Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

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.

16-Bit Microcontroller F^2MC-16LX Family

MB90920 Series Hardware Manual

Doc. # 002-06975 Rev .*A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
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

1. Objectives and Intended Reader

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

The MB90920 series has been developed as one of the general-purpose products of the F^2MC-16LX family, which is an original 16-bit single-chip microcontroller compatible with the Application Specific IC (ASIC).

This manual describes the functions and operations of the MB90920 series for engineers who actually use this semiconductor to design products. Please read this manual first.

2. Organization of this Manual

This manual consists of the following 27 chapters and an appendix:

1. Overview

   This chapter describes features and provides the basic specification of the MB90920 series.

2. CPU

   This chapter describes F^2MC-16LX CPU.

3. Interrupt

   This chapter describes the relationships between interrupts and the extended intelligent I/O service (EI^{2}OS).

4. Reset

   This chapter describes the reset operation.

5. Clock

   This chapter describes the clock.

6. Low-Power Consumption Mode

   This chapter explains the low-power consumption mode.

7. Mode Setting

   This chapter explains the operation mode and memory access mode.

8. I/O Ports

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

9. Watchdog Timer/Time-Base Timer/Watch Timer (Used as Sub Clock)

   This chapter describes the functions and operations of the watchdog timer, time-base timer, and watch timer (used as sub clock).

10. Input Capture

    This chapter describes the input capture operation.

11. 16-Bit Reload Timer

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

12. PPG Timer

    This chapter describes the operations of PPG timer.

13. Real-Time Watch Timer

    This chapter describes the functions and operations of the real-time watch timer.

14. Delay Interrupt Generation Module

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

15. DTP/External Interrupt Circuit

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

16. 8-/10-Bit A/D Converter

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

17. LIN-UART

    This chapter explains the functions and operations of the LIN-UART.

18. CAN Controller

    This chapter describes an overview of the CAN controller and its functions.

19. LCD Controller/Driver

    This chapter describes the functions and operations of the LCD controller/driver.

20. Low-Voltage/CPU Operation Detection Reset Circuit

    This chapter describes the functions and operations of the low-voltage/CPU operation detection reset circuit.

21. Stepping Motor Controller

    This chapter describes the functions and operations of the stepping motor controller.

22. Sound Generator

    This chapter describes the functions and operations of the sound generator.

23. Address Match Detection Function

    This chapter describes the functions and operations of the address match detection function.

24. ROM Mirror Function Select Module

    This chapter describes the ROM mirror function select module.

25. 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".

26. Examples of Serial Programming Connection for Flash Memory Products

    This chapter gives the examples of connection for serial programming using the AF220/AF210/AF120/AF110 Flash Micro-controller Programmer manufactured by Yokogawa Digital Computer Corporation.

27. ROM Security Function

    This chapter explains the ROM security function.

28. Appendix

    The appendix provides the I/O map and describes the available instructions.

Contents

Preface 3

1. Overview 13

1.1 Overview of MB90920 Series 14
1.2 Features 15
1.3 Block Diagram 17
1.4 Package Dimension 18
1.5 Pin Assignment 19
1.6 Pin Functions 20
1.7 I/O Circuit Types 27
1.8 Precautions for Handling Device 32

2. CPU 37

2.1 Outline of CPU 38
2.2 Memory Space 40
2.3 Memory Map 42
2.4 Addressing 44
2.4.1 Addressing with Linear Scheme 45
2.4.2 Addressing with Bank Scheme 46
2.5 Allocation of Multiple-Byte Data in the Memory 48
2.6 Registers 50
2.7 Dedicated Registers 51
2.7.1 Accumulator (A) 54
2.7.2 Stack Pointers (USP, SSP) 57
2.7.3 Processor Status (PS) 59
2.7.4 Program Counter (PC) 63
2.7.5 Direct Page Register (DPR) 64
2.7.6 Bank Registers (PCB, DTB, USB, SSB, ADB) 65
2.8 General-purpose Register 66
2.9 Prefix Codes 68

3. Interrupt 73

3.1 Outline of Interrupts 74
3.2 Interrupt Sources and Interrupt Vectors 76
3.3 Interrupt Control Registers and Peripheral Functions 79
3.3.1 Interrupt Control Registers (ICR00 to ICR15) 81
3.3.2 Functions of Interrupt Control Registers 84
3.4 Hardware Interrupt 87
3.4.1 Hardware Interrupt Operation 90
3.4.2 Operation Flow of Hardware Interrupt 92
3.4.3 Procedure for Using Hardware Interrupt 94
3.4.4 Multiple Interrupts 95

3.4.5 Hardware Interrupt Processing Time 97
3.5 Software Interrupt 99
3.6 Interrupt by Extended Intelligent I/O Service (EI^{2}OS) 101
3.6.1 Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) 103
3.6.2 Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) Registers 105
3.6.3 Operation of the Extended intelligent I/O Service (EI^{2}OS) 108
3.6.4 Extended Intelligent I/O Service (EI^{2}OS) Procedure 110
3.6.5 Extended Intelligent I/O Service (EI^{2}OS) Processing Time 112
3.7 Exception Handling Interrupt by Execution of Undefined Instruction 114
3.8 Stack Operations of Interrupt Handling 115
3.9 Example Program for Interrupt Handling 117

4. Reset 121

4.1 Outline of Reset 122
4.2 Reset Sources and Oscillation Stabilization Wait Time 125
4.3 External Reset Pin 127
4.4 Reset Operation 128
4.5 Reset Source Bit 130
4.6 State of Each Pin by Reset 134
4.7 Reset Output Function 135

5. Clock 137

5.1 Clock 138
5.2 Block Diagram of the Clock Generation Block 141
5.2.1 Register in the Clock Generation Block 144
5.3 Clock Selection Register (CKSCR) 145
5.4 PLL/Sub Clock Control Register (PSCCR) 149
5.5 Clock Mode 151
5.6 Oscillation Stabilization Wait Time 155
5.7 Connection of Oscillator and External Clock 156

6. Low-Power Consumption Mode 159

6.1 Overview of the Low-power Consumption Mode 160
6.2 Block Diagram of Low-power Consumption Circuit 164
6.3 Low-power Consumption Mode Control Register (LPMCR) 167
6.4 CPU Intermittent Operation Mode 170
6.5 Standby Mode 171
6.5.1 Sleep Mode 173
6.5.2 Time-base Timer Mode 176
6.5.3 Watch Mode 178
6.5.4 Stop Mode 180
6.6 State Transition Diagram 183
6.7 Pin State in the Standby Mode and at the Time of Reset 185
6.8 Notes on Using the Low-power Consumption Mode 187

7. Mode Setting 191

7.1 Mode Setting 192
7.2 Mode Pins (MD2 to MD0) 193
7.3 Mode Data 194

8. I/O Ports 197

8.1 I/O Ports 198
8.2 Assignment of Registers and Pins Shared with External Pins 201
8.3 Port 0 203
8.3.1 Port 0 Registers (PDR0, DDR0) 205
8.3.2 Description of Port 0 Operation 206
8.4 Port 1 208
8.4.1 Port 1 Registers (PDR1, DDR1) 210
8.4.2 Description of Port 1 Operation 211
8.5 Port 2 213
8.5.1 Port 2 Data Register (PDR2, DDR2) 215
8.5.2 Description of Port 2 Operation 216
8.6 Port 3 218
8.6.1 Port 3 Registers (PDR3, DDR3) 220
8.6.2 Description of Port 3 Operation 221
8.7 Port 4 223
8.7.1 Port 4 Registers (PDR4, DDR4) 225
8.7.2 Description of Port 4 Operation 226
8.8 Port 5 228
8.8.1 Port 5 Registers (PDR5, DDR5) 231
8.8.2 Description of Port 5 Operation 232
8.9 Port 6 234
8.9.1 Port 6 Registers (PDR6, DDR6, ADER6) 236
8.9.2 Description of Port 6 Operation 237
8.10 Port 7 239
8.10.1 Port 7 Registers (PDR7, DDR7) 241
8.10.2 Description of Port 7 Operation 243
8.11 Port 8 245
8.11.1 Port 8 Registers (PDR8, DDR8) 247
8.11.2 Description of Port 8 Operation 249
8.12 Port 9 251
8.12.1 Registers for Port 9 (PDR9, DDR9) 253
8.12.2 Description of Port 9 Operation 254
8.13 Port C 256
8.13.1 Registers for Port C (PDRC, DDRC) 259
8.13.2 Description of Port C Operation 260
8.14 Port D 263
8.14.1 Registers for Port D (PDRD, DDRD) 265
8.14.2 Description of Port D Operation 266
8.15 Port E 268
8.15.1 Registers for Port E (PDRE, DDRE) 270
8.15.2 Description of Port E Operation 271
8.16 Input Level Select Registers (PIL0 to PIL2) 273
8.17 Sample Program for I/O Ports 276

9. Watchdog Timer/Time-Base Timer/Watch Timer (Used as Sub Clock) 277

9.1 Outline of Watchdog Timer/Time-base Timer/Watch Timer 278
9.2 Block Diagrams of Watchdog Timer/Time-base Timer/Watch Timer 279
9.3 List of Registers for Watchdog Timer/Time-base Timer/Watch Timer 280
9.3.1 Watchdog Timer Control Register (WDTC) 281
9.3.2 Time-base Timer Control Register (TBTC) 283
9.3.3 Watch Timer Control Register (WTC) 285

9.4 Operation of Watchdog Timer/Time-base Timer/Watch Timer 287
9.4.1 Watchdog Timer Operation 288
9.4.2 Operation of Time-base Timer 290
9.4.3 Operation of Watch Timer 293
9.5 Notes on Using the Watchdog Timer/Time-base Timer 294
9.6 Program Example for Watchdog Timer/Time-base Timer 297

10. Input Capture 299

10.1 Outline of Input Capture 300
10.1.1 Block Diagram of Input Capture 301
10.2 List of Input Capture Registers 304
10.2.1 Detailed Description of the Input Capture Registers 307
10.2.2 Input Capture Edge Register (ICE) 309
10.2.3 Detailed Description of 16-Bit Free-run Timer Register 314
10.3 Description of Operations 318
10.3.1 16-bit Input Capture 319
10.3.2 16-bit Free-run Timer 321

11. 16-Bit Reload Timer 323

11.1 Overview of 16-bit Reload Timer 324
11.2 Configuration of 16-bit Reload Timer 326
11.3 Pins of 16-bit Reload Timer 328
11.4 Registers of 16-bit Reload Timer 330
11.4.1 Timer Control Status Registers (Upper) (TMCSR0H to TMCSR3H) 331
11.4.2 Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L) 334
11.4.3 16-bit Timer Registers (TMR0 to TMR3) 337
11.4.4 16-bit Reload Registers (TMRLR0 to TMRLR3) 338
11.5 Interrupts of 16-bit Reload Timer 339
11.6 Operation of 16-bit Reload Timer 340
11.6.1 Internal Clock Mode (Reload Mode) 342
11.6.2 Internal Clock Mode (One Shot Mode) 344
11.6.3 Event Count Mode 346
11.7 Notes on Using 16-bit Reload Timer 348
11.8 Sample Program for 16-bit Reload Timer 349

12. PPG Timer 353

12.1 Overview of PPG Timer 354
12.2 Block Diagram of PPG Timer 355
12.3 Registers of PPG Timer 356
12.3.1 List of PPG Timer Registers 357
12.3.2 Detailed Description of PPG Timer 359
12.4 Interrupt of PPG Timer 366
12.5 PPG Timer Operation 368
12.5.1 PWM Operation 369
12.5.2 One Shot Operation 371
12.5.3 Interrupt Source and Timing 372

13. Real-Time Watch Timer 373

13.1 Overview of Real-time Watch Timer 374
13.2 Registers of Real-time Watch Timer 375
13.2.1 Real-Time Watch Timer Control Register 378
13.2.2 Sub-Second Data Register 380

13.2.3 Second/Minute/Hour/Day Data Registers 381
13.3 Interrupt of Real-time Watch Timer 383

14. Delay Interrupt Generation Module 385

14.1 Overview of Delay Interrupt Generation Module 386
14.2 Operation of Delay Interrupt Generation Module 387

15. DTP/External Interrupt Circuit 389

15.1 Overview of DTP/External Interrupt Circuit 390
15.2 Configuration of DTP/External Interrupt Circuit 392
15.3 Pins of DTP/External Interrupt Circuit 394
15.4 Registers of DTP/External Interrupt Circuit 396
15.4.1 External Interrupt Source Register (EIRR) 397
15.4.2 External Interrupt Enable Register (ENIR) 398
15.4.3 External Interrupt Level Setting Register (ELVRH/ELVRL) 399
15.5 Operations of DTP/External Interrupt Circuit 401
15.5.1 External Interrupt Function 404
15.5.2 DTP Function 405
15.6 Notes on Using the DTP/External Interrupt Circuit 407
15.7 Sample Programs of DTP/External Interrupt Circuit 409

16. 8-/10-Bit A/D Converter 413

16.1 Overview of 8-/10-bit A/D Converter 414
16.2 Block Diagram of 8-/10-bit A/D Converter 416
16.3 Configuration of 8-/10-bit A/D Converter 420
16.3.1 Upper bits in the A/D control status register (ADCS1) 422
16.3.2 Lower Bits in the A/D Control Status Register (ADCS0) 427
16.3.3 A/D Data Registers (ADCR0/ADCR1) 429
16.3.4 A/D Setting Registers (ADSR0/ADSR1) 430
16.3.5 Analog Input Enable Register (ADER6) 434
16.4 Interrupts of 8-/10-bit A/D Converter 435
16.5 Explanation of 8-/10-bit A/D Converter Operations 436
16.5.1 Single Conversion Mode 438
16.5.2 Continuous Conversion Mode 440
16.5.3 Stop Conversion Mode 442
16.5.4 Conversion Operation by EI^{2}OS Function 444
16.5.5 A/D Conversion Data Protection Function 445
16.6 Precautions when Using 8-/10-bit A/D Converter 449

17. LIN-UART 451

17.1 Overview of LIN-UART 452
17.2 Configuration of LIN-UART 455
17.3 Pins of LIN-UART 460
17.4 LIN-UART Registers 462
17.4.1 Serial Control Register (SCR) 463
17.4.2 LIN-UART Serial Mode Register (SMR) 466
17.4.3 Serial Status Register (SSR) 469
17.4.4 Reception Data Register and Transmission Data Register (RDR/TDR) 473
17.4.5 Extended Status Control Register (ESCR) 475
17.4.6 Extended Communication Control Register (ECCR) 478
17.4.7 Baud Rate Generator Registers 0 and 1 (BGRn0/BGRn1) 481
17.5 Interrupts of LIN-UART 482

17.5.1 Timing of Reception Interrupt Generation and Flag Set 485
17.5.2 Timing of Transmission Interrupt Generation and Flag Set 487
17.6 LIN-UART Baud Rates 489
17.6.1 Setting the Baud Rate 491
17.6.2 Reload counter 494
17.7 Operation of LIN-UART 496
17.7.1 Operation in Asynchronous Mode (Operation Modes 0 and 1) 498
17.7.2 Operation in Synchronous Mode (Operating Mode 2) 502
17.7.3 Operation with LIN Function (Operation Mode 3) 506
17.7.4 Serial Pin Direct Access 509
17.7.5 Bidirectional Communication Function (Normal Mode) 510
17.7.6 Master/Slave Type Communication Function (Multiprocessor Mode) 512
17.7.7 LIN Communication Function 515
17.7.8 Sample Flowcharts for LIN-UART in LIN Communication (Operation Mode 3) 516
17.8 Notes on Using LIN-UART 520

18. CAN Controller 523

18.1 Features of CAN Controller 524
18.2 Block Diagram of CAN Controller 525
18.3 Classification of CAN Controller Registers 527
18.3.1 Control Status Register (CSR) 536
18.3.2 Last Event Indication Register (LEIR) 541
18.3.3 Receive and Transmit Error Counters (RTEC) 543
18.3.4 Bit Timing Register (BTR) 544
18.3.5 Message Buffer Valid Register (BVALR) 547
18.3.6 IDE Register (IDER) 548
18.3.7 Transmission Request Register (TREQR) 549
18.3.8 Transmission RTR register (TRTRR) 550
18.3.9 Remote Frame Receive Wait Register (RFWTR) 551
18.3.10Transmission Cancel Register (TCANR) 552
18.3.11Transmission Complete Register (TCR) 553
18.3.12Transmission Interrupt Enable Register (TIER) 554
18.3.13Receive Complete Register (RCR) 555
18.3.14Remote Request Receive Register (RRTRR) 556
18.3.15Receive Overrun Register (ROVRR) 557
18.3.16Receive Interrupt Enable Register (RIER) 558
18.3.17Acceptance Mask Selection Register (AMSR) 559
18.3.18Acceptance Mask Registers 0 and 1 (AMR0/AMR1) 561
18.3.19Message Buffers 563
18.3.20ID Register x (x = 0 to 15) (IDRx) 564
18.3.21DLC Register x (x = 0 to 15) (DLCRx) 568
18.3.22Data Register x (x = 0 to 15) (DTRx) 569
18.4 CAN Controller Transmission 571
18.5 CAN Controller Reception 574
18.6 Using CAN Controller 578
18.7 Procedure of Transmission via Message Buffer (x) 579
18.8 Procedure of Reception Via Message Buffer (x) 581
18.9 Specifying the Multi-level Message Buffer Configuration 583
18.10 CAN WAKE UP Function 586
18.11 Precautions When Using CAN Controller 587
18.12 Sample Program of CAN 588

19. LCD Controller/Driver 591

19.1 Overview of LCD Controller/Driver 592
19.2 Configuration of LCD Controller/Driver 593
19.2.1 Internal Divided Resistor of LCD Controller/Driver 596
19.2.2 External Divided Resistor of LCD Controller/Driver 598
19.3 LCD Controller/Driver Pins 600
19.4 Registers of LCD Controller/Driver 602
19.4.1 Lower Bits in the LCD Control Register (LCRL) 603
19.4.2 Upper Bits in the LCD Control Register (LCRH) 606
19.4.3 LCD Output Control Register 1/2 (LOCR1/LOCR2) 608
19.4.4 LCD Output Control Register 3 (LOCR3) 611
19.5 LCD Controller/Driver Display RAM 612
19.6 Operation of LCD Controller/Driver 616
19.6.1 Output Waveform during the LCD Controller/Driver Operation (1/2 Duty) 618
19.6.2 Output Waveform during the LCD Controller/Driver Operation (1/3 Duty) 621
19.6.3 Output Waveform during the LCD Controller/Driver Operation (1/4 Duty) 624

20. Low-Voltage/CPU Operation Detection Reset Circuit 627

20.1 Overview of the Low-voltage/CPU Operation Detection Reset 628
20.2 Configuration of the Low-Voltage/CPU Operation Detection 630
20.3 Register of the Low-voltage/CPU Operation Detection Reset Circuit 632
20.4 Operation of the Low-voltage/CPU Operation Detection Reset Circuit 634
20.5 Notes on Using the Low-voltage/CPU Operation Detection Reset Circuit 635
20.6 Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637

21. Stepping Motor Controller 639

21.1 Overview of the Stepping Motor Controller 640
21.2 Registers of the Stepping Motor Controller 641
21.2.1 PWM Control Register 643
21.2.2 PWM1 and PWM2 Compare Registers 644
21.2.3 PWM1 and PWM2 Selection Registers 646
21.3 Operation of the Stepping Motor Controller 648
21.4 Notes on Using the Stepping Motor Controller 650

22. Sound Generator 651

22.1 Outline of the Sound Generator 652
22.2 Registers of the Sound Generator 653
22.2.1 Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1) 655
22.2.2 Frequency Data Register (SGFR0/SGFR1) 657
22.2.3 Amplitude Data Register (SGAR0/SGAR1) 658
22.2.4 Decrement Grade Register (SGDR0/SGDR1) 659
22.2.5 Tone Count Register (SGTR0/SGTR1) 660

23. Address Match Detection Function 661

23.1 Outline of the Address Match Detection Function 662
23.2 Sample Application of the Address Match Detection Function 665
23.2.1 Example of Program Error Correction 667
23.2.2 Example of Correction Processing 668

24. ROM Mirror Function Select Module 671
24.1 Outline of the ROM Mirror Function Select Module 672
24.2 ROM Mirror Function Select Register (ROMM) 673

25. Flash Memory 675
25.1 Overview of Flash Memory 676
25.2 Sector Configuration of Flash Memory 677
25.3 Flash Memory Control Status Register (FMCS) 680
25.4 Flash Memory Write Control Registers (FWR0/FWR1) 682
25.5 Starting the Flash Memory Automatic Algorithm 688
25.6 Confirming the Automatic Algorithm Execution State 689
25.6.1 Data Polling Flag (DQ7) 690
25.6.2 Toggle Bit Flag (DQ6) 692
25.6.3 Timing Limit Exceeded Flag (DQ5) 693
25.6.4 Sector Erase Timer Flag (DQ3) 694
25.7 Writing Data to and Erasing Data from Flash Memory 695
25.7.1 Setting Flash Memory to the Read/Reset State 696
25.7.2 Write Data to Flash Memory 697
25.7.3 Erasing All Data of Flash Memory (Chip Erase) 699
25.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase) 700
25.7.5 Suspending Sector Erase of Flash Memory 702
25.7.6 Restarting Sector Erase of Flash Memory 703
25.8 Flash Security Function 704
25.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems 705
25.10 Notes on Using Flash Memory 708

26. Examples of Serial Programming Connection for Flash Memory Products 709
26.1 Basic Configuration for Serial Programming Connection 710
26.2 Example of Connection in Single-Chip Mode (Using Power from User System) 714
26.3 Example of Connection with Flash Microcontroller Programmer (Using Power from the User System) 717

27. ROM Security Function 721
27.1 Overview of ROM Security Function 722

28. Appendix 723
APPENDIX A I/O Maps 724
APPENDIX B Instructions 784
B.1 Instruction Types 785
B.2 Addressing 786
B.3 Direct Addressing 788
B.4 Indirect Addressing 795
B.5 Execution Cycle Count 803
B.6 Effective address field 806
B.7 How to Read the Instruction List 807
B.8 F^2MC-16LX Instruction List 810
B.9 Instruction Map 824
APPENDIX C Main Changes 847

Revision History 877

1. Overview

This chapter describes features and provides the basic specification of the MB90920 series.

1.1 Overview of MB90920 Series

1.2 Features

1.3 Block Diagram

1.4 Package Dimension

1.5 Pin Assignment

1.6 Pin Functions

1.7 I/O Circuit Types

1.8 Precautions for Handling Device

1.1 Overview of MB90920 Series

This section outlines MB90920 series products.

■ Overview of MB90920 Series

Table 1.1-1 gives an outline of MB90920 series products.

Table 1.1-1 Outline of MB90920 Series Products

Features	MB90V920	MB90F921*	MB90F922	MB90F923	MB90F924	MB90922
Models available	EVA product	Flash memory product				Mask ROM product
CPU	F^2MC-16LX CPU					
Clock	Single clock / Dual clock (Optional selection: Single clock product uses X0A/X1A as ports)					
System clock	On-chip PLL clock multiplication scheme (×1, ×2, ×3, ×4, ×6, ×8, and 1/2 for PLL stop) Minimum instruction execution time: 31.25ns (4 MHz oscillation × 8)					
ROM	External	128 K bytes	256 K bytes	384 K bytes	512 K bytes	256 K bytes
RAM	30 K bytes	10 K bytes	10 K bytes	16 K bytes	24 K bytes	10 K bytes
CAN interface	4 channels					
Low-voltage/CPU operation detection reset	No	Yes				
Package	PGA-299	LQFP-120				
Emulator-dedicated power supply	No	-				

* : Planning product

1.2 Features

This section describes the features of MB90920 series.

■ Features

Table 1.2-1 shows the features of MB90920 series.

Table 1.2-1 Features of MB90920 Series (Sheet 1 of 2)

Function	Features
16-bit reload timer (4 channels)	Allows 16-bit reload timer operations (e.g. toggle output and one shot output selectable) and the selection of the event count function.
16-bit free-run timer (1 channel)	Outputs an interrupt signal when an overflow occurs. Operation clock frequency: fsys, fsys/2, fsys/4, fsys/8, fsys/16, fsys/32, fsys/64, fsys/128 (fsys = system clock frequency)
16-bit input capture (8 channels)	Detects a rising edge, falling edge or both edges. 16-bit capture register × 8 Latches the counter value of the 16-bit free-run timer upon edge detection of pin input and generates an interrupt request.
PPG timer (6 channels)	Output pin × 3, external trigger input pin × 1 Operation clock frequency: fcp, fcp/2^2, fcp/2^4, fcp/2^6
Watch timer (Sub clock) (1 channel)	Directly driven by the oscillation clock. Supports the correction of oscillation deviations. Second/minute/hour registers allowing read/write operations Signal interrupt
LIN UART (4 channels)	Uses a dedicated reload timer to allow a wide range of communication speeds to be set. The LIN function can be used as the LIN master or a LIN slave.
SIO (1 channel) (only EVA product)	Transmission can be started from MSB or LSB. Support the internal clock synchronous transmission and the external clock synchronous transmission. Support the positive edge and the negative edge clock synchronization.Baud rate system clock (at 16MHz) =31.25K/62.5K/125K/500K/1M/2Mbps
LCD controller (1 channel)	Segment driver and common driver can drive an LCD panel (liquid crystal display) directly.
A/D converter (8 channels)	Resolution of 10 bits or 8 bits × 8 channels (input multiplex) Conversion time: 3 μs or less (fcp=32MHz) Allows external trigger activation (P50/INT0/ADTG). Can be activated by an internal timer (16-bit reload timer 1)
CAN (4 channels)	Conforms to CAN specification version 2.0 part A and part B. Automatic retransmission is executed if an error occurs. Automatic transmission is executed in response to a remote frame. Supports data and multiple messages for 16 prioritized message buffers for ID. Flexible configuration of acceptance filter: All bit compare/all bit mask/two partial bit masks Supports up to 1 Mbps. CAN wake up function Note: CAN0/CAN2 and CAN1/CAN3 share the transmission/reception pin and the interrupt control registers; therefore, simultaneous transmission/reception can be performed at 2 channels. These CAN's can be used as if 2 channels of 32-message buffer CAN were available.

Table 1.2-1 Features of MB90920 Series (Sheet 2 of 2)

Function	Features
Stepping motor controller (4 channels)	High current output for each channel × 4 Synchronized 8/10-bit PWM for each channel × 2
Sound generator (2 channels)	8-bit PWM signal is mixed with tone frequency from 8-bit reload counter. PWM frequency: 125kHz, 62.5kHz, 31.2kHz, 15.6kHz (when fcp = 32 MHz) Tone frequency: PWM frequency /2/ (reload value +1)
External interrupt (8 channels)	8 independent channels Interrupt sources: "L" → "H" edge/"H" → "L" edge/"L" level/"H" level can be selected.
Real time watch timer (1 channel)	Directly driven by the oscillation clock. Supports the correction of oscillation deviations. Built-in subsecond/second/minute/hour/day registers Signal interrupt
Low-voltage /CPU operation detection reset	Automatically reset if low-voltage is detected. CPU operation detection function
ROM security	Protects ROM content (MASK ROM products only)
Input level	Automotive (0.8Vcc/0.5Vcc) (initial value) CMOS Hysteresis (0.8Vcc/0.2Vcc) CMOS Hysteresis (0.7Vcc/0.3Vcc) (SIN only) can be selected
Input/output port	Push-pull output and Schmitt trigger input Programmable in units of individual bits as input/output or peripheral signal.
Flash memory	Automatic algorithm (equivalent to Embedded Algorithm) Supports data write/sector erase/chip erase/sector erase suspend/erase resume commands. Flag for indicating completion of automatic algorithm processing. Number of delete cycles: 10,000 cycles Data retention period: 10 years Can execute a erase operation for sector. Flash Security Feature for protecting the content of the Flash memory.

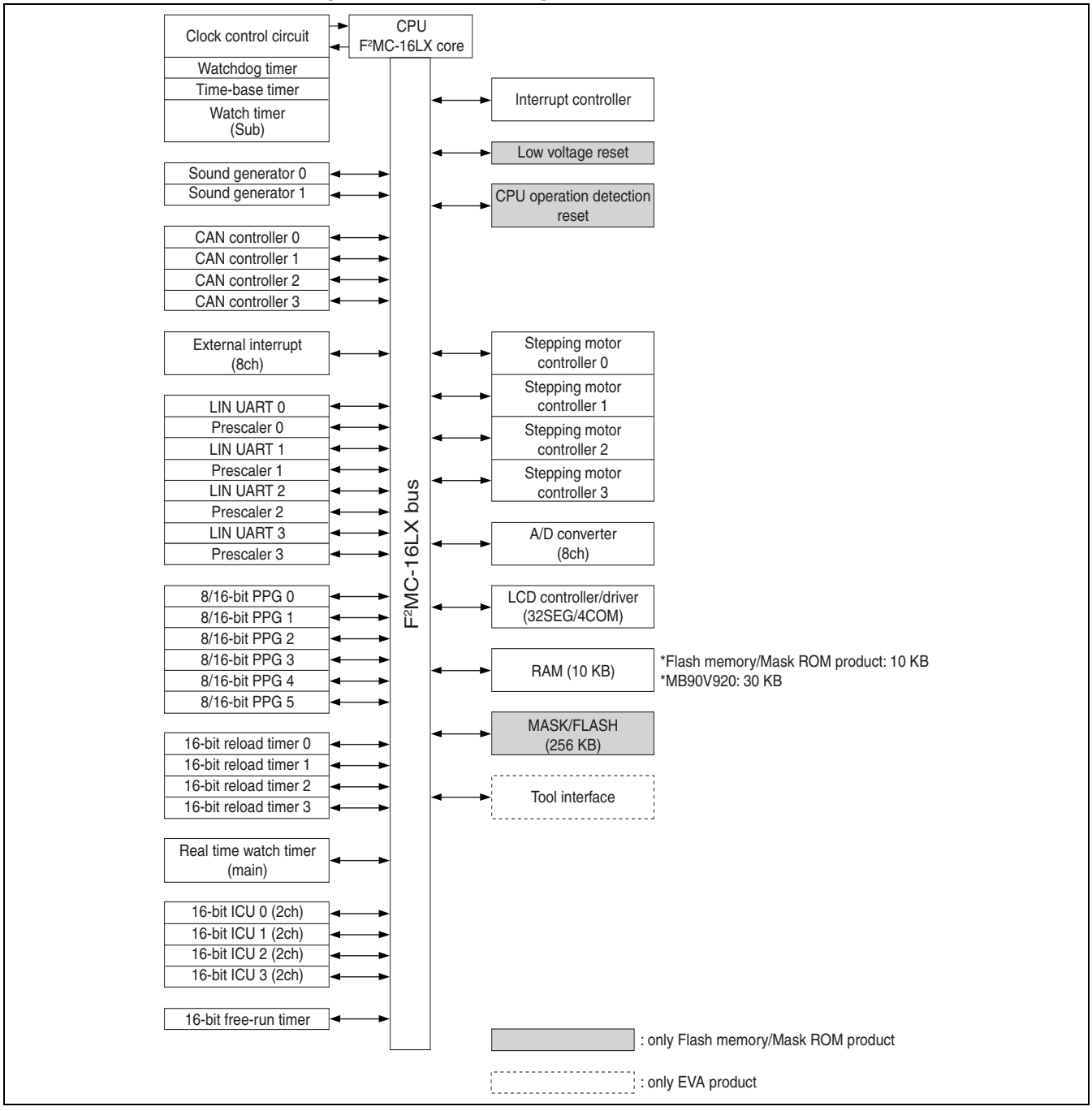
1.3 Block Diagram

This section shows a block diagram of MB90920 series products.

■ Block Diagram

Figure 1.3-1 shows a block diagram of MB90920 series products.

Figure 1.3-1 Block Diagram of MB90920 Series



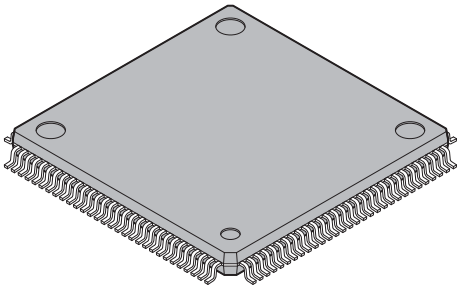
1.4 Package Dimension

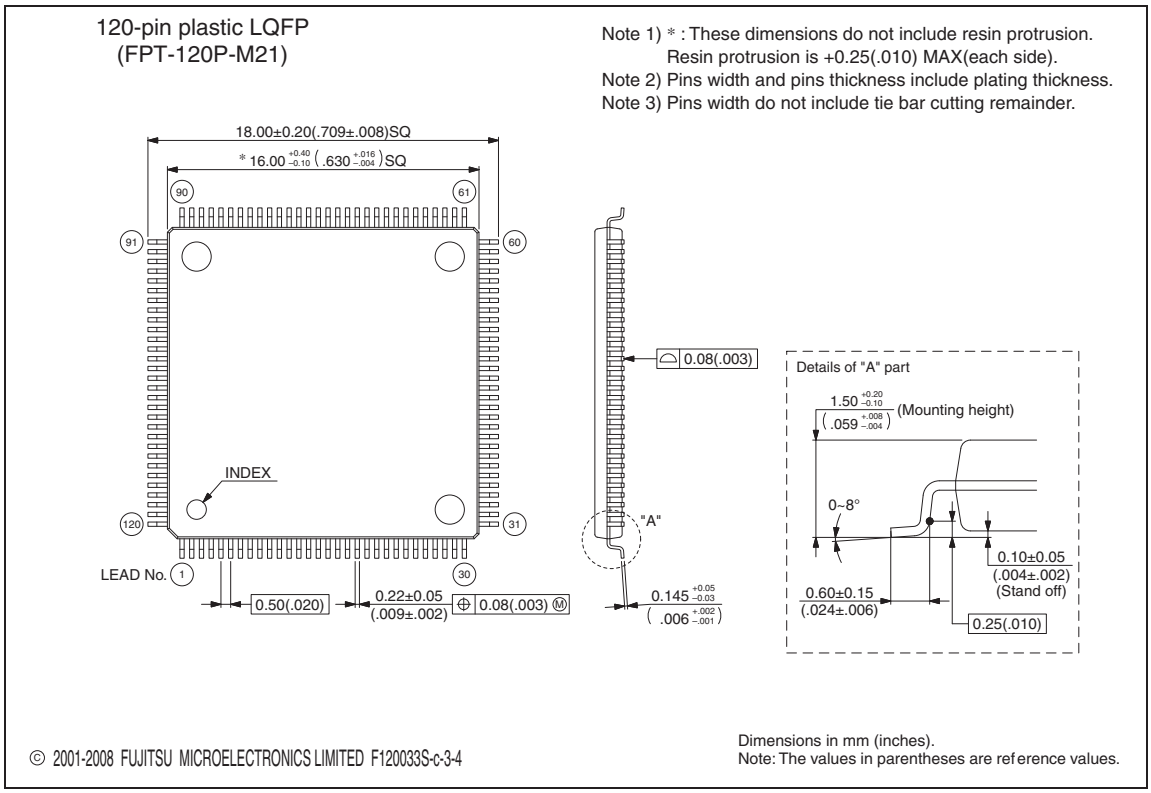
This section shows a package dimension of MB90920 series products.

■ Package Dimension

Figure 1.4-1 shows a package dimension.

Figure 1.4-1 Package Dimension

120-pin plastic LQFP  (FPT-120P-M21)	Lead pitch	0.50 mm
	Package width × package length	16.0 × 16.0 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm MAX
	Weight	0.88 g
	Code (Reference)	P-LFQFP120-16×16-0.50



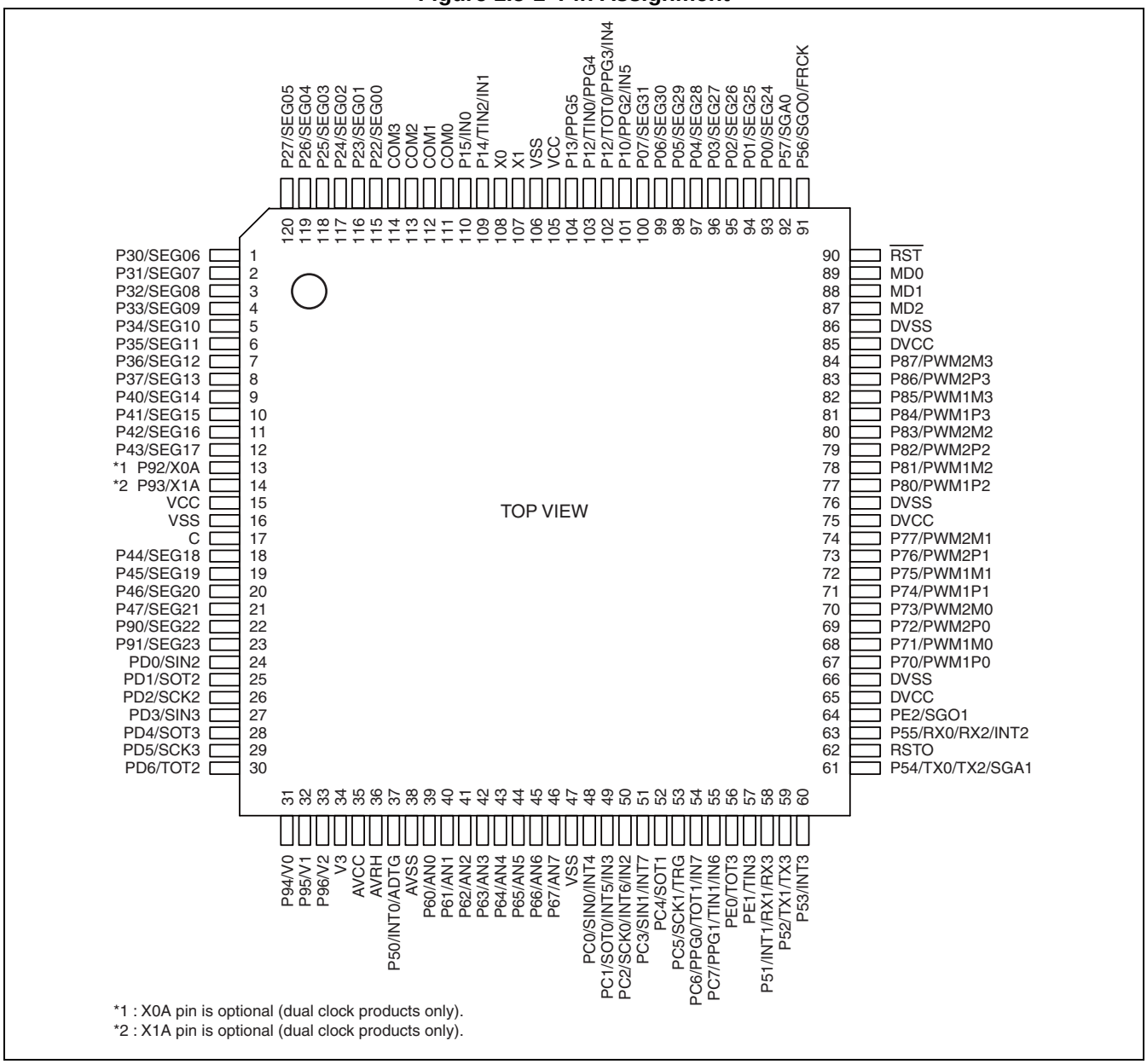
1.5 Pin Assignment

This section shows the pin assignment of MB90920 series products.

■ Pin Assignment

Figure 1.5-1 shows the pin assignment.

Figure 1.5-1 Pin Assignment



*1 : X0A pin is optional (dual clock products only).
*2 : X1A pin is optional (dual clock products only).

1.6 Pin Functions

This section describes the pin functions of MB90920 series products.

■ Description of Pin Functions

Table 1.6-1 describes the pin functions.

Table 1.6-1 Pin Functions (Sheet 1 of 7)

Pin no.	Pin name	Circuit type	Description
108	X0	A	High speed oscillator input pin.
107	X1		High speed oscillator output pin.
13	X0A	B	Low speed oscillator input pin. *1
	P92	I	General-purpose I/O port.
14	X1A	B	Low speed oscillator output pin. *2
	P93	I	General-purpose I/O port.
90	$\overline{\text{RST}}$	C	Reset input pin.
93	P00	F	General-purpose I/O port.
	SEG24		LCD controller/driver segment output pin.
94	P01	F	General-purpose I/O port.
	SEG25		LCD controller/driver segment output pin.
95	P02	F	General-purpose I/O port.
	SEG26		LCD controller/driver segment output pin.
96	P03	F	General-purpose I/O port.
	SEG27		LCD controller/driver segment output pin.
97	P04	F	General-purpose I/O port.
	SEG28		LCD controller/driver segment output pin.
98	P05	F	General-purpose I/O port.
	SEG29		LCD controller/driver segment output pin.
99	P06	F	General-purpose I/O port.
	SEG30		LCD controller/driver segment output pin.
100	P07	F	General-purpose I/O port.
	SEG31		LCD controller/driver segment output pin.
101	P10	I	General-purpose I/O port.
	PPG2		16-bit PPG ch.2 output pin.
	IN5		Input capture ch.5 trigger input pin.

Table 1.6-1 Pin Functions (Sheet 2 of 7)

Pin no.	Pin name	Circuit type	Description
102	P11	I	General-purpose I/O port.
	TOT0		16-bit reload timer ch.0 TOT output pin.
	PPG3		16-bit PPG ch.3 output pin.
	IN4		Input capture ch.4 trigger input pin.
103	P12	I	General-purpose I/O port.
	TIN0		16-bit reload timer ch.0 TIN input pin.
	PPG4		16-bit PPG ch.4 output pin.
104	P13	I	General-purpose I/O port.
	PPG5		16-bit PPG ch.5 output pin.
109	P14	I	General-purpose I/O port.
	TIN2		16-bit reload timer ch.2 TIN input pin.
	IN1		Input capture ch.1 trigger input pin.
110	P15	I	General-purpose I/O port.
	IN0		Input capture ch.0 trigger input pin.
111	COM0	P	LCD controller/driver common output pin.
112	COM1	P	LCD controller/driver common output pin.
113	COM2	P	LCD controller/driver common output pin.
114	COM3	P	LCD controller/driver common output pin.
115	P22	F	General-purpose I/O port.
	SEG00		LCD controller/driver segment output pin.
116	P23	F	General-purpose I/O port.
	SEG01		LCD controller/driver segment output pin.
117	P24	F	General-purpose I/O port.
	SEG02		LCD controller/driver segment output pin.
118	P25	F	General-purpose I/O port.
	SEG03		LCD controller/driver segment output pin.
119	P26	F	General-purpose I/O port.
	SEG04		LCD controller/driver segment output pin.
120	P27	F	General-purpose I/O port.
	SEG05		LCD controller/driver segment output pin.
1	P30	F	General-purpose I/O port.
	SEG06		LCD controller/driver segment output pin.
2	P31	F	General-purpose I/O port.
	SEG07		LCD controller/driver segment output pin.
3	P32	F	General-purpose I/O port.
	SEG08		LCD controller/driver segment output pin.

Table 1.6-1 Pin Functions (Sheet 3 of 7)

Pin no.	Pin name	Circuit type	Description
4	P33	F	General-purpose I/O port.
	SEG09		LCD controller/driver segment output pin.
5	P34	F	General-purpose I/O port.
	SEG10		LCD controller/driver segment output pin.
6	P35	F	General-purpose I/O port.
	SEG11		LCD controller/driver segment output pin.
7	P36	F	General-purpose I/O port.
	SEG12		LCD controller/driver segment output pin.
8	P37	F	General-purpose I/O port.
	SEG13		LCD controller/driver segment output pin.
9	P40	F	General-purpose I/O port.
	SEG14		LCD controller/driver segment output pin.
10	P41	F	General-purpose I/O port.
	SEG15		LCD controller/driver segment output pin.
11	P42	F	General-purpose I/O port.
	SEG16		LCD controller/driver segment output pin.
12	P43	F	General-purpose I/O port.
	SEG17		LCD controller/driver segment output pin.
18	P44	F	General-purpose I/O port.
	SEG18		LCD controller/driver segment output pin.
19	P45	F	General-purpose I/O port.
	SEG19		LCD controller/driver segment output pin.
20	P46	F	General-purpose I/O port.
	SEG20		LCD controller/driver segment output pin.
21	P47	F	General-purpose I/O port.
	SEG21		LCD controller/driver segment output pin.
37	P50	I	General-purpose I/O port.
	INT0		INT0 external interrupt input pin.
	ADTG		A/D converter external trigger input pin.
58	P51	I	General-purpose I/O port.
	INT1		INT1 external interrupt input pin.
	RX1		CAN interface 1 RX input pin.
	RX3		CAN interface 3 RX input pin.
59	P52	I	General-purpose I/O port.
	TX1		CAN interface 1 TX output pin.
	TX3		CAN interface 3 TX output pin.

