 1000000)/400000 - 1 = 39$

Therefore, the baud rate is:

$b = (16 \times 1000000)/(38 + 2) = 400kbps$

Notes:

- When writing into the baud rate generator registers 1 and 0 (BGR0, BGR1), use 16-bit access.
- Set the baud rate generator register when the EN bit of the ISMK register is “0”.
- In operation mode 4 (I^2C mode), use the machine clock at 8MHz or higher and set the baud rate generator not to exceed 400 kbps.
- The reload counter will be stopped when the reload value is set to “0”.

Reload Values and Baud Rates Corresponding to Each Machine Clock Frequency

Table 18-48. Reload Values and Baud Rates

Baud Rate [bps]	8 MHz	10 MHz	16 MHz	20 MHz	24 MHz	32MHz
	Reload Value	Reload Value	Reload Value	Reload Value	Reload Value	Reload Value
400000	19	24	39	49	59	79
200000	39	49	79	99	119	159
100000	79	99	159	199	239	319

These values are for the case when SCL rising of the I^2C bus is “0”. If the SCL rising is slow, the baud rate may become slower than the above values.

Function of Reload Counter

Reload counter consists of a 15-bit register corresponding to reload values and it generates transmission/reception clock from an internal clock. It can also read count values of the transmission reload counter from the baud rate generator registers 0 and 1 (BGR0, BGR1).

Count Start

The reload counter starts counting when a reload value is written into the baud rate generator registers 0 and 1 (BGR0, BGR1).

I²C Flowchart Examples

Figure 18-91. I²C Flowchart Example 1

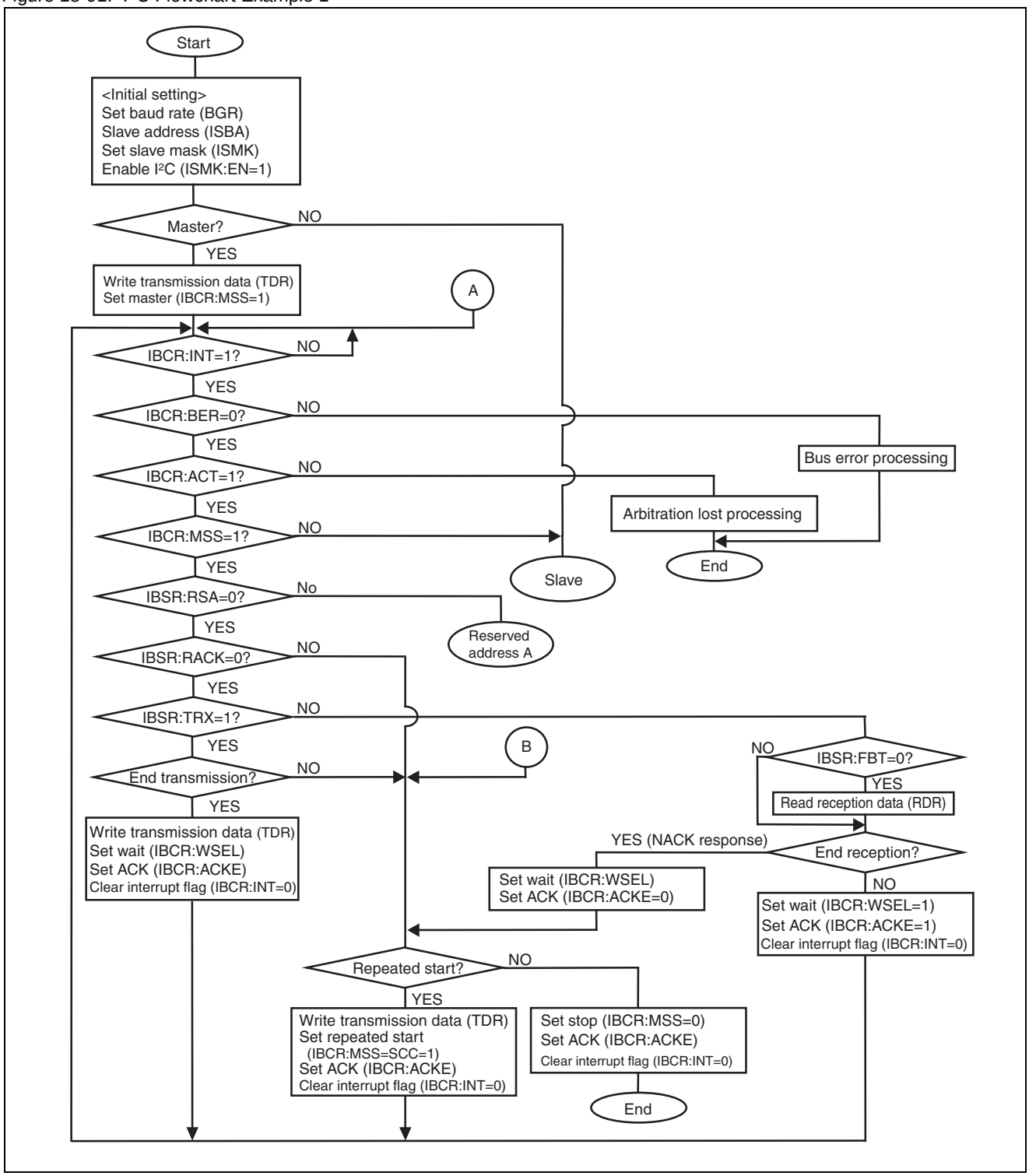


Figure 18-92. I²C Flowchart Example 2

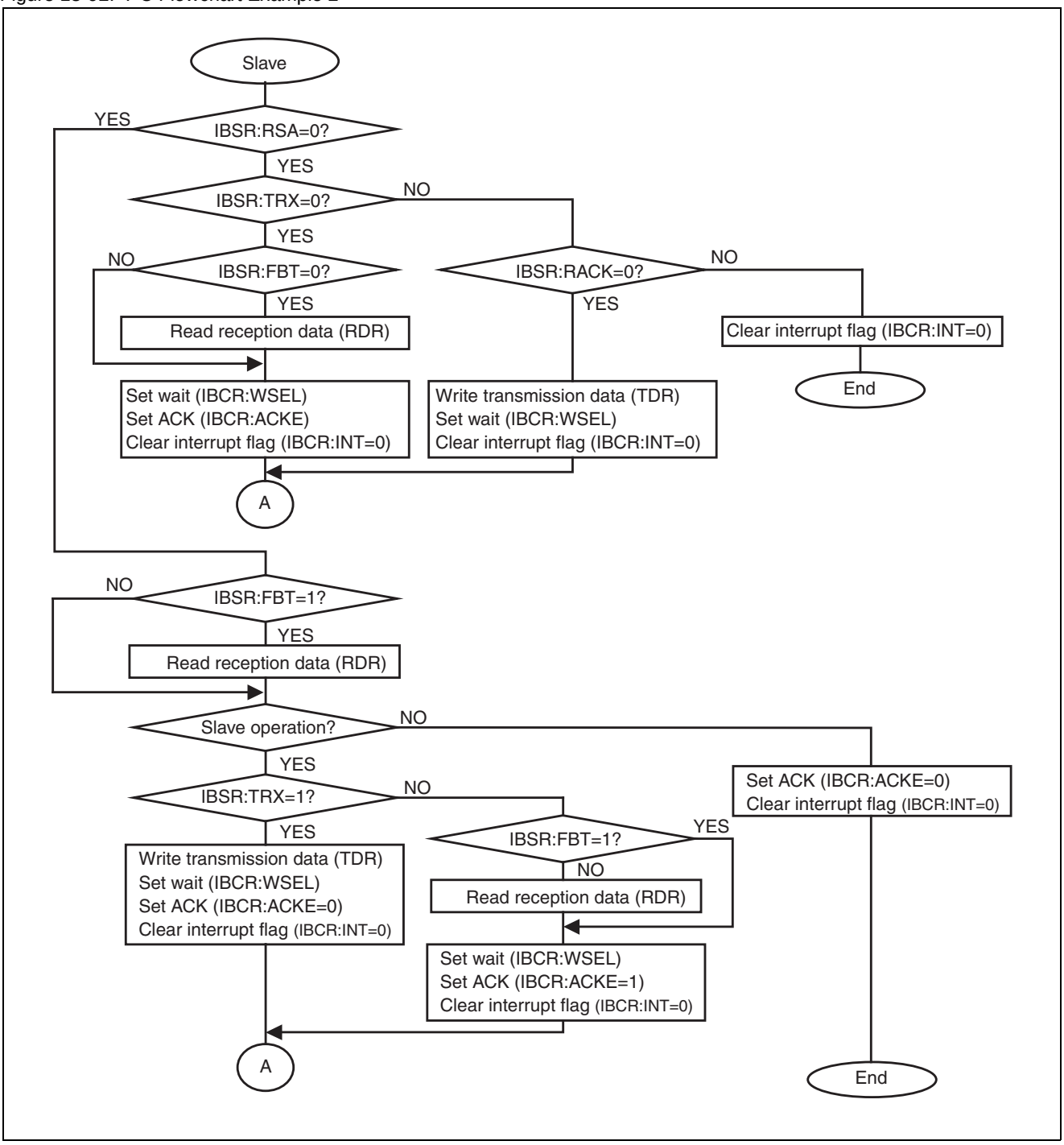
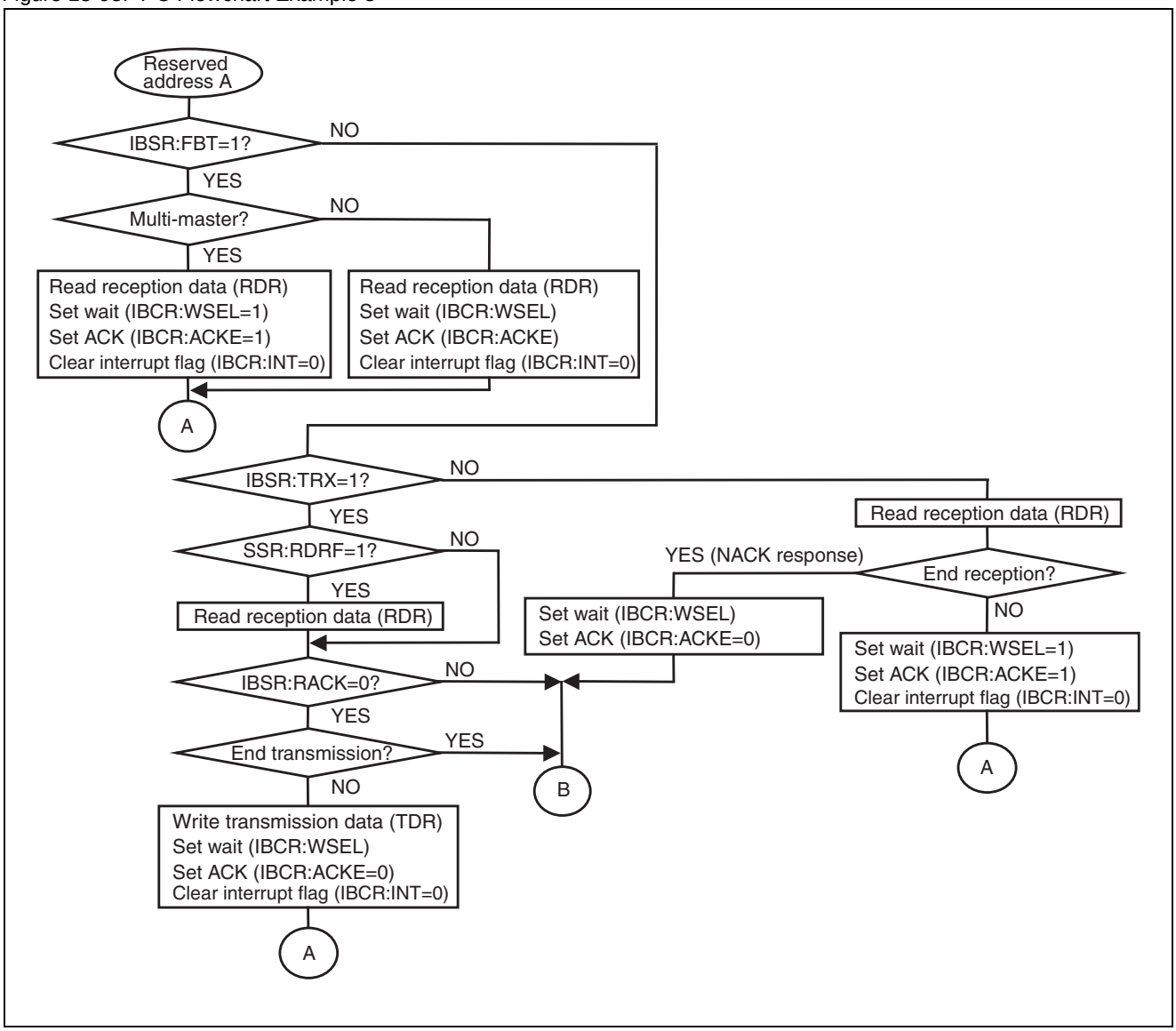


Figure 18-93. I²C Flowchart Example 3

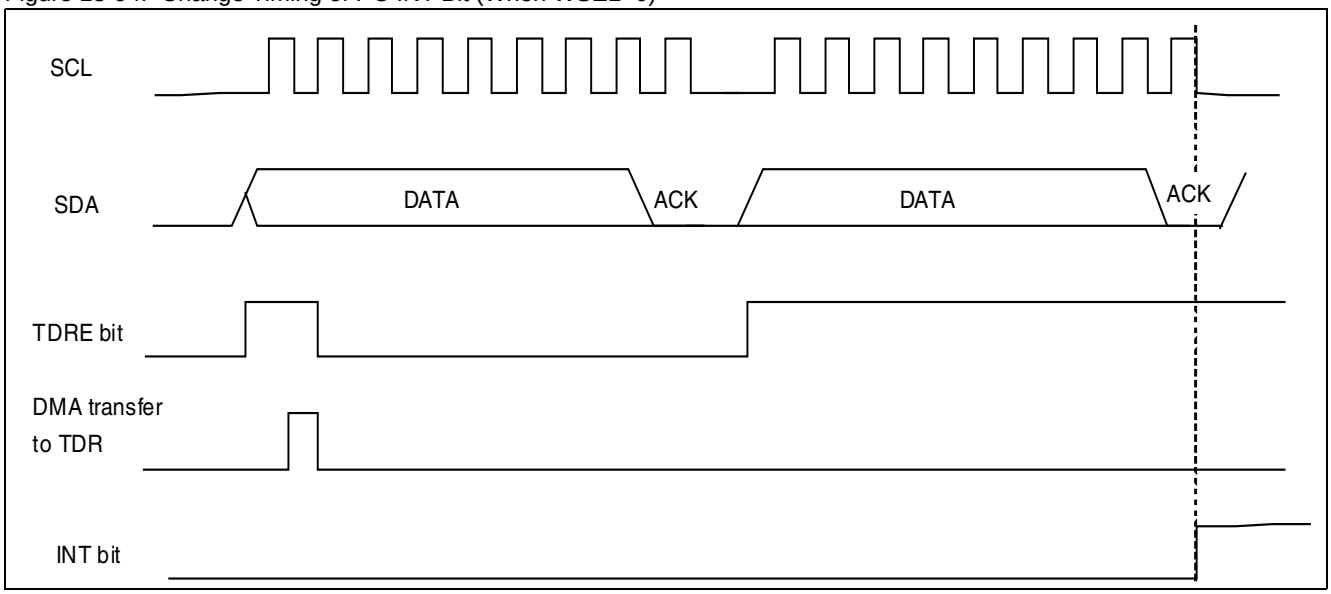


18.7.5 Precautions When Using I²C Mode

This section describes precautions to be taken when using I²C mode.

In I²C mode, interrupt flag (INT) becomes “1” as shown in Figure 18-94 when data on I²C bus are transmitted first 9 bits (when WSEL=0) or first 8 bits (when WSEL=1) with the state that there is no valid data in transmission register (TDR) and transmission data empty flag bit (TDRE) is “1”. If interrupt flag (INT) becomes “1” during DMA transfer, DMA transfer cannot keep operation unless “0” clear is performed by software (all of master transmission, slave transmission, master reception, and slave reception).

Figure 18-94. Change Timing of I²C INT Bit (When WSEL=0)



For the reason of the above specifications, program so that DMA transfer to TDR is operated before interrupt flag (INT) changes to “1”.

When writing transmission data to TDR by DMA or writing data by software after verifying transmission data empty flag (SSR:TDRE), write the data until SCL in ACK field falls because transmission empty flag (SSR:TDRE) may not change to “0”. When writing transmission data after interrupt flag (IBCR:INT) changes to “1”, there is no restrictions.

If writing transmission data delays when transmission processing is performed by DMA transfer or using transmission empty flag with software, use the following setting and procedure.

Setting

Set the timing that interrupt flag (IBCR:INT) changes to “1” at 8 bit (WSEL= “1”).

Procedure

Use the following procedures when performing transmission/reception as a master. If performing as a slave, the following procedures are not required.

1. Write the first byte (slave address) to TDR by software.
2. Activate master (write IBCR:MSS=1) and set the wait (write IBCR:WSEL=1) on 8 bit at the same time.
3. Interrupt flag (IBCR:INT) changes to “1” after transmitting the first byte. Verify the ACK response (IBSR:RACK=0) and write the second byte to TDR by software. Then, set the DMAC, activate DMA transfer and write “0” to interrupt flag (IBCR:INT).
4. Terminate the master operation (write IBCR:MSS=0) or perform re-activation (write IBCR:SCC=1) when transmission/reception has done.

19. Chip Selection Facility

This chapter provides an overview of the chip selection facility, explains the configuration, its operation, the configuration and functions of its registers.

19.1 Overview of Chip Selection Facility

19.2 Configuration of Chip Selection Facility

19.3 Configuration and Functions of Registers for Chip Selection Facility

19.4 Operation of the Chip Selection Facility

19.1 Overview of Chip Selection Facility

The chip selection facility is a module used to generate a chip selection signal for simplified memory connection to the outside. It contains four chip selection output pins, and enables an area within the hardware to be specified via an output setting register, and if the device detects an access to that external address, it outputs a selection signal via the corresponding pin.

Overview of the Chip Selection Facility

The chip selection facility contains two 8-bit registers for each output pin. One register (CAR0 to CAR3) is used to specify the upper 8 bits of the compared address, allowing area within 64 Kbytes to be specified. Another register (CMR0 to CMR3) is used to mask the compared bit so that values above 64 Kbytes can be specified for the area to be detected.

In external bus hold mode, CS output is set to high impedance mode.

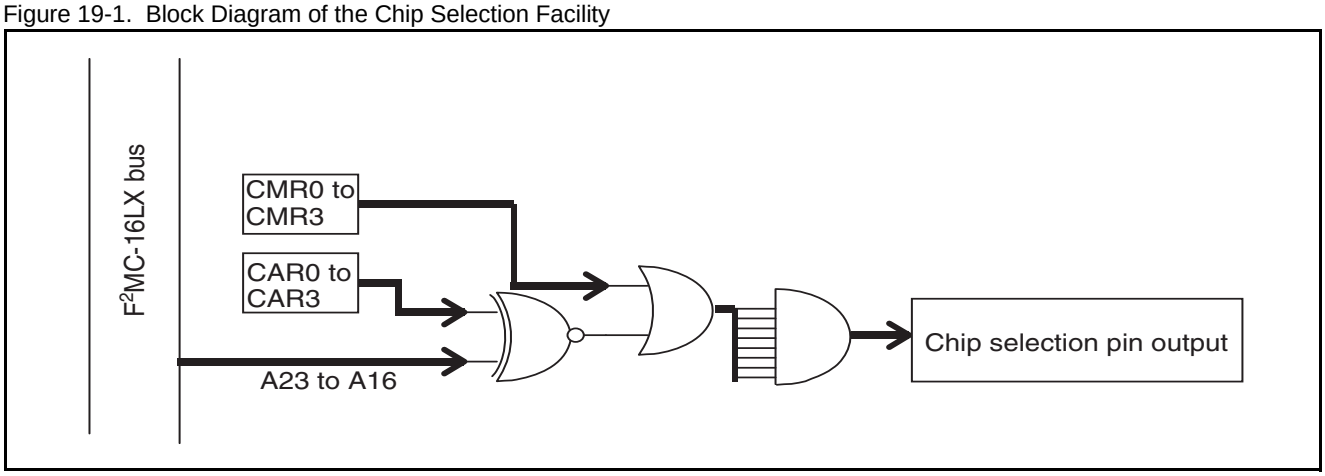
19.2 Configuration of Chip Selection Facility

This section describes the block diagram and pin related to the chip selection facility.

Block Diagram of the Chip Selection Facility

Figure 19-1 shows a block diagram of the chip selection facility.

Figure 19-1. Block Diagram of the Chip Selection Facility



Pin Related to Chip Selection Facility

The pin related to the chip selection facility has four CS0/CS1/CS2/CS3 output pins. CS0/CS1/CS2/CS3 pins share the general-purpose I/O ports (P90/AN8/CS0, P91/AN9/CS1, P92/AN10/CS2 and P93/AN11/CS3), analog input pin of A/D, and the output pin of the chip selection facility.

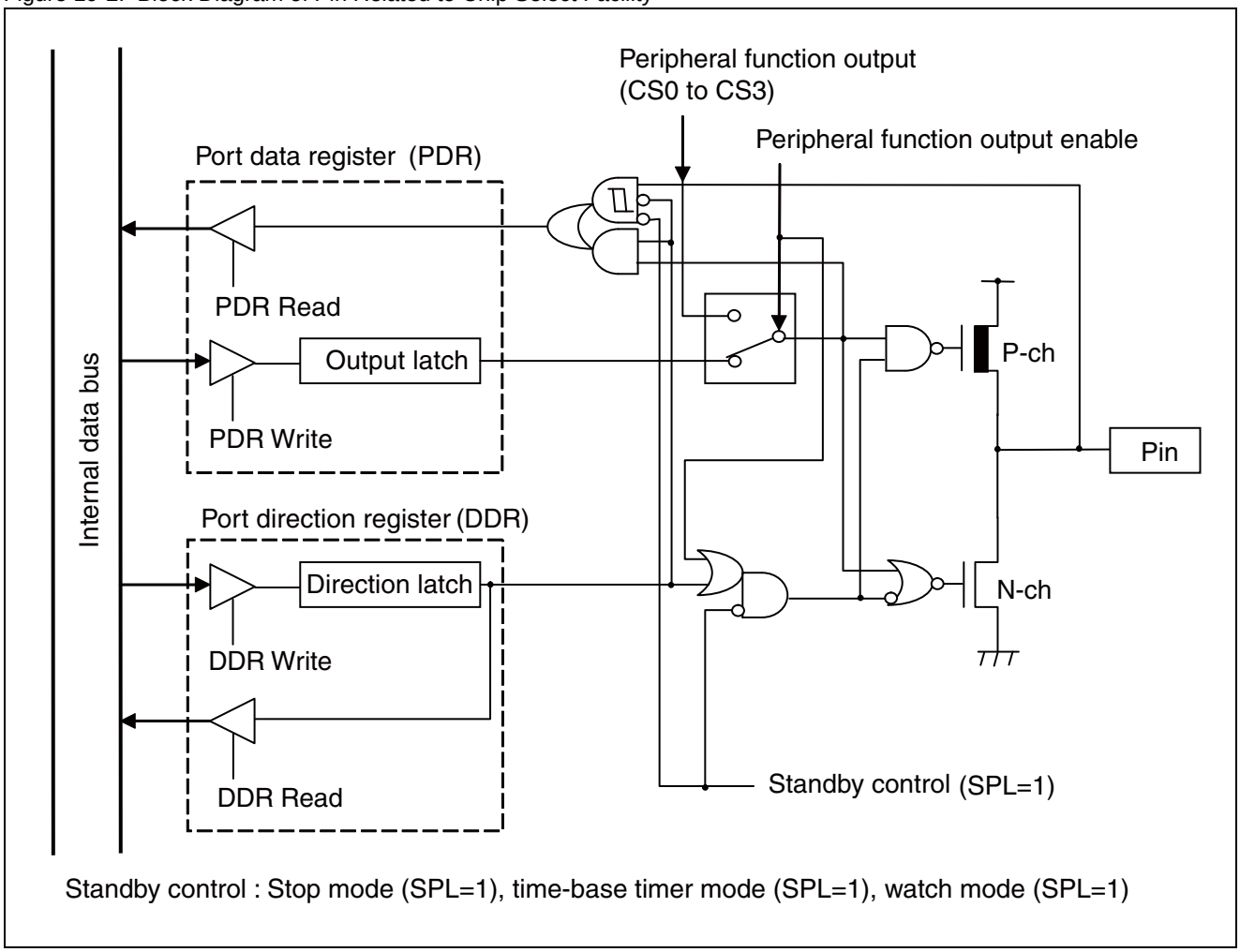
Setting when using as CS0/CS1/CS2/CS3 pins

When the CS0/CS1/CS2/CS3 are used as output by the chip selection facility, be sure to set the chip selection control register (CSCR) to enable output (OPL [3:0] → “1”).

Also, other resource that is assigned to these pins cannot be used when the chip selection facility is used.

Block Diagram of Pin Related to Chip Select Facility

Figure 19-2. Block Diagram of Pin Related to Chip Select Facility



19.3 Configuration and Functions of Registers for Chip Selection Facility

This section describes the configuration and functions of the registers used by the chip selection facility.

List of Registers Used for the Chip Selection Facility

Figure 19-3 lists the registers for the chip selection facility.

Figure 19-3. List of Registers for the Chip Selection Facility

bit 15 8 7 0

CAR0	CMR0	(R/W)
CAR1	CMR1	(R/W)
CAR2	CMR2	(R/W)
CAR3	CMR3	(R/W)
CALR	CSCR	(R/W)

	bit	7	6	5	4	3	2	1	0	
$0000C0_H$										CMR0 to CMR3
$0000C2_H$		M7	M6	M5	M4	M3	M2	M1	M0	Chip selection area MASK registers
$0000C4_H$		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
$0000C6_H$		(0)	(0)	(0)	(0)	(1)	(1)	(1)	(1)	Initial value
$0000C1_H$	bit	15	14	13	12	11	10	9	8	CAR to CAR3
$0000C3_H$		A7	A6	A5	A4	A3	A2	A1	A0	Chip selection area registers
$0000C5_H$		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(W)	(R/W)	(R/W)	Read/Write
$0000C7_H$		(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Initial value
	bit	7	6	5	4	3	2	1	0	CSCR
$0000C8_H$		-	-	-	-	OPL3	OPL2	OPL1	OPL0	Chip selection control register
		(-)	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
		(-)	(-)	(-)	(-)	(0)	(0)	(0)	(*)	Initial value
	bit	15	14	13	12	11	10	9	8	CALR
$0000C9_H$		-	-	-	-	ACTL3	ACTL2	ACTL1	ACTL0	Chip selection active level register
		(-)	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
		(-)	(-)	(-)	(-)	(0)	(0)	(0)	(0)	Initial value

*: The initial value of this bit is "1" or "0". The value depends on the mode pins (MD2, MD1, and MD0 pins).

19.3.1 Chip Selection Area MASK Registers (CMR0 to CMR3)

This section describes the configuration and functions of the chip selection area MASK registers (CMR0 to CMR3).

Chip Selection Area MASK Registers (CMR0 to CMR3)

The diagram below shows the bit configuration of the chip selection area MASK registers (CMR0 to CMR3).

$0000C0_H$	bit	7	6	5	4	3	2	1	0	CMR0 to CMR3
$0000C2_H$		M7	M6	M5	M4	M3	M2	M1	M0	Chip selection area MASK register
$0000C4_H$		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
$0000C6_H$		(0)	(0)	(0)	(0)	(1)	(1)	(1)	(1)	Initial value

[bit7 to bit0] M7 to M0

These bits are used to specify an address decode area for the chip selection pin. Set the corresponding bits to "1" for masking.

These bits are used to specify an area of 128 Kbytes or more.

Note:

If all bits are masked, the CS pin becomes active using all external accessible areas.

19.3.2 Chip Selection Area Registers (CAR0 to CAR3)

This section describes the configuration and functions of the chip selection area registers (CAR0 to CAR3).

Chip Selection Area Registers (CAR0 to CAR3)

The diagram below shows the bit configuration of the chip selection area registers (CAR0 to CAR3).

$0000C1_H$	bit	15	14	13	12	11	10	9	8	CAR0 to CAR3
$0000C3_H$		A7	A6	A5	A4	A3	A2	A1	A0	Chip select area register
$0000C5_H$		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(W)	(R/W)	(R/W)	Read/Write
$0000C7_H$		(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	Initial value

[bit15 to bit8] A7 to A0

These bits are used to set the address decode area for the chip selection pin. They specify the upper 8 bits of the address value, allowing an area within 64 Kbytes to be specified.

Note:

The CS pin is not set to active while CPU is performing internal access (such as built-in RAM, built-in ROM, and I/O).

19.3.3 Chip Selection Control Register (CSCR)

This section describes the configuration and functions of the chip selection control register (CSCR).

Chip Selection Control Register (CSCR)

The diagram below shows the bit configuration of the chip selection control register (CSCR).

bit	7	6	5	4	3	2	1	0	CSCR
$0000C8_H$	-	-	-	-	OPL3	OPL2	OPL1	OPL0	Chip selection control register
	(-)	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
	(-)	(-)	(-)	(-)	(0)	(0)	(0)	(*)	Initial value

*: The initial value of this bit is “1” or “0”. The value depends on the mode pins (MD2, MD1, and MD0 pins).

[bit7 to bit4] Unused bits

These bits are unused. Read values are undefined.

[bit3 to bit0] OPL3 to OPL0

These bits are used to specify whether CS3 to CS0 are output to the external pin.

The operational settings are as follows:

- “0”: Decode output from each CS3 to CS0 pin is prohibited
- “1”: Decode output from each CS3 to CS0 pin is allowed

Notes:

- The initial value of OPL0 is set to “1” in external vector mode, and set to “0” in internal vector mode.
- After all settings have been made if output of the CS3 to CS0 pins is enabled, finally enable the OPL3 to OPL0 to output.
- Change settings during operation only after prohibiting OPL3 to OPL0 to output.

19.3.4 Chip Selection Active Level Register (CALR)

This section describes the configuration and functions of the chip selection active level register (CALR).

Chip Selection Active Level Register (CALR)

The diagram below shows the bit configuration of the chip selection active level register (CALR).

bit	15	14	13	12	11	10	9	8	CALR
0000C9$_H$	-	-	-	-	ACTL3	ACTL2	ACTL1	ACTL0	Chip selector active level register
	(-)	(-)	(-)	(-)	(R/W)	(R/W)	(R/W)	(R/W)	Read/Write
	(-)	(-)	(-)	(-)	(0)	(0)	(0)	(0)	Initial value

[bit15 to bit12] Unused bits

These bits are unused. Read values are undefined.

[bit11 to bit8] ACTL3 to ACTL0

These bits set the active level of each CS3 to CS0 pin.

The followings are set.

- “0”: Each CS3 to CS0 pin outputs “L” after decoding.
- “1”: Each CS3 to CS0 pin outputs “H” after decoding.

Notes:

- Before changing the active level, prohibit output via the chip selection control register.
- Writing these bits in units of words is prohibited. Always write these bits in units of bytes. This will prevent the enabling of output with the write operation at the same time that the active level is changed.

19.4 Operation of the Chip Selection Facility

This section describes the operations of the chip selection facility.

Outline of Operations

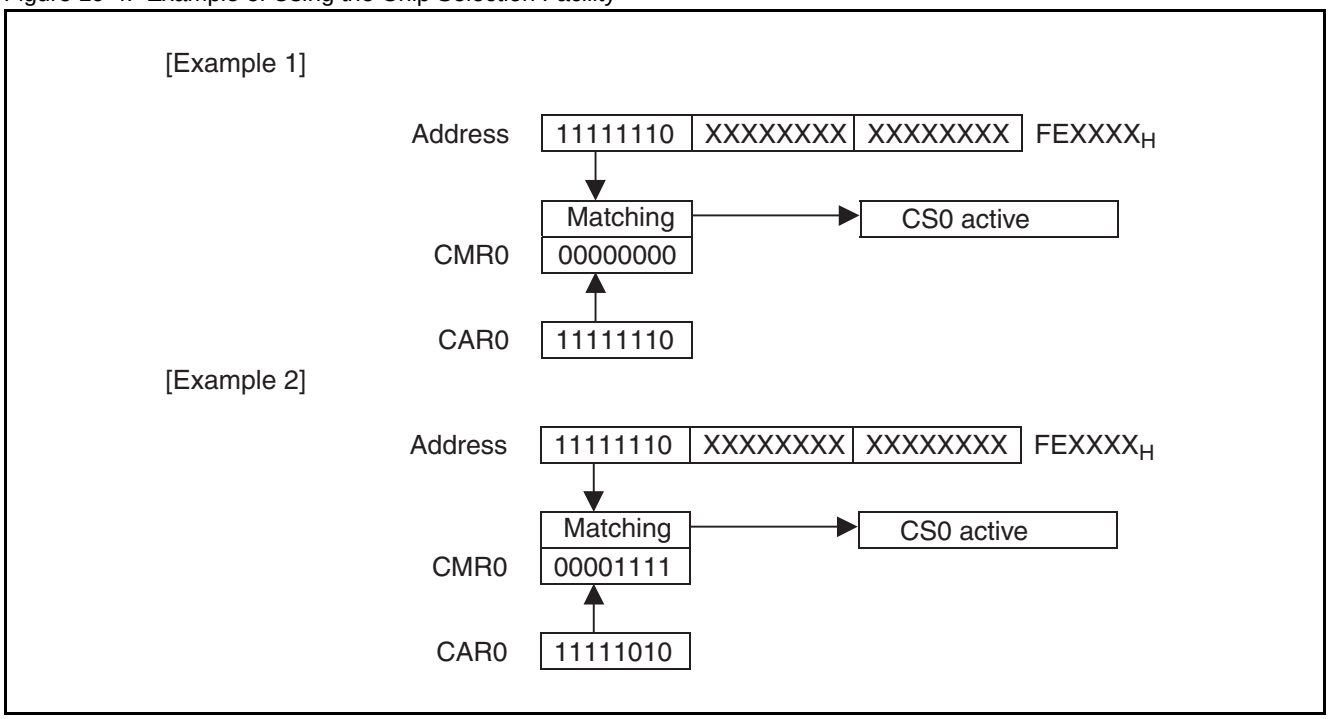
When the CPU accesses program or data, the chip selection facility is activated if a match between the upper 8 bits of an address and CAR0 to CAR3 is detected. Addresses for which the corresponding bits in CMR0 to CMR3 are set to "1" are ignored, and decoding becomes possible for an area from 64 Kbytes to 16 MB.

The CS pin is not set to active while CPU is performing internal access (such as built-in RAM, built-in ROM, and I/O).

Example of Using the Chip Selection Facility

Figure 19-4 shows an example of using the chip selection facility.

Figure 19-4. Example of Using the Chip Selection Facility



Notes on Using the Chip Selection Facility

- The CS0 pin always becomes active to read the reset vector if used in external vector mode. In the address space $F00000_H$ to $FFFFFF_H$ (1 MB: initial value), always use this pin only for the access to program ROM, since the corresponding decode signal will be output immediately after a reset. In this case, the active level of the CS0 pin is set to "L", and "H" is output at reset. In internal vector mode, this pin, like the CS3 to CS1 pins, is used as a general-purpose port. Therefore, switch the CS0 pin to the output state after the corresponding settings have been made.
- Only enable output after the settings of the chip selection area register, chip selection area MASK register, and chip selection active level register have been made.
- Note that, since the chip selection output is shared with pins P90 to P93, chip selection output will not be available while the resource assigned to these pins is in use.
- If the pin is set to the hold state when the external bus is used, output is disabled and the pin is set to high impedance state. In these cases, always set the shared general-purpose port to act as an input.
- In sleep or stop mode, the CS pin is not active.
- The chip selection facility cannot be used during access to built-in DMA.

20. Address Match Detection Function

This chapter explains the functions and operations of the address match detection.

20.1 Overview of Address Match Detection Function

20.2 Block Diagram of Address Match Detection Function

20.3 Configuration for Address Match Detection Function

20.4 Explanation of Operation of Address Match Detection Function

20.1 Overview of Address Match Detection Function

Address match detection function is a function that replaces the next instruction to be executed in program to INT9 instruction and branches to interrupt process program when the address of next instruction to be executed in the program is met with the address set in the detection address set register. Processing with INT9 interrupt enables to take advantage of correction by batch process in a program.

Overview of Address Match Detection Function

- The address of a instruction to be executed next to current instruction in a program is held in an address latch continuously through internal data bus. Address match detection function enables the comparison between the value of the address held in the latch and the address set in the detection address set register, constantly. If the addresses are met, the instruction to be executed next by CPU will be replaced by INT9 instruction and interrupt process program will be executed.
- Detection address set register consists of 6 registers (PADR0 to PADR5) and interrupt enable bit is provided for each register. The generation of an interrupt due to a match between the address held in the address latch and the address set in the detection address setting registers can be enabled and disabled for each register.

20.2 Block Diagram of Address Match Detection Function

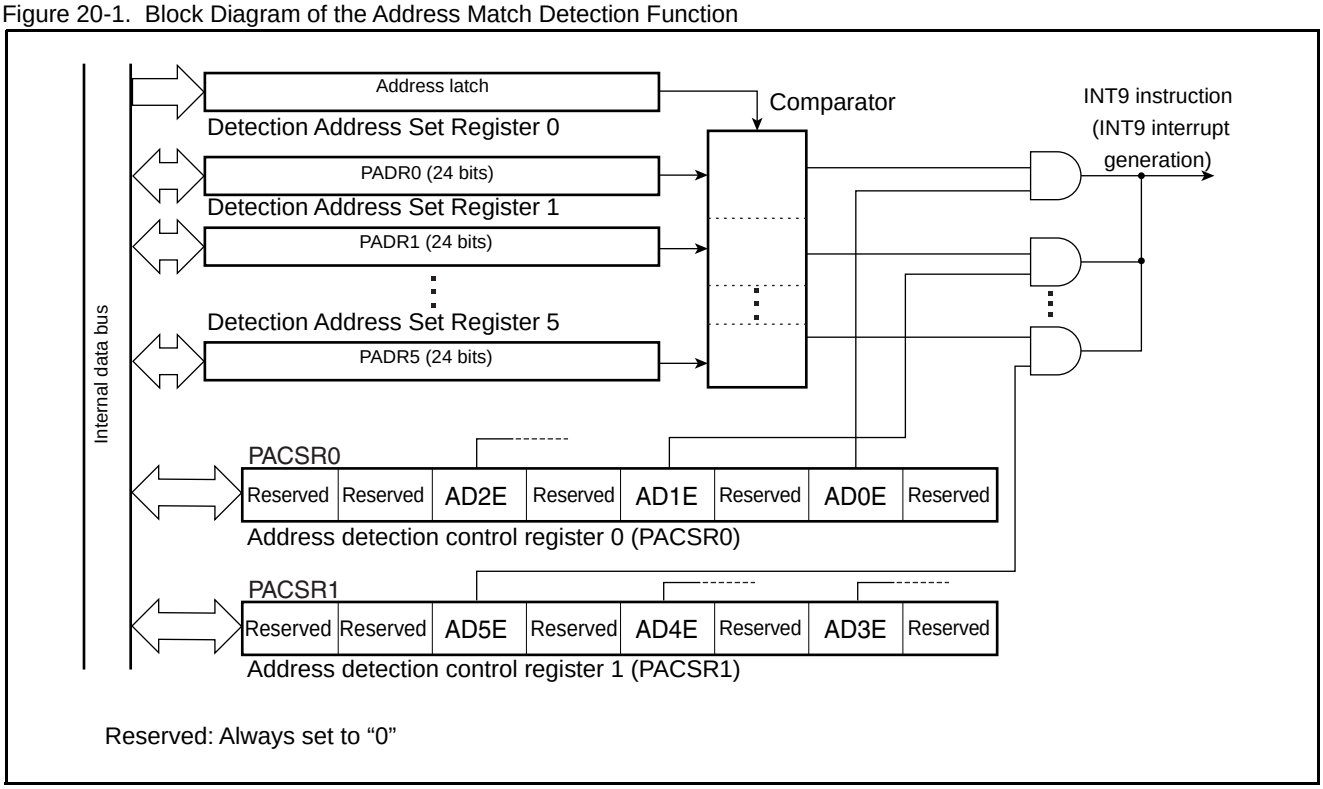
The address match detection module consists of the following blocks:

- Address latch
- Address detection control registers (PACSR0/PACSR1)
- Detection address set registers (PADR0 to PADR5)

Block Diagram of Address Match Detection Function

Figure 20-1 shows the block diagram of the address match detection function.

Figure 20-1. Block Diagram of the Address Match Detection Function



Address latch

The address latch stores the value of the address output to the internal data bus.

Address Detection Control Registers (PACSR0/PACSR1)

The address detection control register enable or disable output of an interrupt at an address match.

Detection Address Set Registers (PADR0 to PADR5)

The detection address set registers set the address that is compared with the value of the address latch.

20.3 Configuration for Address Match Detection Function

This section lists and details the registers used by the address match detection function.

List of Registers and Initial Values of Address Match Detection Function

Figure 20-2. List of Registers and Initial Values of Address Match Detection Function

	bit 7	6	5	4	3	2	1	0
Address Detection Control Register 0 (PACSR0)	0	0	0	0	0	0	0	0

	bit 15	14	13	12	11	10	9	8
Address Detection Control Register 1 (PACSR1)	0	0	0	0	0	0	0	0

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 0 (PADR0): Lower	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 0 (PADR0): Middle	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 0 (PADR0): Upper	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 1 (PADR1): Lower	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 1 (PADR1): Middle	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 1 (PADR1): Upper	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 2 (PADR2): Lower	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 2 (PADR2): Middle	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 2 (PADR2): Upper	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 3 (PADR3): Lower	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 3 (PADR3): Middle	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 3 (PADR3): Upper	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 4 (PADR4): Lower	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 4 (PADR4): Middle	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 4 (PADR4): Upper	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 5 (PADR5): Lower	×	×	×	×	×	×	×	×

	bit 15	14	13	12	11	10	9	8
Detection Address Set Register 5 (PADR5): Middle	×	×	×	×	×	×	×	×

	bit 7	6	5	4	3	2	1	0
Detection Address Set Register 5 (PADR5): Upper	×	×	×	×	×	×	×	×

×: Undefined

20.3.1 Address Detection Control Registers (PACSR0/PACSR1)

The address detection control registers (PACSR0/PACSR1) enable or disable output of an interrupt at an address match. If an address match is detected when output of an interrupt at an address match is enabled, the INT9 interrupt is output.

Address Detection Control Register 0 (PACSR0)

Figure 20-3. Address Detection Control Register 0 (PACSR0)

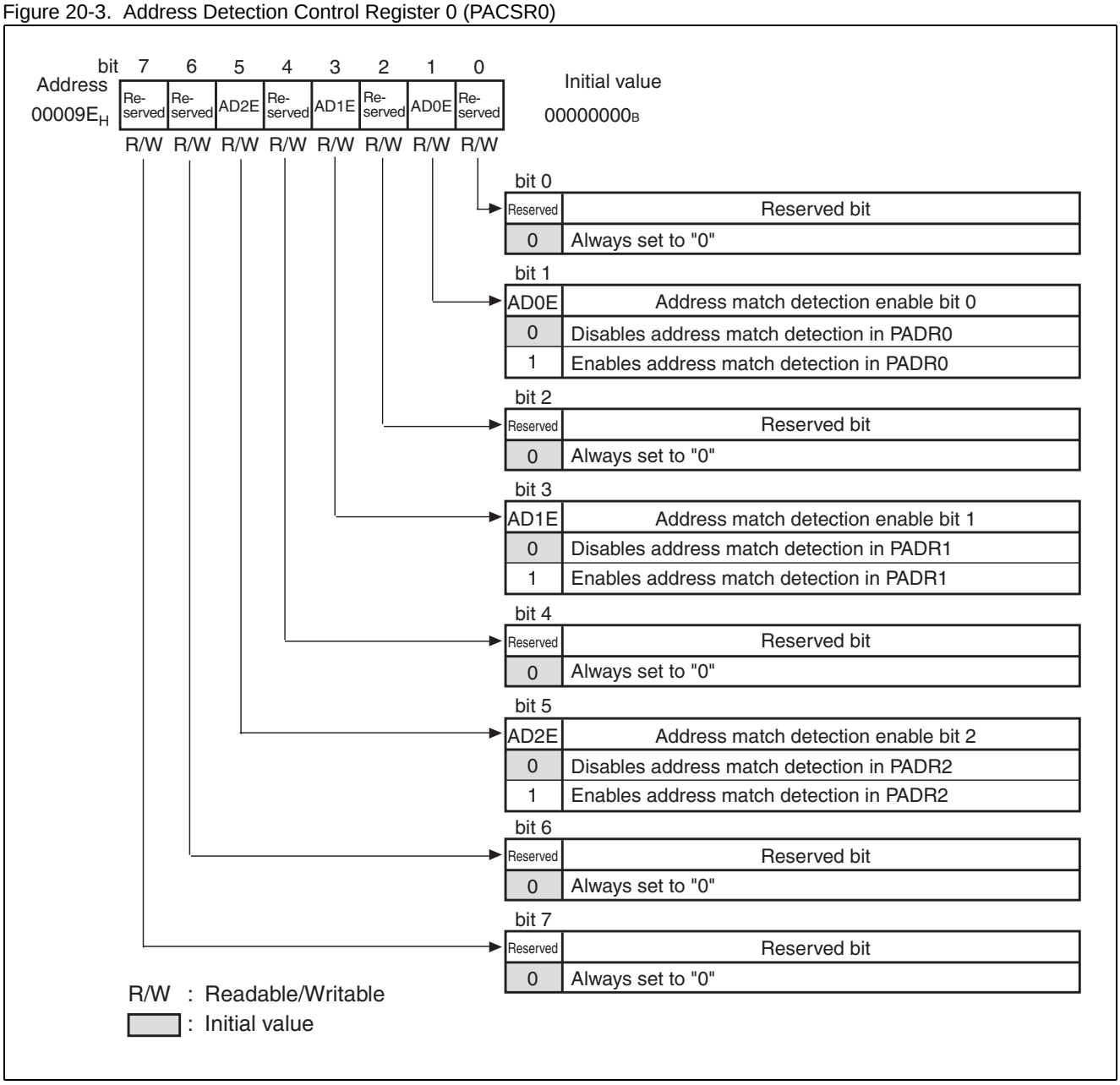


Table 20-1. Functions of Address Detection Control Register (PACSR0)

Bit Name		Function
bit7, bit6	Reserved: reserved bits	Always set to “0”.
bit5	AD2E: Address match detection enable bit2	The address match detection operation with the detection address set register 2 (PADR2) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 2 (PADR2) matches with the value of address latch at enabling the address match detection operation (AD2E = 1), the INT9 instruction is immediately executed.
bit4	Reserved: reserved bit	Always set to “0”.
bit3	AD1E: Address match detection enable bit 1	The address match detection operation with the detection address set register 1 (PADR1) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 1 (PADR1) matches with the value of address latch at enabling the address match detection operation (AD1E = 1), the INT9 instruction is immediately executed.
bit2	Reserved: reserved bit	Always set to “0”.
bit1	AD0E: Address match detection enable bit 0	The address match detection operation with the detection address set register 0 (PADR0) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 0 (PADR0) matches with the value of address latch at enabling the address match detection operation (AD0E = 1), the INT9 instruction is immediately executed.
bit0	Reserved: reserved bit	Always set to “0”.

Address Detection Control Register 1 (PACSR1)

Figure 20-4. Address Detection Control Register 1 (PACSR1)

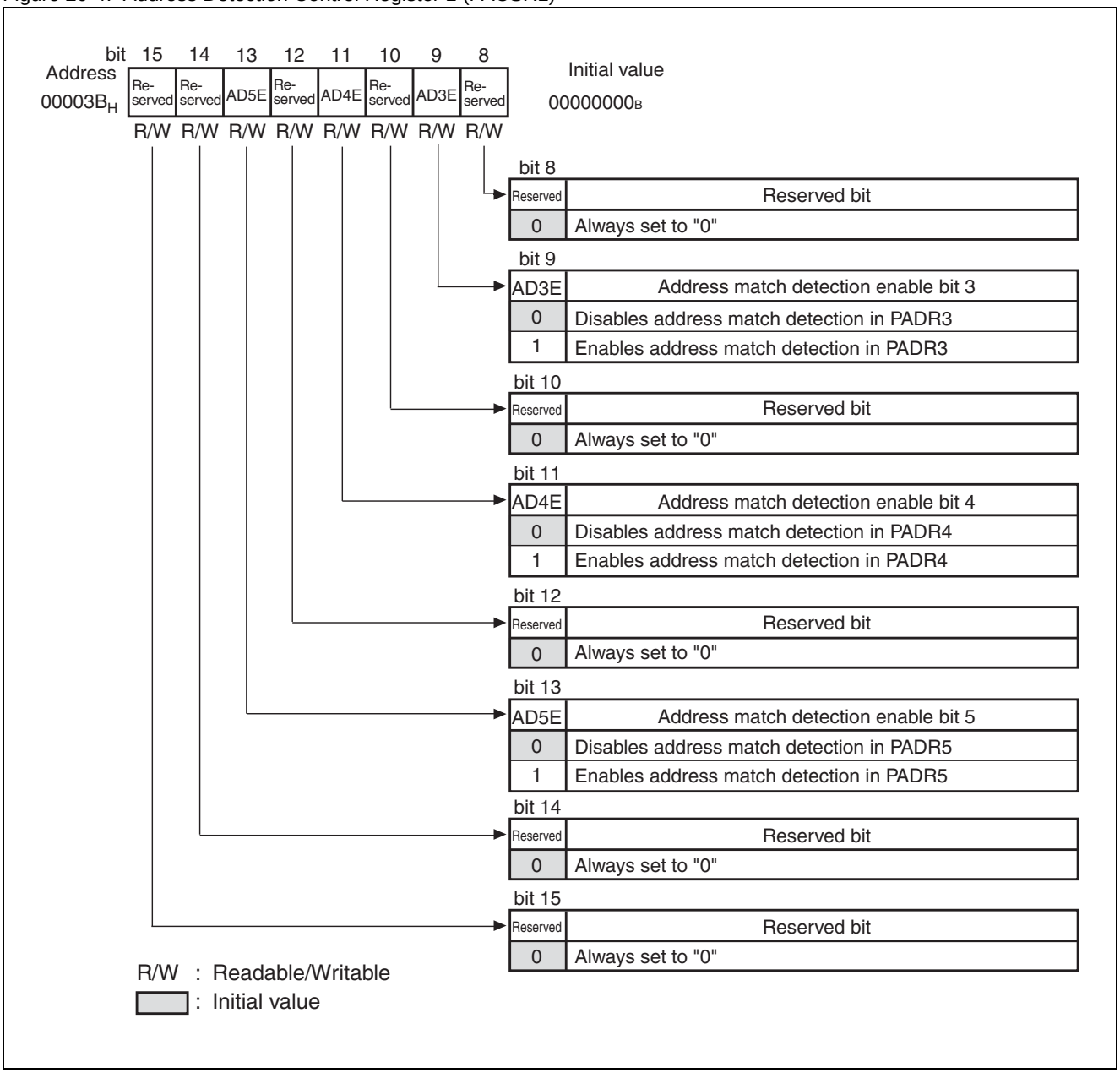


Table 20-2. Functions of Address Detection Control Register 1 (PACSR1)

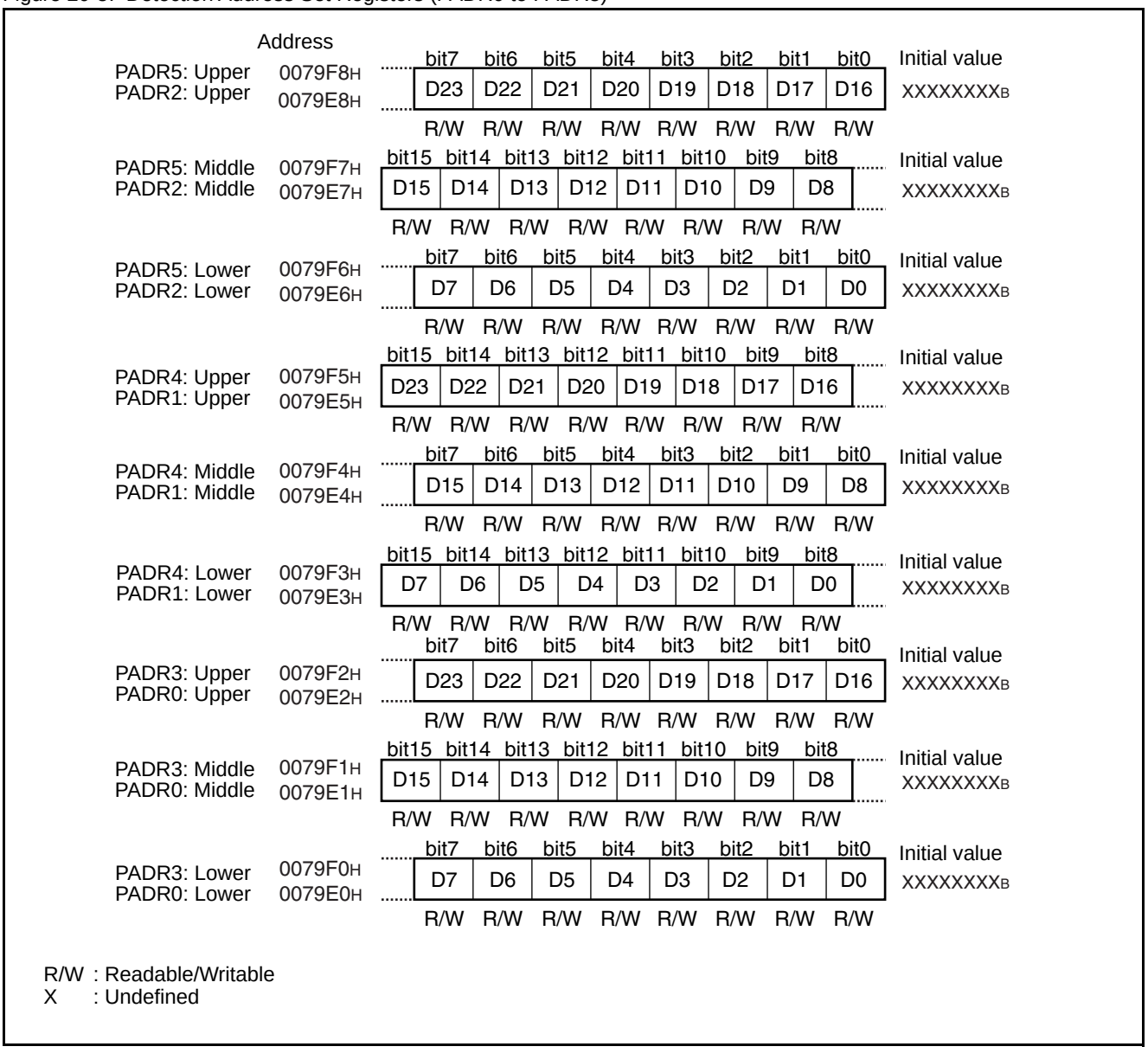
Bit Name		Function
bit15, bit14	Reserved: reserved bits	Always set to “0”.
bit13	AD5E: Address match detection enable bit 5	The address match detection operation with the detection address set register 5 (PADR5) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 5 (PADR5) matches with the value of address latch at enabling the address match detection operation (AD5E = 1), the INT9 instruction is immediately executed.
bit12	Reserved: reserved bit	Always set to “0”.
bit11	AD4E: Address match detection enable bit 4	The address match detection operation with the detection address set register 4 (PADR4) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 4 (PADR4) matches with the value of address latch at enabling the address match detection operation (AD4E = 1), the INT9 instruction is immediately executed.
bit10	Reserved: reserved bit	Always set to “0”.
bit9	AD3E: Address match detection enable bit 3	The address match detection operation with the detection address set register 3 (PADR3) is enabled or disabled. **When set to “0”:** Disables the address match detection operation. **When set to “1”:** Enables the address match detection operation. When the value of detection address set register 3 (PADR3) matches with the value of address latch at enabling the address match detection operation (AD3E = 1), the INT9 instruction is immediately executed.
bit8	Reserved: reserved bit	Always set to “0”.

20.3.2 Detection Address Set Registers (PADR0 to PADR5)

The value of an address to be detected is set in the detection address set registers. When the address of the instruction processed by the program matches the address set in the detection address set registers, the next instruction is forcibly replaced by the INT9 instruction, and the interrupt processing program is executed.

Detection Address Set Registers (PADR0 to PADR5)

Figure 20-5. Detection Address Set Registers (PADR0 to PADR5)



R/W : Readable/Writable
X : Undefined

Functions of Detection Address Set Registers

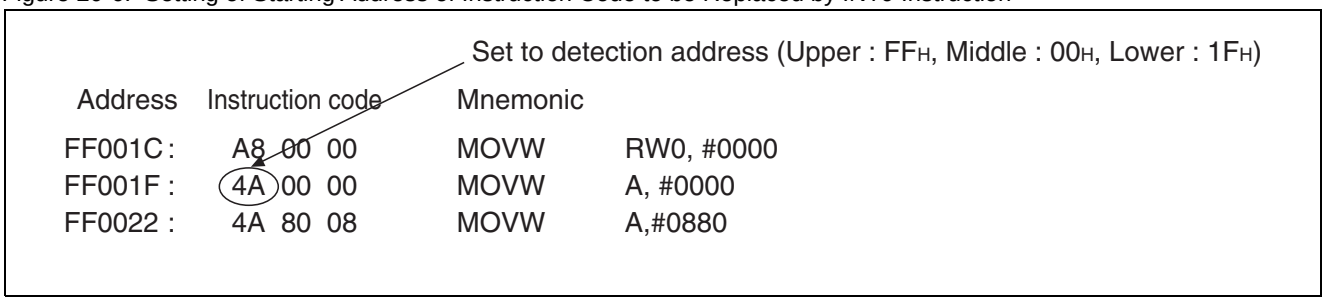
- Detection address set registers consist of 6 registers (PADR0 to PADR5) and they are 3 bytes: upper (bank), middle and lower, and 24 bits total.

Table 20-3. Address Setting of Detection Address Set Registers

Register Name	Interrupt Output Enable	Address Setting	
Detection Address Set Register 0 (PADR0)	PACSR0: AD0E	Upper	Set the upper 8 bits of detection address 0 (bank).
		Middle	Set the middle 8 bits of detection address 0.
		Lower	Set the lower 8 bits of detection address 0.
Detection Address Set Register 1 (PADR1)	PACSR0: AD1E	Upper	Set the upper 8 bits of detection address 1 (bank).
		Middle	Set the middle 8 bits of detection address 1.
		Lower	Set the lower 8 bits of detection address 1.
Detection Address Set Register 2 (PADR2)	PACSR0: AD2E	Upper	Set the upper 8 bits of detection address 2 (bank).
		Middle	Set the middle 8 bits of detection address 2.
		Lower	Set the lower 8 bits of detection address 2.
Detection Address Set Register 3 (PADR3)	PACSR1: AD3E	Upper	Set the upper 8 bits of detection address 3 (bank).
		Middle	Set the middle 8 bits of detection address 3.
		Lower	Set the lower 8 bits of detection address 3.
Detection Address Set Register 4 (PADR4)	PACSR1: AD4E	Upper	Set the upper 8 bits of detection address 4 (bank).
		Middle	Set the middle 8 bits of detection address 4.
		Lower	Set the lower 8 bits of detection address 4.
Detection Address Set Register 5 (PADR5)	PACSR1: AD5E	Upper	Set the upper 8 bits of detection address 5 (bank).
		Middle	Set the middle 8 bits of detection address 5.
		Lower	Set the lower 8 bits of detection address 5.

- In the detection address set registers (PADR0 to PADR5), starting address (first byte) of instruction to be replaced by INT9 instruction should be set.

Figure 20-6. Setting of Starting Address of Instruction Code to be Replaced by INT9 Instruction



Notes:

- When an address of byte other than the first byte is set to the detection address set registers (PADR0 to PADR5), the instruction code is not replaced by INT9 instruction and a program of an interrupt processing is not performed. When the address is set to the second byte or subsequent, the address set by the instruction code is replaced by "01_H" (INT9 instruction code) and, which may cause malfunction.
- The detection address set registers (PADR0 to PADR5) should be set after disabling the address match detection of corresponding address match control registers (PACSR: ADnE=0). If the detection address set registers are changed without disabling the address match detection, the address match detection function will work immediately after an address match occurs during writing address, which may cause malfunction.
- The address match detection function can be used only for addresses of the internal ROM area. If addresses of the external memory area are set, the address match detection function will not work and the INT9 instruction will not be executed.

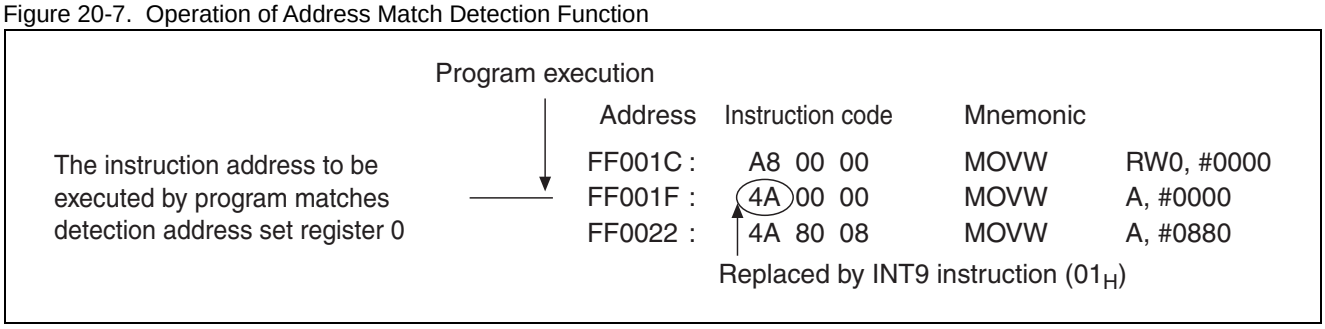
20.4 Explanation of Operation of Address Match Detection Function

If the addresses of the instructions executed in the program match those set in the detection address set registers (PADR0 to PADR5), the address match detection function will replace the first instruction code executed by CPU with the INT9 instruction (01_H) to branch to the interrupt processing program.

Operation of Address Match Detection Function

Figure 20-7 shows the operation when the detection address set and an address match is detected.

Figure 20-7. Operation of Address Match Detection Function



Setting Detection Address

1. Disable the detection address set register 0 (PADR0) where the detect address is set for address match detection (PACSR0: AD0E = 0).
2. Set the detected address in the detection address set register 0 (PADR0). Set "FF_H" at the upper bits, "00_H" at the middle bits, and "$1F_H$" at the lower bits of the detection address set register 0 (PADR0).
3. Enable the detection address set register 0 (PADR0) where the detection address is set for address match detection (PACSR0: AD0E = 1).

Program Execution

1. If the address of the instruction to be executed in the program matches the set detect address, the first instruction code at the matched address is replaced by the INT9 instruction code (01_H).
2. INT9 instruction is executed. INT9 interrupt will be generated and interrupt process program will be executed.

20.4.1 Example of Using Address Match Detection Function

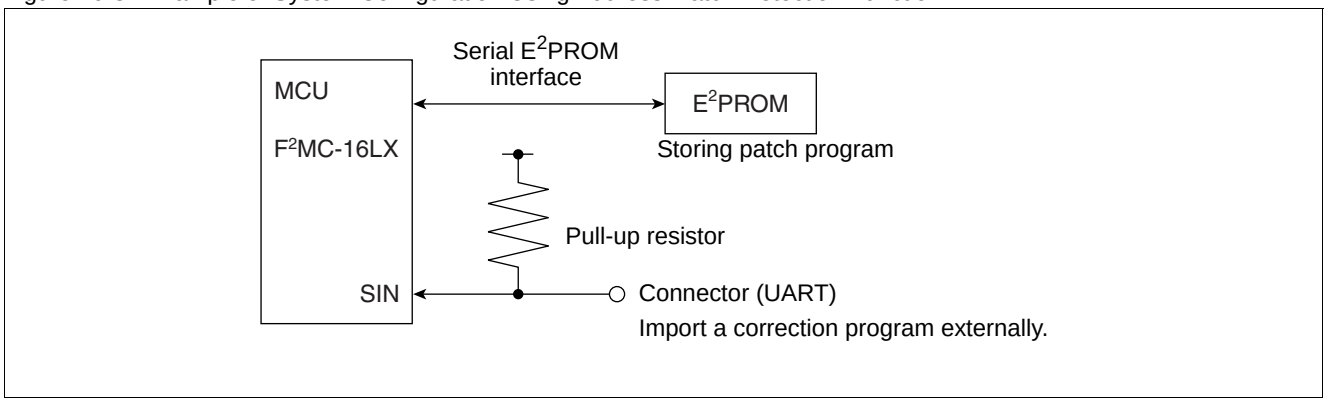
This section gives an example of patch processing for program correction using the address match detection function.

System Configuration and E^{2}PROM Memory Map

System configuration

Figure 20-8 gives an example of the system configuration using the address match detection function.

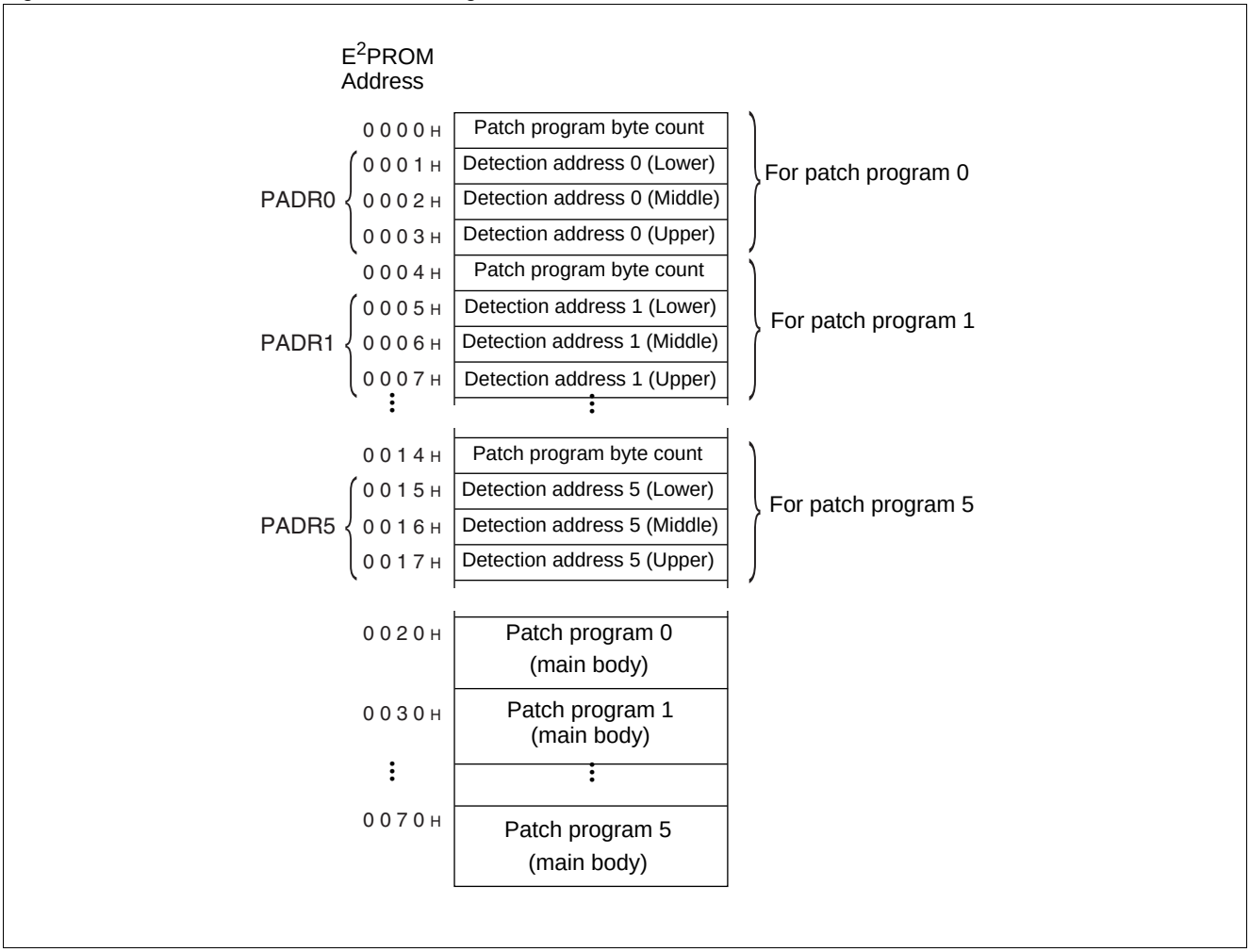
Figure 20-8. Example of System Configuration Using Address Match Detection Function



E^{2}PROM Memory Map

Figure 20-9 shows the allocation of the patch program and data at storing the patch program in E^{2}PROM.

Figure 20-9. Allocation of E^{2}PROM Patch Program and Data



Patch program byte count

The total byte count of the patch program (main body) is stored. If the byte count is "00_H", it indicates that no patch program is provided.

Detection address (24 bits)

The address where the instruction code is replaced by the INT9 instruction code due to program error is stored. This address is set in the detection address set registers (PADR0 to PADR5).

Patch program (main body)

The program executed by the INT9 interrupt processing when the program address matches the detect address is stored. Patch program 0 is allocated from any predetermined address. Patch program 1 is allocated from the address indicating <starting address of patch program 0 + total byte count of patch program 0>.

Patch program 2 to 5 as well.

Setting and Operating State

Initialization

E^2PROM data are all cleared to "00_H".

Occurrence of program error

- By using the connector (UART), information about the patch program is transmitted to the MCU (F^2MC-16LX) from the outside according to the allocation of the E^2PROM patch program and data.
- The MCU (F^2MC-16LX) stores the information received from outside in the E^2PROM.

Reset sequence

- After reset, the MCU (F^2MC-16LX) reads the byte count of the E^2PROM patch program to check the presence or absence of the patch program.
- If the byte count of the patch program is not "00_H", the upper, middle and lower bits at detection addresses 0 to 5 are read and set in the detection address set registers 0 to 5 (PADR0 to PADR5). The patch program (main body) is read according to the byte count of the patch program and written to RAM in the MCU (F^2MC-16LX).
- The patch program (main body) is allocated to the address where the patch program is executed in the INT9 interrupt processing by the address match detection function.
- Address match detection is enabled (PACSR: AD0E = 1, AD1E = 1 to AD5E = 1).

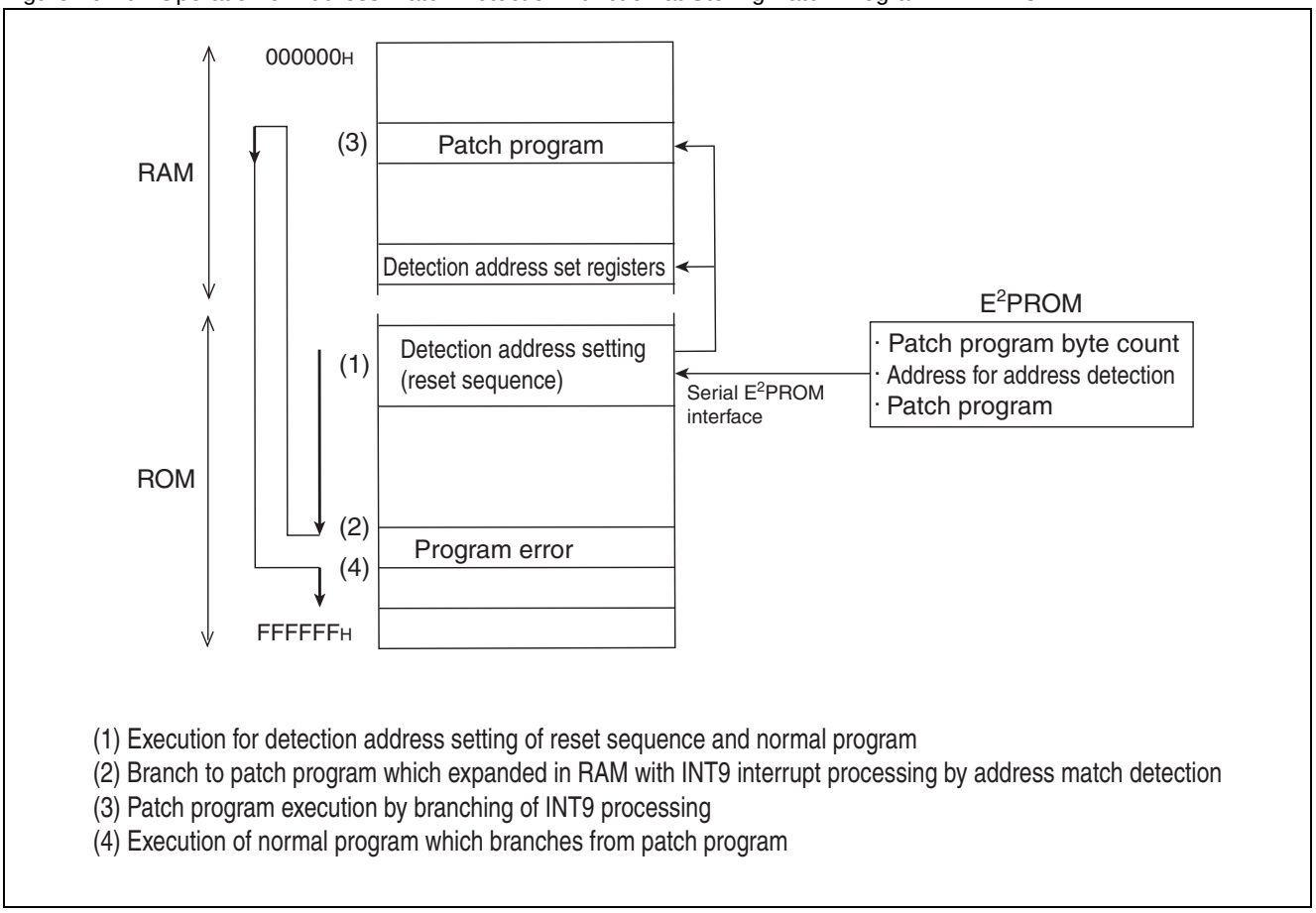
INT9 Interrupt processing

- Interrupt processing is performed by the INT9 instruction. The MB90880 series has no interrupt request flag by address match detection. Therefore, if the stack information in the program counter is discarded, the detect address cannot be checked. When checking the detect address, check the value of program counter stacked in the interrupt processing routine.
- The patch program is executed, branching to the normal program.

Operation of Address Match Detection Function at Storing Patch Program in E^{2}PROM

Figure 20-10 shows the operation of the address match detection function at storing the patch program in E^{2}PROM.

Figure 20-10. Operation of Address Match Detection Function at Storing Patch Program in E^{2}PROM

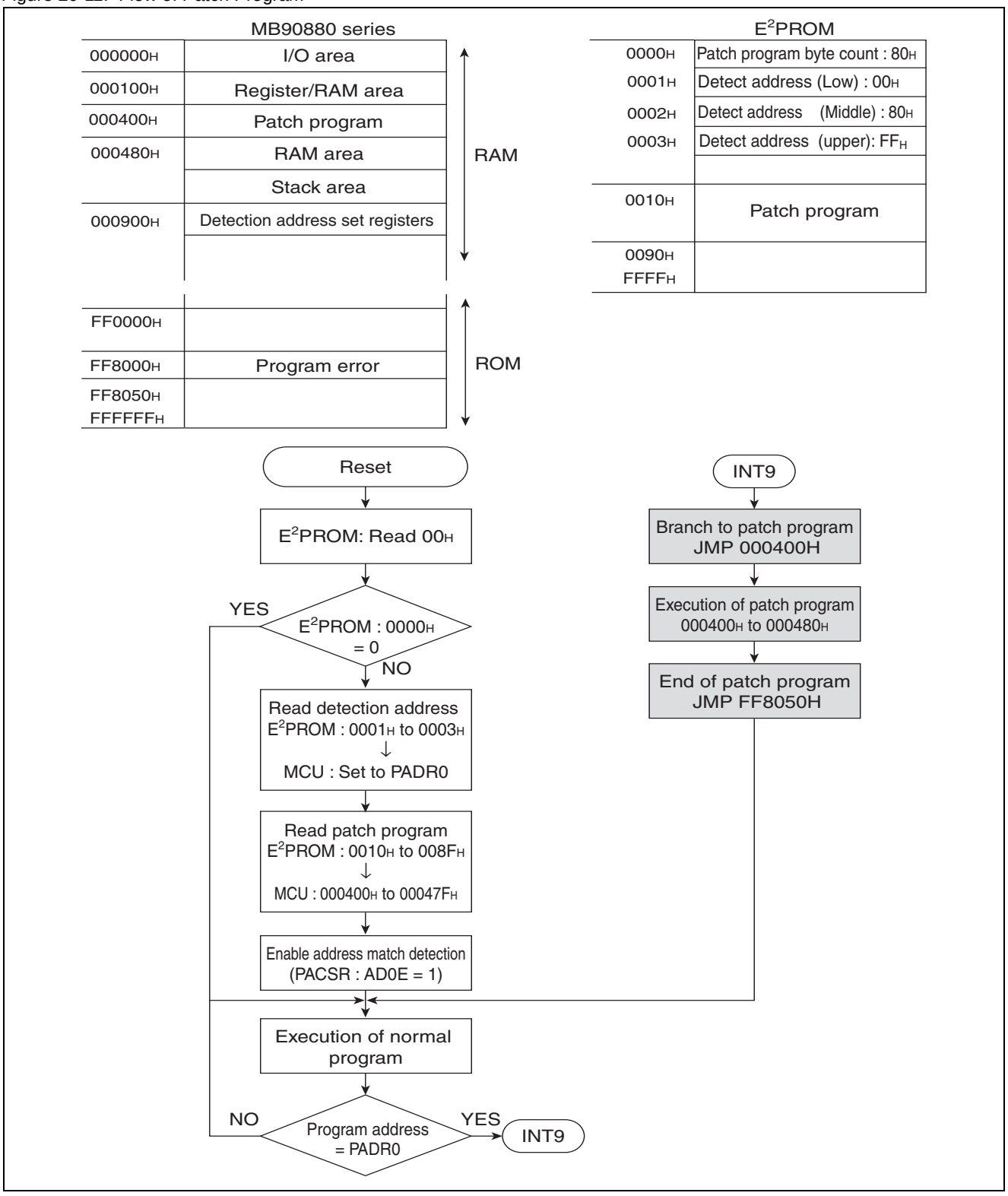


(1) Execution for detection address setting of reset sequence and normal program
(2) Branch to patch program which expanded in RAM with INT9 interrupt processing by address match detection
(3) Patch program execution by branching of INT9 processing
(4) Execution of normal program which branches from patch program

Flow of Patch Program

Figure 20-11 shows the flow of patch program using address match detection function.

Figure 20-11. Flow of Patch Program



21. ROM Mirror Function Selection Module

This chapter provides an overview of the ROM mirror function selection module and explains its registers.

21.1 Overview of ROM Mirror Function Selection Module

21.2 ROM Mirror Function Selection Register (ROMM)

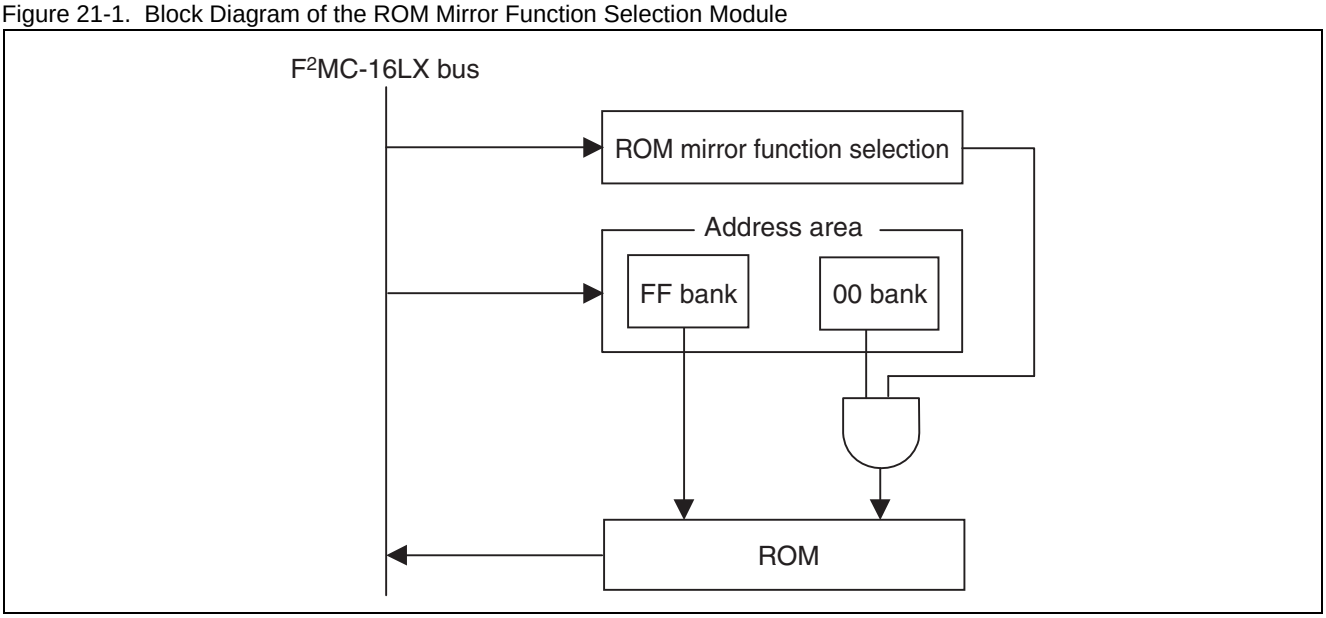
21.1 Overview of ROM Mirror Function Selection Module

The ROM mirror function selection module is set to read the ROM data arranged in FF bank with access to 00 bank.

Block Diagram of the ROM Mirror Function Selection Module

Figure 21-1 shows a block diagram of the ROM mirror function selection module.

Figure 21-1. Block Diagram of the ROM Mirror Function Selection Module



Registers of the ROM Mirror Function Selection Module

Figure 21-2 shows configuration of the ROM mirror function selection module.

Figure 21-2. ROM Mirror Function Selection Register (ROMM)

bit	15	14	13	12	11	10	9	8	Initial value
Address: 00006F$_H$	-	-	-	-	-	-	-	MI	XXXXXXX1$_B$
								R/W	

R/W: Readble/writable

21.2 ROM Mirror Function Selection Register (ROMM)

This section describes the configuration and functions of the ROM mirror function selection register (ROMM).

ROM Mirror Function Selection Register (ROMM)

The diagram below shows the bit configuration of the ROM mirror function selection register (ROMM).

bit	15	14	13	12	11	10	9	8	Initial value
ROMM address: $00006F_H$	-	-	-	-	-	-	-	MI	$XXXXXXX1_B$
								R/W	

R/W: Readble/writable

[bit15 to bit9]

These bits are undefined bits.

- When read: Value is undefined.
- When written: No effect.

[bit8] MI

This bit sets whether to enable or disable the ROM mirror function.

- “1”: Enable the mirror function.
- “0”: Disable the mirror function.

Note:

Do not access this register during accesses to address 008000_H to $00FFFF_H$.

22. Flash Memory

This chapter describes the functions and operations of the 2M/3M/4M-bit flash memory.

The following methods are available for writing/erasing data to/from the flash memory:

- Executing programs to write/erase data
- Writing via the serial programmer
- Writing via the flash memory programmer
- This chapter explains “Executing programs to write/erase data”.

22.1 Overview of Flash Memory

22.2 Sector Configuration of Flash Memory

22.3 Flash Memory Control Status Register (FMCS)

22.4 Flash Memory Write Control Registers (FWR0/FWR1)

22.5 Starting the Flash Memory Automatic Algorithm

22.6 Confirming the Automatic Algorithm Execution State

22.7 Writing Data to and Erasing Data from Flash Memory

22.8 Flash Security Function

22.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems

22.10 Notes on Using Flash Memory

22.1 Overview of Flash Memory

The flash memory is mapped into the F8 to FF banks on the CPU memory map. The flash memory allows the CPU to read-access and program-access the memory in the same way as for masked ROM. Instructions from the CPU allows to write/erase data in the flash memory. As a result, the programming and data can be improved efficiently.

Features of Flash Memory

The flash memory has the following features:

- Automatic algorithm (equivalent to the Embedded Algorithm)
- Suspend/restart to erase function provided
- Detect completion of data-writing/erasing with the data polling and toggle bit
- Detect completion of data-writing/erasing with the CPU interrupts
- Compatible with JEDEC standard commands
- Able to erase on a sector basis (any combination of sectors)
- Minimum of 10,000 data-writing/erasing operations

Capacities and Models of Flash Memory

One of 2M-bit, 3M-bit or 4M-bit flash memory is mounted depending on the model used.

2M-bit flash memory

- Compatible models : MB90F882A(S)
- Capacity : 256K bytes / 128K words
- Sector configuration : 64 K × 2 + 48 K × 2 + 8 K × 4

3M-bit flash memory

- Compatible models : MB90F883B(S)/MB90F883BH(S)/MB90F883C(S)
- Capacity : 384K bytes / 192K words
- Sector configuration : 64 K × 5 + 48 K + 8 K × 2

4M-bit flash memory

- Compatible models : MB90F884B(S)/MB90F884BH(S)/MB90F884C(S)
- Capacity : 512K bytes / 256K words
- Sector configuration : 64 K × 6 + 48 K × 2 + 8 K × 4

How to Write/Erase Data Flash Memory

You cannot read, write, and erase data to the flash memory at the same time. If you perform a data-writing/erasing operation on the flash memory, copy the program on the flash memory to RAM and then perform the operation on the RAM. This makes it possible to perform a data-writing/erasing operation without reading the flash memory.

22.2 Sector Configuration of Flash Memory

This section shows the sector configuration of the flash memory.

Sector Configuration

Figure 22-1 to Figure 22-3 illustrate the sector configuration of the 2M/3M/4M-bit flash memory. The addresses in the figure indicate the upper and lower addresses of each sector.

Figure 22-1. Sector Configuration of 2M-bit flash memory

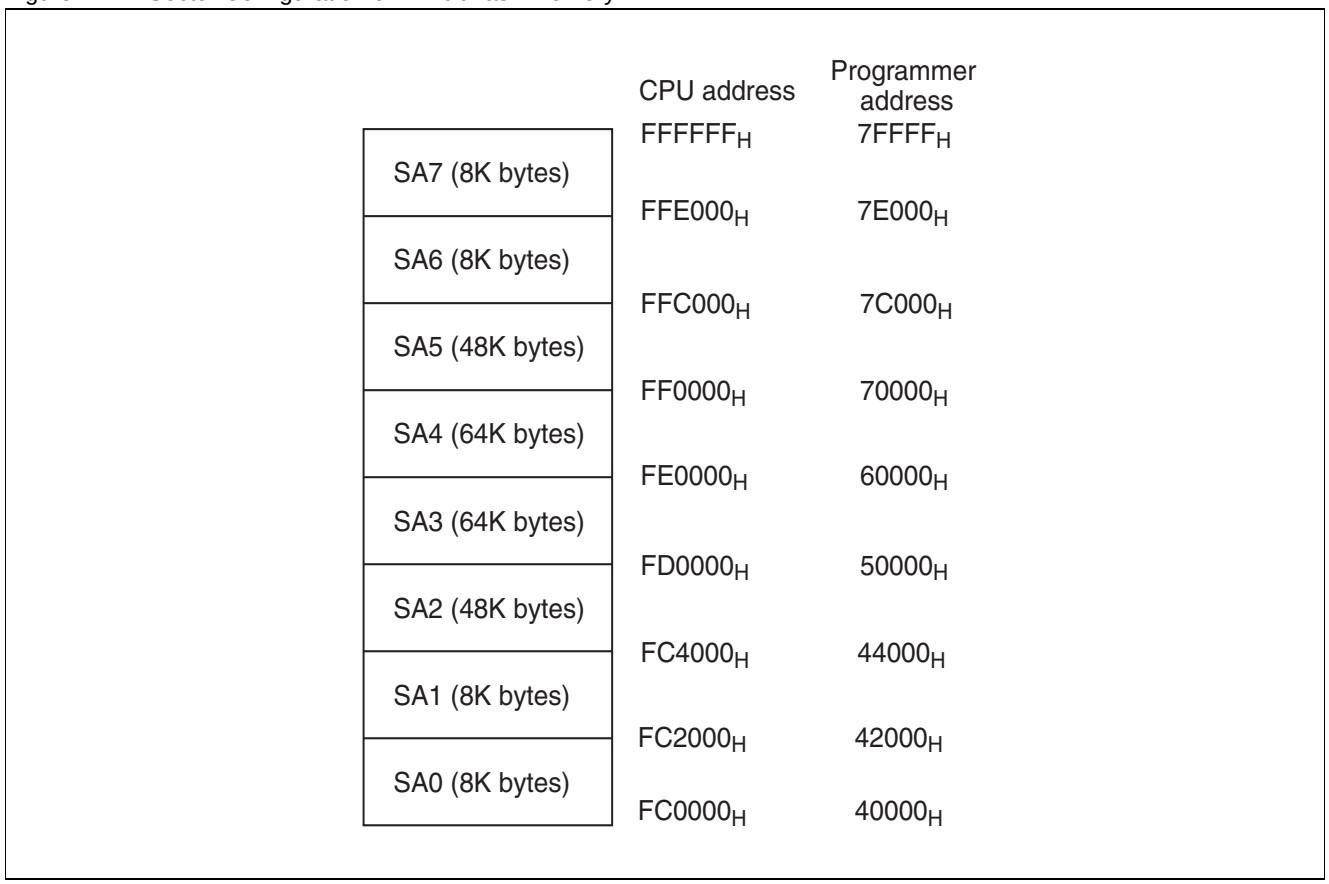


Figure 22-2. Sector Configuration of 3M-bit flash memory

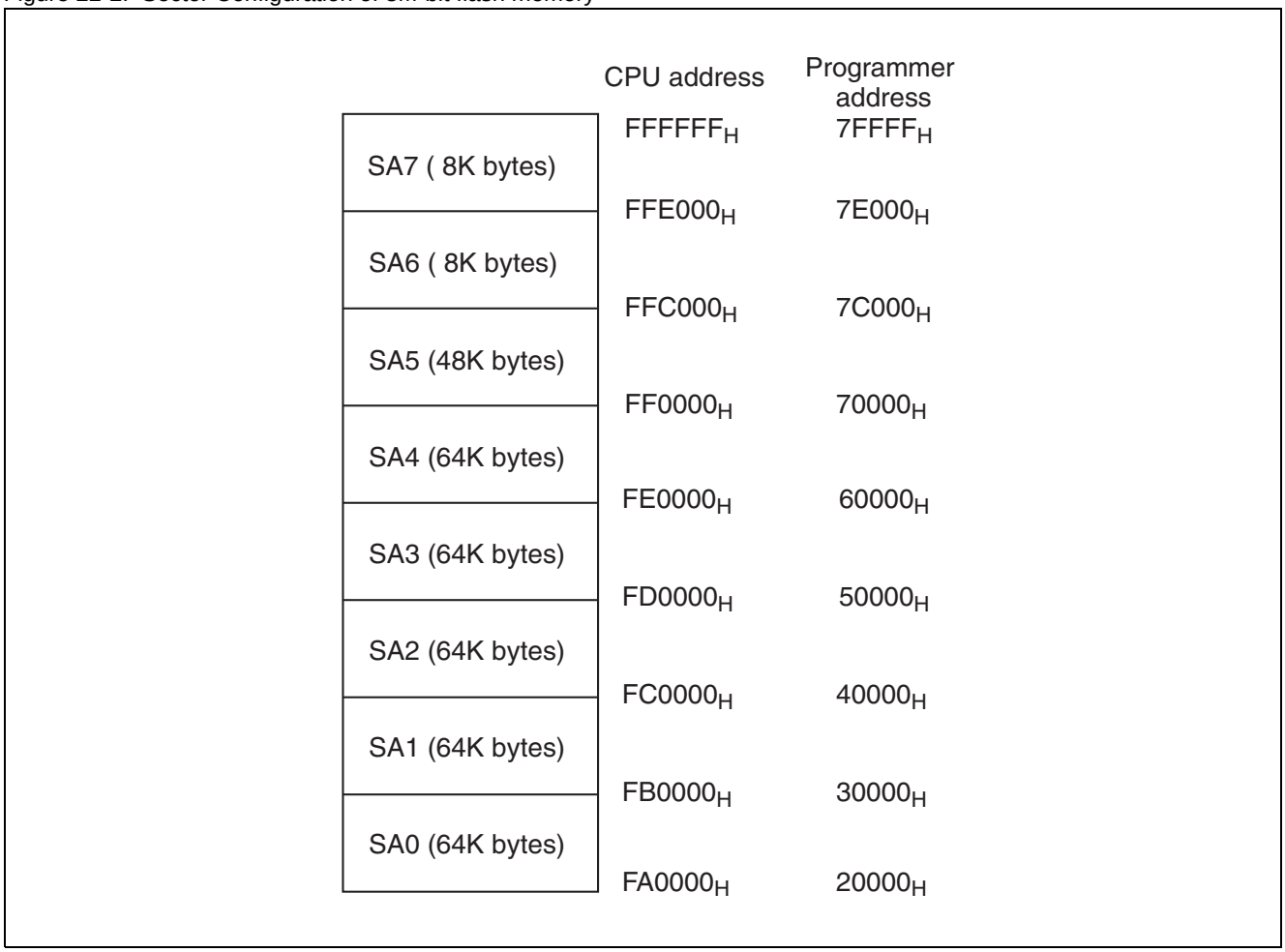
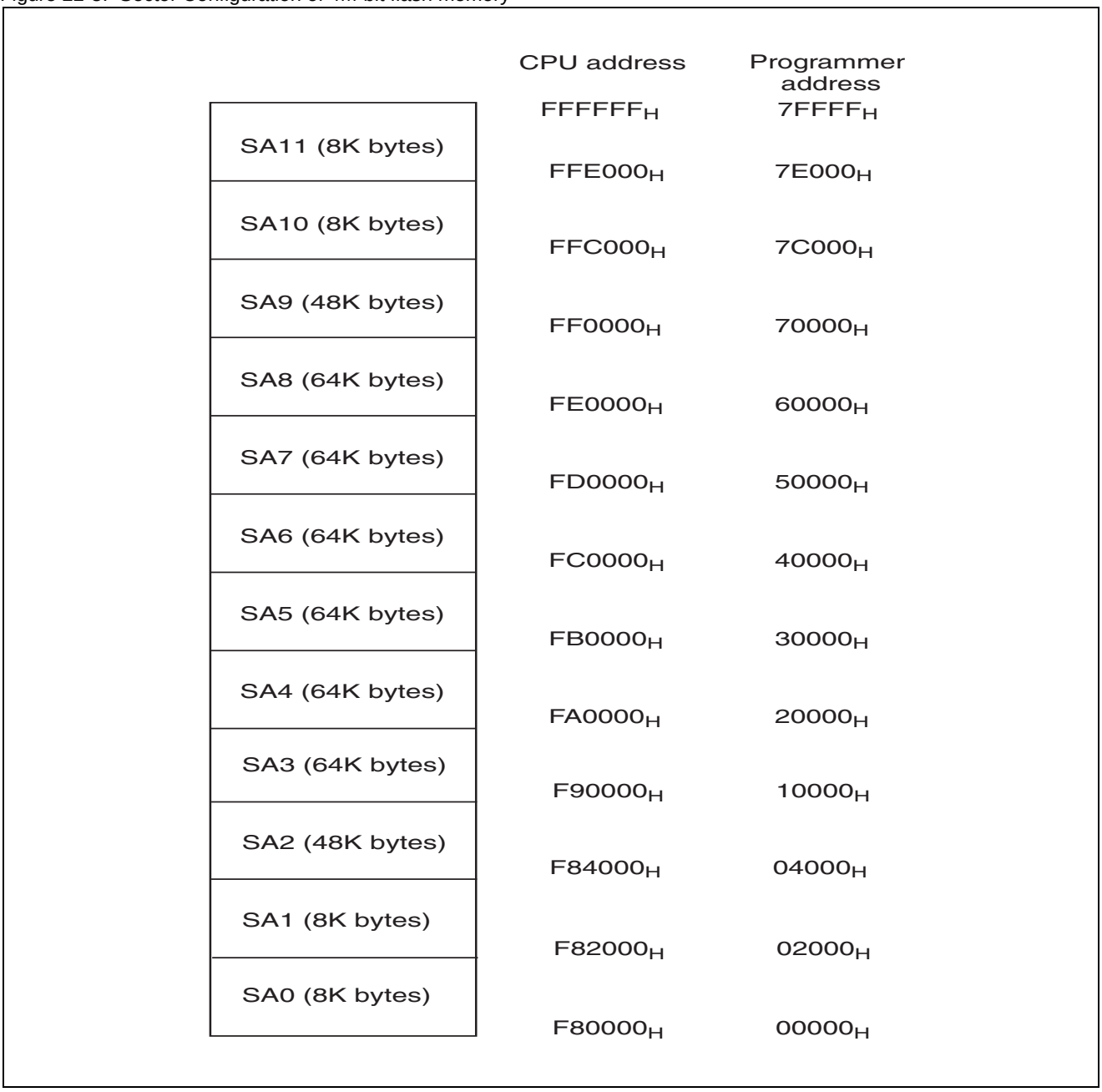


Figure 22-3. Sector Configuration of 4M-bit flash memory



Programmer address

The programmer address in the Figure 22-1 to Figure 22-3 corresponds to the CPU address when writing data to the flash memory with the parallel programmer. Use this programmer address for writing/erasing data with a general-purpose programmer.

22.3 Flash Memory Control Status Register (FMCS)

The flash memory control status resister (FMCS), located in the flash memory interface circuit, is used for the data-writing/erasing operation on the flash memory.

Flash Memory Control Status Register (FMCS)

The figure below shows the bit configuration of the flash memory control status resister (FMCS).

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address: $0000AE_H$	INTE	RDYINT	WE	RDY	Reserved	Reserved	Reserved	Reserved
Read/Write →	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Initial value →	0	0	0	X	0	0	0	0

R/W:Readable/Writable
R: Read only
X: Undefined value

The following explains the function of each bit in the flash memory control status resister (FMCS).

[bit7] INTE: INTerrupt Enable

This bit enables or disables an interrupt request generation upon completion of the automatic algorithm for the flash memory data-writing/erasing operation.

An interrupt to the CPU occurs when the INTE and RDYINT bits are both “1”. The interrupt will not occur if the INTE bit is “0”.

0	Disable interrupt at data-writing/erasing completion
1	Enable interrupt at data-writing/erasing completion

[bit6] RDYINT: ReaDY INTerrupt

This bit is an interrupt request flag that is set upon completion of the automatic algorithm for the flash memory data-writing/erasing operation.

It is set to “1”, when the flash memory data-writing/erasing operation is completed. If this bit is set to “1” when the INTE bit is “1”, an interrupt request occurs upon completion of the automatic algorithm.

Writing “0” clears to set this bit to “0”. When “1” is set, operation is ignored. The bit returns to “1” when read by the read-modify-write (RMW) instruction.

0	No interrupt request generated
1	Data-writing/erasing operation completed (interrupt request generated)

[bit5] WE: Write Enable

This bit is a write enable bit to flash memory area.

When this bit is set to “1”, writing to the flash memory area is performed by writing a command sequence, which allows data to be written to and erased from the flash memory. When this bit is set to “0”, a write attempt to the flash memory area is ignored. This bit activates a flash memory data write/erase command.

When performing no data-writing/erasing operations, set the WE bit to “0” to avoid writing to flash memory by mistake.

0	Flash memory write disabled
1	Flash memory write enabled

[bit4] RDY: ReadDY

This bit is a status bit that indicates the status of data-writing/erasing operation of the flash memory.

Data-writing/erasing to flash memory is disabled upon the bit “0”. Even in this state, a reset command and a sector erase suspend command are acceptable.

0	Data-writing/erasing operation executing
1	Data-writing/erasing operation completed (next data write/erase enabled)

[bit3 to bit0] Reserved bits

These bits are reserved bits. Be sure to set these bits to “0”.

22.4 Flash Memory Write Control Registers (FWR0/FWR1)

The flash memory write control registers (FWR0/FWR1) exist in the flash memory interface to be used to set the flash memory write-protect feature.

The write-protect feature is not available to the MB90F883B (H)(S) and MB90F884B (H)(S).

Flash Memory Write Control Registers (FWR0/FWR1)

The flash memory write control registers (FWR0/FWR1) contain the bits to enable/disable the programming of data into individual sectors (SA0 to SA11). The initial value of each bit is “0” to disable programming. Writing “1” to one of the bits enables programming into the corresponding sector. Writing “0” to the bit prevents an accidental write from being executed to the sector. Once you have written “0” to the bit, therefore, you cannot write to the sector even though you write “1” to the bit. You have to reset the bit before you write to the sector again.

Figure 22-4. Flash Memory Write Control Registers (FWR0/FWR1)

FWR0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address: 0079A6$_H$	SA7E	SA6E	SA5E	SA4E	SA3E	SA2E	SA1E	SA0E
	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W

FWR1	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
Address: 0079A7$_H$	Reserved	Reserved	Reserved	Reserved	SA11E	SA10E	SA9E	SA8E
	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W	(0) R/W

R/W : Readable/writable
0 : Write disabled [Initial value]

Table 22-1. Functions of Flash Memory Write Control Registers (FWR0/FWR1)

Bit name		Function
bit15 to bit0	Reserved bits	Writing has no effect on operation. Read value is fixed to "0".
	SA11E to SA0E: Write-protection setup bits	These bits are used to set the accidental write preventive function for the individual sectors of flash memory. Writing “1” to one of the bits permits programming into the corresponding sector. Writing “0” to the bit write-protects that sector (prevents an accidental write to the sector). A reset initializes the bit to “0” (programming prohibited). **Write-disable** : State of **“0”**. The bit corresponding to each sector can (be set to “1” to) permit programming into that sector, with no “0” written in the flash memory write control register (FWR0/FWR1). (State existing immediately after a reset). **Write-enable** : State of **“1”**. Data can be programmed into the corresponding sector. **Write-protect** : State of “0”. The bit corresponding to each sector cannot (be set to “1” to) permit programming into that sector even by writing “1” to it with “0” written in the flash memory write control register (FWR0/FWR1).

Table 22-2. Write-protection setup bits and corresponding flash sectors Product with 4M-bit flash memory

Product with 4M-bit flash memory

Bit	Bit name	Corresponding flash memory sector
15	Reserved	-
14	Reserved	-
13	Reserved	-
12	Reserved	-
11	SA11E	SA11
10	SA10E	SA10
9	SA9E	SA9
8	SA8E	SA8
7	SA7E	SA7
6	SA6E	SA6
5	SA5E	SA5
4	SA4E	SA4
3	SA3E	SA3
2	SA2E	SA2
1	SA1E	SA1
0	SA0E	SA0

Table 22-3. Write-protection Setup Bits and Corresponding Flash Sectors

Product with 3M-bit flash memory

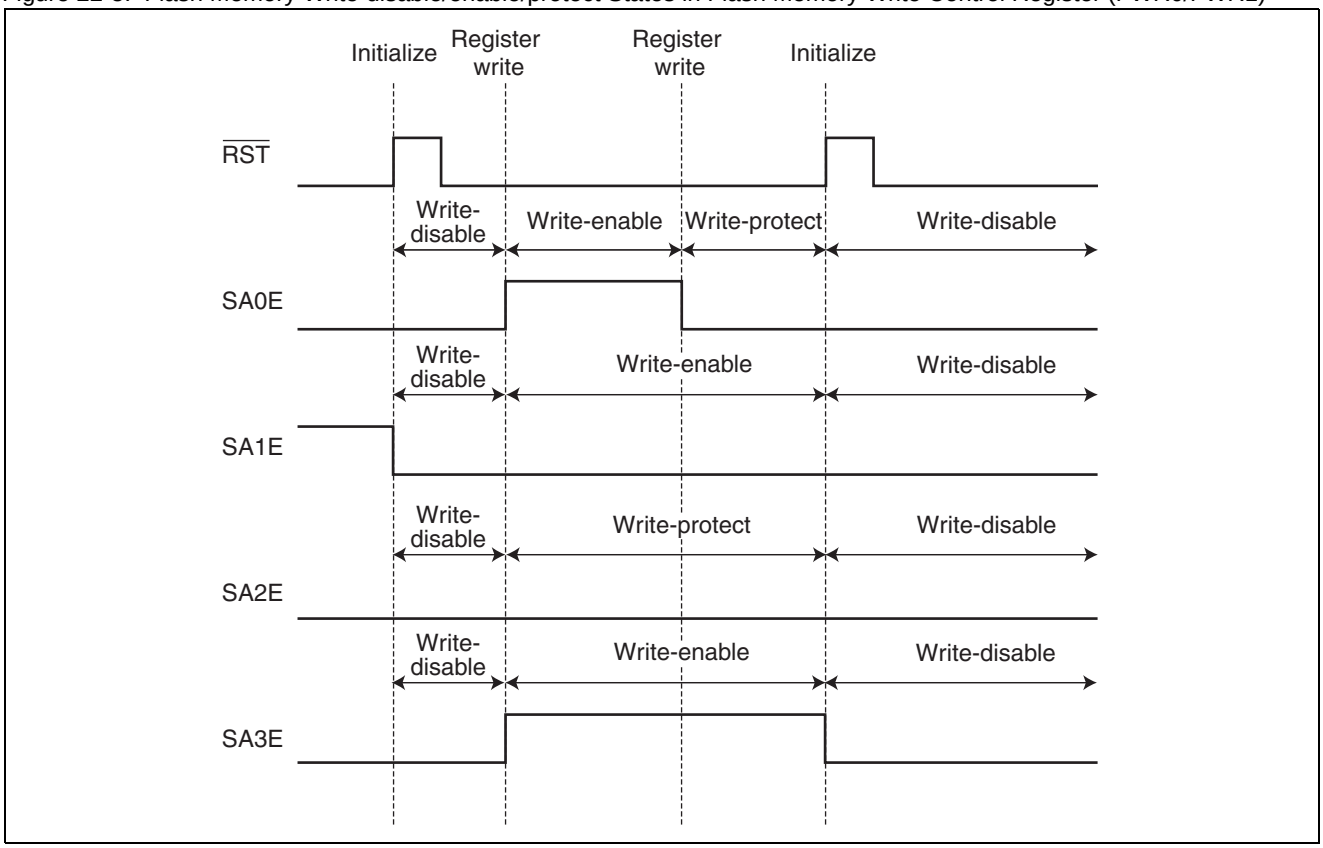
Bit	Bit name	Corresponding flash memory sector
15	Reserved	-
14	Reserved	-
13	Reserved	-
12	Reserved	-
11	SA11E	SA7
10	SA10E	SA6
9	SA9E	SA5
8	SA8E	SA4
7	SA7E	SA3
6	SA6E	SA2
5	SA5E	SA1
4	SA4E	SA0
3	SA3E	No corresponding sector available: Write “0” to these bits.
2	SA2E	
1	SA1E	
0	SA0E	Corresponding to the security bit: Write “1”, only when writing to the security bit.