Table 1.6-1 Pin Functions (Sheet 4 of 7)

Pin no.	Pin name	Circuit type	Description
60	P53	I	General-purpose I/O port.
	INT3		INT3 external interrupt input pin.
61	P54	I	General-purpose I/O port.
	TX0		CAN interface 0 TX output pin.
	TX2		CAN interface 2 TX output pin.
	SGA1		Sound generator ch.1 SGA output pin.
63	P55	I	General-purpose I/O port.
	RX0		CAN interface 0 RX input pin.
	RX2		CAN interface 2 RX input pin.
	INT2		INT2 external interrupt input pin.
91	P56	I	General-purpose I/O port.
	SGO0		Sound generator ch.0 SGO output pin.
	FRCK		Free-run timer clock input pin.
92	P57	I	General-purpose I/O port.
	SGA0		Sound generator ch.0 SGA output pin.
39	P60	H	General-purpose I/O port.
	AN0		A/D converter input pin.
40	P61	H	General-purpose I/O port.
	AN1		A/D converter input pin.
41	P62	H	General-purpose I/O port.
	AN2		A/D converter input pin.
42	P63	H	General-purpose I/O port.
	AN3		A/D converter input pin.
43	P64	H	General-purpose I/O port.
	AN4		A/D converter input pin.
44	P65	H	General-purpose I/O port.
	AN5		A/D converter input pin.
45	P66	H	General-purpose I/O port.
	AN6		A/D converter input pin.
46	P67	H	General-purpose I/O port.
	AN7		A/D converter input pin.
67	P70	L	General-purpose output-only port.
	PWM1P0		Stepping motor controller ch.0 output pin.
68	P71	L	General-purpose output-only port.
	PWM1M0		Stepping motor controller ch.0 output pin.
69	P72	L	General-purpose output-only port.
	PWM2P0		Stepping motor controller ch.0 output pin.

Table 1.6-1 Pin Functions (Sheet 5 of 7)

Pin no.	Pin name	Circuit type	Description
70	P73	L	General-purpose output-only port.
	PWM2M0		Stepping motor controller ch.0 output pin.
71	P74	L	General-purpose output-only port.
	PWM1P1		Stepping motor controller ch.1 output pin.
72	P75	L	General-purpose output-only port.
	PWM1M1		Stepping motor controller ch.1 output pin.
73	P76	L	General-purpose output-only port.
	PWM2P1		Stepping motor controller ch.1 output pin.
74	P77	L	General-purpose output-only port.
	PWM2M1		Stepping motor controller ch.1 output pin.
77	P80	L	General-purpose output-only port.
	PWM1P2		Stepping motor controller ch.2 output pin.
78	P81	L	General-purpose output-only port.
	PWM1M2		Stepping motor controller ch.2 output pin.
79	P82	L	General-purpose output-only port.
	PWM2P2		Stepping motor controller ch.2 output pin.
80	P83	L	General-purpose output-only port.
	PWM2M2		Stepping motor controller ch.2 output pin.
81	P84	L	General-purpose output-only port.
	PWM1P3		Stepping motor controller ch.3 output pin.
82	P85	L	General-purpose output-only port.
	PWM1M3		Stepping motor controller ch.3 output pin.
83	P86	L	General-purpose output-only port.
	PWM2P3		Stepping motor controller ch.3 output pin.
84	P87	L	General-purpose output-only port.
	PWM2M3		Stepping motor controller ch.3 output pin.
22	P90	F	General-purpose I/O port.
	SEG22		LCD controller/driver segment output pin.
23	P91	F	General-purpose I/O port.
	SEG23		LCD controller/driver segment output pin.
31	P94	G	General-purpose I/O port.
	V0		LCD controller/driver reference power supply pin.
32	P95	G	General-purpose I/O port.
	V1		LCD controller/driver reference power supply pin.
33	P96	G	General-purpose I/O port.
	V2		LCD controller/driver reference power supply pin.
34	V3	-	LCD controller/driver reference power supply pin.

Table 1.6-1 Pin Functions (Sheet 6 of 7)

Pin no.	Pin name	Circuit type	Description
48	PC0	J	General-purpose I/O port.
	SIN0		UART ch.0 serial data input pin.
	INT4		INT4 external interrupt input pin.
49	PC1	I	General-purpose I/O port.
	SOT0		UART ch.0 serial data output pin.
	INT5		INT5 external interrupt input pin.
	IN3		Input capture ch.3 trigger input pin.
50	PC2	I	General-purpose I/O port.
	SCK0		UART ch.0 serial clock I/O pin.
	INT6		INT6 external interrupt input pin.
	IN2		Input capture ch.2 trigger input pin.
51	PC3	J	General-purpose I/O port.
	SIN1		UART ch.1 serial data input pin.
	INT7		INT7 external interrupt input pin.
52	PC4	I	General-purpose I/O port.
	SOT1		UART ch.0 serial data output pin.
53	PC5	I	General-purpose I/O port.
	SCK1		UART ch.1 serial clock I/O pin.
	TRG		16-bit PPG ch.0 to ch5 external trigger input pin.
54	PC6	I	General-purpose I/O port.
	PPG0		16-bit PPG ch.0 output pin.
	TOT1		16-bit reload timer ch.1 TOT output pin.
	IN7		Input capture ch.7 trigger input pin.
55	PC7	I	General-purpose I/O port.
	PPG1		16-bit PPG ch.1 output pin.
	TIN1		16-bit reload timer ch.1 TIN input pin.
	IN6		Input capture ch.6 trigger input pin.
24	PD0	J	General-purpose I/O port.
	SIN2		UART ch.2 serial data input pin.
25	PD1	I	General-purpose I/O port.
	SOT2		UART ch.2 serial data output pin.
26	PD2	I	General-purpose I/O port.
	SCK2		UART ch.2 serial clock I/O pin.
27	PD3	J	General-purpose I/O port.
	SIN3		UART ch.3 serial data input pin.
28	PD4	I	General-purpose I/O port.
	SOT3		UART ch.3 serial data output pin.

Table 1.6-1 Pin Functions (Sheet 7 of 7)

Pin no.	Pin name	Circuit type	Description
29	PD5	I	General-purpose I/O port.
	SCK3		UART ch.3 serial clock I/O pin.
30	PD6	I	General-purpose I/O port.
	TOT2		16-bit reload timer ch.2 TOT output pin.
56	PE0	I	General-purpose I/O port.
	TOT3		16-bit reload timer ch.3 TOT output pin.
57	PE1	I	General-purpose I/O port.
	TIN3		16-bit reload timer ch.3 TIN input pin.
64	PE2	I	General-purpose I/O port.
	SGO1		Sound generator ch.1 SGO output pin.
62	RSTO	N	Internal reset signal output pin.
65,75,85	DVCC	-	Dedicated power supply input pins for high current output buffer
66,76,86	DVSS	-	Dedicated GND power supply pins for high current output buffer
35	AVCC	-	Dedicated power supply input pin for A/D converter
38	AVSS	-	Dedicated GND power supply pin for A/D converter
36	AVRH	-	A/D converter Vref+ input pin. Vref- is fixed to AVSS.
89	MD0	D	Mode setting input pin. Connect to VCC.
88	MD1	D	Mode setting input pin. Connect to VCC.
87	MD2	D/E	Mode setting input pin. Connect to VSS.
17	C	-	External capacitor pin. Connect an 0.1 μF capacitor between this pin and VSS.
15,105	VCC	-	Power supply input pins.
16,47,106	VSS	-	GND power supply pins.

1.7 I/O Circuit Types

This section describes the types of the input/output circuits for each pin.

■ I/O Circuit Types

Table 1.7-1 shows the types of input/output circuits for each pin.

Table 1.7-1 I/O Circuit Types (Sheet 1 of 5)

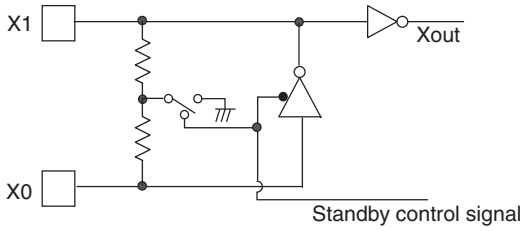
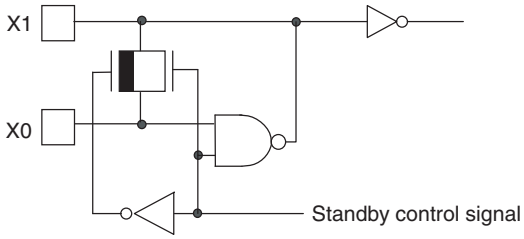
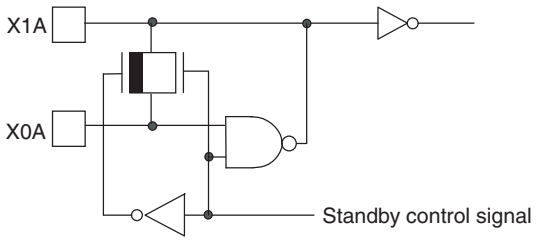
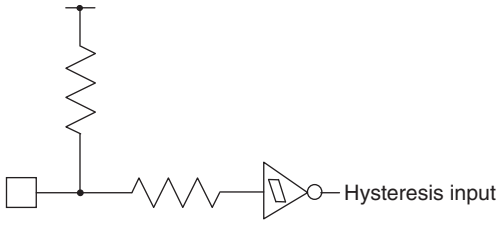
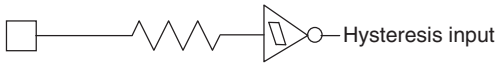
Type	Circuit	Remarks
A		High speed oscillator pin • Oscillation feedback resistance: approx. 1MΩ • (Flash Memory product/Mask ROM product)
		High speed oscillator pin • Oscillation feedback resistance: approx. 1MΩ • (EVA product)
B		Low speed oscillator pin • Oscillation feedback resistance: approx. 10MΩ
C		Input-only pin (pull-up resistor attached) • Pull-up resistor value: approx. 50kΩ • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc)
D		Input-only pin • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc)

Table 1.7-1 I/O Circuit Types (Sheet 2 of 5)

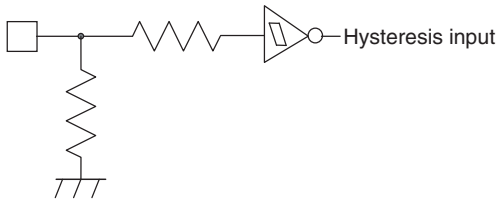
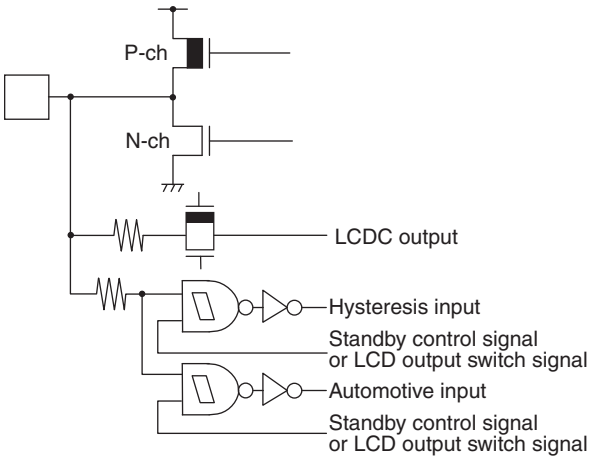
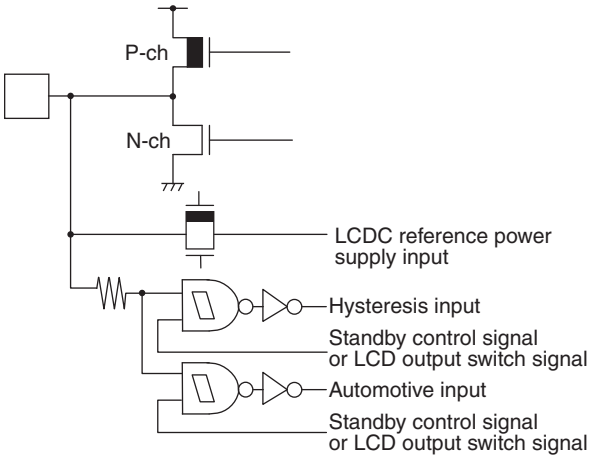
Type	Circuit	Remarks
E		Input-only pin (pull-down resistor attached) • Pull-down resistor value: approx. 50kΩ • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) Note: For the mask ROM and evaluation products only, the MD2 pin uses this circuit.
F		LCDC output/general-purpose port • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)
G		LCDC reference power supply/general-purpose port • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)

Table 1.7-1 I/O Circuit Types (Sheet 3 of 5)

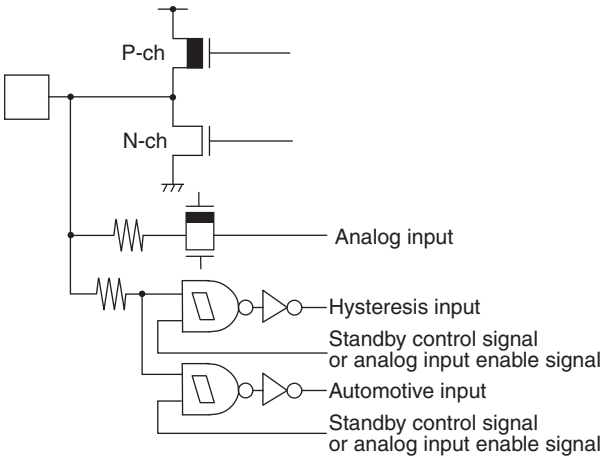
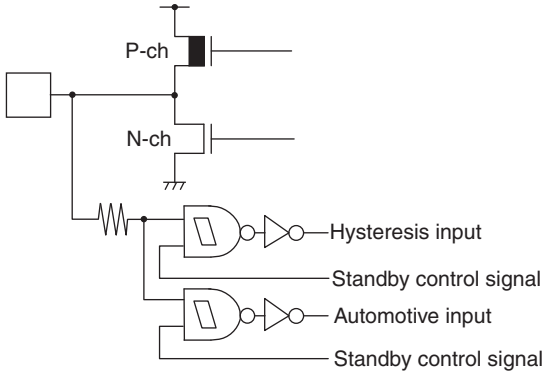
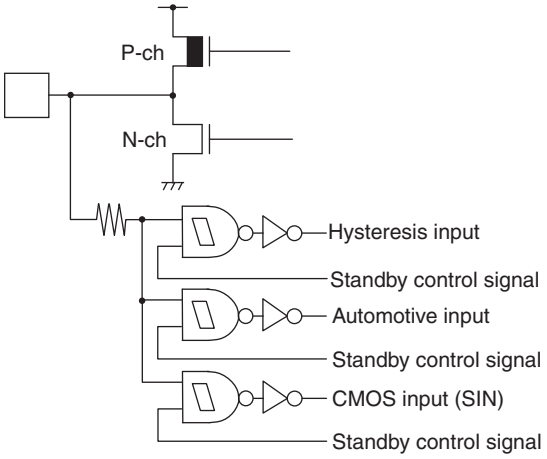
Type	Circuit	Remarks
H		A/D converter input/general-purpose port • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)
I		General-purpose port • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)
J		General-purpose port (Serial input) • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • CMOS input (SIN) (V_{IH} / V_{IL}=0.7Vcc / 0.3Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)

Table 1.7-1 I/O Circuit Types (Sheet 4 of 5)

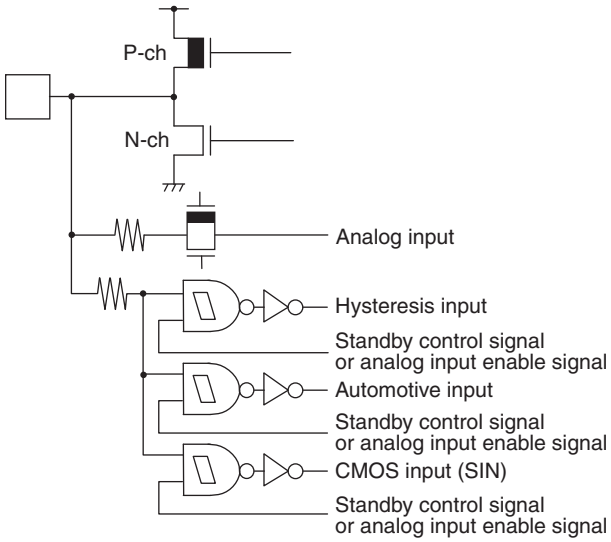
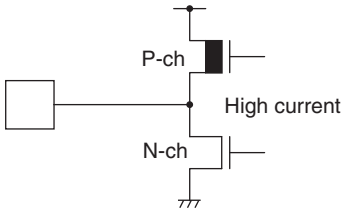
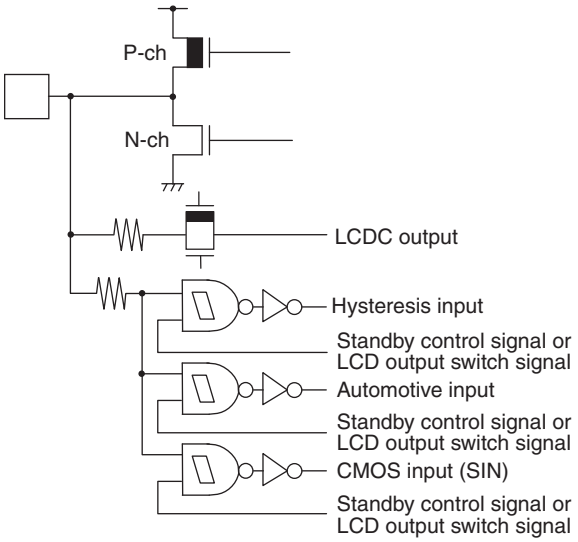
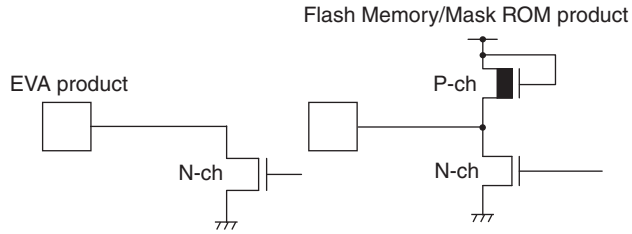
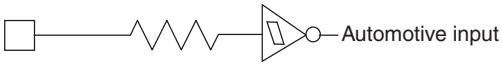
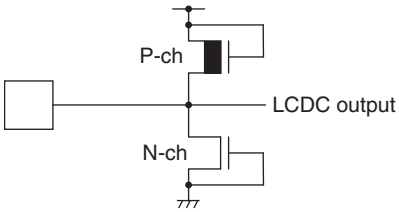
Type	Circuit	Remarks
K		A/D converter input/general-purpose port (Serial input) • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • CMOS input (SIN) (V_{IH} / V_{IL}=0.7Vcc / 0.3Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)
L		High current output port (SMC pin) • CMOS output (I_{OH} / I_{OL} = ±30 mA)
M		LCDC output/general-purpose port (Serial input) • CMOS output (I_{OH} / I_{OL} = ±4 mA) • CMOS hysteresis input (V_{IH} / V_{IL}=0.8Vcc / 0.2Vcc) • CMOS input (SIN) (V_{IH} / V_{IL}=0.7Vcc / 0.3Vcc) • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)

Table 1.7-1 I/O Circuit Types (Sheet 5 of 5)

Type	Circuit	Remarks
N		N-ch open drain pin I_{OL}=4mA
O		Input-only pin • Automotive input (V_{IH} / V_{IL}=0.8Vcc / 0.5Vcc)
P		LCDC output pin (COM pin)

1.8 Precautions for Handling Device

When handling the device, pay special attention to the following 12 points:

- **Strict observance of the maximum voltage rating (for latch-up prevention)**
- **Stabilization of supplied voltage**
- **Voltage start up time at power-on**
- **Processing of unused pins**
- **Processing of A/D converter power supply pins**
- **Use of external clocks**
- **Processing of power supply pins**
- **Power-on sequence for A/D converter power supply/analog input**
- **Handling of power supply for high current output buffer pins (DV_{CC}, DV_{SS})**
- **Pull-up/pull-down resistor**
- **Precautions when sub clock mode is not used**
- **Precautions for PLL clock mode operation**

■ Strict Observance of the Maximum Voltage Rating (for Latch-up Prevention)

Do not apply a voltage higher than V_{CC} or lower than V_{SS} to input pins and output pins of MB90920 series products. Moreover, do not apply a voltage exceeding the rating range between V_{CC} and V_{SS}. 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.

Be careful not to exceed the voltage rating of the digital power supply (V_{CC}) when turning on or turning off the analog power (AV_{CC}, AVRH), the power supply to high current output buffer pins (DV_{CC}) and the power to the analog input.

There is no distinction regarding the power-on sequence of analog power (AV_{CC}, AVRH) and the power supply to high current output buffer pins (DV_{CC}) once the digital power supply (V_{CC}) is supplied.

■ Stabilization of Supplied Voltage

Be sure to stabilize the V_{CC} power supply voltage since a rapid change in voltage may lead to incorrect operation. As a reference for the stabilization, note that the V_{CC} ripple fluctuations (p-p value) for commercial frequency (50 to 60 Hz) must be within 10% of the V_{CC} supply voltage, and the transient fluctuation rate caused by the power switching must not exceed 0.1 V/ms.

■ Voltage Start Up Time at Power-on

To prevent incorrect operation of the built-in step-down circuit, the voltage start up time at power-on must be 50µs or more (0.2V to 2.7V).

■ Processing of Unused Pins

Leaving an unused input pin open may result in incorrect operation because of external noise. In these cases, pull-up or pull-down via a resistor of at least 2kΩ or more must be applied.

An I/O pin that is not in use must be either set to output status and "open", or set to input status and handled as an input pin.

■ Processing of A/D Converter Power Supply Pins

If not used, the A/D converter must be connected so that AV_{CC}= V_{CC} and AV_{SS}=AVRH= V_{SS}.

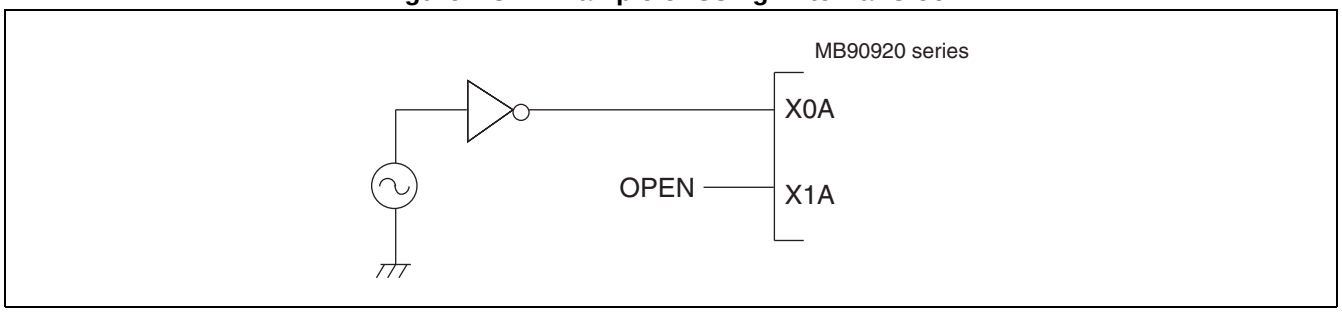
■ Use of External Clock

When an external clock is used, oscillation stabilization wait time shall be applied at recovery from power-on reset, sub clock mode and stop mode.

If an external clock is used, as shown in Figure 1.8-1, drive only the X0A pin and leave the X1A pin open.

The high-speed oscillation pins (X0, X1) cannot use the external clock input.

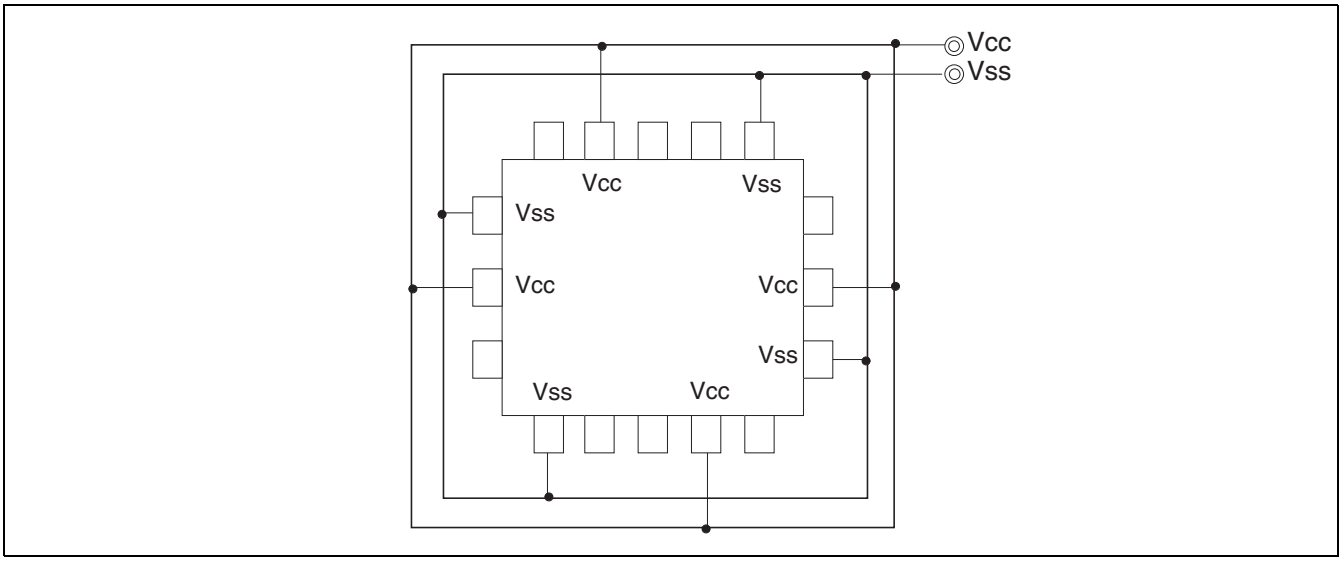
Figure 1.8-1 Example of Using External Clock



■ Processing of Power Supply Pins

To prevent a latch-up, multiple V_{CC} and V_{SS} power pins are connected within the device. However, V_{CC} and V_{SS} power pins must be connected externally to the same power supply to decrease unnecessary radiation, prevent an incorrect operation of the strobe signal due to rising ground level and maintain the total output current standard. (See Figure 1.8-2.)

Figure 1.8-2 Power Supply Pins (V_{CC}/V_{SS})



Use low impedance from the current supply source to connect with the V_{CC} and V_{SS} power pins of the device. Connect a bypass capacitor of approximately 0.1µF between V_{CC} and V_{SS} of the device, near the V_{CC} and V_{SS} power pins.

■ Power-on Sequence for A/D Converter Power Supply and Analog Input

Apply the voltage to the A/D converter power pins (AV_{CC}, AVRH) and analog input pins (AN0 to AN7) always after turning on the digital power supply (V_{CC}). To turn off the device, turn off the digital power (V_{CC}) after turning off the A/D converter and analog input power. In this case, AVRH shall not exceed AV_{CC}. When using a pin which is also used as an analog input as an input port, make sure that the input voltage does not exceed AV_{CC}.

■ Handling of Power Supply for High-current Output Buffer Pins (DV_{CC}, DV_{SS})

- Flash Memory/Mask ROM products (MB90F922/MB90922)

  As the high-current output buffer power supply (DVcc/DVss) and the digital power supply (Vcc) are isolated from each other, DVcc can be set to a potential higher than Vcc.

  Note that, if the power supply for high-current output buffer pin (DVcc/DVss) is turned on prior to the digital power supply (Vcc), however, port 7 or 8 for stepping motor output may momentarily output an "H" or "L" level signal at the rise of DVcc.

  To prevent this, turn on the digital power supply (Vcc) prior to the power supply for high-current output buffer pin.

  Apply a voltage to the power supply for high-current output buffer pin (DVcc/DVss) even when the high-current output buffer pin is used as a general-purpose port.

- EVA product (MB90V920)

  As the MB90V920 does not have the power supply for high-current output buffer (DVcc/DVss) and digital power supply (Vcc) isolated from each other, set DVcc to a potential equal to or lower than Vcc.

  Before turning on the power supply for high-current output buffer pin (DVcc/DVss), be sure to turn on the digital power supply (Vcc). Also, turn off the digital power supply (Vcc) after turning off the power supply for high-current output buffer pin. (It is acceptable to turn on or off the power supply for high-current output buffer pin and the digital power supply at the same time.)

  Apply a voltage to the power supply for high-current output buffer pin (DVcc/DVss) even when the high-current output buffer pin is used as a general-purpose port.

■ Pull-up/Pull-down Resistor

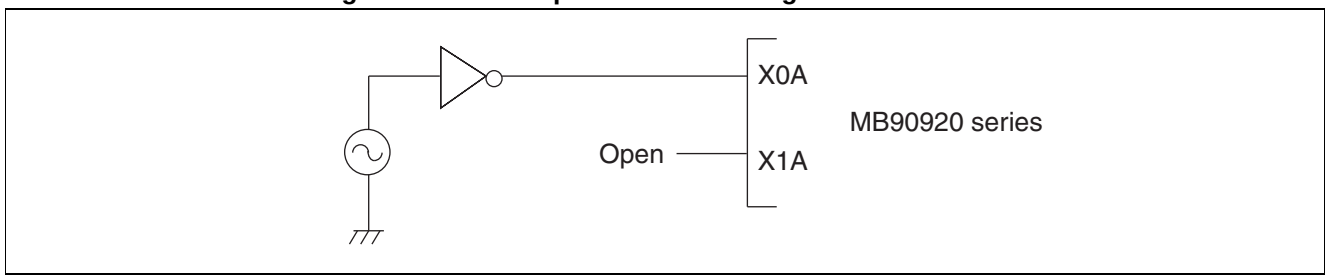
The MB90920 series supports neither internal pull-up nor pull-down resistors. Use external components if necessary.

■ Precautions when Sub Clock Mode is not Used

If no oscillator is connected to the X0A and X1A pins, apply pull-down processing to the X0A pin and leave the X1A pin open.

The following figure shows the usage example.

Figure 1.8-3 Example when not Using Sub Clock Mode



■ Precautions for PLL Clock Mode Operation

If the PLL clock mode is selected on MB90920 series, it may attempt to be working with the free-run frequency of self-oscillating circuit in the PLL when the resonator is disconnected or clock input is stopped. Performance of this operation, however, cannot be guaranteed.

■ Initialization

The device contains internal registers that are initialized only by power-on reset. If such initialization is expected, turn on the power again.

■ Flash Security Function

The security bit is located in the flash memory area. When the protection code 01_H is written to the security bit, the security function becomes active. Therefore, if not using the security function, do not write 01_H at this address.

For the address of the security bit, see "25.8 Flash Security Function".

2. CPU

This chapter describes F^{2}MC-16LX CPU.

2.1 Outline of CPU

2.2 Memory Space

2.3 Memory Map

2.4 Addressing

2.5 Allocation of Multiple-Byte Data in the Memory

2.6 Registers

2.7 Dedicated Registers

2.8 General-purpose Register

2.9 Prefix Codes

2.1 Outline of CPU

The F^{2}MC-16LX CPU core is a 16-bit CPU designed for applications in which high speed real-time processing is required, such as for various consumer devices and in vehicles. The F^{2}MC-16LX instruction set is designed for application in device controllers and is supporting a variety of control operations with high-speed and high efficiency processing.

■ CPU Features

The F^{2}MC-16LX CPU core supports not only 16-bit data but has also a 32-bit accumulator for 32-bit operations. The memory space can be a maximum of 16 MB, and can be accessed either with a linear or bank scheme. The instruction system is based on the F^{2}MC-8L A-T architecture but was further improved with the addition of high-level language compatible instructions, extensions of the addressing mode, as well as by extended instruction for multiplication, division and bit operations. The F^{2}MC-16LX CPU has the features listed below.

● Minimum instruction execution time

31.25ns (4MHz oscillation, multiply-by-8)

● Maximum memory space

16 MB, access by linear/bank schemes

● Instruction system optimized for application in controllers

- Various data types: bit/byte/word/long word
- Extended addressing mode: 23 types
- Using 32-bit accumulator for more powerful high-precision operations (support of 32-bit data), signed multiplication/ division and extended RETI instruction

● Powerful interrupt functions

Eight priority levels (programmable)

● CPU-independent automatic transfer function

Extended intelligent I/O service up to 16 channels

● Support for high-level languages (C language)/Instruction system supporting multi-task processing

Using system stack pointer/symmetric instruction sets/barrel shift instruction

● Increased execution speed

4-byte queuing

Note:

MB90920 series uses only single-chip mode, accessing only memory space of built-in ROM, built-in RAM and built-in circuits for peripherals.

2.2 Memory Space

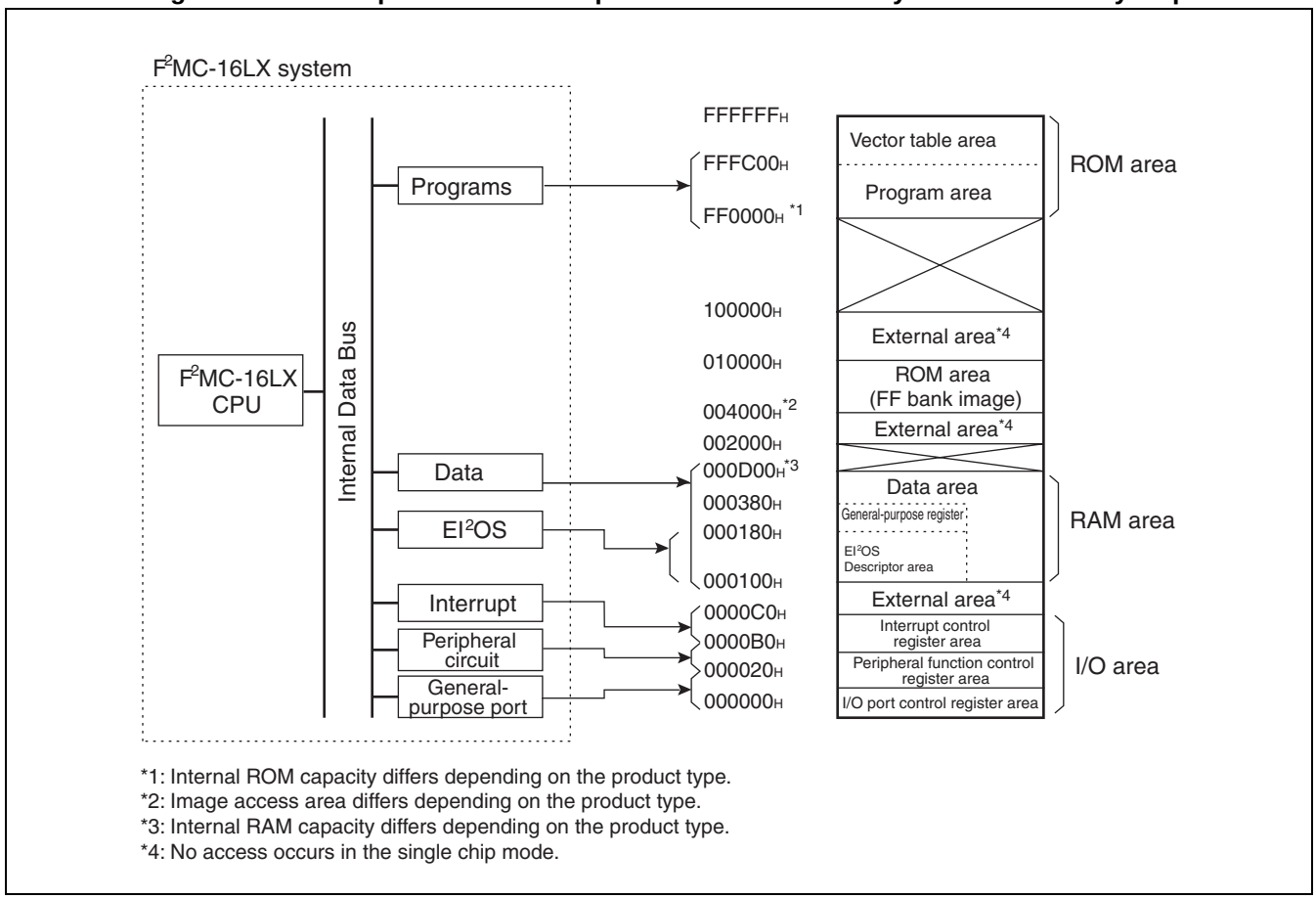
F²MC-16LX CPU has a memory space of 16 MB. The F²MC-16LX CPU controls general-purpose data, program data and I/O data, all of which are allocated within the 16 MB memory space. A part of the memory space is used for special applications, such as for extended intelligent I/O service (EI²OS) descriptors, general-purpose registers and vector tables.

■ Memory Space

General-purpose data, programs, and I/O data is allocated anywhere within the 16 MB memory space of the F²MC-16LX CPU. The CPU indicates such addresses using a 24-bit address bus to access each peripheral function.

Figure 2.2-1 shows the relationship between the F²MC-16LX system and the memory map.

Figure 2.2-1 Example of Relationship Between F²MC-16LX system and Memory Map



*1: Internal ROM capacity differs depending on the product type.
*2: Image access area differs depending on the product type.
*3: Internal RAM capacity differs depending on the product type.
*4: No access occurs in the single chip mode.

■ ROM Area

● Vector table area (Address: $FFFC00_H$ to $FFFFFF_H$)

- Used as vector tables for vector call instructions, interrupt vectors and reset vectors.
- Assigned to the highest portion of ROM area
  for setting the start address of the corresponding routine to address data in the applicable vector table.

● Program area (Address: Up to $FFFBFF_H$)

- ROM is built in as an internal program area.
- The internal ROM capacity differs depending on the product type.

■ RAM Area

● Data area (Address: from 000100_H)

- Static RAM is built-in as an internal data area.
- The internal RAM capacity differs depending on the product type.

● General-purpose register area (address: 000180_H to $00037F_H$)

- Supplemental registers are provided for 8-bit, 16-bit and 32-bit operations and transfer.
- Since this area is allocated to a part of the RAM area, it can also be used as ordinary RAM.
- When used as a general-purpose register, it allows a high-speed access with short instructions by general-purpose register addressing.

● Extended intelligent I/O service (EI^2OS) descriptor area (address: 000100_H to $00017F_H$)

- Stores transfer mode, I/O address, transfer count and buffer address.
- Since this area is allocated to a part of the RAM area, it can also be used as ordinary RAM.

■ I/O Area

● Interrupt control register area (address: $0000B0_H$ to $0000BF_H$)

Interrupt control register (ICR00 to ICR15) supports all peripheral functions that have interrupt functions, providing interrupt level settings and extended intelligent I/O service (EI^2OS) control.

● Peripheral function control register area (address: $0000C0_H$ to $0000EF_H$)

Provides control of built-in peripheral functions and data input/output.

● I/O port control register area (Address: 000000_H to $00001F_H$)

Provides I/O port control and data input/output.

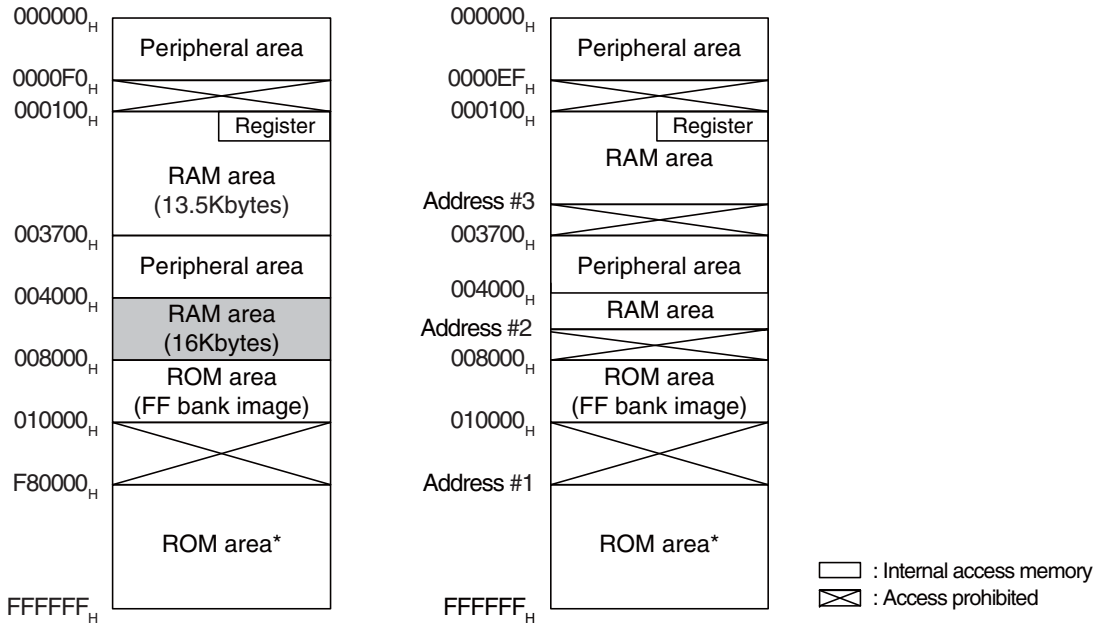
2.3 Memory Map

This section describes the memory map for the different types of MB90920 series products.

■ Memory Map

Figure 2.3-1 shows the memory map of MB90920 series.

Figure 2.3-1 Memory Map



Parts No.	ROM(FLASH) capacity	RAM capacity	Address #1	Address #2	Address #3
MB90F921 / MB90921(now planning)	128Kbytes	10Kbytes	$FE0000_H$	004000_H	002900_H
MB90F922 / MB90922	256Kbytes	10Kbytes	$FC0000_H$	004000_H	002900_H
MB90F923 / MB90923	384Kbytes	16Kbytes	$FA0000_H$	$004A00_H$	003700_H
MB90F924 / MB90924	512Kbytes	24Kbytes	$F80000_H$	$006A00_H$	003700_H

*: EVA product has no built-in ROM. This area should be as the ROM decode area of the tool.

Notes:

- If "no ROM mirror function" is selected, see "24. ROM Mirror Function Select Module".
- The upper 00 bank allows referencing ROM data in the FF bank as an image for effectively using the C compiler's small model. Because the FF bank's lower 16-bit address is set to the same value, the table in ROM can be referenced without specifying "far" with a pointer. For example, when accessing $00C000_H$, the ROM content at $FFC000_H$ is actually accessed. Since the ROM area in the FF bank exceeds 48K bytes, the whole area can not be referenced via the 00 bank image. Therefore, the ROM data in $FF4000_H$ to $FFFFFF_H$ is referenced as an image in 004000_H to $00FFFF_H$, and the ROM data table is then stored in the area $FF8000_H$ to "$FFFFFF_H$.

2.4 Addressing

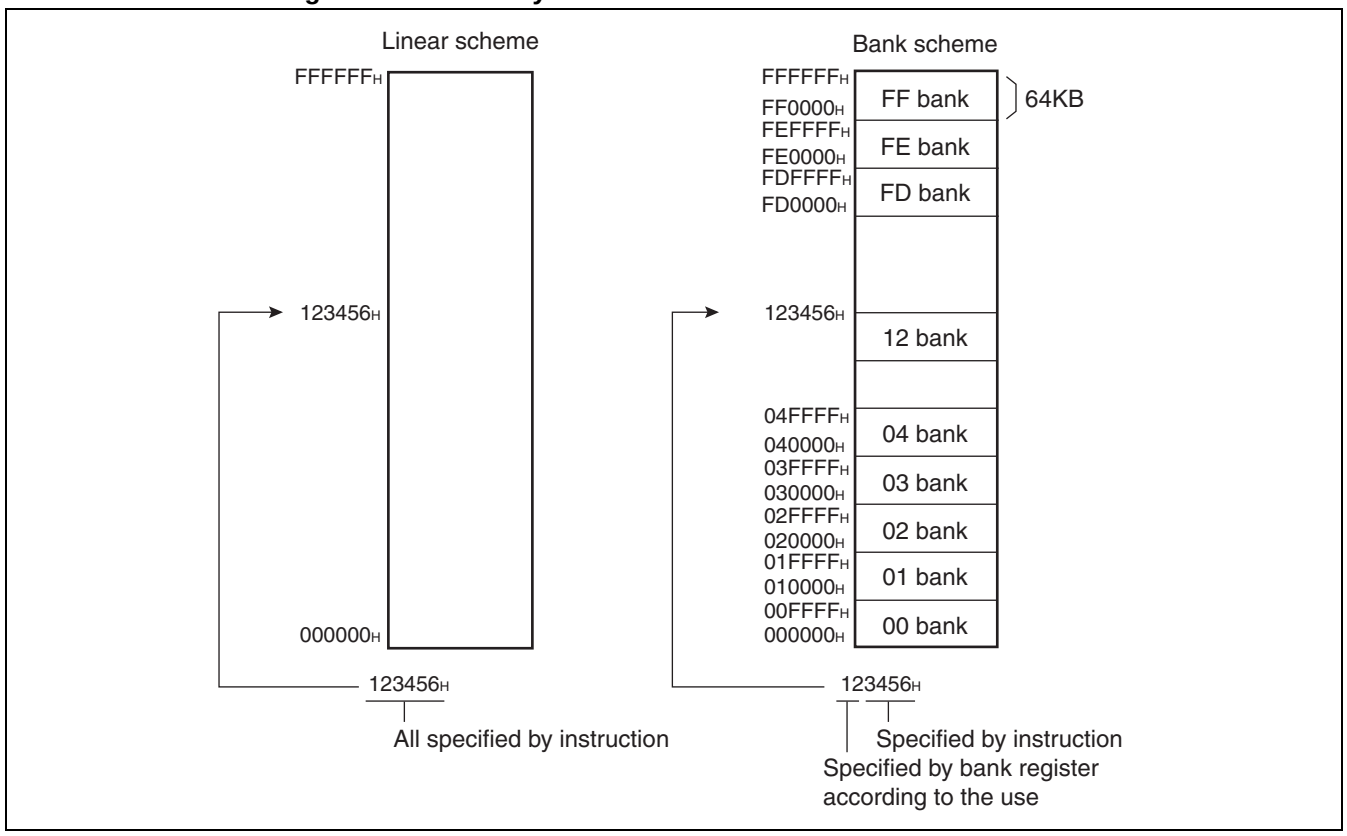
Both linear and bank address generation schemes are available.
The linear scheme is used to directly specify all 24-bit addresses within the instruction.
The bank scheme is used to specify upper 8-bit addresses via the bank register depending on how the data will be used and to specify the lower 16-bit addresses with instructions.
The F^{2}MC-16LX series basically uses bank addressing.

■ Linear Addressing and Bank Addressing

Linear scheme addressing is used to access the 16 MB space as a continuous address space. The bank scheme is used to divide and control the 16 MB space by dividing it into 256 banks of 64 K bytes each.

An outline of memory control with linear and bank schemes is shown in Figure 2.4-1.

Figure 2.4-1 Memory Control with Linear and Bank Schemes

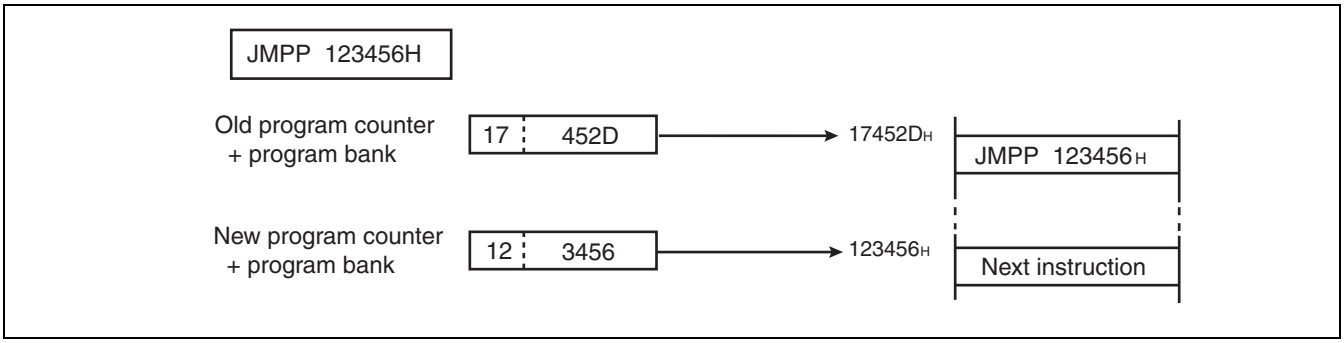


2.4.1 Addressing with Linear Scheme

There are two types of linear addressing: Directly addressing 24-bit addresses with an operand, and using the lower 24 bits of 32-bit general-purpose registers as address.

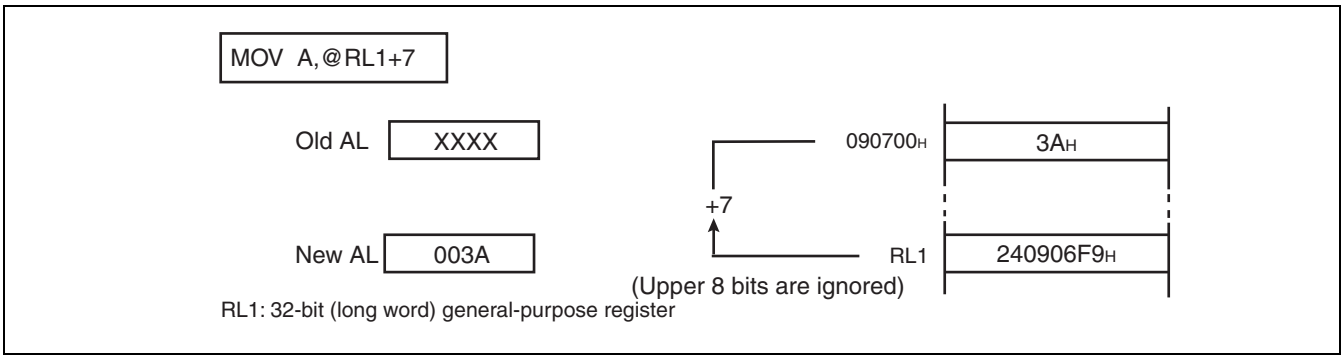
■ Specification with 24-bit Operand

Figure 2.4-2 Example of Linear Addressing (Specification with 24-bit Operand)



■ Indirect Specification with 32-bit Register

Figure 2.4-3 Example of Linear Addressing (Indirect Specification with 32-bit Register)



2.4.2 Addressing with Bank Scheme

When applying the bank scheme, the 16 MB memory space is divided into 256 banks of 64 K bytes each, and the bank address corresponding to each space is specified via a bank register. The upper 8 bits of the address are specified with the bank address and the lower 16 bits are specified with an instruction.
Five bank register types are available depending on use, as listed below.

- **Program Bank Register (PCB)**
- **Data Bank Register (DTB)**
- **User Stack Bank Register (USB)**
- **System Stack Bank Register (SSB)**
- **Additional Data Bank Register (ADB)**

■ Bank Register and Access Space

Table 2.4-1 shows the access space and major use of each bank register.

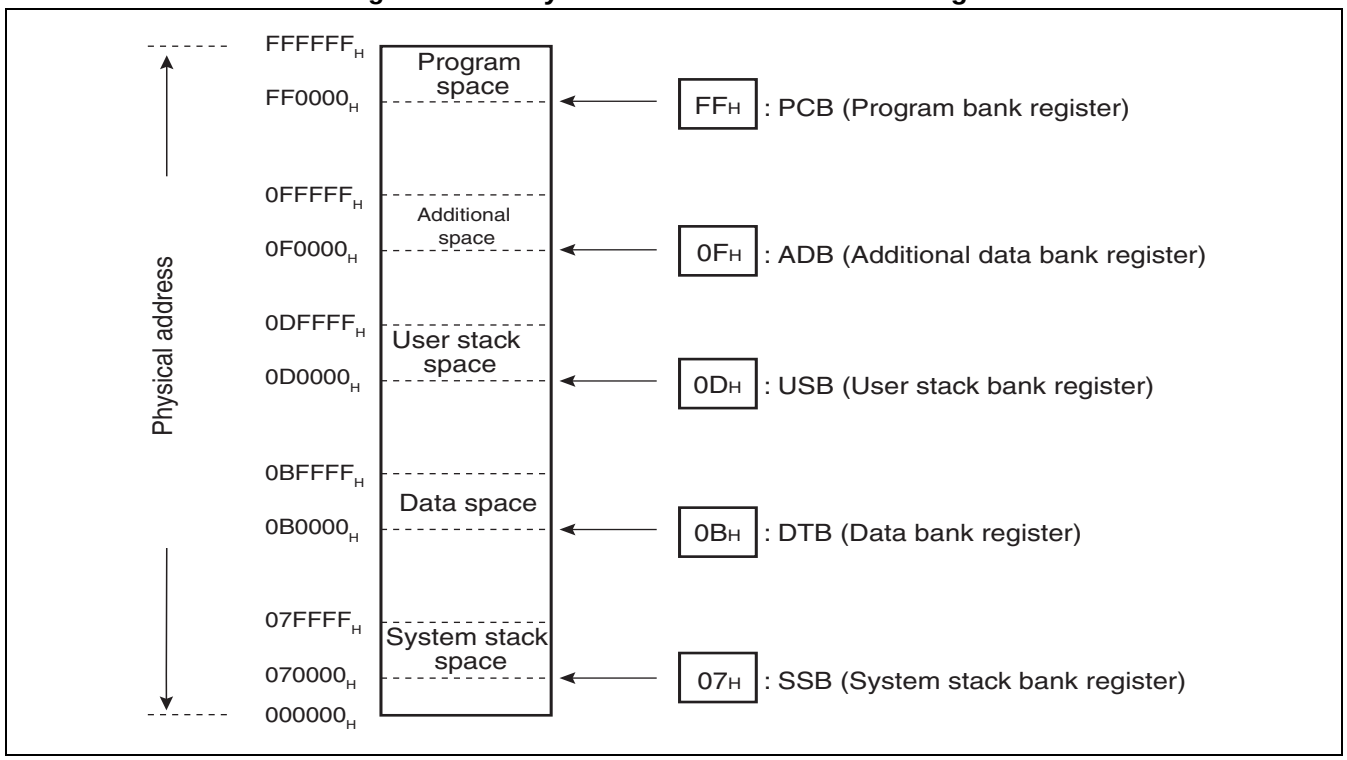
Table 2.4-1 Access Space and Major Application for Each Bank Register

Bank register name	Access space	Main use	Initial value at reset
Program bank register (PCB)	Program (PC) space	Stores instruction code, vector table and immediate data.	FF_H
Data bank register (DTB)	Data (DT) space	Stores readable/writable data and accesses the control register/data register for built-in/external peripheral components.	00_H
User stack bank register (USB)	Stack (SP) space	Area used for PUSH/POP instructions and stack accesses for storing interrupt registers. Use SSB if the stack flag (S in CCR) within the condition register is set to "1", or USB if set to "0".	00_H
System stack bank register (SSB) *			00_H
Additional data bank register (ADB)	Additional (AD) space	Stores data that cannot be fully entered in the data (DT) space.	00_H

*: SSB is used always for stacks if an interrupt occurs.

Figure 2.4-4 shows the relationship between the memory space divided into banks and each register. Refer to Section "2.7.6 Bank Registers (PCB, DTB, USB, SSB, ADB)" for details.

Figure 2.4-4 Physical Address of Each Bank Register



■ Bank Addressing and Default Space

In order to improve the efficiency of instruction code processing, a default address space is defined for each addressing scheme, as shown in Table 2.4-2. To use a space other than the default space, specify the prefix code corresponding to the bank at the beginning of the instruction; this enables accessing an arbitrary bank space corresponding to the prefix code. For the details of the prefix code, see Section "2.9 Prefix Codes".

Table 2.4-2 Addressing and Default Space

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

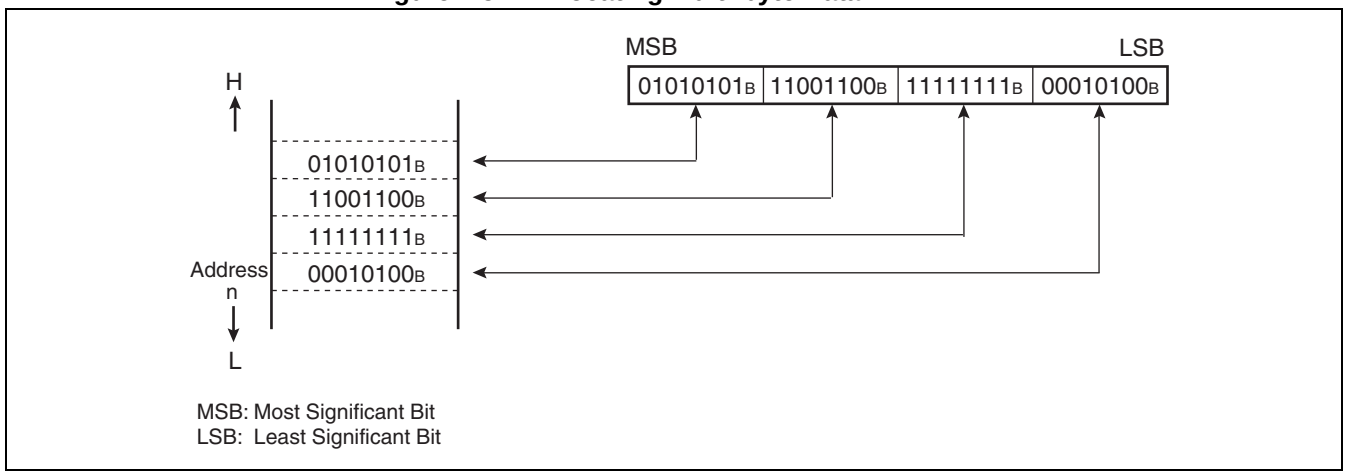
2.5 Allocation of Multiple-Byte Data in the Memory

Multiple-byte data is written to memory, starting from the lower address in sequence. For 32-bit data, first the lower 16 bits are transferred, then the upper 16 bits.If a reset signal is input immediately after writing the lower part of the data, the upper part of the data sometimes cannot be written.

■ Allocating Multi-byte Data in RAM

Figure 2.5-1 shows the placement of multiple-byte data 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.

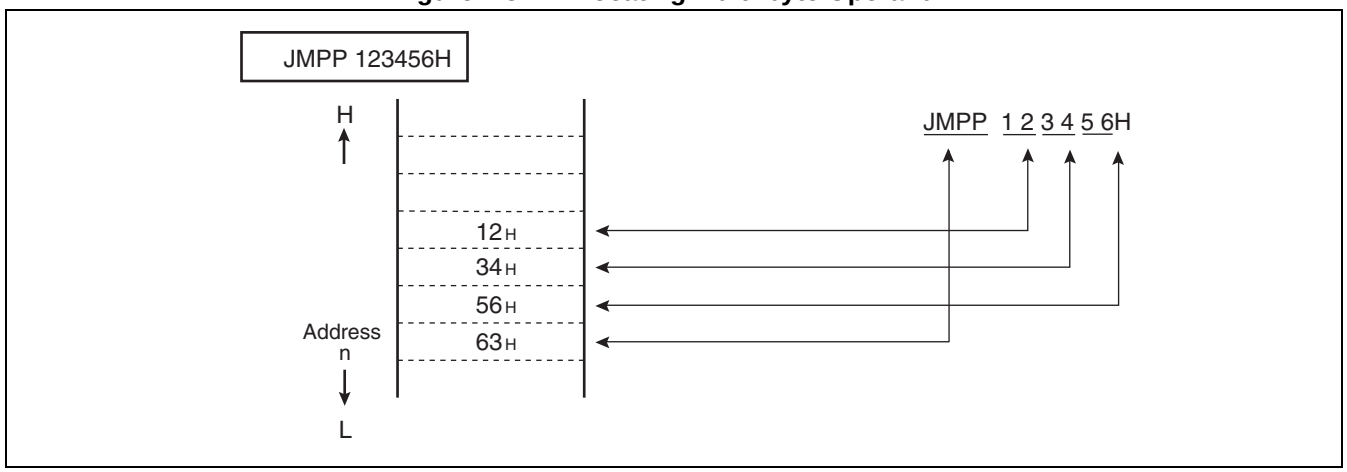
Figure 2.5-1 Allocating Multi-byte Data in RAM



■ Allocating Multi-byte Operand

Figure 2.5-2 shows the placement of a multiple-byte operand in memory.

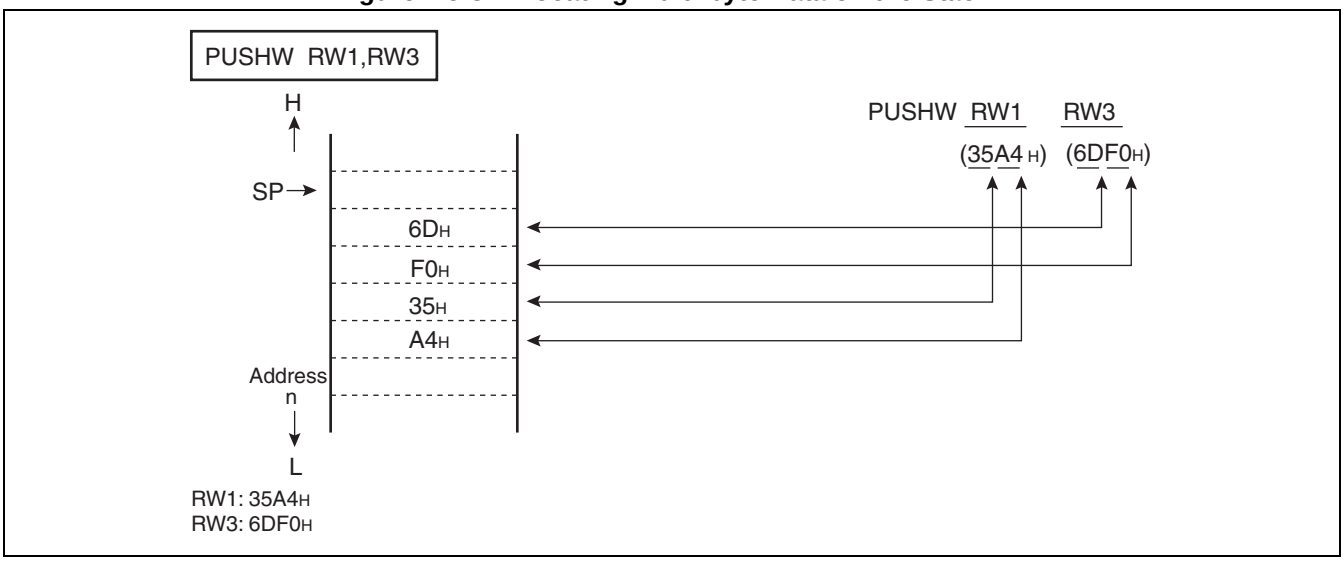
Figure 2.5-2 Allocating Multi-byte Operand



■ Allocating Multi-byte Data on the Stack

Figure 2.5-3 shows the allocation of multiple-byte data on the stack.

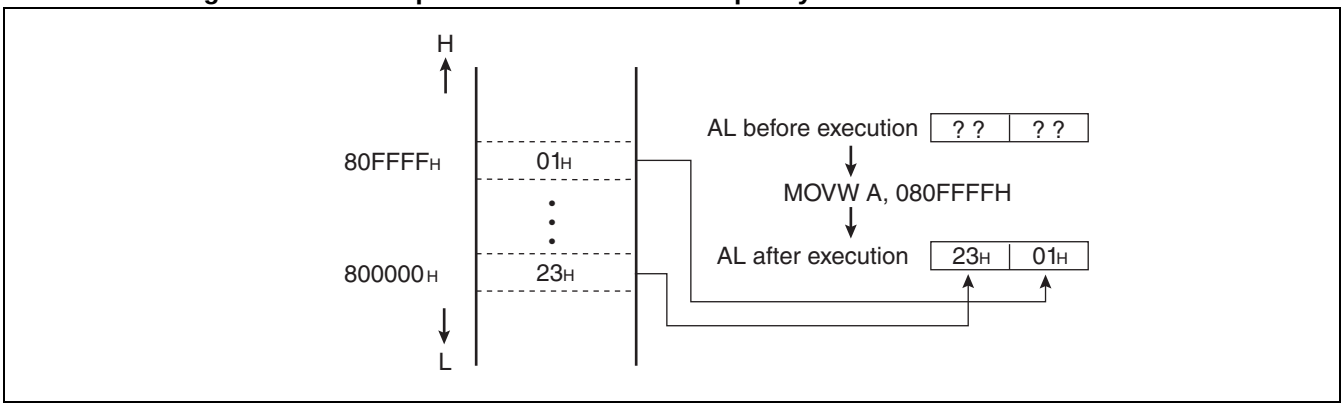
Figure 2.5-3 Allocating Multi-byte Data on the Stack



■ Accessing Multiple-Byte Data

Basically, all accesses are executed within the current bank, and the next multi-byte access instruction after the address $FFFF_H$ has been accessed will access the address 0000_H in the same bank. Figure 2.5-4 shows an example for the execution of an multiple-byte data access instruction.

Figure 2.5-4 Example for Execution of Multiple-byte Data Access Instruction



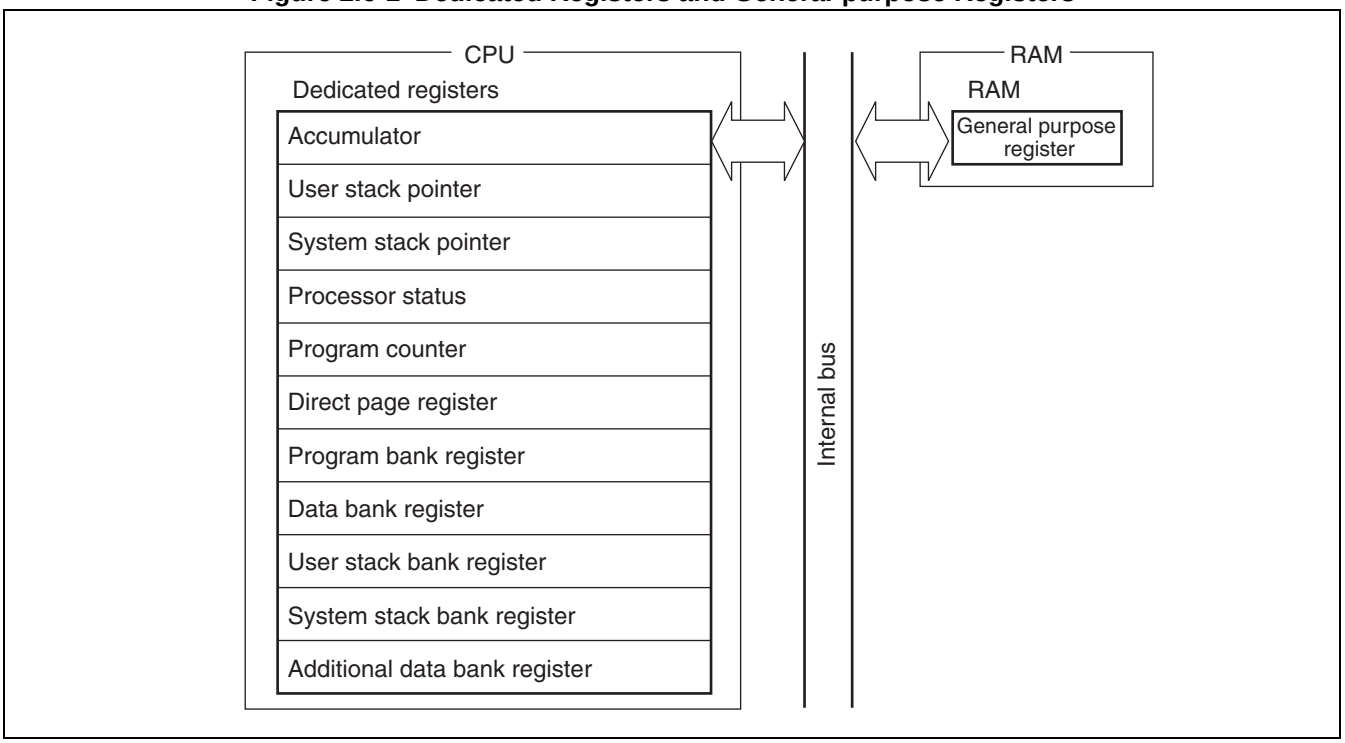
2.6 Registers

The F^{2}MC-16LX registers mostly has two types: CPU-internal dedicated registers, and general-purpose registers in the built-in RAM.

■ Dedicated Registers and General-purpose Registers

Dedicated registers consist of dedicated hardware in the CPU and their use is limited by the CPU architecture. General-purpose registers are allocated in the RAM within the same CPU address space. In addition to accessing without address specification, in the same way as a dedicated register, the use of these registers may be specified by the user in the same way as for normal memory. Figure 2.6-1 shows the allocation of the dedicated registers and the general-purpose registers in the device.

Figure 2.6-1 Dedicated Registers and General-purpose Registers



2.7 Dedicated Registers

The CPU contains the following 11 types of dedicated registers:

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

■ Configuration of Dedicated Register

Figure 2.7-1 shows the configuration of the dedicated registers, and Table 2.7-1 shows the initial values of the dedicated registers.

Figure 2.7-1 Configuration of Dedicated Registers

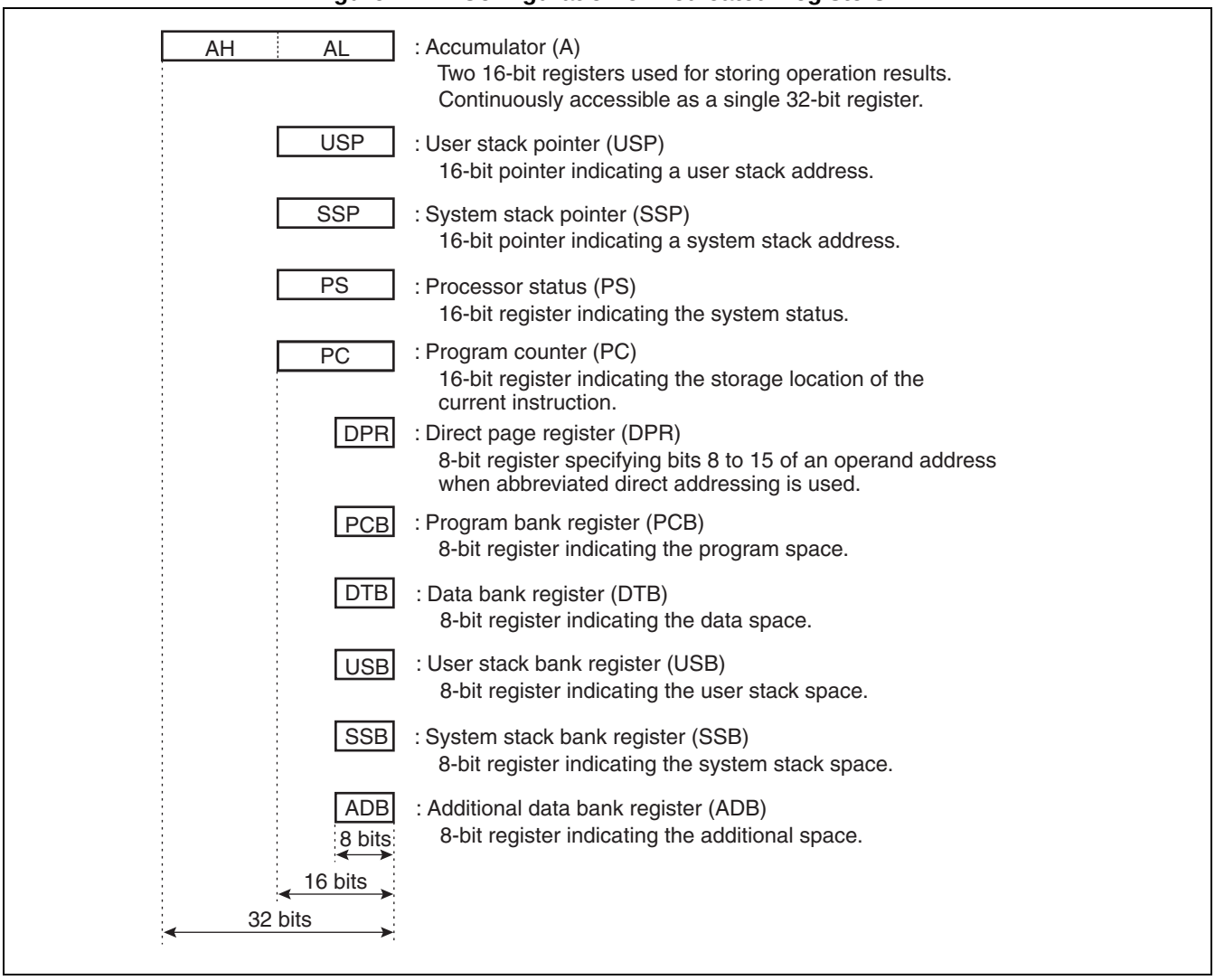


Table 2.7-1 Initial Values of Dedicated Registers

Dedicated register	Initial value
Accumulator (A)	Undefined
User stack pointer (USP)	Undefined
System stack pointer (SSP)	Undefined
Processor status (PS)	bit15 to bit13 bit12 to bit 8 bit 7 to bit 0 PS: ILM | RP | CCR 0 0 0 0 0 0 0 0 - 0 1 x x x x x - : Undefined X: Undefined value
Program counter (PC)	Value stored in the reset vector (contents of address $FFFFDC_H$ and $FFFFDD_H$)
Direct page register (DPR)	01_H

Table 2.7-1 Initial Values of Dedicated Registers

Dedicated register	Initial value
Program bank register (PCB)	Value stored in the reset vector (contents of address $FFFFDE_H$)
Data bank register (DTB)	00_H
User stack bank register (USB)	00_H
System stack bank register (SSB)	00_H
Additional data bank register (ADB)	00_H

Note:

The initial values above are used for controlling devices. Different values are to be used for ICEs (such as an emulator.)

2.7.1 Accumulator (A)

The accumulator (A) consists of two 16-bit registers (AH and AL) to temporarily store operation results or other data items.
The A register is used as a 32-/16-/8-bit register, for the purpose of executing a variety of operations between memory and other registers, or between AH and AL registers. If the data in the word length or less is transferred to the AL register, data in the AL register before transfer is automatically transferred to the AH register by the data save function (some instructions do not save data).

■ Accumulator (A)

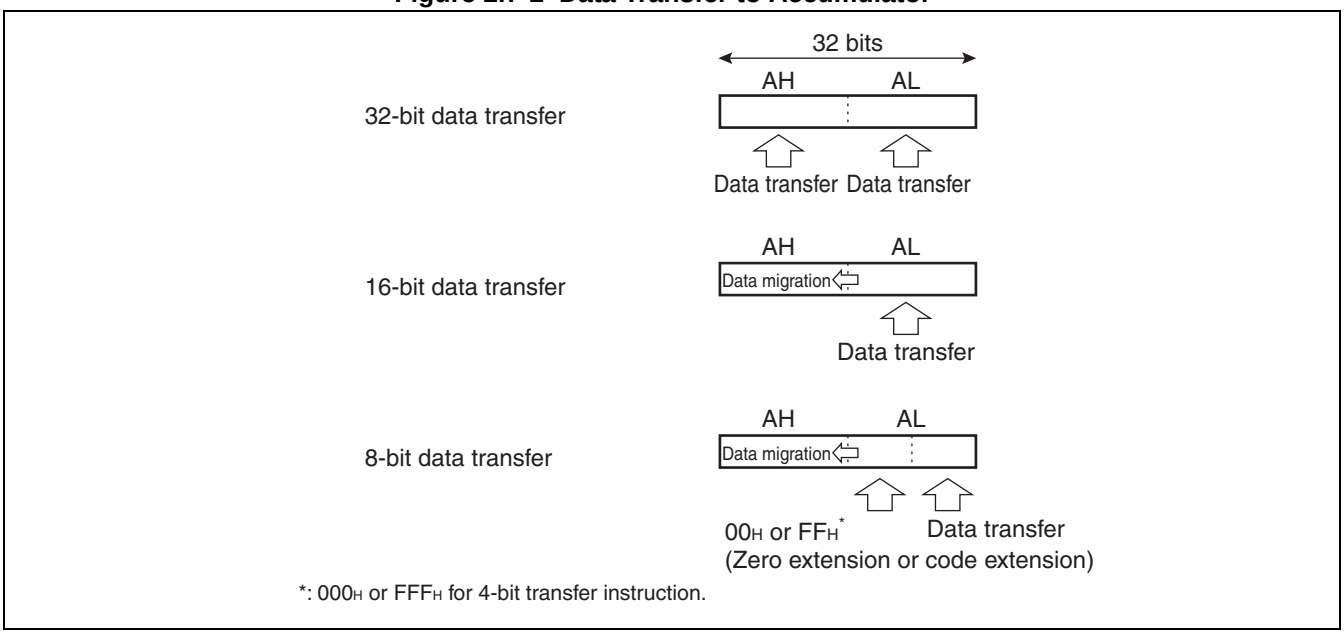
● Data transfer to the accumulator

The accumulator is used to handle 32-bit (long word), 16-bit (word) and 8-bit (byte) data. There are exceptions in which it is also used for 4-bit data transfer instructions (MOVN). The explanation of 8-bit data handling also applies to these cases.

- AH and AL registers are coupled for handling 32-bit data.
- Only the AL register is used for 16-bit or 8-bit data.
- If data of less than byte length is transferred to the AL register, it is stored in the AL register as 16-bit data by code extension or zero extension. Data in the AL register may also be handled as either word or byte data.

Data transfer to the accumulator is shown in Figure 2.7-2. An example of the actual transfer operation is shown in Figure 2.7-3 to Figure 2.7-6.

Figure 2.7-2 Data Transfer to Accumulator



- Byte-based arithmetic operations of the accumulator

When executing a byte-based arithmetic operation instruction for the AL register, the upper 8 bits of the AL register before the operation are ignored, and the upper 8 bits of the operation result are all set to "0".

- Initial value of accumulator

The initial value after a reset is undefined.

Figure 2.7-3 Example of Accumulator (A) Transfers Between AL and AH (8-bit Immediate Data, Zero Extension)

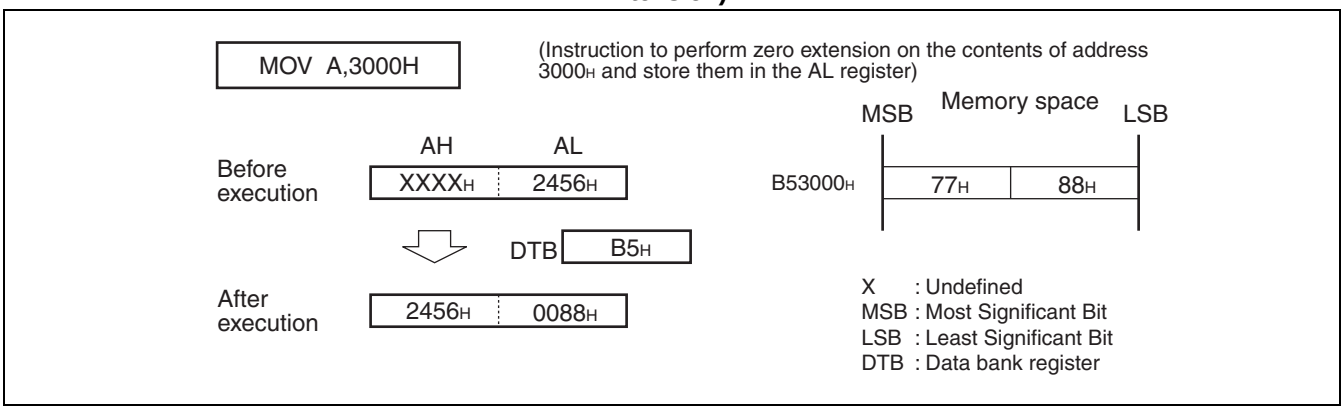


Figure 2.7-4 Example of Accumulator (A) Transfers Between AL and AH (8-bit Immediate Data, Code Extension)

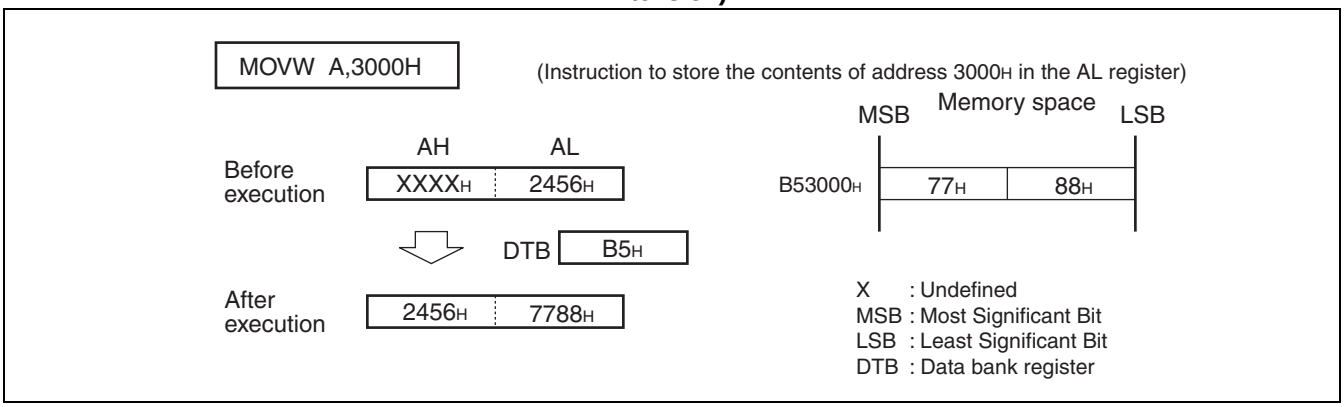


Figure 2.7-5 Example of 32-bit Data Transfer to Accumulator (A) (Register Indirect)

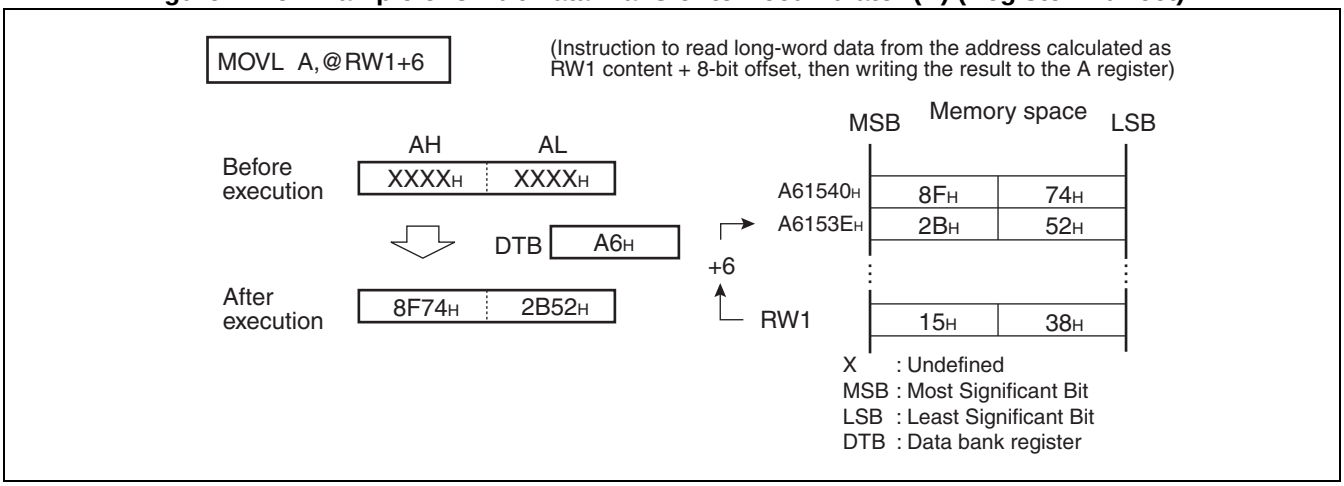
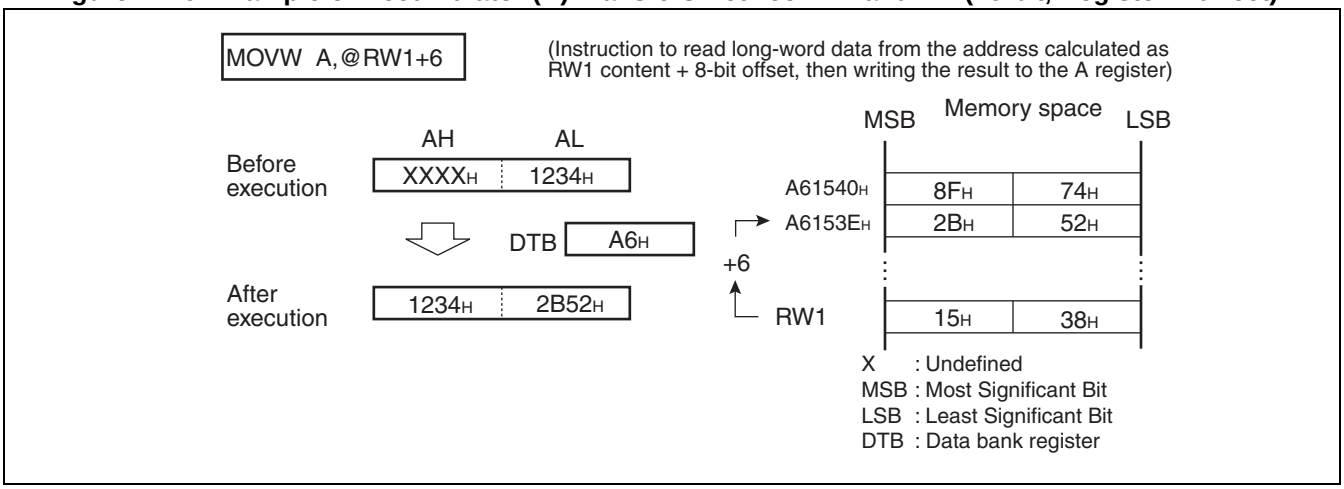


Figure 2.7-6 Example of Accumulator (A) Transfers Between AL and AH (16-bit, Register Indirect)



2.7.2 Stack Pointers (USP, SSP)

There are two types of stack pointers: a user stack pointer (USP) and a system stack pointer (SSP). These are registers used to indicate the destination address in memory for data relocation or recovery when executing the PUSH instruction, POP instruction, or subroutines. The upper 8 bits of the stack address are specified by either the user stack bank register (USB) or the system stack bank register (SSB).
If the S flag in the condition code register (CCR) is set to "0", the USP and USB registers become enabled. If the S flag is set to "1", SSP and SSB registers are enabled.

■ Stack Selection

The F^{2}MC-16LX uses two types of stacks: system stacks and user stacks. The stack address is specified with an S flag in the processor status register (PS: CCR), as shown in Table 2.7-2.