Table 22-4. Write-protection setup bits and corresponding flash sectors

Product with 2M-bit flash memory

Bit	Bit name	Corresponding flash memory sector
15	Reserved	-
14	Reserved	-
13	Reserved	-
12	Reserved	-
11	Reserved	-
10	Reserved	-
9	Reserved	-
8	Reserved	-
7	SA7E	SA7
6	SA6E	SA6
5	SA5E	SA5
4	SA4E	SA4
3	SA3E	SA3
2	SA2E	SA2
1	SA1E	SA1
0	SA0E	SA0

Figure 22-5. Flash Memory Write-disable/enable/protect States in Flash Memory Write Control Register (FWR0/FWR1)



Write-disable:

State of “0”. The bit corresponding to each sector can (be set to “1” to) permit programming into that sector, with no “0” written in the flash memory write control register (FWR0/FWR1). (State existing immediately after a reset).

Write-enable:

State of “1”. Data can be programmed into the corresponding sector.

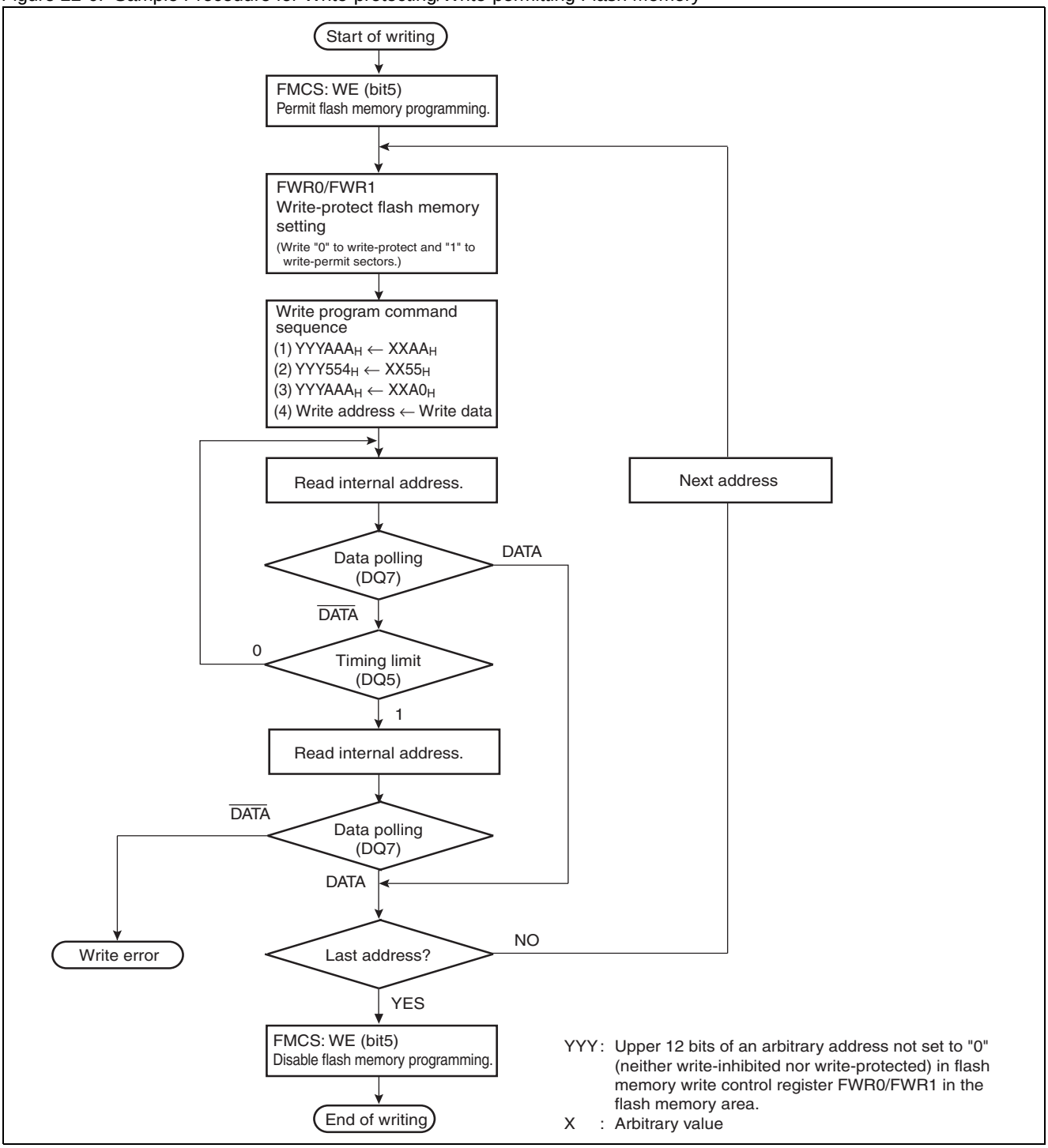
Write-protect:

State of “0”. The bit corresponding to each sector cannot (be set to “1” to) permit programming into that sector even by writing “1” to it with “0” written in the flash memory write control register (FWR0/FWR1).

Setup Flowchart for Flash Memory Write Control Registers (FWR0/FWR1)

Set the FMCS:WE bit and permit data-writing/erasing or protect writing for each sector by setting the corresponding bit in the flash memory write control register (FWR0/FWR1) to “1” or “0”, respectively. To these registers, be sure to write data in words. No bit manipulation instruction can be used for setting their bits.

Figure 22-6. Sample Procedure for Write-protecting/Write-permitting Flash Memory



Note on Setting the FMCS:WE Bit

To program into flash memory, set FMCS:WE to “1” to write-permit it and set the flash memory write control register (FWR0/FWR1). When FMCS:WE inhibits writing (contains “0”), the write to flash memory is not operated even though it is permitted by the flash memory write control register (FWR0/FWR1).

22.5 Starting the Flash Memory Automatic Algorithm

Four types of commands are available for starting the flash memory automatic algorithm: Reset, Data Write, Chip Erase, and Sector Erase. Control of suspend and restart is enabled for Sector erase.

Command Sequence Table

Table 22-5 lists the commands used for flash memory data-writing/erasing. Use word access to write any data to the flash memory area. In the command sequence, upper bytes “XX” are ignored.

Table 22-5. Command Sequence Table

Command sequence	programming cycle	1st write cycle		2nd write cycle		3rd write cycle		4th write cycle		5th write cycle		6th write cycle	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Reset*1	1	yyyXXX	XXF0	-	-	-	-	-	-	-	-	-	-
Reset*1	3	yyyAAA	XXAA	yyy554	XX55	yyyAAA	XXF0	-	-	-	-	-	-
Data write*2	4	yyyAAA	XXAA	yyy554	XX55	yyyAAA	XXA0	PA (even)	PD (word)	-	-	-	-
Chip erase	6	yyyAAA	XXAA	yyy554	XX55	yyyAAA	XX80	yyyAAA	XXAA	yyy554	XX55	yyyAAA	XX10
Sector erase*2	6	yyyAAA	XXAA	yyy554	XX55	yyyAAA	XX80	yyyAAA	XXAA	yyy554	XX55	SA (even)	XX30
Sector erase suspend		Entering address “yyyXXXX” data ($xxB0_H$) suspends erasing during sector erasure.											
Sector erase restart		Entering address “yyyXXXX” data ($xx30_H$) restarts erasing after sector erasing is suspended.											

PA: Data write address. Only even number addresses can be specified.

SA: Sector address (see Section “22.2 Sector Configuration of Flash Memory“.)

PD: Write data. Only word data can be specified.

yyy: Upper 12 bits of the arbitrary address which is not set “0” (write prohibited/write-protect prohibited) in the flash memory writing control register (FWR0/FWR1)

*1: Both of the two types of Reset commands can reset the flash memory to read mode.

*2: For PA and SA, specify the address which is not set to “0” by the flash memory writing control register (FWR0/FWR1).

Notes:

- The addresses in the table above are based on the memory map. Addresses and data are described with the hexadecimal number. However, “X” indicates an arbitrary number.
- If the chip erase command is issued by accessing to the sector enabled to write where the sector is enabled/disabled to write, the contents in all sectors will be erased including the sectors where the writing is disabled.

22.6 Confirming the Automatic Algorithm Execution State

Data-writing/erasing operations of the flash memory are controlled using the automatic algorithm. The flash memory has hardware sequence flags for informing its internal operating state and completion of operation. When the flash memory area is read during the execution of the automatic algorithm, the hardware sequence flags can be read.

Hardware Sequence Flags

The hardware sequence flags have the four-bit output of the data polling flag (DQ7), toggle bit flag (DQ6), timing limit exceeded flag (DQ5), and sector erase timer flag (DQ3).

Table 22-6 lists the bit assignments of the hardware sequence flags.

Table 22-6. Bit Assignments of Hardware Sequence Flags

Bit No.	7	6	5	4	3	2	1	0
Hardware Sequence Flags	DQ7	DQ6	DQ5	-	DQ3	-	-	-

The hardware sequence flags can be referenced by read-accessing the addresses of the target sectors in the flash memory area after setting of the command sequence (see Table 22-5).

The automatic algorithm execution state can be confirmed with the following methods.

- Confirmation with referring to the hardware sequence flags
- Confirmation with referring to the RDY bit of flash memory control register (FMCS)

When programming, the next data-writing/erasing operation should be executed after the completion of automatic algorithm execution is confirmed with one of these methods. The following sections describe each hardware sequence flag separately.

Table 22-7 lists the functions of the hardware sequence flags.

Table 22-7. Hardware Sequence Flag Functions List

Status			DQ7	DQ6	DQ5	DQ3
Status change for normal operation	Data Write → Write completed (write address specified)		$\overline{DQ7}$ → DATA:7	Toggle → DATA:6	0 → DATA:5	0 → DATA:3
	Chip erase → Erase completed		0 → DATA:7	Toggle → DATA:6	0 → DATA:5	1 → DATA:3
	Sector erase	Sector erase time-out → Erase started	1 → 0	Toggle	0	0 → 1
		Sector erase → Erase completed	0 → DATA:7	Toggle → DATA:6	0 → DATA:5	1 → DATA:3
	Erase → Sector erase suspended (Sector being erased)		0 → 1	Toggle → 1	0	1 → 0
	Sector erase suspend → Erase restarted (Sector being erased)		1 → 0	1 → Toggle	0	0 → 1
	Sector erase suspended (Sector not being erased)		DATA:7	DATA:6	DATA:5	DATA:3
Abnormal operation	Data Write		DQ7	Toggle	1	0
	Chip/sector erase		0	Toggle	1	1

22.6.1 Data Polling Flag (DQ7)

The data polling flag (DQ7) is a hardware sequence flag to indicate that the automatic algorithm is being executed or has terminated by the data polling function.

Data Polling Flag (DQ7) State Transition

The data polling flag state transition is shown in Table 22-8 and Table 22-9.

Table 22-8. Data Polling Flag State Transition (State Change for Normal Operation)

Operating Status	Data write → Completed	Chip/sector erase → Completed	Sector erase time-out → Sector erase started	Sector erase→ Erase suspended Sector being erased	Sector erase suspend → Restarted Sector being erased	Sector erase suspended Sector not being erased
DQ7	$\overline{DQ7}$ → DATA:7	0 → 1 (DATA:7)	1 → 0	0 → 1	1 → 0	DATA:7

Table 22-9. Data Polling Flag State Transition (State Change for Abnormal Operation)

Operating Status	Data write	Chip/sector Erase
DQ7	DQ7	0

DAta Write

Read-access during execution of the automatic algorithm for data-writing operation causes the flash memory to output the opposite data of bit7 last written, regardless of the value at the address specified by the address signal. When the flash memory is read-accessed upon completion of the automatic algorithm, it outputs bit7 of the value read from the address located by the address signal.

Sector Erase

Read-access from the sector which is currently being erased during execution of the automatic sector erase algorithm causes the flash memory to output "0". In this series, after the Sector Erase command is issued, "1" is output for 50 to 160 µs before "0" is output, due to functional restrictions. When the sector erase operation is completed, the flash memory outputs "1".

For information about restrictions on the data polling flag (DQ7) during the sector erase operation and how to avoid related problems, refer to "22.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems".

Chip Erase

Read-access during execution of the automatic chip erase algorithm causes the flash memory to output "0", regardless of the value at the address specified by the address signal. When the chip erase operation is completed, the flash memory outputs "1".

Sector Erase Suspended

Read-access for an access from sector erase suspended causes the flash memory to output "1" if the address specified by the address signal belongs to the sector being erased. The flash memory outputs bit7 (DATA: 7) of the read value at the address specified by the address signal if the address specified by the address signal does not belong to the sector being erased. Referencing this flag together with the toggle bit flag (DQ6) enables to see whether the flash memory is in the sector erase suspended state and which sector is being erased.

Note:
When the automatic algorithm is being started, read-access to the specified address is ignored. After the termination of data polling flag (DQ7) is confirmed, data can be read. A data read after completion of the automatic algorithm should therefore follow the read access which confirms the completion of data polling.

22.6.2 Toggle Bit Flag (DQ6)

In the same manner of the data polling flag (DQ7), the toggle bit flag (DQ6) is a hardware sequence flag to indicate that the automatic algorithm is being executed or has terminated by the toggle bit function.

Toggle Bit Flag (DQ6) State Transition

The toggle bit flag state transition is shown in Table 22-10 and Table 22-11.

Table 22-10. Toggle Bit Flag State Transition (State Change for Normal Operation)

Operating status	Data Write → Completed	Chip/sector erase → Completed	Sector erase time-out → Sector erase started	Sector erase → Erase suspended Sector being erased	Sector erase suspend → Restarted Sector being erased	Sector erase suspended Sector not being erased
DQ6	Toggle → DATA:6	Toggle → DATA:6	Toggle	Toggle → 1	1 → Toggle	DATA:6

Table 22-11. Toggle Bit Flag State Transition (State Change for Abnormal Operation)

Operating Status	Data Write	Chip/sector Erase
DQ6	Toggle	Toggle

Data Write and Chip/Sector Erase

Continuous read-access during execution of the data write and chip/sector erase automatic algorithm causes the flash memory to toggle the “1” or “0” state alternately for every read cycle, regardless of the value at the address specified by the address signal. When the flash memory is continuously read accessed upon completion of the data write or chip/sector erase automatic algorithm, it stops toggling bit6 and outputs bit6 (DATA: 6) of the value read from the address located by the address signal.

Sector Erase Suspended

When the flash memory is read-accessed during sector erase suspended, it outputs “1” if the address located by the address signal belongs to the sector being erased. If the address does not belong to the sector being erased, the flash memory outputs bit6 (DATA:6) of the value read from the address located by the address signal.

22.6.3 Timing Limit Exceeded Flag (DQ5)

The timing limit exceeded flag (DQ5) is a hardware sequence flag to indicate that the automatic algorithm has exceeded the time (internal pulse count) specified inside the flash memory.

Timing Limit Exceeded Flag (DQ5) State Transition

The timing limit exceeded flag state transition is shown in Table 22-12 and Table 22-13.

Table 22-12. Timing Limit Exceeded Flag State Transition (State Change for Normal Operation)

Operating status	Data write → Completed	Chip/sector erase → Completed	Sector erase time-out → Sector erase started	Sector erase → Erase suspended Sector being erased	Sector erase suspend → Restarted Sector being erased	Sector erase suspended Sector not being erased
DQ5	0 → DATA:5	0 → DATA:5	0	0	0	DATA:5

Table 22-13. Timing Limit Exceeded Flag State Transition (State Change for Abnormal Operation)

Operating status	Data write	Chip/sector Erase
DQ5	1	1

Data Write and Chip/Sector Erase

When the flash memory is read-accessed after the data write and chip/sector erase automatic algorithm is started, this flag outputs “0” if the specified time (time required for data-writing/erasing operation) has not been exceeded, or “1” if the time has been exceeded. Because this is done regardless of whether the automatic algorithm is being executed or has terminated, this flag can be indicated whether data-writing/erasing is successfully executed or not. Therefore, if the automatic algorithm is still executed by the data polling function or toggle bit function with this flag “1”, that indicates the data-writing is failed.

For example, writing “1” to a flash memory address where “0” has been written will cause the fail state to occur. In this case, the flash memory will be locked and the automatic algorithm will not terminate to execute. In rare cases, it may terminate normally with writing “1”. As a result, valid data will not be output from the data polling flag (DQ7). In addition, the toggle bit flag (DQ6) will exceed the time limit without stopping the toggle operation and the timing limit exceeded flag (DQ5) will output “1”. Note that this state indicates that the flash memory is not malfunctioned, but has not been operated correctly. When this state occurs, execute the Reset command.

22.6.4 Sector Erase Timer Flag (DQ3)

The sector erase timer flag (DQ3) is to indicate whether the automatic algorithm is being executed during the sector erase time-out period after the Sector Erase command has been started.

Transition of State of Sector Erase Timer Flag (DQ3)

The sector erase timer flag state transition is shown in Table 22-14 and Table 22-15.

Table 22-14. Sector Erase Timer Flag State Transition (State Change for Normal Operation)

Operating status	Data write → Completed	Chip/sector erase → Completed	Sector erase time-out → Sector erase started	Sector erase → Erase suspend Sector being erased	Sector erase suspend → Restarted Sector being erased	Sector erase suspended Sector not being erased
DQ3	0 → DATA:3	1 → DATA:3	0 → 1	1 → 0	0 → 1	DATA:3

Table 22-15. Sector Erase Timer Flag State Transition (State Change for Abnormal Operation)

Operating status	Data write	Chip/sector Erase
DQ3	0	1

Sector Erase

Read-access after the Sector Erase command has been started causes the flash memory to output “0” if the automatic algorithm is being executed during the sector erase time-out period, regardless of the value at the address specified by the address signal of the sector that issued the command. The flash memory outputs “1” if the sector erase time-out period has been exceeded.

When the data polling function or toggle bit function indicates that the erase algorithm is being executed, internally controlled erase has already started if this flag is “1”. Continuous write of the sector erase codes and issues of commands (except the Sector Erase Suspend) will be ignored until erase is terminated.

If this flag is “0”, the flash memory accepts an additional sector erase code to be written. To confirm this, it is advisable to check the state of this flag before continuing to write sector erase codes. If the flag is “1” at the second status check, the additional sector erase code may not have been accepted.

Sector Erase Suspended

When the flash memory is read-accessed during sector erase suspended, it outputs “1” if the address located by the address signal belongs to the sector being erased. If the address does not belong to the sector being erased, the flash memory outputs bit3 (DATA:3) of the value read from the address located by the address signal.

22.7 Writing Data to and Erasing Data from Flash Memory

This section describes the procedures for writing data to and erasing data from the flash memory by the activation of the automatic algorithm.

Data-Writing/Erasing Flash Memory

The automatic algorithm can be activated by writing any command sequence of the Reset, Data Write, Chip Erase, Sector Erase, Sector Erase Suspend, or Erase Restart (see Table 22-5) from the CPU to flash memory. Writing from CPU to the flash memory must be performed continuously. In addition, the termination of the automatic algorithm can be confirmed with the data polling function. After the normal termination, the flash memory is returned to the read/reset state.

The following items related to data-writing/erasing operations of the flash memory are described respectively:

- Setting the read/reset state
- Writing data
- Erasing all data (erasing all chips)
- Erasing optional data (erasing sectors)
- Suspending sector erase
- Restarting sector erase

22.7.1 Setting Flash Memory to the Read/Reset State

This section describes the procedure for issuing the Read/Reset command to set the flash memory to the read/reset state.

Setting Flash Memory to the Read/Reset State

To set the flash memory to the read/reset state, issue the Reset command in the command sequence table (see Table 22-5) continuously to the target sector in flash memory.

The Reset command has two types of command sequences to execute the first and third write operations. However, there are no essential differences between these command sequences.

The read/reset state is the initial state of the flash memory. When power is on and when a command terminates normally, the flash memory is set to the read/reset state. In the read/reset state, other commands wait for input.

In the read/reset state, data is readable by regular read-access. In the same manner to the mask ROM, program access from the CPU is enabled. The Read/Reset command is not required to read data for a regular read. Use the command mainly to initialize an automatic algorithm, for example, when the command has failed to terminate normally for some reason.

22.7.2 Write Data to Flash Memory

This section describes the procedure for issuing the Write command to write data to the flash memory.

Write Data to the Flash Memory

To start the automatic algorithm for writing data into flash memory, write the Data Write command in the command sequence table (see Table 22-5) continuously to the target sector in flash memory. Once the target address and data are written in the fourth cycle, the automatic algorithm is activated to start automatic data-writing.

Specifying addresses

Only even-numbered addresses can be specified as the write addresses to be specified in the fourth cycle of the command sequence. Odd-numbered addresses cannot be written correctly. That is, writing to even addresses must be done in units of word data.

Although data-writing can be performed in any order of addresses and beyond a sector boundary, each data write command can write only one word of data.

Notes on writing data

When the flash memory data “0” is written to “1”, the data polling flag (DQ7) or toggle bit flag (DQ6) does not indicate the termination. Therefore, the flash memory elements are determined as malfunctioning and enter the following status. Do not set the data on the flash memory “0” back to “1” by writing data.

- The time prescribed for writing is exceeded, and the timing limit exceeded flag (DQ5) indicates an error.
- The data is occasionally viewed as if dummy data “1” had been written on the flash memory (When data is read in the read/reset state, the data remains “0”.

All commands are ignored during the execution of the automatic write algorithm. If a hardware reset is activated during writing at an address, the data at that address is not guaranteed.

Data-Writing Procedure to the Flash Memory

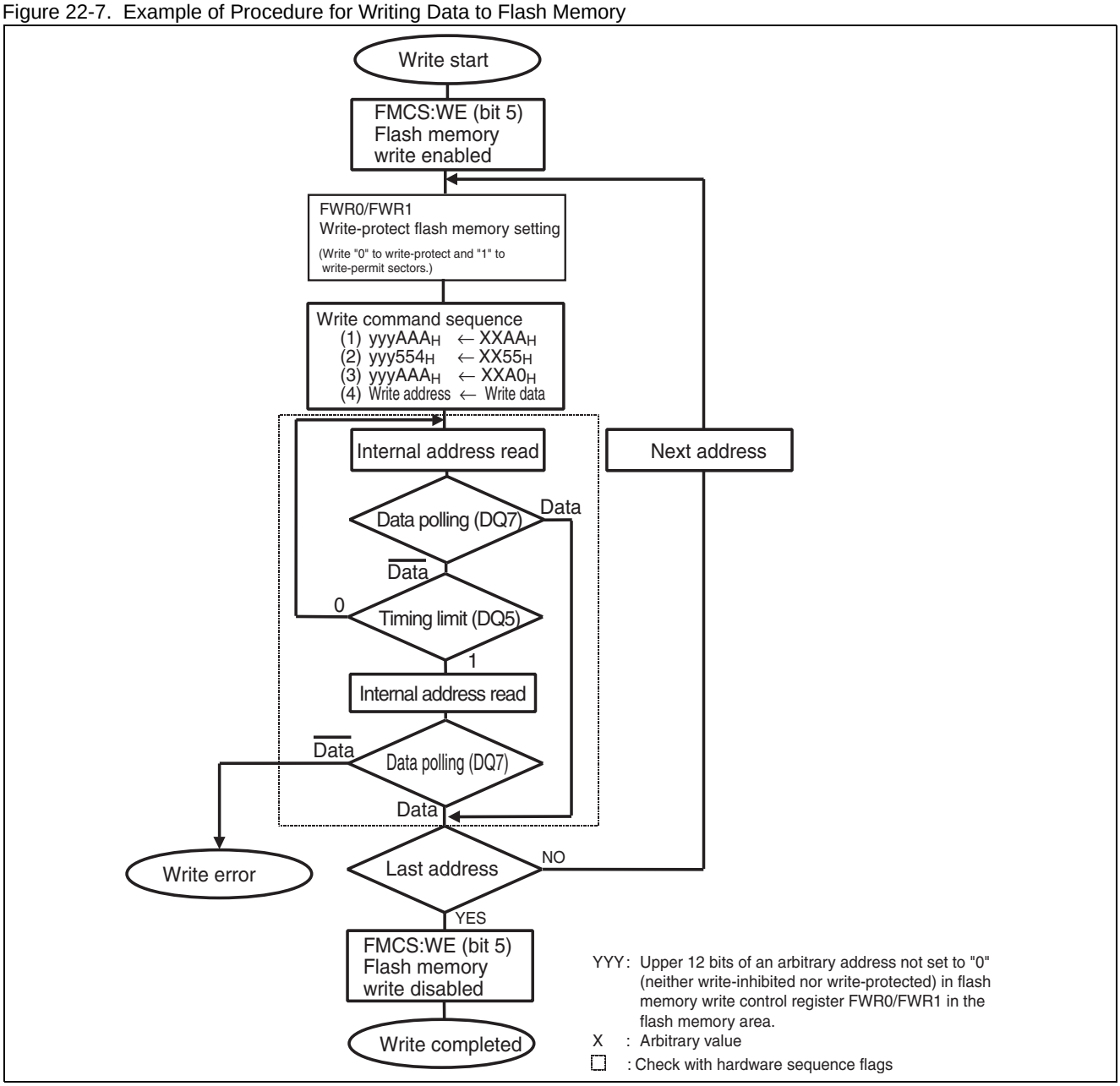
Figure 22-7 shows an example of the data-writing procedure. The hardware sequence flags (see Section "22.6 Confirming the Automatic Algorithm Execution State") can be used to determine the state of the automatic algorithm in the flash memory. In this procedure, the data polling flag (DQ7) is used to confirm that data-writing has terminated.

The data read to check the flag is read from the address where the last data was written.

The data polling flag (DQ7) changes with a skew almost at the same time that the timing limit exceeded flag (DQ5) changes. Therefore, even if the timing limit exceeded flag (DQ5) is "1", the data polling flag bit (DQ7) must be rechecked.

In the same manner of the toggle bit flag (DQ6), the toggle operation might be stopped almost at the same time that the timing limit exceeded flag bit (DQ5) changes to "1". The toggle bit flag (DQ6) must therefore be rechecked.

Figure 22-7. Example of Procedure for Writing Data to Flash Memory



22.7.3 Erasing All Data of Flash Memory (Chip Erase)

This section describes the procedure for issuing the Chip Erase command to erase all data in the flash memory.

Erasing All Data of Flash Memory (Chip Erase)

To erase all data from flash memory, write the Chip Erase command in the command sequence table (see Table 22-5) continuously to the target sectors in flash memory. The Chip Erase command is executed in six write operations. The chip erase operation starts upon completion of the write in the sixth cycle.

Notes on Chip Erase

If the chip erase command is issued by accessing to the sector enabled to write where the sector is enabled/disabled to write, the contents in all sectors will be erased including the sectors where the writing is disabled.

22.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase)

This section describes the procedure for issuing the Sector Erase command to erase one or more optional sectors in flash memory. The data by individual sector can be erased. Multiple sectors can also be specified at one time.

Erasing Optional Data (Erasing Sectors) in Flash Memory

To erase optional data in the flash memory, write the Sector Erase command in the command sequence table (see Table 22-5) continuously to the target sectors in flash memory.

Specifying Sectors

The Sector Erase command is executed in six write operations. The sector erase code (30_H) is written to an accessible even-numbered address in the target sector in the sixth cycle. Then, a sector erase time-out period of at least 50 μs is started. To erase multiple sectors, write the erase code (30_H) to the addresses in the target sectors after the above processing operation.

Notes on specifying multiple sectors

Erasing sectors starts at the end of the sector erase time-out period of at least 50 μs after writing the last sector erase code. Therefore, to erase multiple sectors at one time, the address in each sector to be erased and the sector erase code (in the sixth cycle of the command sequence) must be input within 50 μs. The sector erase timer (hardware sequence flag: DQ3) can be used to check whether writing of the subsequent sector erase code is valid. At this time, specify so that the address used for reading the sector erase timer indicates the sector to be erased.

Erasing Sectors in the Flash Memory

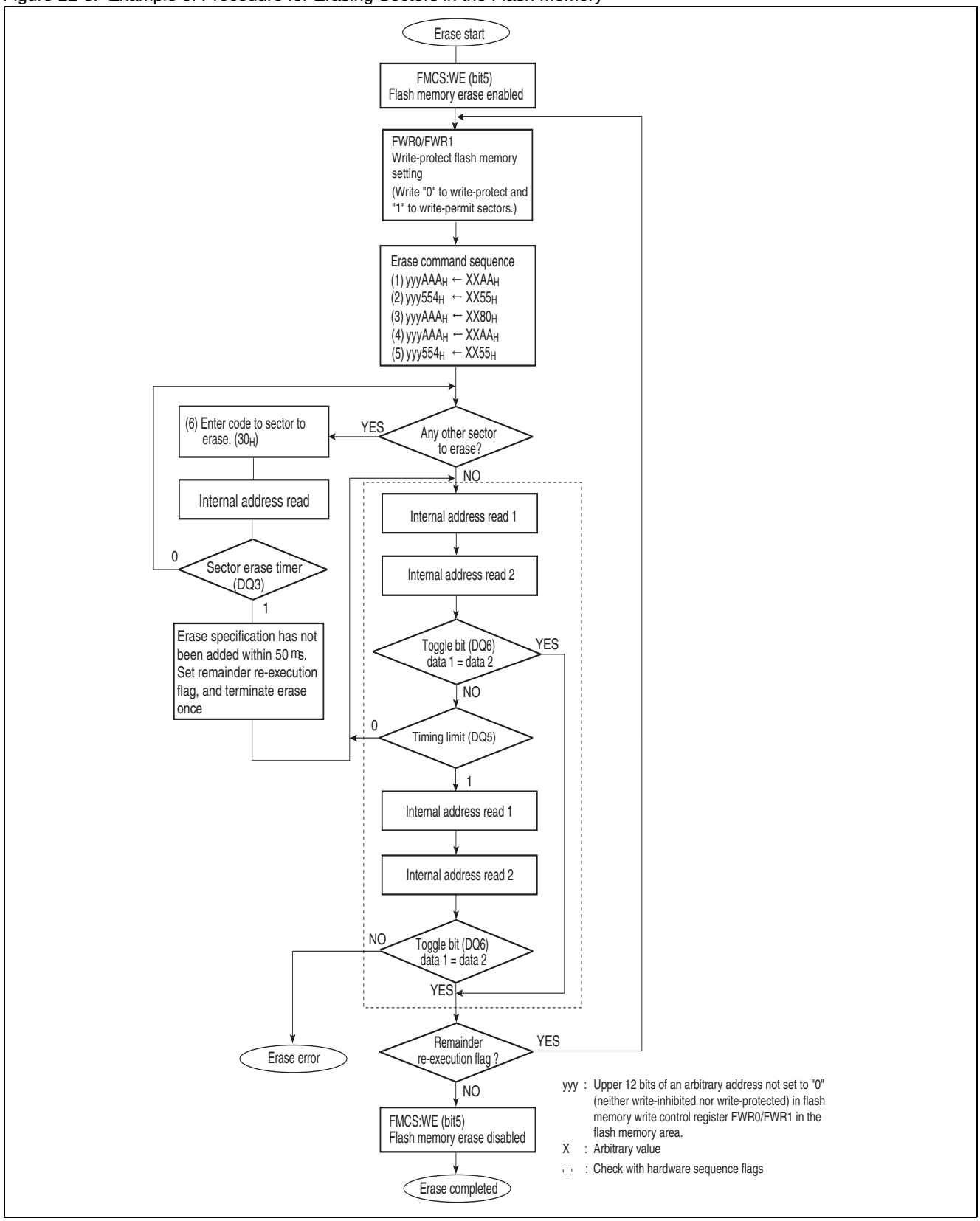
Figure 22-8 shows an example of the procedure for erasing sectors in the flash memory. Here, the toggle bit flag (DQ6) is used to confirm that erasing has terminated.

Note that the data to read to check the flag is read from the sector to be erased.

The toggle bit flag (DQ6) stops toggling its output, almost concurrently with the change of the timing limit exceeded flag (DQ5) to “1”. The toggle bit flag (DQ6) must therefore be rechecked even when the timing limit exceeded flag (DQ5) is “1” (processing in Table 22-8).

Since the data polling flag (DQ7) also changes at the same time as the timing limit exceeded flag (DQ5) changes, the data polling flag (DQ7) must be rechecked.

Figure 22-8. Example of Procedure for Erasing Sectors in the Flash Memory



22.7.5 Suspending Sector Erase of Flash Memory

This section describes the procedure for issuing the Sector Erase Suspend command to suspend erasing of flash memory sectors. During the suspension, data can be read from sectors that are not being erased.

Suspending Sector Erase of Flash Memory

Erasing of flash memory sectors can be suspended by writing the Sector Erase Suspended command in the command sequence table (see Table 22-5) to the flash memory area.

The Sector Erase Suspend command suspends the sector erase operation being executed and enables data to be read from sectors that are not being erased. In the sector erase suspended state, only reading is enabled; data cannot be written.

The Sector Erase Suspend command is valid only during sector erase operations which include a sector erase time-out period. The command will be ignored during chip erase or write operations.

The Sector Erase Suspend command is implemented by writing the erase suspend code ($B0_H$). At this time, specify an optional address in the flash memory for the address. An Erase Suspend command reissued during sector erase suspended will be ignored.

Entering the Sector Erase Suspend command during the sector erase time-out period will immediately terminate sector erase time-out period, cancel the erase operation, and set the erase stop state.

Entering the Erase Suspend command during the sector erase operation after the sector erase time-out period has terminated will set the erase suspend state after a maximum period of 20 μs has elapsed. The Sector Erase Suspend command should be entered after 20 μs or more is elapsed after the Sector Erase command or Sector Erase Restart command is issued.

22.7.6 Restarting Sector Erase of Flash Memory

This section describes the procedure for issuing the Sector Erase Restart command to restart suspended erasing of flash memory sectors.

Restarting Sector Erase of Flash Memory

To restart a suspended sector erase operation, write the Sector Erase Restart command in the command sequence table (see Table 22-5) to the flash memory area.

The Sector Erase Restart command is used to restart erasing of sectors from the sector erase suspended state set using the Sector Erase Suspend command. The Sector Erase Restart command is implemented by writing the erase restart code (30_H). At this time, specify the address of a sector in the flash memory area, which is allowed for writing operations.

If a Sector Erase Restart command is issued during sector erasure, the command will be ignored.

22.8 Flash Security Function

The flash security function enables to protect the contents in the flash memory. This feature is not available to the MB90F883B (H)(S) and MB90F884B (H)(S).

Overview

If protection code is written as data in the security bit, access to the flash memory can be restricted. Once the access to the flash memory is protected, the protected state cannot be released until the chip erase is performed. Data in the flash memory cannot be read/written from the external pins unless the protected state is released.

This function is compatible for the applications necessary for the security of self-contained programs and data stored in the flash memory.

The Protection Code and addresses of the security bit depend on the flash memory size. Table 22-16 shows the address of security bit.

Table 22-16. Address and Protection Code of Flash Security Bit

Model name	Flash memory size	Security bit address	Protection code
MB90F882A(S)	2M-bit flash memory	$FC0000_H$	$01XX_H$*
MB90F883C(S)	3M-bit flash memory	$F80000_H$	$01XX_H$*
MB90F884C(S)	4M-bit flash memory	$F80000_H$	$01XX_H$*

*: Any value is applied to the lower byte XX.

How to Apply Security

Write the protection code to the security bit. The security is applied after the external reset or power-on.

How to Release the Security

Execute the memory data erase function.

Operation with the Security Enabled

Read: invalid data is read

Write: cannot be written

Others

- To set the general-purpose parallel writer, follow the specification of parallel writer.
- It is recommended to write the protection code after programming the flash memory is completed. It enables to prevent the contents of the memory to be carelessly protected during programming.

Notes:

- Security bits are allocated in the flash memory area. Do not write protection code as data in the security bit, unless using the security function.
- Security cannot be applied for each sector by specifying the sector of the flash memory. The security function works for the entire area of the flash memory.
- Note that the flash memory fault cannot be analyzed with the security applied.

22.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems

This series has some restrictions on how to use the data polling flag (DQ7) during execution of the automatic sector erase algorithm. This section describes such restrictions and how to avoid related problems.

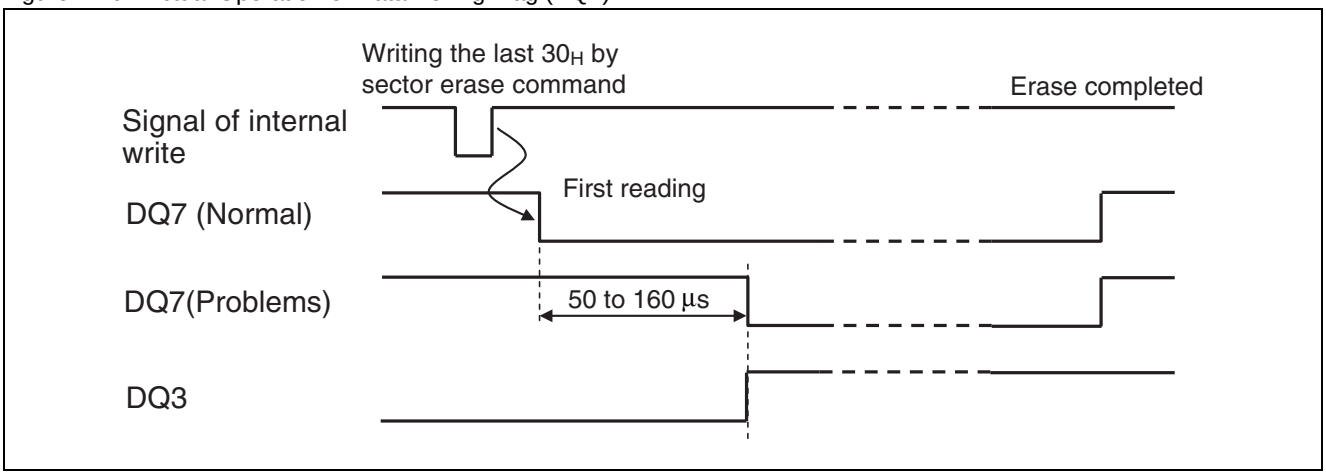
Description of Problems due to Restrictions

The data polling flag (DQ7) is used to indicate that the execution of the automatic algorithm is currently in progress or completed, by using the data polling function. In its original operation, as shown in Figure 22-9, DQ7 outputs "0" after the sector erase command is issued when the automatic algorithm is being started, and returns to "1" upon the completion of the erase operation. Therefore, the DQ7 polling algorithm indicates the completion of the erase operation by outputting "1".

In this series, DQ7 continues to output "1" for 50 to 160µs, after the Sector Erase command is issued, and then it outputs "0". When the erase operation is completed, it then returns to "1". For this reason, if the sector erase polling is started while "1" is still being output immediately after the sector erase command is issued, the erroneous judgment that the erase operation has been completed may occur, although the erase operation has not actually started.

The timing for DQ7 to change from "1" to "0" after the sector erase command is accepted is the same as the timing for the sector erase timer flag (DQ3), which indicates the sector erase timeout period, to change from "0" to "1".

Figure 22-9. Actual Operation of Data Polling Flag (DQ7)



The following or other problems may occur, as a result of the erroneous judgment that the erase operation has been completed,

1. Runaway or abnormal operation may occur, because the value of the sequence flag is read from the flash memory even when the CPU attempts to fetch instruction/data; therefore, the value of the program cannot be read properly.
2. If the next command is issued after the erroneous judgment that the sector erase operation has been completed occurs, the first command may be cancelled, resulting in a return to the read state, or the next command may not be accepted.

How to Avoid Problems

Use one of the following methods to avoid the problems.

Polling using the toggle bit flag (DQ6)

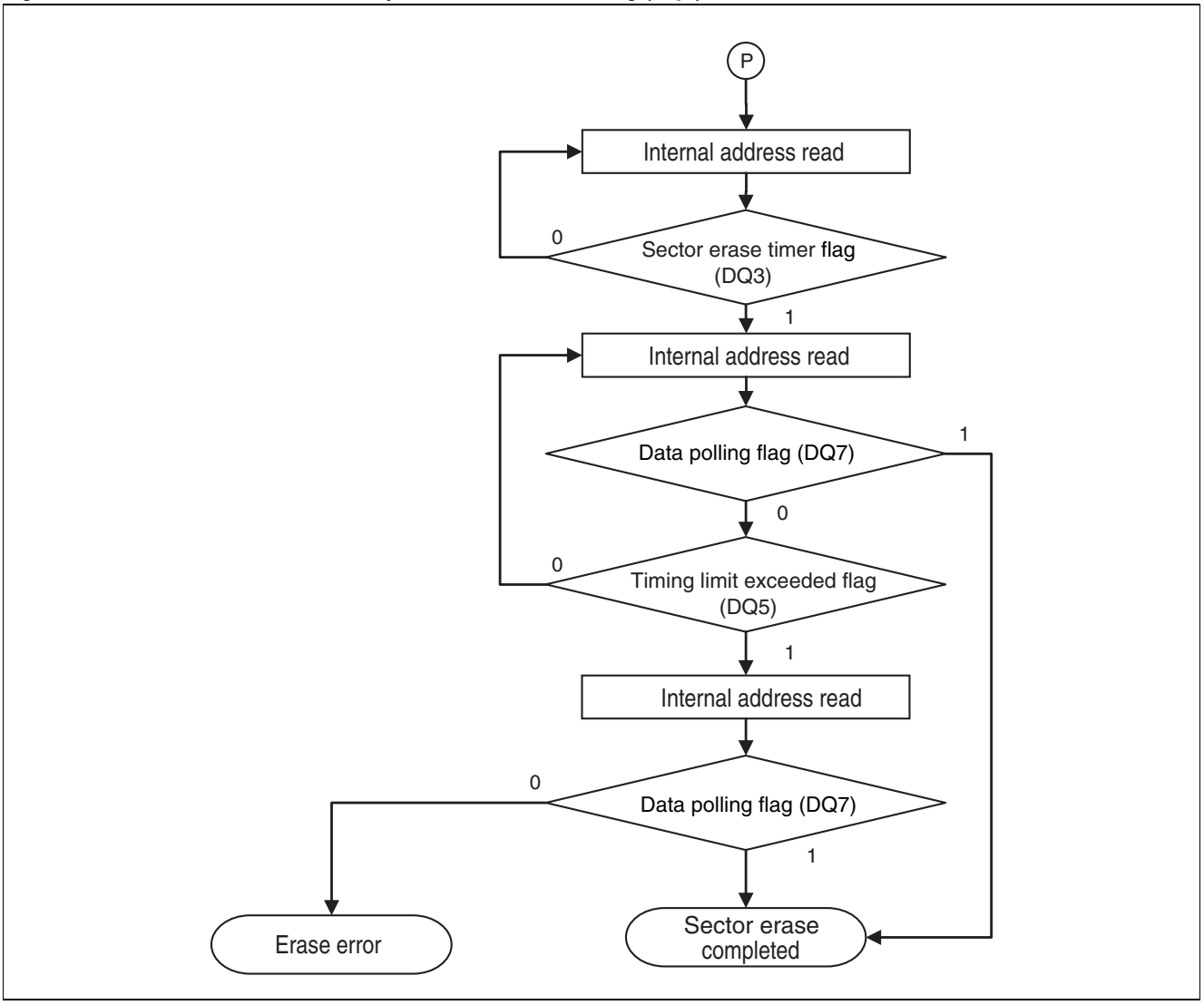
Determine the state of the automatic algorithm using DQ6, as shown in Figure 22-8 in "22.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase)".

In the same manner as the data polling flag (DQ7), the toggle bit flag (DQ6) indicates that the automatic algorithm is being executed or has terminated by the toggle bit function.

Starting polling of DQ7 after the sector erase timeout period elapses

Before starting the polling of DQ7, wait for 160µs or more by software after the sector erase command is issued, or wait until DQ3 is set to "1" (end of the sector erase timeout period). Figure 22-10 shows the judgment method using DQ3 after the sector erase command is issued.

Figure 22-10. How to Avoid Problems by Sector Erase Timer Flag (DQ3)

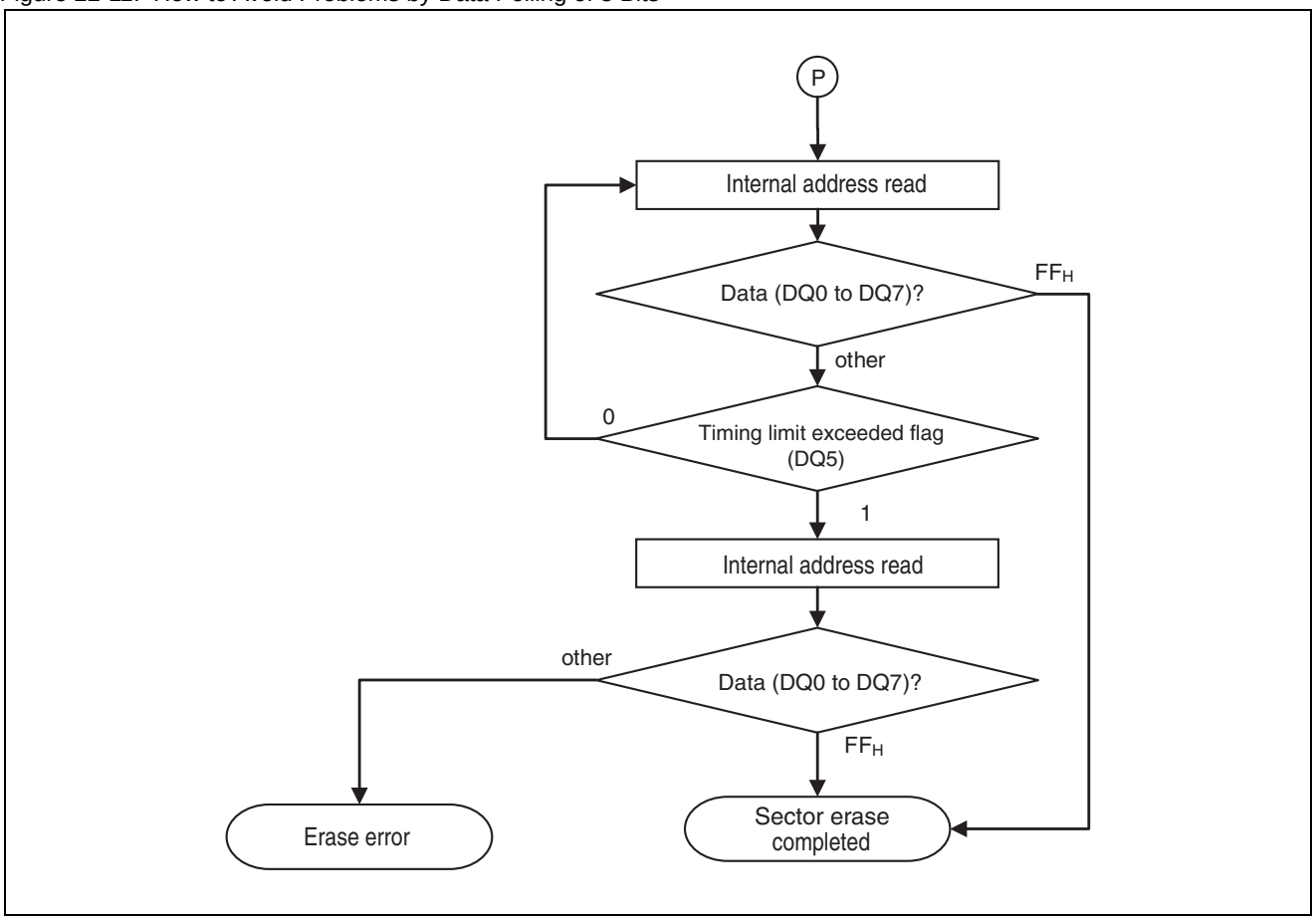


Data polling using the 8 bits of hardware sequence flags

Make a judgment by data polling using the 8 bits of hardware sequence flags, rather than using only the polling of DQ7.

Figure 22-11 shows the judgment method using the data polling of the 8 bits after the sector erase command is issued.

Figure 22-11. How to Avoid Problems by Data Polling of 8 Bits



22.10 Notes on Using Flash Memory

This section provides notes on using flash memory.

Notes on Using Flash Memory

Input of hardware reset ($\overline{RST}$)

To input a hardware reset when the automatic algorithm is not activated, a minimum of 500 ns should be taken as the “L” level width. In this case, a minimum of 700 ns is required until data can be read from flash memory after a hardware reset has been activated.

Similarly, to input a hardware reset when the automatic algorithm is activated, where data-writing/erasing is in progress, a minimum of 500 ns should be taken as the “L” level width. In this case, 20 μs are required until data can be read after the operation being executed for initializing the flash memory is terminated.

A hardware reset during writing makes data being written undefined.

Program access to flash memory

When the automatic algorithm is operating, read access to the flash memory is prohibited. With the CPU's memory access mode set to the internal ROM mode, data-writing/erasing should be started after switching the program area to another area such as RAM. In this case, when sectors containing interrupt vectors are erased, data write/erase interrupt processing cannot be executed.

To make sure that flash memory data-writing/erasing has been started or completed, check the hardware sequence flags as well as the RDYINT or RDY flag. This is because, when a sector erase command is issued, for example, the sector erase operation is not started during the sector erase wait period even after the RDY flag is set to "0", where DQ3 (sector erase timer flag) must be checked to make sure that the sector erase operation is started. When there is some source of setting the DQ5 (timing limit excess flag) at the checking time of the end of the automatic algorithm, the RDYINT or RDY flag rejects to be set, where the program may enter an infinite loop if only these flags are referenced.

Extended intelligent I/O service (EI^{2}OS)

The program/erase interrupt by automatic algorithm end issued from the flash memory interface circuit to the CPU is not acceptable to DMA and EI^{2}OS and thus neither feature is available.

Cancellation of software reset and watchdog timer reset

When reset conditions are satisfied while the automatic algorithm is in the active state during data-writing/erasing operation, CPU may run away. This is because the flash memory may not yet be in the read state when the automatic algorithm is continued without initializing the flash memory due to the reset conditions and CPU starts a sequence after reset is released. It is necessary to prohibit the reset conditions while writing data to or erasing data from the flash memory.

23. Examples Of Flash Memory Products Serial Programming Connection

This chapter shows an example of a serial programming connection using the AF220/AF210/AF120/AF110 flash microcontroller programmer by Yokogawa Digital Computer Corporation.

23.1 Basic Configuration of Flash Memory Products Serial Programming Connection

23.2 Examples of Serial Programming Connections

23.1 Basic Configuration of Flash Memory Products Serial Programming Connection

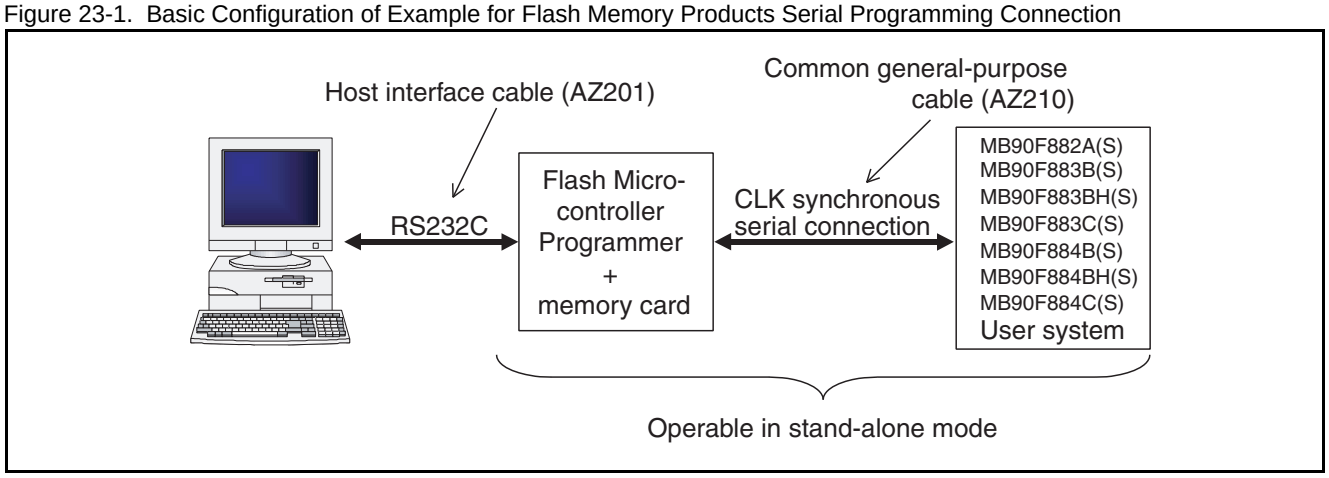
The MB90F882A(S)/MB90F883B(S)/MB90F883BH(S)/MB90F883C(S)/MB90F884B(S)/MB90F884BH(S)/MB90F884C(S) supports serial on-board writing (Cypress standard) of the flash memory. This section provides the related specifications.

Basic Configuration of Flash Memory Products Serial Programming Connection

Cypress standard serial on-board writing uses the Yokogawa Digital Computer Corporation flash microcontroller programmer. It is possible to write by selecting either of the program that operates in the single-chip mode or the internal ROM external bus mode.