Table 2.7-2 Specifying Stack Address

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

: Initial value

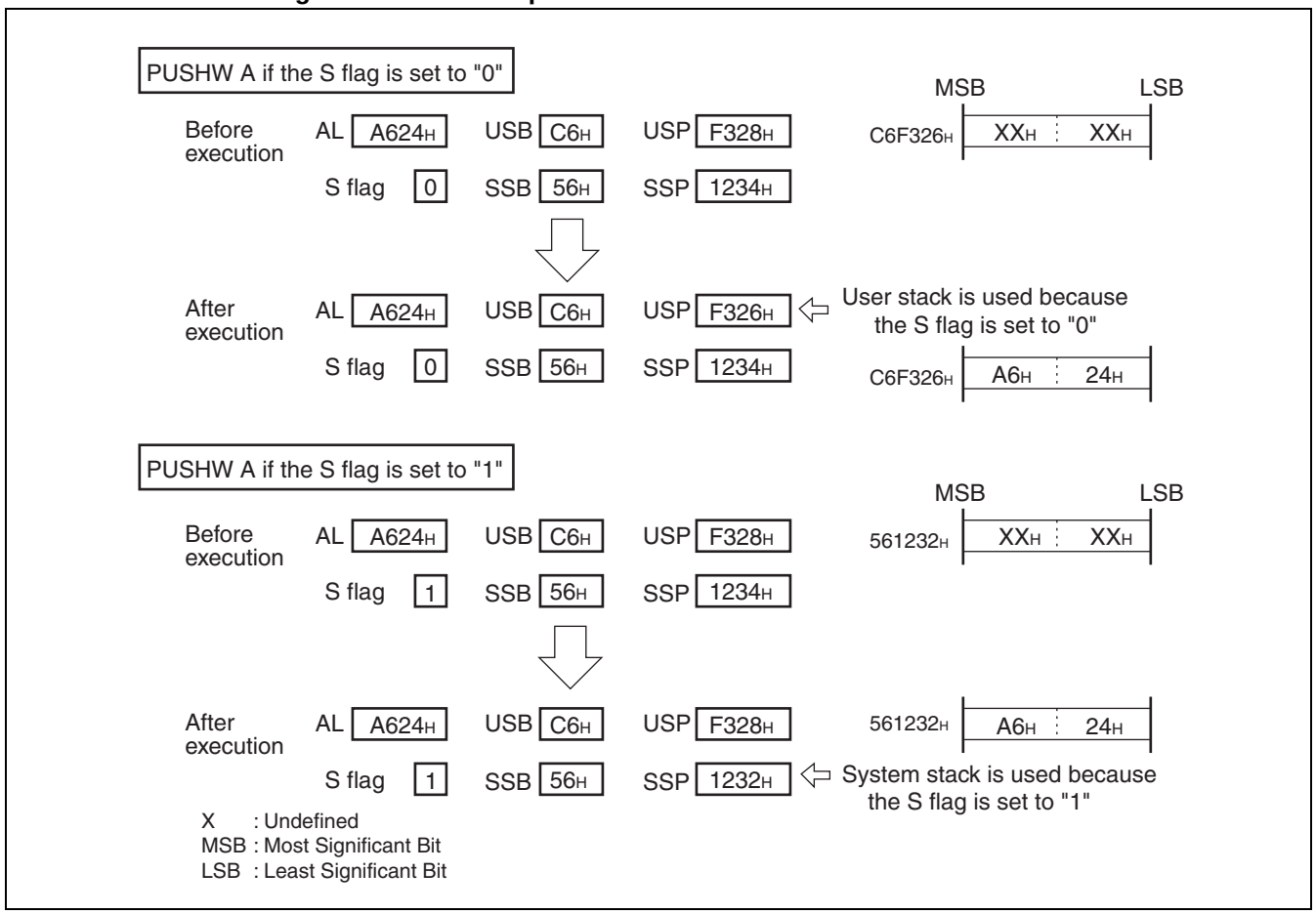
Resetting initializes the S flag to "1", after which the system stack is used by default. Normally, stack operations in an interrupt routine use the system stack, and in stack operations other than operation by an interrupt routine, the user stack is used. Especially in cases when stack space does not need to be divided, use the system stack only.

Note:

Once an interrupt is accepted, the S flag is set to "1". Thus, the system stack is always used in case of an interrupt.

Figure 2.7-7 shows an example of stack operations when the system stack is used.

Figure 2.7-7 Stack Operation Instructions and Stack Pointers



Note:

In ordinary cases, set the stack pointer to an even-numbered address. If it is set to an odd-numbered address, word accesses are performed in two steps, which degrades processing efficiency.

The initial value after resetting the USP register and SSP register is undefined.

■ System Stack Pointer (SSP)

Using the system stack pointer (SSP) requires setting the S flag in the condition code register (CCR) within the processor status register (PS) to "1". In this case, the upper 8 bits of the address used in the stack operation are indicated in the system stack bank register (SSB).

■ User Stack Pointer (USP)

Using the user stack pointer (USP) requires setting the S flag in the condition code register (CCR) within the processor status register (PS) to "0". In this case, the upper 8 bits of the address used in the stack operation are indicated in the user stack bank register (USB).

2.7.3 Processor Status (PS)

The processor status register (PS) consists of CPU control bits and a variety of bits indicating the CPU state.

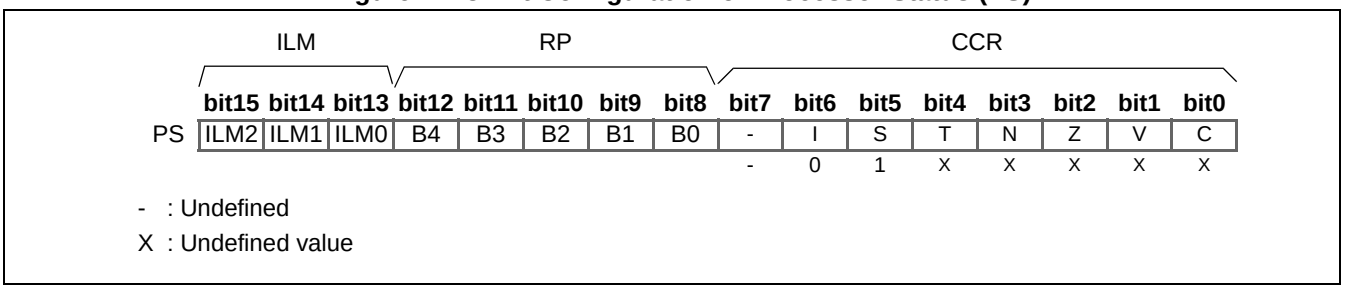
■ Bit Configuration of Processor Status (PS)

The PS register consists of three registers listed below.

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

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

Figure 2.7-8 Bit Configuration of Processor Status (PS)



● Interrupt level mask register (ILM)

Indicates the interrupt level which is currently accepted by the CPU. It is compared with the interrupt level setting bit (ICR: IL0 to IL2) in the interrupt control register, which is set corresponding to an interrupt request provided by each peripheral function.

● Register bank pointer (RP)

This pointer is used to specify the start address of the memory block (register bank) that is used as a general-purpose register in the RAM area.

The general-purpose register consists of 32 banks in total, and a bank can be specified by setting RP to a value of 0 to 31.

● Condition code register (CCR)

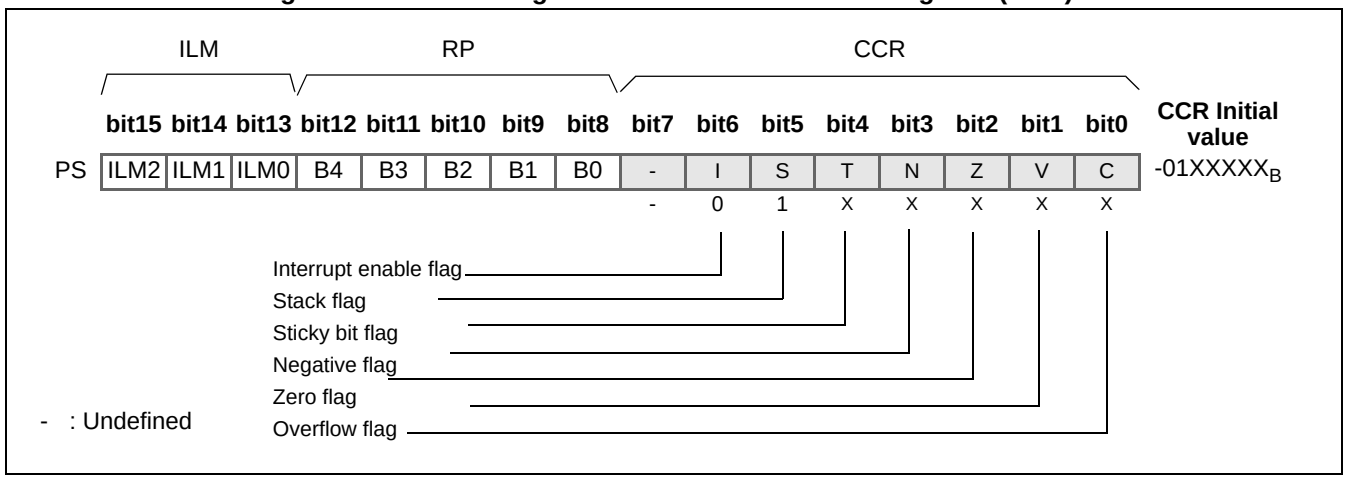
This register consists of a variety of flags, which are set "1" or reset "0" by the execution result of instructions and interrupt generation.

■ Condition Code Register (PS:CCR)

This register consists of 8 bits including bits to represent operation result and the contents of data transfers as well as the bits to control the acceptance of interrupt requests.

Figure 2.7-9 shows the bit configuration of the CCR register. For the status of the condition code register (CCR) after an instruction is executed, see the "Programming Manual".

Figure 2.7-9 Bit Configuration of Condition Code Register (CCR)



● Interrupt enable flag (I)

For interrupt requests other than software interrupts, the following applies: if the I flag is set to "1", the interrupt is permitted, and if the flag is set to "0", the interrupt is prohibited. This flag is cleared by reset.

● Stack flag (S)

This flag indicates that a pointer is used for a stack operation. If the S flag is set to "0", the user stack pointer (USP) is enabled, and if it is set to "1", the system stack pointer (SSP) is enabled. This flag is set when an interrupt is accepted or reset is performed.

● Sticky bit flag (T)

Executing a logical right shift instruction or arithmetic right shift instruction sets this flag to "1" if at least one "1" was shifted out over the carry bit.; otherwise, this flag is set to "0". This flag is also set to "0" if the shifting distance is "0".

● Negative flag (N)

Set to "1" if the highest bit of an operation result is "1". Otherwise, cleared to "0".

● Zero flag (Z)

Set to "1" if all bits of an operation result are zeroes. Otherwise, cleared to "0".

● Overflow flag (V)

Set to "1" if a signed-value overflow occurs when an operation is executed. Otherwise, cleared to "0".

- Carry flag (C)

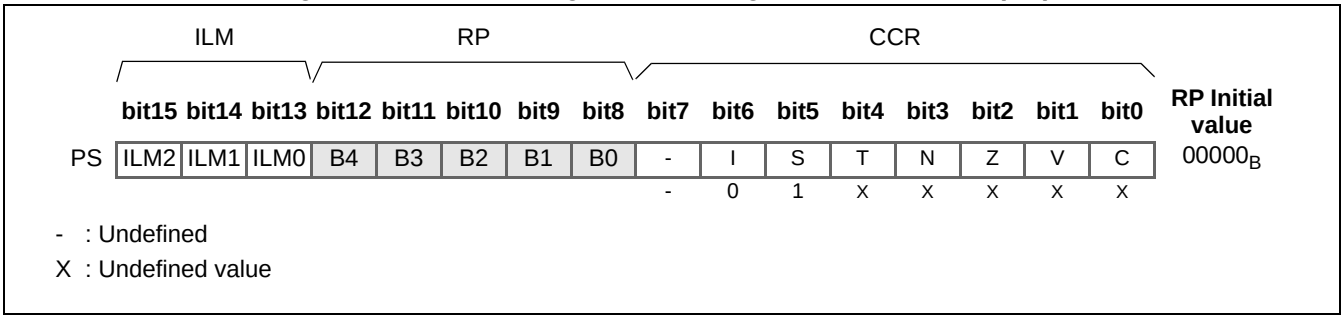
Set to "1" if, when an operation is executed, either carry-up from the highest bit or carry-down to the highest bit occurs. Otherwise, cleared to "0".

■ Register Bank Pointer (PS:RP)

Used to indicate the start address in the general-purpose register bank currently used. This pointer is used to convert the actual address at the general-purpose register addressing.

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

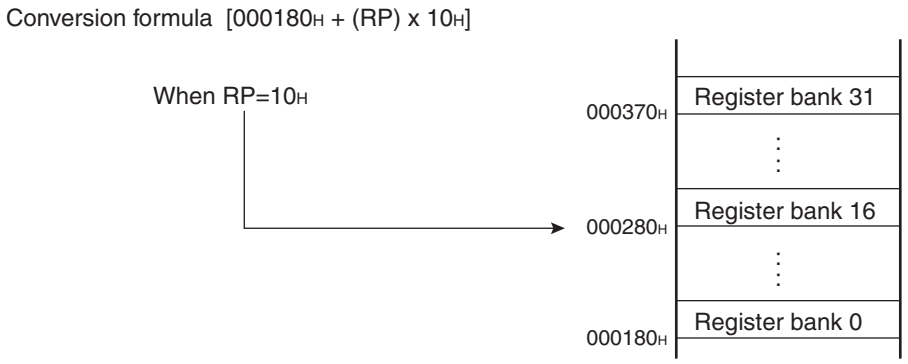
Figure 2.7-10 Bit Configuration of Register Bank Pointer (RP)



■ General-purpose Register Area and Register Bank Pointer

The register bank pointer is to indicate the relationship between the general-purpose registers in the F^{2}MC-16LX and the internal RAM addresses. For the relationship between the RP content and the actual address it indicates, see the conversion rule in Figure 2.7-11.

Figure 2.7-11 Physical Address Conversion Rule for General-purpose Register Area



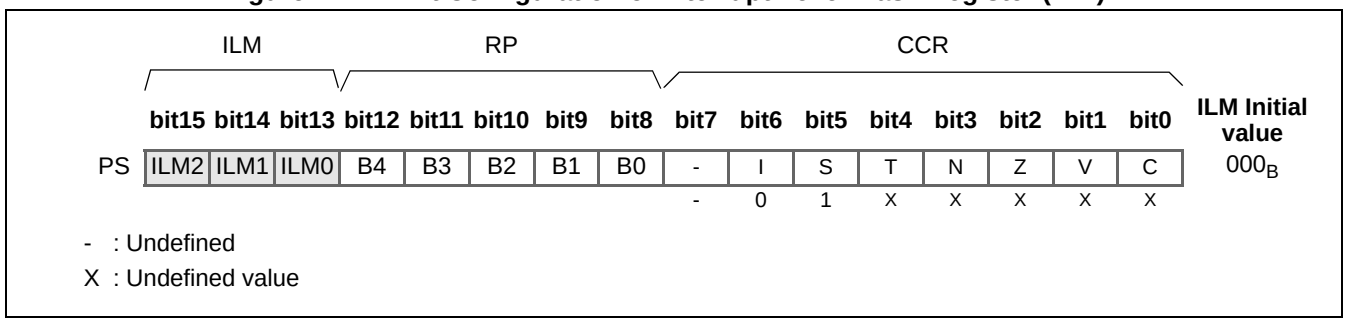
- RP can have a value from 00_H to $1F_H$. Thus, the start address of register banks can be set to a value in the range 000180_H to $00037F_H$.
- There are assembler instructions for immediate data transfer of 8 bit data to RP, but only the lower 5 bits are used in actual operation.
- The initial value of the RP register after a reset is 00_H.

■ Interrupt Level Mask Register (PS:ILM)

The interrupt level mask register (ILM) is a 3-bit register used to indicate the interrupt level the CPU can accept.

Figure 2.7-12 shows the bit configuration of the interrupt level mask register (ILM). For the details of the interrupt, see "CHAPTER 3 INTERRUPTS".

Figure 2.7-12 Bit Configuration of Interrupt Level Mask Register (ILM)



The interrupt level mask register (ILM) indicates the interrupt level currently accepted by the CPU, which is compared with the values of the bits IL0 to IL2 in the interrupt control register (ICR00 to ICR15) that are set in accordance with the interrupt request of each peripheral function. The CPU will perform interrupt processing, only if the interrupt enable flag indicates that interrupts are enabled (CCR: I=1) and the interrupt request has a value lower than indicated in these bits (the interrupt level).

- If the interrupt is accepted, the interrupt level value is set in the interrupt level mask register (ILM), and any subsequent interrupt that has the same or a lower level is not accepted.
- The interrupt level mask register (ILM) is initialized to "0" by a reset. After that, the interrupt level will be set to the highest level, indicating interrupt prohibit state.
- There are assembler instructions for immediate transfer of 8-bit data to the interrupt level mask register (ILM), but only the lower 3 bits will be used during the operation.

Table 2.7-3 Interrupt Level Mask Register (ILM) and Interrupt Level Priority

ILM2	ILM1	ILM0	Interrupt level	Interrupt level priority
0	0	0	0	High (interrupt prohibited)
0	0	1	1	↑
0	1	0	2	↕
0	1	1	3	↕
1	0	0	4	↕
1	0	1	5	↕
1	1	0	6	↓
1	1	1	7	Low

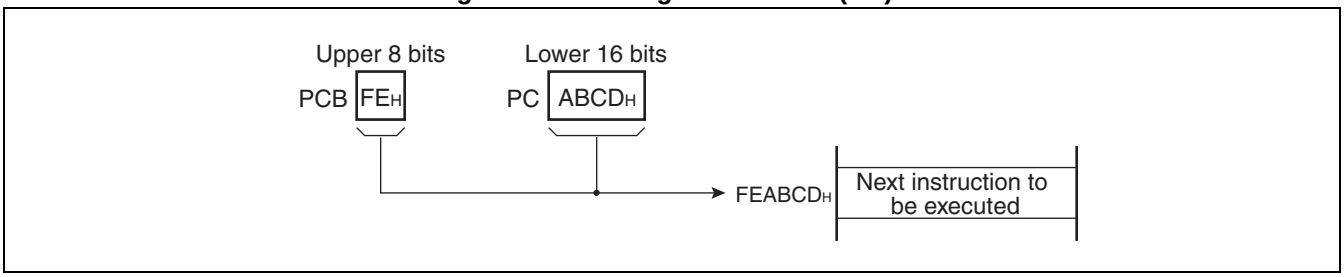
2.7.4 Program Counter (PC)

The program counter (PC) is a 16-bit counter that indicates the lower 16-bits of the address in memory at which the instruction code that the CPU will execute next is stored.

■ Program Counter (PC)

The address at which the instruction code that will be executed next by the CPU is stored consists of the upper 8 bits, specified by the program bank register (PCB), and the lower 16 bits specified by the program counter (PC). The actual address is created by combining these two parts to 24 bits, as shown in Figure 2.7-13. The content of the PC is updated by a condition branch instruction, subroutine call instruction, interrupts or resets. The PC is also used as the base pointer for reading an operand.

Figure 2.7-13 Program Counter (PC)



Note:

Neither PC nor PCB can be directly rewritten by a program (e.g. by a MOV PC, #FF instruction).

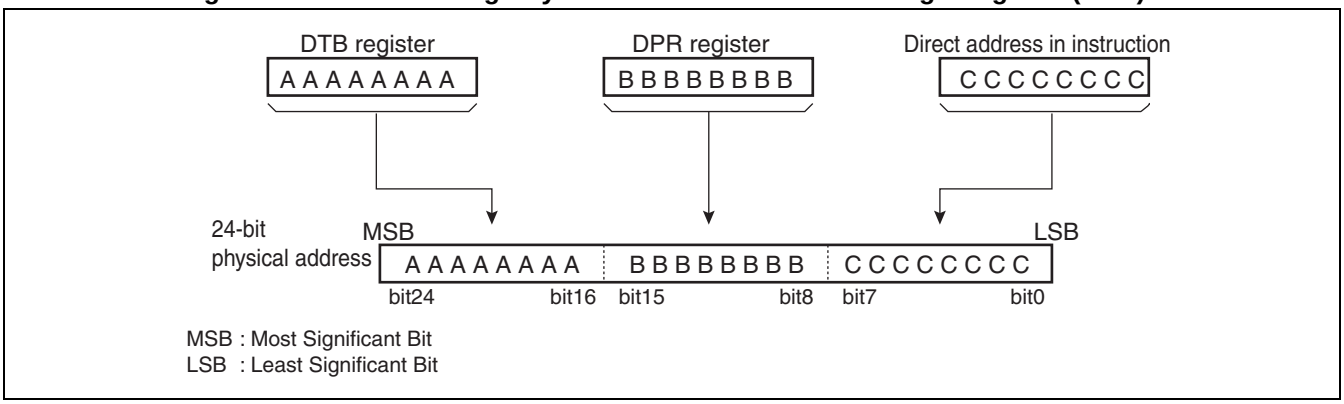
2.7.5 Direct Page Register (DPR)

The direct page register (DPR) is an 8-bit register used to specify bits 8 to 15 (addr8 to addr15) at the operand address when an instruction applying the abbreviated direct addressing scheme is executed.

■ Direct Page Register (DPR)

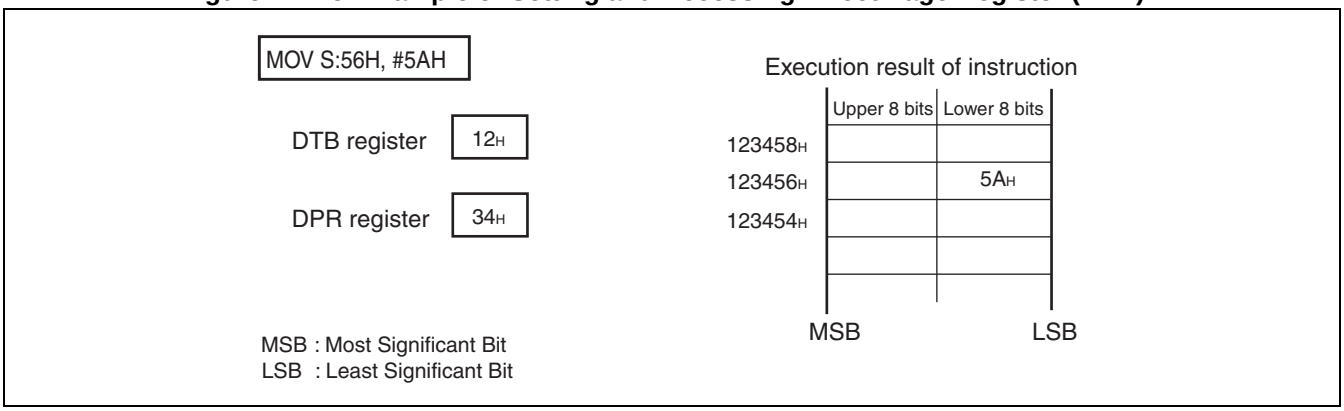
The DPR is used to specify bits 8 to 15 (addr8 to addr15) in the operand address when an instruction of the abbreviated direct addressing scheme is executed as shown in Figure 2.7-14. The DPR has a length of eight bits, and is initialized to "01_H" at a reset. Instructions can be used to read and write the register.

Figure 2.7-14 Generating Physical Address from Direct Page Register (DPR)



An example of setting and accessing the direct page register (DPR) is shown in Figure 2.7-15.

Figure 2.7-15 Example of Setting and Accessing Direct Page Register (DPR)



2.7.6 Bank Registers (PCB, DTB, USB, SSB, ADB)

The bank registers are used to specify the upper 8-bit address for bank scheme addressing. They consist of the 5 registers listed below.

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

Each bank register indicates a memory bank in which program space, data space, user stack space, system stack space, or additional space are allocated.

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

● Program bank register (PCB)

The PCB is a bank register to specify a program (PC) space. The PCB is rewritten if a JMPP, CALLP, RETP, or RETI instruction which branches over the entire 16 MB space are executed, a software interrupt instruction is executed, a hardware interrupt occurs or an exception is generated.

● Data bank register (DTB)

The DTB is a bank register used to specify a data (DT) space.

● User stack bank register (USB)/system stack bank register (SSB)

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

● Additional data bank register (ADB)

The ADB is a bank register used to specify an additional (AD) space.

● Setting each bank and data accessing

All of the bank registers have a length of bytes. PCB is initialized to FF_H by resetting, while others are initialized to 00_H. The PCB can be read but not written. Reading and writing is allowed for all bank registers other than the PCB.

Note:

The MB90920 series supports only the device built-in memory space.

For information about the operations of each register, see Section "2.4.2 Addressing with Bank Scheme".

2.8 General-purpose Register

The general-purpose register is a memory block located in RAM at the addresses 000180_H to $00037F_H$, where 16 bits × 8 are allocated per bank. It may be used as either general-purpose 8-bit register (byte register R0 to R7), 16-bit register (word register RW0 to RW7) or 32-bit register (long word register RL0 to RL3).
The general-purpose register allows high-speed RAM accesses via shorter instructions. It is allocated in blocks within the register bank, which facilitates protection of register contents and division in units of functions.
When used as a long word register, it can be used as a linear pointer for directly accessing the entire space.

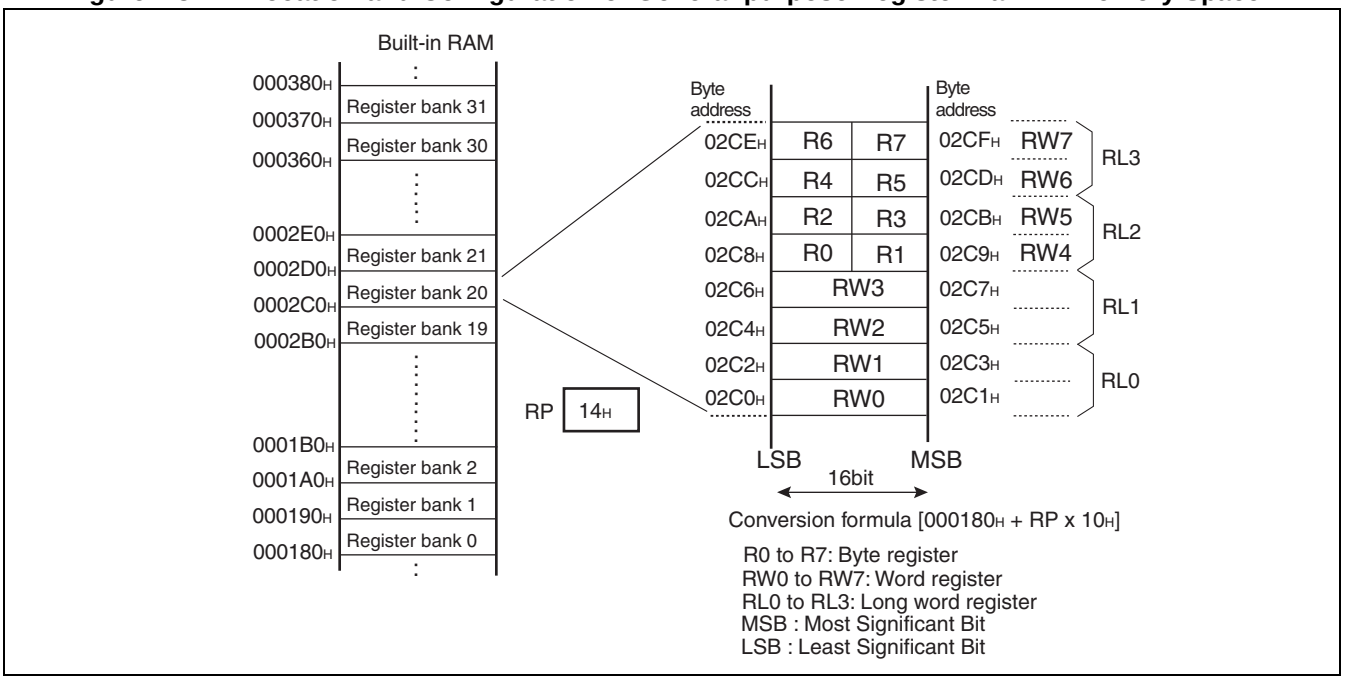
■ Configuration of General-purpose Register

The general-purpose register contains 32 banks in total which are allocated in RAM at the addresses 000180_H to $00037F_H$. The register bank pointer (RP) specifies which bank to use. Similarly, the bank currently in use can be identified by reading RP. The start address of each bank can be determined from RP as follows:

Start address of general-purpose register = $000180_H + RP \times 10_H$

Figure 2.8-1 shows the allocation and configuration of the general-purpose register bank within the memory space.

Figure 2.8-1 Allocation and Configuration of General-purpose Register Bank in Memory Space



Note:

The register bank pointer (RP) is initialized to 00_H after a reset.

■ Register Bank

The register bank consists of general-purpose registers (byte register R0 to R7, word register RW0 to RW7, long word register RL0 to RL3) used for a variety of operations and as pointers. The long word register is used as a linear pointer for directly accessing the entire memory space. The contents of the registers in the register bank is not initialized by resetting, as similar to the normal RAM, and registers retain their state before the reset. The contents of the registers at the time of power-on are undefined. Table 2.8-1 shows typical functions of the general-purpose registers.

Table 2.8-1 Typical Functions of General-purpose Registers

Register name	Function
R0 to R7	Used as an operand for various instructions. Note: R0 is also used as a counter for barrel shift and normalize instructions.
RW0 to RW7	Used as a pointer. Used as an operand for various instructions. Note: RW0 is also used as a counter for string instructions.
RL0 to RL3	Used as a long pointer. Used as an operand for various instructions.

2.9 Prefix Codes

Placing a prefix code before an instruction changes some of the operations.
The MB90920 series has three types of prefix codes:

- **Bank Select Prefix (PCB, DTB, ADB, SPB)**
- **Common Register Bank Prefix (CMR)**
- **Flag Change Suppress Prefix (NCC)**

■ Prefix Codes

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

Placing the bank select prefix before an instruction allows selecting of the memory space accessed by the instruction, irrespective of the addressing scheme.

● Common register bank prefix (CMR)

Placing the common register bank prefix before an instruction for accessing the register bank changes a register access of the instruction to the common bank located at the addresses 000180_H to $00018F_H$ (register bank selected when RP=0), irrespective of the current register bank pointer (RP) value.

● Flag Change Suppress Prefix (NCC)

Placing the prefix code for the flag change suppress flag before an instruction will avoid flag changes caused by executing the instruction.

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

The memory space used for data access is specified individually for each addressing scheme. Placing the bank select prefix before an instruction allows selection of the memory space accessed by the instruction, irrespective of the addressing scheme. Table 2.9-1 shows the bank select prefixes and the corresponding memory spaces.

Table 2.9-1 Bank Select Prefix

Bank select prefix	Space selected
PCB	Program space
DTB	Data space
ADB	Additional space
SPB	The user stack space is used if the S flag in the condition code register (CCR) is set to "0". The system stack space is used if the flag is set to "1".

If the bank select prefix is used, some instructions may execute irregular operations. Table 2.9-2 shows instructions not affected by the bank select prefix and Table 2.9-3 shows instructions that require caution when they are used with bank select prefix.

Table 2.9-2 Instructions Not Affected by Bank Select Prefix

Instruction type	Instruction	Effect of bank select prefix
String instruction	MOVS MOVSW SCEQ SCWEQ FILS FILSW	The bank register specified by the operand is used irrespective of whether a prefix is present.
Stack operation Instruction	PUSHW POPW	The user stack bank (USB) is used if the S flag is set to "0". The system stack bank (SSB) is used if the flag is set to "1", irrespective of whether the prefix is present.
I/O access instruction	MOV A, io MOVX A, io MOVW A, io MOV io, A MOVW io, A MOV io, #imm8 MOVW io, #imm16 MOVB A, io:bp MOVB io:bp, A SETB io:bp CLRB io:bp BBC io:bp, rel BBS io:bp, rel WBTC io, bp WBTS io:bp	The I/O space (Address 000000_H to $0000FF_H$) is accessed irrespective of whether the prefix is present.
Interrupt return instruction	RETI	The system stack bank (SSB) is used irrespective of whether the prefix is present.

Table 2.9-3 Instructions Requiring Caution When Bank Select Prefix is Used

Instruction type	Instruction	Description
Flag change instruction	AND CCR, #imm8 OR CCR, #imm8	The prefix also affects the subsequent instruction.
ILM setting instruction	MOV ILM, #imm8	The prefix also affects the subsequent instruction.
PS recovery instruction	POPW PS	Do not use a bank select prefix to the PS recovery instruction.

■ Common Register Bank Prefix (CMR)

To facilitate the data exchange between multiple tasks, a method must be provided for easily accessing the same register bank no matter what the value of the register bank pointer (RP) is. For this purpose, the F^2MC -16LX series provides a register bank that can be commonly used for each task. This is called the common bank. The common bank is located in the area from address 000180_H to $00018F_H$.

Placing the common register bank prefix (CMR) before an instruction for accessing a register bank allows changing register access by the instruction to the common bank located at address 000180_H to $00018F_H$ (i.e. the register bank selected when RP=0) no matter what the value of the register bank pointer (RP) is. However, the instructions listed in Table 2.9-4 should be handled with caution.

Table 2.9-4 Instructions Requiring Caution When Common Register Bank Prefix (CMR) is Used

Instruction type	Instruction	Description
String instruction	MOVS MOVSW SCEQ SCWEQ FILS FILSW	Do not add the CMR prefix to string instructions.
Flag change instruction	AND CCR, #imm8 OR CCR, #imm8	The prefix also affects the subsequent instruction.
PS recovery instruction	POPW PS	The prefix also affects the subsequent instruction.
ILM setting instruction	MOV ILM, #imm8	The prefix also affects the subsequent instruction.

■ Flag Change Suppress Prefix (NCC)

The flag change suppress prefix (NCC) is used to suppress unnecessary flag changes. Placing an NCC prefix before the instruction for which you want to suppress the flag change will prevent a flag change due to execution of the instruction. Changes can be suppressed for the flags T, N, Z, V and C. However, the instructions listed in Table 2.9-5 should be handled with caution.

Table 2.9-5 Instructions Requiring Caution When Flag Change Suppress Prefix (NCC) is Used

Instruction type	Instruction	Description
String instruction	MOVS MOVSW SCEQ SCWEQ FILS FILSW	Do not add the NCC prefix to string instructions.
Flag change instruction	AND CCR, #imm8 OR CCR, #imm8	The condition code register (CCR) changes according to the instruction specification regardless of the prefix. The prefix also affects the subsequent instruction.
PS recovery instruction	POPW PS	The condition code register (CCR) changes according to the instruction specification regardless of the prefix. The prefix also affects the subsequent instruction.
ILM Setting Instruction	MOV ILM, #imm8	The prefix also affects the subsequent instruction.
Interrupt instruction Interrupt return instruction	INT #vct8 INT9 INT adder16 INTP addr24 RETI	The condition code register (CCR) changes according to the instruction specification regardless of the prefix.
Contextual switch instruction	JCTX @A	The condition code register (CCR) changes according to the instruction specification regardless of the prefix.

■ Restrictions on Prefix Codes

The three restrictions listed below are applied when using prefix codes.

- No interrupt/hold request is accepted when a prefix code or an interrupt/hold suppress instruction is used.
- The effect of a prefix code is delayed if the prefix is placed before an interrupt/hold instruction.
- If conflicting prefix codes are placed in sequence, only the last one is effective.

Table 2.9-6 shows restrictions applying to prefix codes and interrupt/hold suppress instructions.

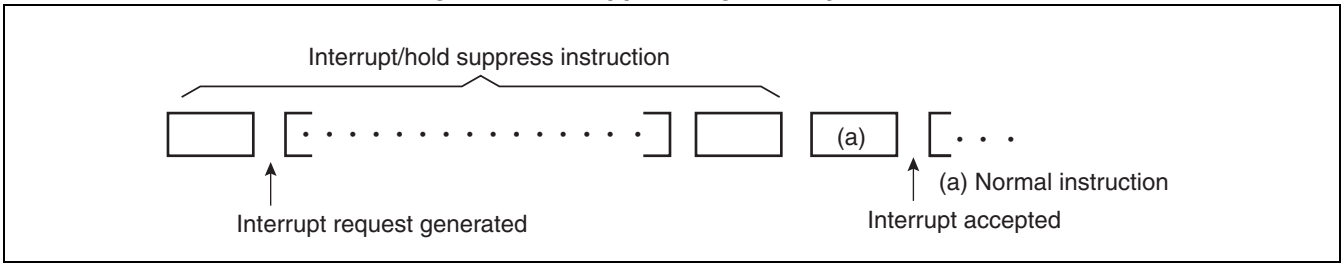
Table 2.9-6 Prefix Codes and Interrupt/Hold Suppress Instructions

	Prefix code	Interrupt/hold suppress instruction (Instruction that delays the effect of the prefix code)
Instruction that accepts neither interrupt nor hold requests	PCB DTB ADB SPB CMR NCC	MOV ILM, #imm8 OR CCR, #imm8 AND CCR, #imm8 POPW PS

● Suppressing interrupt/hold

As shown in Figure 2.9-1, while prefix code or interrupt/hold instruction are executed, any generated interrupt hold requests are not accepted. In such cases, interrupt/hold processing is not performed until another instruction has been executed after the one with the prefix code or the interrupt/hold suppress instruction.

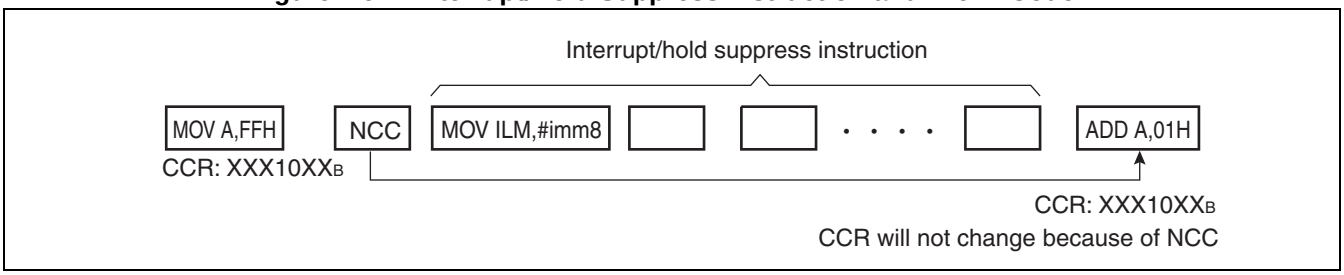
Figure 2.9-1 Suppressing Interrupt/Hold



● Delayed effect of prefix codes

If, as shown in Figure 2.9-2, a prefix code is placed before an interrupt/hold suppress instruction, the prefix code becomes effective with the first instruction after the interrupt/hold suppress instruction is issued.

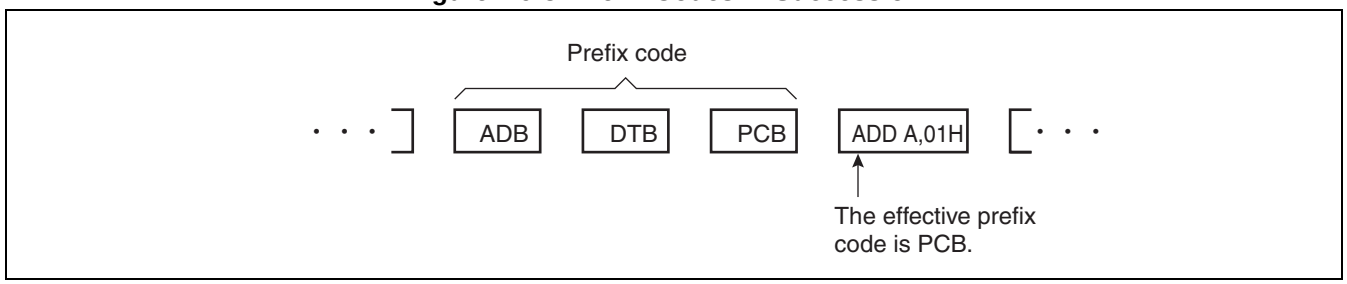
Figure 2.9-2 Interrupt/Hold Suppress Instruction and Prefix Code



- Prefix codes in succession

If, as shown in Figure 2.9-3, conflicting prefix codes (PCB, ADB, DTB, SPB) are specified in sequence, only the last one is effective.

Figure 2.9-3 Prefix Codes in Succession



3. Interrupt

This chapter describes the relationships between interrupts and the extended intelligent I/O service (EI^{2}OS).

3.1 Outline of Interrupts

3.2 Interrupt Sources and Interrupt Vectors

3.3 Interrupt Control Registers and Peripheral Functions

3.4 Hardware Interrupt

3.5 Software Interrupt

3.6 Interrupt by Extended Intelligent I/O Service (EI^{2}OS)

3.7 Exception Handling Interrupt by Execution of Undefined Instruction

3.8 Stack Operations of Interrupt Handling

3.9 Example Program for Interrupt Handling

3.1 Outline of Interrupts

F^{2}MC-16LX has 4 types of interrupt functions to interrupt the processing currently being performed and transferring the control to a separately defined program if an event occurs.

- **Hardware interrupt**
- **Software Interrupt**
- **Extended intelligent I/O service (EI^{2}OS) interrupt**
- **Exception handling**

■ Types of Interrupts and Corresponding Functions

● Hardware interrupt

Transfers control to a user-defined interrupt handling program in response to an interrupt request from a peripheral function.

● Software Interrupt

Transfers control to a user-defined interrupt-handling program by executing a software interrupt dedicated instruction (e.g. INT instruction).

● Extended intelligent I/O service (EI^{2}OS) interrupt

EI^{2}OS is a function for automatic data transfer between a peripheral function and memory. Data transfer as far as previously performed by the interrupt-handling program is provided in the same way as via DMA (direct memory access). After the completion of data transfer of the specified count, the interrupt handling program is automatically executed.

Interrupts from EI^{2}OS are a type of hardware interrupt.

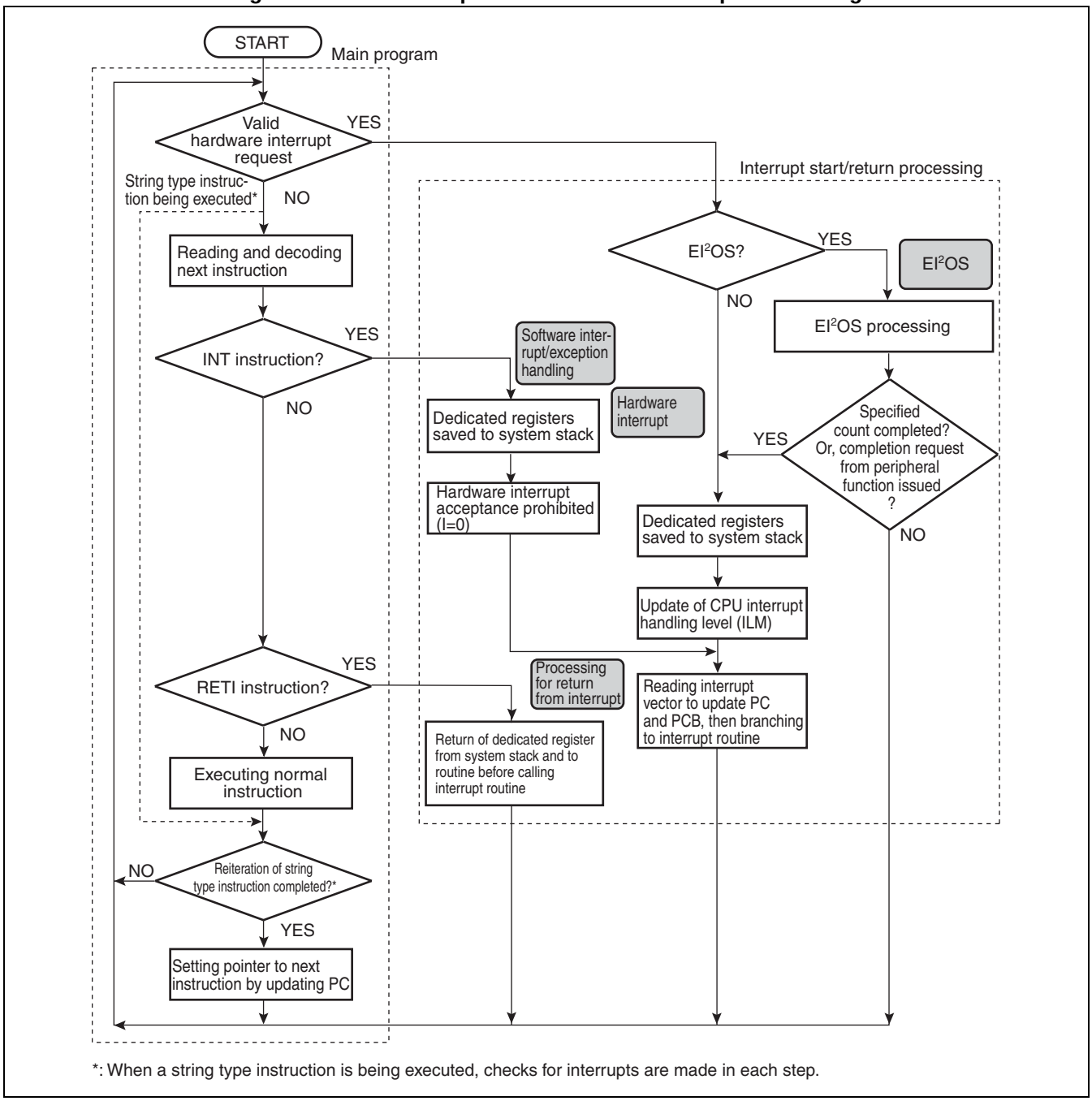
● Exception handling

Exception handling is basically performed in the same way as interrupt handling. If an exception event (execution of an undefined function) is detected at the instruction boundary, the normal course of processing is interrupted. This is equivalent to an "INT10" software interrupt instruction.

■ Interrupt Operation

F^{2}MC-16LX has 4 types of interrupt functions for starting and returning processing, as shown in Figure 3.1-1.

Figure 3.1-1 Overall Operational Flow of Interrupt Processing



*: When a string type instruction is being executed, checks for interrupts are made in each step.

3.2 Interrupt Sources and Interrupt Vectors

F^2MC-16LX has functions corresponding to 256 types of interrupt sources. There are 256 interrupt vector tables allocated starting with the highest address in memory. The interrupt vectors are shared by all interrupts.
Software interrupts may use all of the above interrupts (INT0 to INT256), but some of these interrupt vectors will be shared by hardware interrupts and exception handling interrupts. Only specific interrupt vectors and a interrupt control register (ICR) are available for setting the hardware interrupts for each peripheral function.

■ Interrupt Vector

Interrupt vector tables, which are referenced for interrupt handling, are allocated starting with the highest address of the memory area ($FFFC00_H$ to $FFFFFF_H$). The interrupt vectors for EI^2OS, exception handling, hardware interrupts, and software interrupts share the same area. The allocation of interrupt numbers and interrupt vectors in memory is shown in Table 3.2-1.

Table 3.2-1 List of Interrupt Vectors

Software interrupt instruction	Vector address L	Vector address M	Vector address H	Mode data	Interrupt No.	Hardware interrupt
INT0	$FFFFFC_H$	$FFFFFD_H$	$FFFFFE_H$	Unused	#0	None
:	:	:	:	:	:	:
INT7	$FFFFE0_H$	$FFFFE1_H$	$FFFFE2_H$	Unused	#7	None
INT8	$FFFFDC_H$	$FFFFDD_H$	$FFFFDE_H$	$FFFFDF_H$	#8	(RESET vector)
INT9	$FFFFD8_H$	$FFFFD9_H$	$FFFFDA_H$	Unused	#9	None
INT10	$FFFFD4_H$	$FFFFD5_H$	$FFFFD6_H$	Unused	#10	<Exception handling>
INT11	$FFFFD0_H$	$FFFFD1_H$	$FFFFD2_H$	Unused	#11	Hardware interrupt #0
INT12	$FFFFCC_H$	$FFFFCD_H$	$FFFFCE_H$	Unused	#12	Hardware interrupt #1
INT13	$FFFFC8_H$	$FFFFC9_H$	$FFFFCA_H$	Unused	#13	Hardware interrupt #2
INT14	$FFFFC4_H$	$FFFFC5_H$	$FFFFC6_H$	Unused	#14	Hardware interrupt #3
:	:	:	:	:	:	:
INT254	$FFFC04_H$	$FFFC05_H$	$FFFC06_H$	Unused	#254	None
INT255	$FFFC00_H$	$FFFC01_H$	$FFFC02_H$	Unused	#255	None

Note:

We recommend allocating also unused interrupt vectors at the addresses for exception handling.

■ Interrupt Sources and Interrupt Vectors/Interrupt Control Registers

Table 3.2-2 shows the relationship between interrupt sources and the interrupt vectors/interrupt control registers except for software interrupts.

Table 3.2-2 Interrupt Sources and Interrupt Vectors/Interrupt Control Registers

Interrupt source	EI^{2}OS support	Interrupt vector			Interrupt control register		Priority *2
		Number		Address	ICR	Address	
Reset	×	#08	08_H	$FFFFDC_H$	-	-	High
INT9 instruction	×	#09	09_H	$FFFFD8_H$	-	-	
Exception handling	×	#10	$0A_H$	$FFFFD4_H$	-	-	
CAN0 RX/CAN2 RX	×	#11	$0B_H$	$FFFFD0_H$	ICR00	$0000B0_H$ *1	
CAN0 TX/NS / CAN2 TX/NS	×	#12	$0C_H$	$FFFFCC_H$			
CAN1 RX/CAN3 RX	×	#13	$0D_H$	$FFFFC8_H$	ICR01	$0000B1_H$ *1	
CAN1 TX/NS /CAN3 TX/NX /SIO	×	#14	$0E_H$	$FFFFC4_H$			
Input capture 0	△	#15	$0F_H$	$FFFFC0_H$	ICR02	$0000B2_H$ *1	
DTP/external interrupt - ch.0/ch.1 detected	△	#16	10_H	$FFFFBC_H$			
Reload timer 0	△	#17	11_H	$FFFFB8_H$	ICR03	$0000B3_H$ *1	
Reload timer 2	△	#18	12_H	$FFFFB4_H$			
Input capture 1	△	#19	13_H	$FFFFB0_H$	ICR04	$0000B4_H$ *1	
DTP/external interrupt - ch.2/ch.3 detected	△	#20	14_H	$FFFFAC_H$			
Input capture 2	△	#21	15_H	$FFFFA8_H$	ICR05	$0000B5_H$ *1	
Reload timer 3	△	#22	16_H	$FFFFA4_H$			
Input capture 3/4/5/6/7	△	#23	17_H	$FFFFA0_H$	ICR06	$0000B6_H$ *1	
DTP/external interrupt - ch.4/ch.5 detected, UART3 RX	△	#24	18_H	$FFFF9C_H$			
PPG timer 0	△	#25	19_H	$FFFF98_H$	ICR07	$0000B7_H$ *1	
DTP/external interrupt - ch.6/ch.7 detected, UART3 TX	△	#26	$1A_H$	$FFFF94_H$			
PPG timer 1	△	#27	$1B_H$	$FFFF90_H$	ICR08	$0000B8_H$ *1	
Reload timer 1	△	#28	$1C_H$	$FFFF8C_H$			
PPG timer 2/3/4/5	○	#29	$1D_H$	$FFFF88_H$	ICR09	$0000B9_H$ *1	
Real time watch timer Watch timer (sub clock)	×	#30	$1E_H$	$FFFF84_H$			
Free-run timer overflow/clear	×	#31	$1F_H$	$FFFF80_H$	ICR10	$0000BA_H$ *1	
A/D converter conversion completed	○	#32	20_H	$FFFF7C_H$			
Sound generator 0/1	×	#33	21_H	$FFFF78_H$	ICR11	$0000BB_H$ *1	
Time-base timer	×	#34	22_H	$FFFF74_H$			
UART2 received	◎	#35	23_H	$FFFF70_H$	ICR12	$0000BC_H$ *1	
UART2 transmitted	△	#36	24_H	$FFFF6C_H$			
UART1 received	◎	#37	25_H	$FFFF68_H$	ICR13	$0000BD_H$ *1	
UART1 transmitted	△	#38	26_H	$FFFF64_H$			
UART0 received	◎	#39	27_H	$FFFF60_H$	ICR14	$0000BE_H$ *1	
UART0 transmitted	△	#40	28_H	$FFFF5C_H$			
Flash memory status	×	#41	29_H	$FFFF58_H$	ICR15	$0000BF_H$ *1	
Delayed interrupt generator module	×	#42	$2A_H$	$FFFF54_H$			Low

Table 3.2-2 Interrupt Sources and Interrupt Vectors/Interrupt Control Registers

⊙: Available with EI^{2}OS stop function, ○: Available, ×: Not available

△: Available only if no interrupt sources that share ICR are used

*1:
- Peripheral functions that share ICR register have the same interrupt level.
- When sharing ICR registers while using the extended intelligent I/O service (EI^{2}OS) for peripheral functions, either a normal interrupt or an extended intelligent I/O service can be used.
- If, in case of peripheral functions that share an ICR register, one of the functions specifies an extended intelligent I/O service (EI^{2}OS), the other is not allowed to use an interrupt.

*2: Priority assigned if interrupts with the same level occur.

3.3 Interrupt Control Registers and Peripheral Functions

Interrupt control registers (ICR00 to ICR15) are located in the interrupt controller corresponding to all the peripheral functions that include interrupt functions. These registers control the interrupt and extended intelligent I/O service (EI^2OS).

■ List of Interrupt Control Registers

Table 3.3-1 shows a list of interrupt control registers and the corresponding peripheral functions.

Table 3.3-1 List of Interrupt Control Registers

Address	Register	Abbreviation	Corresponding peripheral function
$0000B0_H$	Interrupt control register 00	ICR00	CAN0/CAN2
$0000B1_H$	Interrupt control register 01	ICR01	CAN1/CAN3
$0000B2_H$	Interrupt control register 02	ICR02	Input capture 0, DTP/external interrupt 0/1
$0000B3_H$	Interrupt control register 03	ICR03	Reload timer 0/2
$0000B4_H$	Interrupt control register 04	ICR04	Input capture 1, DTP/external interrupt 2/3
$0000B5_H$	Interrupt control register 05	ICR05	Input capture 2, reload timer 3
$0000B6_H$	Interrupt control register 06	ICR06	Input capture 3/4/5/6/7, DTP/external interrupt 4/5, UART3
$0000B7_H$	Interrupt control register 07	ICR07	PPG timer 0, DTP/external interrupt 6/7, UART3
$0000B8_H$	Interrupt control register 08	ICR08	PPG timer 1, reload timer 1
$0000B9_H$	Interrupt control register 09	ICR09	PPG timer 2/3/4/5, real time watch timer, watch timer
$0000BA_H$	Interrupt control register 10	ICR10	Free-run timer, A/D converter
$0000BB_H$	Interrupt control register 11	ICR11	Time-base timer, sound generator 0/1
$0000BC_H$	Interrupt control register 12	ICR12	UART2
$0000BD_H$	Interrupt control register 13	ICR13	UART1
$0000BE_H$	Interrupt control register 14	ICR14	UART0
$0000BF_H$	Interrupt control register 15	ICR15	Flash memory, delayed interrupt generator module

■ Functions of Interrupt Control Registers

The interrupt control registers (ICR) have the 4 functions listed below.

- Setting the interrupt level for the corresponding peripheral function
- Selecting whether the interrupt for the corresponding peripheral function is set to either normal interrupt or extended intelligent I/O service (EI^2OS)

- Selecting the channel of extended intelligent I/O service (EI^2OS)
- Indicating the status of extended intelligent I/O service (EI^2OS)

Some functions of interrupt control registers (ICR) differ depending on the time of either a read or a write operation, as shown in Figure 3.3-1 and Figure 3.3-2.

Note:

Do not access the interrupt control register (ICR) with read-modify-write (RMW) instructions as it might cause a malfunction.

3.3.1 Interrupt Control Registers (ICR00 to ICR15)

The interrupt control registers correspond to all of the peripheral functions that use interrupt functions and control processing at interrupt request generation. Some functions of these registers differ depending on the time of either a read or a write operation.

■ Interrupt Control Registers (ICR00 to ICR15)

Figure 3.3-1 Interrupt Control Registers (ICR00 to ICR15) during Write Operations

Write Operation

Address: 0000B0H to 0000BFH

MSB							LSB	Initial value
ICS3	ICS2	ICS1	ICS0	ISE	IL2	IL1	IL0	00000111B

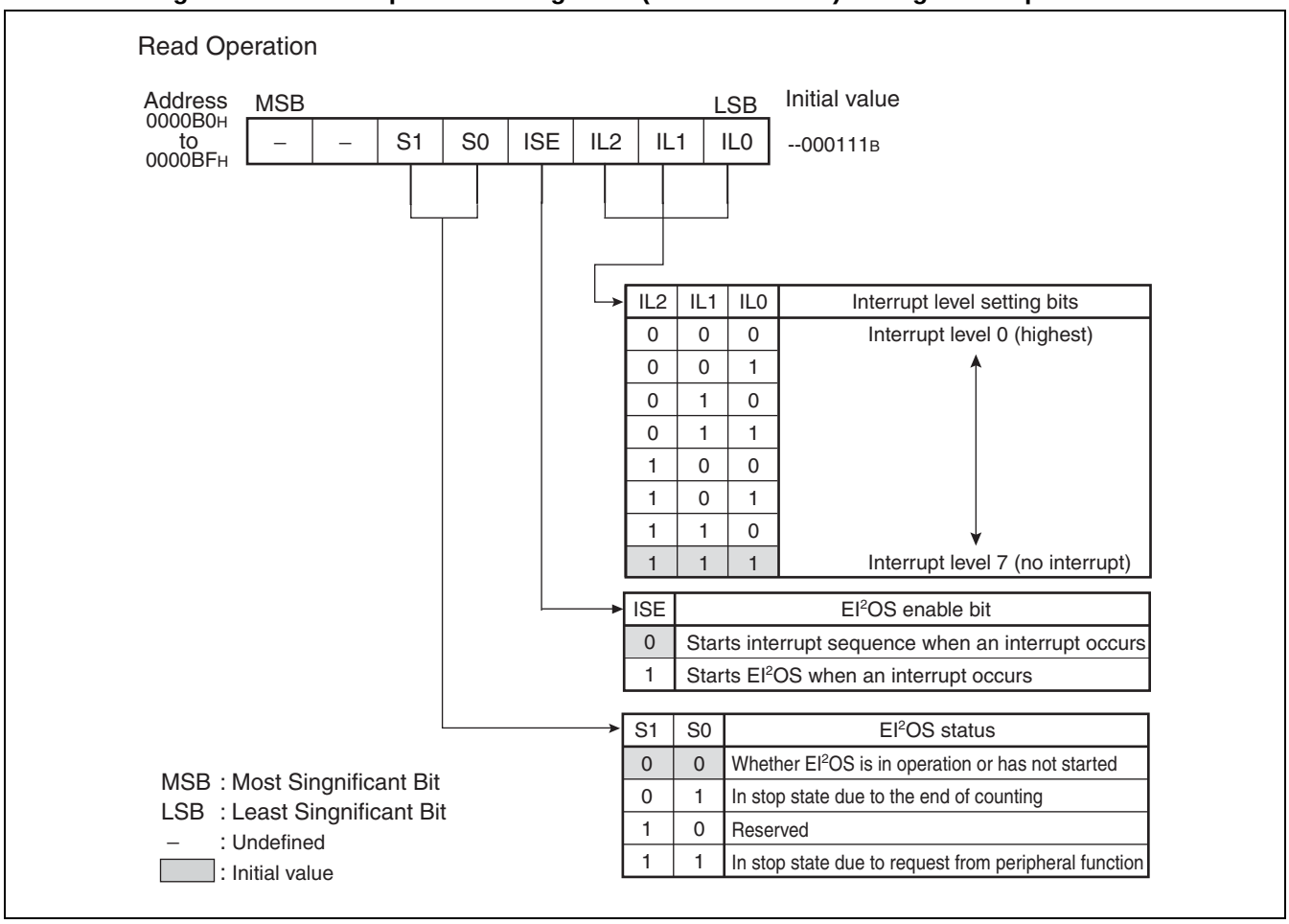
IL2	IL1	IL0	Interrupt level setting bits
0	0	0	Interrupt level 0 (highest)
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	Interrupt level 7 (no interrupt)

ISE	EI²OS enable bit
0	Starts interrupt sequence when an interrupt occurs
1	Starts EI²OS when an interrupt occurs

ICS3	ICS2	ICS1	ICS0	EI²OS channel selection bits	
				Channel	Descriptor address
0	0	0	0	0	000100H
0	0	0	1	1	000108H
0	0	1	0	2	000110H
0	0	1	1	3	000118H
0	1	0	0	4	000120H
0	1	0	1	5	000128H
0	1	1	0	6	000130H
0	1	1	1	7	000138H
1	0	0	0	8	000140H
1	0	0	1	9	000148H
1	0	1	0	10	000150H
1	0	1	1	11	000158H
1	1	0	0	12	000160H
1	1	0	1	13	000168H
1	1	1	0	14	000170H
1	1	1	1	15	000178H

MSB : Most Singnificant Bit
LSB : Least Singnificant Bit
: Initial value

Figure 3.3-2 Interrupt Control Registers (ICR00 to ICR15) during Read Operations



3.3.2 Functions of Interrupt Control Registers

The interrupt control registers (ICR00 to ICR15) consist of bits with the following 4 functions:

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

■ Configuration of Interrupt Control Register (ICR)

Figure 3.3-3 shows the bit configuration of the interrupt control register (ICR).

Figure 3.3-3 Configuration of Interrupt Control Register (ICR)

Interrupt control register (ICR) during write operations

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
$0000B0_H$ to $0000BF_H$	ICS3	ICS2	ICS1	ICS0	ISE	IL2	IL1	IL0	00000111_B
	W	W	W	W	R/W	R/W	R/W	R/W	

Interrupt control register (ICR) during read operations

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
$0000B0_H$ to $0000BF_H$	-	-	S1	S0	ISE	IL2	IL1	IL0	$--000111_B$
	-	-	R	R	R/W	R/W	R/W	R/W	

R/W: Readable/Writable
R: Read only
W: Write only
-: Undefined

Bits ICS3 to ICS0 are valid only if the extended intelligent I/O service (EI^{2}OS) has been started. To start EI^{2}OS, set the ISE bit to "1", otherwise, set it to "0". If EI^{2}OS is not started, ICS3 to ICS0 do not need to be set.

ICS1 and ICS0 are valid only in write operations, and S1 and S0 are valid only in read operations.

■ Functions of Interrupt Control Registers

● Interrupt level setting bits (IL2 to IL0)

These bits specify the interrupt level of the corresponding peripheral function. The interrupt level is initialized to level 7 (no interrupt) at reset. The interrupt level setting bits correspond to each interrupt level as shown in Table 3.3-2.

Table 3.3-2 Relationship between Interrupt Level Setting Bits and Interrupt Levels

IL2	IL1	IL0	Interrupt level
0	0	0	0 (Highest interrupt)
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	6 (Lowest interrupt)
1	1	1	7 (No interrupt)

● Extended intelligent I/O service (EI^{2}OS) enable bit (ISE)

If, when an interrupt request is generated, this bit is set to "1", EI^{2}OS starts. If it is set to "0", an interrupt sequence starts. If the EI^{2}OS end condition is satisfied (i.e., the S1 and S0 bits are other than 00_B), the ISE bit is cleared. If the corresponding peripheral function does not use any EI^{2}OS functions, the ISE bit must be set to "0" by software. The ISE bit is initialized to "0" at reset.

● Extended intelligent I/O service (EI^{2}OS) channel selection bits (ICS3 to ICS0)

The EI^{2}OS channel is specified by these write-only bits. The value set by these bits determines the EI^{2}OS descriptor address. The ICS bits are initialized to 0000_B at reset. Table 3.3-3 shows the relationship between the EI^{2}OS channel selection bits and the descriptor address.

Table 3.3-3 Relationship between EI^{2}OS Channel Selection Bits and Descriptor Addresses

ICS3	ICS2	ICS1	ICS0	Channel selected	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

- Extended intelligent I/O service (EI^{2}OS) status bits (S1, S0)

These bits are read-only bits. Their value is checked at the end of EI^{2}OS operation to determine the operational status (in operation or ended). They are initialized to 00_B at reset. Table 3.3-4 shows the relationship between the S0/S1 bits and the EI^{2}OS status.

Table 3.3-4 Relationship between EI^{2}OS Status Bits and EI^{2}OS Status

S1	S0	EI^{2}OS status
0	0	EI^{2}OS is in operation or not active
0	1	In stop state because of the end of counting
1	0	Reserved
1	1	In stop state because of a request from a peripheral function

3.4 Hardware Interrupt

Hardware interrupts are used to temporarily stop the execution of program that is being executed by the CPU in response to an interrupt request signal from a peripheral function and then transfer control to a user-defined interrupt handling program. The extended intelligent I/O service (EI^{2}OS) or an external interrupt may also be executed as a kind of hardware interrupt.

■ Functions of Hardware Interrupt

● Functions of Hardware Interrupt

During the processing for hardware interrupts, the interrupt level of an interrupt request signal output by the peripheral function is compared with the value of the interrupt level mask register (ILM) in the CPU processor status register (PS). The I flag in the processor status register (PS) is then referenced to determine whether the interrupt is acceptable.

If the hardware interrupt is accepted, the register contents in the CPU are automatically saved to the system stack and the interrupt level currently requested is stored in the interrupt level mask register (ILM). After that, the control branches to the corresponding interrupt vector.

● Multiple Interrupts

Hardware interrupts can be generated in multiple.

● Extended intelligent I/O service (EI^{2}OS)

EI^{2}OS is a function for automatic transfer between memory and I/O, generating a hardware interrupt when the number of transfers reaches a pre-defined count. EI^{2}OS cannot start multiple times concurrently: While one EI^{2}OS processing is running, all other interrupt requests and EI^{2}OS requests are retained.

● External Interrupt

External interrupts (including wake-up interrupts) are accepted as hardware interrupts transferred via a peripheral function (interrupt request detection circuit).

● Interrupt Vector

The interrupt vector table to be referenced for interrupt handling is allocated at the memory addresses $FFFC00_H$ to $FFFFFF_H$, which are shared with software interrupts.

For the allocation of interrupt numbers and interrupt vectors, see Section "3.2 Interrupt Sources and Interrupt Vectors".

■ Structure of Hardware Interrupt

The hardware interrupt unit exists in the four separate sections, as shown in Table 3.4-1. To use hardware interrupts, these four sections must be set using the program.

Table 3.4-1 Hardware Interrupt Unit

	Hardware interrupt unit	Function
Peripheral functions	Interrupt enable bit, interrupt request bit	Controls interrupt requests from peripheral functions
Interrupt controller	Interrupt control register (ICR)	Sets interrupt level and controls EI^2OS
CPU	Interrupt enable flag (I)	Identifies whether interrupts are enabled
	Interrupt level mask register (ILM)	Compares request interrupt level and current interrupt level
	Microcode	Executes interrupt handling routine
Addresses $FFFC00_H$ to $FFFFFF_H$ in memory	Interrupt vector table	Stores branch addresses for interrupt handling

■ Suppressing Hardware Interrupt

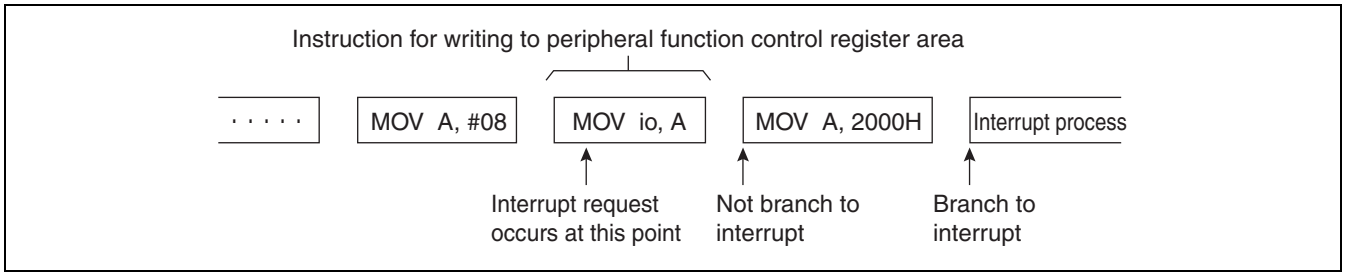
Acceptance of hardware interrupt requests is suppressed under the following conditions.

● Suppressing hardware interrupts when writing to peripheral function control register area

When writing to the peripheral function control register area, hardware interrupt requests are not accepted. This avoids incorrect interrupt-related operations by the CPU when rewriting the interrupt control registers with each peripheral function. The peripheral function control register area is not the I/O addressing area from 000000_H to $0000FF_H$, but the area allocated for the control registers among the peripheral function control registers and data register.

Figure 3.4-1 shows the hardware interrupt operations when writing to the peripheral function control register area.

Figure 3.4-1 Hardware Interrupt Requests when Writing to the Peripheral Function Control Register Area



- Suppressing hardware interrupts by interrupt suppress instructions

The 10 types of hardware interrupt suppress instructions shown in Table 3.4-2 will suppress detection of hardware interrupt requests and ignore any such interrupt request. Even if a valid hardware interrupt request is issued when these instructions are being executed, interrupt handling is not executed until another type of instruction is executed.

Table 3.4-2 Hardware Interrupt Suppress Instructions

	Prefix code	Interrupt/hold suppress instruction (Instruction that delays the effect of the prefix code)	
Instruction that accepts neither interrupt nor hold requests	PCB DTB ADB SPB CMR NCC	MOV OR AND POPW	ILM, #imm8 CCR, #imm8 CCR, #imm8 PS

- Hardware interrupt suppression during the execution of software interrupts

When software interrupt processing starts, other interrupt requests are not acceptable by the reason of clearing the I flag to "0".

3.4.1 Hardware Interrupt Operation

This section describes the operation from generation of a hardware interrupt request to the completion of interrupt handling.

■ Starting Hardware Interrupt

● Operation of peripheral function (generating an interrupt request)

A peripheral function that uses hardware interrupt requests has an "interrupt request flag" to indicate whether an interrupt request is present and an "interrupt enable flag" to enable or disable interrupt requests to the CPU. The interrupt request flag is set when a peripheral function-specific event is generated, and an interrupt request is issued to the interrupt controller if the interrupt enable flag is set to "enable".

● Operation of interrupt controller (control of interrupt requests)

The interrupt controller compares interrupt levels (IL) in interrupt requests that are received at the same time, selects the request with the highest priority level (the lowest IL value) and reports it to the CPU. If multiple requests have the same priority level, the request with the smallest interrupt number has the priority.

● CPU operation (acceptance of interrupt requests and interrupt handling)

The CPU compares the interrupt level (ICR: IL2 to IL0) received with the interrupt level mask register (ILM). If IL < ILM and interrupts are enabled (I bit in the PS register is set to "1"), processing of the interrupt handling microcode starts after the instruction currently being executed is completed. If the ISE bit of the interrupt control register (ICR) is referenced and found to be "0" during processing of the start of the interrupt handling microcode, the execution of interrupt handling will continue (If ISE is set to "1", EI^2OS will start).

During interrupt handling, the contents of dedicated registers (12 bytes of A, DPR, ADB, DTB, PCB, PC and PS) are first saved to the system stack (into the system stack area indicated by SSB and SSP). Then, the interrupt vector is loaded into the program counter (PCB, PC), ILM is updated and the stuffing (S) flag is set (i.e., the CCR's S flag is set to "1" and the system stack becomes enabled).

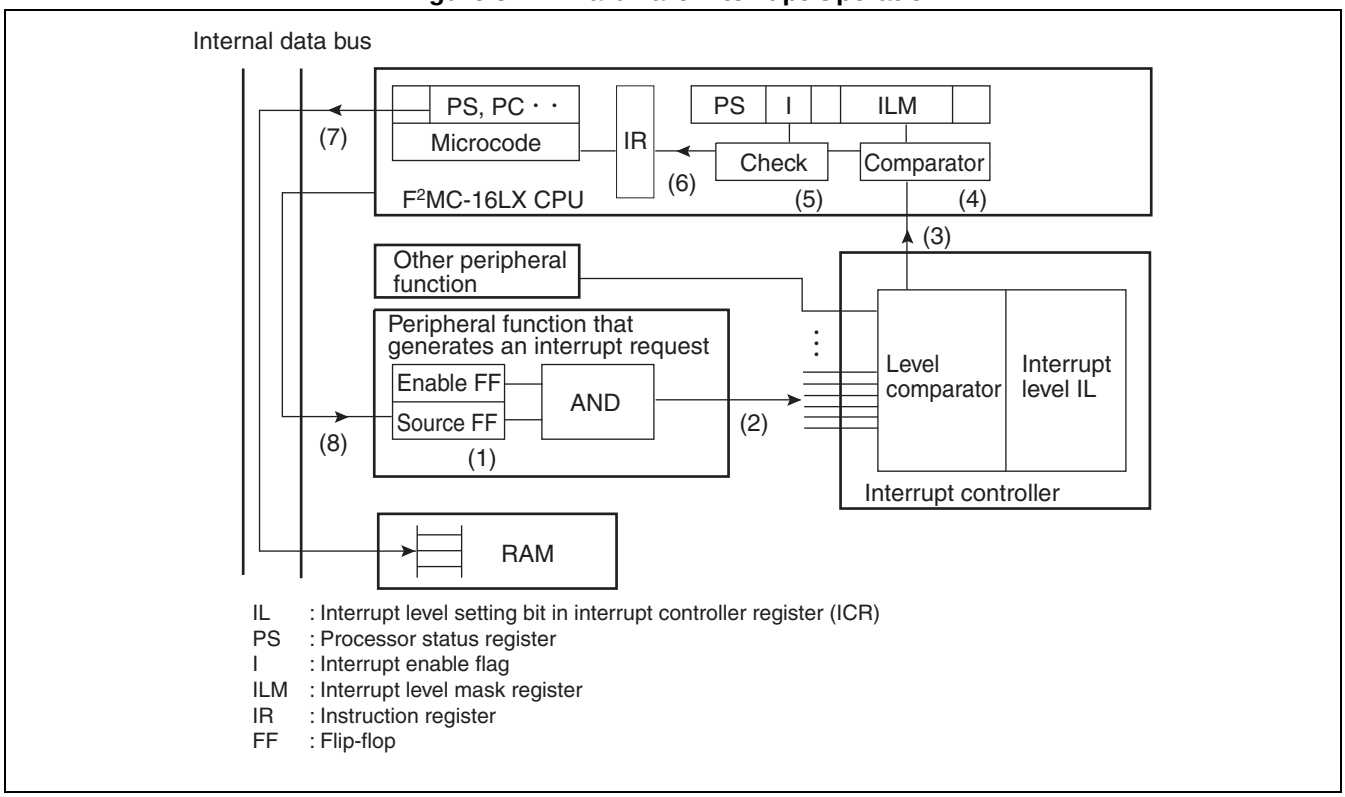
■ Return from Hardware Interrupt

If, in the interrupt handling program, the interrupt request flag of the peripheral function, which is an interrupt source, is cleared and the RETI instruction is executed, the 12 bytes of data saved to the system stack are returned to the dedicated registers to restart the processing that was being executed before the interrupt branch. By clearing the interrupt request flag, the interrupt request that was output to the interrupt controller by a peripheral function is automatically canceled.

■ Hardware Interrupt Operation

Figure 3.4-2 shows the operation from the generation of a hardware interrupt to the completion of interrupt handling.

Figure 3.4-2 Hardware Interrupt Operation



(1) An interrupt source is generated by the peripheral function.

(2) If a reference to the interrupt enable bit by the peripheral function indicates that interrupts are enabled, the interrupt request is issued from the peripheral to the interrupt controller.

(3) The interrupt controller that receives the interrupt request checks the priority of interrupt requests generated at the same time, then transfers the interrupt level (IL) corresponding to the interrupt request to the CPU.

(4) The CPU compares the interrupt level (IL) requested from the interrupt controller with the interrupt level mask register (ILM).

(5) If the comparison shows that the priority of the interrupt is higher than the current interrupt handling level, the I flag of the condition code register (CCR) is checked.

(6) If the check in (5) indicates that the I flag is set to interrupt enabled (I bit set to "1"), ILM is set to the requested level (IL) after the instruction currently being executed is completed.

(7) The register content is saved and processing branches to the interrupt handling routine.

(8) The user's interrupt handling routine clears the interrupt source generated in (1) for executing the RETI instruction. This completes interrupt handling.

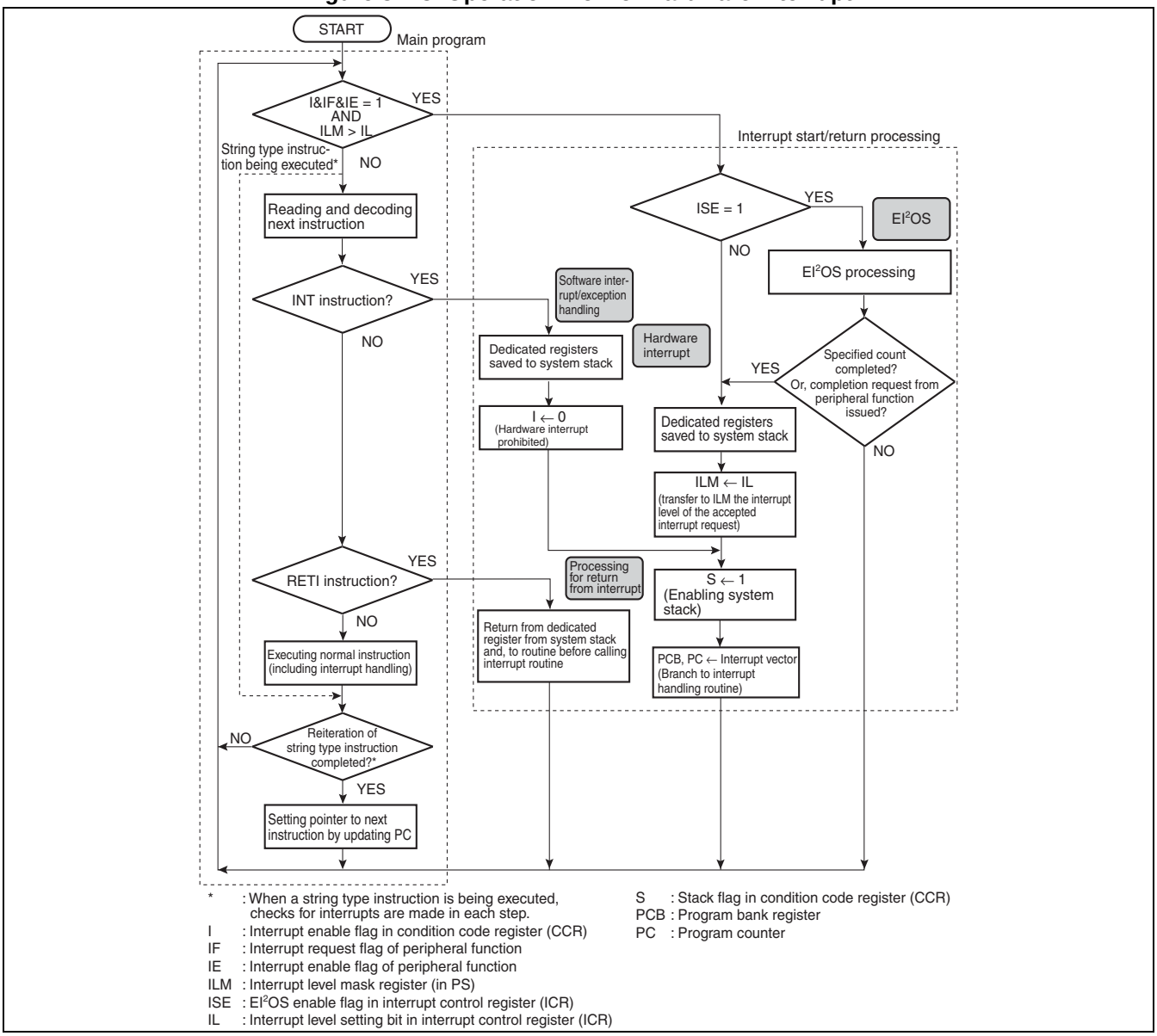
3.4.2 Operation Flow of Hardware Interrupt

If an interrupt request is generated from a peripheral function, the interrupt controller transfers the respective interrupt level to the CPU. If the CPU state allows the acceptance of interrupts, the instruction currently being executed is temporarily suspended to execute the interrupt handling routine or start the extended intelligent I/O service (EI^{2}OS). If a software interrupt is generated by the INT instruction, the interrupt handling routine is executed irrespective of the CPU state. Moreover, if a software interrupt is generated by an INT instruction, hardware interrupts are disabled.

■ Operation Flow of Hardware Interrupt

Figure 3.4-3 shows the operation flow of hardware interrupts.

Figure 3.4-3 Operation Flow of Hardware Interrupt



* : When a string type instruction is being executed, checks for interrupts are made in each step.
I : Interrupt enable flag in condition code register (CCR)
IF : Interrupt request flag of peripheral function
IE : Interrupt enable flag of peripheral function
ILM : Interrupt level mask register (in PS)
ISE : EI²OS enable flag in interrupt control register (ICR)
IL : Interrupt level setting bit in interrupt control register (ICR)
S : Stack flag in condition code register (CCR)
PCB : Program bank register
PC : Program counter

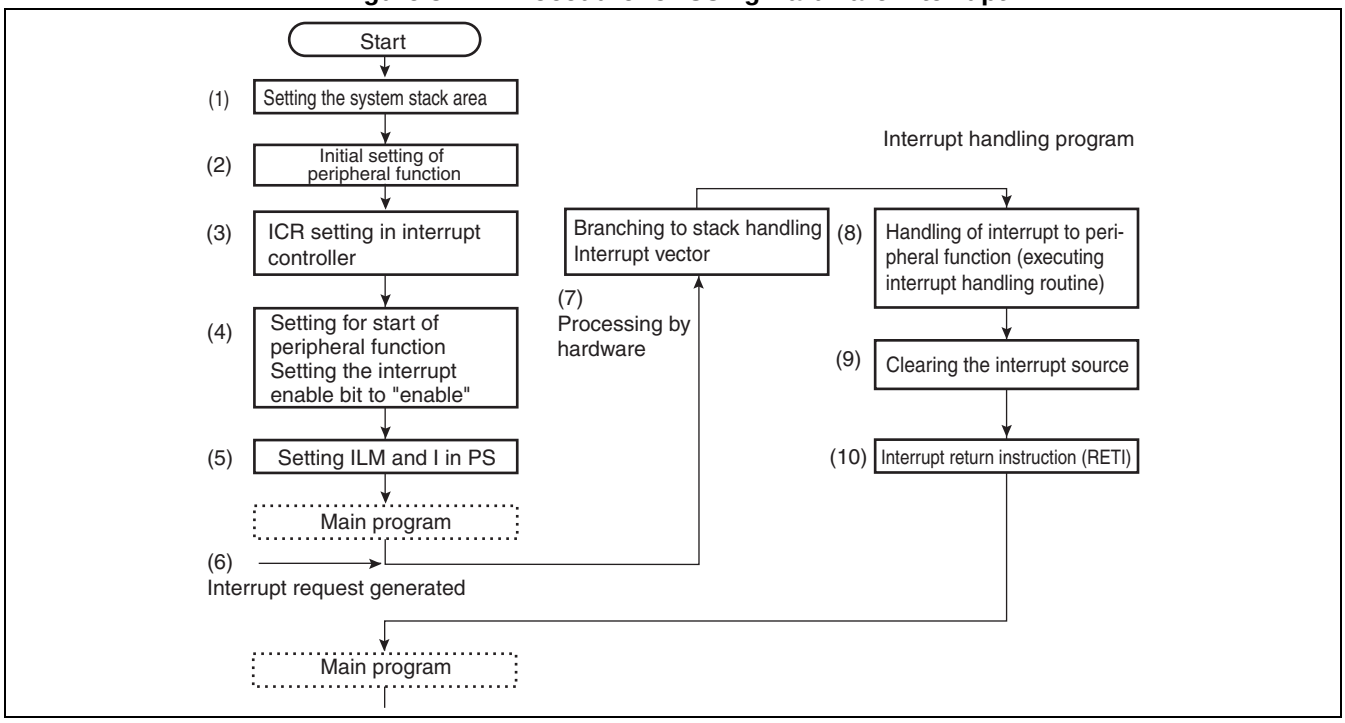
3.4.3 Procedure for Using Hardware Interrupt

To use an hardware interrupt, the system stack area, peripheral function, and interrupt control register (ICR) must be specified in advance.

■ Procedure for Using Hardware Interrupt

An example procedure for using hardware interrupts is shown in Figure 3.4-4.

Figure 3.4-4 Procedure for Using Hardware Interrupt



(1) Specify the system stack area.

(2) Initialize the peripheral function for generating interrupt requests.

(3) Specify the interrupt control register (ICR) in the interrupt controller.

(4) Set the peripheral function to operation start state and the interrupt enable bit to "enable".

(5) Set the interrupt level mask register (ILM) and interrupt enable flag (I) to allow interrupts to be accepted.

(6) A hardware interrupt request is generated when the peripheral function generates an interrupt.

(7) The hardware interrupt handling saves register contents for branching to the interrupt handling program.

(8) Process the interrupt generation for the peripheral function using the interrupt handling program.

(9) Clear the interrupt request from the peripheral function.

(10) Execute the interrupt return instruction to return to the program that was executed before branching.

3.4.4 Multiple Interrupts

For hardware interrupts, multiple interrupts are enabled by setting different interrupt levels to a interrupt level setting bit (IL0, IL1, IL2) in the interrupt control register (ICR) for multiple interrupt requests from the peripheral function. However, multi-startup is not allowed for the extended intelligent I/O service.

■ Multiple Interrupts Operation

If, when an interrupt handling routine is being executed, an interrupt request with a higher priority level is issued, the interrupt request with the higher priority level is accepted by interrupting the current interrupt handling. After processing the interrupt with the higher level has been completed, the processing of the original interrupt is restarted. The interrupt level can be set to a value from 0 to 7. When set to level 7, the CPU will not accept the interrupt request.

If, when interrupt handling is being executed, another interrupt is issued which has the same priority or lower as the current one, the new interrupt request will be retained until processing of the current interrupt is completed unless the I flag or ILM is changed. By setting the I flag in the condition code register (CCR) to "interrupt disabled" (I in CCR set to "0") or the interrupt level mask register (ILM) to "interrupt disabled" (ILM set to 000_B) during the interrupt handling routine, starting multiple interrupts can be temporarily prohibited.

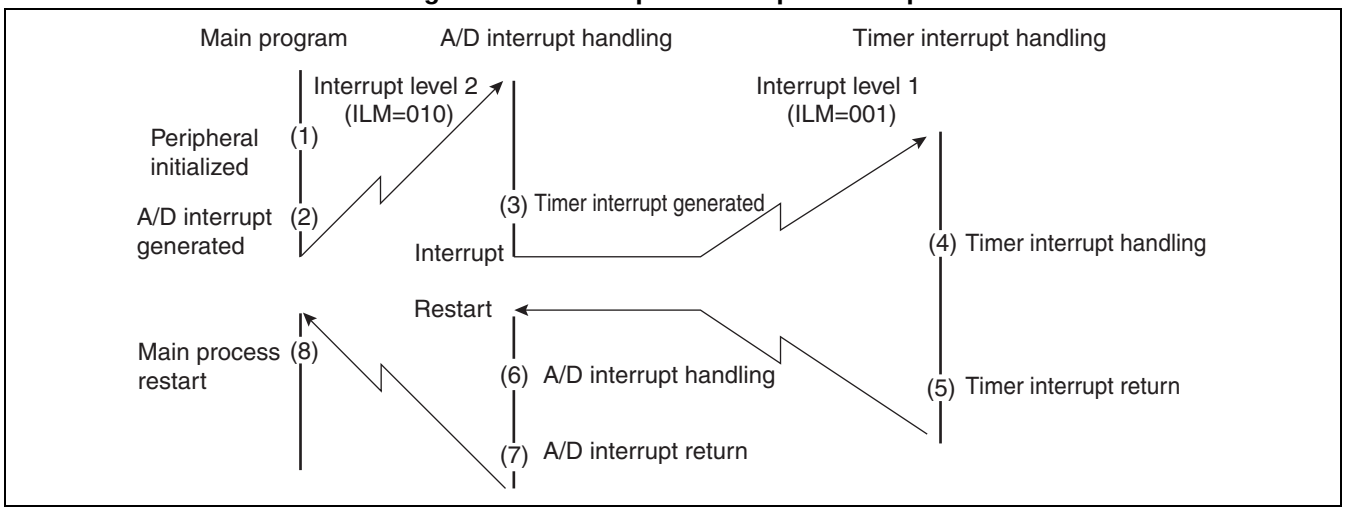
Note:

Starting multiple instances of the extended intelligent I/O service (EI^2OS) is prohibited. If an extended intelligent I/O service (EI^2OS) is being processed, all other interrupt requests and extended intelligent I/O service requests are retained.