Figure 23-1 shows the basic configuration for the example for serial programming connection.

Figure 23-1. Basic Configuration of Example for Flash Memory Products Serial Programming Connection



For information on the functions of and operational procedures related to the flash microcontroller programmer (AF220/AF210/AF120/AF110), the general-purpose common cable (AZ210) for connection, and the connector, contact Yokogawa Digital Computer Corporation.

Pins Used for Cypress Standard Serial On-Board Writing

Table 23-1 shows the functions of the related pins used for Cypress standard serial on-board writing.

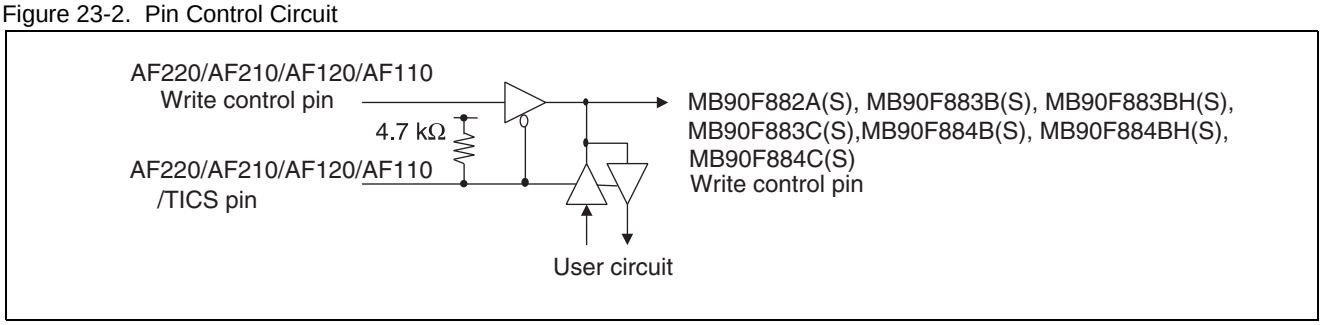
Table 23-1. Function of Pins

Pin	Function	Additional information
MD2, MD1, MD0	Mode pins	Setting MD2=1, MD1=1, and MD0=0 to enter the serial programming mode.
X0, X1	Oscillation pins	As, in the serial programming mode, CPU internal operation clock is the PLL clock multiplied-by-1, the internal operation clock frequency is equal to the oscillation clock frequency. Consequently, the frequencies that can be input to the high-speed oscillation input pin for serial writing are from 4 MHz to 25 MHz.
P00, P01	Programming program start pins	Input “L” level to P00, and “H” level to P01
RST	Reset pin	-
UI0	Serial data input pin	Use UART0 for CLK synch mode.
UO0	Serial data output pin	
UCK0	Serial clock input pin	
V_{CC}	Power voltage supply pin	Programming voltage (V_{CC} = 3.3 V ± 5%)
V_{SS}	GND pin	Must be shared with GND of the flash microcontroller programmer.

To use the P00, P01, UI0 and UCK0 pins within the user system as well, the control circuit shown in the Figure 23-2 is required.

Using the flash microcontroller programmer’s /TICS signal for outputting “L”, the user circuit can be disconnected in serial programming mode.

Figure 23-2. Pin Control Circuit



Oscillation Clock Frequency and Serial Clock Input Frequency

The serial clock frequencies that can be used for input to the MB90F882A(S), MB90F883B(S), MB90F883BH(S), MB90F883C(S), MB90F884B(S), MB90F884BH(S), and MB90F884C(S) can be calculated from the following formula.

Use the flash microcontroller programmer settings to set the serial clock input frequency for the required oscillation clock frequency.

Serial clock frequency to be input = 0.125 × oscillation clock frequency

Table 23-2 shows a serial clock frequency that can be input.

Table 23-2. Example of Serial Clock Frequency that can be Input

Oscillation clock frequency	Maximum serial clock frequency that can be input to microcontroller	Maximum serial clock frequency that can be set for AF220/AF210/AF120/AF110	Maximum serial clock frequency that can be set for AF200
8 MHz	1MHz	850kHz	500kHz
16 MHz	2MHz	1.25MHz	500kHz

System Configuration of Flash Microcontroller Programmer (Manufactured by Yokogawa Digital Computer Corporation)

Table 23-3 shows the system configuration of the flash microcontroller programmer.

Table 23-3. System Configuration of the Flash Microcontroller Programmer

Type		Function	
Main body	AF220/AC4P	Model with built-in Ethernet interface	/100 V to 220 V power adapter
	AF210/AC4P	Standard model	/100 V to 220 V power adapter
	AF120/AC4P	Model with built-in single key Ethernet interface	/100 V to 220 V power adapter
	AF110/AC4P	Single key model	/100 V to 220 V power adapter
AZ221		Programmer dedicated RS232C cable for PC/AT	
AZ210		Standard target probe (a) length: 1 m	
FF201		Cypress F^2MC-16LX flash microcontroller control module	
AZ290		Remote controller	
/P2		2M bytes PC Card (Option) FLASH memory capacity of up to 128 Kbytes supported	
/P4		4M bytes PC Card (Option) FLASH memory capacity of up to 512 Kbytes supported	

Inquiries: Yokogawa Digital Computer Corporation

Telephone number: (81)-42-333-6224

23.2 Examples of Serial Programming Connections

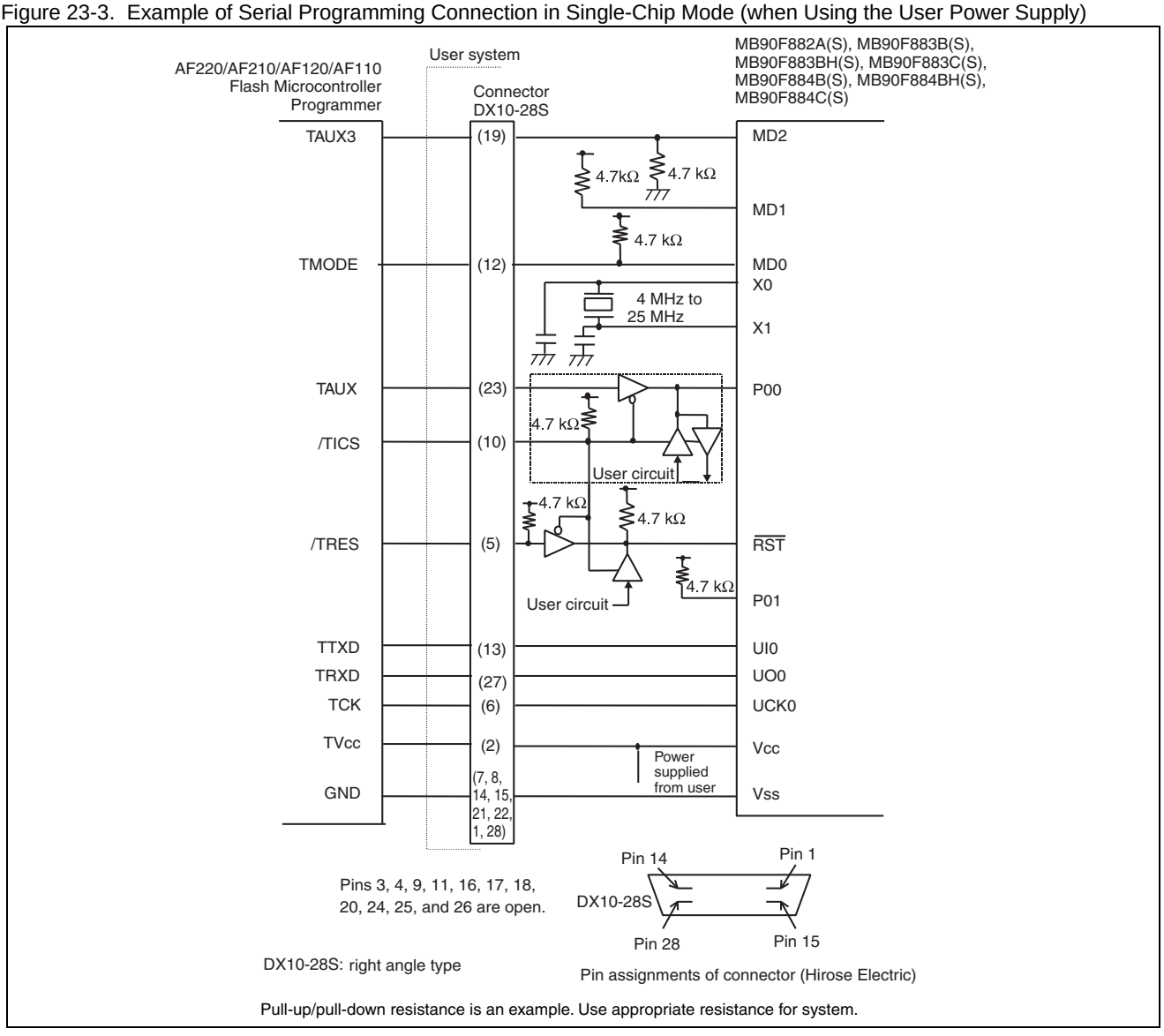
Examples for the following two types of connections are shown below.

- Example of connection in single-chip mode (When Using the User Power Supply)
- Example of connection with flash microcontroller programmer (When Using the User Power Supply)

Example of Connection in Single-Chip Mode (When Using the User Power Supply)

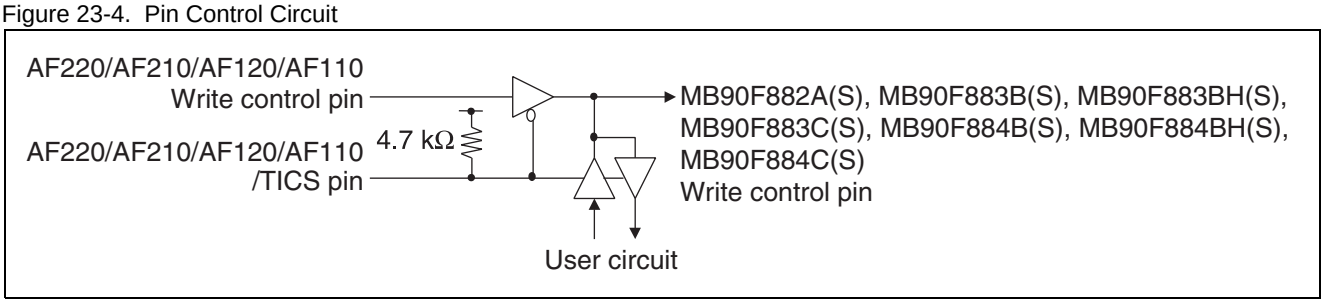
In the user system, mode pins MD2 and MD0, which are set to single-chip mode, are supplied with the inputs MD2=1 and MD0=0 by TAUX3 and TMODE of AF220/AF210/AF120/AF110, and the system is set to serial programming mode (serial programming mode: MD2, MD1, MD0=110_B).

Figure 23-3. Example of Serial Programming Connection in Single-Chip Mode (when Using the User Power Supply)



Similarly to P00, using the UI0, UO0 and UCK0 pins in the user system requires a control circuit as shown in Figure 23-4. The user circuit is disconnected in serial programming mode by the flash microcontroller programmer's "/TICS" signal for outputting "L".

Figure 23-4. Pin Control Circuit



Connect to AF220/AF210/AF120/AF110 when the user power supply is turned off.

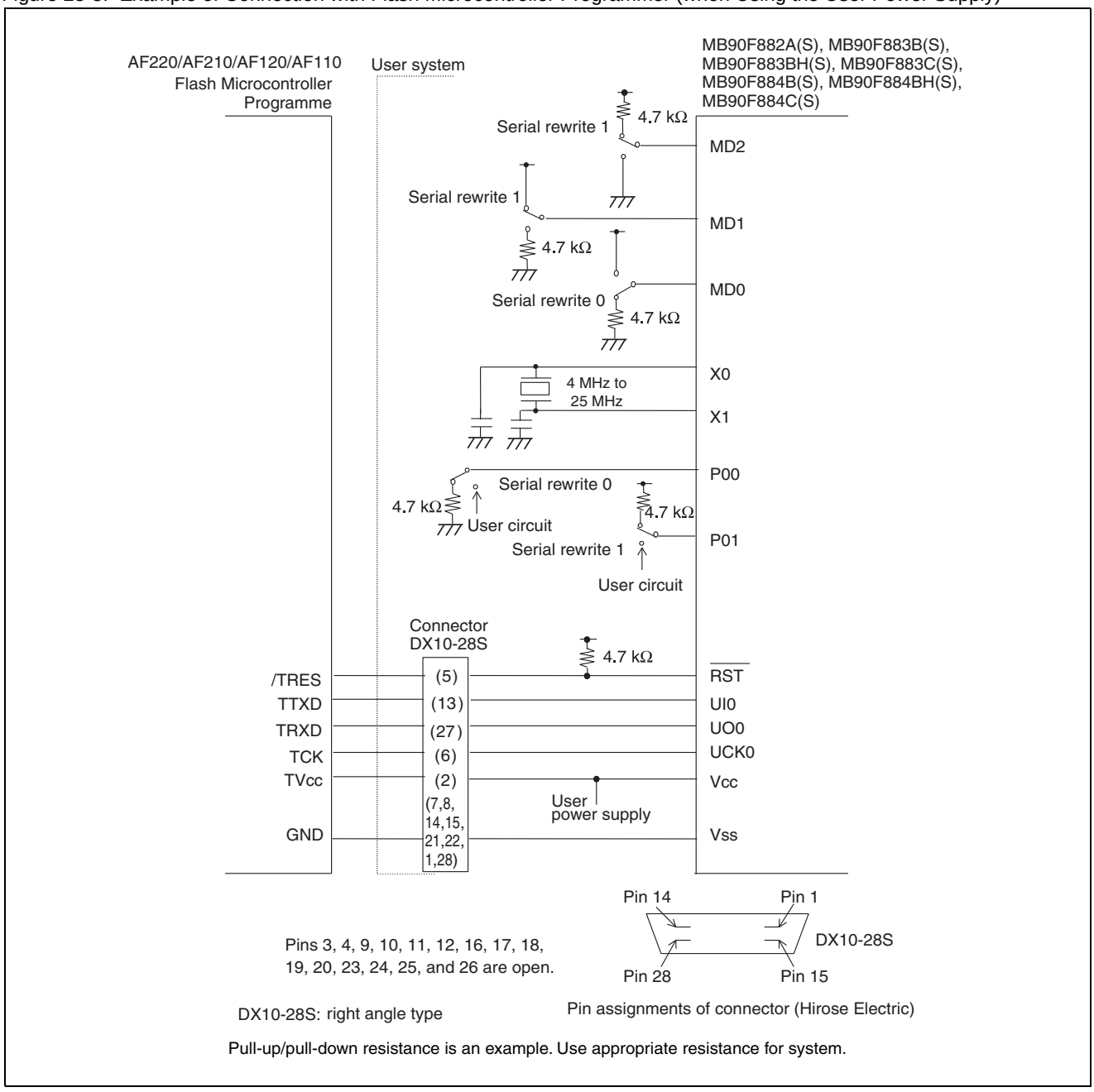
Pull-up resistance is an example. Use appropriate resistance for system.

Example of Connection with Flash Microcontroller Programmer (When Using the User Power Supply)

If, in serial programming mode, pins (MD2, MD0, and P00) are set as shown below, MD2, MD0, and P00 do not need to be connected with the flash microcontroller programmer.

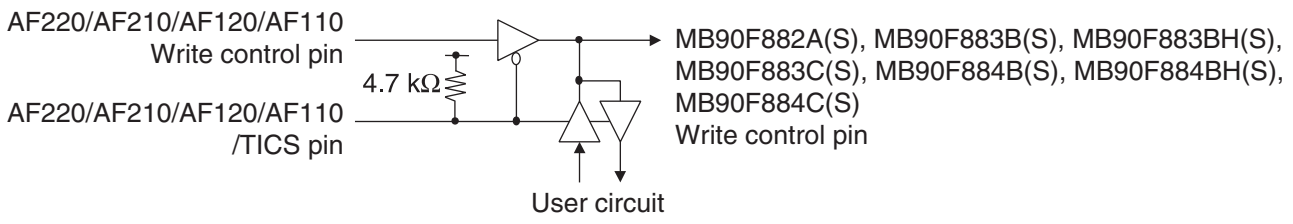
Figure 23-5 shows an example of connection with the flash microcontroller programmer.

Figure 23-5. Example of Connection with Flash Microcontroller Programmer (when Using the User Power Supply)



Using the pins UI0, UO0 and UCK0 in the user system requires a control circuit as shown in Figure 23-6. The user circuit is disconnected in serial programming mode by the flash microcontroller programmer's "/TICS" signal for outputting "L".

Figure 23-6. Pin Control Circuit



Connect to AF220/AF210/AF120/AF110 when the user power supply is turned off.

Pull-up resistance is an example. Use appropriate resistance for system.

24. Delay Interrupt Generation Module

This chapter describes the functions and operations of the delay interrupt generation module.

24.1 Overview of Delay Interrupt Generation Module

24.2 Operation of Delay Interrupt Generation Module

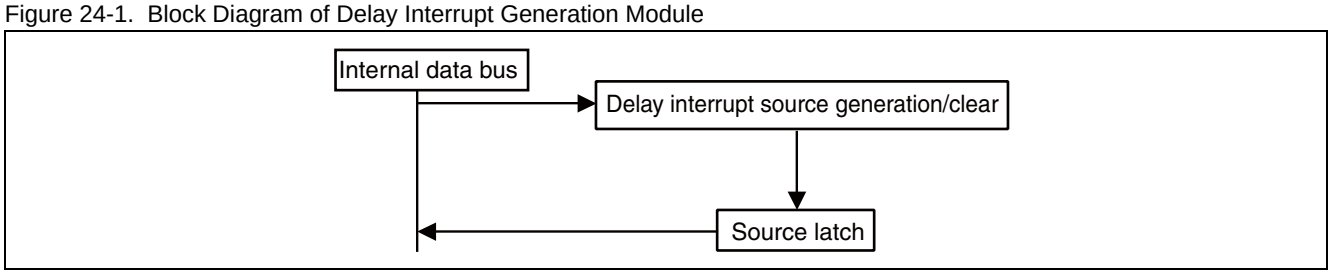
24.1 Overview of Delay Interrupt Generation Module

The delay interrupt generation module is used to generate an interrupt for task switching. Using this module enables software to issue/cancel an interrupt request to F^{2}MC-16LX CPU.

Block Diagram of Delay Interrupt Generation Module

Figure 24-1 shows a block diagram of the delay interrupt generation module.

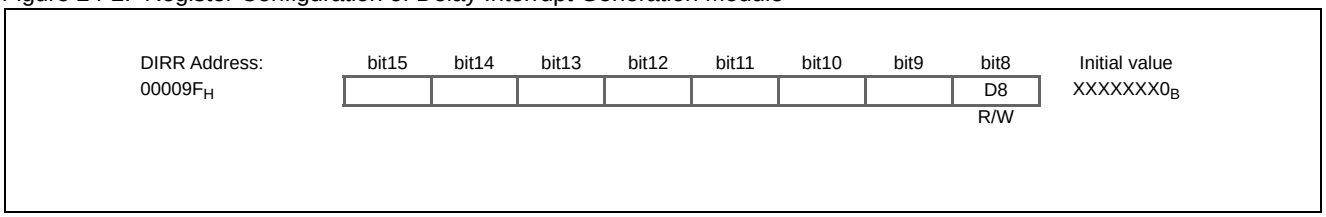
Figure 24-1. Block Diagram of Delay Interrupt Generation Module



List of Registers of Delay Interrupt Generation Module

The following figure shows the register configuration of the delay generation module [delay interrupt source generation/clear register (DIRR: Delayed Interrupt Request Register)].

Figure 24-2. Register Configuration of Delay Interrupt Generation Module



This register becomes a source clear state when reset.

DIRR is used to generate/cancel a delay interrupt request. Writing "1" to this register generates a delay interrupt request; writing "0" cancels the request. This register becomes a source clear state when reset. Although the undefined bit area can be written to either “0” or “1”, in consideration of future extensions, it is recommended that the set bit and clear bit instructions are used to access this register.

Interrupts of Delay Interrupt Generation Module and EI^{2}OS

Table 24-1 lists the interrupts of the delay interrupt generation module and EI^{2}OS.

Table 24-1. Interrupts of Delay Interrupt Generation Module and EI^{2}OS

Interrupt No.	Interrupt level setting register		Vector table address			EI^{2}OS
	Register name	Address	Lower	Upper	Bank	
#42(2A$_H$)	ICR15	0000BF$_H$	FFFF54$_H$	FFFF55$_H$	FFFF56$_H$	X

X: Not available

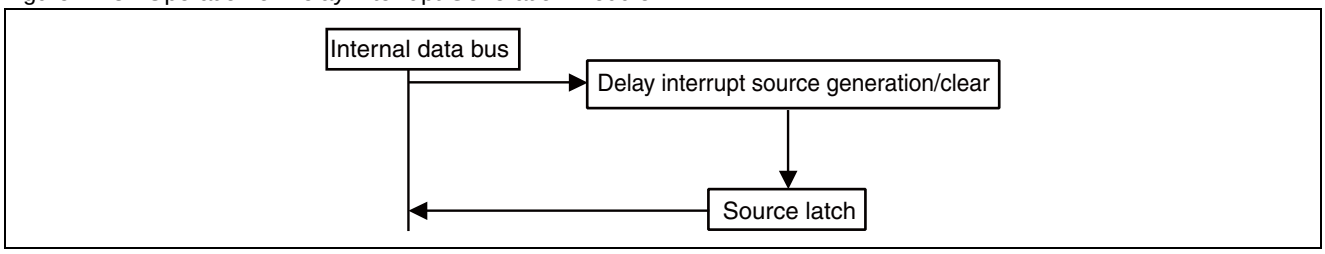
24.2 Operation of Delay Interrupt Generation Module

When the CPU writes “1” to the relevant bit in DIRR by software, the request latch in the delay interrupt generation module is set to generate an interrupt request to the interrupt controller.

Operation of Delay Interrupt Generation Module

When the CPU writes “1” to the relevant bit in DIRR by software, the request latch in the delay interrupt generation module is set to generate an interrupt request to the interrupt controller. If any other interrupt request has a priority lower than that of the interrupt from the delay interrupt generation module, or if there are no other interrupt requests, the interrupt controller issues an interrupt request to the F^{2}MC-16LX CPU. The F^{2}MC-16LX CPU compares the ILM bit in its internal request. If the request level is higher than that of the ILM bit, the CPU starts a hardware interrupt handling microprogram immediately after the instruction in current execution is completed. As a result, the interrupt handling routine is executed in response to the interrupt generated by the delay interrupt generation module. Writing “0” to the relevant bit of DIRR within the interrupt handling routine clears the interrupt source of the delay interrupt generation module, and also switches the task. Figure 24-3 shows the operation of the delay interrupt generation module.

Figure 24-3. Operation of Delay Interrupt Generation Module



Notes on Using Delay Interrupt Generation Module

Delay interrupt request latch

This latch is set by writing “1” to the relevant bit of DIRR, and cleared by writing “0” to that bit. Therefore, note that the interrupt handling starts again immediately after the return from the interrupt handling unless the software has been designed to clear the source within the interrupt handling routine.

A. Appendix

The appendix provides the memory map and lists the instructions used in the F^{2}MC-16LX.

A.1 Memory Map

A.2 I/O Map

A.3 Interrupt Source, Interrupt Vector, and Interrupt Control Register

A.4 Instructions

A.1 Memory Map

Memory space is divided according to three usage modes.

Memory Space

The memory space is divided according to three usage modes shown in Figure A-1.

Figure A-1. Memory Map

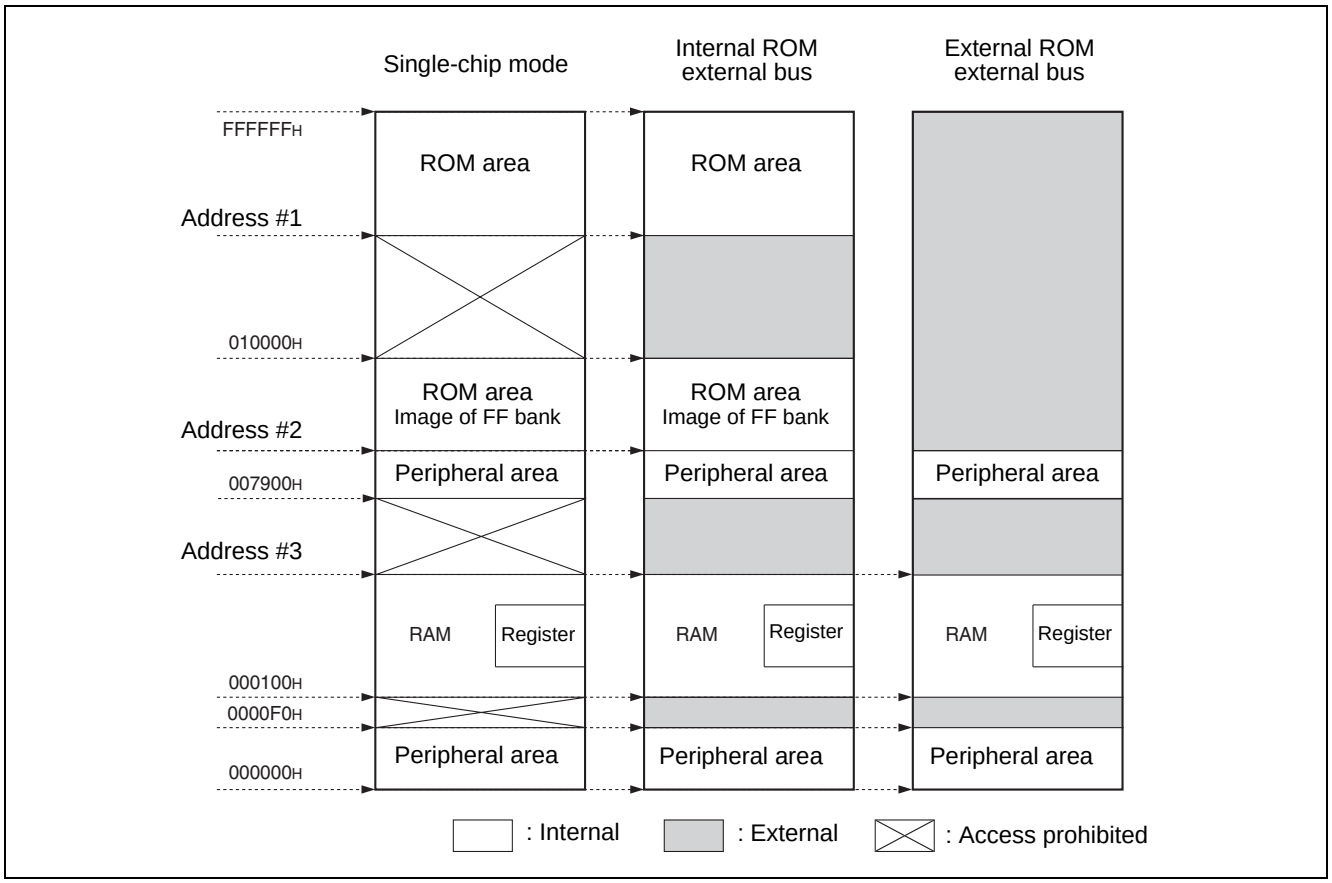


Table A-1 Lists the correspondence of address # 1 and address # 2 and address # 3 of each product.

Table A-1. Lists correspondence of address # 1 and address # 2 and address # 3 of each product

Product No.	Address #1	Address #2	Address #3
MB90882(S)	$FC0000_H$	008000_H fixed	004100_H
MB90F882A(S)	$FC0000_H$		004100_H
MB90F883B(S)/ MB90F883BH(S)/ MB90F883C(S)	$FA0000_H$		006100_H
MB90F884B(S)/ MB90F884BH(S)/ MB90F884C(S)	$F80000_H$		007900_H
MB90V880A-101/102	($F80000_H$)		007900_H

Note:

The image of the ROM data in the FF bank appears at the top of the 00 bank in order to enable efficient use of the C compiler small model. The lower 16-bit address for the FF bank will be assigned to the same address for the 00 bank, so that tables in ROM can be referenced without declaring a “far” indication with the pointer.
For example, when accessing the address $00C000_H$, the actual access is performed to address $FFC000_H$ in ROM.
Here the FF bank ROM area exceeds 32 KB, it is not possible to see the entire area in the 00 bank image. Therefore, the ROM data in $FF8000_H$ to $FFFFFF_H$ can be seen in the 00 bank image, while the data in $FF0000_H$ to $FF7FFF_H$ can only be seen in the FF bank.
In MB90F883B(S), MB90F883BH(S), and MB90F883C(S), the area in a range of 006100_H to $0078FF_H$ and $F80000_H$ to $F9FFFF_H$ cannot be used as an external area.

A.2 I/O Map

Table A-2 shows the addresses assigned to the registers for each peripheral function.

I/O Maps

Table A-2. I/O Map (Sheet 1 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
000000_H	PDR0	Port 0 data register	R/W	Port 0	XXXXXXXX_B
000001_H	PDR1	Port 1 data register	R/W	Port 1	XXXXXXXX_B
000002_H	PDR2	Port 2 data register	R/W	Port 2	XXXXXXXX_B
000003_H	PDR3	Port 3 data register	R/W	Port 3	XXXXXXXX_B
000004_H	PDR4	Port 4 data register	R/W	Port 4	XXXXXXXX_B
000005_H	PDR5	Port 5 data register	R/W	Port 5	XXXXXXXX_B
000006_H	PDR6	Port 6 data register	R/W	Port 6	XXXXXXXX_B
000007_H	PDR7	Port 7 data register	R/W	Port 7	XXXXXXXX_B
000008_H	PDR8	Port 8 data register	R/W	Port 8	XXXXXXXX_B
000009_H	PDR9	Port 9 data register	R/W	Port 9	XXXXXXXX_B
$00000A_H$	PDRA	Port A data register	R/W	Port A	XXXXXXXX_B
$00000B_H$	UDER	Up-down timer input enable register	R/W	Up-down timer input control	$XX000000_B$
$00000C_H$	ILSR0	Serial input level selection register 0	R/W	Multi-function serial control	00000000_B
$00000D_H$	ILSR1	Serial input level selection register 1	R/W		00000000_B
$00000E_H$	ILSR2	Serial input level selection register 2	R/W		$---00000_B$
$00000F_H$	Reserved				
000010_H	DDR0	Port 0 direction register	R/W	Port 0	00000000_B
000011_H	DDR1	Port 1 direction register	R/W	Port 1	00000000_B
000012_H	DDR2	Port 2 direction register	R/W	Port 2	00000000_B
000013_H	DDR3	Port 3 direction register	R/W	Port 3	00000000_B
000014_H	DDR4	Port 4 direction register	R/W	Port 4	00000000_B
000015_H	DDR5	Port 5 direction register	R/W	Port 5	00000000_B
000016_H	DDR6	Port 6 direction register	R/W	Port 6	00000000_B
000017_H	DDR7	Port 7 direction register	R/W	Port 7	-0000000_B
000018_H	DDR8	Port 8 direction register	R/W	Port 8	00000000_B
000019_H	DDR9	Port 9 direction register	R/W	Port 9	00000000_B
$00001A_H$	DDRA	Port A direction register	R/W	Port A	$----0000_B$
$00001B_H$	ADER0	Analog input enable register 0	R/W	Port 6, A/D	11111111_B
$00001C_H$	ADER1	Analog input enable register 1	R/W	Port 9, A/D	11111111_B
$00001D_H$	ADER2	Analog input enable register 2	R/W	Port 7, A/D	$----1111_B$
$00001E_H$	RDR0	Port 0 input resistance register	R/W	Port 0 (pull-up resistance control)	00000000_B
$00001F_H$	RDR1	Port 1 input resistance register	R/W	Port 1 (pull-up resistance control)	00000000_B

Table A-2. I/O Map (Sheet 2 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
000020_H	SMR0	Serial bus mode register ch.0	R/W	Multi-function serial ch.0	$\$\$\$\$\$\$\$_B$
000021_H	SCR0/IBCR0	Serial bus control register/ I^2C bus control register ch.0	R/W		$\$\$\$\$\$\$\$_B$
000022_H	ESCR0/IBSR0	Extended communication control register/ I^2C bus status register ch.0	R/W		$\$\$\$\$\$\$\$_B$
000023_H	SSR0	Serial status register ch.0	R/W		$\$\$\$\$\$\$\$_B$
000024_H	RDR00/TDR00	Transmission/reception data register 0 ch.0	R,W		$\$\$\$\$\$\$\$_B$
000025_H	RDR10/TDR10	Transmission/reception data register 1 ch.0	R,W		$\$\$\$\$\$\$\$_B$
000026_H	BGR00	Baud rate generator register 0 ch.0	R/W		$\$\$\$\$\$\$\$_B$
000027_H	BGR10	Baud rate generator register 1 ch.0	R/W		$\$\$\$\$\$\$\$_B$
000028_H	ISBA0	7-bit slave address register ch.0	R/W		00000000_B
000029_H	ISMK0	7-bit slave address mask register ch.0	R/W		01111111_B
$00002A_H$	SMR1	Serial bus mode register ch.1	R/W	Multi-function serial ch.1	$\$\$\$\$\$\$\$_B$
$00002B_H$	SCR1/IBCR1	Serial bus control register / I^2C bus control register ch.1	R/W		$\$\$\$\$\$\$\$_B$
$00002C_H$	ESCR1/IBSR1	Extended communication control register / I^2C bus status register ch.1	R/W		$\$\$\$\$\$\$\$_B$
$00002D_H$	SSR1	Serial status register ch.1	R/W		$\$\$\$\$\$\$\$_B$
$00002E_H$	RDR01/TDR01	Transmission/reception data register 0 ch.1	R,W		$\$\$\$\$\$\$\$_B$
$00002F_H$	RDR11/TDR11	Transmission/reception data register 1 ch.1	R,W		$\$\$\$\$\$\$\$_B$
000030_H	BGR01	Baud rate generator register 0 ch.1	R/W		$\$\$\$\$\$\$\$_B$
000031_H	BGR11	Baud rate generator register 1 ch.1	R/W		$\$\$\$\$\$\$\$_B$
000032_H	ISBA1	7-bit slave address register ch.1	R/W		00000000_B
000033_H	ISMK1	7-bit slave address mask register ch.1	R/W		01111111_B
000034_H	ADCSL	Lower A/D control status register	R/W	A/D Converter	00011110_B
000035_H	ADCSH	Upper A/D control status register	R/W		00000000_B
000036_H	ADCRL	Lower A/D data register	R		XXXXXXXX_B
000037_H	ADCRH	Upper A/D data register	R		$111111XX_B$
000038_H	ADSRL	Lower A/D conversion channel setting register	R/W		00000000_B
000039_H	ADSRH	Upper A/D conversion channel setting register	R/W		00000000_B
$00003A_H$	Use prohibition				
$00003B_H$	PACSR1	Address detection control status register 1	R/W	Address match detection function	00000000_B
$00003C_H$	OLSR0	Output level selection register 0	R/W	Port 7 (N-ch open-drain control)	$-000----_B$
$00003D_H$	OLSR1	Output level selection register 1	R/W	Port 8 (N-ch open-drain control)	00000000_B

Table A-2. I/O Map (Sheet 3 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$00003E_H$	SMR2	Serial bus mode register ch.2	R/W	Multi-function serial ch.2	$\$\$\$\$\$\$\$_B$
$00003F_H$	SCR2/IBCR2	Serial bus control register / I^2C bus control register ch.2	R/W		$\$\$\$\$\$\$\$_B$
000040_H	ESCR2/IBSR2	Extended communication control register / I^2C bus status register ch.2	R/W		$\$\$\$\$\$\$\$_B$
000041_H	SSR2	Serial status register ch.2	R/W		$\$\$\$\$\$\$\$_B$
000042_H	RDR02/TDR02	Transmission/reception data register 0 ch.2	R,W		$\$\$\$\$\$\$\$_B$
000043_H	RDR12/TDR12	Transmission/reception data register 1 ch.2	R,W		$\$\$\$\$\$\$\$_B$
000044_H	BGR02	Baud rate generator register 0 ch.2	R/W		$\$\$\$\$\$\$\$_B$
000045_H	BGR12	Baud rate generator register 1 ch.2	R/W		$\$\$\$\$\$\$\$_B$
000046_H	ISBA2	7-bit slave address register ch.2	R/W		00000000_B
000047_H	ISMK2	7-bit slave address mask register ch.2	R/W		01111111_B
000048_H	SMR3	Serial bus mode register ch.3	R/W	Multi-function serial ch.3	$\$\$\$\$\$\$\$_B$
000049_H	SCR3/IBCR3	Serial bus control register / I^2C bus control register ch.3	R/W		$\$\$\$\$\$\$\$_B$
$00004A_H$	ESCR3/IBSR3	Extended communication control register / I^2C bus status register ch.3	R/W		$\$\$\$\$\$\$\$_B$
$00004B_H$	SSR3	Serial status register ch.3	R/W		$\$\$\$\$\$\$\$_B$
$00004C_H$	RDR03/TDR03	Transmission/reception data register 0 ch.3	R,W		$\$\$\$\$\$\$\$_B$
$00004D_H$	RDR13/TDR13	Transmission/reception data register 1 ch.3	R,W		$\$\$\$\$\$\$\$_B$
$00004E_H$	BGR03	Baud rate generator register 0 ch.3	R/W		$\$\$\$\$\$\$\$_B$
$00004F_H$	BGR13	Baud rate generator register 1 ch.3	R/W		$\$\$\$\$\$\$\$_B$
000050_H	ISBA3	7-bit slave address register ch.3	R/W		00000000_B
000051_H	ISMK3	7-bit slave address mask register ch.3	R/W		01111111_B
000052_H	SMR4	Serial bus mode register ch.4	R/W	Multi-function serial ch.4	$\$\$\$\$\$\$\$_B$
000053_H	SCR4/IBCR4	Serial bus control register / I^2C bus control register ch.4	R/W		$\$\$\$\$\$\$\$_B$
000054_H	ESCR4/IBSR4	Extended communication control register / I^2C bus status register ch.4	R/W	Multi-function serial ch.4	$\$\$\$\$\$\$\$_B$
000055_H	SSR4	Serial status register ch.4	R/W		$\$\$\$\$\$\$\$_B$
000056_H	RDR04/TDR04	Transmission/reception data register 0 ch.4	R,W		$\$\$\$\$\$\$\$_B$
000057_H	RDR14/TDR14	Transmission/reception data register 1 ch.4	R,W		$\$\$\$\$\$\$\$_B$
000058_H	BGR04	Baud rate generator register 0 ch.4	R/W		$\$\$\$\$\$\$\$_B$
000059_H	BGR14	Baud rate generator register 1 ch.4	R/W		$\$\$\$\$\$\$\$_B$
$00005A_H$	ISBA4	7-bit slave address register ch.4	R/W		00000000_B
$00005B_H$	ISMK4	7-bit slave address mask register ch.4	R/W		01111111_B

Table A-2. I/O Map (Sheet 4 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$00005C_H$	SMR5	Serial bus mode register ch.5	R/W	Multi-function serial ch.5	$\$\$\$\$\$\$\$_B$
$00005D_H$	SCR5/IBCR5	Serial bus control register / I^2C bus control register ch.5	R/W		$\$\$\$\$\$\$\$_B$
$00005E_H$	ESCR5/IBSR5	Extended communication control register / I^2C bus status register ch.5	R/W		$\$\$\$\$\$\$\$_B$
$00005F_H$	SSR5	Serial status register ch.5	R/W		$\$\$\$\$\$\$\$_B$
000060_H	RDR05/TDR05	Transmission/reception data register 0 ch.5	R,W		$\$\$\$\$\$\$\$_B$
000061_H	RDR15/TDR15	Transmission/reception data register 1 ch.5	R,W		$\$\$\$\$\$\$\$_B$
000062_H	BGR05	Baud rate generator register 0 ch.5	R/W		$\$\$\$\$\$\$\$_B$
000063_H	BGR15	Baud rate generator register 1 ch.5	R/W		$\$\$\$\$\$\$\$_B$
000064_H	ISBA5	7-bit slave address register ch.5	R/W		00000000_B
000065_H	ISMK5	7-bit slave address mask register ch.5	R/W		01111111_B
000066_H	OCCP0	Lower output compare register (ch.0)	R/W	16-bit output timer output compare (ch.0 to ch.5)	00000000_B
000067_H		Upper output compare register (ch.0)			00000000_B
000068_H	OCCP1	Lower output compare register (ch.1)	R/W		00000000_B
000069_H		Upper output compare register (ch.1)			00000000_B
$00006A_H$	OCCP2	Lower output compare register (ch.2)	R/W		00000000_B
$00006B_H$		Upper output compare register (ch.2)			00000000_B
$00006C_H$	OCCP3	Lower output compare register (ch.3)	R/W		00000000_B
$00006D_H$		Upper output compare register (ch.3)			00000000_B
$00006E_H$	Use prohibition				
$00006F_H$	ROMM	ROM mirror function selection register	R/W	ROM mirror function	$-------1_B$
000070_H	OCCP4	Lower output compare register (ch.4)	R/W	ROM mirror function	00000000_B
000071_H		Upper output compare register (ch.4)			00000000_B
000072_H	OCCP5	Lower output compare register (ch.5)	R/W		00000000_B
000073_H		Upper output compare register (ch.5)			00000000_B
000074_H	OCS01	Lower output compare control register (ch.0, ch.1)	R/W		$0000--00_B$
000075_H		Upper output compare control register (ch.0, ch.1)	R/W		$---00000_B$
000076_H	OCS23	Lower output compare control register (ch.2, ch.3)	R/W		$0000--00_B$
000077_H		Upper output compare control register (ch.2, ch.3)	R/W		$---00000_B$
000078_H	OCS45	Lower output compare control register (ch.4, ch.5)	R/W		$0000--00_B$
000079_H		Upper output compare control register (ch.4, ch.5)	R/W		$---00000_B$
$00007A_H$	IPCP0	Lower input capture data register (ch.0)	R	16-bit output timer input capture (ch.0, ch.1)	XXXXXXXX_B
$00007B_H$		Upper input capture data register (ch.0)	R		XXXXXXXX_B
$00007C_H$	IPCP1	Lower input capture data register (ch.1)	R		XXXXXXXX_B
$00007D_H$		Upper input capture data register (ch.1)	R		XXXXXXXX_B
$00007E_H$	ICS01	Input capture control status register	R/W		00000000_B
$00007F_H$	ICE01	Input capture edge register	R		$------XX_B$

Table A-2. I/O Map (Sheet 5 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
000080_H	TCDT	Lower timer counter data register	R/W	16-bit output timer free-run timer	00000000_B
000081_H	TCDT	Upper timer counter data register	R/W		00000000_B
000082_H	TCCS	Timer control status register	R/W		00000000_B
000083_H	TCCS	Timer control status register	R/W		XX-00000_B
000084_H	CPCLR	Lower compare clear register	R/W		XXXXXXXX_B
000085_H		Upper compare clear register			XXXXXXXX_B
000086_H to $00009A_H$	Use prohibition				
$00009B_H$	DCSR	DMAC descriptor channel specification register	R/W	DMAC	00000000_B
$00009C_H$	DSRL	DMAC lower status register	R/W	DMAC	00000000_B
$00009D_H$	DSRH	DMAC upper status register	R/W	DMAC	00000000_B
$00009E_H$	PACSR0	Address detection control status register 0	R/W	Address match detection function	00000000_B
$00009F_H$	DIRR	Delayed interrupt source generation/ release register	R/W	Delayed interrupt generation module	-------0_B
$0000A0_H$	LPMCR	Low power consumption mode control register	W, R/W	Low power consumption	00011000_B
$0000A1_H$	CKSCR	Clock selection register	R, R/W		11111100_B
$0000A2_H$, $0000A3_H$	Use prohibition				
$0000A4_H$	DSSR	DMAC stop status register	R/W	DMAC	00000000_B
$0000A5_H$	ARSR	Auto ready function selection register	W	External pin	0011--00_B
$0000A6_H$	HACR	External address output control register	W		********$_B$
$0000A7_H$	EPCR	Bus control signal selection register	W		1000*10-$_B$
$0000A8_H$	WDTC	Watchdog timer control register	R, W	Watchdog timer	$XXXXX111_B$
$0000A9_H$	TBTC	Time-base timer control register	W, R/W	Time-base timer	$1XX00100_B$
$0000AA_H$	WTC	Watch timer control register	R, R/W	Watch timer	10001000_B
$0000AB_H$	Use prohibition				
$0000AC_H$	DERL	DMAC lower enable register	R/W	DMAC	00000000_B
$0000AD_H$	DERH	DMAC upper enable register	R/W		00000000_B
$0000AE_H$	FMCS	Flash memory control status register	W, R/W	Flash memory I/F	$000X0000_B$
$0000AF_H$	Use prohibition				
$0000B0_H$	ICR00	Interrupt control register 00	W, R/W	-	00000111_B
$0000B1_H$	ICR01	Interrupt control register 01	W, R/W	-	00000111_B
$0000B2_H$	ICR02	Interrupt control register 02	W, R/W	-	00000111_B
$0000B3_H$	ICR03	Interrupt control register 03	W, R/W	-	00000111_B
$0000B4_H$	ICR04	Interrupt control register 04	W, R/W	-	00000111_B
$0000B5_H$	ICR05	Interrupt control register 05	W, R/W	-	00000111_B
$0000B6_H$	ICR06	Interrupt control register 06	W, R/W	-	00000111_B
$0000B7_H$	ICR07	Interrupt control register 07	W, R/W	-	00000111_B
$0000B8_H$	ICR08	Interrupt control register 08	W, R/W	-	00000111_B
$0000B9_H$	ICR09	Interrupt control register 09	W, R/W	-	00000111_B
$0000BA_H$	ICR10	Interrupt control register 10	W, R/W	-	00000111_B
$0000BB_H$	ICR11	Interrupt control register 11	W, R/W	-	00000111_B

Table A-2. I/O Map (Sheet 6 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$0000BC_H$	ICR12	Interrupt control register 12	W, R/W	-	00000111_B
$0000BD_H$	ICR13	Interrupt control register 13	W, R/W	-	00000111_B
$0000BE_H$	ICR14	Interrupt control register 14	W, R/W	-	00000111_B
$0000BF_H$	ICR15	Interrupt control register 15	W, R/W	-	00000111_B
$0000C0_H$	CMR0	Chip select area MASK register 0	R/W	Chip select function	00001111_B
$0000C1_H$	CAR0	Chip select area register 0	R/W	-	11111111_B
$0000C2_H$	CMR1	Chip select area MASK register 1	R/W	-	00001111_B
$0000C3_H$	CAR1	Chip select area register 1	R/W	-	11111111_B
$0000C4_H$	CMR2	Chip select area MASK register 2	R/W	-	00001111_B
$0000C5_H$	CAR2	Chip select area register 2	R/W	-	11111111_B
$0000C6_H$	CMR3	Chip select area MASK register 3	R/W	-	00001111_B
$0000C7_H$	CAR3	Chip select area register 3	R/W	-	11111111_B
$0000C8_H$	CSCR	Chip select control register	R/W	-	$----000*_B$
$0000C9_H$	CALR	Chip select active level register	R/W	-	$----0000_B$
$0000CA_H$ to $0000CE_H$	Use prohibition				
$0000CF_H$	PLLOS	PLL output selection register	W	PLL	$------X0_B$
$0000D0_H$	BAPL	DMA buffer address pointer (low)	R/W	DMAC	XXXXXXXX_B
$0000D1_H$	BAPM	DMA buffer address pointer (middle)	R/W		XXXXXXXX_B
$0000D2_H$	BAPH	DMA buffer address pointer (high)	R/W		XXXXXXXX_B
$0000D3_H$	MACS	DMA control register	R/W		XXXXXXXX_B
$0000D4_H$	IOAL	DMAI/O register address pointer (low)	R/W		XXXXXXXX_B
$0000D5_H$	IOAH	DMAI/O register address pointer (high)	R/W		XXXXXXXX_B
$0000D6_H$	DCTL	DMA data counter (low)	R/W		XXXXXXXX_B
$0000D7_H$	DCTH	DMA data counter (high)	R/W		XXXXXXXX_B
$0000D8_H$ to $0000DF_H$	Use prohibition				
$0000E0_H$	ENIR0	Interrupt/DTP enable register 0	R/W	DTP / external interrupt	00000000_B
$0000E1_H$	EIRR0	Interrupt/DTP source register 0	R/W		XXXXXXXX_B
$0000E2_H$	ELVR0	Request level setting register 0	R/W		00000000_B
$0000E3_H$		Request level setting register 0	R/W		00000000_B
$0000E4_H$	ENIR1	Interrupt/DTP enable register 1	R/W	DTP / external interrupt	00000000_B
$0000E5_H$	EIRR1	Interrupt/DTP source register 1	R/W		XXXXXXXX_B
$0000E6_H$	ELVR1	Request level setting register 1	R/W		00000000_B
$0000E7_H$		Request level setting register 1	R/W		00000000_B
$0000E8_H$	ENIR2	Interrupt/DTP enable register 2	R/W	DTP / external interrupt	$XXXX0000_B$
$0000E9_H$	EIRR2	Interrupt/DTP source register 2	R/W		XXXXXXXX_B
$0000EA_H$	ELVR2	Request level setting register 2	R/W		00000000_B
$0000EB_H$		Request level setting register 2	R/W		00000000_B
$0000EC_H$ to $0000EF_H$	Use prohibition				

Table A-2. I/O Map (Sheet 7 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$0000F0_H$ to $0000FF_H$	External area				
000100_H to $\#_H$*	RAM area				
007900_H	PCNTL0	PPG0 lower control status register	R/W	16-bit PPG0	00000000_B
007901_H	PCNTH0	PPG0 upper control status register	R/W		00000001_B
007902_H	PCNTL1	PPG1 lower control status register	R/W	16-bit PPG1	00000000_B
007903_H	PCNTH1	PPG1 upper control status register	R/W		00000001_B
007904_H	PCNTL2	PPG2 lower control status register	R/W	16-bit PPG2	00000000_B
007905_H	PCNTH2	PPG2 upper control status register	R/W		00000001_B
007906_H	PCNTL3	PPG3 lower control status register	R/W	16-bit PPG3	00000000_B
007907_H	PCNTH3	PPG3 upper control status register	R/W		00000001_B
007908_H	PCNTL4	PPG4 lower control status register	R/W	16-bit PPG4	00000000_B
007909_H	PCNTH4	PPG4 upper control status register	R/W		00000001_B
$00790A_H$	PCNTL5	PPG5 lower control status register	R/W	16-bit PPG5	00000000_B
$00790B_H$	PCNTH5	PPG5 upper control status register	R/W		00000001_B
$00790C_H$	PCNTL6	PPG6 lower control status register	R/W	16-bit PPG6	00000000_B
$00790D_H$	PCNTH6	PPG6 upper control status register	R/W		00000001_B
$00790E_H$	PCNTL7	PPG7 lower control status register	R/W	16-bit PPG7	00000000_B
$00790F_H$	PCNTH7	PPG7 upper control status register	R/W		00000001_B
007910_H	PPGDIV	PPG0 output division setting register	R/W	16-bit PPG0	11111100_B
007911_H	Use prohibition				
007912_H	PDCRL0	PPG0 down counter register	R	16-bit PPG0	11111111_B
007913_H	PDCRH0				11111111_B
007914_H	PCSRL0	PPG0 period setting register	W		11111111_B
007915_H	PCSRH0				11111111_B
007916_H	PUDUTL0	PPG0 duty setting register	W		00000000_B
007917_H	PUDUTH0				00000000_B
007918_H	Use prohibition				
007919_H	Use prohibition				
$00791A_H$	PDCRL1	PPG1 down counter register	R	16-bit PPG1	11111111_B
$00791B_H$	PDCRH1				11111111_B
$00791C_H$	PCSRL1	PPG1 period setting register	W		11111111_B
$00791D_H$	PCSRH1				11111111_B
$00791E_H$	PUDUTL1	PPG1 duty setting register	W		00000000_B
$00791F_H$	PUDUTH1				00000000_B
007920_H	Use prohibition				
007921_H	Use prohibition				

Table A-2. I/O Map (Sheet 8 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
007922_H	PDCRL2	PPG2 down counter register	R	16-bit PPG2	11111111_B
007923_H	PDCRH2				11111111_B
007924_H	PCSRL2	PPG2 period setting register	W		11111111_B
007925_H	PCSRH2				11111111_B
007926_H	PUDUTL2	PPG2 duty setting register	W		00000000_B
007927_H	PUDUTH2				00000000_B
007928_H	Use prohibition				
007929_H	Use prohibition				
$00792A_H$	PDCRL3	PPG3 down counter register	R	16-bit PPG3	11111111_B
$00792B_H$	PDCRH3				11111111_B
$00792C_H$	PCSRL3	PPG3 period setting register	W		11111111_B
$00792D_H$	PCSRH3				11111111_B
$00792E_H$	PUDUTL3	PPG3 duty setting register	W		00000000_B
$00792F_H$	PUDUTH3				00000000_B
007930_H	Use prohibition				
007931_H	Use prohibition				
007932_H	PDCRL4	PPG4 down counter register	R	16-bit PPG4	11111111_B
007933_H	PDCRH4				11111111_B
007934_H	PCSRL4	PPG4 period setting register	W		11111111_B
007935_H	PCSRH4				11111111_B
007936_H	PUDUTL4	PPG4 duty setting register	W		00000000_B
007937_H	PUDUTH4				00000000_B
007938_H	Use prohibition				
007939_H	Use prohibition				
$00793A_H$	PDCRL5	PPG5 down counter register	R	16-bit PPG5	11111111_B
$00793B_H$	PDCRH5				11111111_B
$00793C_H$	PCSRL5	PPG5 period setting register	W		11111111_B
$00793D_H$	PCSRH5				11111111_B
$00793E_H$	PUDUTL5	PPG5 duty setting register	W		00000000_B
$00793F_H$	PUDUTH5				00000000_B
007940_H	Use prohibition				
007941_H	Use prohibition				
007942_H	PDCRL6	PPG6 down counter register	R	16-bit PPG6	11111111_B
007943_H	PDCRH6				11111111_B
007944_H	PCSRL6	PPG6 period setting register	W		11111111_B
007945_H	PCSRH6				11111111_B
007946_H	PUDUTL6	PPG6 duty setting register	W		00000000_B
007947_H	PUDUTH6				00000000_B
007948_H	Use prohibition				
007949_H	Use prohibition				