■ Example of Multiple Interrupts

In the following example for processing multiple interrupts, the timer interrupt has priority over A/D converter interrupts: the interrupt level of the A/D converter is set to 2, and the timer interrupt level is set to 1. When a timer interrupt is generated while an A/D converter interrupt is being processed with this setting, the processing shown in Figure 3.4-5 is performed.

Figure 3.4-5 Example of Multiple Interrupts



● A/D interrupt generation

When A/D converter interrupt handling starts, the interrupt level mask register (ILM) is automatically set to the same interrupt level (IL2 to IL0 in ICR) as that for the A/D converter (in this example, (2). If a new interrupt request of level 1 or 0 is generated in this case, the new interrupt is processed with priority.

● End of interrupt handling

If, after an interrupt handling is completed, a return instruction (RETI) is executed, the contents of dedicated registers (A, DPR, ADB, DTB, PCB, PC, PS) that were saved to the stack are returned, and the interrupt level mask register (ILM) is set to the value immediately before the interrupt.

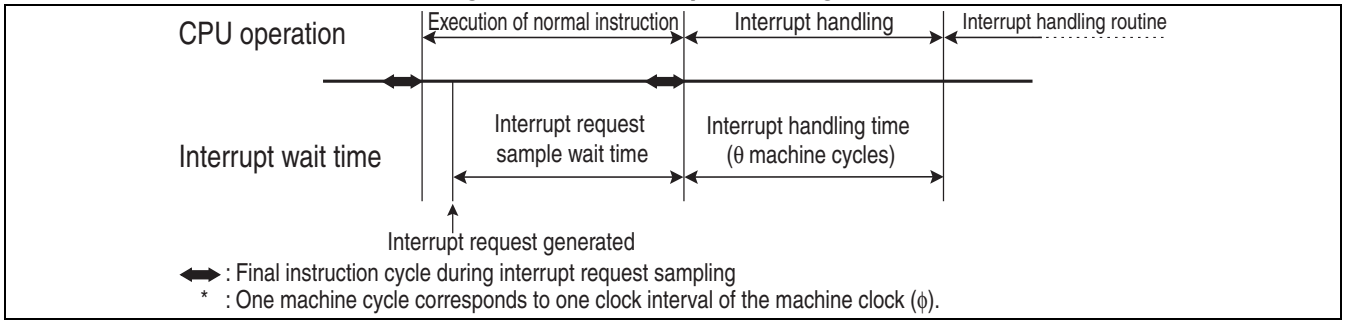
3.4.5 Hardware Interrupt Processing Time

The duration between the generation of a hardware interrupt request and the execution of the interrupt handling routine must include the time until the instruction currently being executed is completed plus the interrupt handling time.

■ Hardware Interrupt Processing Time

The interrupt request sample wait time and interrupt handling time (time required for preparing for interrupt handling) are required while an interrupt request is generated and accepted, then the interrupt handling routine is executed. Figure 3.4-6 shows the interrupt handling time.

Figure 3.4-6 Interrupt Handling Time



● Interrupt request sample wait time

It means the time from when an interrupt request is generated until the instruction currently being executed is completed. Whether an interrupt request is generated is determined by sampling the interrupt requests in the final cycle of each instruction. Therefore, the CPU cannot detect an interrupt request while an instruction is still being executed, which results in wait time.

The interrupt request sample wait time reaches the maximum value if an interrupt request occurs immediately after the start of a sequence of the type "POPW RW0, ... RW7 instruction" (45 machine cycles), because these sequences have the longest execution cycle.

● Interrupt handling time (θ machine cycles)

After an interrupt request is accepted, the CPU performs such operations as saving the contents of the dedicated registers to the system stack, acquiring the interrupt vector, and so on. This requires an interrupt handling time of θ machine cycles. The interrupt handling time can be obtained from the following formulas:

At the start of interrupt processing: $\theta = 24 + 6 \times Z$ machine cycles

At the return from an interrupt: $\theta = 11 + 6 \times Z$ machine cycles (RETI instruction)

The interrupt handling time depends on the address indicated by the stack pointer. Table 3.4-3 shows the required correction value (Z) for the interrupt handling time.

One machine cycle corresponds to one clock cycle of the machine clock (ϕ).

Table 3.4-3 Correction Value (Z) for Interrupt Handling Time

Address indicated by the stack pointer	Correction value (Z)
External, 8-bit	+4
External, even-numbered address	+1
External, odd-numbered address	+4
Internal, even-numbered address	0

Table 3.4-3 Correction Value (Z) for Interrupt Handling Time

Address indicated by the stack pointer	Correction value (Z)
Internal, odd-numbered address	+2

3.5 Software Interrupt

Software interrupts have the function to transfer control from the program currently executed by the CPU to the interrupt handling program defined by the user when a software interrupt instruction (INT instruction) is executed. Hardware interrupts are disabled while a software interrupt is being executed.

■ Starting Software Interrupt

● Starting Software Interrupt

To issue a software interrupt, use the INT instruction. Software interrupt requests have no interrupt request flag or enable flag, but always generate an interrupt request if the INT instruction is executed.

● Disabling hardware interrupts

The INT instruction has no interrupt level, so the interrupt level mask register (ILM) is not updated. During the execution of the INT instruction, the I flag in the condition code register (CCR) is set to "0" to mask the hardware interrupt. To enable hardware interrupts during software interrupt handling, set the I flag to "1" in the software interrupt handling routine.

● Software Interrupt Operation

When the CPU receives and executes an INT instruction, it starts execution of the microcode for software interrupt handling. Use this microcode for saving the contents of the CPU internal registers to the system stack, masking hardware interrupts (I flag in CCR set to "0") and branching to the corresponding interrupt vector.

For the allocation of interrupt numbers and interrupt vectors, see Section "3.2 Interrupt Sources and Interrupt Vectors".

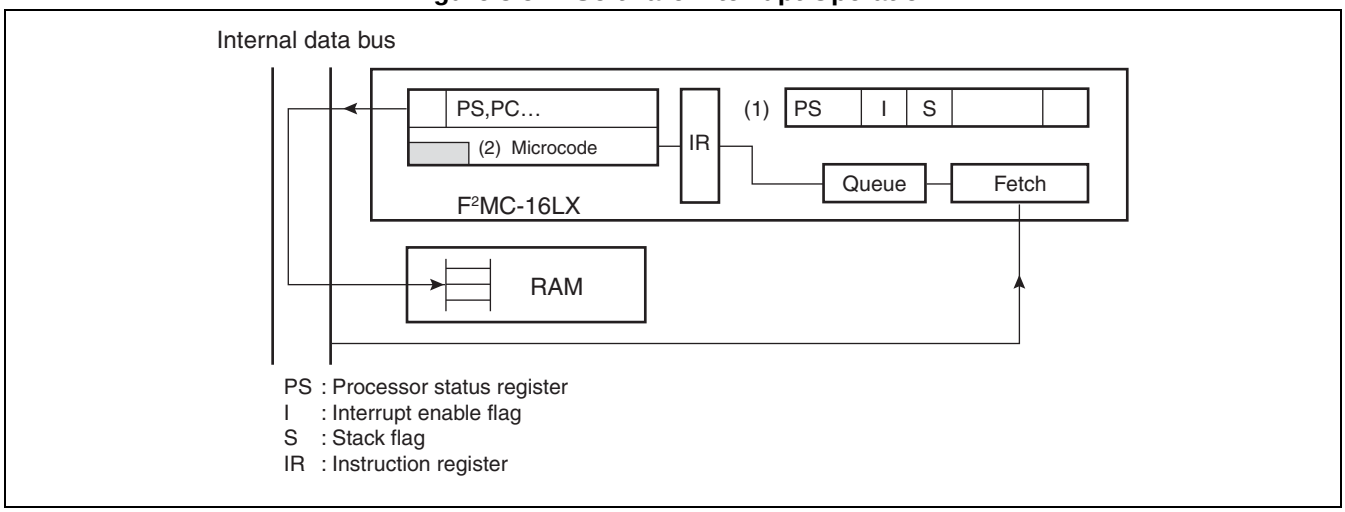
■ Return from Software Interrupt

When an interrupt return instruction (RETI instruction) is executed in the interrupt handling program, the 12-byte data saved to the system stack is restored to the dedicated registers. Moreover, the control returns to the processing that was being executed immediately before the interrupt branch.

■ Software Interrupt Operation

Figure 3.5-1 shows the operation performed from software interrupt generation to completion of interrupt handling.

Figure 3.5-1 Software Interrupt Operation



(1) Execute the software interrupt instruction.

(2) Based on the microcode for the software interrupt instruction, perform the necessary processing, such as saving the contents of the dedicated registers, and perform branch processing.

(3) Execute the RETI instruction in the user's interrupt handling routine to end interrupt handling.

■ Precaution for Software Interrupt

If the program bank register (PCB) is set to FF_H, the CALLV instruction's vector area overlaps the table for INT #vct8 instruction. When making the software, be sure that the addresses for the CALLV instruction and INT #vct8 instruction do not overlap.

3.6 Interrupt by Extended Intelligent I/O Service (EI^{2}OS)

The extended intelligent I/O service (EI^{2}OS) is a function to automatically transfer data between the peripheral function (I/O) and memory, and generates a hardware interrupt when the data transfer is completed.

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

The extended intelligent I/O service is processed as a type of hardware interrupt. It provides a function to automatically transfer data between the peripheral function (I/O) and memory. The data transfer that was previously performed by the interrupt-handling program is provided in the same way as via DMA (direct memory access). After completion, the end condition is set, and then an automatic branch to the interrupt handling routine occurs. The user only has to make a program for starting and ending EI^{2}OS. It is not necessary to provide a program for data transfer while the service is being processed.

● Advantages of using the extended intelligent I/O service (EI^{2}OS)

Compared with data transfer as provided by the interrupt handling routine, using EI^{2}OS provides the advantages listed below:

- No transfer program is required, which reduces overall program size.
- Transfer may be suspended based on the state of the peripheral function (I/O), which eliminates the need to transfer unnecessary data.
- Either incrementing or not updating the buffer address may be selected.
- Either incrementing or not updating the I/O register address may be selected.

● Completion interrupt for the extended intelligent I/O service (EI^{2}OS)

When data transfer by EI^{2}OS is completed, the end condition is stored in the S1 and S0 bits of the interrupt control register (ICR). After this, the system automatically branches to the interrupt handling routine.

The completion source for EI^{2}OS can be determined by using the interrupt handling program to check the EI^{2}OS status (S1 and S0 bits in ICR).

The interrupt number and interrupt vector for each peripheral are fixed. For detailed information, see Section "3.2 Interrupt Sources and Interrupt Vectors".

● Interrupt control register (ICR)

This register is located in the interrupt controller and used to start the EI^{2}OS, specify EI^{2}OS channels, and indicate the state when EI^{2}OS ends.

● Extended intelligent I/O service (EI^{2}OS) descriptor (ISD)

This is a data item of 8 byte, which is located in the RAM at the addresses 000100_H to $00017F_H$, provided for 16 channels. It retains the transfer mode, I/O address and transfer counts, and buffer addresses. The corresponding channel is specified by the interrupt control register (ICR).

Note:

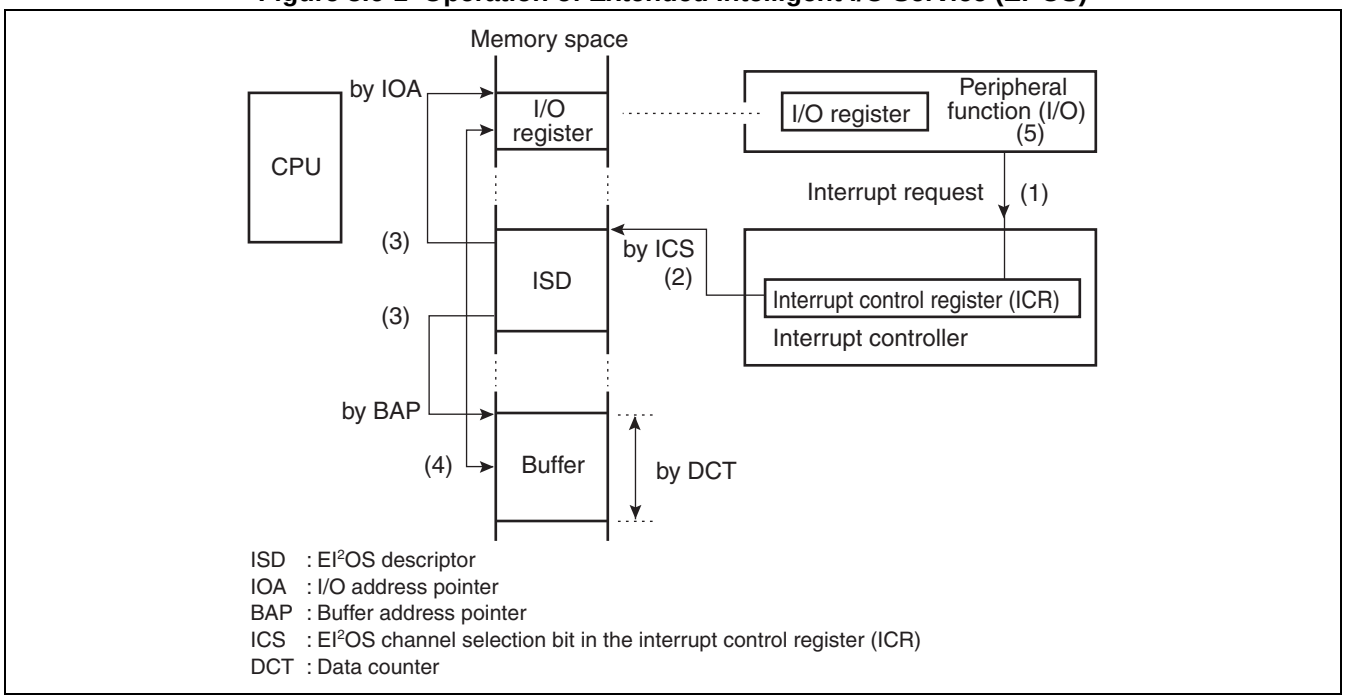
During the operation of the extended intelligent I/O service (EI^{2}OS), program execution of the CPU

is stopped.

■ Operation of Extended Intelligent I/O Service (EI^{2}OS)

Figure 3.6-1 shows the EI^{2}OS operation.

Figure 3.6-1 Operation of Extended Intelligent I/O Service (EI^{2}OS)



(1) I/O requests the transfer.

(2) Interrupt controller selects the descriptor.

(3) Transfer source/destination are read from the descriptor.

(4) Data transfer is performed between I/O and memory.

(5) Interrupt source is automatically cleared.

3.6.1 Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD)

The extended intelligent I/O service (EI^{2}OS) descriptor (ISD) is located in the internal RAM at the addresses 000100_H to $00017F_H$ and consists of 8 bytes × 16 channels.

■ Configuration of Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD)

The EI^{2}OS Descriptor (ISD) consists of 8 bytes × 16 channels. Each ISD has the structure shown in Figure 3.6-2. The relationship between the channel number and ISD address is indicated in Table 3.6-1.

Figure 3.6-2 Configuration of EI^{2}OS Descriptor (ISD)

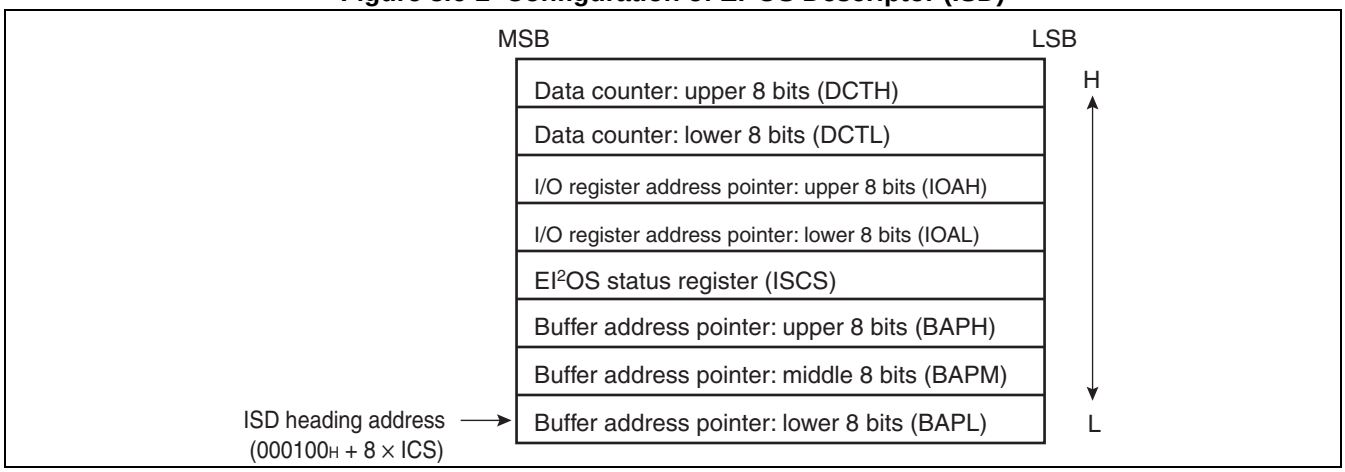


Table 3.6-1 Relationship between Channel Number and Descriptor Address

Channel No.	Descriptor address
0	000100_H
1	000108_H
2	000110_H
3	000118_H
4	000120_H
5	000128_H
6	000130_H
7	000138_H
8	000140_H
9	000148_H
10	000150_H
11	000158_H
12	000160_H
13	000168_H

Table 3.6-1 Relationship between Channel Number and Descriptor Address

Channel No.	Descriptor address
14	000170_H
15	000178_H

3.6.2 Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) Registers

The extended intelligent I/O service (EI^{2}OS) descriptor (ISD) consists of the registers listed below.

- **Data counter (DCT)**
- **I/O register address pointer (IOA)**
- **Extended intelligent I/O service (EI^{2}OS) status register (ISCS)**
- **Buffer address pointer (BAP)**

Note that the initial value of each register is undefined at reset.

■ Data Counter (DCT)

The data counter (DCT) is a register with a length of 16 bits for storing the transfer data count. Whenever an item of data is transferred, the counter is decremented by one. When this counter reaches zero, EI^{2}OS operation ends. The maximum transfer count that can be specified by the data counter (DCT) is 65,536 (64 Kbytes). Figure 3.6-3 shows the configuration of the DCT.

Figure 3.6-3 Configuration of Data Counter (DCT)

	DCTH						DCTL										
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
DCT	B15	B14	B13	B12	B11	B10	B09	B08	B07	B06	B05	B04	B03	B02	B01	B00	XXXXXXXXXXXXXXXX$_B$
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W:Readable/Writable
X: Undefined value

■ I/O Register Address Pointer (IOA)

The I/O register address pointer (IOA) is a register with a length of 16 bits to indicate the lower address (A15 to A00) of the I/O register for data transfer with the buffer. The upper address (A23 to A16) is set to all 0's, allowing I/O addresses from 000000_H to $00FFFF_H$ to be specified. Figure 3.6-4 shows the configuration of the IOA.

Figure 3.6-4 Configuration of I/O Register Address Pointer (IOA)

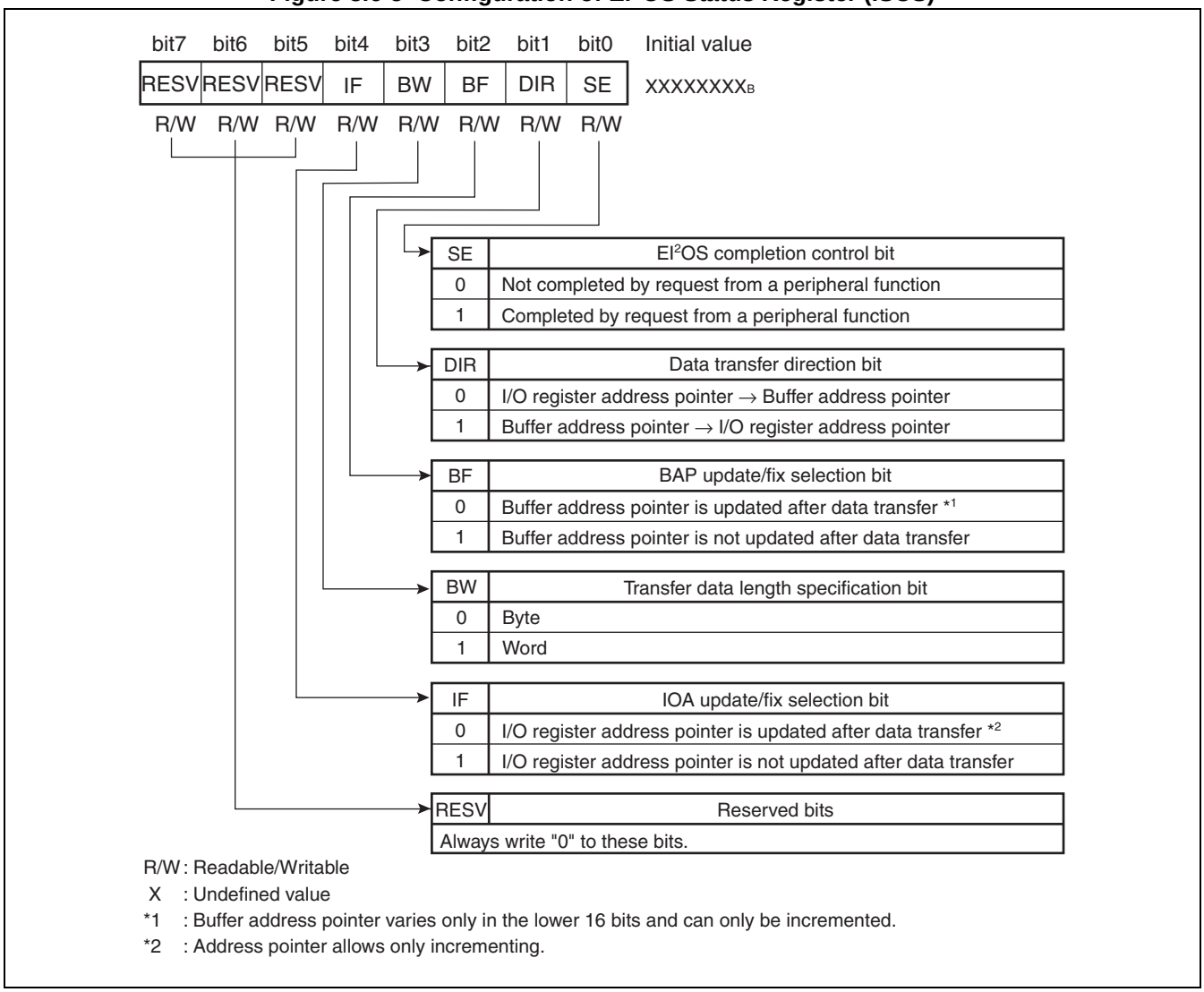
	IOAH						IOAL										
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
IOA	A15	A14	A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00	XXXXXXXXXXXXXXXX$_B$
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W:Readable/Writable
X: Undefined value

■ Extended Intelligent I/O Service (EI^{2}OS) Status Register (ISCS)

The extended intelligent I/O service (EI^{2}OS) status register (ISCS) consists of 8 bits indicating whether buffer address pointer and I/O register address pointer can be updated or are fixed. They also indicate the type of data transfer (in bytes or in words) as well as the direction of transfer. Figure 3.6-5 shows the configuration of the ISCS.

Figure 3.6-5 Configuration of EI^{2}OS Status Register (ISCS)



R/W : Readable/Writable
X : Undefined value
*1 : Buffer address pointer varies only in the lower 16 bits and can only be incremented.
*2 : Address pointer allows only incrementing.

■ Buffer Address Pointer (BAP)

The buffer address pointer (BAP) is a register consisting of 24 bits. It is used to store the address for the next EI^{2}OS data transfer. A BAP is separately assigned for the respective EI^{2}OS channel; each EI^{2}OS channel can be used to transfer data between any 16 Mbyte address and the I/O. If the BF bit of the EI^{2}OS status register (ISCS) (bit in the EI^{2}OS status register indicating whether BAP can be updated or is fixed) is set to "updated", the BAP varies only in the lower 16 bits (BAPM, BAPL), while the upper 8 bits (BAPH) will not be changed.

The buffer address pointer (BAP) can specify the area 000000_H to $FFFFFF_H$.

Figure 3.6-6 shows the configuration of the buffer address pointer (BAP).

Figure 3.6-6 Configuration of Buffer Address Pointer (BAP)

	bit23 to bit16	bit15 to bit8	bit7 to bit0	Initial value
BAP	BAPH	BAPM	BAPL	XXXXXX_H
	R/W	R/W	R/W	

R/W: Readable/Writable
X: Undefined value

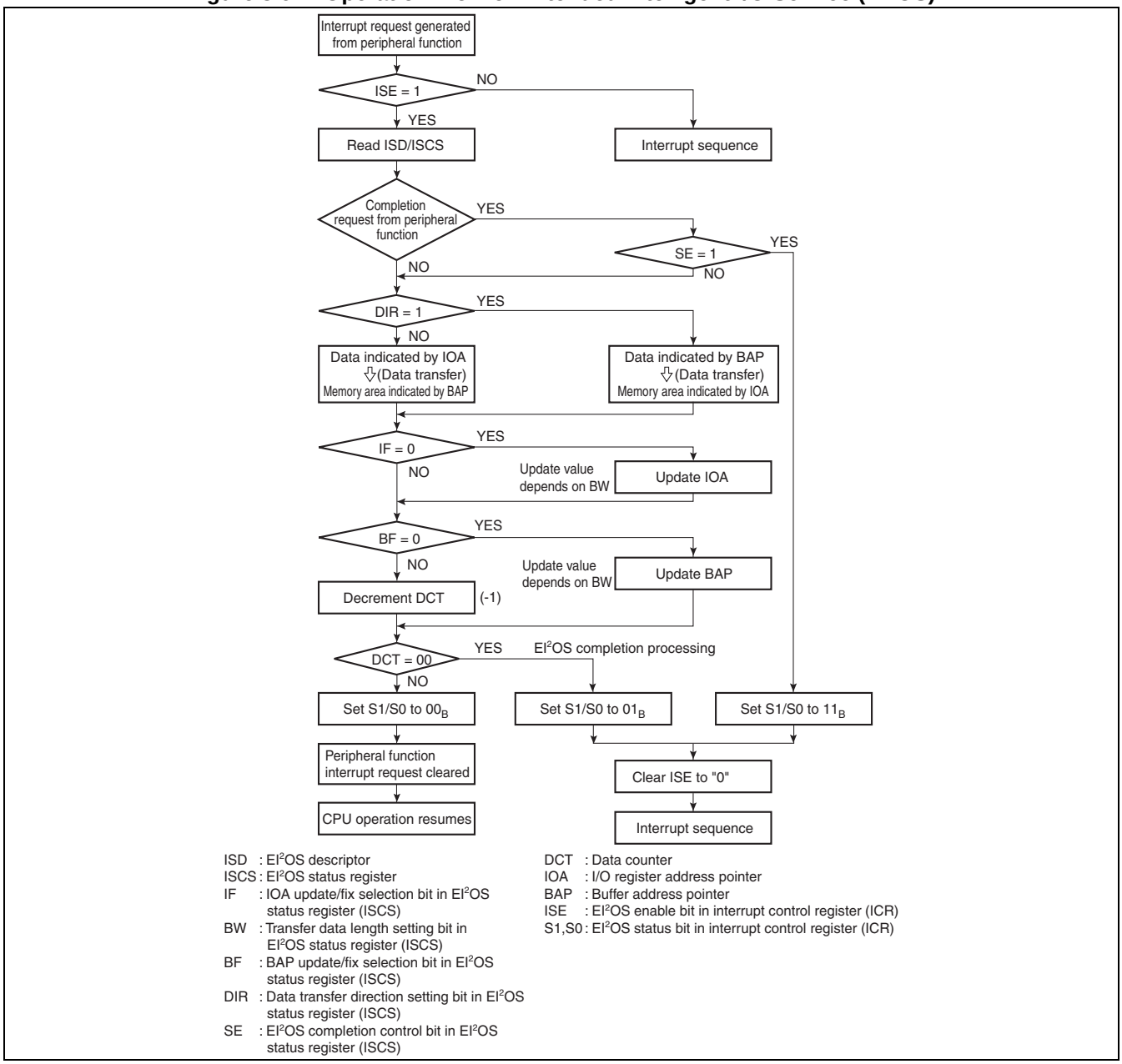
3.6.3 Operation of the Extended intelligent I/O Service (EI^{2}OS)

If a peripheral function issues an interrupt request and the corresponding interrupt control register (ICR) is set to start EI^{2}OS, the CPU allows EI^{2}OS data transfer. After the data transfer is completed for the count specified, the system will automatically perform the hardware interrupt.

■ Processing Procedure for Extended Intelligent I/O Service (EI^{2}OS)

Figure 3.6-7 shows the operational flow of EI^{2}OS processing using the CPU-internal microcode.

Figure 3.6-7 Operation Flow of Extended Intelligent I/O Service (EI^{2}OS)



ISD : EI^{2}OS descriptor
ISCS : EI^{2}OS status register
IF : IOA update/fix selection bit in EI^{2}OS status register (ISCS)
BW : Transfer data length setting bit in EI^{2}OS status register (ISCS)
BF : BAP update/fix selection bit in EI^{2}OS status register (ISCS)
DIR : Data transfer direction setting bit in EI^{2}OS status register (ISCS)
SE : EI^{2}OS completion control bit in EI^{2}OS status register (ISCS)

DCT : Data counter
IOA : I/O register address pointer
BAP : Buffer address pointer
ISE : EI^{2}OS enable bit in interrupt control register (ICR)
S1,S0 : EI^{2}OS status bit in interrupt control register (ICR)

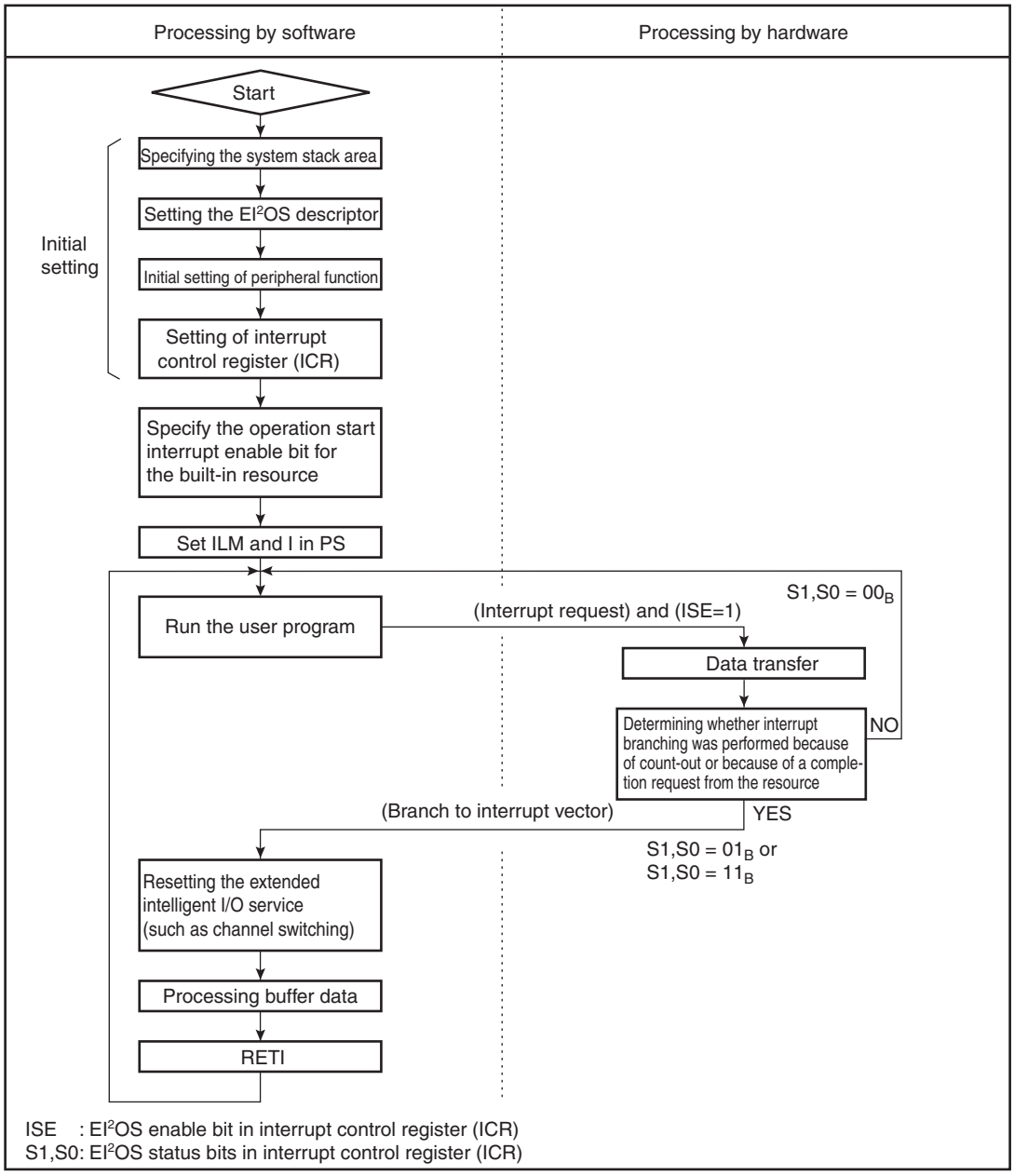
3.6.4 Extended Intelligent I/O Service (EI^{2}OS) Procedure

The extended intelligent I/O service (EI^{2}OS) needs to set the system stack area, extended intelligent I/O service (EI^{2}OS) descriptor, peripheral function, interrupt control register (ICR) and others.

■ Procedure for Using the Extended Intelligent I/O Service (EI^{2}OS)

Figure 3.6-8 shows the procedure for the extended intelligent I/O service (EI^{2}OS).

Figure 3.6-8 Procedure for Using the Extended Intelligent I/O Service (EI^{2}OS)



3.6.5 Extended Intelligent I/O Service (EI^{2}OS) Processing Time

The time required for processing the extended intelligent I/O service (EI^{2}OS) varies depending on the following factors:

- **Setting of EI^{2}OS status register (ISCS)**
- **Address (area) indicated by I/O register address pointer (IOA)**
- **Address (area) indicated by buffer address pointer (BAP)**
- **Bus width of external data bus for external access**
- **Data length of data to be transferred**

At the end of data transfer by EI^{2}OS, the interrupt handling time is added to the total time for processing, because a hardware interrupt occurs.

■ Processing Time for Extended Intelligent I/O Service (EI^{2}OS) (Time Consumed per Transfer)

● When data transfer continues

The EI^{2}OS process time required for data transfer is shown in Table 3.6-2 based on the setting of the EI^{2}OS status register (ISCS).

Table 3.6-2 Execution Time for Extended Intelligent I/O Service

Setting of EI^{2}OS completion control bit (SE)		Completion by completion request from peripherals		Ignoring completion requests from peripherals	
Setting of the IF bit indicating whether IOA can be updated or is fixed		Fixed	Updated	Fixed	Updated
Setting of the BF bit indicating whether BAP address can be updated or is fixed	Fixed	32	34	33	35
	Updated	34	36	35	37

Unit: Machine cycle (1 machine cycle corresponds to one clock interval of the machine clock (ϕ))

As shown in Table 3.6-3, settings must be corrected depending on the conditions under which EI^{2}OS is executed.

Table 3.6-3 Correction Values for Data Transfer during EI^{2}OS Execution

I/O register address pointer			Internal access		External access	
			B/Even	Odd	B/Even	8/Odd
Buffer address pointer	Internal access	B/Even	0	+2	+1	+4
		Odd	+2	+4	+3	+6
	External access	B/Even	+1	+3	+2	+5
		8/Odd	+4	+6	+5	+8

B: Byte data transfer
8: External bus width for 8-bit, word transfer
Even: Even address, word transfer
Odd: Odd address, word transfer

- When the data counter (DCT) stops counting (after the final data transfer is completed)

When data transfer by EI^{2}OS is completed, the interrupt handling time is added to the total processing time, since a hardware interrupt is generated. The EI^{2}OS processing time required when counting ends can be obtained from the following formula.

EI^{2}OS processing time when counting ends =
EI^{2}OS processing time by the end of data transfer + $\underline{(21+6\times Z)}$ machine cycles
↑
Interrupt handling time

One machine cycle corresponds to one clock cycle of the machine clock (ϕ).

The interrupt handling time varies depending on the address indicated by the stack pointer. Table 3.6-4 shows the applicable correction values (Z) for the interrupt handling time.

Table 3.6-4 Correction Values (Z) for Interrupt Handling Time

Address indicated by the stack pointer	Correction value (Z)
External, 8-bit	+4
External, even-numbered address	+1
External, odd-numbered address	+4
Internal, even-numbered address	0
Internal, odd-numbered address	+2

- When data transfer is ended by completion request from the peripheral function (I/O)

If EI^{2}OS data transfer is not fully completed (ICR: S1, S0 = 11_B) due to the completion request received from the peripheral function (I/O), data transfer is not executed and a hardware interrupt is generated.

The EI^{2}OS processing time can in this case be obtained from the formula shown below. The parameter Z in the formula represents a correction value for the interrupt handling time (see Table 3.6-4).

EI^{2}OS processing time until the transfer ended : $36 + 6 \times Z$ machine cycles

One machine cycle corresponds to one clock cycle of the machine clock (ϕ).

3.7 Exception Handling Interrupt by Execution of Undefined Instruction

F^{2}MC-16LX handles exception handling by undefined instructions. Exception handling is basically performed in the same way as interrupt handling, i.e., the normal flow of processing is interrupted for starting exception handling if an exception event is detected at the instruction boundary.
In general, exception handling is performed when an unexpected operation is executed. Therefore, it should only be used for debugging or by recovery software for emergency use.

■ Exception Handling Interrupt by Execution of Undefined Instruction

● Exception handling operation

F^{2}MC-16LX counts any code that is not defined in the instruction map as an undefined instruction. If undefined instructions are executed, the same processing as for a software interrupt instruction such as "INT # 10" is performed.

For except handling, the following processing is performed before branching to the interrupt routine.

- The contents of the A, DPR, ADB, DTB, PCB, PC, and PS registers are saved to the system stack.
- The I flag of the condition code register (CCR) is cleared to "0", masking hardware interrupts.
- The S flag of the condition code register (CCR) is set to "1", enabling the use of the system stack.

The program counter (PC) value saved to the stack represents the address under which the undefined instruction is stored. In other words, the address at which an instruction code of 2 bytes or more is stored which has been identified as "undefined". If the type of the exception cause needs to be identified in the exception handling routine, use this PC value.

● Return from exception handling

After returning from exception handling with the RETI instruction, exception handling will start again, since the PC points to an undefined instruction. Take appropriate action, for example, reset a software.

3.8 Stack Operations of Interrupt Handling

If an interrupt is accepted, the content of the dedicated register is automatically saved to the system stack before processing branches to interrupt handling. Return from the stack is also automatically performed after interrupt handling is completed.

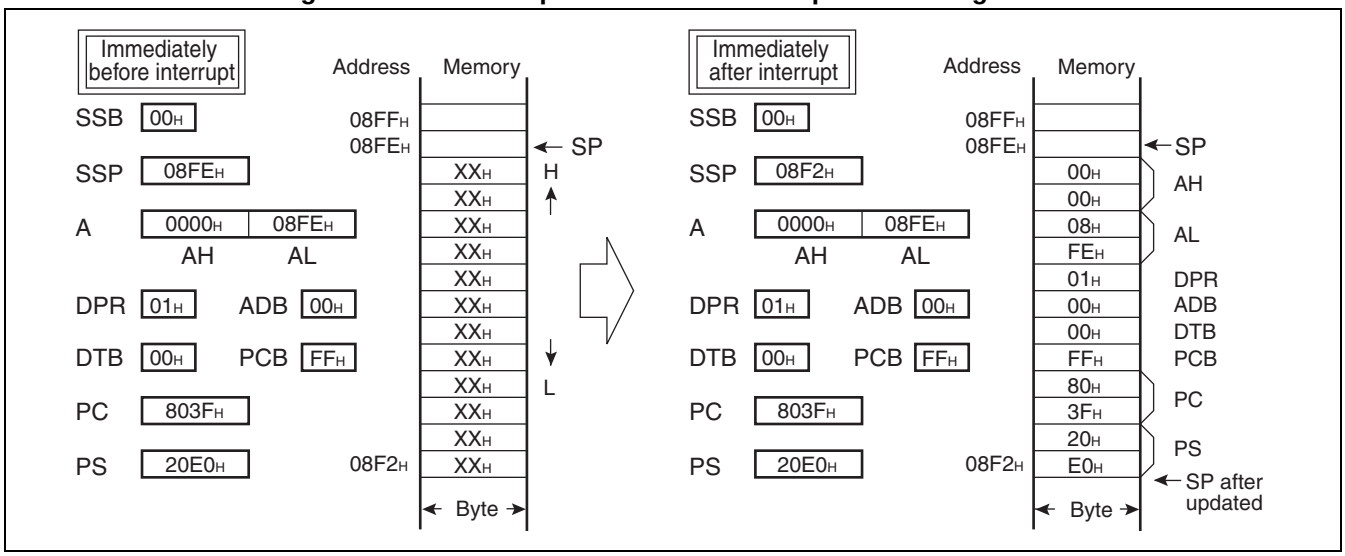
■ Stack Operation when Interrupt Handling Starts

When an interrupt is accepted, the CPU automatically saves the current contents of the dedicated registers to the system stack, in the following order:

1) Accumulator (A)
2) Direct page register (DPR)
3) Additional data bank register (ADB)
4) Data bank register (DTB)
5) Program bank register (PCB)
6) Program counter (PC)
7) Processor status register (PS)

Figure 3.8-1 shows stack operation at the beginning of interrupt handling.

Figure 3.8-1 Stack Operation when Interrupt Processing Starts



■ Stack Operation after Return from Interrupt Handling

When the interrupt return instruction (RETI) is executed at the end of interrupt handling, the values of PS, PC, PCB, DTB, ADB, DPR, and A are returned from the stack in the reverse order at the start of interrupt handling. This will restore the dedicated registers to their states immediately before the interrupt started.

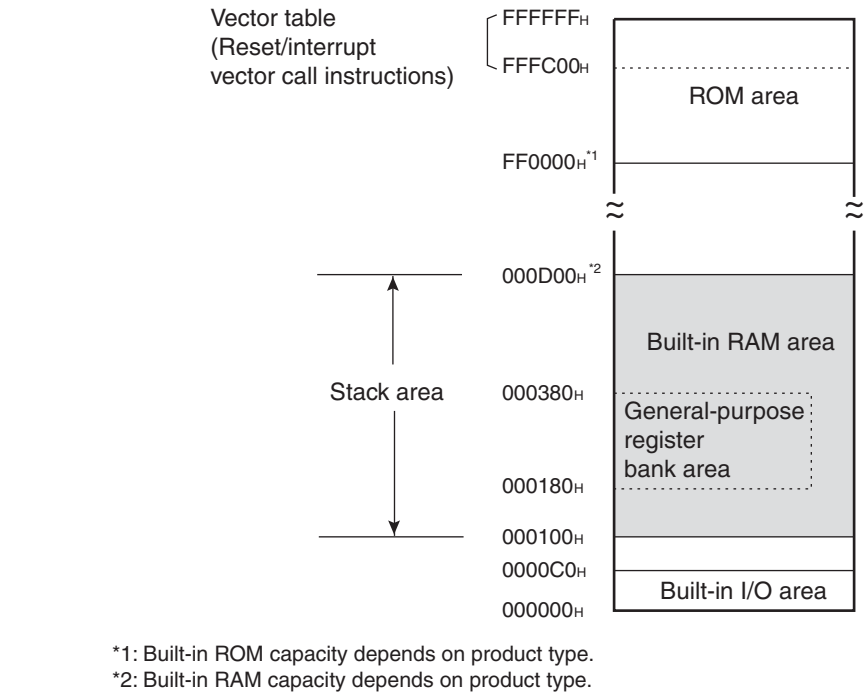
■ Stack Area

● Allocation of the stack area

The stack area is used for saving/returning the program counter (PC) as required for executing subroutine call instructions (CALL) and vector call instructions (CALLV) in addition to interrupt handling. Moreover, it is used for temporarily saving/ returning the contents of registers by PUSHW and POPW instructions. The stack area is allocated in RAM in addition to the data area.

Figure 3.8-2 shows the allocation of the stack area.

Figure 3.8-2 Stack Area



Notes:

- In ordinary cases, use even addresses for setting the addresses of stack pointers (SSP, USP).
- Avoid overlapped allocation of the system stack area, user stack area, and data area.

● System stack and user stack

For interrupt handling, the system stack area is used. Even if the user stack area is being used when an interrupt occurs, processing forcibly switches to the system stack. Thus, the system stack area must also be correctly prepared in a system where mainly the user stack area is used. Unless it is necessary to separate the available free stack space, use only the system stack.

3.9 Example Program for Interrupt Handling

An example program for interrupt handling is shown below.

■ Example Program for Interrupt Handling

This is an example of an interrupt handling program that uses the external interrupt 0 (INT0) instruction.

A coding example of the program is as follows.

[Coding example]

```
DDR5    EQU    000015H            ; Port 5 direction register
ENIR    EQU    000030H            ; DTP/interrupt enable register
EIRR    EQU    000031H            ; DTP/interrupt source register
ELVRL   EQU    000032H            ; Request level setting resister
ICR02   EQU    0000B2H            ; Interrupt control register 02
STACK   SSEG                      ; Stack
        RW     100
STACK_TRW      1
STACK   ENDS
;----------Main program-----------------------------------------------
CODE    CSEG
START:
        MOV    RP, #0             ; General-purpose register uses heading
                                  ; bank
        MOV    ILM, #07H          ; Set ILM in PS to level 7
        MOV    A, #!STACK_T       ; System stack setting
        MOV    SSB, A
        MOVW   A, #STACK_T        ; Stack pointer setting. Because the S-
        MOVW   SP, A              ; flag is 1, the stack pointer is
                                  ; set to SSP
        MOV    DDR5, #00000000B   ; Set P50/INT0 pin to input
        AND    CCR, #0E0H         ; Clear bit0 to bit4 of CCR in PS
        OR     CCR, #40H          ; Set I-flag in CCR in PS to enable
                                  ; interrupts
        MOV    I:ICR02, #00H      ; Set interrupt level to 0 (highest)
        MOV    I:ELVR, #00000001B ; Specify INT0 to be "H"-level request
        MOV    I:EIRR, #00H       ; Clear INT0 interrupt source
        MOV    I:ENIR, #01H       ; Enable INT0 input
        :
LOOP:   NOP                       ; Dummy loop
        NOP
        NOP
        NOP
        BRA    LOOP               ; Unconditional jump
```

```
;----------Interrupt program-----------------------------------------
ED_INT1:
       MOV    I:EIRR, #00H       ; Prohibit acceptance of new INT0
       NOP
       NOP
       NOP
       NOP
       NOP
       NOP
       RETI                      ; Return from interrupt
CODE   ENDS
;----------Vector setting--------------------------------------------
VECT   CSEG   ABS=0FFH
       ORG    0FFBCH             ; Specify a vector for interrupt
                                 ; #16(10H)
       DSL    ED_INT1
       ORG    0FFDCH             ; Reset vector setting
       DSL    START
       DB     00H                ; Set to single-chip mode
VECT   ENDS
       END    START
```

■ Specification of Processing for Sample Program of Extended Intelligent I/O Service (EI^{2}OS)

1) If "H" level is detected for the signal input to the INT0 pin, the extended intelligent I/O service (EI^{2}OS) will start.

2) If the INT0 pin enters "H" level, EI^{2}OS starts and transfers the data in port 0 to the memory at address 3000_H.

3) After the transfer of 100-byte data is completed, an interrupt is generated due to the end of EI^{2}OS transfer.

A coding example of the program is as follows.

[Coding example]

```
       .SECTION  IO,IO, LOCATE=0x000000
       .ORG      0015H
DDR5   .RES.B    01H           ; Port 5 direction register
       .ORG      0030H
ENIR   .RES.B    01H           ; DTP/Interrupt enable register
EIRR   .RES.B    01H           ; DTP/Interrupt source register
ELVR   .RES.B    01H           ; Request level setting register
       .ORG      00B2H
ICR02  .RES.B    01H           ; Interrupt control register 02
       .ORG      0100H
BAPL   .RES.B    01H           ; Lower byte of buffer address pointer
BAPM   .RES.B    01H           ; Middle byte of buffer address pointer
BAPH   .RES.B    01H           ; Upper byte of buffer address pointer
ISCS   .RES.B    01H           ; EI2OS status
IOAL   .RES.B    01H           ; Lower byte of I/O address pointer
IOAH   .RES.B    01H           ; Upper byte of I/O address pointer
DCTL   .RES.B    01H           ; Lower byte of data counter
DCTH   .RES.B    01H           ; Upper byte of data counter

       .SECTION  STACK,STACK   ; Stack
       .RES.B    0FEH
STACKT .RES.B    01H
;----------Main program-----------------------------------------------
       .SECTION     PROG,CODE
START:
       AND       CCR, #0BFH    ; Clear the I-flag in CCR in PS to
                               ; disable interrupts
       MOV       RP, #00       ; Set the register bank pointer
       MOV       A, #bnksym STACKT
                               ; Specify the system stack
       MOV       SSB, A
       MOVW      A, #STACKT    ; Set the stack pointer.
       MOVW      SP, A         ; In this case, SSP is set, because the
                               ; S-flag is set to "1".
       MOV       I:DDR5, #00000000B
                               ; Set the P50/INT0 pin to "input"
```

```
        MOV       BAPL, #00H    ; Set the buffer address to 003000H
        MOV       BAPM, #30H
        MOV       BAPH, #00H
        MOV       ISCS, #00010001B
                                ; I/O addresses not updated, but byte
                                ; transfer and buffer address updated
                                ; Transferred from I/O to buffer,
                                ; completed by peripheral function (resource)
        MOV       IOAL, #00H    ; Set the transfer source address (Port
                                ; 0:000000H)
        MOV       IOAH, #00H
        MOV       DCTL, #064H   ; Set the transfer byte count (100 bytes)
        MOV       DCTH,   #00H
        MOV       I:ICR02, #00001000B
                                ; EI2OS channel 0, EI2OS enabled
                                ; Interrupt level 0 (highest level)
        MOV       I:ELVR, #00000001B
                                ; Specify INT0 to be "H"-level request.
        MOV       I:EIRR, #00H  ; Clear INT0 interrupt source
        MOV       I:ENIR, #01H  ; INT0 interrupt enabled
        MOV       ILM, #07H     ; Set ILM in PS to level 7.
        AND       CCR, #0E0H    ; Clear bit0 to bit4 of CCR in PS.
        OR        CCR,    #040H; Set the I-flag in CCR in PS to
                                ; enable interrupts
        :
LOOP:   BRA       LOOP ; Infinite loop
;----------Interrupt program-----------------------------------------
WARI    CLRB        EIRR:0      ; DTP/interrupt request flag cleared
        :
        User processing         ; Checking EI2OS completion source,
        :                       ; processing of data in the buffer, EI2OS
                                ; reset, etc.
        RETI

;----------Vector setting--------------------------------------------
        .SECTION  VECT,CODE,  LOCATE=0xFFFF54
        .ORG      0FFFFBCH      ; Set vector to interrupt #16(10H)
        .DATA.E   WARI
        .ORG      0FFFFDCH      ; Reset vector setting
        .DATA.E   START         ;
        .DATA.B   00H           ; Mode data setting

        END       START
```

4. Reset

This chapter describes the reset operation.

4.1 Outline of Reset

4.2 Reset Sources and Oscillation Stabilization Wait Time

4.3 External Reset Pin

4.4 Reset Operation

4.5 Reset Source Bit

4.6 State of Each Pin by Reset

4.7 Reset Output Function

4.1 Outline of Reset

If a reset source occurs, the CPU immediately suspends the processing currently being executed and enters the reset clear wait state. After the reset is cleared, processing starts at the address indicated by the reset vector.
There are six reset sources:

- **Power-on reset**
- **External reset request from $\overline{\text{RST}}$ pin**
- **Software reset request**
- **Watchdog timer overflow**
- **Lower power voltage detected**
- **Counter overflow of CPU operation detection function**

■ Reset Sources

The reset sources are shown in Table 4.1-1.

Table 4.1-1 Reset Sources

Reset	Source	Machine clock	Watchdog timer	Oscillation stabilization wait
Power-on	System powered on	Main clock (MCLK)	Stopped	Yes
External pin	"L" level input to $\overline{\text{RST}}$ pin	Main Clock (MCLK)	Stopped	No
Software	Set the internal reset signal generation bit (RST) in the low-power consumption mode control register (LPMCR) to "0".	Main clock (MCLK)	Stopped	No
Watchdog timer	Watchdog timer overflow	Main clock (MCLK)	Stopped	No
Low-voltage detection	Low-voltage of power supply detected	Main clock (MCLK)	Stopped	Yes
CPU operation detection function	Counter overflow of CPU operation detection function	Main clock (MCLK)	Stopped	No

MCLK: Main clock (divide-by-2 clock of oscillation clock)

● Power-on reset

Power-on reset is a reset that occurs when the power is turned on. The oscillation stabilization wait time is fixed to 2^{16} oscillation clock cycles (2^{16}/HCLK). When the oscillation stabilization wait time has elapsed, the reset is executed.

● External reset

External reset is a reset that occurs when "L" level is input to the external reset pin ($\overline{\text{RST}}$ pin). The required time period for

"L" level input to $\overline{RST}$ pin is 16 machine cycles (16/ϕ) or more. For an external reset, no oscillation stabilization wait time applies.

Note:

Only if the $\overline{RST}$ pin generates a reset request, when a reset source occurs during a write operation (while a transfer type instruction such as MOV is executed), the system enters reset clear wait state after the end of the instruction. This ensures that the write operation completes normally when a reset occurs during the write operation.
However, string type instruction (such as MOVS) accepts resets before the transfer for the specified counter is completed. For this reason, it cannot be assured that all data will be transferred in this case.

- Software resets

In a software reset, an internal reset is performed if the internal reset signal generation bit (RST) in the low-power consumption mode control register (LPMCR) is cleared to "0". The oscillation stabilization wait time does not apply for software resets.

- Watchdog resets

In a watchdog reset, a reset is performed when the watchdog timer overflows, if the watchdog control bit (WTE) in the watchdog timer control register (WDTC) is not cleared to "0" within the specified time after starting the watchdog timer. The oscillation stabilization wait time does not apply for watchdog resets.

- Low-voltage detection resets

In a low-voltage detection reset, a reset occurs if the voltage of the power supply is lower than specified value. The oscillation stabilization wait time is fixed to 2^{16} oscillation clock cycles (2^{16}/HCLK). When the oscillation stabilization wait time has elapsed, the reset is executed. This function always works after the power turned on.

- CPU operation detection resets

In CPU operation detection resets, a reset is performed when the CPU operation detection function counter overflows, if the CPU operation detection circuit clear bit (CL) in the low-voltage/CPU operation detection reset control register (LVRC) is not cleared to "0" within the specified time after power-on. This function always works after the power turned on.

Definitions of the clock:

HCLK : Oscillation clock frequency
MCLK : Main clock frequency
SCLK : Sub clock frequency
ϕ : Machine clock frequency (CPU operation clock)
1/ϕ : Machine cycle (CPU operation clock cycle)

Refer to Section "5.1 Clock" for details.

Note:

If a reset is generated in stop mode or sub clock mode, (2^{16}/HCLK (about 16.39ms when using an oscillator of HCLK=4MHz) is used as the oscillation stabilization wait time.

Refer to Section "5.1 Clock" for details.

4.2 Reset Sources and Oscillation Stabilization Wait Time

The MB90920 series has 6 types of reset sources. The oscillation stabilization wait time at reset depends on the reset source.

■ Reset Sources and Oscillation Stabilization Wait Time

Table 4.2-1 shows the reset sources and oscillation stabilization wait time.

Table 4.2-1 Reset Sources and Oscillation Stabilization Wait Time

Reset	Reset source	Oscillation stabilization wait time ( ): for an oscillation clock frequency of 4 MHz
Power-on	System powered on	2^{13}/HCLK (approx. 2.05ms) + 2^{16}/HCLK (approx. 16.39ms) = approx. 18.44ms Note: 2^{13}/HCLK (approx. 2.05ms) is the stabilization time of the step-down circuit.
Watchdog	Watchdog timer overflow	Not used; WS1 and WS0 bit are initialized to 11_B.
External	"L" input from $\overline{RST}$ pin	Not used; WS1 and WS0 bits are initialized to 11_B however.
Low-voltage detection	Low-power voltage detected	2^{16}/HCLK (approx. 16.39ms)
CPU operation detection	CPU operation detection timer overflow	Not used; WS1 and WS0 bits are initialized to 11_B however.
Software	RST bit in the low-power consumption mode control register (LPMCR) set to "0"	Not used; WS1 and WS0 bits are initialized to 11_B however.

HCLK : Oscillation clock frequency
WS1, WS0 : Oscillation stabilization wait time selection bit in the clock selection register (CKSCR)

Figure 4.2-1 shows the oscillation stabilization wait time when a power-on reset occurs.

Figure 4.2-1 Oscillation Stabilization Wait Time for Power-on Reset

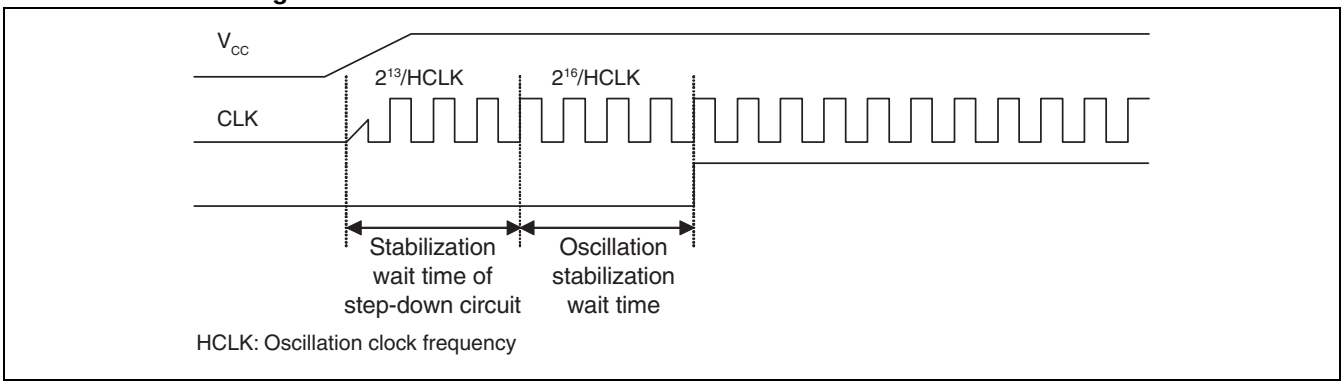


Table 4.2-2 Oscillation Stabilization Wait Time Depending on Clock Selection Register (CKSCR) Settings

WS1	WS0	Oscillation stabilization wait time ( ): for an oscillation clock frequency of 4MHz
0	0	2^{15}/HCLK (approx. 8.19ms)
0	1	2^{13}/HCLK (approx. 2.05ms)
1	0	2^{14}/HCLK (approx. 4.10ms)
1	1	2^{16}/HCLK (approx. 16.39ms) (other than power-on reset) 2^{13}/HCLK (approx. 2.05ms) + 2^{16}/HCLK (approx. 16.39ms)= approx. 18.44ms (at power-on reset)

HCLK: Oscillation clock frequency

Note:

Ceramic and crystal resonators generally need the oscillation stabilization wait time of several ms to a dozen ms or so from the starting of the oscillation until the stabilization to the natural frequency. Therefore, set an appropriate value for the resonator used. Refer to Section "5.1 Clock" for details.

■ Oscillation Stabilization Wait Reset State

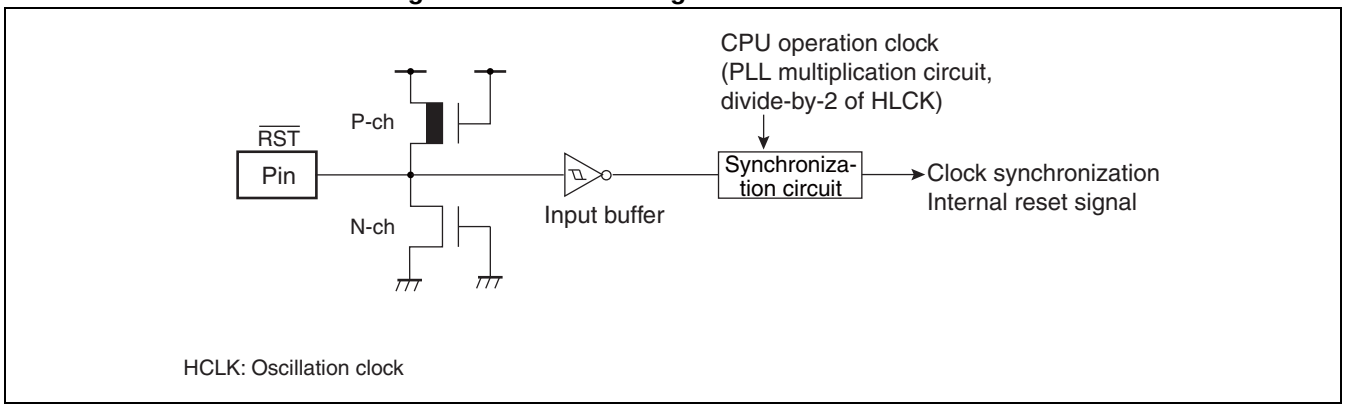
The reset operation for the power-on reset and the reset in stop mode and sub clock mode will be performed after the oscillation stabilization wait time set by the time-base timer has elapsed. If, in this case, the external reset input is not cleared, the reset operation will be performed after the external reset is released.

4.3 External Reset Pin

The external reset pin ($\overline{\text{RST}}$ pin) is a reset-input dedicated pin which generates an internal reset if "L" level is input. The MB90920 series starts reset operations in synchronization with CPU operation clock, however, only resets through external pins are performed asynchronously.

■ Block Diagram of External Reset Pin

Figure 4.3-1 Block Diagram for Internal Reset



Note:

Inputs to the $\overline{\text{RST}}$ pin are accepted during cycles in which memory is not affected in order to prevent memory from being destroyed by a reset during a write operation.
A clock is required to initialize the internal circuit. In particular, an operation with an external clock requires clock input together with reset input.

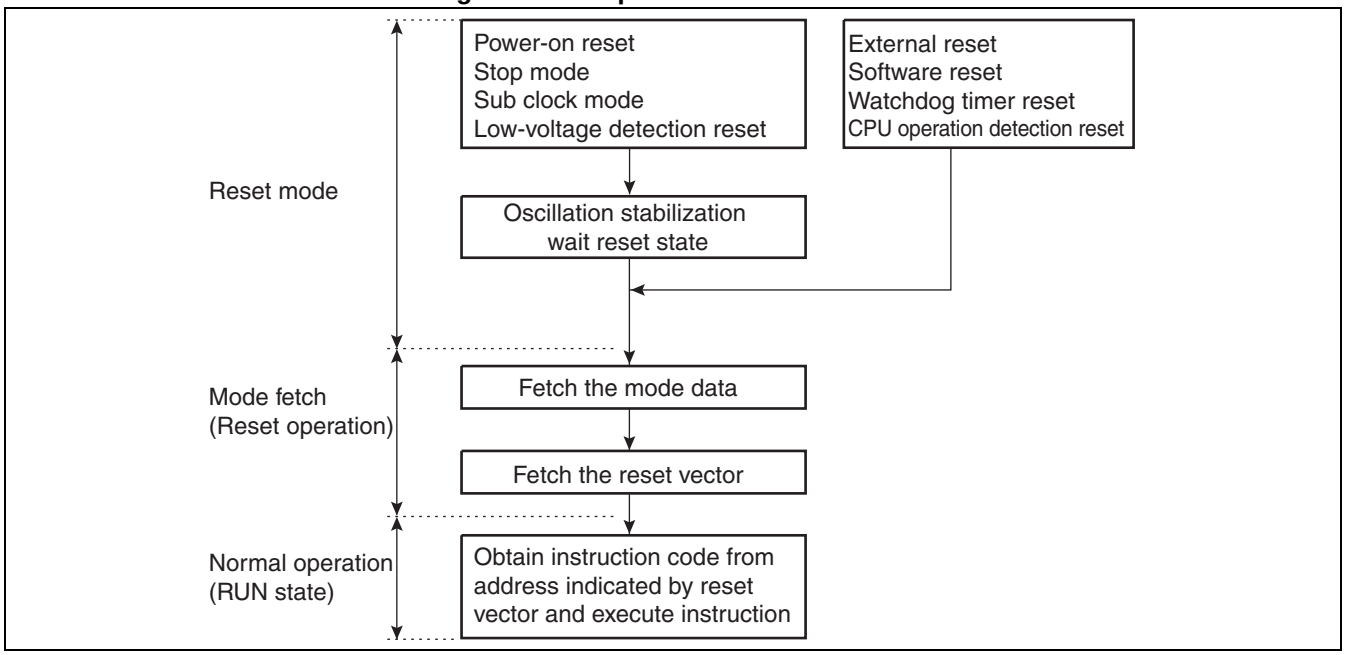
4.4 Reset Operation

If a reset is released, the target for reading mode data and the reset vector is selected based on the setting of the mode pins, and a mode fetch is performed. With this mode fetch operation, the CPU operation mode and the execution start address after the reset operation is completed are determined. At power-on, as well as at return from sub clock mode or stop mode by reset, mode fetch is performed after the oscillation stabilization wait time has elapsed.

■ Outline of Reset Operation

Figure 4.4-1 shows the operation flow for reset.

Figure 4.4-1 Operation Flow for Reset



■ Mode Pins

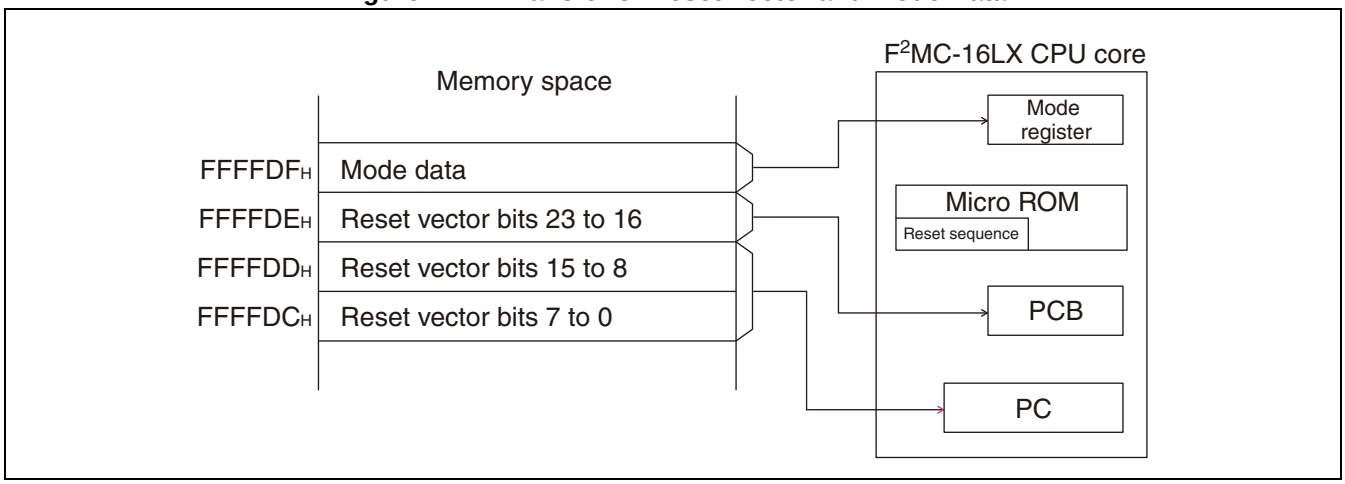
The mode pins (MD2 to MD0) specify how to fetch the reset vector and mode data. Reset vector and mode data are read during the reset sequence. For further information about mode pins, see Section "7.2 Mode Pins (MD2 to MD0)".

■ Mode Fetch

If a reset is released, the CPU performs a hardware-based transfer of the reset vector and mode data to the relevant register inside the CPU core. Reset vector and mode data are allocated in a 4-byte area at the addresses $FFFFDC_H$ to $FFFFDF_H$. The CPU outputs these addresses to the bus immediately after the reset is released to fetch the reset vector and mode data. With this mode fetch operation, the CPU starts processing from the address indicated by the reset vector.

Figure 4.4-2 shows the transfer of reset vector and mode data.

Figure 4.4-2 Transfer of Reset Vector and Mode Data



- Mode data (Address: $FFFFDF_H$)

The mode register setting can be changed only by a reset operation, and the setting becomes effective after the reset operation. For further information about mode data, see Section "7.3 Mode Data".

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

The execution start address after the reset operation is written to this area. Execution will start from the address in this vector.

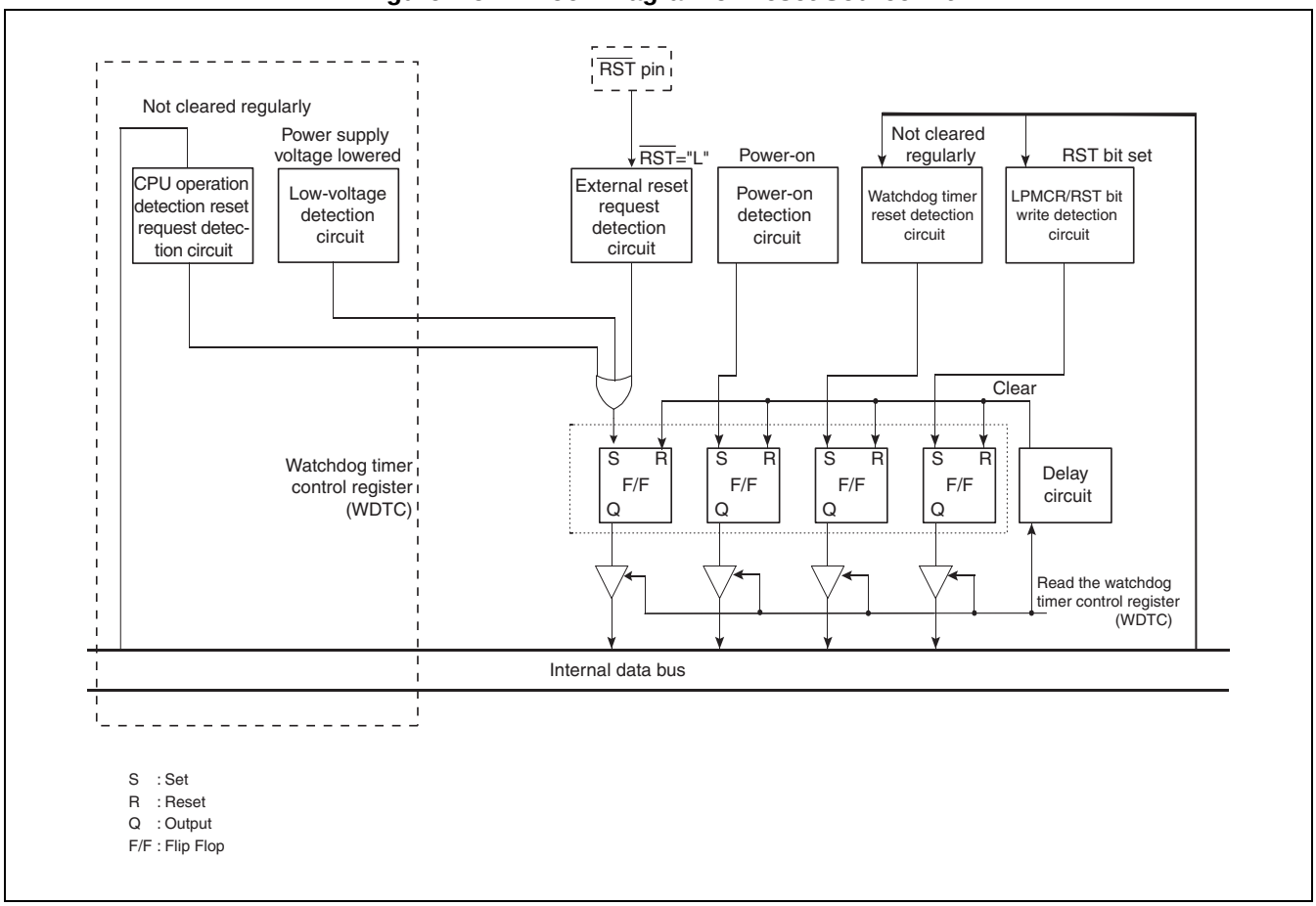
4.5 Reset Source Bit

The source for reset generation can be identified by reading the watchdog timer control register (WDTC) and the low-voltage/CPU operation detection reset control register (LVRC).

■ Reset Source Bit

Each reset source has the corresponding flip flop as shown in Figure 4.5-1. These contents can be obtained by reading the watchdog timer control register (WDTC). If, after the reset is released, the reset source must be identified, process the read value of the watchdog timer control register (WDTC) with software and branch to the appropriate program.

Figure 4.5-1 Block Diagram of Reset Source Bit



■ Correspondence Between Reset Source Bit and Reset Source

The configuration of reset source bits in the watchdog timer control register (WDTC) is shown in Figure 4.5-2, and the correspondence between the contents of reset source bits and reset sources is shown in Table 4.5-1. For further information, see Section "9.3.1 Watchdog Timer Control Register (WDTC)".

Figure 4.5-2 Configuration of Reset Source Bits (Watchdog Timer Control Register)

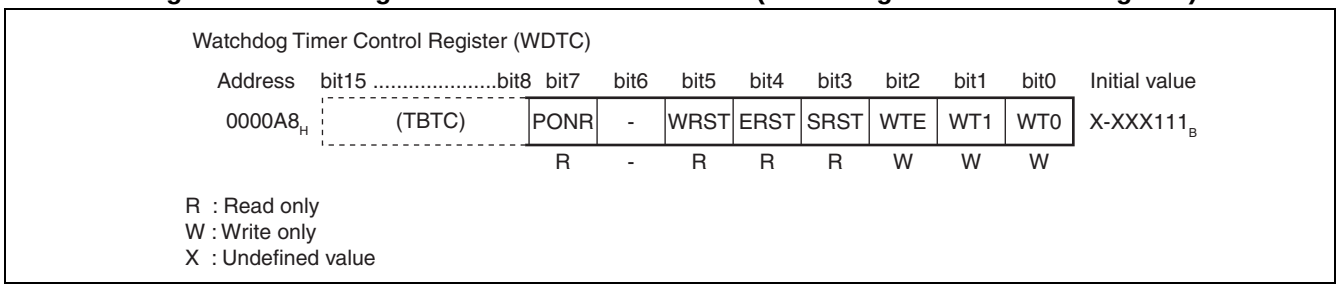


Table 4.5-1 Correspondence between the Contents of Reset Source Bits and Reset Source

Reset source	PONR	WRST	ERST	SRST
Power-on reset request generated	1	X	X	X
Reset request generated due to watchdog timer overflow	△	1	△	△
External reset requested by $\overline{RST}$ pin, CPU operation detection reset request generated *2	△	△	1	△
Low-voltage detection reset request generated *1	1	△	1	△
Software reset request generated	△	△	△	1

△ : Retains the previous state

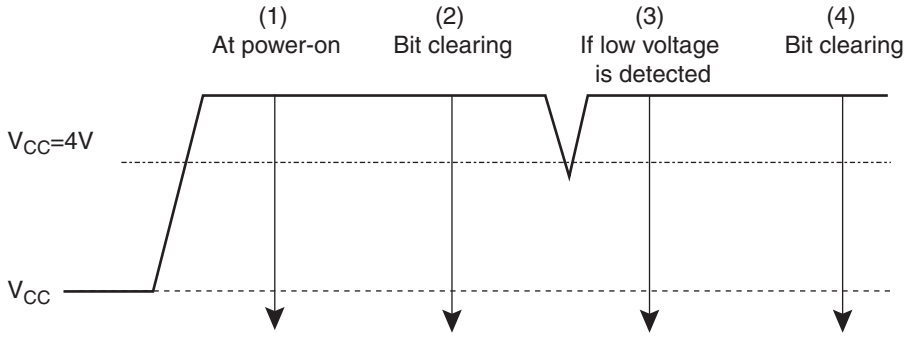
X: Undefined value

*1: If a low-voltage detection reset request is generated, the LVRF bit in the low-voltage/CPU operation detection reset control register (LVRC) is set to "1".

*2: If a CPU operation detection reset request is generated, the CPUF bit in the low-voltage/CPU operation detection reset control register (LVRC) is set to "1".

■ State of Reset Source Bits

Figure 4.5-3 State of Reset Source Bits



	(1)		(2)		(3)		(4)
PONR bit (Power-on or LVRF = 1)	1	→	0	→	1	→	0
ERST bit (External reset input, CPU operation detection)	"1" or "0"	→	0	→	1	→	0
LVRF bit* (Low voltage detection, 4.2V ±0.2V)	"1" or "0"	→	0	→	1	→	0

*: The LVRF bit is in the low voltage/ CPU operation detection reset control register (LVRC).

(1) At power-on

When power is turned on, the power-on reset bit (PONR) and LVRF bit are set to "1". However, if power is turned on without an ordinary startup, LVRF bit may be set to "0".

(2) Bit clearing (Clearing bits by reading the WDTC register and writing "0" to LVRF bit)

(3) If low voltage (4.2V ±0.2V) is detected

If low voltage (4.2V ±0.2V) is detected, LVRF bit, PONR bit, and ERST bit are set to "1".

(4) Bit clearing (Clearing bits by reading the WDTC register and writing "0" to LVRF bit)

■ Notes on the Reset Source Bit

● When multiple reset sources occur

When multiple reset sources occur, the corresponding reset source bits in the watchdog timer control register (WDTC) are set to "1". For example, if an external reset request from the $\overline{\text{RST}}$ pin and overflow of the watchdog timer occur at the same time, the ERST and WRST bits are both set to "1".

● Clearing the reset source bit

The reset source bit is cleared only if the watchdog timer control register (WDTC) is read out. A flag that is set to the corresponding reset source bit will not be cleared and will remain set to "1" even if other reset sources cause a reset thereafter.

4.6 State of Each Pin by Reset

This section describes the state of each pin by reset.

■ State of Pins During Reset

The state of a pin during reset is determined by the setting of the mode pins (MD2 to MD0=011_B).

For the state of each pin during reset, see Section "6.7 Pin State in the Standby Mode and at the Time of Reset".

- Setting internal vector mode

I/O pins (peripheral function pins) are all set to high impedance and internal ROM becomes the target for reading the mode data.

■ State of Pin After Reading the Mode Data

The state of a pin after the mode data is read out is determined by the mode data (M1, M0=00_B).

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

I/O pins (peripheral function pins) are all set to high impedance and internal ROM becomes the target for reading the mode data.

Note:

Make sure that, if a pin is set to high impedance because a reset source is occurred, this does not cause incorrect operation of devices connected to the pin.

4.7 Reset Output Function

If all 6 types of reset sources (Section "4.2 Reset Sources and Oscillation Stabilization Wait Time") occur, "L" is output from RSTO pin.

■ Reset Output Function

Upon reception of a reset by MB90920 series, "L" is output from RSTO pin during reset.

RSTO pin is N-ch open drain.

5. Clock

This chapter describes the clock.

5.1 Clock

5.2 Block Diagram of the Clock Generation Block

5.3 Clock Selection Register (CKSCR)

5.4 PLL/Sub Clock Control Register (PSCCR)

5.5 Clock Mode

5.6 Oscillation Stabilization Wait Time

5.7 Connection of Oscillator and External Clock

5.1 Clock

The clock generation block controls the internal clock which is the operation clock of the CPU and peripheral functions. The clock which is generated in the clock generation block is referred to as a machine clock and its one cycle as a machine cycle. In addition, the clock which is provided from a high-speed oscillator is referred to as an oscillation clock and the 2-frequency division of the oscillation clock is referred to as a main clock. The 4- or 2-frequency division of the clock provided from a low-speed oscillator is referred to as a sub clock, and a clock with the PLL oscillation is referred to as a PLL clock.

■ Clock

The clock generation block contains the oscillation circuit that generates the oscillation clock with connecting the oscillator to the oscillation pin. The oscillation clock can also be supplied with inputting an external clock to the oscillation pin. The clock generation block also contains the PLL clock multiplier circuit, which can generate six clocks whose frequencies are multiplication of the oscillation clock frequency. The clock generation block controls the oscillation stabilization wait time and PLL clock multiplication, and performs switching operation of the internal clock with the clock selector.

- Oscillation clock (HCLK)

The oscillation clock is generated either with connecting the oscillator to high-speed oscillation pins (X0 and X1) or with inputting an external clock.

- Main clock (MCLK)

The clock has the divided-by-two frequency of the oscillation clock. It supplies the input clock to the time-base timer and clock selector.

- Sub clock (SCLK)

The sub clock is generated with dividing the clock, which was generated with connecting an oscillator to the low-speed oscillation pins (X0A and X1A) or with inputting an external clock, by 4 or 2. The division ratio of the sub clock is set with SCDS bit in the PLL/sub clock control register (PSCCR). This clock can be used as the operating clock in the watch timer or the low-speed machine clock.

- PLL clock (PCLK)

The PLL clock is obtained by multiplying the oscillation clock with the PLL clock multiplier circuit (PLL oscillation circuit). Depending on the multiplication rate selection bits (CKSCR: CS1 and CS0, PSCCR: CS2), one of 6 types of clocks can be selected.

- Machine clock

This is the operating clock for the CPU and peripheral functions. One cycle of the machine clock is defined as a machine cycle (1/ϕ). The machine clock can be selected among any of the main clock, sub clock, or 6 types of PLL clocks.

Note:

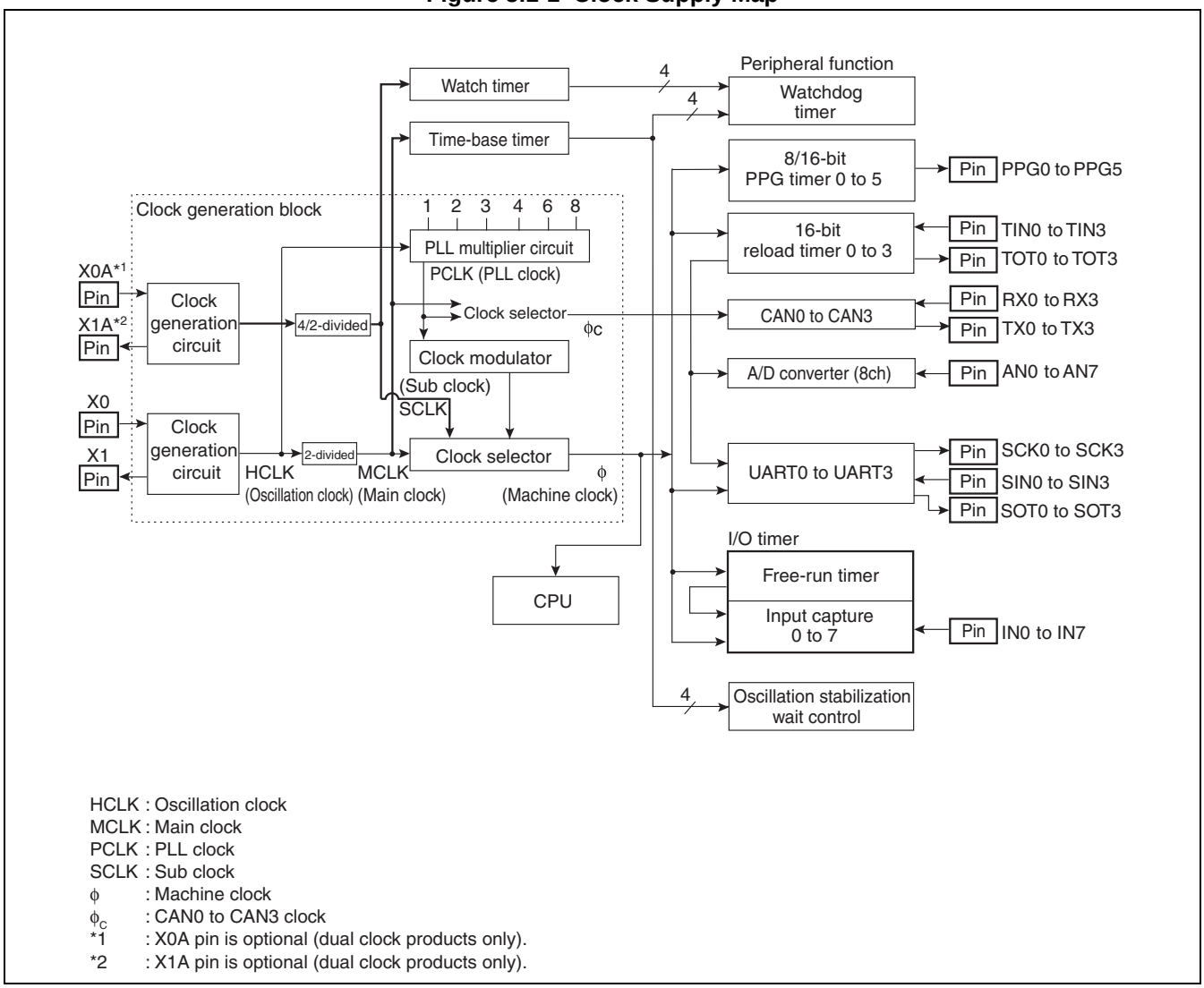
When the operating voltage is 5V, an oscillation clock of 3 MHz to 16 MHz can be generated. When inputting an external clock, an external clock from 3 MHz to 32 MHz can be used. The maximum operating frequency for the CPU and peripheral functions is 32 MHz. Therefore, if the multiplier rate that exceeds the maximum operating frequency is set, devices will not operate correctly. Therefore, when 32 MHz of external clock is input, only 1 can be set for the multiplication rate of the PLL clock. Although the PLL oscillation operates at the range of 4 MHz to 32 MHz, the PLL oscillation range depends on the operating voltage and multiplication rate. Refer to "Data sheet" for details.

■ Clock Supply Map

Machine clock generated in the clock generation block is supplied as the operating clock for the CPU and peripheral functions. The operation of CPU and peripheral functions are affected by switching (clock mode) between a main clock and sub clock/ PLL clock and by a change in the PLL clock multiplication rate. Since some peripheral functions receive divided output from a time-base timer, a peripheral can select a operating clock.

Figure 5.1-1 shows the clock supply map.

Figure 5.1-1 Clock Supply Map



HCLK : Oscillation clock
MCLK : Main clock
PCLK : PLL clock
SCLK : Sub clock
ϕ : Machine clock
ϕ_C : CAN0 to CAN3 clock
*1 : X0A pin is optional (dual clock products only).
*2 : X1A pin is optional (dual clock products only).