Table A-2. I/O Map (Sheet 9 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$00794A_H$	PDCRL7	PPG7 down counter register	R	16-bit PPG7	11111111_B
$00794B_H$	PDCRH7				11111111_B
$00794C_H$	PCSRL7	PPG7 period setting register	W		11111111_B
$00794D_H$	PCSRH7				11111111_B
$00794E_H$	PUDUTL7	PPG7 duty setting register	W		00000000_B
$00794F_H$	PUDUTH7				00000000_B
007950_H	Use prohibition				
007951_H	Use prohibition				
007952_H	TMCR0	Timer control register ch.0	R/W	Base timer ch.0	00000000_B
007953_H					00000000_B
007954_H	STC0	Status control register ch.0	R/W		00000000_B
007955_H	Use prohibition				
007956_H	TMR0	Timer register ch.0	R/W	Base timer ch.0	00000000_B/ XXXXXXXX_B
007957_H					00000000_B/ XXXXXXXX_B
007958_H	PCSR0/PRLL0	Period/L-width setting register ch.0	R/W		XXXXXXXX_B
007959_H					XXXXXXXX_B
$00795A_H$	PDUT0/PRLH0/ DTBF0	Duty/H-width/data buffer register ch.0	R/W		XXXXXXXX_B/ 00000000_B
$00795B_H$					XXXXXXXX_B/ 00000000_B
$00795C_H$	TMCR1	Timer control register ch.1	R/W	Base timer ch.1	00000000_B
$00795D_H$					00000000_B
$00795E_H$	STC1	Status control register ch.1	R/W		00000000_B
$00795F_H$	Use prohibition				
007960_H	TMR1	Timer register ch.1	R/W	Base timer ch.1	00000000_B/ XXXXXXXX_B
007961_H					00000000_B/ XXXXXXXX_B
007962_H	PCSR1/PRLL1	Period/L-width setting register ch.1	R/W		XXXXXXXX_B
007963_H					XXXXXXXX_B
007964_H	PDUT1/PRLH1/ DTBF1	Duty/H-width/data buffer register ch.1	R/W		XXXXXXXX_B/ 00000000_B
007965_H					XXXXXXXX_B/ 00000000_B
007966_H	TMCR2	Timer control register ch.2	R/W	Base timer ch.2	00000000_B
007967_H					00000000_B
007968_H	STC2	Status control register ch.2	R/W		00000000_B
007969_H	Use prohibition				

Table A-2. I/O Map (Sheet 10 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$00796A_H$	TMR2	Timer register ch.2	R/W	Base timer ch.2	00000000_B/ XXXXXXXX_B
$00796B_H$					00000000_B/ XXXXXXXX_B
$00796C_H$	PCSR2/PRLL2	Period/L-width setting register ch.2	R/W		XXXXXXXX_B
$00796D_H$					XXXXXXXX_B
$00796E_H$	PDUT2/PRLH2/ DTBF2	Duty/H-width/data buffer register ch.2	R/W		XXXXXXXX_B/ 00000000_B
$00796F_H$					XXXXXXXX_B/ 00000000_B
007970_H	TMCR3	Timer control register ch.3	R/W	Base timer ch.3	00000000_B
007971_H					00000000_B
007972_H	STC3	Status control register ch.3	R/W		00000000_B
007973_H	Use prohibition				
007974_H	TMR3	Timer register ch.3	R/W	Base timer ch.3	00000000_B/ XXXXXXXX_B
007975_H					00000000_B/ XXXXXXXX_B
007976_H	PCSR3/PRLL3	Period/L-width setting register ch.3	R/W		XXXXXXXX_B
007977_H					XXXXXXXX_B
007978_H	PDUT3/PRLH3/ DTBF3	Duty/H-width/data buffer register ch.3	R/W		XXXXXXXX_B/ 00000000_B
007979_H					XXXXXXXX_B/ 00000000_B
$00797A_H$	UDCR0	Up-down count register (ch.0)	R	8/16-bit up/down timer counter	00000000_B
$00797B_H$	UDCR1	Up-down count register (ch.1)	R		00000000_B
$00797C_H$	RCR0	Reload/compare register (ch.0)	W		00000000_B
$00797D_H$	RCR1	Reload/compare register (ch.1)	W		00000000_B
$00797E_H$	CCRL0	Lower counter control register (ch.0)	W, R/W		$XX00X000_B$
$00797F_H$	CCRH0	Upper counter control register (ch.0)	R/W		00000000_B
007980_H	CCRL1	Lower counter control register (ch.1)	W, R/W		$XX00X000_B$
007981_H	CCRH1	Upper counter control register (ch.1)	R/W		-0000000_B
007982_H	CSR0	Counter status register (ch.0)	R, R/W		00000000_B
007983_H	Use prohibition				
007984_H	CSR1	Counter status register (ch.1)	R, R/W		00000000_B
007985_H to $00798F_H$	Use prohibition				

Table A-2. I/O Map (Sheet 11 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
007990_H	SMR6	Serial bus mode register ch.6	R/W	Multi-function serial ch.6	$\$\$\$\$\$\$\$_B$
007991_H	SCR6/IBCR6	Serial bus control register / I^2C bus control register ch.6	R/W		$\$\$\$\$\$\$\$_B$
007992_H	ESCR6/IBSR6	Extended communication control register / I^2C bus status register ch.6	R/W		$\$\$\$\$\$\$\$_B$
007993_H	SSR6	Serial status register ch.6	R/W		$\$\$\$\$\$\$\$_B$
007994_H	RDR06/TDR06	Transmission/reception data register 0 ch.6	R,W		$\$\$\$\$\$\$\$_B$
007995_H	RDR16/TDR16	Transmission/reception data register 1 ch.6	R,W		$\$\$\$\$\$\$\$_B$
007996_H	BGR06	Baud rate generator register 0 ch.6	R/W		$\$\$\$\$\$\$\$_B$
007997_H	BGR16	Baud rate generator register 1 ch.6	R/W		$\$\$\$\$\$\$\$_B$
007998_H	ISBA6	7-bit slave address register ch.6	R/W		00000000_B
007999_H	ISMK6	7-bit slave address mask register ch.6	R/W		01111111_B
$00799A_H$	PAFSR	PPG pin assignment switching register	R/W	PPG pin switching control	$----0000_B$
$00799B_H$	PMSSR	PPG multi-channel start register	R/W	PPG multi-start control	00000000_B
$00799C_H$	Use prohibition				
$00799D_H$	P9FSR	Serial pin switching register 1	R/W	Multi-function serial pin control	$-----000_B$
$00799C_H$ to $0079A1_H$	Use prohibition				
$0079A2_H$	P7FSR	Serial pin switching register 0	R/W	Multi-function serial pin control	$----000X_B$
$0079A3_H$	LSYNS	LIN SYNCH FIELD switching register	R/W	Input capture input control	10001000_B
$0079A4_H$ to $0079A5_H$	Use prohibition				
$0079A6_H$	FWR0	Flash memory write control register 0	R/W	Flash memory I/F	00000000_B
$0079A7_H$	FWR1	Flash memory write control register 1	R/W	Flash memory I/F	00000000_B
$0079A8_H$ to $0079DF_H$	Use prohibition				
$0079E0_H$	PADR0	Detection address register 0 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079E1_H$		Detection address register 0 (middle)			XXXXXXXX_B
$0079E2_H$		Detection address register 0 (upper)			XXXXXXXX_B
$0079E3_H$	PADR1	Detection address register 1 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079E4_H$		Detection address register 1 (middle)			XXXXXXXX_B
$0079E5_H$		Detection address register 1 (upper)			XXXXXXXX_B
$0079E6_H$	PADR2	Detection address register 2 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079E7_H$		Detection address register 2 (middle)			XXXXXXXX_B
$0079E8_H$		Detection address register 2 (upper)			XXXXXXXX_B
$0079E9_H$ to $0079EF_H$	Use prohibition				
$0079F0_H$	PADR3	Detection address register 3 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079F1_H$		Detection address register 3 (middle)			XXXXXXXX_B
$0079F2_H$		Detection address register 3 (upper)			XXXXXXXX_B

Table A-2. I/O Map (Sheet 12 of 12)

Address	Register abbreviation	Register name	R/W	Resource	Initial value
$0079F3_H$	PADR4	Detection address register 4 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079F4_H$		Detection address register 4 (middle)			XXXXXXXX_B
$0079F5_H$		Detection address register 4 (upper)			XXXXXXXX_B
$0079F6_H$	PADR5	Detection address register 5 (lower)	R/W	Address match detection function	XXXXXXXX_B
$0079F7_H$		Detection address register 5 (middle)			XXXXXXXX_B
$0079F8_H$		Detection address register 5 (upper)			XXXXXXXX_B
$0079F9_H$ to $007FFF_H$	Use prohibition				

Explanation on R/W

R/W : Readable/Writable

R : Read only

W : Write only

Explanation on initial value

0 : The initial value of this bit is “0”.

1 : The initial value of this bit is “1”.

X : The initial value of this bit is undefined.

- : This bit is not used.

* : The initial value of this bit is “1” or “0”.
It varies depending on the mode pin (MD2, MD1 or MD0 pin) .

+ : The initial value of this bit is “1” or “0”.

$: The initial value of this bit varies depending on the operation mode of the resource.

#H* : Varies depending on the RAM area of the device.

A.3 Interrupt Source, Interrupt Vector, and Interrupt Control Register

Table A-3 shows the relationship between interrupt source and the interrupt vector, and interrupt control register.

Interrupt Source, Interrupt Vector, and Interrupt Control Register

Table A-3. Relationship between Interrupt Source and Interrupt Vector, and Interrupt Control Register (Sheet 1 of 2)

Interrupt source	Clearing of EI^{2}OS	µDMAC channel no.	Interrupt vector		Interrupt control register	
			No.	Address	No.	Address
Reset	×	—	#08	FFFFDC$_H$	—	—
INT9 instruction	×	—	#09	FFFFD8$_H$	—	—
Exception	×	—	#10	FFFFD4$_H$	—	—
INT0 (IRQ0/IRQ1)	◯	0	#11	FFFFD0$_H$	ICR00	0000B0$_H$
INT1 (IRQ2 to IRQ7)	◯	—	#12	FFFFCC$_H$		
INT2 (IRQ8 to IRQ15)	◯	—	#13	FFFFC8$_H$	ICR01	0000B1$_H$
INT3 (IRQ16 to IRQ23)	◯	—	#14	FFFFC4$_H$		
Base timer ch.0 (source 0,1)	◯	1	#15	FFFFC0$_H$	ICR02	0000B2$_H$
Base timer ch.1 (source 0,1)	◯	2	#16	FFFFBC$_H$		
Base timer ch.2 (source 0,1)	◯	3	#17	FFFFB8$_H$	ICR03	0000B3$_H$
Base timer ch.3 (source 0,1)	◯	4	#18	FFFFB4$_H$		
PPG0/PPG4 counter borrow	◯	5	#19	FFFFB0$_H$	ICR04	0000B4$_H$
PPG1/PPG5 counter borrow	◯	6	#20	FFFFAC$_H$		
PPG2/PPG6 counter borrow	◯	7	#21	FFFFA8$_H$	ICR05	0000B5$_H$
PPG3/PPG7 counter borrow	×	8	#22	FFFFA4$_H$		
8/16-bit up/down counter/timer (ch.0/ch.1) compare / underflow / overflow / up-down inversion	×	—	#23	FFFFA0$_H$	ICR06	0000B6$_H$
Input capture fetch (ch.0/ch.1)	◯	—	#24	FFFF9C$_H$		
Output compare (ch.0/ch.1/ch.2) match	◯	—	#25	FFFF98$_H$	ICR07	0000B7$_H$
Output compare (ch.3/ch.4/ch.5) match	◯	—	#26	FFFF94$_H$		
A/D converter	◯	—	#27	FFFF90$_H$	ICR08	0000B8$_H$
Overflow in 16-bit free-run timer / compare clear / multi-function serial ch.4/5/6 status	◯	9	#28	FFFF8C$_H$		
Multi-function serial ch.4 reception	◯	10	#29	FFFF88$_H$	ICR09	0000B9$_H$
Multi-function serial ch.4 transition	◯	11	#30	FFFF84$_H$		
Multi-function serial ch.5 reception	◯	12	#31	FFFF80$_H$	ICR10	0000BA$_H$
Multi-function serial ch.5 transition	◯	13	#32	FFFF7C$_H$		
Multi-function serial ch.6 reception	◯	14	#33	FFFF78$_H$	ICR11	0000BB$_H$
Multi-function serial ch.6 transition	◯	15	#34	FFFF74$_H$		
Multi-function serial ch.0/ch.1 reception / status	◎	—	#35	FFFF70$_H$	ICR12	0000BC$_H$
Multi-function serial ch.0/ch.1 transmission	◯	—	#36	FFFF6C$_H$		

Table A-3. Relationship between Interrupt Source and Interrupt Vector, and Interrupt Control Register (Sheet 2 of 2)

Interrupt source	Clearing of EI^{2}OS	µDMAC channel no.	Interrupt vector		Interrupt control register	
			No.	Address	No.	Address
Multi-function serial ch.2 reception / status	◎	—	#37	FFFF68$_H$	ICR13	0000BD$_H$
Multi-function serial ch.2 transmission	○	—	#38	FFFF64$_H$		
Multi-function serial ch.3 reception / status	◎	—	#39	FFFF60$_H$	ICR14	0000BE$_H$
Multi-function serial ch.3 transmission	○	—	#40	FFFF5C$_H$		
Flash writing/erase, time-base timer, watch timer*	×	—	#41	FFFF58$_H$	ICR15	0000BF$_H$
Delayed interrupt generation module	×	—	#42	FFFF54$_H$		

x : The interrupt request flag is not cleared by the interrupt clear signal.

○ : The interrupt request flag is cleared by the interrupt clear signal.

◎ : The interrupt request flag is cleared by the interrupt clear signal. Stop request function provided at receiving only.

* : Flash writing/erase, the time-base timer and watch timer cannot be used simultaneously.

Note:

If a resource has two interrupt sources for the same interrupt number, both of the interrupt request flags are cleared by the EI^{2}OS/µDMAC interrupt clear signal. Therefore, when either of the two sources for the EI^{2}OS/µDMAC function is used, the other interrupt function cannot be used. In this case, set the interrupt request enable bit in the appropriate resource to “0” and take measures by software polling.

A.4 Instructions

Appendix A.4 describes the instructions used by the F^{2}MC-16LX.

A.4.1 Instruction Types
A.4.2 Addressing
A.4.3 Direct Addressing
A.4.4 Indirect Addressing
A.4.5 Execution Cycle Count
A.4.6 Effective address field
A.4.7 How to Read the Instruction List
A.4.8 F^{2}MC-16LX Instruction List
A.4.9 Instruction Map

A.4.1 Instruction Types

The F^{2}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^{2}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

A.4.2 Addressing

With the F^{2}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^{2}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 A-4 lists the address formats specified by the effective address field.

Table A-4. 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

A.4.3 Direct Addressing

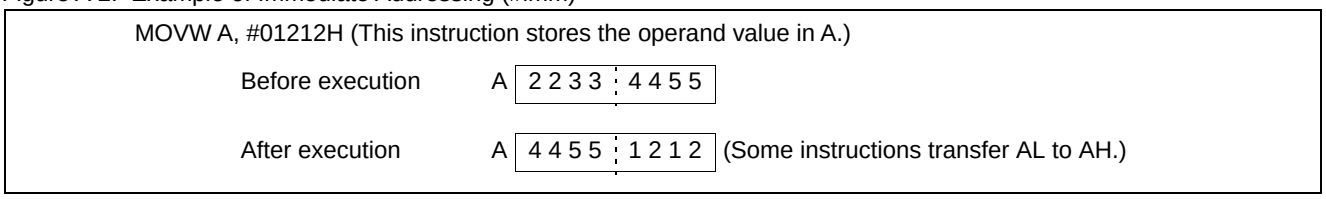
An operand value, register, or address is specified explicitly in direct addressing mode.

Direct Addressing

Immediate addressing (#imm)

Specify an operand value explicitly (#imm4/ #imm8/ #imm16/ #imm32).

Figure A-2. Example of Immediate Addressing (#imm)



Register direct addressing

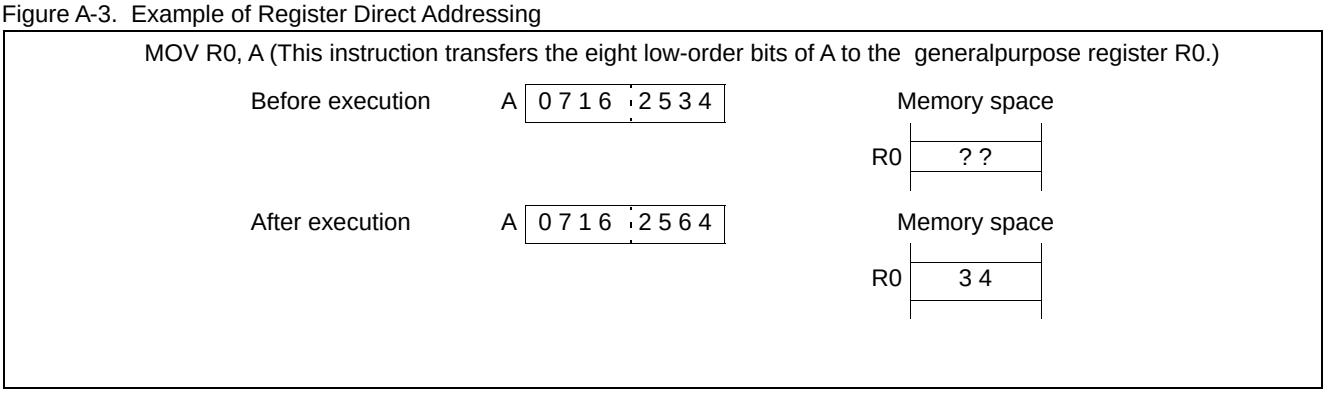
Specify a register explicitly as an operand. Table A-5 lists the registers that can be specified. Figure A-3 shows an example of register direct addressing.

Table A-5. 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.

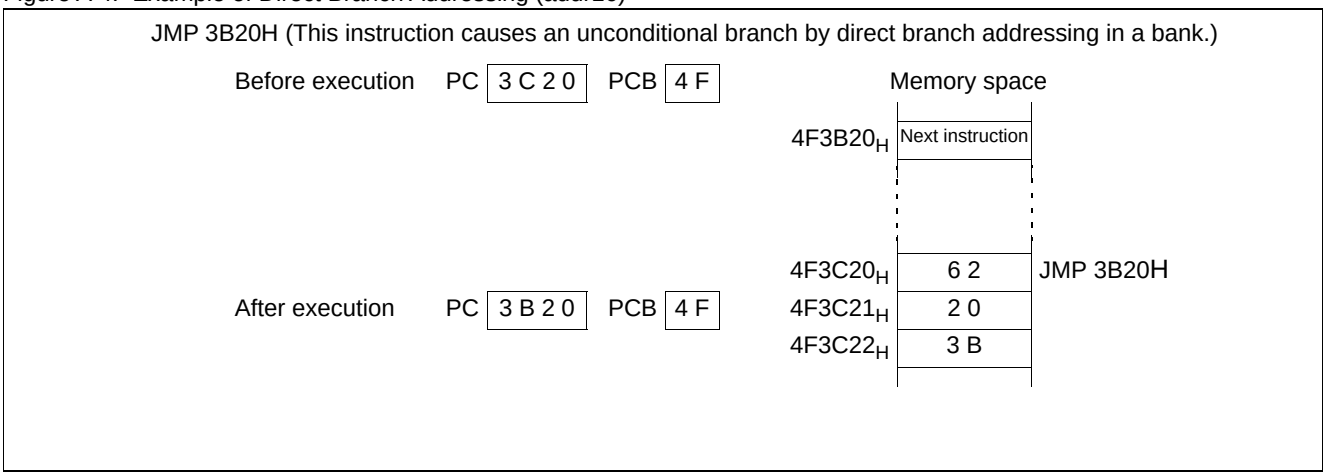
Figure A-3. Example of Register Direct Addressing



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).

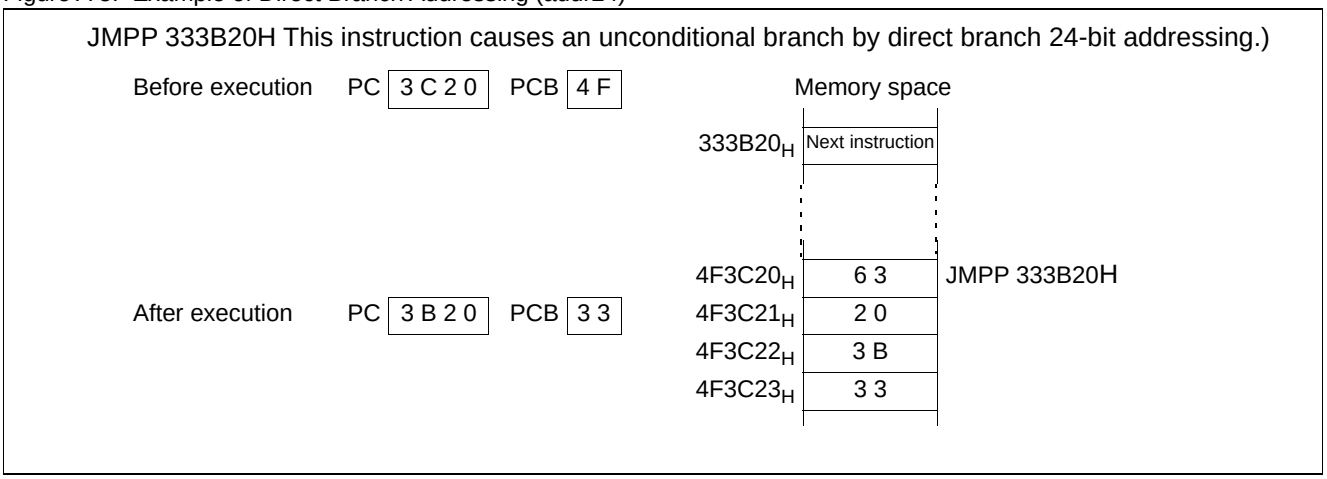
Figure A-4. Example of Direct Branch Addressing (addr16)



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.

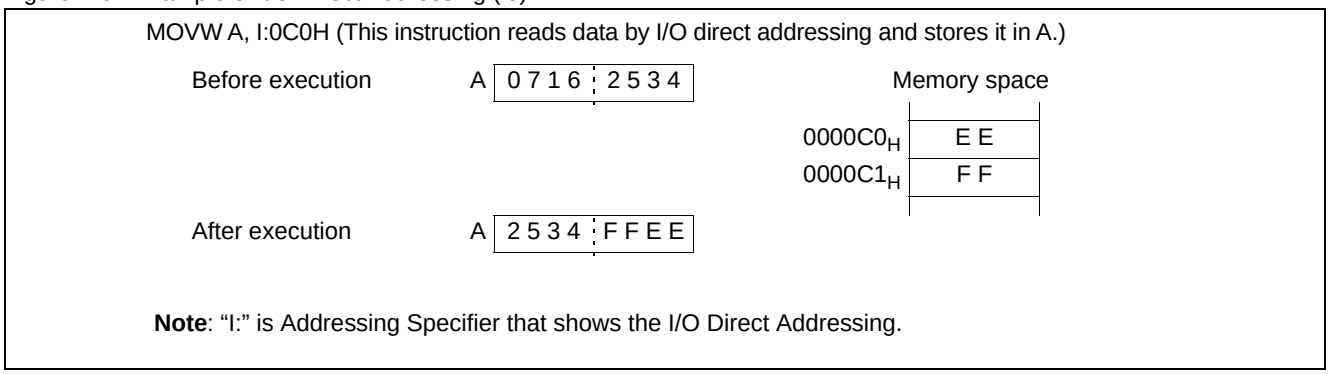
Figure A-5. Example of Direct Branch Addressing (addr24)



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_H to $0000FF_H$ 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.

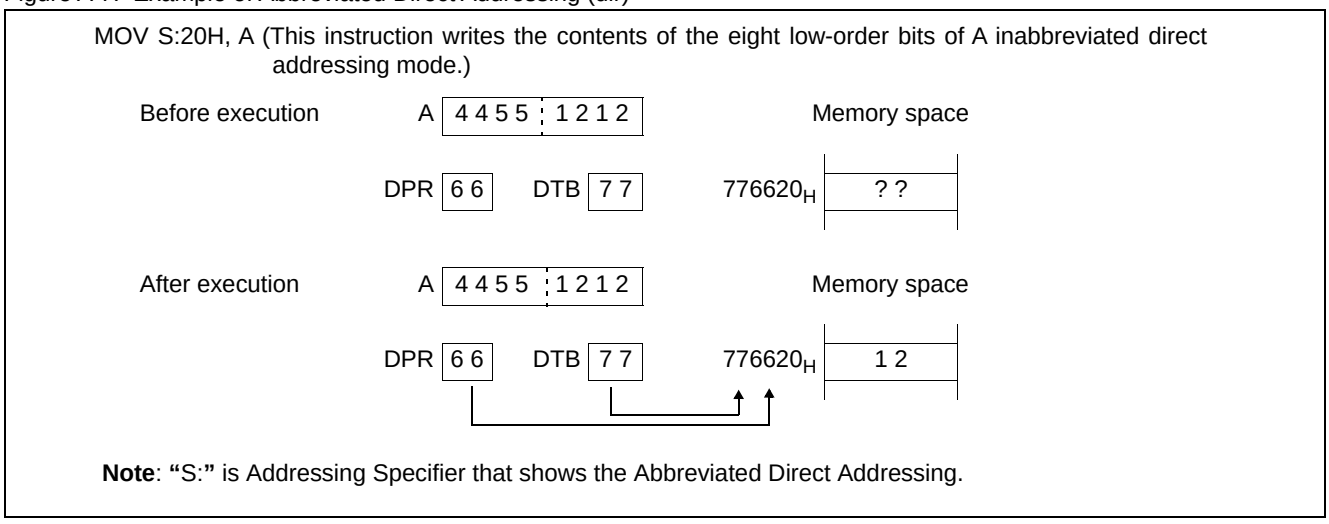
Figure A-6. Example of I/O Direct Addressing (io)



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).

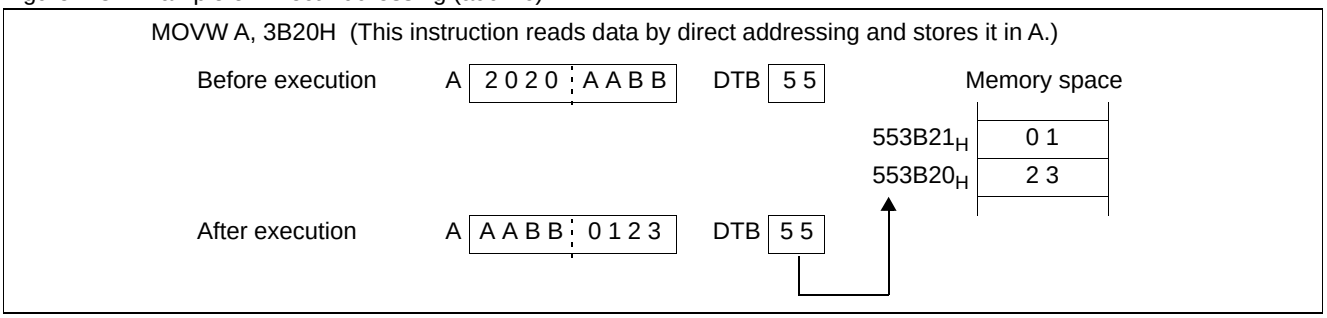
Figure A-7. Example of Abbreviated Direct Addressing (dir)



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.

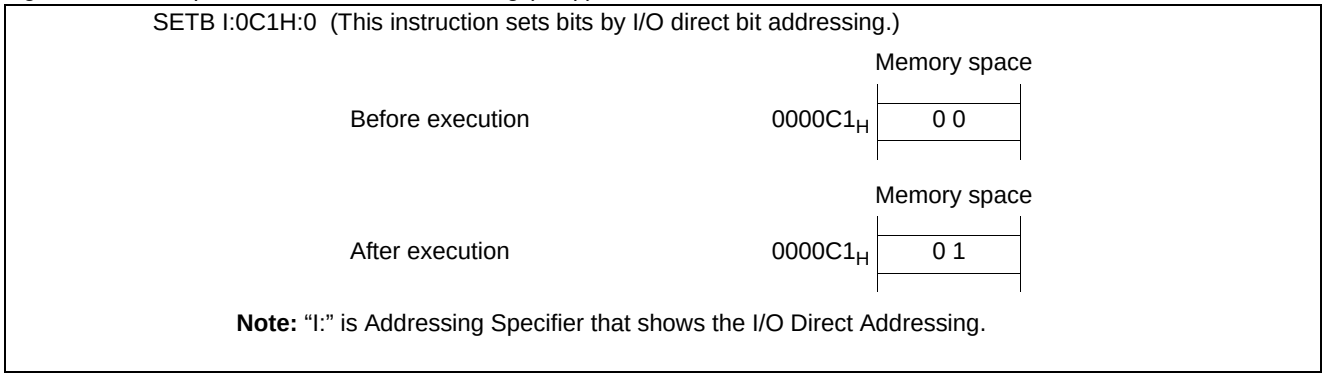
Figure A-8. Example of Direct Addressing (addr16)



I/O direct bit addressing (io:bp)

Specify bits in physical addresses 000000_H to $0000FF_H$ 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 A-9. Example of I/O Direct Bit Addressing (io:bp)

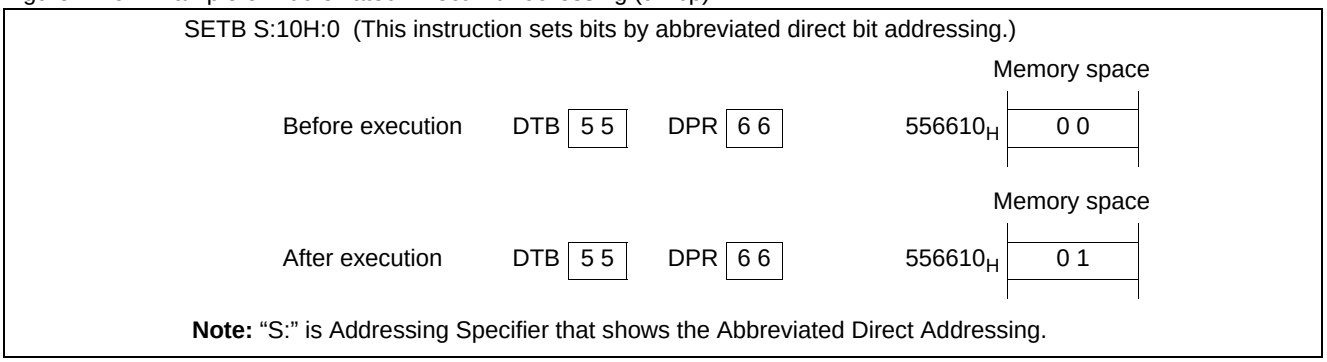


Note: "I:" is Addressing Specifier that shows the I/O Direct Addressing.

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 A-10. Example of Abbreviated Direct Bit Addressing (dir:bp)

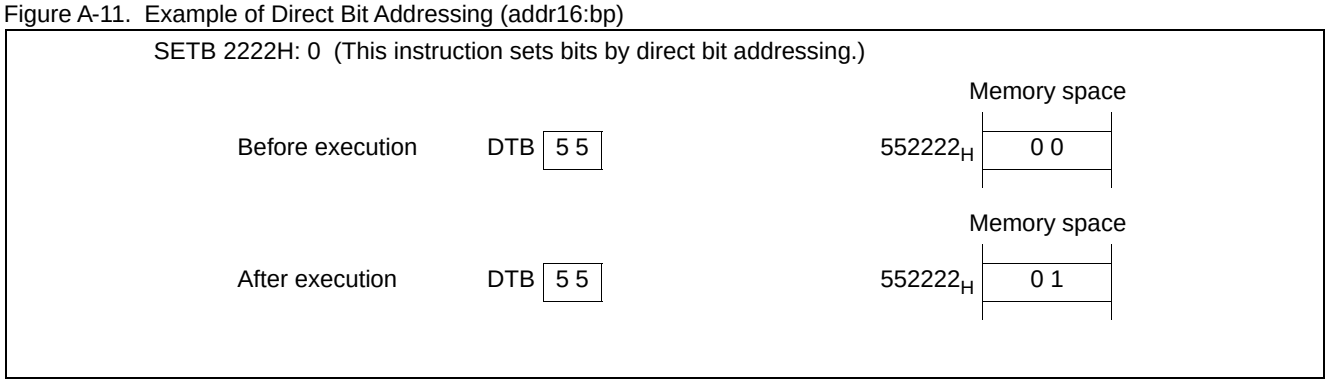


Note: "S:" is Addressing Specifier that shows the Abbreviated Direct Addressing.

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 A-11. 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 A-12. Example of Vector Addressing (#vct)

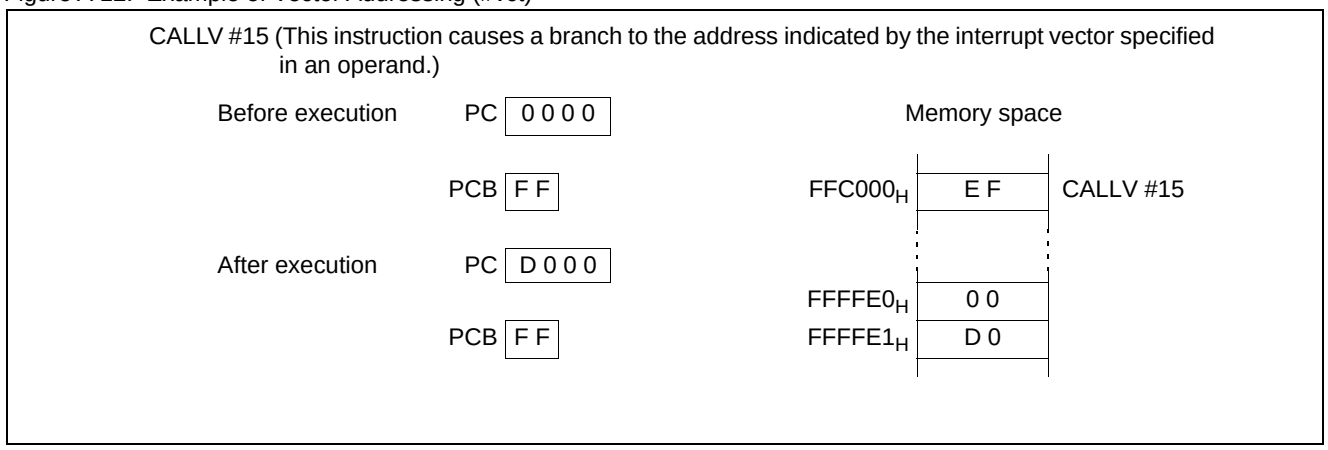


Table A-6. CALLV Vector List

Instruction	Vector address L	Vector address H
CALLV #0	$XXFFFE_H$	$XXFFFF_H$
CALLV #1	$XXFFFC_H$	$XXFFFD_H$
CALLV #2	$XXFFFA_H$	$XXFFFB_H$
CALLV #3	$XXFFF8_H$	$XXFFF9_H$
CALLV #4	$XXFFF6_H$	$XXFFF7_H$
CALLV #5	$XXFFF4_H$	$XXFFF5_H$
CALLV #6	$XXFFF2_H$	$XXFFF3_H$
CALLV #7	$XXFFF0_H$	$XXFFF1_H$
CALLV #8	$XXFFEE_H$	$XXFFEF_H$
CALLV #9	$XXFFEC_H$	$XXFFED_H$
CALLV #10	$XXFFEA_H$	$XXFFEB_H$
CALLV #11	$XXFFE8_H$	$XXFFE9_H$
CALLV #12	$XXFFE6_H$	$XXFFE7_H$
CALLV #13	$XXFFE4_H$	$XXFFE5_H$
CALLV #14	$XXFFE2_H$	$XXFFE3_H$
CALLV #15	$XXFFE0_H$	$XXFFE1_H$

Note: A PCB register value is set in XX.

Note:

When the program counter bank register (PCB) is FF_H, the vector area overlaps the vector area of INT #vct8 (#0 to #7). Use vector addressing carefully (see Table A-6).

A.4.4 Indirect Addressing

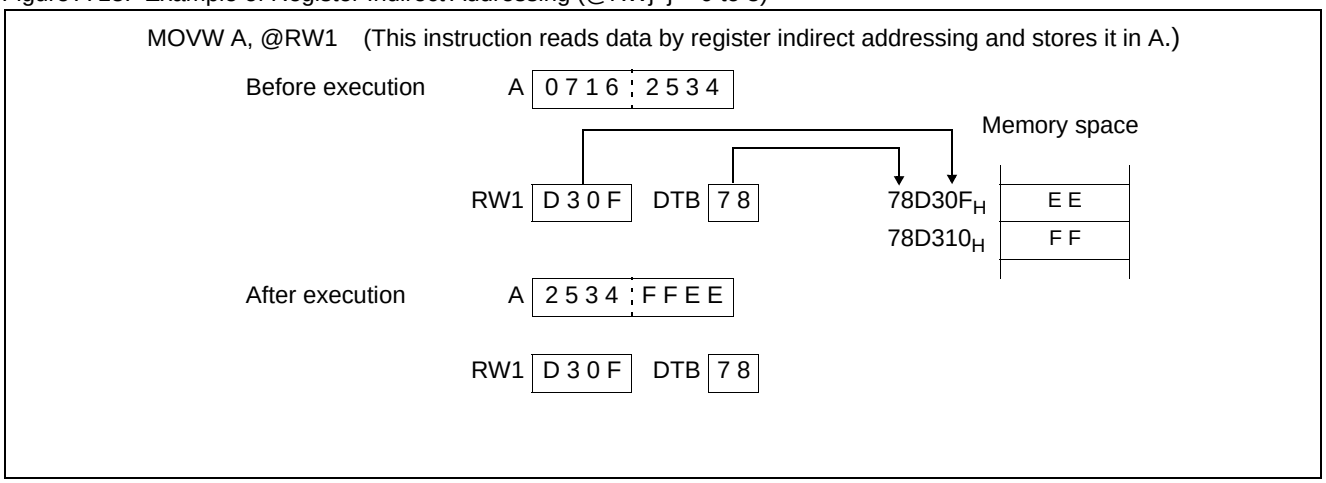
In indirect addressing mode, an address is specified indirectly by the address data of an operand.

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 A-13. Example of Register Indirect Addressing (@RWj j = 0 to 3)

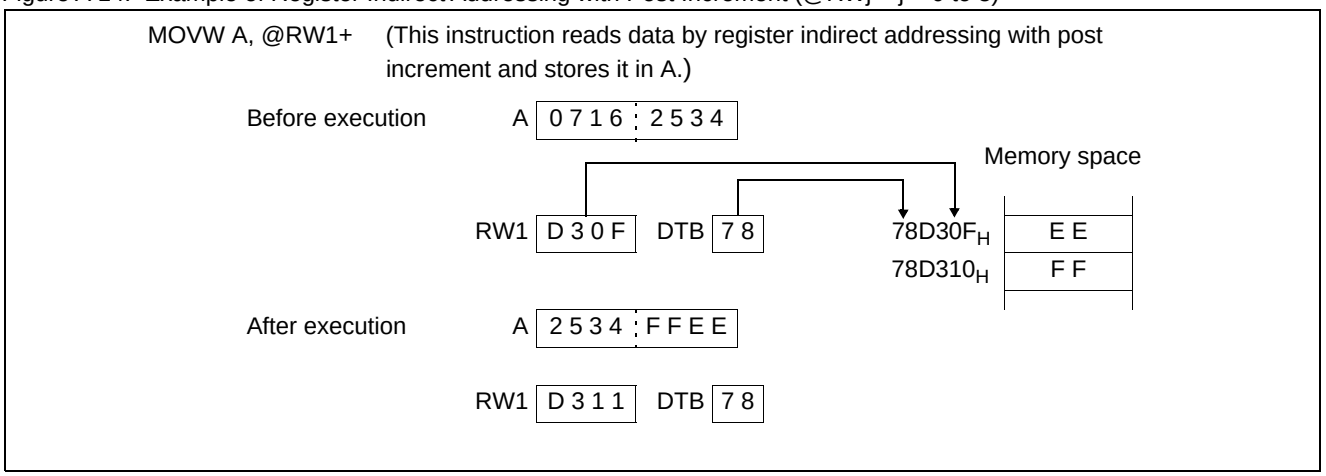


Register indirect addressing with post increment (@RWj+ j = 0 to 3)

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.

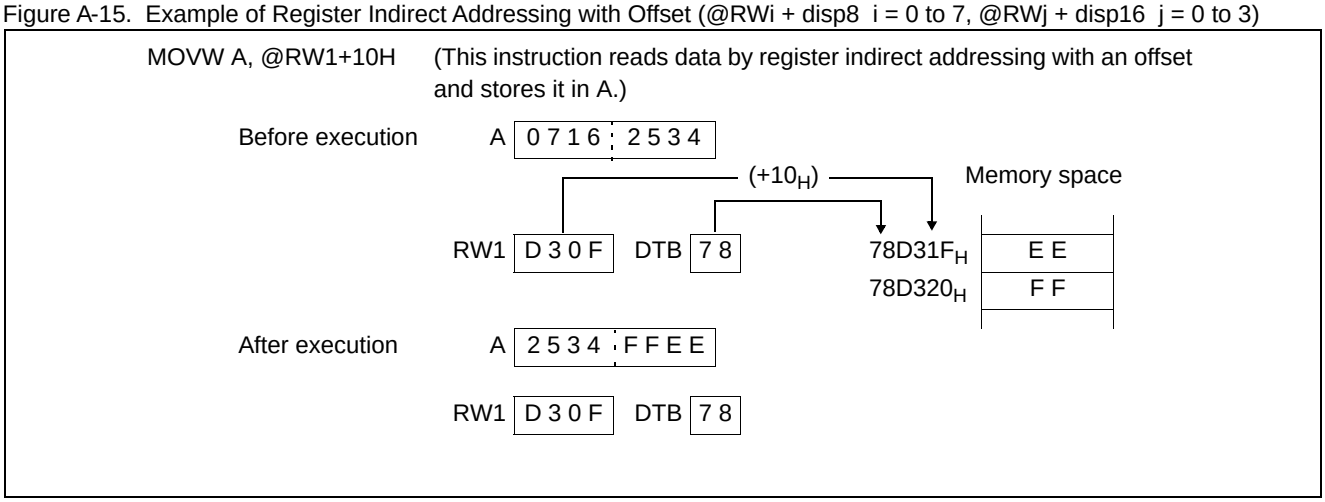
Figure A-14. Example of Register Indirect Addressing with Post Increment (@RWj+ j = 0 to 3)



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.

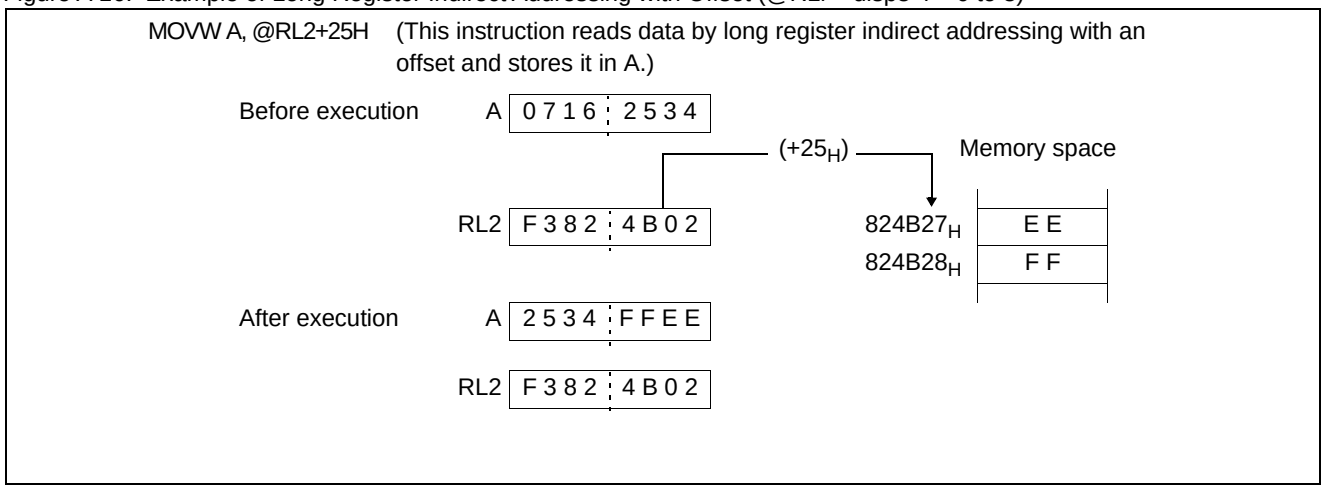
Figure A-15. Example of Register Indirect Addressing with Offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)



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 A-16. Example of Long Register Indirect Addressing with Offset (@RLi + disp8 i = 0 to 3)

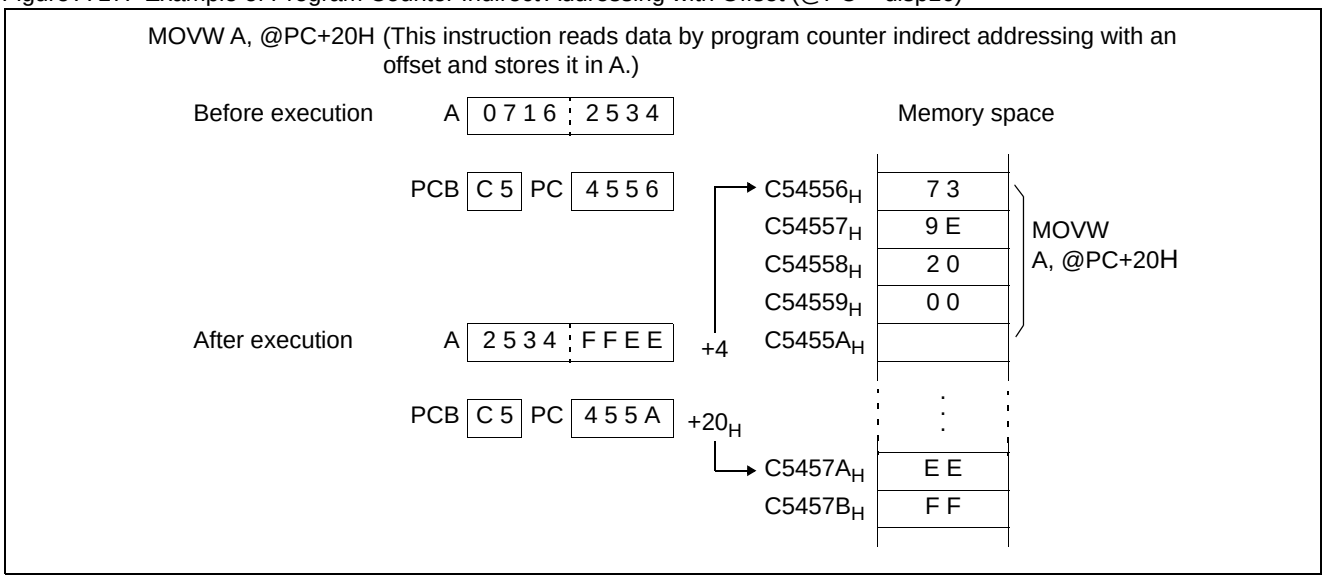


Program counter indirect addressing with offset (@PC + disp16)

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

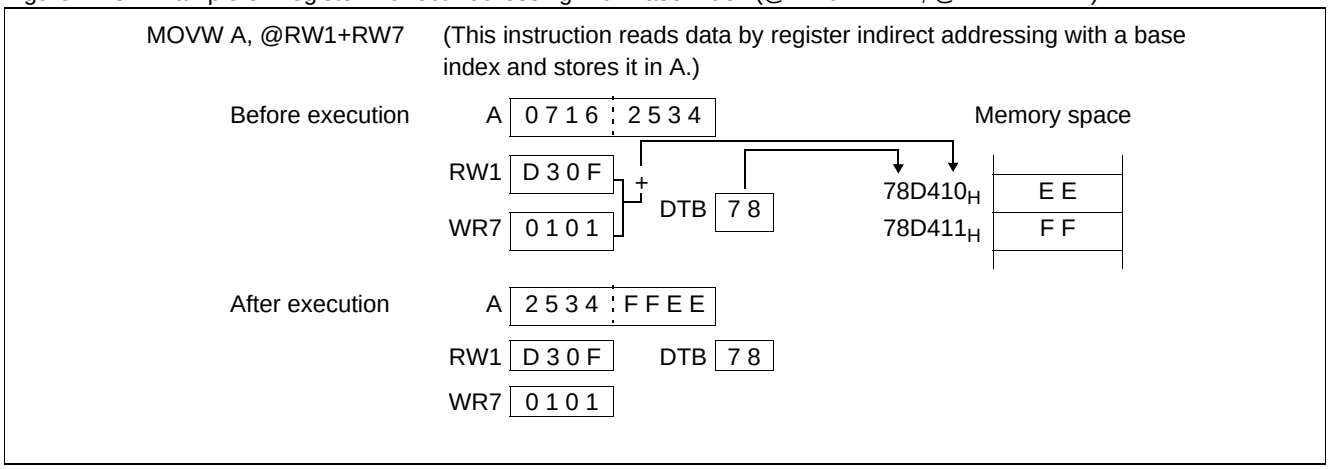
Figure A-17. Example of Program Counter Indirect Addressing with Offset (@PC + disp16)



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).

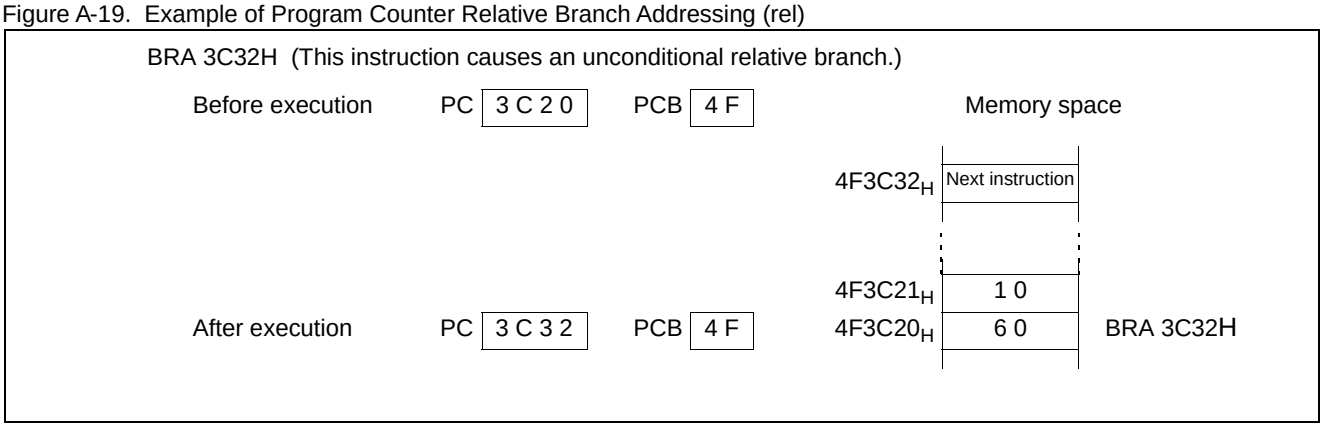
Figure A-18. Example of Register Indirect Addressing with Base Index (@RW0 + RW7, @RW1 + RW7)



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 A-19. Example of Program Counter Relative Branch Addressing (rel)



Register list (rlst)

Specify a register to be pushed onto or popped from a stack.

Figure A-20. Configuration of the Register List

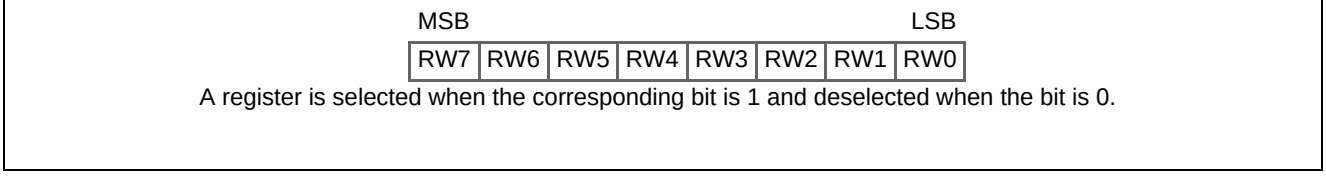
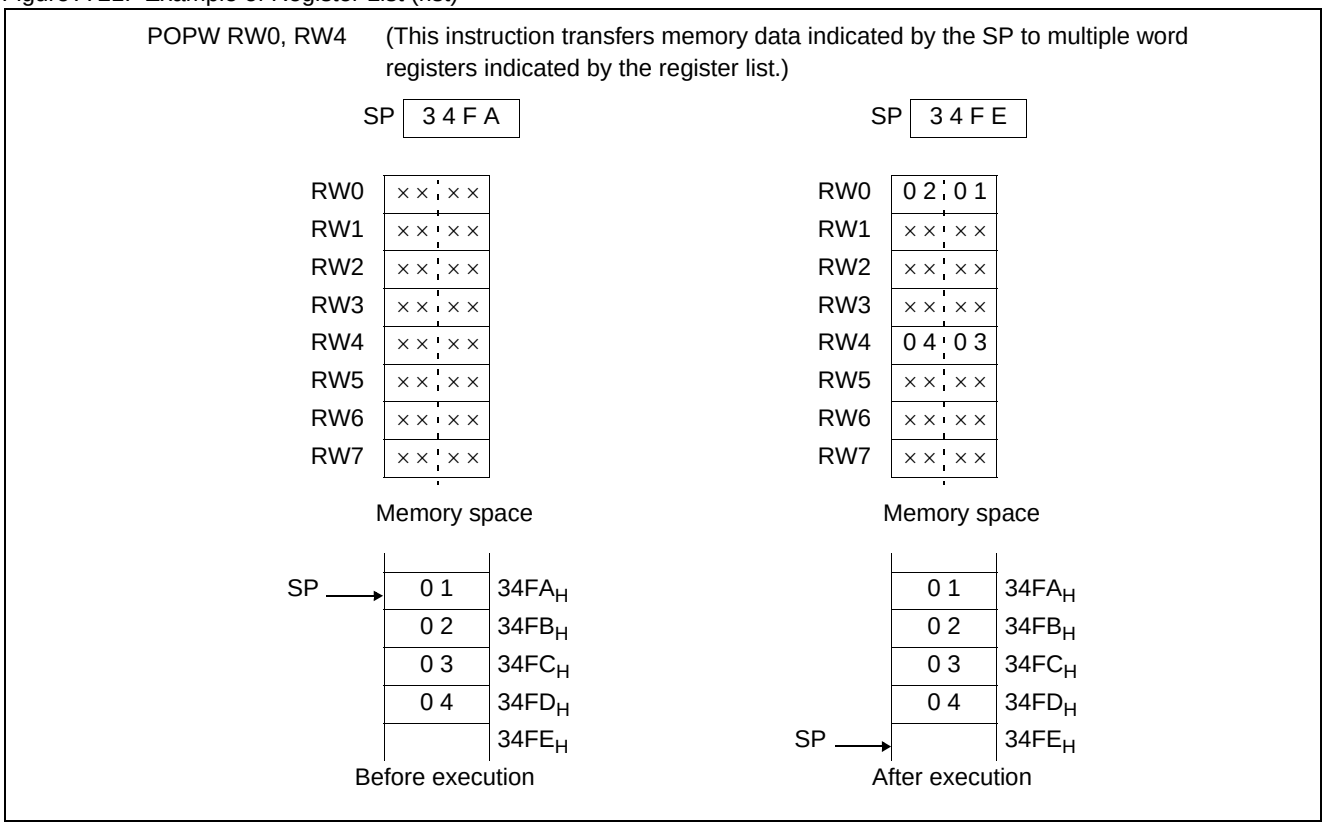


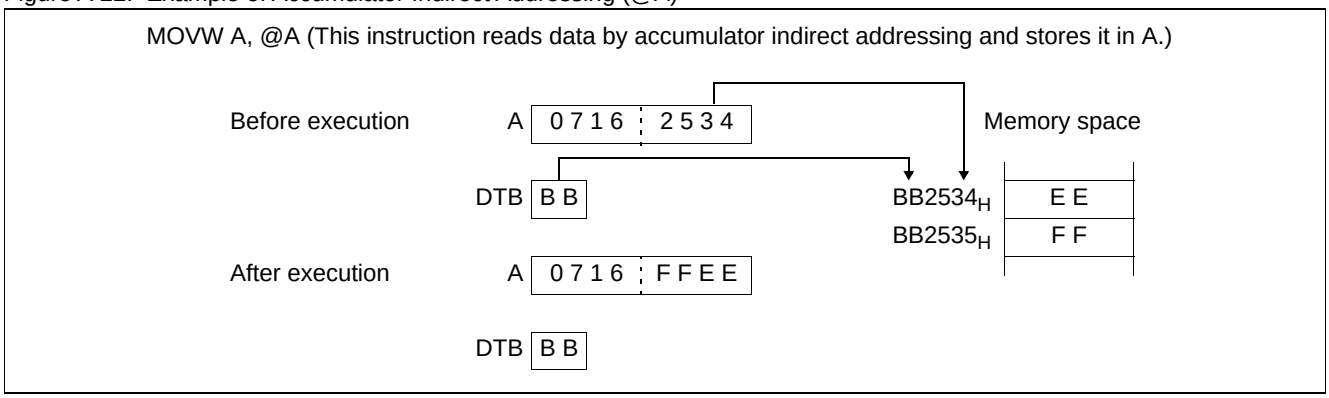
Figure A-21. Example of Register List (rlst)



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).

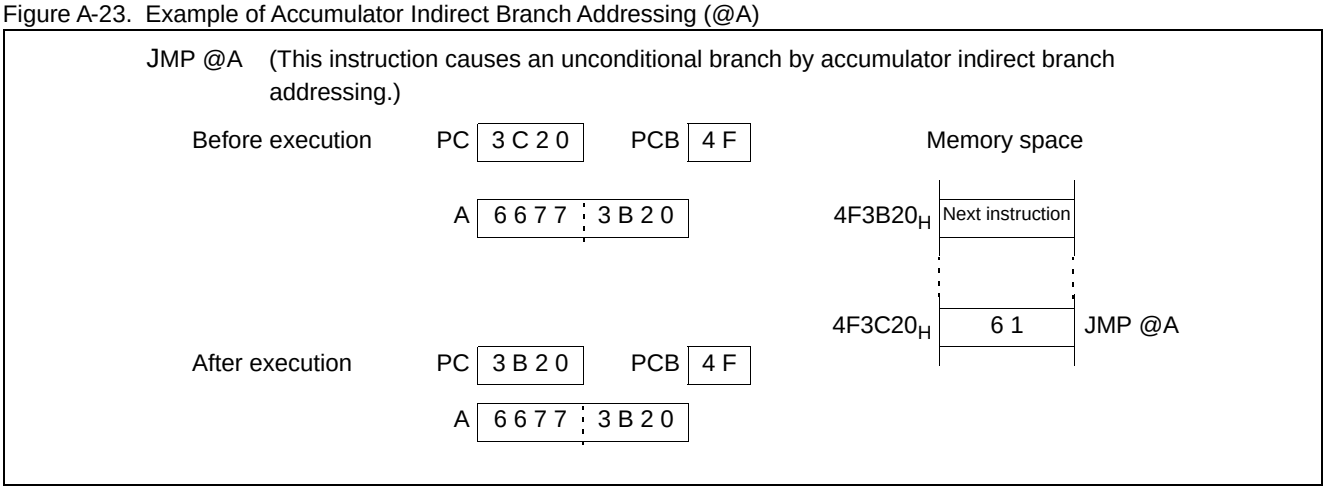
Figure A-22. Example of Accumulator Indirect Addressing (@A)



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.

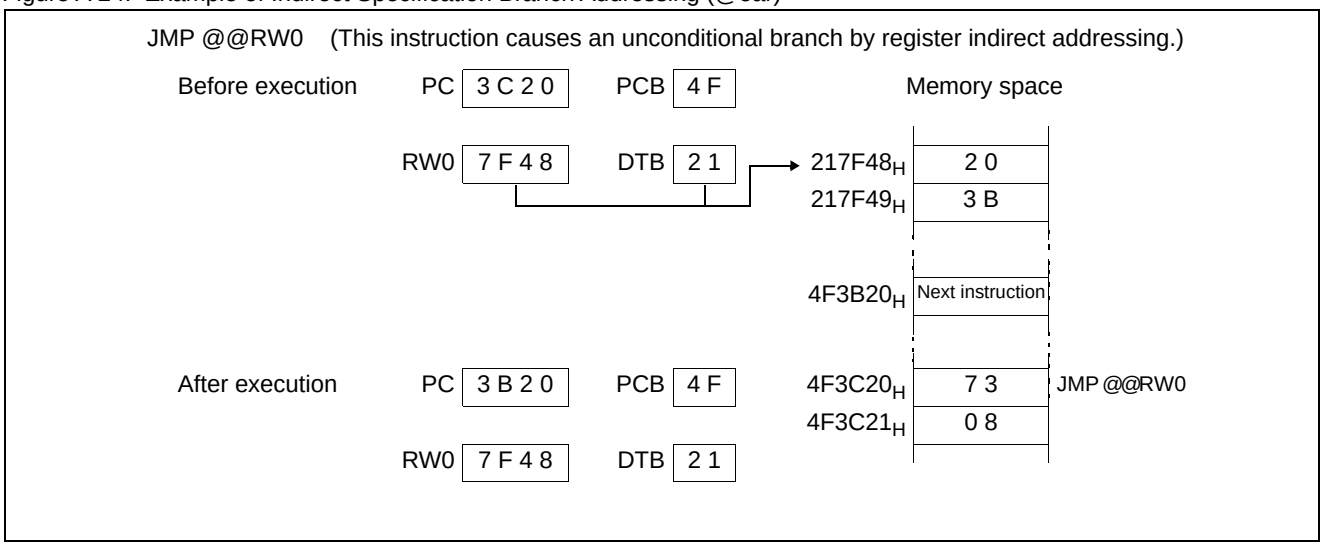
Figure A-23. Example of Accumulator Indirect Branch Addressing (@A)



Indirect specification branch addressing (@ear)

The address of the branch destination is the word data at the address indicated by ear.

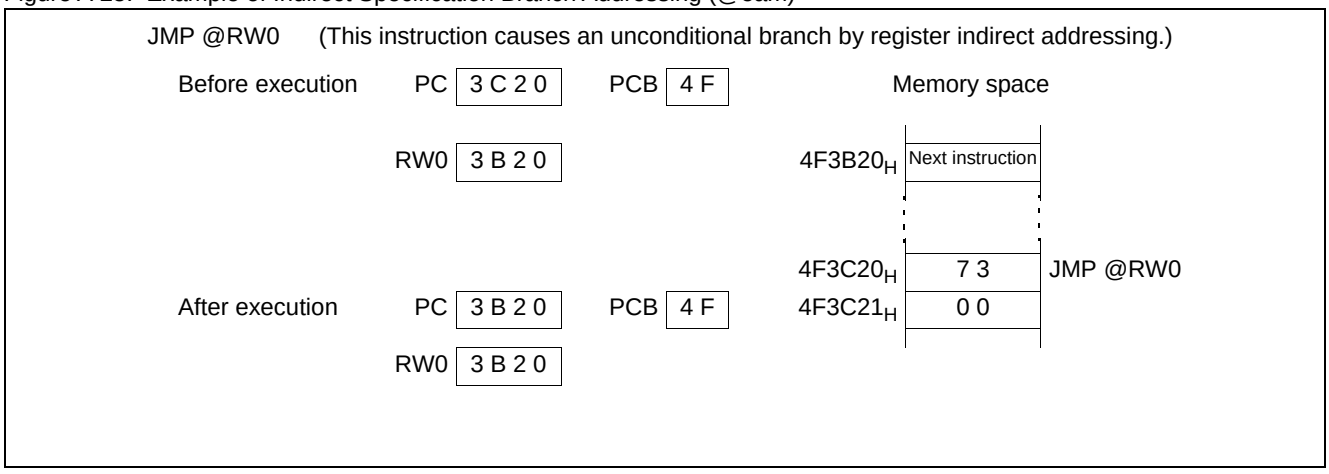
Figure A-24. Example of Indirect Specification Branch Addressing (@ear)



Indirect specification branch addressing (@eam)

The address of the branch destination is the word data at the address indicated by eam.

Figure A-25. Example of Indirect Specification Branch Addressing (@eam)



A.4.5 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 A-7 lists execution cycle counts and Table A-8. and Table A-9. summarize correction value data.

Table A-7. 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 "A.4.8 F^{2}MC-16LX Instruction List".

Table A-8. 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 "A.4.8 F^{2}MC-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 A-9. 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.

A.4.6 Effective address field

Table A-10. Table A-11. shows the effective address field.

Effective Address Field

Table A-11. 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 "A.4.8 F^{2}MC-16LX Instruction List".

A.4.7 How to Read the Instruction List

Table A-12 describes the items used in "A.4.8 F2MC-16LX Instruction List", and Table A-13 describes the symbols used in the same list.

Description of Instruction Presentation Items and Symbols

Table A-12. Description of Items in the Instruction List

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 A-4 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_H or FF_H to AH after AL sign extension.
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 Note: Cannot be used for an address that has different meanings between read and write operations.

Table A-13. Explanation on Symbols in the Instruction List

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
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_H to $0000FF_H$)
#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_H to 07_H)
eam	Effective addressing (code 08_H to $1F_H$)
rlst	Register list

A.4.8 F²MC-16LX Instruction List

Table A-14 to Table A-31 list the instructions used by the F²MC-16LX.

F²MC-16LX Instruction List

Table A-14. 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 A-7 and Table A-8 for information on (a) and (b) in the table.

Table A-15. 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	SP,A	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 A-7 and Table A-8 for information on (a), (c), and (d) in the table.

Table A-16. 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	-	-	-	-	*	*	*	*	-
ADDDC	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	-	-	-	-	*	*	*	*	-
SUBDC	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 A-7 and Table A-8 for information on (a) to (d) in the table.

Table A-17. 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) ← (ear) + 1	-	-	-	-	-	*	*	*	-	-
INC	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) + 1	-	-	-	-	-	*	*	*	-	*
DEC	ear	2	3	2	0	byte (ear) ← (ear) - 1	-	-	-	-	-	*	*	*	-	-
DEC	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) - 1	-	-	-	-	-	*	*	*	-	*
INCW	ear	2	3	2	0	word (ear) ← (ear) + 1	-	-	-	-	-	*	*	*	-	-
INCW	eam	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) + 1	-	-	-	-	-	*	*	*	-	*
DECW	ear	2	3	2	0	word (ear) ← (ear) - 1	-	-	-	-	-	*	*	*	-	-
DECW	eam	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) - 1	-	-	-	-	-	*	*	*	-	*
INCL	ear	2	7	4	0	long (ear) ← (ear) + 1	-	-	-	-	-	*	*	*	-	-
INCL	eam	2+	9+(a)	0	2 × (d)	long (eam) ← (eam) + 1	-	-	-	-	-	*	*	*	-	*
DECL	ear	2	7	4	0	long (ear) ← (ear) - 1	-	-	-	-	-	*	*	*	-	-
DECL	eam	2+	9+(a)	0	2 × (d)	long (eam) ← (eam) - 1	-	-	-	-	-	*	*	*	-	*

Note:

See Table A-7 and Table A-8 for information on (a) to (d) in the table.

Table A-18. 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 A-7 and Table A-8 for information on (a) to (d) in the table.

Table A-19. 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 A-7 and Table A-8 for information on (a) to (c) in the table.

Table A-20. 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 A-7 and Table A-8 for information on (a) to (c) in the table.

Table A-21. 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 A-7 and Table A-8 for information on (a) to (c) in the table.

Table A-22. 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 A-7 and Table A-8 for information on (a) and (d) in the table.

Table A-23. 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 A-7 and Table A-8 for information on (a) to (c) in the table.

Table A-24. 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 A-25. 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 A-7 and Table A-8 for information on (a) and (b) in the table.

Table A-26. 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/BNE	rel	2	*1	0	0	Branch on (Z) = 0	-	-	-	-	-	-	-	-	-	-
BC/BLO	rel	2	*1	0	0	Branch on (C) = 1	-	-	-	-	-	-	-	-	-	-
BNC/BHS	rel	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 × (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 A-7 and Table A-8 for information on (a) to (d) in the table.

Table A-27. 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 A-7 and Table A-8 for information on (a) to (d) in the table.

Table A-28. 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) ← (SP) - 2, ((SP)) ← (A)	-	-	-	-	-	-	-	-	-	-
PUSHW	AH	1	4	0	(c)	word (SP) ← (SP) - 2, ((SP)) ← (AH)	-	-	-	-	-	-	-	-	-	-
PUSHW	PS	1	4	0	(c)	word (SP) ← (SP) - 2, ((SP)) ← (PS)	-	-	-	-	-	-	-	-	-	-
PUSHW	rlst	2	*3	*5	*4	(SP) ← (SP) - 2n, ((SP)) ← (rlst)	-	-	-	-	-	-	-	-	-	-
POPW	A	1	3	0	(c)	word (A) ← ((SP)), (SP) ← (SP) + 2	-	*	-	-	-	-	-	-	-	-
POPW	AH	1	3	0	(c)	word (AH) ← ((SP)), (SP) ← (SP) + 2	-	-	-	-	-	-	-	-	-	-
POPW	PS	1	4	0	(c)	word (PS) ← ((SP)), (SP) ← (SP) + 2	-	-	*	*	*	*	*	*	*	-
POPW	rlst	2	*2	*5	*4	(rlst) ← ((SP)), (SP) ← (SP) + 2n	-	-	-	-	-	-	-	-	-	-
JCTX	@A	1	14	0	6 × (c)	Context switch instruction	-	-	*	*	*	*	*	*	*	-
AND	CCR,#imm8	2	3	0	0	byte (CCR) ← (CCR) and imm8	-	-	*	*	*	*	*	*	*	-
OR	CCR,#imm8	2	3	0	0	byte (CCR) ← (CCR) or imm8	-	-	*	*	*	*	*	*	*	-
MOV	RP,#imm8	2	2	0	0	byte (RP) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV	ILM,#imm8	2	2	0	0	byte (ILM) ← imm8	-	-	-	-	-	-	-	-	-	-
MOVEA	RWi,ear	2	3	1	0	word (RWi) ← ear	-	-	-	-	-	-	-	-	-	-
MOVEA	RWi,eam	2+	2+(a)	1	0	word (RWi) ← eam	-	-	-	-	-	-	-	-	-	-
MOVEA	A,ear	2	1	0	0	word (A) ← ear	-	*	-	-	-	-	-	-	-	-
MOVEA	A,eam	2+	1+(a)	0	0	word (A) ← eam	-	*	-	-	-	-	-	-	-	-
ADDSP	#imm8	2	3	0	0	word (SP) ← (SP) + ext(imm8)	-	-	-	-	-	-	-	-	-	-
ADDSP	#imm16	3	3	0	0	word (SP) ← (SP) + imm16	-	-	-	-	-	-	-	-	-	-
MOV	A,brg1	2	*1	0	0	byte (A) ← (brg1)	Z	*	-	-	-	*	*	-	-	-
MOV	brg2,A	2	1	0	0	byte (brg2) ← (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 × (POP count) + 2 × (POP last register number), 7 when RLST = 0 (no transfer register)

*3: 29 + 3 × (PUSH count) - 3 × (PUSH last register number), 8 when RLST = 0 (no transfer register)

*4: (POP count) × (c) or (PUSH count) × (c)

*5: (POP count) or (PUSH count)

Note:

See Table A-7 and Table A-8 for information on (a) and (c) in the table.

Table A-29. 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) ← (dir:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB	A,addr16:bp	4	5	0	(b)	byte (A) ← (addr16:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB	A,io:bp	3	4	0	(b)	byte (A) ← (io:bp)b	Z	*	-	-	-	*	*	-	-	-
MOVB	dir:bp,A	3	7	0	2 × (b)	bit (dir:bp)b ← (A)	-	-	-	-	-	*	*	-	-	*
MOVB	addr16:bp,A	4	7	0	2 × (b)	bit (addr16:bp)b ← (A)	-	-	-	-	-	*	*	-	-	*
MOVB	io:bp,A	3	6	0	2 × (b)	bit (io:bp)b ← (A)	-	-	-	-	-	*	*	-	-	*
SETB	dir:bp	3	7	0	2 × (b)	bit (dir:bp)b ← 1	-	-	-	-	-	-	-	-	-	*
SETB	addr16:bp	4	7	0	2 × (b)	bit (addr16:bp)b ← 1	-	-	-	-	-	-	-	-	-	*
SETB	io:bp	3	7	0	2 × (b)	bit (io:bp)b ← 1	-	-	-	-	-	-	-	-	-	*
CLRB	dir:bp	3	7	0	2 × (b)	bit (dir:bp)b ← 0	-	-	-	-	-	-	-	-	-	*
CLRB	addr16:bp	4	7	0	2 × (b)	bit (addr16:bp)b ← 0	-	-	-	-	-	-	-	-	-	*
CLRB	io:bp	3	7	0	2 × (b)	bit (io:bp)b ← 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 × (b)	Branch on (addr16:bp) b = 1, bit (addr16:bp) b ← 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 A-7 and Table A-8 for information on (b) in the table.

Table A-30. 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 ↔ (A)8-15	-	-	-	-	-	-	-	-	-	-
SWAPW	1	2	0	0	word (AH) ↔ (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 A-31. 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 × (RW0) when the counter expires, or 7n + 5 when a match occurs

*2: 5 when RW0 is 0; otherwise, 4 + 8 × (RW0)

*3: (b) × (RW0) + (b) × (RW0) When the source and destination access different areas, calculate the (b) item individually.

*4: (b) × n

*5: 2 × (b) × (RW0)

*6: (c) × (RW0) + (c) × (RW0) When the source and destination access different areas, calculate the (c) item individually.

*7: (c) × n

*8: (b) × (RW0)

Note:

m: RW0 value (counter value), n: Loop count

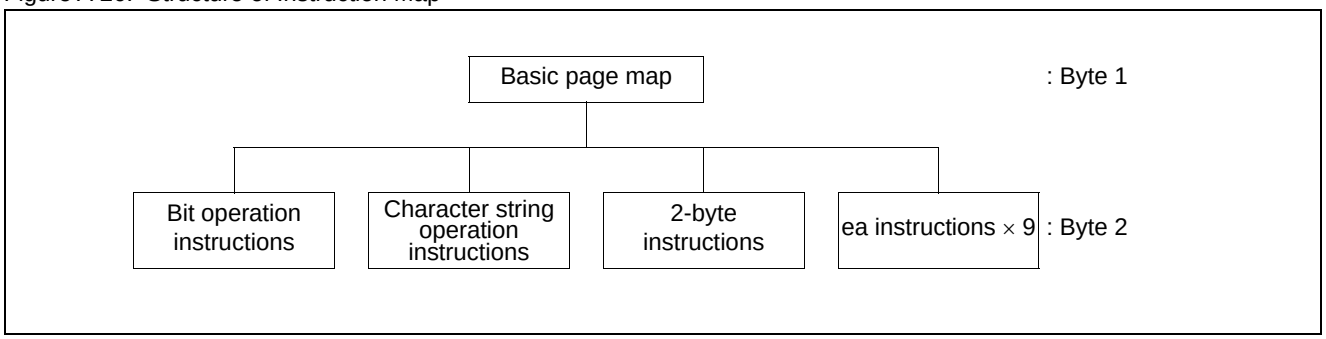
See Table A-7 and Table A-8 for information on (b) and (c) in the table.

A.4.9 Instruction Map

Each F^{2}M\C-16LX instruction code consists of 1 or 2 bytes. Therefore, the instruction map consists of multiple pages. Table A-33 to Table A-52 summarize the F^{2}MC-16LX instruction map.

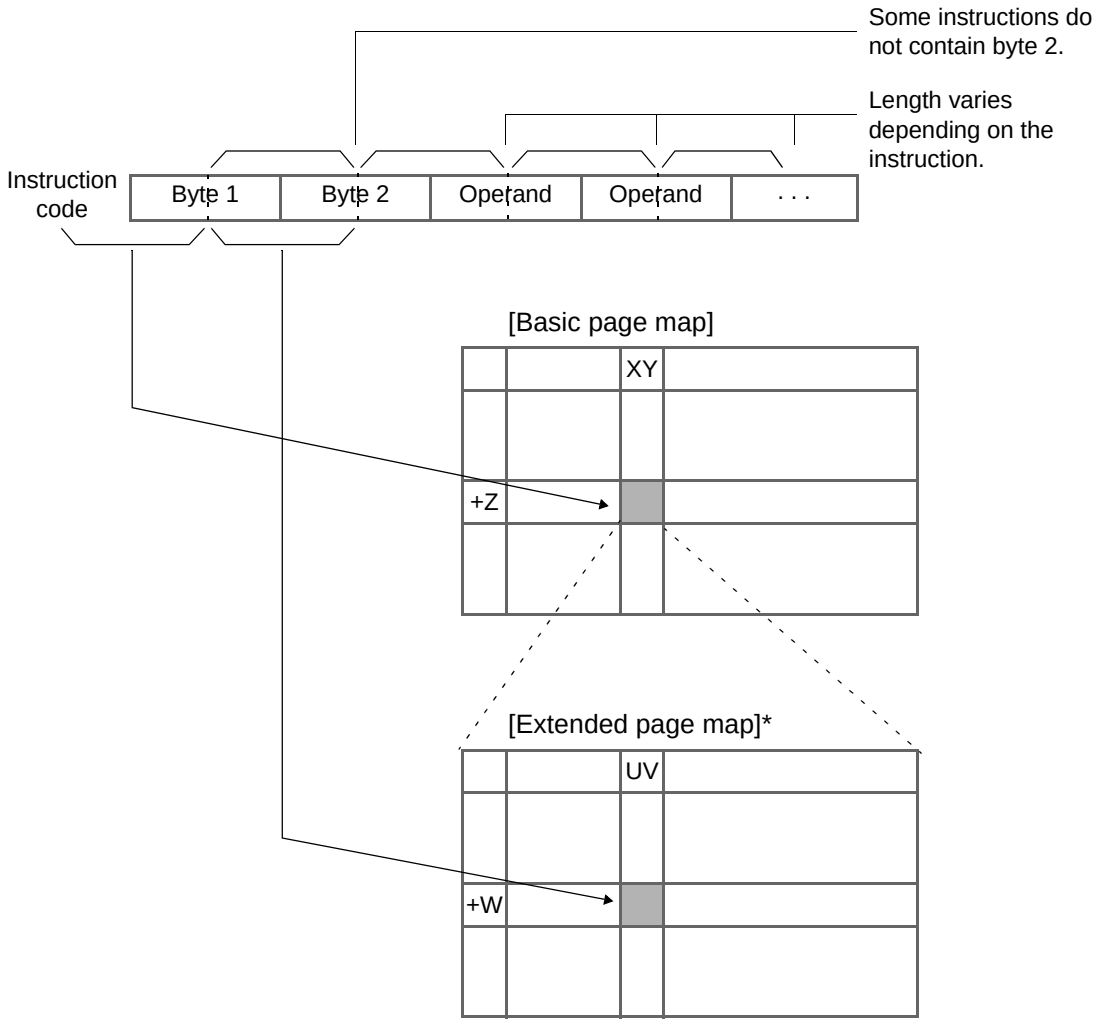
Structure of Instruction Map

Figure A-26. 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 A-27 shows the correspondence between an actual instruction code and instruction map.

Figure A-27. Correspondence between Actual Instruction Code and Instruction Map



*: The extended page map is a generic name of maps for bit operation instructions, character string operation instructions, 2-byte instructions, and ea instructions. Actually, there are multiple extended page maps for each type of instructions.

An example of an instruction code is shown in Table A-32.

Table A-32. 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
CBNE @RW2+d8, #8, rel	70 +0=70	F0 +2=F2

Table A-33. Basic Page Map

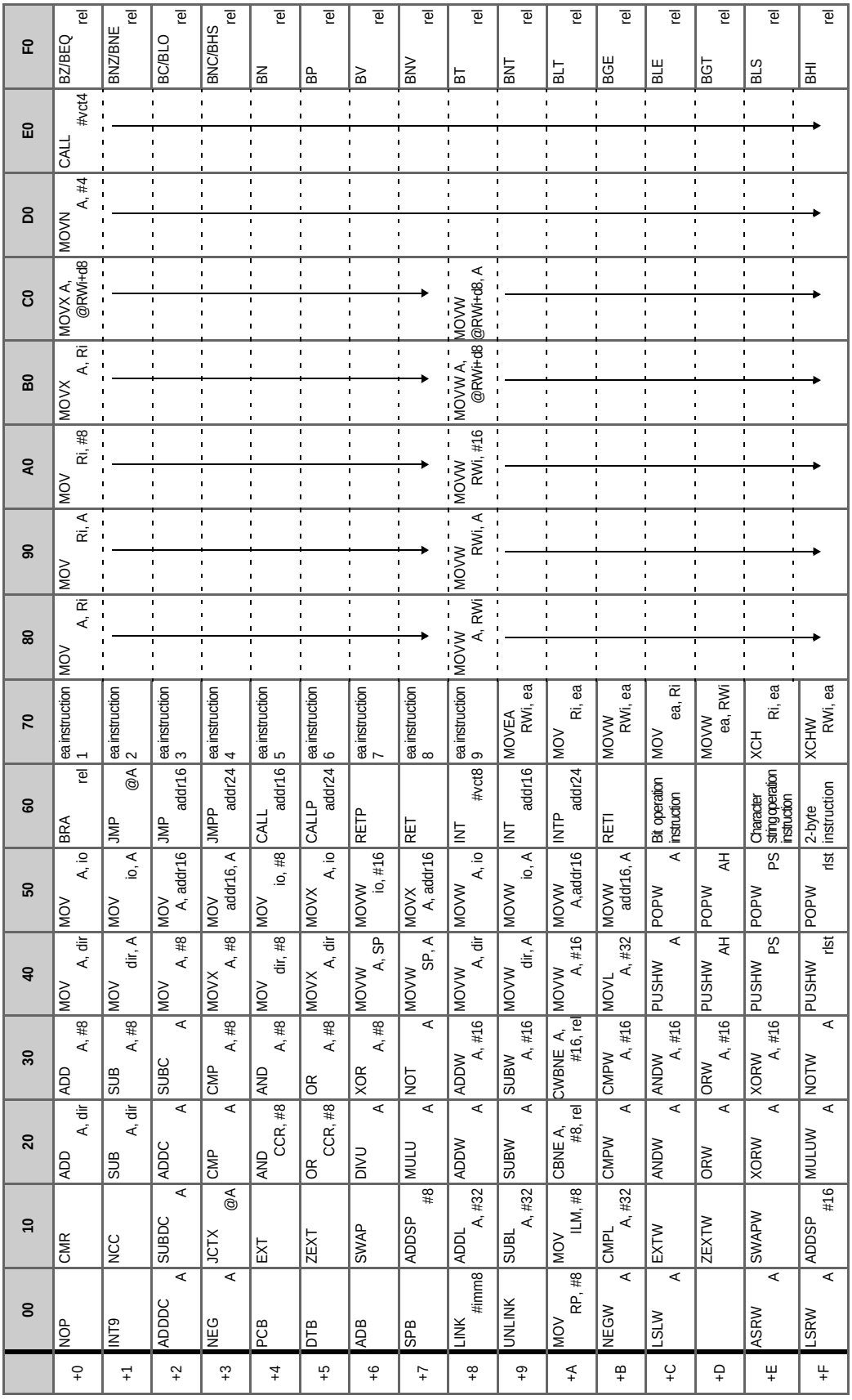


Table A-34. Bit Operation Instruction Map (First Byte = $6C_H$)

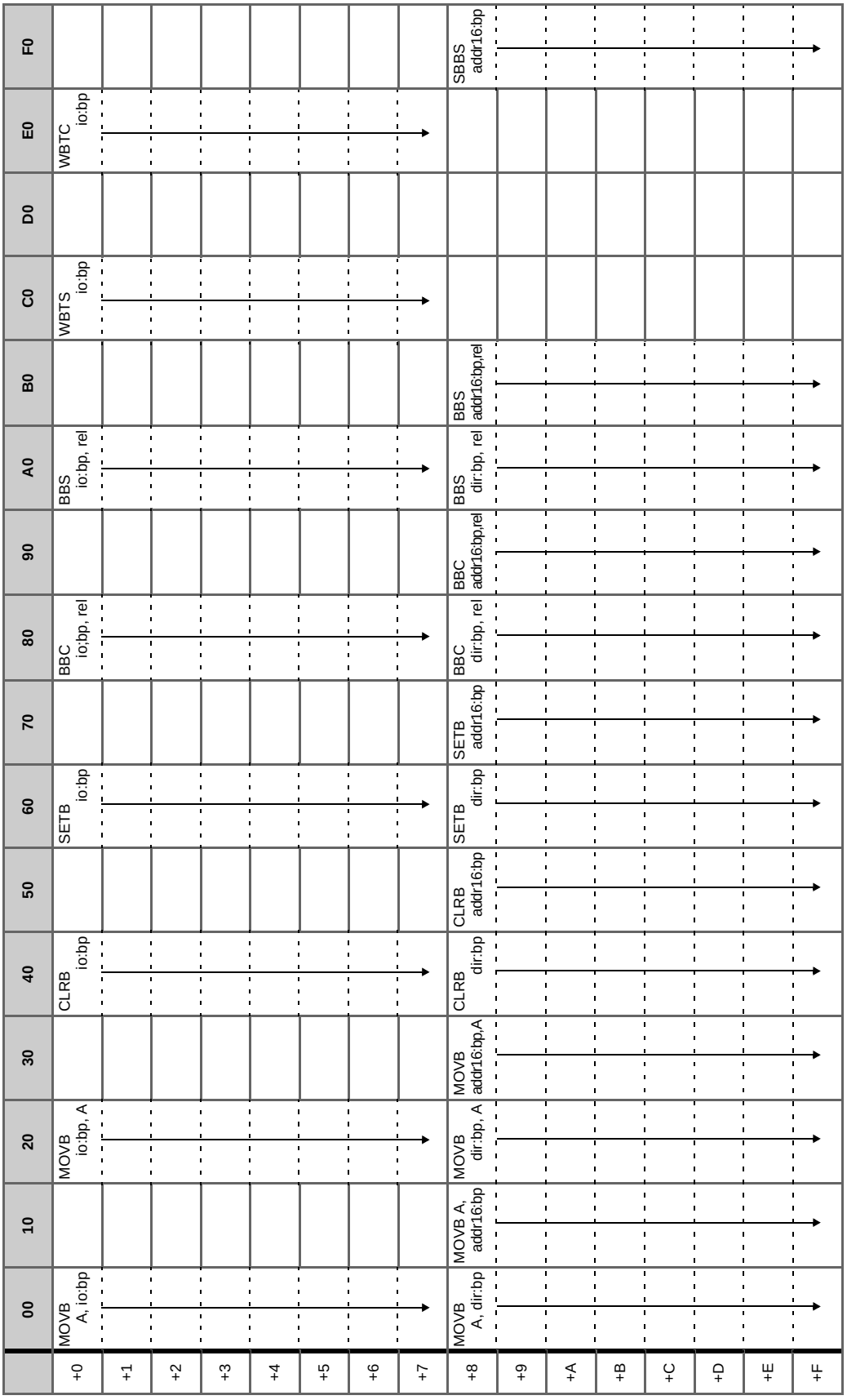


Table A-35. Character String Operation Instruction Map (First Byte = $6E_H$)

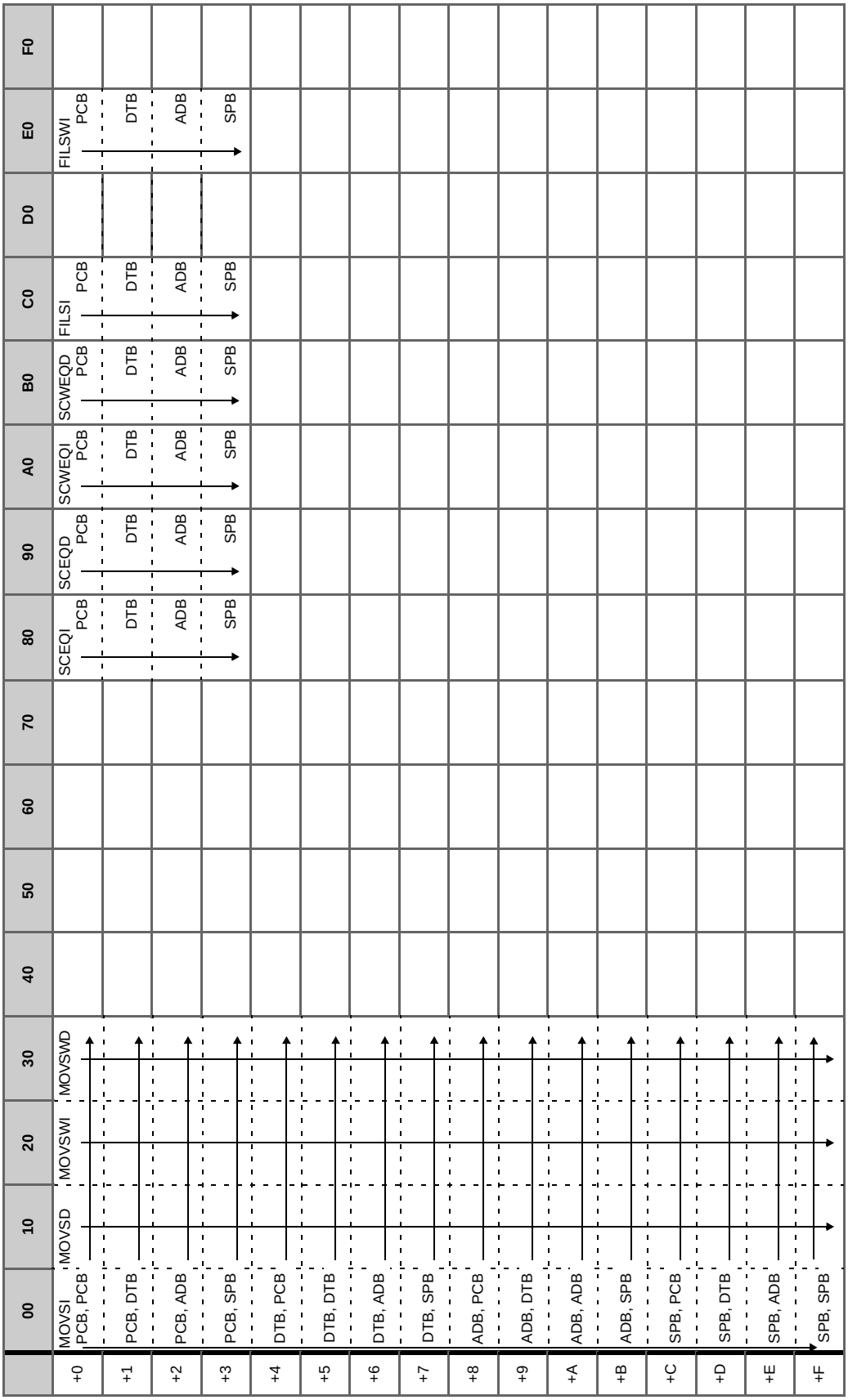


Table A-36. 2-byte Instruction Map (First Byte = $6F_H$)

	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	ROLC A	ROLC A														
+8				MOVW @RL0+d8,A	MOVW A,@RL0+d8			MUL A								
+9								MULW A								
+A				MOVW @RL1+d8,A	MOVW A,@RL1+d8			DIV 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 A-37. ea Instruction 1 (First Byte = 70_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
	ADDL A,	ADDL A,	SUBL A,	SUBL A,	CWBNE ↓	CWBNE ↓	CMPL A,	CMPL A,	ANDL A,	ANDL A,	ORL A,	ORL A,	XORL A,	XORL A,	CBNE ↓	CBNE ↓
+0	A, RL0	@RW0+d8	A, RL0	@RW0+d8	RW0, #16, rel	@RW0+d8 #16, rel	A, RL0	@RW0+d8	A, RL0	@RW0+d8	A, RL0	@RW0+d8	A, RL0	@RW0+d8	R0, #8, rel	@RW0+d8, #8, rel
+1	A, RL0	@RW1+d8	A, RL0	@RW1+d8	RW1, #16, rel	@RW1+d8 #16, rel	A, RL0	@RW1+d8	A, RL0	@RW1+d8	A, RL0	@RW1+d8	A, RL0	@RW1+d8	R1, #8, rel	@RW1+d8, #8, rel
+2	A, RL1	@RW2+d8	A, RL1	@RW2+d8	RW2, #16, rel	@RW2+d8 #16, rel	A, RL1	@RW2+d8	A, RL1	@RW2+d8	A, RL1	@RW2+d8	A, RL1	@RW2+d8	R2, #8, rel	@RW2+d8, #8, rel
+3	A, RL1	@RW3+d8	A, RL1	@RW3+d8	RW3, #16, rel	@RW3+d8 #16, rel	A, RL1	@RW3+d8	A, RL1	@RW3+d8	A, RL1	@RW3+d8	A, RL1	@RW3+d8	R3, #8, rel	@RW3+d8, #8, rel
+4	A, RL2	@RW4+d8	A, RL2	@RW4+d8	RW4, #16, rel	@RW4+d8 #16, rel	A, RL2	@RW4+d8	A, RL2	@RW4+d8	A, RL2	@RW4+d8	A, RL2	@RW4+d8	R4, #8, rel	@RW4+d8, #8, rel
+5	A, RL2	@RW5+d8	A, RL2	@RW5+d8	RW5, #16, rel	@RW5+d8 #16, rel	A, RL2	@RW5+d8	A, RL2	@RW5+d8	A, RL2	@RW5+d8	A, RL2	@RW5+d8	R5, #8, rel	@RW5+d8, #8, rel
+6	A, RL3	@RW6+d8	A, RL3	@RW6+d8	RW6, #16, rel	@RW6+d8 #16, rel	A, RL3	@RW6+d8	A, RL3	@RW6+d8	A, RL3	@RW6+d8	A, RL3	@RW6+d8	R6, #8, rel	@RW6+d8, #8, rel
+7	A, RL3	@RW7+d8	A, RL3	@RW7+d8	RW7, #16, rel	@RW7+d8 #16, rel	A, RL3	@RW7+d8	A, RL3	@RW7+d8	A, RL3	@RW7+d8	A, RL3	@RW7+d8	R7, #8, rel	@RW7+d8, #8, rel
+8	A,@RW0	@RW0+d16	A,@RW0	@RW0+d16	@RW0 #16, rel	@RW0+d16 #16, rel	A,@RW0	@RW0+d16	A,@RW0	@RW0+d16	A,@RW0	@RW0+d16	A,@RW0	@RW0+d16	@RW0, #8, rel	@RW0+d16, #8, rel
+9	A,@RW1	@RW1+d16	A,@RW1	@RW1+d16	@RW1 #16, rel	@RW1+d16 #16, rel	A,@RW1	@RW1+d16	A,@RW1	@RW1+d16	A,@RW1	@RW1+d16	A,@RW1	@RW1+d16	@RW1, #8, rel	@RW1+d16, #8, rel
+A	A,@RW2	@RW2+d16	A,@RW2	@RW2+d16	@RW2 #16, rel	@RW2+d16 #16, rel	A,@RW2	@RW2+d16	A,@RW2	@RW2+d16	A,@RW2	@RW2+d16	A,@RW2	@RW2+d16	@RW2, #8, rel	@RW2+d16, #8, rel
+B	A,@RW3	@RW3+d16	A,@RW3	@RW3+d16	@RW3 #16, rel	@RW3+d16 #16, rel	A,@RW3	@RW3+d16	A,@RW3	@RW3+d16	A,@RW3	@RW3+d16	A,@RW3	@RW3+d16	@RW3, #8, rel	@RW3+d16, #8, rel
+C	A,@RW0+	@RW0+RW7	A,@RW0+	@RW0+RW7	Use prohibited	@RW0+RW7 #16, rel	A,@RW0+	@RW0+RW7	A,@RW0+	@RW0+RW7	A,@RW0+	@RW0+RW7	A,@RW0+	@RW0+RW7	Use prohibited	@RW0+RW7, #8, rel
+D	A,@RW1+	@RW1+RW7	A,@RW1+	@RW1+RW7	Use prohibited	@RW1+RW7 #16, rel	A,@RW1+	@RW1+RW7	A,@RW1+	@RW1+RW7	A,@RW1+	@RW1+RW7	A,@RW1+	@RW1+RW7	Use prohibited	@RW1+RW7, #8, rel
+E	A,@RW2+	@PC+d16	A,@RW2+	@PC+d16	Use prohibited	@PC+d16, #16, rel	A,@RW2+	@PC+d16	A,@RW2+	@PC+d16	A,@RW2+	@PC+d16	A,@RW2+	@PC+d16	Use prohibited	@PC+d16, #8, rel
+F	A,@RW3+	addr16	A,@RW3+	addr16	Use prohibited	addr16, #16, rel	A,@RW3+	addr16	A,@RW3+	addr16	A,@RW3+	addr16	A,@RW3+	addr16	Use prohibited	addr16, #8, rel

Table A-38. ea Instruction 2 (First Byte = 71_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	JMPP @RL0	JMPP @@RW0+d8	CALLP @RL0	CALLP @@RW0+d8	INCL RL0	INCL @RW0+d8	DECL RL0	DECL @RW0+d8	MOVL A, RL0	MOVL A, @RW0+d8	MOVL RL0, A	MOVL @RW0+d8,A	MOV R0, #8	MOV @RW0+d8,#8	MOVEA A, RW0	MOVEA A, @RW0+d8
+1	JMPP @RL0	JMPP @@RW1+d8	CALLP @RL0	CALLP @@RW1+d8	INCL RL0	INCL @RW1+d8	DECL RL0	DECL @RW1+d8	MOVL A, RL0	MOVL A, @RW1+d8	MOVL RL0, A	MOVL @RW1+d8,A	MOV R1, #8	MOV @RW1+d8,#8	MOVEA A, RW1	MOVEA A, @RW1+d8
+2	JMPP @RL1	JMPP @@RW2+d8	CALLP @RL1	CALLP @@RW2+d8	INCL RL1	INCL @RW2+d8	DECL RL1	DECL @RW2+d8	MOVL A, RL1	MOVL A, @RW2+d8	MOVL RL1, A	MOVL @RW2+d8,A	MOV R2, #8	MOV @RW2+d8,#8	MOVEA A, RW2	MOVEA A, @RW2+d8
+3	JMPP @RL1	JMPP @@RW3+d8	CALLP @RL1	CALLP @@RW3+d8	INCL RL1	INCL @RW3+d8	DECL RL1	DECL @RW3+d8	MOVL A, RL1	MOVL A, @RW3+d8	MOVL RL1, A	MOVL @RW3+d8,A	MOV R3, #8	MOV @RW3+d8,#8	MOVEA A, RW3	MOVEA A, @RW3+d8
+4	JMPP @RL2	JMPP @@RW4+d8	CALLP @RL2	CALLP @@RW4+d8	INCL RL2	INCL @RW4+d8	DECL RL2	DECL @RW4+d8	MOVL A, RL2	MOVL A, @RW4+d8	MOVL RL2, A	MOVL @RW4+d8,A	MOV R4, #8	MOV @RW4+d8,#8	MOVEA A, RW4	MOVEA A, @RW4+d8
+5	JMPP @RL2	JMPP @@RW5+d8	CALLP @RL2	CALLP @@RW5+d8	INCL RL2	INCL @RW5+d8	DECL RL2	DECL @RW5+d8	MOVL A, RL2	MOVL A, @RW5+d8	MOVL RL2, A	MOVL @RW5+d8,A	MOV R5, #8	MOV @RW5+d8,#8	MOVEA A, RW5	MOVEA A, @RW5+d8
+6	JMPP @RL3	JMPP @@RW6+d8	CALLP @RL3	CALLP @@RW6+d8	INCL RL3	INCL @RW6+d8	DECL RL3	DECL @RW6+d8	MOVL A, RL3	MOVL A, @RW6+d8	MOVL RL3, A	MOVL @RW6+d8,A	MOV R6, #8	MOV @RW6+d8,#8	MOVEA A, RW6	MOVEA A, @RW6+d8
+7	JMPP @RL3	JMPP @@RW7+d8	CALLP @RL3	CALLP @@RW7+d8	INCL RL3	INCL @RW7+d8	DECL RL3	DECL @RW7+d8	MOVL A, RL3	MOVL A, @RW7+d8	MOVL RL3, A	MOVL @RW7+d8,A	MOV R7, #8	MOV @RW7+d8,#8	MOVEA A, RW7	MOVEA A, @RW7+d8
+8	JMPP @@RW0	JMPP @ @RW0+d16	CALLP @@RW0	CALLP @ @RW0+d16	INCL @RW0	INCL @RW0+d16	DECL @RW0	DECL @RW0+d16	MOVL A,@RW0	MOVL A, @RW0+d16	MOVL @RW0,A	MOVL @RW0+d16,A	MOV @RW0, #8	MOV @RW0+d16,#8	MOVEA A,@RW0	MOVEA A, @RW0+d16
+9	JMPP @@RW1	JMPP @ @RW1+d16	CALLP @@RW1	CALLP @ @RW1+d16	INCL @RW1	INCL @RW1+d16	DECL @RW1	DECL @RW1+d16	MOVL A,@RW1	MOVL A, @RW1+d16	MOVL @RW1,A	MOVL @RW1+d16,A	MOV @RW1, #8	MOV @RW1+d16,#8	MOVEA A,@RW1	MOVEA A, @RW1+d16
+A	JMPP @@RW2	JMPP @ @RW2+d16	CALLP @@RW2	CALLP @ @RW2+d16	INCL @RW2	INCL @RW2+d16	DECL @RW2	DECL @RW2+d16	MOVL A,@RW2	MOVL A, @RW2+d16	MOVL @RW2,A	MOVL @RW2+d16,A	MOV @RW2, #8	MOV @RW2+d16,#8	MOVEA A,@RW2	MOVEA A, @RW2+d16
+B	JMPP @@RW3	JMPP @ @RW3+d16	CALLP @@RW3	CALLP @ @RW3+d16	INCL @RW3	INCL @RW3+d16	DECL @RW3	DECL @RW3+d16	MOVL A,@RW3	MOVL A, @RW3+d16	MOVL @RW3,A	MOVL @RW3+d16,A	MOV @RW3, #8	MOV @RW3+d16,#8	MOVEA A,@RW3	MOVEA A, @RW3+d16
+C	JMPP @@RW0+	JMPP @ @RW0+RW7	CALLP @@RW0+	CALLP @ @RW0+RW7	INCL @RW0+	INCL @RW0+RW7	DECL @RW0+	DECL @RW0+RW7	MOVL A,@RW0+	MOVL A, @RW0+RW7	MOVL @RW0+,A	MOVL @RW0+RW7,A	MOV @RW0+, #8	MOV @RW0+RW7,#8	MOVEA A,@RW0+	MOVEA A, @RW0+RW7
+D	JMPP @@RW1+	JMPP @ @RW1+RW7	CALLP @@RW1+	CALLP @ @RW1+RW7	INCL @RW1+	INCL @RW1+RW7	DECL @RW1+	DECL @RW1+RW7	MOVL A,@RW1+	MOVL A, @RW1+RW7	MOVL @RW1+,A	MOVL @RW1+RW7,A	MOV @RW1+, #8	MOV @RW1+RW7,#8	MOVEA A,@RW1+	MOVEA A, @RW1+RW7
+E	JMPP @@RW2+	JMPP @@PC+d16	CALLP @@RW2+	CALLP @@PC+d16	INCL @RW2+	INCL @PC+d16	DECL @RW2+	DECL @PC+d16	MOVL A,@RW2+	MOVL A, @PC+d16	MOVL @RW2+,A	MOVL @PC+d16, A	MOV @RW2+, #8	MOV @PC+d16, #8	MOVEA A,@RW2+	MOVEA A, @PC+d16
+F	JMPP @@RW3+	JMPP @addr16	CALLP @@RW3+	CALLP @addr16	INCL @RW3+	INCL addr16	DECL @RW3+	DECL addr16	MOVL A,@RW3+	MOVL A, addr16	MOVL @RW3+,A	MOVL addr16, A	MOV @RW3+, #8	MOV addr16, #8	MOVEA A,@RW3+	MOVEA A, addr16

Table A-39. ea Instruction 3 (First Byte = 72_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ROLC R0	ROLC @RW0+d8	RORC R0	RORC @RW0+d8	INC R0	INC @RW0+d8	DEC R0	DEC @RW0+d8	MOV A, R0	MOV A, @RW0+d8	MOV R0, A	MOV A, @RW0+d8,A	MOVX A, R0	MOVX A, @RW0+d8	XCH A, R0	XCH A, @RW0+d8
+1	ROLC R1	ROLC @RW1+d8	RORC R1	RORC @RW1+d8	INC R1	INC @RW1+d8	DEC R1	DEC @RW1+d8	MOV A, R1	MOV A, @RW1+d8	MOV R1, A	MOV A, @RW1+d8,A	MOVX A, R1	MOVX A, @RW1+d8	XCH A, R1	XCH A, @RW1+d8
+2	ROLC R2	ROLC @RW2+d8	RORC R2	RORC @RW2+d8	INC R2	INC @RW2+d8	DEC R2	DEC @RW2+d8	MOV A, R2	MOV A, @RW2+d8	MOV R2, A	MOV A, @RW2+d8,A	MOVX A, R2	MOVX A, @RW2+d8	XCH A, R2	XCH A, @RW2+d8
+3	ROLC R3	ROLC @RW3+d8	RORC R3	RORC @RW3+d8	INC R3	INC @RW3+d8	DEC R3	DEC @RW3+d8	MOV A, R3	MOV A, @RW3+d8	MOV R3, A	MOV A, @RW3+d8,A	MOVX A, R3	MOVX A, @RW3+d8	XCH A, R3	XCH A, @RW3+d8
+4	ROLC R4	ROLC @RW4+d8	RORC R4	RORC @RW4+d8	INC R4	INC @RW4+d8	DEC R4	DEC @RW4+d8	MOV A, R4	MOV A, @RW4+d8	MOV R4, A	MOV A, @RW4+d8,A	MOVX A, R4	MOVX A, @RW4+d8	XCH A, R4	XCH A, @RW4+d8
+5	ROLC R5	ROLC @RW5+d8	RORC R5	RORC @RW5+d8	INC R5	INC @RW5+d8	DEC R5	DEC @RW5+d8	MOV A, R5	MOV A, @RW5+d8	MOV R5, A	MOV A, @RW5+d8,A	MOVX A, R5	MOVX A, @RW5+d8	XCH A, R5	XCH A, @RW5+d8
+6	ROLC R6	ROLC @RW6+d8	RORC R6	RORC @RW6+d8	INC R6	INC @RW6+d8	DEC R6	DEC @RW6+d8	MOV A, R6	MOV A, @RW6+d8	MOV R6, A	MOV A, @RW6+d8,A	MOVX A, R6	MOVX A, @RW6+d8	XCH A, R6	XCH A, @RW6+d8
+7	ROLC R7	ROLC @RW7+d8	RORC R7	RORC @RW7+d8	INC R7	INC @RW7+d8	DEC R7	DEC @RW7+d8	MOV A, R7	MOV A, @RW7+d8	MOV R7, A	MOV A, @RW7+d8,A	MOVX A, R7	MOVX A, @RW7+d8	XCH A, R7	XCH A, @RW7+d8
+8	ROLC @RW0	ROLC @RW0+d16	RORC @RW0	RORC @RW0+d16	INC @RW0	INC @RW0+d16	DEC @RW0	DEC @RW0+d16	MOV A,@RW0	MOV A, @RW0+d16	MOV @RW0, A	MOV @RW0+d16,A	MOVX A,@RW0	MOVX A, @RW0+d16	XCH A,@RW0	XCH A, @RW0+d16
+9	ROLC @RW1	ROLC @RW1+d16	RORC @RW1	RORC @RW1+d16	INC @RW1	INC @RW1+d16	DEC @RW1	DEC @RW1+d16	MOV A,@RW1	MOV A, @RW1+d16	MOV @RW1, A	MOV @RW1+d16,A	MOVX A,@RW1	MOVX A, @RW1+d16	XCH A,@RW1	XCH A, @RW1+d16
+A	ROLC @RW2	ROLC @RW2+d16	RORC @RW2	RORC @RW2+d16	INC @RW2	INC @RW2+d16	DEC @RW2	DEC @RW2+d16	MOV A,@RW2	MOV A, @RW2+d16	MOV @RW2, A	MOV @RW2+d16,A	MOVX A,@RW2	MOVX A, @RW2+d16	XCH A,@RW2	XCH A, @RW2+d16
+B	ROLC @RW3	ROLC @RW3+d16	RORC @RW3	RORC @RW3+d16	INC @RW3	INC @RW3+d16	DEC @RW3	DEC @RW3+d16	MOV A,@RW3	MOV A, @RW3+d16	MOV @RW3, A	MOV @RW3+d16,A	MOVX A,@RW3	MOVX A, @RW3+d16	XCH A,@RW3	XCH A, @RW3+d16
+C	ROLC @RW0+	ROLC @RW0+RW7	RORC @RW0+	RORC @RW0+RW7	INC @RW0+	INC @RW0+RW7	DEC @RW0+	DEC @RW0+RW7	MOV A,@RW0+	MOV A, @RW0+RW7	MOV @RW0+, A	MOV @RW0+RW7,A	MOVX A,@RW0+	MOVX A, @RW0+RW7	XCH A,@RW0+	XCH A, @RW0+RW7
+D	ROLC @RW1+	ROLC @RW1+RW7	RORC @RW1+	RORC @RW1+RW7	INC @RW1+	INC @RW1+RW7	DEC @RW1+	DEC @RW1+RW7	MOV A,@RW1+	MOV A, @RW1+RW7	MOV @RW1+, A	MOV @RW1+RW7,A	MOVX A,@RW1+	MOVX A, @RW1+RW7	XCH A,@RW1+	XCH A, @RW1+RW7
+E	ROLC @RW2+	ROLC @PC+d16	RORC @RW2+	RORC @PC+d16	INC @RW2+	INC @PC+d16	DEC @RW2+	DEC @PC+d16	MOV A,@RW2+	MOV A, @PC+d16	MOV @RW2+, A	MOV @PC+d16, A	MOVX A,@RW2+	MOVX A, @PC+d16	XCH A,@RW2+	XCH A, @PC+d16
+F	ROLC @RW3+	ROLC addr16	RORC @RW3+	RORC addr16	INC @RW3+	INC addr16	DEC @RW3+	DEC addr16	MOV A,@RW3+	MOV A, addr16	MOV @RW3+, A	MOV addr16, A	MOVX A,@RW3+	MOVX A, addr16	XCH A,@RW3+	XCH A, addr16

Table A-40. ea Instruction 4 (First Byte = 73_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	JMP @RW0	JMP @@RW0+d8	CALL @RW0	CALL @@RW0+d8	INCW RW0	INCW @RW0+d8	DECW RW0	DECW @RW0+d8	MOVW A, RW0	MOVW A, @RW0+d8	MOVW RW0, A	MOVW @RW0+d8,A	MOVW RW0, #16	MOVW @RW0+d8,#16	XCHW A, RW0	XCHW A, @RW0+d8
+1	JMP @RW1	JMP @@RW1+d8	CALL @RW1	CALL @@RW1+d8	INCW RW1	INCW @RW1+d8	DECW RW1	DECW @RW1+d8	MOVW A, RW1	MOVW A, @RW1+d8	MOVW RW1, A	MOVW @RW1+d8,A	MOVW RW1, #16	MOVW @RW1+d8,#16	XCHW A, RW1	XCHW A, @RW1+d8
+2	JMP @RW2	JMP @@RW2+d8	CALL @RW2	CALL @@RW2+d8	INCW RW2	INCW @RW2+d8	DECW RW2	DECW @RW2+d8	MOVW A, RW2	MOVW A, @RW2+d8	MOVW RW2, A	MOVW @RW2+d8,A	MOVW RW2, #16	MOVW @RW2+d8,#16	XCHW A, RW2	XCHW A, @RW2+d8
+3	JMP @RW3	JMP @@RW3+d8	CALL @RW3	CALL @@RW3+d8	INCW RW3	INCW @RW3+d8	DECW RW3	DECW @RW3+d8	MOVW A, RW3	MOVW A, @RW3+d8	MOVW RW3, A	MOVW @RW3+d8,A	MOVW RW3, #16	MOVW @RW3+d8,#16	XCHW A, RW3	XCHW A, @RW3+d8
+4	JMP @RW4	JMP @@RW4+d8	CALL @RW4	CALL @@RW4+d8	INCW RW4	INCW @RW4+d8	DECW RW4	DECW @RW4+d8	MOVW A, RW4	MOVW A, @RW4+d8	MOVW RW4, A	MOVW @RW4+d8,A	MOVW RW4, #16	MOVW @RW4+d8,#16	XCHW A, RW4	XCHW A, @RW4+d8
+5	JMP @RW5	JMP @@RW5+d8	CALL @RW5	CALL @@RW5+d8	INCW RW5	INCW @RW5+d8	DECW RW5	DECW @RW5+d8	MOVW A, RW5	MOVW A, @RW5+d8	MOVW RW5, A	MOVW @RW5+d8,A	MOVW RW5, #16	MOVW @RW5+d8,#16	XCHW A, RW5	XCHW A, @RW5+d8
+6	JMP @RW6	JMP @@RW6+d8	CALL @RW6	CALL @@RW6+d8	INCW RW6	INCW @RW6+d8	DECW RW6	DECW @RW6+d8	MOVW A, RW6	MOVW A, @RW6+d8	MOVW RW6, A	MOVW @RW6+d8,A	MOVW RW6, #16	MOVW @RW6+d8,#16	XCHW A, RW6	XCHW A, @RW6+d8
+7	JMP @RW7	JMP @@RW7+d8	CALL @RW7	CALL @@RW7+d8	INCW RW7	INCW @RW7+d8	DECW RW7	DECW @RW7+d8	MOVW A, RW7	MOVW A, @RW7+d8	MOVW RW7, A	MOVW @RW7+d8,A	MOVW RW7, #16	MOVW @RW7+d8,#16	XCHW A, RW7	XCHW A, @RW7+d8
+8	JMP @@RW0	JMP @RW0+d16	CALL @@RW0	CALL @ @RW0+d16	INCW @RW0	INCW @RW0+d16	DECW @RW0	DECW @RW0+d16	MOVW A,@RW0	MOVW A, @RW0+d16	MOVW @RW0,A	MOVW @RW0+d16,A	MOVW @RW0, #16	MOVW @RW0+d16,#16	XCHW A,@RW0	XCHW A, @RW0+d16
+9	JMP @@RW1	JMP @ @RW1+d16	CALL @@RW1	CALL @ @RW1+d16	INCW @RW1	INCW @RW1+d16	DECW @RW1	DECW @RW1+d16	MOVW A,@RW1	MOVW A, @RW1+d16	MOVW @RW1, A	MOVW @RW1+d16,A	MOVW @RW1, #16	MOVW @RW1+d16,#16	XCHW A,@RW1	XCHW A, @RW1+d16
+A	JMP @@RW2	JMP @ @RW2+d16	CALL @@RW2	CALL @ @RW2+d16	INCW @RW2	INCW @RW2+d16	DECW @RW2	DECW @RW2+d16	MOVW A,@RW2	MOVW A, @RW2+d16	MOVW @RW2, A	MOVW @RW2+d16,A	MOVW @RW2, #16	MOVW @RW2+d16,#16	XCHW A,@RW2	XCHW A, @RW2+d16
+B	JMP @@RW3	JMP @ @RW3+d16	CALL @@RW3	CALL @ @RW3+d16	INCW @RW3	INCW @RW3+d16	DECW @RW3	DECW @RW3+d16	MOVW A,@RW3	MOVW A, @RW3+d16	MOVW @RW3, A	MOVW @RW3+d16,A	MOVW @RW3, #16	MOVW @RW3+d16,#16	XCHW A,@RW3	XCHW A, @RW3+d16
+C	JMP @@RW0+	JMP @ @RW0+RW7	CALL @@RW0+	CALL @ @RW0+RW7	INCW @RW0+	INCW @RW0+RW7	DECW @RW0+	DECW @RW0+RW7	MOVW A,@RW0+	MOVW A, @RW0+RW7	MOVW @RW0+, A	MOVW @RW0+RW7,A	MOVW @RW0+, #16	MOVW @RW0+RW7,#16	XCHW A,@RW0+	XCHW A, @RW0+RW7
+D	JMP @@RW1+	JMP @ @RW1+RW7	CALL @@RW1+	CALL @ @RW1+RW7	INCW @RW1+	INCW @RW1+RW7	DECW @RW1+	DECW @RW1+RW7	MOVW A,@RW1+	MOVW A, @RW1+RW7	MOVW @RW1+, A	MOVW @RW1+RW7,A	MOVW @RW1+, #16	MOVW @RW1+RW7,#16	XCHW A,@RW1+	XCHW A, @RW1+RW7
+E	JMP @@RW2+	JMP @@PC+d16	CALL @@RW2+	CALL @@PC+d16	INCW @RW2+	INCW @PC+d16	DECW @RW2+	DECW @PC+d16	MOVW A,@RW2+	MOVW A, @PC+d16	MOVW @RW2+, A	MOVW @PC+d16, A	MOVW @RW2+, #16	MOVW @PC+d16, #16	XCHW A,@RW2+	XCHW A, @PC+d16
+F	JMP @@RW3+	JMP @addr16	CALL @@RW3+	CALL @addr16	INCW @RW3+	INCW addr16	DECW @RW3+	DECW addr16	MOVW A,@RW3+	MOVW A, addr16	MOVW @RW3+, A	MOVW addr16, A	MOVW @RW3+, #16	MOVW addr16, #16	XCHW A,@RW3+	XCHW A, addr16

Table A-41. ea Instruction 5 (First Byte = 74_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADD A, R0	ADD A, @RW0+d8	SUB A, R0	SUB A, @RW0+d8	ADDC A, R0	ADDC A, @RW0+d8	CMP A, R0	CMP A, @RW0+d8	AND A, R0	AND A, @RW0+d8	OR A, R0	OR A, @RW0+d8	XOR A, R0	XOR A, @RW0+d8	DBNZ R0, rel	DBNZ @RW0 +d8, rel
+1	ADD A, R1	ADD A, @RW1+d8	SUB A, R1	SUB A, @RW1+d8	ADDC A, R1	ADDC A, @RW1+d8	CMP A, R1	CMP A, @RW1+d8	AND A, R1	AND A, @RW1+d8	OR A, R1	OR A, @RW1+d8	XOR A, R1	XOR A, @RW1+d8	DBNZ R1, rel	DBNZ @RW1 +d8, rel
+2	ADD A, R2	ADD A, @RW2+d8	SUB A, R2	SUB A, @RW2+d8	ADDC A, R2	ADDC A, @RW2+d8	CMP A, R2	CMP A, @RW2+d8	AND A, R2	AND A, @RW2+d8	OR A, R2	OR A, @RW2+d8	XOR A, R2	XOR A, @RW2+d8	DBNZ R2, rel	DBNZ @RW2 +d8, rel
+3	ADD A, R3	ADD A, @RW3+d8	SUB A, R3	SUB A, @RW3+d8	ADDC A, R3	ADDC A, @RW3+d8	CMP A, R3	CMP A, @RW3+d8	AND A, R3	AND A, @RW3+d8	OR A, R3	OR A, @RW3+d8	XOR A, R3	XOR A, @RW3+d8	DBNZ R3, rel	DBNZ @RW3 +d8, rel
+4	ADD A, R4	ADD A, @RW4+d8	SUB A, R4	SUB A, @RW4+d8	ADDC A, R4	ADDC A, @RW4+d8	CMP A, R4	CMP A, @RW4+d8	AND A, R4	AND A, @RW4+d8	OR A, R4	OR A, @RW4+d8	XOR A, R4	XOR A, @RW4+d8	DBNZ R4, rel	DBNZ @RW4 +d8, rel
+5	ADD A, R5	ADD A, @RW5+d8	SUB A, R5	SUB A, @RW5+d8	ADDC A, R5	ADDC A, @RW5+d8	CMP A, R5	CMP A, @RW5+d8	AND A, R5	AND A, @RW5+d8	OR A, R5	OR A, @RW5+d8	XOR A, R5	XOR A, @RW5+d8	DBNZ R5, rel	DBNZ @RW5 +d8, rel
+6	ADD A, R6	ADD A, @RW6+d8	SUB A, R6	SUB A, @RW6+d8	ADDC A, R6	ADDC A, @RW6+d8	CMP A, R6	CMP A, @RW6+d8	AND A, R6	AND A, @RW6+d8	OR A, R6	OR A, @RW6+d8	XOR A, R6	XOR A, @RW6+d8	DBNZ R6, rel	DBNZ @RW6 +d8, rel
+7	ADD A, R7	ADD A, @RW7+d8	SUB A, R7	SUB A, @RW7+d8	ADDC A, R7	ADDC A, @RW7+d8	CMP A, R7	CMP A, @RW7+d8	AND A, R7	AND A, @RW7+d8	OR A, R7	OR A, @RW7+d8	XOR A, R7	XOR A, @RW7+d8	DBNZ R7, rel	DBNZ @RW7 +d8, rel
+8	ADD A,@RW0	ADD A, @RW0+d16	SUB A,@RW0	SUB A, @RW0+d16	ADDC A,@RW0	ADDC A, @RW0+d16	CMP A,@RW0	CMP A, @RW0+d16	AND A,@RW0	AND A, @RW0+d16	OR A,@RW0	OR A, @RW0+d16	XOR A,@RW0	XOR A, @RW0+d16	DBNZ @RW0, rel	DBNZ @RW0 +d16, rel
+9	ADD A,@RW1	ADD A, @RW1+d16	SUB A,@RW1	SUB A, @RW1+d16	ADDC A,@RW1	ADDC A, @RW1+d16	CMP A,@RW1	CMP A, @RW1+d16	AND A,@RW1	AND A, @RW1+d16	OR A,@RW1	OR A, @RW1+d16	XOR A,@RW1	XOR A, @RW1+d16	DBNZ @RW1, rel	DBNZ @RW1 +d16, rel
+A	ADD A,@RW2	ADD A, @RW2+d16	SUB A,@RW2	SUB A, @RW2+d16	ADDC A,@RW2	ADDC A, @RW2+d16	CMP A,@RW2	CMP A, @RW2+d16	AND A,@RW2	AND A, @RW2+d16	OR A,@RW2	OR A, @RW2+d16	XOR A,@RW2	XOR A, @RW2+d16	DBNZ @RW2, rel	DBNZ @RW2 +d16, rel
+B	ADD A,@RW3	ADD A, @RW3+d16	SUB A,@RW3	SUB A, @RW3+d16	ADDC A,@RW3	ADDC A, @RW3+d16	CMP A,@RW3	CMP A, @RW3+d16	AND A,@RW3	AND A, @RW3+d16	OR A,@RW3	OR A, @RW3+d16	XOR A,@RW3	XOR A, @RW3+d16	DBNZ @RW3, rel	DBNZ @RW3 +d16, rel
+C	ADD A,@RW0+	ADD A, @RW0+RW7	SUB A,@RW0+	SUB A, @RW0+RW7	ADDC A,@RW0+	ADDC A, @RW0+RW7	CMP A,@RW0+	CMP A, @RW0+RW7	AND A,@RW0+	AND A, @RW0+RW7	OR A,@RW0+	OR A, @RW0+RW7	XOR A,@RW0+	XOR A, @RW0+RW7	DBNZ @RW0+, rel	DBNZ @RW0 +RW7, rel
+D	ADD A,@RW1+	ADD A, @RW1+RW7	SUB A,@RW1+	SUB A, @RW1+RW7	ADDC A,@RW1+	ADDC A, @RW1+RW7	CMP A,@RW1+	CMP A, @RW1+RW7	AND A,@RW1+	AND A, @RW1+RW7	OR A,@RW1+	OR A, @RW1+RW7	XOR A,@RW1+	XOR A, @RW1+RW7	DBNZ @RW1+, rel	DBNZ @RW1 +RW7, rel
+E	ADD A,@RW2+	ADD A, @PC+d16	SUB A,@RW2+	SUB A, @PC+d16	ADDC A,@RW2+	ADDC A, @PC+d16	CMP A,@RW2+	CMP A, @PC+d16	AND A,@RW2+	AND A, @PC+d16	OR A,@RW2+	OR A, @PC+d16,	XOR A,@RW2+	XOR A, @PC+d16	DBNZ @RW2+, rel	DBNZ @PC +d16, rel
+F	ADD A,@RW3+	ADD A, addr16	SUB A,@RW3+	SUB A, addr16	ADDC A,@RW3+	ADDC A, addr16	CMP A,@RW3+	CMP A, addr16	AND A,@RW3+	AND A, addr16	OR A,@RW3+	OR A, addr16	XOR A,@RW3+	XOR A, addr16	DBNZ @RW3+, rel	DBNZ addr16, rel

Table A-42. ea Instruction 6 (First Byte = 75_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADD R0, A	ADD @RW0+d8, A	SUB R0, A	SUB @RW0+d8, A	SUBC A, R0	SUBC A, @RW0+d8	NEG R0	NEG @RW0+d8	AND R0, A	AND @RW0+d8, A	OR R0, A	OR @RW0+d8, A	XOR R0, A	XOR @RW0+d8, A	NOT R0	NOT @RW0+d8
+1	ADD R1, A	ADD @RW1+d8, A	SUB R1, A	SUB @RW1+d8, A	SUBC A, R1	SUBC A, @RW1+d8	NEG R1	NEG @RW1+d8	AND R1, A	AND @RW1+d8, A	OR R1, A	OR @RW1+d8, A	XOR R1, A	XOR @RW1+d8, A	NOT R1	NOT @RW1+d8
+2	ADD R2, A	ADD @RW2+d8,A	SUB R2, A	SUB @RW2+d8,A	SUBC A, R2	SUBC A, @RW2+d8	NEG R2	NEG @RW2+d8	AND R2, A	AND @RW2+d8,A	OR R2, A	OR @RW2+d8,A	XOR R2, A	XOR @RW2+d8,A	NOT R2	NOT @RW2+d8
+3	ADD R3, A	ADD @RW3+d8, A	SUB R3, A	SUB @RW3+d8, A	SUBC A, R3	SUBC A, @RW3+d8	NEG R3	NEG @RW3+d8	AND R3, A	AND @RW3+d8, A	OR R3, A	OR @RW3+d8, A	XOR R3, A	XOR @RW3+d8, A	NOT R3	NOT @RW3+d8
+4	ADD R4, A	ADD @RW4+d8, A	SUB R4, A	SUB @RW4+d8, A	SUBC A, R4	SUBC A, @RW4+d8	NEG R4	NEG @RW4+d8	AND R4, A	AND @RW4+d8, A	OR R4, A	OR @RW4+d8, A	XOR R4, A	XOR @RW4+d8, A	NOT R4	NOT @RW4+d8
+5	ADD R5, A	ADD @RW5+d8, A	SUB R5, A	SUB @RW5+d8, A	SUBC A, R5	SUBC A, @RW5+d8	NEG R5	NEG @RW5+d8	AND R5, A	AND @RW5+d8, A	OR R5, A	OR @RW5+d8, A	XOR R5, A	XOR @RW5+d8, A	NOT R5	NOT @RW5+d8
+6	ADD R6, A	ADD @RW6+d8, A	SUB R6, A	SUB @RW6+d8, A	SUBC A, R6	SUBC A, @RW6+d8	NEG R6	NEG @RW6+d8	AND R6, A	AND @RW6+d8, A	OR R6, A	OR @RW6+d8, A	XOR R6, A	XOR @RW6+d8, A	NOT R6	NOT @RW6+d8
+7	ADD R7, A	ADD @RW7+d8, A	SUB R7, A	SUB @RW7+d8, A	SUBC A, R7	SUBC A, @RW7+d8	NEG R7	NEG @RW7+d8	AND R7, A	AND @RW7+d8, A	OR R7, A	OR @RW7+d8, A	XOR R7, A	XOR @RW7+d8, A	NOT R7	NOT @RW7+d8
+8	ADD @RW0, A	ADD @RW0+d16,A	SUB @RW0, A	SUB @RW0+d16,A	SUBC A, @RW0	SUBC A, @RW0+d16	NEG @RW0	NEG @RW0+d16	AND @RW0, A	AND @RW0+d16,A	OR @RW0, A	OR @RW0+d16,A	XOR @RW0, A	XOR @RW0+d16,A	NOT @RW0	NOT @RW0+d16
+9	ADD @RW1, A	ADD @R @RW1+d16,A	SUB @RW1, A	SUB @RW1+d16,A	SUBC A, @RW1	SUBC A, @RW1+d16	NEG @RW1	NEG @RW1+d16	AND @RW1, A	AND @RW1+d16,A	OR @RW1, A	OR @RW1+d16,A	XOR @RW1, A	XOR @RW1+d16,A	NOT @RW1	NOT @RW1+d16
+A	ADD @RW2, A	ADD @R @RW2+d16,A	SUB @RW2, A	SUB @RW2+d16,A	SUBC A, @RW2	SUBC A, @RW2+d16	NEG @RW2	NEG @RW2+d16	AND @RW2, A	AND @RW2+d16,A	OR @RW2, A	OR @RW2+d16,A	XOR @RW2, A	XOR @RW2+d16,A	NOT @RW2	NOT @RW2+d16
+B	ADD @RW3, A	ADD @R @RW3+d16,A	SUB @RW3, A	SUB @RW3+d16,A	SUBC A, @RW3	SUBC A, @RW3+d16	NEG @RW3	NEG @RW3+d16	AND @RW3, A	AND @RW3+d16,A	OR @RW3, A	OR @RW3+d16,A	XOR @RW3, A	XOR @RW3+d16,A	NOT @RW3	NOT @RW3+d16
+C	ADD @RW0+, A	ADD @R @RW0+RW7,A	SUB @RW0+, A	SUB @RW0+RW7,A	SUBC A,@RW0+	SUBC A, @RW0+RW7	NEG @RW0+	NEG @RW0+RW7	AND @RW0+, A	AND @RW0+RW7,A	OR @RW0+, A	OR @RW0+RW7,A	XOR @RW0+, A	XOR @RW0+RW7,A	NOT @RW0+	NOT @RW0+RW7
+D	ADD @RW1+, A	ADD @R @RW1+RW7,A	SUB @RW1+, A	SUB @RW1+RW7,A	SUBC A,@RW1+	SUBC A, @RW1+RW7	NEG @RW1+	NEG @RW1+RW7	AND @RW1+, A	AND @RW1+RW7,A	OR @RW1+, A	OR @RW1+RW7,A	XOR @RW1+, A	XOR @RW1+RW7,A	NOT @RW1+	NOT @RW1+RW7
+E	ADD @RW2+, A	ADD @PC+d16,A	SUB @RW2+, A	SUB @PC+d16,A	SUBC A,@RW2+	SUBC A, @PC+d16	NEG @RW2+	NEG @PC+d16	AND @RW2+, A	AND @PC+d16,A	OR @RW2+, A	OR @PC+d16,A	XOR @RW2+, A	XOR @PC+d16,A	NOT @RW2+	NOT @PC+d16
+F	ADD @RW3+, A	ADD addr16, A	SUB @RW3+, A	SUB addr16, A	SUBC A,@RW3+	SUBC A, addr16	NEG @RW3+	NEG addr16	AND @RW3+, A	AND addr16, A	OR @RW3+, A	OR addr16, A	XOR @RW3+, A	XOR addr16, A	NOT @RW3+	NOT addr16

Table A-43. ea Instruction 7 (First Byte = 76_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADDW A, RW0	ADDW A, @RW0+d8	SUBW A, RW0	SUBW A, @RW0+d8	ADDCW A, RW0	ADDCW A, @RW0+d8	CMPW A, RW0	CMPW A, @RW0+d8	ANDW A, RW0	ANDW A, @RW0+d8	ORW A, RW0	ORW A, @RW0+d8	XORW A, RW0	XORW A, @RW0+d8	DWBNZ RW0, rel	DWBNZ @RW0 +d8,rel
+1	ADDW A, RW1	ADDW A, @RW1+d8	SUBW A, RW1	SUBW A, @RW1+d8	ADDCW A, RW1	ADDCW A, @RW1+d8	CMPW A, RW1	CMPW A, @RW1+d8	ANDW A, RW1	ANDW A, @RW1+d8	ORW A, RW1	ORW A, @RW1+d8	XORW A, RW1	XORW A, @RW1+d8	DWBNZ RW1, rel	DWBNZ @RW1 +d8,rel
+2	ADDW A, RW2	ADDW A, @RW2+d8	SUBW A, RW2	SUBW A, @RW2+d8	ADDCW A, RW2	ADDCW A, @RW2+d8	CMPW A, RW2	CMPW A, @RW2+d8	ANDW A, RW2	ANDW A, @RW2+d8	ORW A, RW2	ORW A, @RW2+d8	XORW A, RW2	XORW A, @RW2+d8	DWBNZ RW2, rel	DWBNZ @RW2 +d8,rel
+3	ADDW A, RW3	ADDW A, @RW3+d8	SUBW A, RW3	SUBW A, @RW3+d8	ADDCW A, RW3	ADDCW A, @RW3+d8	CMPW A, RW3	CMPW A, @RW3+d8	ANDW A, RW3	ANDW A, @RW3+d8	ORW A, RW3	ORW A, @RW3+d8	XORW A, RW3	XORW A, @RW3+d8	DWBNZ RW3, rel	DWBNZ @RW3 +d8,rel
+4	ADDW A, RW4	ADDW A, @RW4+d8	SUBW A, RW4	SUBW A, @RW4+d8	ADDCW A, RW4	ADDCW A, @RW4+d8	CMPW A, RW4	CMPW A, @RW4+d8	ANDW A, RW4	ANDW A, @RW4+d8	ORW A, RW4	ORW A, @RW4+d8	XORW A, RW4	XORW A, @RW4+d8	DWBNZ RW4, rel	DWBNZ @RW4 +d8,rel
+5	ADDW A, RW5	ADDW A, @RW5+d8	SUBW A, RW5	SUBW A, @RW5+d8	ADDCW A, RW5	ADDCW A, @RW5+d8	CMPW A, RW5	CMPW A, @RW5+d8	ANDW A, RW5	ANDW A, @RW5+d8	ORW A, RW5	ORW A, @RW5+d8	XORW A, RW5	XORW A, @RW5+d8	DWBNZ RW5, rel	DWBNZ @RW5 +d8,rel
+6	ADDW A, RW6	ADDW A, @RW6+d8	SUBW A, RW6	SUBW A, @RW6+d8	ADDCW A, RW6	ADDCW A, @RW6+d8	CMPW A, RW6	CMPW A, @RW6+d8	ANDW A, RW6	ANDW A, @RW6+d8	ORW A, RW6	ORW A, @RW6+d8	XORW A, RW6	XORW A, @RW6+d8	DWBNZ RW6, rel	DWBNZ @RW6 +d8,rel
+7	ADDW A, RW7	ADDW A, @RW7+d8	SUBW A, RW7	SUBW A, @RW7+d8	ADDCW A, RW7	ADDCW A, @RW7+d8	CMPW A, RW7	CMPW A, @RW7+d8	ANDW A, RW7	ANDW A, @RW7+d8	ORW A, RW7	ORW A, @RW7+d8	XORW A, RW7	XORW A, @RW7+d8	DWBNZ RW7, rel	DWBNZ @RW7 +d8,rel
+8	ADDW A,@RW0	ADDW A, @RW0+d16	SUBW A,@RW0	SUBW A, @RW0+d16	ADDCW A,@RW0	ADDCW A, @RW0+d16	CMPW A,@RW0	CMPW A, @RW0+d16	ANDW A,@RW0	ANDW A, @RW0+d16	ORW A,@RW0	ORW A, @RW0+d16	XORW A,@RW0	XORW A, @RW0+d16	DWBNZ @RW0, rel	DWBNZ @RW0 +d16,rel
+9	ADDW A,@RW1	ADDW A, @RW1+d16	SUBW A,@RW1	SUBW A, @RW1+d16	ADDCW A,@RW1	ADDCW A, @RW1+d16	CMPW A,@RW1	CMPW A, @RW1+d16	ANDW A,@RW1	ANDW A, @RW1+d16	ORW A,@RW1	ORW A, @RW1+d16	XORW A,@RW1	XORW A, @RW1+d16	DWBNZ @RW1, rel	DWBNZ @RW1 +d16,rel
+A	ADDW A,@RW2	ADDW A, @RW2+d16	SUBW A,@RW2	SUBW A, @RW2+d16	ADDCW A,@RW2	ADDCW A, @RW2+d16	CMPW A,@RW2	CMPW A, @RW2+d16	ANDW A,@RW2	ANDW A, @RW2+d16	ORW A,@RW2	ORW A, @RW2+d16	XORW A,@RW2	XORW A, @RW2+d16	DWBNZ @RW2, rel	DWBNZ @RW2 +d16,rel
+B	ADDW A,@RW3	ADDW A, @RW3+d16	SUBW A,@RW3	SUBW A, @RW3+d16	ADDCW A,@RW3	ADDCW A, @RW3+d16	CMPW A,@RW3	CMPW A, @RW3+d16	ANDW A,@RW3	ANDW A, @RW3+d16	ORW A,@RW3	ORW A, @RW3+d16	XORW A,@RW3	XORW A, @RW3+d16	DWBNZ @RW3, rel	DWBNZ @RW3 +d16,rel
+C	ADDW A,@RW0+	ADDW A, @RW0+RW7	SUBW A,@RW0+	SUBW A, @RW0+RW7	ADDCW A,@RW0+	ADDCW A, @RW0+RW7	CMPW A,@RW0+	CMPW A, @RW0+RW7	ANDW A,@RW0+	ANDW A, @RW0+RW7	ORW A,@RW0+	ORW A, @RW0+RW7	XORW A,@RW0+	XORW A, @RW0+RW7	DWBNZ @RW0+, rel	DWBNZ @RW0 +RW7,rel
+D	ADDW A,@RW1+	ADDW A, @RW1+RW7	SUBW A,@RW1+	SUBW A, @RW1+RW7	ADDCW A,@RW1+	ADDCW A, @RW1+RW7	CMPW A,@RW1+	CMPW A, @RW1+RW7	ANDW A,@RW1+	ANDW A, @RW1+RW7	ORW A,@RW1+	ORW A, @RW1+RW7	XORW A,@RW1+	XORW A, @RW1+RW7	DWBNZ @RW1+, rel	DWBNZ @RW1 +RW7,rel
+E	ADDW A,@RW2+	ADDW A, @PC+d16	SUBW A,@RW2+	SUBW A, @PC+d16	ADDCW A,@RW2+	ADDCW A, @PC+d16	CMPW A,@RW2+	CMPW A, @PC+d16	ANDW A,@RW2+	ANDW A, @PC+d16	ORW A,@RW2+	ORW A, @PC+d16,	XORW A,@RW2+	XORW A, @PC+d16	DWBNZ @RW2+, rel	DWBNZ @PC +d16,rel
+F	ADDW A,@RW3+	ADDW A, addr16	SUBW A,@RW3+	SUBW A, addr16	ADDCW A,@RW3+	ADDCW A, addr16	CMPW A,@RW3+	CMPW A, addr16	ANDW A,@RW3+	ANDW A, addr16	ORW A,@RW3+	ORW A, addr16	XORW A,@RW3+	XORW A, addr16	DWBNZ @RW3+, rel	DWBNZ addr16, rel

Table A-44. ea Instruction 8 (First Byte = 77_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	ADDW RW0, A	ADDW @RW0+d8, A	SUBW RW0, A	SUBW @RW0+d8, A	SUBCW A, RW0	SUBCW A, @RW0+d8	NEGW RW0	NEGW @RW0+d8	ANDW RW0, A	ANDW @RW0+d8, A	ORW RW0, A	ORW @RW0+d8, A	XORW RW0, A	XORW @RW0+d8, A	NOTW RW0	NOTW @RW0+d8
+1	ADDW RW1, A	ADDW @RW1+d8, A	SUBW RW1, A	SUBW @RW1+d8, A	SUBCW A, RW1	SUBCW A, @RW1+d8	NEGW RW1	NEGW @RW1+d8	ANDW RW1, A	ANDW @RW1+d8, A	ORW RW1, A	ORW @RW1+d8, A	XORW RW1, A	XORW @RW1+d8, A	NOTW RW1	NOTW @RW1+d8
+2	ADDW RW2, A	ADDW @RW2+d8,A	SUBW RW2, A	SUBW @RW2+d8,A	SUBCW A, RW2	SUBCW A, @RW2+d8	NEGW RW2	NEGW @RW2+d8	ANDW RW2, A	ANDW @RW2+d8,A	ORW RW2, A	ORW @RW2+d8,A	XORW RW2, A	XORW @RW2+d8,A	NOTW RW2	NOTW @RW2+d8
+3	ADDW RW3, A	ADDW @RW3+d8, A	SUBW RW3, A	SUBW @RW3+d8, A	SUBCW A, RW3	SUBCW A, @RW3+d8	NEGW RW3	NEGW @RW3+d8	ANDW RW3, A	ANDW @RW3+d8, A	ORW RW3, A	ORW @RW3+d8, A	XORW RW3, A	XORW @RW3+d8, A	NOTW RW3	NOTW @RW3+d8
+4	ADDW RW4, A	ADDW @RW4+d8, A	SUBW RW4, A	SUBW @RW4+d8, A	SUBCW A, RW4	SUBCW A, @RW4+d8	NEGW RW4	NEGW @RW4+d8	ANDW RW4, A	ANDW @RW4+d8, A	ORW RW4, A	ORW @RW4+d8, A	XORW RW4, A	XORW @RW4+d8, A	NOTW RW4	NOTW @RW4+d8
+5	ADDW RW5, A	ADDW @RW5+d8, A	SUBW RW5, A	SUBW @RW5+d8, A	SUBCW A, RW5	SUBCW A, @RW5+d8	NEGW RW5	NEGW @RW5+d8	ANDW RW5, A	ANDW @RW5+d8, A	ORW RW5, A	ORW @RW5+d8, A	XORW RW5, A	XORW @RW5+d8, A	NOTW RW5	NOTW @RW5+d8
+6	ADDW RW6, A	ADDW @RW6+d8, A	SUBW RW6, A	SUBW @RW6+d8, A	SUBCW A, RW6	SUBCW A, @RW6+d8	NEGW RW6	NEGW @RW6+d8	ANDW RW6, A	ANDW @RW6+d8, A	ORW RW6, A	ORW @RW6+d8, A	XORW RW6, A	XORW @RW6+d8, A	NOTW RW6	NOTW @RW6+d8
+7	ADDW RW7, A	ADDW @RW7+d8, A	SUBW RW7, A	SUBW @RW7+d8, A	SUBCW A, RW7	SUBCW A, @RW7+d8	NEGW RW7	NEGW @RW7+d8	ANDW RW7, A	ANDW @RW7+d8, A	ORW RW7, A	ORW @RW7+d8, A	XORW RW7, A	XORW @RW7+d8, A	NOTW RW7	NOTW @RW7+d8
+8	ADDW @RW0, A	ADDW @RW0+d16,A	SUBW @RW0, A	SUBW @RW0+d16,A	SUBCW A, @RW0	SUBCW A, @RW0+d16	NEGW @RW0	NEGW @RW0+d16	ANDW @RW0, A	ANDW @RW0+d16,A	ORW @RW0, A	ORW @RW0+d16,A	XORW @RW0, A	XORW @RW0+d16,A	NOTW @RW0	NOTW @RW0+d16
+9	ADDW @RW1, A	ADDW @RW1+d16,A	SUBW @RW1, A	SUBW @RW1+d16,A	SUBCW A, @RW1	SUBCW A, @RW1+d16	NEGW @RW1	NEGW @RW1+d16	ANDW @RW1, A	ANDW @RW1+d16,A	ORW @RW1, A	ORW @RW1+d16,A	XORW @RW1, A	XORW @RW1+d16,A	NOTW @RW1	NOTW @RW1+d16
+A	ADDW @RW2, A	ADDW @RW2+d16,A	SUBW @RW2, A	SUBW @RW2+d16,A	SUBCW A, @RW2	SUBCW A, @RW2+d16	NEGW @RW2	NEGW @RW2+d16	ANDW @RW2, A	ANDW @RW2+d16,A	ORW @RW2, A	ORW @RW2+d16,A	XORW @RW2, A	XORW @RW2+d16,A	NOTW @RW2	NOTW @RW2+d16
+B	ADDW @RW3, A	ADDW @RW3+d16,A	SUBW @RW3, A	SUBW @RW3+d16,A	SUBCW A, @RW3	SUBCW A, @RW3+d16	NEGW @RW3	NEGW @RW3+d16	ANDW @RW3, A	ANDW @RW3+d16,A	ORW @RW3, A	ORW @RW3+d16,A	XORW @RW3, A	XORW @RW3+d16,A	NOTW @RW3	NOTW @RW3+d16
+C	ADDW @RW0+, A	ADDW @RW0+RW7,A	SUBW @RW0+, A	SUBW @RW0+RW7,A	SUBCW A,@RW0+	SUBCW A, @RW0+RW7	NEGW @RW0+	NEGW @RW0+RW7	ANDW @RW0+, A	ANDW @RW0+RW7,A	ORW @RW0+, A	ORW @RW0+RW7,A	XORW @RW0+, A	XORW @RW0+RW7,A	NOTW @RW0+	NOTW @RW0+RW7
+D	ADDW @RW1+, A	ADDW @RW1+RW7,A	SUBW @RW1+, A	SUBW @RW1+RW7,A	SUBCW A,@RW1+	SUBCW A, @RW1+RW7	NEGW @RW1+	NEGW @RW1+RW7	ANDW @RW1+, A	ANDW @RW1+RW7,A	ORW @RW1+, A	ORW @RW1+RW7,A	XORW @RW1+, A	XORW @RW1+RW7,A	NOTW @RW1+	NOTW @RW1+RW7
+E	ADDW @RW2+, A	ADDW @PC+d16,A	SUBW @RW2+, A	SUBW @PC+d16,A	SUBCW A,@RW2+	SUBCW A, @PC+d16	NEGW @RW2+	NEGW @PC+d16	ANDW @RW2+, A	ANDW @PC+d16,A	ORW @RW2+, A	ORW @PC+d16,A	XORW @RW2+, A	XORW @PC+d16,A	NOTW @RW2+	NOTW @PC+d16
+F	ADDW @RW3+, A	ADDW addr16, A	SUBW @RW3+, A	SUBW addr16, A	SUBCW A,@RW3+	SUBCW A, addr16	NEGW @RW3+	NEGW addr16	ANDW @RW3+, A	ANDW addr16, A	ORW @RW3+, A	ORW addr16, A	XORW @RW3+, A	XORW addr16, A	NOTW @RW3+	NOTW addr16

Table A-45. ea Instruction 9 (First Byte = 78_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MULU A, R0	MULU A, @RW0+d8	MULUW A, RW0	MULUW A, @RW0+d8	MUL A, R0	MUL A, @RW0+d8	MULW A, RW0	MULW A, @RW0+d8	DIVU A, R0	DIVU A, @RW0+d8	DIVUW A, RW0	DIVUW A, @RW0+d8	DIV A, R0	DIV A, @RW0+d8	DIVW A, RW0	DIVW A, @RW0+d8
+1	MULU A, R1	MULU A, @RW1+d8	MULUW A, RW1	MULUW A, @RW1+d8	MUL A, R1	MUL A, @RW1+d8	MULW A, RW1	MULW A, @RW1+d8	DIVU A, R1	DIVU A, @RW1+d8	DIVUW A, RW1	DIVUW A, @RW1+d8	DIV A, R1	DIV A, @RW1+d8	DIVW A, RW1	DIVW A, @RW1+d8
+2	MULU A, R2	MULU A, @RW2+d8	MULUW A, RW2	MULUW A, @RW2+d8	MUL A, R2	MUL A, @RW2+d8	MULW A, RW2	MULW A, @RW2+d8	DIVU A, R2	DIVU A, @RW2+d8	DIVUW A, RW2	DIVUW A, @RW2+d8	DIV A, R2	DIV A, @RW2+d8	DIVW A, RW2	DIVW A, @RW2+d8
+3	MULU A, R3	MULU A, @RW3+d8	MULUW A, RW3	MULUW A, @RW3+d8	MUL A, R3	MUL A, @RW3+d8	MULW A, RW3	MULW A, @RW3+d8	DIVU A, R3	DIVU A, @RW3+d8	DIVUW A, RW3	DIVUW A, @RW3+d8	DIV A, R3	DIV A, @RW3+d8	DIVW A, RW3	DIVW A, @RW3+d8
+4	MULU A, R4	MULU A, @RW4+d8	MULUW A, RW4	MULUW A, @RW4+d8	MUL A, R4	MUL A, @RW4+d8	MULW A, RW4	MULW A, @RW4+d8	DIVU A, R4	DIVU A, @RW4+d8	DIVUW A, RW4	DIVUW A, @RW4+d8	DIV A, R4	DIV A, @RW4+d8	DIVW A, RW4	DIVW A, @RW4+d8
+5	MULU A, R5	MULU A, @RW5+d8	MULUW A, RW5	MULUW A, @RW5+d8	MUL A, R5	MUL A, @RW5+d8	MULW A, RW5	MULW A, @RW5+d8	DIVU A, R5	DIVU A, @RW5+d8	DIVUW A, RW5	DIVUW A, @RW5+d8	DIV A, R5	DIV A, @RW5+d8	DIVW A, RW5	DIVW A, @RW5+d8
+6	MULU A, R6	MULU A, @RW6+d8	MULUW A, RW6	MULUW A, @RW6+d8	MUL A, R6	MUL A, @RW6+d8	MULW A, RW6	MULW A, @RW6+d8	DIVU A, R6	DIVU A, @RW6+d8	DIVUW A, RW6	DIVUW A, @RW6+d8	DIV A, R6	DIV A, @RW6+d8	DIVW A, RW6	DIVW A, @RW6+d8
+7	MULU A, R7	MULU A, @RW7+d8	MULUW A, RW7	MULUW A, @RW7+d8	MUL A, R7	MUL A, @RW7+d8	MULW A, RW7	MULW A, @RW7+d8	DIVU A, R7	DIVU A, @RW7+d8	DIVUW A, RW7	DIVUW A, @RW7+d8	DIV A, R7	DIV A, @RW7+d8	DIVW A, RW7	DIVW A, @RW7+d8
+8	MULU A,@RW0	MULU A, @RW0+d16	MULUW A,@RW0	MULUW A, @RW0+d16	MUL A, @RW0	MUL A, @RW0+d16	MULW A,@RW0	MULW A, @RW0+d16	DIVU A,@RW0	DIVU A, @RW0+d16	DIVUW A,@RW0	DIVUW A, @RW0+d16	DIV A,@RW0	DIV A, @RW0+d16	DIVW A,@RW0	DIVW A, @RW0+d16
+9	MULU A,@RW1	MULU A, @RW1+d16	MULUW A,@RW1	MULUW A, @RW1+d16	MUL A, @RW1	MUL A, @RW1+d16	MULW A,@RW1	MULW A, @RW1+d16	DIVU A,@RW1	DIVU A, @RW1+d16	DIVUW A,@RW1	DIVUW A, @RW1+d16	DIV A,@RW1	DIV A, @RW1+d16	DIVW A,@RW1	DIVW A, @RW1+d16
+A	MULU A,@RW2	MULU A, @RW2+d16	MULUW A,@RW2	MULUW A, @RW2+d16	MUL A, @RW2	MUL A, @RW2+d16	MULW A,@RW2	MULW A, @RW2+d16	DIVU A,@RW2	DIVU A, @RW2+d16	DIVUW A,@RW2	DIVUW A, @RW2+d16	DIV A,@RW2	DIV A, @RW2+d16	DIVW A,@RW2	DIVW A, @RW2+d16
+B	MULU A,@RW3	MULU A, @RW3+d16	MULUW A,@RW3	MULUW A, @RW3+d16	MUL A, @RW3	MUL A, @RW3+d16	MULW A,@RW3	MULW A, @RW3+d16	DIVU A,@RW3	DIVU A, @RW3+d16	DIVUW A,@RW3	DIVUW A, @RW3+d16	DIV A,@RW3	DIV A, @RW3+d16	DIVW A,@RW3	DIVW A, @RW3+d16
+C	MULU A,@RW0+	MULU A, @RW0+RW7	MULUW A,@RW0+	MULUW A, @RW0+RW7	MUL A,@RW0+	MUL A, @RW0+RW7	MULW A,@RW0+	MULW A, @RW0+RW7	DIVU A,@RW0+	DIVU A, @RW0+RW7	DIVUW A,@RW0+	DIVUW A, @RW0+RW7	DIV A,@RW0+	DIV A, @RW0+RW7	DIVW A,@RW0+	DIVW A, @RW0+RW7
+D	MULU A,@RW1+	MULU A, @RW1+RW7	MULUW A,@RW1+	MULUW A, @RW1+RW7	MUL A,@RW1+	MUL A, @RW1+RW7	MULW A,@RW1+	MULW A, @RW1+RW7	DIVU A,@RW1+	DIVU A, @RW1+RW7	DIVUW A,@RW1+	DIVUW A, @RW1+RW7	DIV A,@RW1+	DIV A, @RW1+RW7	DIVW A,@RW1+	DIVW A, @RW1+RW7
+E	MULU A,@RW2+	MULU A, @PC+d16	MULUW A,@RW2+	MULUW A, @PC+d16	MUL A,@RW2+	MUL A, @PC+d16	MULW A,@RW2+	MULW A, @PC+d16	DIVU A,@RW2+	DIVU A, @PC+d16	DIVUW A,@RW2+	DIVUW A, @PC+d16	DIV A,@RW2+	DIV A, @PC+d16	DIVW A,@RW2+	DIVW A, @PC+d16
+F	MULU A, @RW3+	MULU A, addr16	MULUW A, @RW3+	MULUW A, addr16	MUL A, @RW3+	MUL A, addr16	MULW A,@RW3+	MULW A, addr16	DIVU A, @RW3+	DIVU A, addr16	DIVUW A, @RW3+	DIVUW A, addr16	DIV A, @RW3+	DIV A, addr16	DIVW A, @RW3+	DIVW A, addr16

Table A-46. MOVEA RWi, ea Instruction (First Byte = 79_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVEA RW0,RW0	MOVEA RW0, @RW0+d8	MOVEA RW1,RW0	MOVEA RW1, @RW0+d8	MOVEA RW2,RW0	MOVEA RW2, @RW0+d8	MOVEA RW3,RW0	MOVEA RW3, @RW0+d8	MOVEA RW4,RW0	MOVEA RW4, @RW0+d8	MOVEA RW5,RW0	MOVEA RW5, @RW0+d8	MOVEA RW6,RW0	MOVEA RW6, @RW0+d8	MOVEA RW7,RW0	MOVEA RW7, @RW0+d8
+1	MOVEA RW0,RW1	MOVEA RW0, @RW1+d8	MOVEA RW1,RW1	MOVEA RW1, @RW1+d8	MOVEA RW2,RW1	MOVEA RW2, @RW1+d8	MOVEA RW3,RW1	MOVEA RW3, @RW1+d8	MOVEA RW4,RW1	MOVEA RW4, @RW1+d8	MOVEA RW5,RW1	MOVEA RW5, @RW1+d8	MOVEA RW6,RW1	MOVEA RW6, @RW1+d8	MOVEA RW7,RW1	MOVEA RW7, @RW1+d8
+2	MOVEA RW0,RW2	MOVEA RW0, @RW2+d8	MOVEA RW1,RW2	MOVEA RW1, @RW2+d8	MOVEA RW2,RW2	MOVEA RW2, @RW2+d8	MOVEA RW3,RW2	MOVEA RW3, @RW2+d8	MOVEA RW4,RW2	MOVEA RW4, @RW2+d8	MOVEA RW5,RW2	MOVEA RW5, @RW2+d8	MOVEA RW6,RW2	MOVEA RW6, @RW2+d8	MOVEA RW7,RW2	MOVEA RW7, @RW2+d8
+3	MOVEA RW0,RW3	MOVEA RW0, @RW3+d8	MOVEA RW1,RW3	MOVEA RW1, @RW3+d8	MOVEA RW2,RW3	MOVEA RW2, @RW3+d8	MOVEA RW3,RW3	MOVEA RW3, @RW3+d8	MOVEA RW4,RW3	MOVEA RW4, @RW3+d8	MOVEA RW5,RW3	MOVEA RW5, @RW3+d8	MOVEA RW6,RW3	MOVEA RW6, @RW3+d8	MOVEA RW7,RW3	MOVEA RW7, @RW3+d8
+4	MOVEA RW0,RW4	MOVEA RW0, @RW4+d8	MOVEA RW1,RW4	MOVEA RW1, @RW4+d8	MOVEA RW2,RW4	MOVEA RW2, @RW4+d8	MOVEA RW3,RW4	MOVEA RW3, @RW4+d8	MOVEA RW4,RW4	MOVEA RW4, @RW4+d8	MOVEA RW5,RW4	MOVEA RW5, @RW4+d8	MOVEA RW6,RW4	MOVEA RW6, @RW4+d8	MOVEA RW7,RW4	MOVEA RW7, @RW4+d8
+5	MOVEA RW0,RW5	MOVEA RW0, @RW5+d8	MOVEA RW1,RW5	MOVEA RW1, @RW5+d8	MOVEA RW2,RW5	MOVEA RW2, @RW5+d8	MOVEA RW3,RW5	MOVEA RW3, @RW5+d8	MOVEA RW4,RW5	MOVEA RW4, @RW5+d8	MOVEA RW5,RW5	MOVEA RW5, @RW5+d8	MOVEA RW6,RW5	MOVEA RW6, @RW5+d8	MOVEA RW7,RW5	MOVEA RW7, @RW5+d8
+6	MOVEA RW0,RW6	MOVEA RW0, @RW6+d8	MOVEA RW1,RW6	MOVEA RW1, @RW6+d8	MOVEA RW2,RW6	MOVEA RW2, @RW6+d8	MOVEA RW3,RW6	MOVEA RW3, @RW6+d8	MOVEA RW4,RW6	MOVEA RW4, @RW6+d8	MOVEA RW5,RW6	MOVEA RW5, @RW6+d8	MOVEA RW6,RW6	MOVEA RW6, @RW6+d8	MOVEA RW7,RW6	MOVEA RW7, @RW6+d8
+7	MOVEA RW0,RW7	MOVEA RW0, @RW7+d8	MOVEA RW1,RW7	MOVEA RW1, @RW7+d8	MOVEA RW2,RW7	MOVEA RW2, @RW7+d8	MOVEA RW3,RW7	MOVEA RW3, @RW7+d8	MOVEA RW4,RW7	MOVEA RW4, @RW7+d8	MOVEA RW5,RW7	MOVEA RW5, @RW7+d8	MOVEA RW6,RW7	MOVEA RW6, @RW7+d8	MOVEA RW7,RW7	MOVEA RW7, @RW7+d8
+8	MOVEA RW0, @RW0	MOVEA RW0, @RW0+d16	MOVEA RW1, @RW0	MOVEA RW1, @RW0+d16	MOVEA RW2, @RW0	MOVEA RW2, @RW0+d16	MOVEA RW3, @RW0	MOVEA RW3, @RW0+d16	MOVEA RW4, @RW0	MOVEA RW4, @RW0+d16	MOVEA RW5, @RW0	MOVEA RW5, @RW0+d16	MOVEA RW6, @RW0	MOVEA RW6, @RW0+d16	MOVEA RW7, @RW0	MOVEA RW7, @RW0+d16
+9	MOVEA RW0, @RW1	MOVEA RW0, @RW1+d16	MOVEA RW1, @RW1	MOVEA RW1, @RW1+d16	MOVEA RW2, @RW1	MOVEA RW2, @RW1+d16	MOVEA RW3, @RW1	MOVEA RW3, @RW1+d16	MOVEA RW4, @RW1	MOVEA RW4, @RW1+d16	MOVEA RW5, @RW1	MOVEA RW5, @RW1+d16	MOVEA RW6, @RW1	MOVEA RW6, @RW1+d16	MOVEA RW7, @RW1	MOVEA RW7, @RW1+d16
+A	MOVEA RW0, @RW2	MOVEA RW0, @RW2+d16	MOVEA RW1, @RW2	MOVEA RW1, @RW2+d16	MOVEA RW2, @RW2	MOVEA RW2, @RW2+d16	MOVEA RW3, @RW2	MOVEA RW3, @RW2+d16	MOVEA RW4, @RW2	MOVEA RW4, @RW2+d16	MOVEA RW5, @RW2	MOVEA RW5, @RW2+d16	MOVEA RW6, @RW2	MOVEA RW6, @RW2+d16	MOVEA RW7, @RW2	MOVEA RW7, @RW2+d16
+B	MOVEA RW0, @RW3	MOVEA RW0, @RW3+d16	MOVEA RW1, @RW3	MOVEA RW1, @RW3+d16	MOVEA RW2, @RW3	MOVEA RW2, @RW3+d16	MOVEA RW3, @RW3	MOVEA RW3, @RW3+d16	MOVEA RW4, @RW3	MOVEA RW4, @RW3+d16	MOVEA RW5, @RW3	MOVEA RW5, @RW3+d16	MOVEA RW6, @RW3	MOVEA RW6, @RW3+d16	MOVEA RW7, @RW3	MOVEA RW7, @RW3+d16
+C	MOVEA RW0, @RW0+	MOVEA RW0, @RW0+RW7	MOVEA RW1, @RW0+	MOVEA RW1, @RW0+RW7	MOVEA RW2, @RW0+	MOVEA RW2, @RW0+RW7	MOVEA RW3, @RW0+	MOVEA RW3, @RW0+RW7	MOVEA RW4, @RW0+	MOVEA RW4, @RW0+RW7	MOVEA RW5, @RW0+	MOVEA RW5, @RW0+RW7	MOVEA RW6, @RW0+	MOVEA RW6, @RW0+RW7	MOVEA RW7, @RW0+	MOVEA RW7, @RW0+RW7
+D	MOVEA RW0, @RW1+	MOVEA RW0, @RW1+RW7	MOVEA RW1, @RW1+	MOVEA RW1, @RW1+RW7	MOVEA RW2, @RW1+	MOVEA RW2, @RW1+RW7	MOVEA RW3, @RW1+	MOVEA RW3, @RW1+RW7	MOVEA RW4, @RW1+	MOVEA RW4, @RW1+RW7	MOVEA RW5, @RW1+	MOVEA RW5, @RW1+RW7	MOVEA RW6, @RW1+	MOVEA RW6, @RW1+RW7	MOVEA RW7, @RW1+	MOVEA RW7, @RW1+RW7
+E	MOVEA RW0, @RW2+	MOVEA RW0, @PC+d16	MOVEA RW1, @RW2+	MOVEA RW1, @PC+d16	MOVEA RW2, @RW2+	MOVEA RW2, @PC+d16	MOVEA RW3, @RW2+	MOVEA RW3, @PC+d16	MOVEA RW4, @RW2+	MOVEA RW4, @PC+d16	MOVEA RW5, @RW2+	MOVEA RW5, @PC+d16	MOVEA RW6, @RW2+	MOVEA RW6, @PC+d16	MOVEA RW7, @RW2+	MOVEA RW7, @PC+d16
+F	MOVEA RW0, @RW3+	MOVEA RW0, addr16	MOVEA RW1, @RW3+	MOVEA RW1, addr16	MOVEA RW2, @RW3+	MOVEA RW2, addr16	MOVEA RW3, @RW3+	MOVEA RW3, addr16	MOVEA RW4, @RW3+	MOVEA RW4, addr16	MOVEA RW5, @RW3+	MOVEA RW5, addr16	MOVEA RW6, @RW3+	MOVEA RW6, addr16	MOVEA RW7, @RW3+	MOVEA RW7, addr16

Table A-47. MOV Ri, ea Instruction (First Byte = $7A_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOV R0, R0	MOV R0, @RW0+d8	MOV R1, R0	MOV R1, @RW0+d8	MOV R2, R0	MOV R2, @RW0+d8	MOV R3, R0	MOV R3, @RW0+d8	MOV R4, R0	MOV R4, @RW0+d8	MOV R5, R0	MOV R5, @RW0+d8	MOV R6, R0	MOV R6, @RW0+d8	MOV R7, R0	MOV R7, @RW0+d8
+1	MOV R0, R1	MOV R0, @RW1+d8	MOV R1, R1	MOV R1, @RW1+d8	MOV R2, R1	MOV R2, @RW1+d8	MOV R3, R1	MOV R3, @RW1+d8	MOV R4, R1	MOV R4, @RW1+d8	MOV R5, R1	MOV R5, @RW1+d8	MOV R6, R1	MOV R6, @RW1+d8	MOV R7, R1	MOV R7, @RW1+d8
+2	MOV R0, R2	MOV R0, @RW2+d8	MOV R1, R2	MOV R1, @RW2+d8	MOV R2, R2	MOV R2, @RW2+d8	MOV R3, R2	MOV R3, @RW2+d8	MOV R4, R2	MOV R4, @RW2+d8	MOV R5, R2	MOV R5, @RW2+d8	MOV R6, R2	MOV R6, @RW2+d8	MOV R7, R2	MOV R7, @RW2+d8
+3	MOV R0, R3	MOV R0, @RW3+d8	MOV R1, R3	MOV R1, @RW3+d8	MOV R2, R3	MOV R2, @RW3+d8	MOV R3, R3	MOV R3, @RW3+d8	MOV R4, R3	MOV R4, @RW3+d8	MOV R5, R3	MOV R5, @RW3+d8	MOV R6, R3	MOV R6, @RW3+d8	MOV R7, R3	MOV R7, @RW3+d8
+4	MOV R0, R4	MOV R0, @RW4+d8	MOV R1, R4	MOV R1, @RW4+d8	MOV R2, R4	MOV R2, @RW4+d8	MOV R3, R4	MOV R3, @RW4+d8	MOV R4, R4	MOV R4, @RW4+d8	MOV R5, R4	MOV R5, @RW4+d8	MOV R6, R4	MOV R6, @RW4+d8	MOV R7, R4	MOV R7, @RW4+d8
+5	MOV R0, R5	MOV R0, @RW5+d8	MOV R1, R5	MOV R1, @RW5+d8	MOV R2, R5	MOV R2, @RW5+d8	MOV R3, R5	MOV R3, @RW5+d8	MOV R4, R5	MOV R4, @RW5+d8	MOV R5, R5	MOV R5, @RW5+d8	MOV R6, R5	MOV R6, @RW5+d8	MOV R7, R5	MOV R7, @RW5+d8
+6	MOV R0, R6	MOV R0, @RW6+d8	MOV R1, R6	MOV R1, @RW6+d8	MOV R2, R6	MOV R2, @RW6+d8	MOV R3, R6	MOV R3, @RW6+d8	MOV R4, R6	MOV R4, @RW6+d8	MOV R5, R6	MOV R5, @RW6+d8	MOV R6, R6	MOV R6, @RW6+d8	MOV R7, R6	MOV R7, @RW6+d8
+7	MOV R0, R7	MOV R0, @RW7+d8	MOV R1, R7	MOV R1, @RW7+d8	MOV R2, R7	MOV R2, @RW7+d8	MOV R3, R7	MOV R3, @RW7+d8	MOV R4, R7	MOV R4, @RW7+d8	MOV R5, R7	MOV R5, @RW7+d8	MOV R6, R7	MOV R6, @RW7+d8	MOV R7, R7	MOV R7, @RW7+d8
+8	MOV R0,@RW0	MOV R0, @RW0+d16	MOV R1,@RW0	MOV R1, @RW0+d16	MOV R2,@RW0	MOV R2, @RW0+d16	MOV R3,@RW0	MOV R3, @RW0+d16	MOV R4,@RW0	MOV R4, @RW0+d16	MOV R5,@RW0	MOV R5, @RW0+d16	MOV R6,@RW0	MOV R6, @RW0+d16	MOV R7,@RW0	MOV R7, @RW0+d16
+9	MOV R0,@RW1	MOV R0, @RW1+d16	MOV R1,@RW1	MOV R1, @RW1+d16	MOV R2,@RW1	MOV R2, @RW1+d16	MOV R3,@RW1	MOV R3, @RW1+d16	MOV R4,@RW1	MOV R4, @RW1+d16	MOV R5,@RW1	MOV R5, @RW1+d16	MOV R6,@RW1	MOV R6, @RW1+d16	MOV R7,@RW1	MOV R7, @RW1+d16
+A	MOV R0,@RW2	MOV R0, @RW2+d16	MOV R1,@RW2	MOV R1, @RW2+d16	MOV R2,@RW2	MOV R2, @RW2+d16	MOV R3,@RW2	MOV R3, @RW2+d16	MOV R4,@RW2	MOV R4, @RW2+d16	MOV R5,@RW2	MOV R5, @RW2+d16	MOV R6,@RW2	MOV R6, @RW2+d16	MOV R7,@RW2	MOV R7, @RW2+d16
+B	MOV R0,@RW3	MOV R0, @RW3+d16	MOV R1,@RW3	MOV R1, @RW3+d16	MOV R2,@RW3	MOV R2, @RW3+d16	MOV R3,@RW3	MOV R3, @RW3+d16	MOV R4,@RW3	MOV R4, @RW3+d16	MOV R5,@RW3	MOV R5, @RW3+d16	MOV R6,@RW3	MOV R6, @RW3+d16	MOV R7,@RW3	MOV R7, @RW3+d16
+C	MOV R0, @RW0+	MOV R0, @RW0+RW7	MOV R1, @RW0+	MOV R1, @RW0+RW7	MOV R2, @RW0+	MOV R2, @RW0+RW7	MOV R3, @RW0+	MOV R3, @RW0+RW7	MOV R4, @RW0+	MOV R4, @RW0+RW7	MOV R5, @RW0+	MOV R5, @RW0+RW7	MOV R6, @RW0+	MOV R6, @RW0+RW7	MOV R7, @RW0+	MOV R7, @RW0+RW7
+D	MOV R0, @RW1+	MOV R0, @RW1+RW7	MOV R1, @RW1+	MOV R1, @RW1+RW7	MOV R2, @RW1+	MOV R2, @RW1+RW7	MOV R3, @RW1+	MOV R3, @RW1+RW7	MOV R4, @RW1+	MOV R4, @RW1+RW7	MOV R5, @RW1+	MOV R5, @RW1+RW7	MOV R6, @RW1+	MOV R6, @RW1+RW7	MOV R7, @RW1+	MOV R7, @RW1+RW7
+E	MOV R0, @RW2+	MOV R0, @PC+d16	MOV R1, @RW2+	MOV R1, @PC+d16	MOV R2, @RW2+	MOV R2, @PC+d16	MOV R3, @RW2+	MOV R3, @PC+d16	MOV R4, @RW2+	MOV R4, @PC+d16	MOV R5, @RW2+	MOV R5, @PC+d16	MOV R6, @RW2+	MOV R6, @PC+d16	MOV R7, @RW2+	MOV R7, @PC+d16
+F	MOV R0, @RW3+	MOV R0, addr16	MOV R1, @RW3+	MOV R1, addr16	MOV R2, @RW3+	MOV R2, addr16	MOV R3, @RW3+	MOV R3, addr16	MOV R4, @RW3+	MOV R4, addr16	MOV R5, @RW3+	MOV R5, addr16	MOV R6, @RW3+	MOV R6, addr16	MOV R7, @RW3+	MOV R7, addr16

Table A-48. MOVW RWi, ea Instruction (First Byte = $7B_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVW RW0, RW0	MOVW @RW0+d8	MOVW RW1, RW0	MOVW RW1, @RW0+d8	MOVW RW2, RW0	MOVW RW2, @RW0+d8	MOVW RW3, RW0	MOVW RW3, @RW0+d8	MOVW RW4, RW0	MOVW RW4, @RW0+d8	MOVW RW5, RW0	MOVW RW5, @RW0+d8	MOVW RW6, RW0	MOVW RW6, @RW0+d8	MOVW RW7, RW0	MOVW RW7, @RW0+d8
+1	MOVW RW0, RW1	MOVW @RW1+d8	MOVW RW1, RW1	MOVW RW1, @RW1+d8	MOVW RW2, RW1	MOVW RW2, @RW1+d8	MOVW RW3, RW1	MOVW RW3, @RW1+d8	MOVW RW4, RW1	MOVW RW4, @RW1+d8	MOVW RW5, RW1	MOVW RW5, @RW1+d8	MOVW RW6, RW1	MOVW RW6, @RW1+d8	MOVW RW7, RW1	MOVW RW7, @RW1+d8
+2	MOVW RW0, RW2	MOVW @RW2+d8	MOVW RW1, RW2	MOVW RW1, @RW2+d8	MOVW RW2, RW2	MOVW RW2, @RW2+d8	MOVW RW3, RW2	MOVW RW3, @RW2+d8	MOVW RW4, RW2	MOVW RW4, @RW2+d8	MOVW RW5, RW2	MOVW RW5, @RW2+d8	MOVW RW6, RW2	MOVW RW6, @RW2+d8	MOVW RW7, RW2	MOVW RW7, @RW2+d8
+3	MOVW RW0, RW3	MOVW @RW3+d8	MOVW RW1, RW3	MOVW RW1, @RW3+d8	MOVW RW2, RW3	MOVW RW2, @RW3+d8	MOVW RW3, RW3	MOVW RW3, @RW3+d8	MOVW RW4, RW3	MOVW RW4, @RW3+d8	MOVW RW5, RW3	MOVW RW5, @RW3+d8	MOVW RW6, RW3	MOVW RW6, @RW3+d8	MOVW RW7, RW3	MOVW RW7, @RW3+d8
+4	MOVW RW0, RW4	MOVW @RW4+d8	MOVW RW1, RW4	MOVW RW1, @RW4+d8	MOVW RW2, RW4	MOVW RW2, @RW4+d8	MOVW RW3, RW4	MOVW RW3, @RW4+d8	MOVW RW4, RW4	MOVW RW4, @RW4+d8	MOVW RW5, RW4	MOVW RW5, @RW4+d8	MOVW RW6, RW4	MOVW RW6, @RW4+d8	MOVW RW7, RW4	MOVW RW7, @RW4+d8
+5	MOVW RW0, RW5	MOVW @RW5+d8	MOVW RW1, RW5	MOVW RW1, @RW5+d8	MOVW RW2, RW5	MOVW RW2, @RW5+d8	MOVW RW3, RW5	MOVW RW3, @RW5+d8	MOVW RW4, RW5	MOVW RW4, @RW5+d8	MOVW RW5, RW5	MOVW RW5, @RW5+d8	MOVW RW6, RW5	MOVW RW6, @RW5+d8	MOVW RW7, RW5	MOVW RW7, @RW5+d8
+6	MOVW RW0, RW6	MOVW @RW6+d8	MOVW RW1, RW6	MOVW RW1, @RW6+d8	MOVW RW2, RW6	MOVW RW2, @RW6+d8	MOVW RW3, RW6	MOVW RW3, @RW6+d8	MOVW RW4, RW6	MOVW RW4, @RW6+d8	MOVW RW5, RW6	MOVW RW5, @RW6+d8	MOVW RW6, RW6	MOVW RW6, @RW6+d8	MOVW RW7, RW6	MOVW RW7, @RW6+d8
+7	MOVW RW0, RW7	MOVW @RW7+d8	MOVW RW1, RW7	MOVW RW1, @RW7+d8	MOVW RW2, RW7	MOVW RW2, @RW7+d8	MOVW RW3, RW7	MOVW RW3, @RW7+d8	MOVW RW4, RW7	MOVW RW4, @RW7+d8	MOVW RW5, RW7	MOVW RW5, @RW7+d8	MOVW RW6, RW7	MOVW RW6, @RW7+d8	MOVW RW7, RW7	MOVW RW7, @RW7+d8
+8	MOVW RW0,@RW0	MOVW @RW0+d16	MOVW RW1,@RW0	MOVW RW1, @RW0+d16	MOVW RW2,@RW0	MOVW RW2, @RW0+d16	MOVW RW3,@RW0	MOVW RW3, @RW0+d16	MOVW RW4,@RW0	MOVW RW4, @RW0+d16	MOVW RW5,@RW0	MOVW RW5, @RW0+d16	MOVW RW6,@RW0	MOVW RW6, @RW0+d16	MOVW RW7,@RW0	MOVW RW7, @RW0+d16
+9	MOVW RW0,@RW1	MOVW @RW1+d16	MOVW RW1,@RW1	MOVW RW1, @RW1+d16	MOVW RW2,@RW1	MOVW RW2, @RW1+d16	MOVW RW3,@RW1	MOVW RW3, @RW1+d16	MOVW RW4,@RW1	MOVW RW4, @RW1+d16	MOVW RW5,@RW1	MOVW RW5, @RW1+d16	MOVW RW6,@RW1	MOVW RW6, @RW1+d16	MOVW RW7,@RW1	MOVW RW7, @RW1+d16
+A	MOVW RW0,@RW2	MOVW @RW2+d16	MOVW RW1,@RW2	MOVW RW1, @RW2+d16	MOVW RW2,@RW2	MOVW RW2, @RW2+d16	MOVW RW3,@RW2	MOVW RW3, @RW2+d16	MOVW RW4,@RW2	MOVW RW4, @RW2+d16	MOVW RW5,@RW2	MOVW RW5, @RW2+d16	MOVW RW6,@RW2	MOVW RW6, @RW2+d16	MOVW RW7,@RW2	MOVW RW7, @RW2+d16
+B	MOVW RW0,@RW3	MOVW @RW3+d16	MOVW RW1,@RW3	MOVW RW1, @RW3+d16	MOVW RW2,@RW3	MOVW RW2, @RW3+d16	MOVW RW3,@RW3	MOVW RW3, @RW3+d16	MOVW RW4,@RW3	MOVW RW4, @RW3+d16	MOVW RW5,@RW3	MOVW RW5, @RW3+d16	MOVW RW6,@RW3	MOVW RW6, @RW3+d16	MOVW RW7,@RW3	MOVW RW7, @RW3+d16
+C	MOVW RW0,@RW0+	MOVW @RW0+RW7	MOVW RW1,@RW0+	MOVW RW1, @RW0+RW7	MOVW RW2,@RW0+	MOVW RW2, @RW0+RW7	MOVW RW3,@RW0+	MOVW RW3, @RW0+RW7	MOVW RW4,@RW0+	MOVW RW4, @RW0+RW7	MOVW RW5,@RW0+	MOVW RW5, @RW0+RW7	MOVW RW6,@RW0+	MOVW RW6, @RW0+RW7	MOVW RW7,@RW0+	MOVW RW7, @RW0+RW7
+D	MOVW RW0, @RW1+	MOVW @RW1+RW7	MOVW RW1, @RW1+	MOVW RW1, @RW1+RW7	MOVW RW2, @RW1+	MOVW RW2, @RW1+RW7	MOVW RW3, @RW1+	MOVW RW3, @RW1+RW7	MOVW RW4, @RW1+	MOVW RW4, @RW1+RW7	MOVW RW5, @RW1+	MOVW RW5, @RW1+RW7	MOVW RW6, @RW1+	MOVW RW6, @RW1+RW7	MOVW RW7, @RW1+	MOVW RW7, @RW1+RW7
+E	MOVW RW0, @RW2+	MOVW @PC+d16	MOVW RW1, @RW2+	MOVW RW1, @PC+d16	MOVW RW2, @RW2+	MOVW RW2, @PC+d16	MOVW RW3, @RW2+	MOVW RW3, @PC+d16	MOVW RW4, @RW2+	MOVW RW4, @PC+d16	MOVW RW5, @RW2+	MOVW @PC+d16	MOVW RW6, @RW2+	MOVW RW6, @PC+d16	MOVW RW7, @RW2+	MOVW RW7, @PC+d16
+F	MOVW RW0, @RW3+	MOVW RW0, addr16	MOVW RW1, @RW3+	MOVW RW1, addr16	MOVW RW2, @RW3+	MOVW RW2, addr16	MOVW RW3, @RW3+	MOVW RW3, addr16	MOVW RW4, @RW3+	MOVW RW4, addr16	MOVW RW5, @RW3+	MOVW RW5, addr16	MOVW RW6, @RW3+	MOVW RW6, addr16	MOVW RW7, @RW3+	MOVW RW7, addr16

Table A-49. MOV ea, Ri Instruction (First Byte = $7C_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOV R0, R0	MOV @RW0+d8, R0	MOV R0, R1	MOV @RW0+d8, R1	MOV R0, R2	MOV @RW0+d8, R2	MOV R0, R3	MOV @RW0+d8, R3	MOV R0, R4	MOV @RW0+d8, R4	MOV R0, R5	MOV @RW0+d8, R5	MOV R0, R6	MOV @RW0+d8, R6	MOV R0, R7	MOV @RW0+d8, R7
+1	MOV R1, R0	MOV @RW1+d8, R0	MOV R1, R1	MOV @RW1+d8, R1	MOV R1, R2	MOV @RW1+d8, R2	MOV R1, R3	MOV @RW1+d8, R3	MOV R1, R4	MOV @RW1+d8, R4	MOV R1, R5	MOV @RW1+d8, R5	MOV R1, R6	MOV @RW1+d8, R6	MOV R1, R7	MOV @RW1+d8, R7
+2	MOV R2, R0	MOV @RW2+d8, R0	MOV R2, R1	MOV @RW2+d8, R1	MOV R2, R2	MOV @RW2+d8, R2	MOV R2, R3	MOV @RW2+d8, R3	MOV R2, R4	MOV @RW2+d8, R4	MOV R2, R5	MOV @RW2+d8, R5	MOV R2, R6	MOV @RW2+d8, R6	MOV R2, R7	MOV @RW2+d8, R7
+3	MOV R3, R0	MOV @RW3+d8, R0	MOV R3, R1	MOV @RW3+d8, R1	MOV R3, R2	MOV @RW3+d8, R2	MOV R3, R3	MOV @RW3+d8, R3	MOV R3, R4	MOV @RW3+d8, R4	MOV R3, R5	MOV @RW3+d8, R5	MOV R3, R6	MOV @RW3+d8, R6	MOV R3, R7	MOV @RW3+d8, R7
+4	MOV R4, R0	MOV @RW4+d8, R0	MOV R4, R1	MOV @RW4+d8, R1	MOV R4, R2	MOV @RW4+d8, R2	MOV R4, R3	MOV @RW4+d8, R3	MOV R4, R4	MOV @RW4+d8, R4	MOV R4, R5	MOV @RW4+d8, R5	MOV R4, R6	MOV @RW4+d8, R6	MOV R4, R7	MOV @RW4+d8, R7
+5	MOV R5, R0	MOV @RW5+d8, R0	MOV R5, R1	MOV @RW5+d8, R1	MOV R5, R2	MOV @RW5+d8, R2	MOV R5, R3	MOV @RW5+d8, R3	MOV R5, R4	MOV @RW5+d8, R4	MOV R5, R5	MOV @RW5+d8, R5	MOV R5, R6	MOV @RW5+d8, R6	MOV R5, R7	MOV @RW5+d8, R7
+6	MOV R6, R0	MOV @RW6+d8, R0	MOV R6, R1	MOV @RW6+d8, R1	MOV R6, R2	MOV @RW6+d8, R2	MOV R6, R3	MOV @RW6+d8, R3	MOV R6, R4	MOV @RW6+d8, R4	MOV R6, R5	MOV @RW6+d8, R5	MOV R6, R6	MOV @RW6+d8, R6	MOV R6, R7	MOV @RW6+d8, R7
+7	MOV R7, R0	MOV @RW7+d8, R0	MOV R7, R1	MOV @RW7+d8, R1	MOV R7, R2	MOV @RW7+d8, R2	MOV R7, R3	MOV @RW7+d8, R3	MOV R7, R4	MOV @RW7+d8, R4	MOV R7, R5	MOV @RW7+d8, R5	MOV R7, R6	MOV @RW7+d8, R6	MOV R7, R7	MOV @RW7+d8, R7
+8	MOV @RW0, R0	MOV @RW0+d16, R0	MOV @RW0, R1	MOV @RW0+d16, R1	MOV @RW0, R2	MOV @RW0+d16, R2	MOV @RW0, R3	MOV @RW0+d16, R3	MOV @RW0, R4	MOV @RW0+d16, R4	MOV @RW0, R5	MOV @RW0+d16, R5	MOV @RW0, R6	MOV @RW0+d16, R6	MOV @RW0, R7	MOV @RW0+d16, R7
+9	MOV @RW1, R0	MOV @RW1+d16, R0	MOV @RW1, R1	MOV @RW1+d16, R1	MOV @RW1, R2	MOV @RW1+d16, R2	MOV @RW1, R3	MOV @RW1+d16, R3	MOV @RW1, R4	MOV @RW1+d16, R4	MOV @RW1, R5	MOV @RW1+d16, R5	MOV @RW1, R6	MOV @RW1+d16, R6	MOV @RW1, R7	MOV @RW1+d16, R7
+A	MOV @RW2, R0	MOV @RW2+d16, R0	MOV @RW2, R1	MOV @RW2+d16, R1	MOV @RW2, R2	MOV @RW2+d16, R2	MOV @RW2, R3	MOV @RW2+d16, R3	MOV @RW2, R4	MOV @RW2+d16, R4	MOV @RW2, R5	MOV @RW2+d16, R5	MOV @RW2, R6	MOV @RW2+d16, R6	MOV @RW2, R7	MOV @RW2+d16, R7
+B	MOV @RW3, R0	MOV @RW3+d16, R0	MOV @RW3, R1	MOV @RW3+d16, R1	MOV @RW3, R2	MOV @RW3+d16, R2	MOV @RW3, R3	MOV @RW3+d16, R3	MOV @RW3, R4	MOV @RW3+d16, R4	MOV @RW3, R5	MOV @RW3+d16, R5	MOV @RW3, R6	MOV @RW3+d16, R6	MOV @RW3, R7	MOV @RW3+d16, R7
+C	MOV @RW0+, R0	MOV @RW0+RW7, R0	MOV @RW0+, R1	MOV @RW0+RW7, R1	MOV @RW0+, R2	MOV @RW0+RW7, R2	MOV @RW0+, R3	MOV @RW0+RW7, R3	MOV @RW0+, R4	MOV @RW0+RW7, R4	MOV @RW0+, R5	MOV @RW0+RW7, R5	MOV @RW0+, R6	MOV @RW0+RW7, R6	MOV @RW0+, R7	MOV @RW0+RW7, R7
+D	MOV @RW1+, R0	MOV @RW1+RW7, R0	MOV @RW1+, R1	MOV @RW1+RW7, R1	MOV @RW1+, R2	MOV @RW1+RW7, R2	MOV @RW1+, R3	MOV @RW1+RW7, R3	MOV @RW1+, R4	MOV @RW1+RW7, R4	MOV @RW1+, R5	MOV @RW1+RW7, R5	MOV @RW1+, R6	MOV @RW1+RW7, R6	MOV @RW1+, R7	MOV @RW1+RW7, R7
+E	MOV @RW2+, R0	MOV @PC+d16, R0	MOV @RW2+, R1	MOV @PC+d16, R1	MOV @RW2+, R2	MOV @PC+d16, R2	MOV @RW2+, R3	MOV @PC+d16, R3	MOV @RW2+, R4	MOV @PC+d16, R4	MOV @RW2+, R5	MOV @PC+d16, R5	MOV @RW2+, R6	MOV @PC+d16, R6	MOV @RW2+, R7	MOV @PC+d16, R7
+F	MOV @RW3+, R0	MOV addr16, R0	MOV @RW3+, R1	MOV addr16, R1	MOV @RW3+, R2	MOV addr16, R2	MOV @RW3+, R3	MOV addr16, R3	MOV @RW3+, R4	MOV addr16, R4	MOV @RW3+, R5	MOV addr16, R5	MOV @RW3+, R6	MOV addr16, R6	MOV @RW3+, R7	MOV addr16, R7

Table A-50. MOVW ea, Rwi Instruction (First Byte = $7D_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	MOVW RW0, RW0	MOVW @RW0+d8, RW0	MOVW RW0, RW1	MOVW @RW0+d8, RW1	MOVW RW0, RW2	MOVW @RW0+d8, RW2	MOVW RW0, RW3	MOVW @RW0+d8, RW3	MOVW RW0, RW4	MOVW @RW0+d8, RW4	MOVW RW0, RW5	MOVW @RW0+d8, RW5	MOVW RW0, RW6	MOVW @RW0+d8, RW6	MOVW RW0, RW7	MOVW @RW0+d8, RW7
+1	MOVW RW1, RW0	MOVW @RW1+d8, RW0	MOVW RW1, RW1	MOVW @RW1+d8, RW1	MOVW RW1, RW2	MOVW @RW1+d8, RW2	MOVW RW1, RW3	MOVW @RW1+d8, RW3	MOVW RW1, RW4	MOVW @RW1+d8, RW4	MOVW RW1, RW5	MOVW @RW1+d8, RW5	MOVW RW1, RW6	MOVW @RW1+d8, RW6	MOVW RW1, RW7	MOVW @RW1+d8, RW7
+2	MOVW RW2, RW0	MOVW @RW2+d8, RW0	MOVW RW2, RW1	MOVW @RW2+d8, RW1	MOVW RW2, RW2	MOVW @RW2+d8, RW2	MOVW RW2, RW3	MOVW @RW2+d8, RW3	MOVW RW2, RW4	MOVW @RW2+d8, RW4	MOVW RW2, RW5	MOVW @RW2+d8, RW5	MOVW RW2, RW6	MOVW @RW2+d8, RW6	MOVW RW2, RW7	MOVW @RW2+d8, RW7
+3	MOVW RW3, RW0	MOVW @RW3+d8, RW0	MOVW RW3, RW1	MOVW @RW3+d8, RW1	MOVW RW3, RW2	MOVW @RW3+d8, RW2	MOVW RW3, RW3	MOVW @RW3+d8, RW3	MOVW RW3, RW4	MOVW @RW3+d8, RW4	MOVW RW3, RW5	MOVW @RW3+d8, RW5	MOVW RW3, RW6	MOVW @RW3+d8, RW6	MOVW RW3, RW7	MOVW @RW3+d8, RW7
+4	MOVW RW4, RW0	MOVW @RW4+d8, RW0	MOVW RW4, RW1	MOVW @RW4+d8, RW1	MOVW RW4, RW2	MOVW @RW4+d8, RW2	MOVW RW4, RW3	MOVW @RW4+d8, RW3	MOVW RW4, RW4	MOVW @RW4+d8, RW4	MOVW RW4, RW5	MOVW @RW4+d8, RW5	MOVW RW4, RW6	MOVW @RW4+d8, RW6	MOVW RW4, RW7	MOVW @RW4+d8, RW7
+5	MOVW RW5, RW0	MOVW @RW5+d8, RW0	MOVW RW5, RW1	MOVW @RW5+d8, RW1	MOVW RW5, RW2	MOVW @RW5+d8, RW2	MOVW RW5, RW3	MOVW @RW5+d8, RW3	MOVW RW5, RW4	MOVW @RW5+d8, RW4	MOVW RW5, RW5	MOVW @RW5+d8, RW5	MOVW RW5, RW6	MOVW @RW5+d8, RW6	MOVW RW5, RW7	MOVW @RW5+d8, RW7
+6	MOVW RW6, RW0	MOVW @RW6+d8, RW0	MOVW RW6, RW1	MOVW @RW6+d8, RW1	MOVW RW6, RW2	MOVW @RW6+d8, RW2	MOVW RW6, RW3	MOVW @RW6+d8, RW3	MOVW RW6, RW4	MOVW @RW6+d8, RW4	MOVW RW6, RW5	MOVW @RW6+d8, RW5	MOVW RW6, RW6	MOVW @RW6+d8, RW6	MOVW RW6, RW7	MOVW @RW6+d8, RW7
+7	MOVW RW7, RW0	MOVW @RW7+d8, RW0	MOVW RW7, RW1	MOVW @RW7+d8, RW1	MOVW RW7, RW2	MOVW @RW7+d8, RW2	MOVW RW7, RW3	MOVW @RW7+d8, RW3	MOVW RW7, RW4	MOVW @RW7+d8, RW4	MOVW RW7, RW5	MOVW @RW7+d8, RW5	MOVW RW7, RW6	MOVW @RW7+d8, RW6	MOVW RW7, RW7	MOVW @RW7+d8, RW7
+8	MOVW @RW0, RW0	MOVW @RW0+d16, RW0	MOVW @RW0, RW1	MOVW @RW0+d16, RW1	MOVW @RW0, RW2	MOVW @RW0+d16, RW2	MOVW @RW0, RW3	MOVW @RW0+d16, RW3	MOVW @RW0, RW4	MOVW @RW0+d16, RW4	MOVW @RW0, RW5	MOVW @RW0+d16, RW5	MOVW @RW0, RW6	MOVW @RW0+d16, RW6	MOVW @RW0, RW7	MOVW @RW0+d16, RW7
+9	MOVW @RW1, RW0	MOVW @RW1+d16, RW0	MOVW @RW1, RW1	MOVW @RW1+d16, RW1	MOVW @RW1, RW2	MOVW @RW1+d16, RW2	MOVW @RW1, RW3	MOVW @RW1+d16, RW3	MOVW @RW1, RW4	MOVW @RW1+d16, RW4	MOVW @RW1, RW5	MOVW @RW1+d16, RW5	MOVW @RW1, RW6	MOVW @RW1+d16, RW6	MOVW @RW1, RW7	MOVW @RW1+d16, RW7
+A	MOVW @RW2, RW0	MOVW @RW2+d16, RW0	MOVW @RW2, RW1	MOVW @RW2+d16, RW1	MOVW @RW2, RW2	MOVW @RW2+d16, RW2	MOVW @RW2, RW3	MOVW @RW2+d16, RW3	MOVW @RW2, RW4	MOVW @RW2+d16, RW4	MOVW @RW2, RW5	MOVW @RW2+d16, RW5	MOVW @RW2, RW6	MOVW @RW2+d16, RW6	MOVW @RW2, RW7	MOVW @RW2+d16, RW7
+B	MOVW @RW3, RW0	MOVW @RW3+d16, RW0	MOVW @RW3, RW1	MOVW @RW3+d16, RW1	MOVW @RW3, RW2	MOVW @RW3+d16, RW2	MOVW @RW3, RW3	MOVW @RW3+d16, RW3	MOVW @RW3, RW4	MOVW @RW3+d16, RW4	MOVW @RW3, RW5	MOVW @RW3+d16, RW5	MOVW @RW3, RW6	MOVW @RW3+d16, RW6	MOVW @RW3, RW7	MOVW @RW3+d16, RW7
+C	MOVW @RW0+, RW0	MOVW @RW0+RW7, RW0	MOVW @RW0+, RW1	MOVW @RW0+RW7, RW1	MOVW @RW0+, RW2	MOVW @RW0+RW7, RW2	MOVW @RW0+, RW3	MOVW @RW0+RW7, RW3	MOVW @RW0+, RW4	MOVW @RW0+RW7, RW4	MOVW @RW0+, RW5	MOVW @RW0+RW7, RW5	MOVW @RW0+, RW6	MOVW @RW0+RW7, RW6	MOVW @RW0+, RW7	MOVW @RW0+RW7, RW7
+D	MOVW @RW1+, RW0	MOVW @RW1+RW7, RW0	MOVW @RW1+, RW1	MOVW @RW1+RW7, RW1	MOVW @RW1+, RW2	MOVW @RW1+RW7, RW2	MOVW @RW1+, -RW7, RW3	MOVW @RW1+RW7, RW3	MOVW @RW1+, RW4	MOVW @RW1+RW7, RW4	MOVW @RW1+, RW5	MOVW @RW1+RW7, RW5	MOVW @RW1+, RW6	MOVW @RW1+RW7, RW6	MOVW @RW1+, RW7	MOVW @RW1+RW7, RW7
+E	MOVW @RW2+, RW0	MOVW @PC +d16, RW0	MOVW @RW2+, RW1	MOVW @PC +d16, RW1	MOVW @RW2+, RW2	MOVW @PC +d16, RW2	MOVW @RW2+, RW3	MOVW @PC +d16, RW3	MOVW @RW2+, RW4	MOVW @PC +d16, RW4	MOVW @RW2+, RW5	MOVW @PC +d16, RW5	MOVW @RW2+, RW6	MOVW @PC +d16, RW6	MOVW @RW2+, RW7	MOVW @PC +d16, RW7
+F	MOVW @RW3+, RW0	MOVW addr16, RW0	MOVW @RW3+, RW1	MOVW addr16, RW1	MOVW @RW3+, RW2	MOVW addr16, RW2	MOVW @RW3+, RW3	MOVW addr16, RW3	MOVW @RW3+, RW4	MOVW addr16, RW4	MOVW @RW3+, RW5	MOVW addr16, RW5	MOVW @RW3+, RW6	MOVW addr16, RW6	MOVW @RW3+, RW7	MOVW addr16, RW7

Table A-51. XCH Ri, ea Instruction (First Byte = $7E_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	XCH R0, R0	XCH R0, @RW0+d8	XCH R1, R0	XCH R1, @RW0+d8	XCH R2, R0	XCH R2, @RW0+d8	XCH R3, R0	XCH R3, @RW0+d8	XCH R4, R0	XCH R4, @RW0+d8	XCH R5, R0	XCH R5, @RW0+d8	XCH R6, R0	XCH R6, @RW0+d8	XCH R7, R0	XCH R7, @RW0+d8
+1	XCH R0, R1	XCH R0, @RW1+d8	XCH R1, R1	XCH R1, @RW1+d8	XCH R2, R1	XCH R2, @RW1+d8	XCH R3, R1	XCH R3, @RW1+d8	XCH R4, R1	XCH R4, @RW1+d8	XCH R5, R1	XCH R5, @RW1+d8	XCH R6, R1	XCH R6, @RW1+d8	XCH R7, R1	XCH R7, @RW1+d8
+2	XCH R0, R2	XCH R0, @RW2+d8	XCH R1, R2	XCH R1, @RW2+d8	XCH R2, R2	XCH R2, @RW2+d8	XCH R3, R2	XCH R3, @RW2+d8	XCH R4, R2	XCH R4, @RW2+d8	XCH R5, R2	XCH R5, @RW2+d8	XCH R6, R2	XCH R6, @RW2+d8	XCH R7, R2	XCH R7, @RW2+d8
+3	XCH R0, R3	XCH R0, @RW3+d8	XCH R1, R3	XCH R1, @RW3+d8	XCH R2, R3	XCH R2, @RW3+d8	XCH R3, R3	XCH R3, @RW3+d8	XCH R4, R3	XCH R4, @RW3+d8	XCH R5, R3	XCH R5, @RW3+d8	XCH R6, R3	XCH R6, @RW3+d8	XCH R7, R3	XCH R7, @RW3+d8
+4	XCH R0, R4	XCH R0, @RW4+d8	XCH R1, R4	XCH R1, @RW4+d8	XCH R2, R4	XCH R2, @RW4+d8	XCH R3, R4	XCH R3, @RW4+d8	XCH R4, R4	XCH R4, @RW4+d8	XCH R5, R4	XCH R5, @RW4+d8	XCH R6, R4	XCH R6, @RW4+d8	XCH R7, R4	XCH R7, @RW4+d8
+5	XCH R0, R5	XCH R0, @RW5+d8	XCH R1, R5	XCH R1, @RW5+d8	XCH R2, R5	XCH R2, @RW5+d8	XCH R3, R5	XCH R3, @RW5+d8	XCH R4, R5	XCH R4, @RW5+d8	XCH R5, R5	XCH R5, @RW5+d8	XCH R6, R5	XCH R6, @RW5+d8	XCH R7, R5	XCH R7, @RW5+d8
+6	XCH R0, R6	XCH R0, @RW6+d8	XCH R1, R6	XCH R1, @RW6+d8	XCH R2, R6	XCH R2, @RW6+d8	XCH R3, R6	XCH R3, @RW6+d8	XCH R4, R6	XCH R4, @RW6+d8	XCH R5, R6	XCH R5, @RW6+d8	XCH R6, R6	XCH R6, @RW6+d8	XCH R7, R6	XCH R7, @RW6+d8
+7	XCH R0, R7	XCH R0, @RW7+d8	XCH R1, R7	XCH R1, @RW7+d8	XCH R2, R7	XCH R2, @RW7+d8	XCH R3, R7	XCH R3, @RW7+d8	XCH R4, R7	XCH R4, @RW7+d8	XCH R5, R7	XCH R5, @RW7+d8	XCH R6, R7	XCH R6, @RW7+d8	XCH R7, R7	XCH R7, @RW7+d8
+8	XCH R0,@RW0	XCH R0, @RW0+d16	XCH R1,@RW0	XCH R1, @RW0+d16	XCH R2,@RW0	XCH R2, @RW0+d16	XCH R3,@RW0	XCH R3, @RW0+d16	XCH R4,@RW0	XCH R4, @RW0+d16	XCH R5,@RW0	XCH R5, @RW0+d16	XCH R6,@RW0	XCH R6, @RW0+d16	XCH R7,@RW0	XCH R7, @RW0+d16
+9	XCH R0,@RW1	XCH R0, @RW1+d16	XCH R1,@RW1	XCH R1, @RW1+d16	XCH R2,@RW1	XCH R2, @RW1+d16	XCH R3,@RW1	XCH R3, @RW1+d16	XCH R4,@RW1	XCH R4, @RW1+d16	XCH R5,@RW1	XCH R5, @RW1+d16	XCH R6,@RW1	XCH R6, @RW1+d16	XCH R7,@RW1	XCH R7, @RW1+d16
+A	XCH R0,@RW2	XCH R0, @RW2+d16	XCH R1,@RW2	XCH R1, @RW2+d16	XCH R2,@RW2	XCH R2, @RW2+d16	XCH R3,@RW2	XCH R3, @RW2+d16	XCH R4,@RW2	XCH R4, @RW2+d16	XCH R5,@RW2	XCH R5, @RW2+d16	XCH R6,@RW2	XCH R6, @RW2+d16	XCH R7,@RW2	XCH R7, @RW2+d16
+B	XCH R0,@RW3	XCH R0, @RW3+d16	XCH R1,@RW3	XCH R1, @RW3+d16	XCH R2,@RW3	XCH R2, @RW3+d16	XCH R3,@RW3	XCH R3, @RW3+d16	XCH R4,@RW3	XCH R4, @RW3+d16	XCH R5,@RW3	XCH R5, @RW3+d16	XCH R6,@RW3	XCH R6, @RW3+d16	XCH R7,@RW3	XCH R7, @RW3+d16
+C	XCH R0,@RW0+	XCH R0, @RW0+RW7	XCH R1,@RW0+	XCH R1, @RW0+RW7	XCH R2,@RW0+	XCH R2, @RW0+RW7	XCH R3,@RW0+	XCH R3, @RW0+RW7	XCH R4,@RW0+	XCH R4, @RW0+RW7	XCH R5,@RW0+	XCH R5, @RW0+RW7	XCH R6,@RW0+	XCH R6, @RW0+RW7	XCH R7,@RW0+	XCH R7, @RW0+RW7
+D	XCH R0,@RW1+	XCH R0, @RW1+RW7	XCH R1,@RW1+	XCH R1, @RW1+RW7	XCH R2,@RW1+	XCH R2, @RW1+RW7	XCH R3,@RW1+	XCH R3, @RW1+RW7	XCH R4,@RW1+	XCH R4, @RW1+RW7	XCH R5,@RW1+	XCH R5, @RW1+RW7	XCH R6,@RW1+	XCH R6, @RW1+RW7	XCH R7,@RW1+	XCH R7, @RW1+RW7
+E	XCH R0,@RW2+	XCH R0, @PC+d16	XCH R1,@RW2+	XCH R1, @PC+d16	XCH R2,@RW2+	XCH R2, @PC+d16	XCH R3,@RW2+	XCH R3, @PC+d16	XCH R4,@RW2+	XCH R4, @PC+d16	XCH R5,@RW2+	XCH R5, @PC+d16	XCH R6,@RW2+	XCH R6, @PC+d16	XCH R7,@RW2+	XCH R7, @PC+d16
+F	XCH R0,@RW3+	XCH R0, addr16	XCH R1,@RW3+	XCH R1, addr16	XCH R2,@RW3+	XCH R2, addr16	XCH R3,@RW3+	XCH R3, addr16	XCH R4,@RW3+	XCH R4, addr16	XCH R5,@RW3+	XCH R5, addr16	XCH R6,@RW3+	XCH R6, addr16	XCH R7,@RW3+	XCH R7, addr16

Table A-52. XCHW RWi, ea Instruction (First Byte = $7F_H$)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	XCHW RW0, RW0	XCHW RW0, @RW0+d8	XCHW RW1, RW0	XCHW RW1, @RW0+d8	XCHW RW2, RW0	XCHW RW2, @RW0+d8	XCHW RW3, RW0	XCHW RW3, @RW0+d8	XCHW RW4, RW0	XCHW RW4, @RW0+d8	XCHW RW5, RW0	XCHW RW5, @RW0+d8	XCHW RW6, RW0	XCHW RW6, @RW0+d8	XCHW RW7, RW0	XCHW RW7, @RW0+d8
+1	XCHW RW0, RW1	XCHW RW0, @RW1+d8	XCHW RW1, RW1	XCHW RW1, @RW1+d8	XCHW RW2, RW1	XCHW RW2, @RW1+d8	XCHW RW3, RW1	XCHW RW3, @RW1+d8	XCHW RW4, RW1	XCHW RW4, @RW1+d8	XCHW RW5, RW1	XCHW RW5, @RW1+d8	XCHW RW6, RW1	XCHW RW6, @RW1+d8	XCHW RW7, RW1	XCHW RW7, @RW1+d8
+2	XCHW RW0, RW2	XCHW RW0, @RW2+d8	XCHW RW1, RW2	XCHW RW1, @RW2+d8	XCHW RW2, RW2	XCHW RW2, @RW2+d8	XCHW RW3, RW2	XCHW RW3, @RW2+d8	XCHW RW4, RW2	XCHW RW4, @RW2+d8	XCHW RW5, RW2	XCHW RW5, @RW2+d8	XCHW RW6, RW2	XCHW RW6, @RW2+d8	XCHW RW7, RW2	XCHW RW7, @RW2+d8
+3	XCHW RW0, RW3	XCHW RW0, @RW3+d8	XCHW RW1, RW3	XCHW RW1, @RW3+d8	XCHW RW2, RW3	XCHW RW2, @RW3+d8	XCHW RW3, RW3	XCHW RW3, @RW3+d8	XCHW RW4, RW3	XCHW RW4, @RW3+d8	XCHW RW5, RW3	XCHW RW5, @RW3+d8	XCHW RW6, RW3	XCHW RW6, @RW3+d8	XCHW RW7, RW3	XCHW RW7, @RW3+d8
+4	XCHW RW0, RW4	XCHW RW0, @RW4+d8	XCHW RW1, RW4	XCHW RW1, @RW4+d8	XCHW RW2, RW4	XCHW RW2, @RW4+d8	XCHW RW3, RW4	XCHW RW3, @RW4+d8	XCHW RW4, RW4	XCHW RW4, @RW4+d8	XCHW RW5, RW4	XCHW RW5, @RW4+d8	XCHW RW6, RW4	XCHW RW6, @RW4+d8	XCHW RW7, RW4	XCHW RW7, @RW4+d8
+5	XCHW RW0, RW5	XCHW RW0, @RW5+d8	XCHW RW1, RW5	XCHW RW1, @RW5+d8	XCHW RW2, RW5	XCHW RW2, @RW5+d8	XCHW RW3, RW5	XCHW RW3, @RW5+d8	XCHW RW4, RW5	XCHW RW4, @RW5+d8	XCHW RW5, RW5	XCHW RW5, @RW5+d8	XCHW RW6, RW5	XCHW RW6, @RW5+d8	XCHW RW7, RW5	XCHW RW7, @RW5+d8
+6	XCHW RW0, RW6	XCHW RW0, @RW6+d8	XCHW RW1, RW6	XCHW RW1, @RW6+d8	XCHW RW2, RW6	XCHW RW2, @RW6+d8	XCHW RW3, RW6	XCHW RW3, @RW6+d8	XCHW RW4, RW6	XCHW RW4, @RW6+d8	XCHW RW5, RW6	XCHW RW5, @RW6+d8	XCHW RW6, RW6	XCHW RW6, @RW6+d8	XCHW RW7, RW6	XCHW RW7, @RW6+d8
+7	XCHW RW0, RW7	XCHW RW0, @RW7+d8	XCHW RW1, RW7	XCHW RW1, @RW7+d8	XCHW RW2, RW7	XCHW RW2, @RW7+d8	XCHW RW3, RW7	XCHW RW3, @RW7+d8	XCHW RW4, RW7	XCHW RW4, @RW7+d8	XCHW RW5, RW7	XCHW RW5, @RW7+d8	XCHW RW6, RW7	XCHW RW6, @RW7+d8	XCHW RW7, RW7	XCHW RW7, @RW7+d8
+8	XCHW RW0,@RW0	XCHW RW0, @RW0+d16	XCHW RW1,@RW0	XCHW RW1, @RW0+d16	XCHW RW2,@RW0	XCHW RW2, @RW0+d16	XCHW RW3,@RW0	XCHW RW3, @RW0+d16	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 RW0, @RW1+d16	XCHW RW1,@RW1	XCHW RW1, @RW1+d16	XCHW RW2,@RW1	XCHW RW2, @RW1+d16	XCHW RW3,@RW1	XCHW RW3, @RW1+d16	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 RW0, @RW2+d16	XCHW RW1,@RW2	XCHW RW1, @RW2+d16	XCHW RW2,@RW2	XCHW RW2, @RW2+d16	XCHW RW3,@RW2	XCHW RW3, @RW2+d16	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 RW0, @RW3+d16	XCHW RW1,@RW3	XCHW RW1, @RW3+d16	XCHW RW2,@RW3	XCHW RW2, @RW3+d16	XCHW RW3,@RW3	XCHW RW3, @RW3+d16	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 RW0, @RW0+RW7	XCHW RW1,@RW0+	XCHW RW1, @RW0+RW7	XCHW RW2,@RW0+	XCHW RW2, @RW0+RW7	XCHW RW3,@RW0+	XCHW RW3, @RW0+RW7	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 RW0, @RW1+RW7	XCHW RW1,@RW1+	XCHW RW1, @RW1+RW7	XCHW RW2,@RW1+	XCHW RW2, @RW1+RW7	XCHW RW3,@RW1+	XCHW RW3, @RW1+RW7	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 RW0, @PC+d16	XCHW RW1,@RW2+	XCHW RW1, @PC+d16	XCHW RW2,@RW2+	XCHW RW2, @PC+d16	XCHW RW3,@RW2+	XCHW RW3, @PC+d16	XCHW RW4,@RW2+	XCHW RW4, @PC+d16	XCHW RW5,@RW2+	XCHW RW5, @PC+d16	XCHW RW6,@RW2+	XCHW RW6, @PC+d16	XCHW RW7,@RW2+	XCHW RW7, @PC+d16
+F	XCHW RW0,@RW3+	XCHW RW0, addr16	XCHW RW1,@RW3+	XCHW RW1, addr16	XCHW RW2,@RW3+	XCHW RW2, addr16	XCHW RW3,@RW3+	XCHW RW3, addr16	XCHW RW4,@RW3+	XCHW RW4, addr16	XCHW RW5,@RW3+	XCHW RW5, addr16	XCHW RW6,@RW3+	XCHW RW6, addr16	XCHW RW7,@RW3+	XCHW RW7, addr16

Major Changes

Spansion Publication Number: CM44-10139-6E

Page	Changes (For details, refer to main body.)	
371	Chapter 16 DTP/External Interrupts 16.1 Overview of DTP/External Interrupt Unit	Added the <Notes>.
420	Chapter 18 Multi-function Serial Interface 18.2 Pins of the Multi-function Serial Interface	Added “18.2 Pins of the Multi-function Serial Interface”.
642	Chapter 22 Flash Memory 22.8 Flash Security Function	Corrected “Table 22.8-1 Address and Protection Code of Flash Security Bit”.
655 to 657	Chapter 24 Delay Interrupt Generation Module	Added the new chapter.

NOTE: Please see “Revision History” about later revised information.

Revision History

Revision History

Document Title: MB90880 Series F^{2}MC-16LX Hardware Manual Document Number: 002-04554				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
**	—	12/14/2012	AKIH	Migrated to Cypress and assigned document number 002-04554. No change to document contents or format.
*A	5571623	03/01/2017	AKIH	Updated to Cypress format.