5.2 Block Diagram of the Clock Generation Block

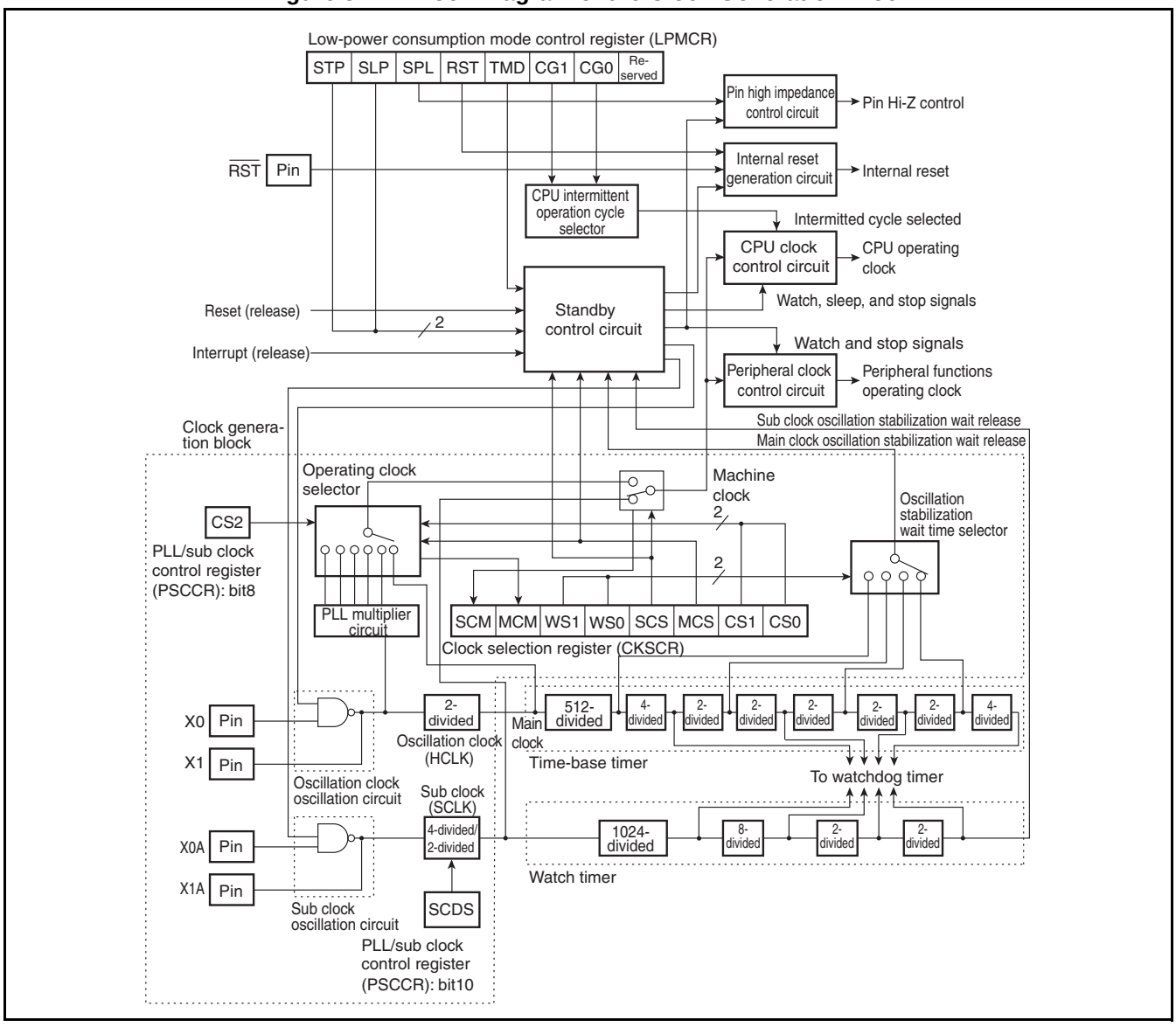
The clock generation block consists of the following blocks:

- **Oscillation clock generation circuit/sub clock generation circuit**
- **PLL multiplier circuit**
- **Clock selector**
- **Clock selection register (CKSCR)**
- **PLL/sub clock control register (PSCCR)**
- **Oscillation stabilization wait time selector**

■ Block Diagram of the Clock Generation Block

Figure 5.2-1 shows a block diagram of the clock generation block. The standby control circuit and time-base timer circuit are included also.

Figure 5.2-1 Block Diagram of the Clock Generation Block



- Oscillation clock generation circuit

The oscillation clock (HCLK) is generated either with connecting the oscillator to the high-speed oscillation pins (X0 and X1) or with inputting an external clock.

- Sub clock generation circuit

The sub clock (SCLK) is generated either with connecting the oscillator to the low-speed oscillation pins (X0A and X1A) or with inputting an external clock.

- PLL multiplier circuit

The PLL multiplier circuit multiplies the oscillation clock with the PLL oscillation and supplies the clock as a PLL clock (PCLK) to the clock selector.

- Clock selector

It selects a clock to be supplied to the CPU and peripheral functions among the main clock, sub clock, and 6 types of PLL clocks.

- Clock selection register (CKSCR)

The clock selection register switches an oscillation clock/PLL clock and main clock/sub clock, and selects an oscillation stabilization wait time and PLL clock multiplication rate.

- PLL/sub clock control register (PSCCR)

This register selects the PLL clock multiplication rate (selects with the setting of CS0 and CS1 bits in the clock selection register and CS2 bit in this register), and sets the sub clock division ratio (divide-by-two/divide-by-four).

- Oscillation stabilization wait time selector

This selector selects an oscillation stabilization wait time for the oscillation clock. Selection is made from among 4 types of time-base timer outputs.

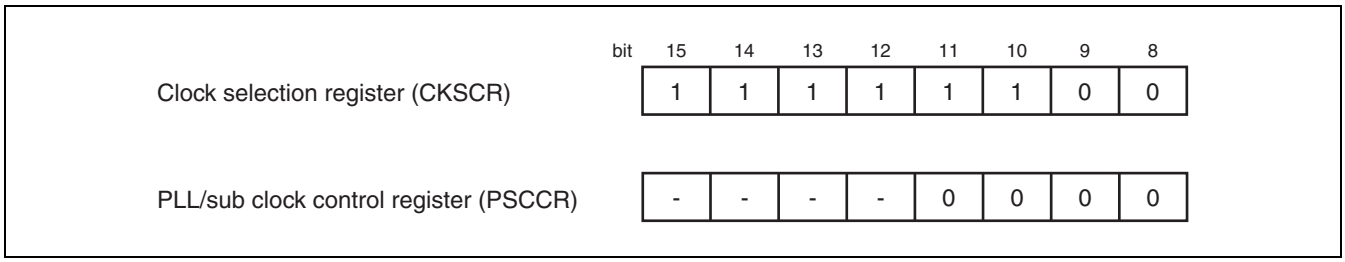
5.2.1 Register in the Clock Generation Block

This section explains the register in the clock generation block.

■ List of the Registers in the Clock Generation Block and Its Initial Values

Figure 5.2-2 lists the clock selection register and its initial values.

Figure 5.2-2 List of Clock Selection Register and Its Initial Values



5.3 Clock Selection Register (CKSCR)

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

■ Clock Selection Register (CKSCR)

Figure 5.3-1 Clock Selection Register (CKSCR)

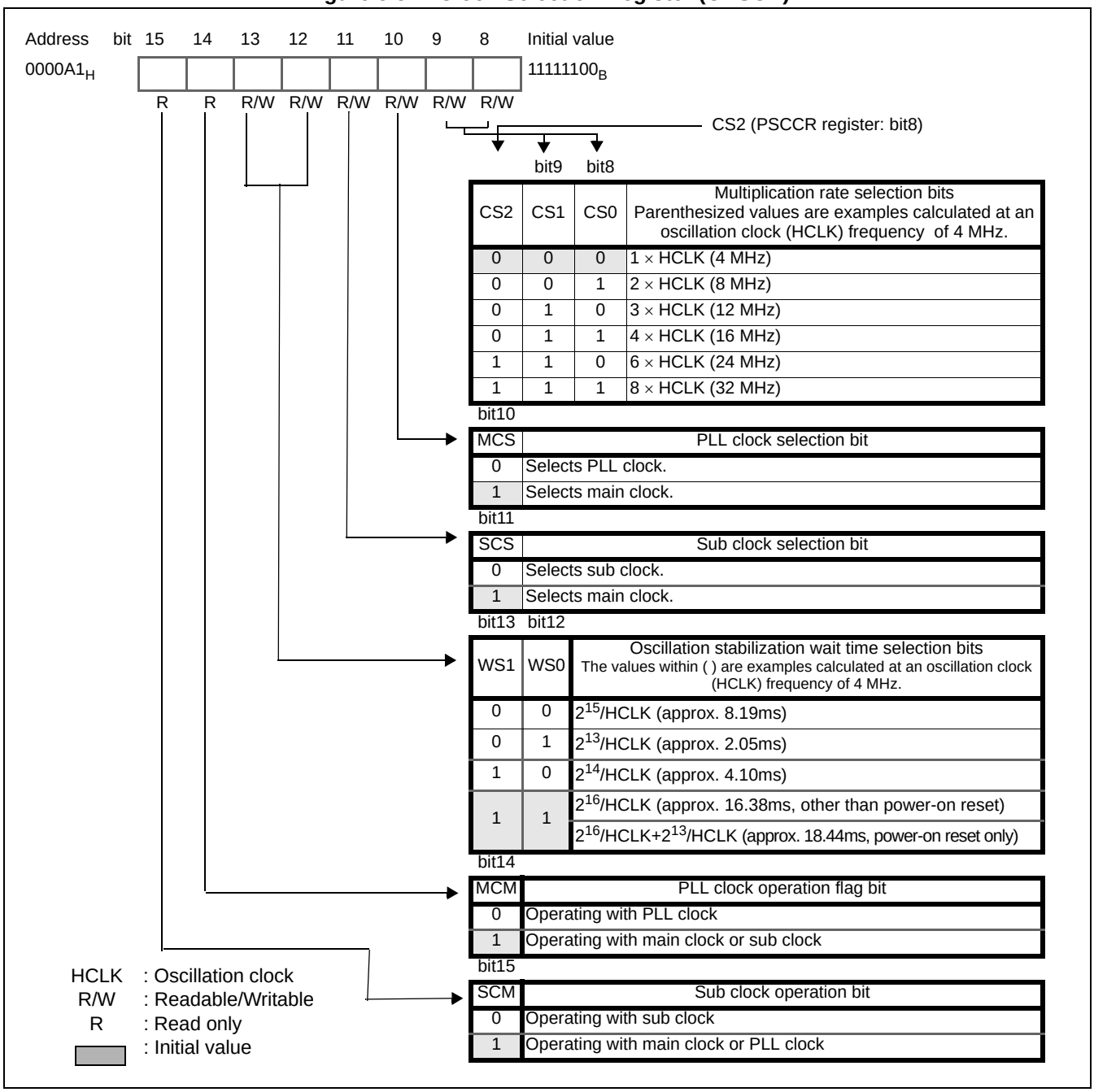


Table 5.3-1 Functions of Clock Selection Register (CKSCR) (Sheet 1 of 3)

Bit name		Function
bit15	SCM: Sub clock operation flag bit	Indicates whether the main clock or sub clock has been selected as the machine clock. • When the sub clock operation flag bit (CKSCR: SCM) is "0" and the sub clock selection bit (CKSCR: SCS) is "1", it indicates that it is in the transition period from the sub clock to the main clock. Also, the sub clock operation flag bit (CKSCR: SCM) is "1" and the sub clock selection bit (CKSCR: SCS) is "0", it indicates that it is in the transition period from the main clock to the sub clock. • Writing has no effect on operation.
bit14	MCM: PLL clock operation flag bit	Indicates whether the main clock or PLL clock has been selected as the machine clock. • When the PLL clock operation flag bit (CKSCR: MCM) is "1" and the PLL clock selection bit (CKSCR: MCS) is "0", it indicates that it is during the PLL clock oscillation stabilization wait time. • Writing has no effect on operation.
bit13, bit12	WS1 and WS0: Oscillation stabilization wait time selection bits	Selects the oscillation stabilization wait time of the oscillation clock when the stop mode is canceled, when the transition from sub clock mode to main clock mode occurs, and when the transition from sub clock mode to PLL clock mode occurs. • Selects one of four time-base timer outputs. Returned to the initial value with any reset. Note: Set the oscillation stabilization wait time to an appropriate value depending on the oscillator used. Refer to "7.2 Reset Source and Oscillation Stabilization Wait times" for details. When the switching from the main clock mode to PLL clock mode is executed, the oscillation stabilization wait time is fixed at 2^{14}/HCLK (during operation at an oscillation clock frequency of 4 MHz: approx. 4.1 ms). When the CPU switches from sub clock mode to PLL clock mode or when it returns from PLL stop mode to PLL clock mode, the oscillation stabilization wait time follows the values specified in these bits. Since the PLL clock oscillation stabilization wait time requires 2^{14}/HCLK or more, for switching from sub clock mode to PLL clock mode and transiting to PLL stop, set these bits to 10_B or 11_B.

Table 5.3-1 Functions of Clock Selection Register (CKSCR) (Sheet 2 of 3)

Bit name		Function
bit11	SCS: Sub clock selection bit	Specifies whether the main clock or sub clock to be selected as the machine clock. • When the machine clock is switched from the main clock to sub clock (CKSCR: SCS=1 to 0), the main clock mode changes to the sub clock mode of 1/SCLK (when the oscillation clock frequency is 32,768 kHz and divide-by-four setting: approx. 130 μs) in synchronization with the sub clock. • When the machine clock is switched from the sub clock to the main clock (CKSCR: SCS=0 to 1), the clock mode changes to the main clock mode after the main clock oscillation stabilization wait time is generated. The time-base timer is cleared automatically. Any reset causes the bit to return to the initial value. Notes: • When both MCS and SCS bits are "0", the SCS bit is preferred, and it is set to the sub clock mode. • When both sub clock selection bit (CKSCR: MCS) and PLL clock selection bit (CKSCR: SCS) are "0", the sub clock is preferred. • When switching from the main clock to the sub clock (CKSCR: SCS=1 to 0), write after disabling time-base timer interrupt with the time-base timer interrupt enable bit (TBTC; TBIE), or interrupt level mask register (ILM: ILM2 to 0). • When turning on the power or releasing from the stop mode, a sub clock oscillation stabilization wait time 2^{14}/SCLK (when the oscillation clock frequency is 32.768 kHz and divide-by-four setting: approx. 2 seconds) is generated. Therefore, if the switching from the main clock mode to the sub clock mode is executed during that period, an oscillation stabilization wait time is generated.
bit10	MCS: PLL clock selection bit	Specifies whether the main clock or PLL clock to be selected as the machine clock. When the machine clock is switched from the main clock to the PLL clock (CKSCR: MCS=1 to 0), the clock mode changes to the PLL clock mode after the PLL clock oscillation stabilization wait time is generated. The time-base timer is cleared automatically. When the switching from the main clock mode to PLL clock mode is executed, the oscillation stabilization wait time is fixed at 2^{14}/HCLK (during operation at an oscillation clock frequency of 4 MHz: approx. 4.1ms). When the switching from the sub clock mode to the PLL clock mode is executed, the oscillation stabilization wait time depends on the value set in the oscillation stabilization wait time selection bits (CKSCR: WS1 and WS0) Returns to the initial value by any reset. Notes: • When both MCS and SCS bits are "0", the SCS bit is preferred, and it is set to the sub clock mode. • When switching from the main clock to the PLL clock (CKSCR: MCS=1 to 0), write after disabling time-base timer interrupt with the time-base timer interrupt enable bit (TBTC: TBIE), or interrupt level mask register (ILM: ILM2 to 0).

Table 5.3-1 Functions of Clock Selection Register (CKSCR) (Sheet 3 of 3)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit9, bit8</td><td>CS1 and CS0:
Multiplication rate selection bits</td><td>

These bits and CS2 bit in the PLL/sub clock control register (PSCCR) select a multiplication rate for the PLL clock.
One of six types of PLL clock multiplication rate can be selected.
Any reset causes the bits to return to the initial value.
Setting of CS0, CS1, and CS2.

<table>
<tr><th>CS2</th><th>CS1</th><th>CS0</th><th>PLL clock multiplication rate</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>×1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>×2</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>×3</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>×4</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>×6</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>×8</td></tr>
</table>
Note:
When the PLL clock is selected (CKSCR: MCS=0), writing is inhibited. To rewrite the multiplier, write "1" to the PLL clock selection bit (CKSCR: MCS), rewrite the multiplication rate selection bits (CKSCR: CS1 and CS0), then set the PLL clock selection bit (CKSCR: MCS) back to "0".
</td></tr>
</table>

5.4 PLL/Sub Clock Control Register (PSCCR)

The PLL/sub clock control register selects the PLL multiplication rate and sub clock division ratio. This register is write-only. The value read from all bits is"1".

■ PLL/Sub Clock Control Register (PSCCR)

Figure 5.4-1 shows the configuration of the PLL/sub clock control register (PSCCR). Table 5.4-1 describes the function of each bit in the PLL/sub clock control register (PSCCR).

Figure 5.4-1 Configuration of PLL/Sub Clock Control Register (PSCCR)

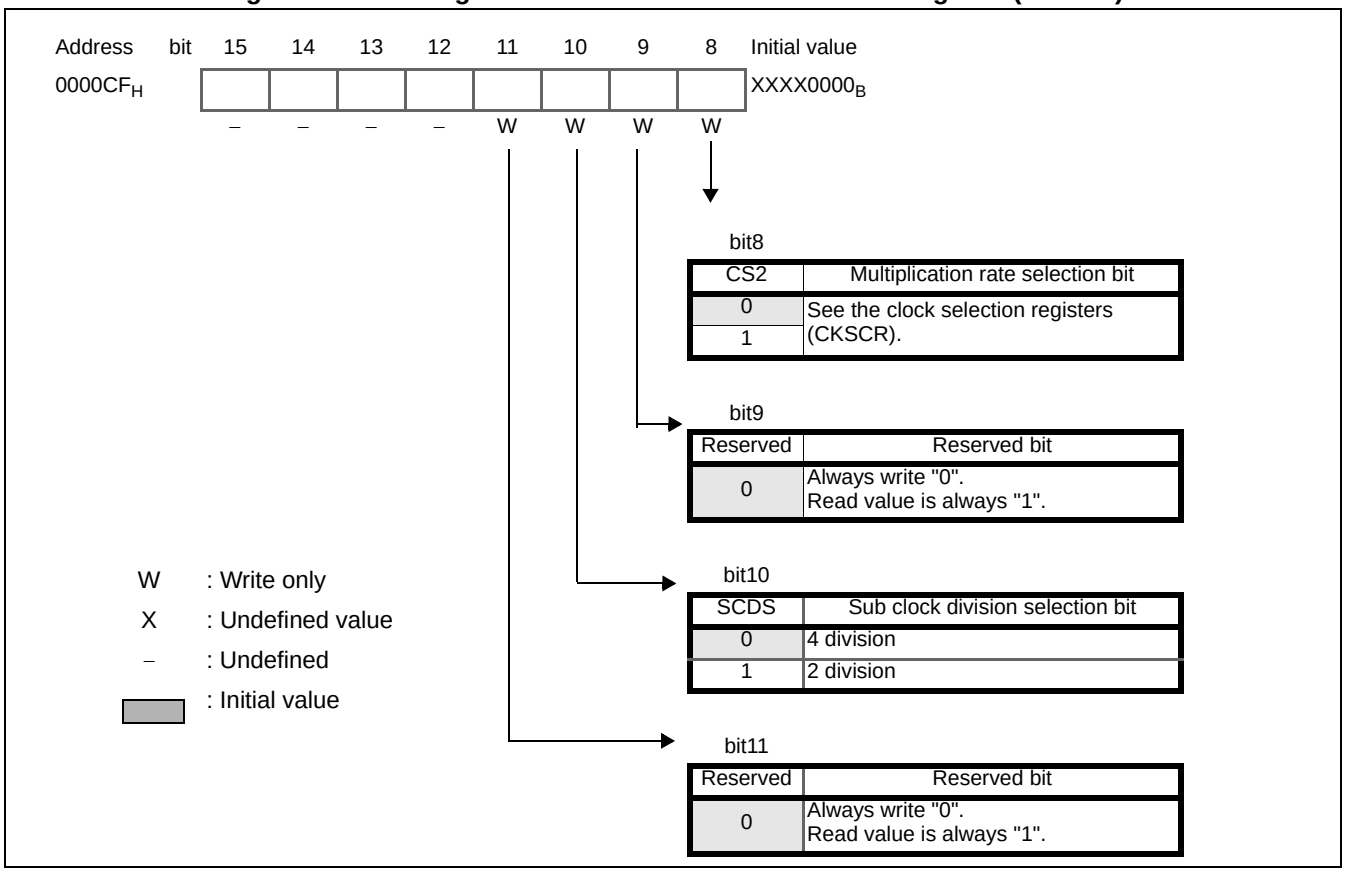


Table 5.4-1 Function Description of Each Bit in the PLL/Sub Clock Control Register (PSCCR)

<table>
<tr><th colspan="2">Bit name</th><th>Function</th></tr>
<tr><td>bit15
to
bit12</td><td>Unused bits</td><td>These bits are not used.
• Writing to these bits has no effect.
• Read value is always "1".</td></tr>
<tr><td>bit11</td><td>Reserved bit</td><td>• Always write "0".
• Read value is always "1".</td></tr>
<tr><td>bit10</td><td>SCDS:
Sub clock
division selection bit</td><td>Selects a division ratio for the sub clock.
• When "0" is written, 4 division is selected.
• When "1" is written, 2 division is selected.
• Read value is always "1".
• This bit is initialized to "0" with all reset sources.</td></tr>
<tr><td>bit9</td><td>Reserved bit</td><td>• Always write "0".
• Read value is always "1".</td></tr>
<tr><td>bit8</td><td>CS2:
Multiplication rate selection bit</td><td>• This bit, and CS1 and CS0 bits in the clock selection register (CKSCR) determine the PLL multiplication rate.
<table>
<tr><th>CS2</th><th>CS1</th><th>CS0</th><th>PLL clock multiplication rate</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>×1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>×2</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>×3</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>×4</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>×6</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>×8</td></tr>
</table>
• Read value is always "1".
• This bit is initialized to "0" with all reset sources.
Note:
When MCS or MCM bit is "0", it is prohibited to change the value of this bit. Change the value in the main clock mode.</td></tr>
</table>

Note: The PSCCR register is write-only register. The read value is different from the write value. Do not use read-modify-write (RMW) instructions (such as SETB/CLRB).

5.5 Clock Mode

There are main clock mode, PLL clock mode, and sub clock mode.

■ Clock Mode

● Main clock mode

The main clock mode uses the divided-by-two clock (oscillation clock), generated either by connecting the oscillator to the high-speed oscillation pins (X0 and X1) or by inputting an external clock, as the operating clock for the CPU and peripheral functions.

● Sub clock mode

The sub clock mode uses the divided-by-four or divided-by-two clock, generated either by connecting the oscillator to the low-speed oscillation pins (X0A and X1A) or by inputting an external clock, as the operating clock for the CPU and peripheral functions. The division ratio for the sub clock can be selected with SCDS bit in the PLL/sub clock control register (PSCCR).

● PLL clock mode

The PLL clock mode uses the clock multiplied by the PLL clock multiplier circuit (PLL oscillation circuit) as operating clock for the CPU and peripheral functions. The multiplication rate for PLL clock can be set with the clock selection register (CKSCR: CS1 and CS0) and PLL/sub clock control register (PSCCR: CS2).

■ Transition of Clock Mode

The clock mode can make the transition to the main clock mode, sub clock mode, or PLL clock mode with setting the PLL clock selection bit (CKSCR: MCS) and sub clock selection bit (CKSCR: SCS).

● Transition from main clock mode to PLL clock mode

When the PLL clock selection bit (CKSCR: MCS) is rewritten from "1" to "0", after the PLL oscillation stabilization wait time (2^{14}/HCLK) is elapsed, the transition from main clock to PLL clock mode is made.

● Transition from PLL clock mode to main clock mode

When the PLL clock selection bit (CKSCR: MCS) is rewritten from "0" to "1", with the timing the PLL clock and main clock edges match (after 1 to 12 PLL clocks), the transition from PLL clock to main clock mode is made.

● Transition from main clock mode to sub clock mode

When the sub clock selection bit (CKSCR: SCS) is rewritten from "1" to "0", with the timing of sub clock edge detected, the transition from main clock mode to sub clock mode is made.

- Transition from sub clock mode to main clock mode

When the sub clock selection bit (CKSCR: SCS) is rewritten from "0" to "1", after the main clock oscillation stabilization wait time is elapsed, the transition from sub clock mode to main clock mode is made.

- Transition from PLL clock mode to sub clock mode

When the sub clock selection bit (CKSCR: SCS) is rewritten from "1" to "0", the transition from PLL clock mode to sub clock mode is made.

- Transition from sub clock mode to PLL clock mode

When the sub clock selection bit (CKSCR: SCS) is rewritten from "0" to "1", after the main clock oscillation stabilization wait time is elapsed, the transition from sub clock mode to PLL clock mode is made.

■ Selection of PLL Clock Multiplication Rate

6 types of PLL clock multiplication rates (1 to 4, 6, and 8 multiplications) can be set with writing 000_B to 011_B, 110_B, and 111_B to the multiplication rate selection bits (CKSCR: CS1 and CS0, PSCCR: CS2).

■ Machine Clock

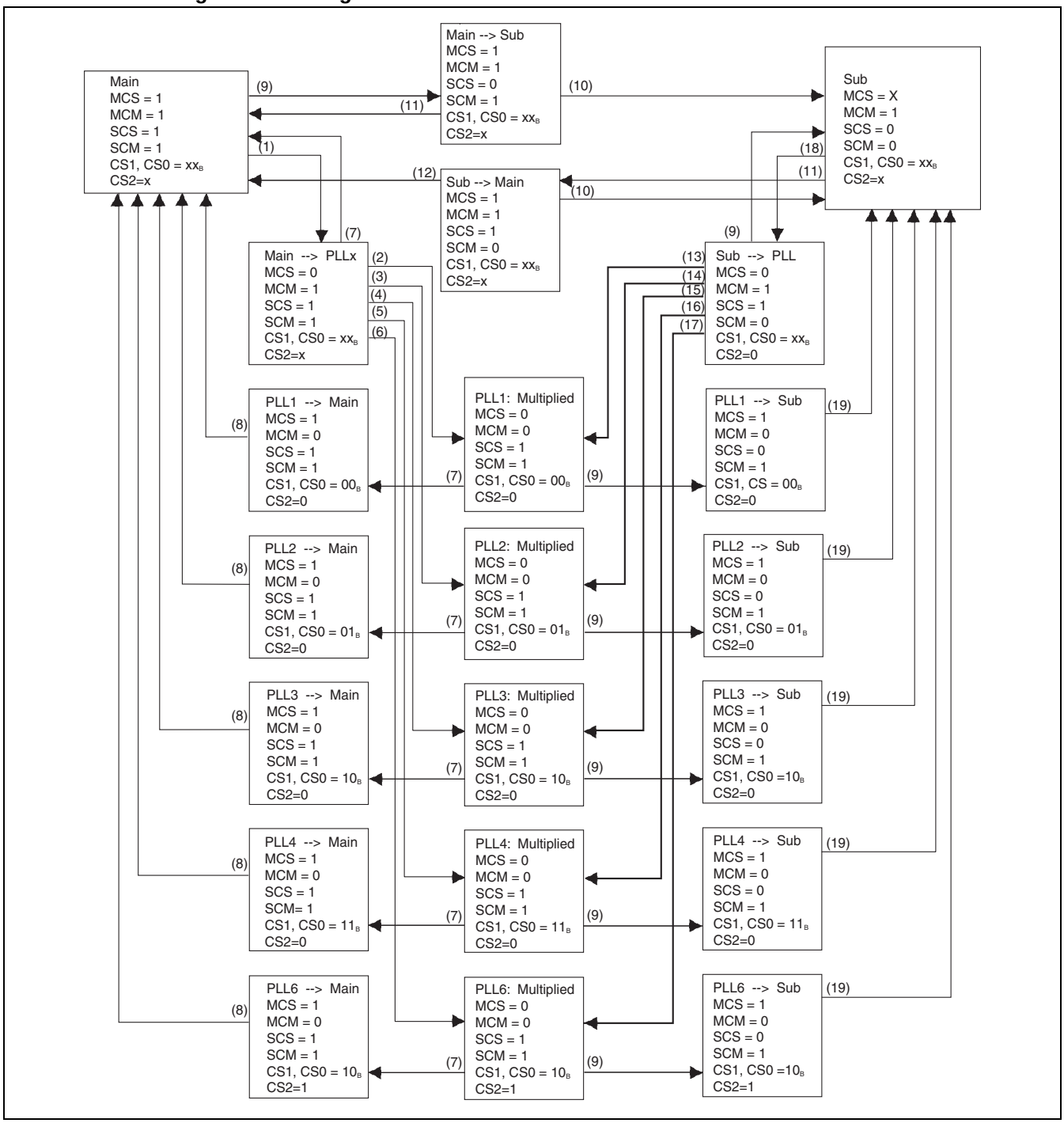
The PLL clock, main clock, or sub clock output from the PLL multiplication circuit is used as the machine clock. This machine clock is supplied to the CPU and peripheral functions. One of the main clock, PLL clock, and sub clock can be selected with writing to the sub clock selection bit (CKSCR: SCS) and PLL clock selection bit (CKSCR: MCS).

Notes:

- Even if the PLL clock selection bit (CKSCR: MCS) and sub clock selection bit (CKSCR: SCS) are rewritten, the machine clock is not switched immediately. If a machine clock-dependent peripheral function is operated, confirm that the machine clock is switched surely with referring to the PLL clock operation flag bit (CKSCR: MCM) or sub clock operation flag bit (CKSCR: SCM) before the operation.
- When the PLL clock selection bit (CKSCR: MCS) is "0" (PLL clock mode) and the sub clock selection bit (CKSCR: SCS) is "0" (sub clock mode), the SCS bit is preferred, and the transition to the sub clock mode occurs.
- When switching the clock mode, do not switch to the other clock mode or low-power consumption mode until the switching is completed. The completion of switching can be checked with referring to the MCM bit and SCM bit in the clock selection register (CKSCR). Before the switching is completed, if a switching to the other clock mode or low-power consumption mode is performed, the switching may have no effect.

Figure 5.5-1 shows a diagram of the state transition with machine clock switching.

Figure 5.5-1 Diagram for State Transition with Machine Clock Selection



(1) Write "0" to MCS bit
(2) End of PLL clock oscillation stabilization wait time & CS1, CS0 = 00_B & CS2 = 0
(3) End of PLL clock oscillation stabilization wait time & CS1, CS0 = 01_B & CS2 = 0
(4) End of PLL clock oscillation stabilization wait time & CS1, CS0 = 10_B & CS2 = 0
(5) End of PLL clock oscillation stabilization wait time & CS1, CS0 = 11_B & CS2 = 0
(6) End of PLL clock oscillation stabilization wait time & CS1, CS0 = 10_B & CS2 = 1
(7) Write "1" to MCS bit (the resets included)
(8) Synchronous timing of PLL clock and main clock
(9) Write "0" to SCS bit
(10) Synchronous timing of main clock and sub clock
(11) Write "1" to SCS bit (MCS1)
(12) End of main clock oscillation stabilization wait time
(13) End of main clock oscillation stabilization wait time & CS1, CS0 = 00_B & CS2 = 0
(14) End of main clock oscillation stabilization wait time & CS1, CS0 = 01_B & CS2 = 0
(15) End of main clock oscillation stabilization wait time & CS1, CS0 = 10_B & CS2 = 0
(16) End of main clock oscillation stabilization wait time & CS1, CS0 = 11_B & CS2 = 0
(17) End of main clock oscillation stabilization wait time & CS1, CS0 = 10_B & CS2 = 1
(18) Write "1" to SCS bit (MCS0)
(19) Synchronous timing of PLL clock and sub clock

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

Notes:

- The initial value of the machine clock is the main clock (CKSCR: MCS=1, SCS=1).
- When both SCS and MCS are "0", SCS is preferred, and the sub clock is selected.
- When switching from the sub clock mode to PLL clock mode, set CKSCR: WS1, WS0 to 10_B or 11_B.

5.6 Oscillation Stabilization Wait Time

When the power is turned on or when the stop mode is released where the oscillation clock is stopped, the oscillation clock requires some time to be stabilized (oscillation stabilization wait time) after the oscillation is started. Also, when the clock mode is switched from main to PLL, main to sub, sub to main, or sub to PLL, the oscillation stabilization wait time is required.

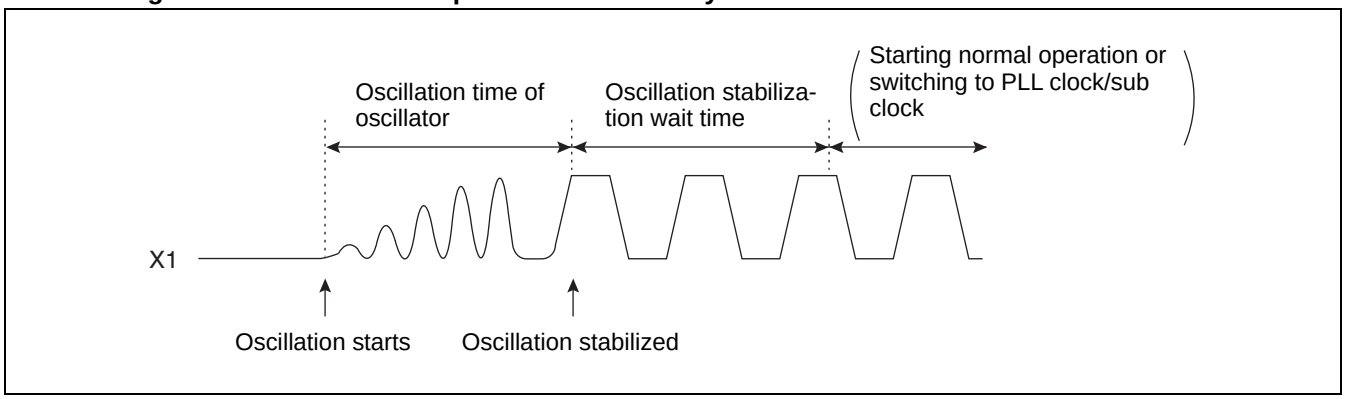
■ Operation During Oscillation Stabilization Wait Time

Ceramic and crystal oscillators require the time period of several to several dozens of milliseconds after the oscillation is started until it is stabilized at the natural frequency (oscillation frequency). Accordingly, CPU operation should be disabled immediately after the oscillation starts and the machine clock is supplied to the CPU when the oscillation is stabilized after the oscillation stabilization wait time has elapsed.

However, the oscillation stabilization wait time depends on the type of oscillator (such as ceramic and crystal). The proper oscillation stabilization wait time for the oscillator used must be selected. The oscillation stabilization wait time can be set with the clock selection register (CKSCR).

When the clock mode is switched from main to PLL, main to sub, sub to main, or sub to PLL, the CPU runs in the clock mode set before the switching during the oscillation stabilization wait time. When the oscillation stabilization wait time has elapsed, the CPU starts to run in the clock mode switched. Figure 5.6-1 shows the oscillation operation right after the oscillation starts.

Figure 5.6-1 Oscillation Operation Immediately after Oscillation Stabilization Wait Time



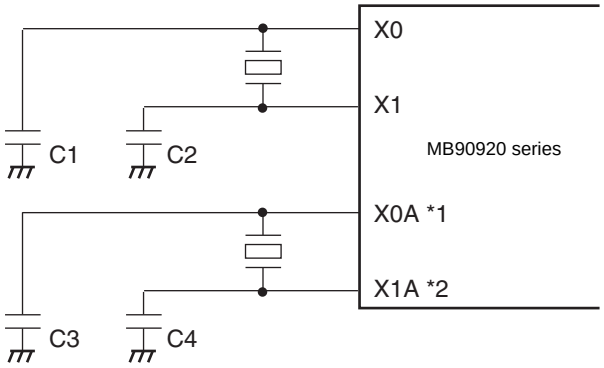
5.7 Connection of Oscillator and External Clock

The MB90920 series contains a system clock generation circuit. An internal clock is generated with connecting the oscillator to the oscillation pin. A clock externally input to the oscillation pin can be used as the oscillation clock.

■ Connection of Oscillator and External Clock

- Example of connection of crystal oscillator or ceramic resonator

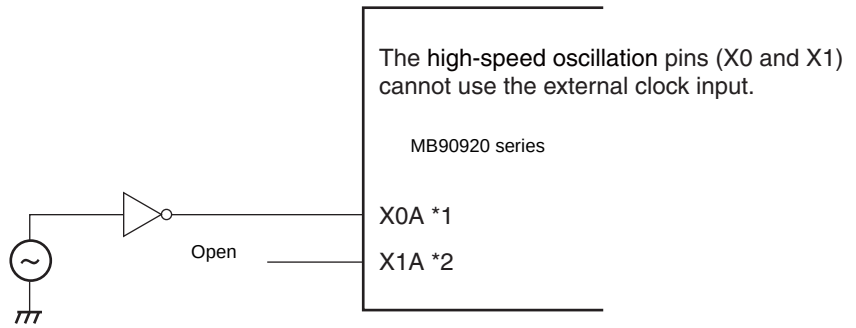
Figure 5.7-1 Example of Connection of Crystal Oscillator or Ceramic Resonator



*1 : X0A pin is optional (dual clock products only).
*2 : X1A pin is optional (dual clock products only).

- Example of connection of external clock

Figure 5.7-2 Example of Connection of External Clock



*1 : X0A pin is optional (dual clock products only).

*2 : X1A pin is optional (dual clock products only).

6. Low-Power Consumption Mode

This chapter explains the low-power consumption mode.

6.1 Overview of the Low-power Consumption Mode

6.2 Block Diagram of Low-power Consumption 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

6.7 Pin State in the Standby Mode and at the Time of Reset

6.8 Notes on Using the Low-power Consumption Mode

6.1 Overview of the Low-power Consumption Mode

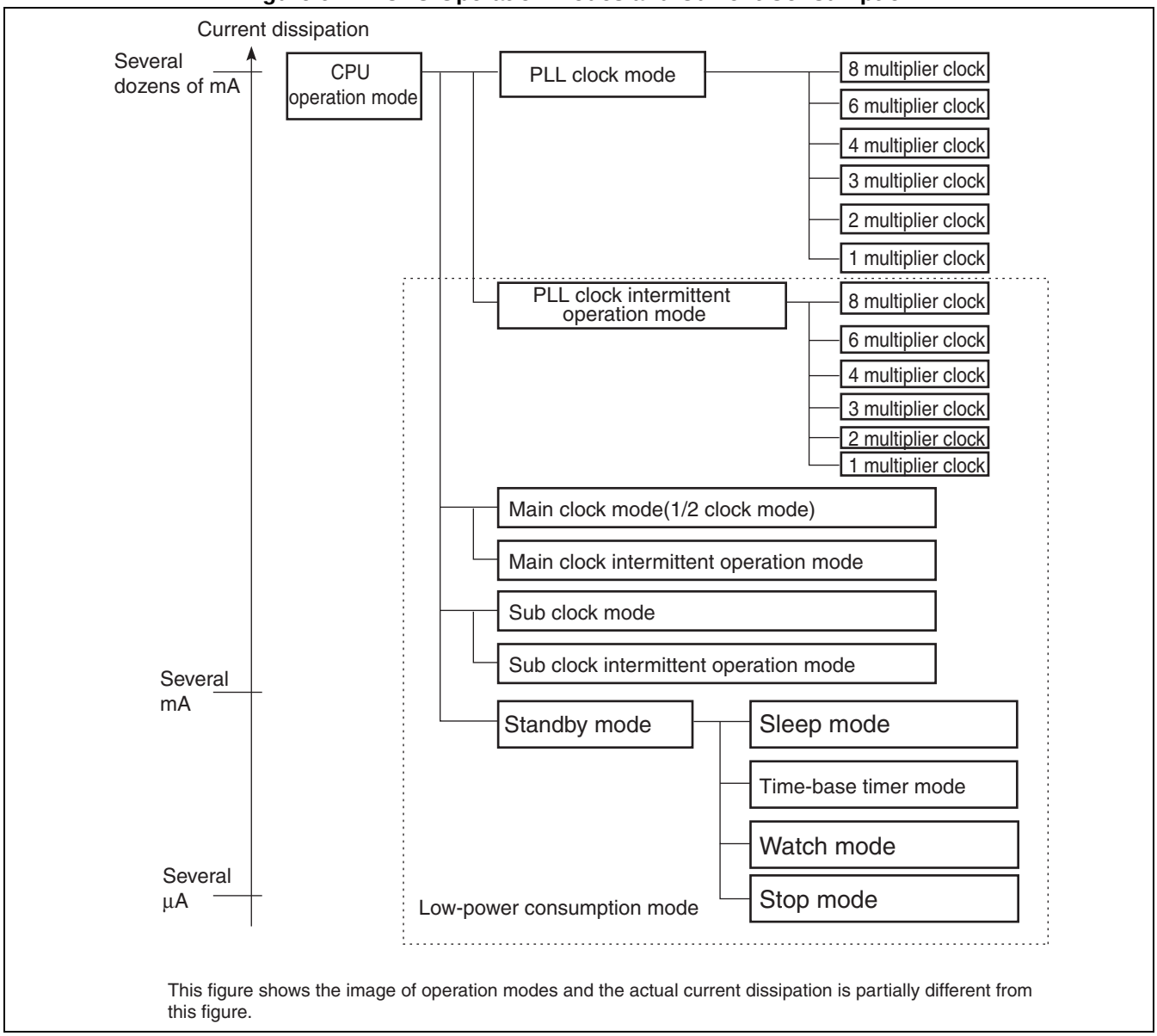
This series has the following CPU operating modes based on the operating clock selection and clock operation control.

- **Clock modes (PLL clock mode, 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 Modes and Current Consumption

Figure 6.1-1 shows the relationship between the CPU operation modes and their current consumption.

Figure 6.1-1 CPU Operation Modes and Current Consumption



■ Clock Mode

● PLL clock mode

In this mode the PLL multiplier clock of the oscillation clock (HCLK) is to operate the CPU and peripheral functions.

● Main clock mode

In this mode the divide-by-two clock of the oscillation clock (HCLK) is to operate the CPU and peripheral functions. During the main clock mode, the PLL multiplier circuit stops.

● Sub clock mode

In this mode divide-by-four clock of the sub clock (SCLK) is to operate the CPU and peripheral functions. During the sub

clock mode, the main clock and PLL multiplier circuit stop.

Reference:

See Section "5.5 Clock Mode" for the clock mode.

■ CPU Intermittent Operation Mode

In this mode, the power dissipation is reduced with operating the CPU intermittently while providing high-speed clock to peripheral functions. During the CPU intermittent operation mode, intermittent clock is input only to the CPU when a register, built-in memory, peripheral function, or external unit is accessed.

■ Standby Mode

In this mode, the power dissipation is reduced with stopping to provide the clock to the CPU (sleep mode), stopping to provide the clock to the CPU and peripheral functions (time-base timer mode) using a standby control circuit, or stopping the oscillation clock (stop mode).

- PLL sleep mode

In this mode, during the PLL clock mode, providing operation clock to the CPU is stopped. Others than the CPU are operated with PLL clock.

- Main sleep mode

In this mode, providing the operation clock to the CPU is stopped during the main clock mode. Others than the CPU are operated with the main clock.

- Sub sleep mode

In this mode, providing the operation clock to the CPU is stopped during the sub clock mode. Others than the CPU are operated with divide-by-four clock of the sub clock.

- Time-base timer mode

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

- Watch mode

In this mode, only watch timer is operated. Only the sub clock is operated, and main clock and PLL multiplier circuit are stopped.

- Stop mode

In this mode, the source oscillation is stopped, and all functions are stopped.

Note:

Since the oscillation clock is stopped in the stop mode, data can be held with the lowest power dissipation. When switching the clock mode, do not switch to other clock modes or the low-power consumption mode until the switching is completed. Check the switching is completed by referring to the MCM bit and SCM bit in the clock selection register (CKSCR). Before the switching is completed, if a switching to the other clock mode and low-power consumption mode is performed, the switching may have no effect.

6.2 Block Diagram of Low-power Consumption Circuit

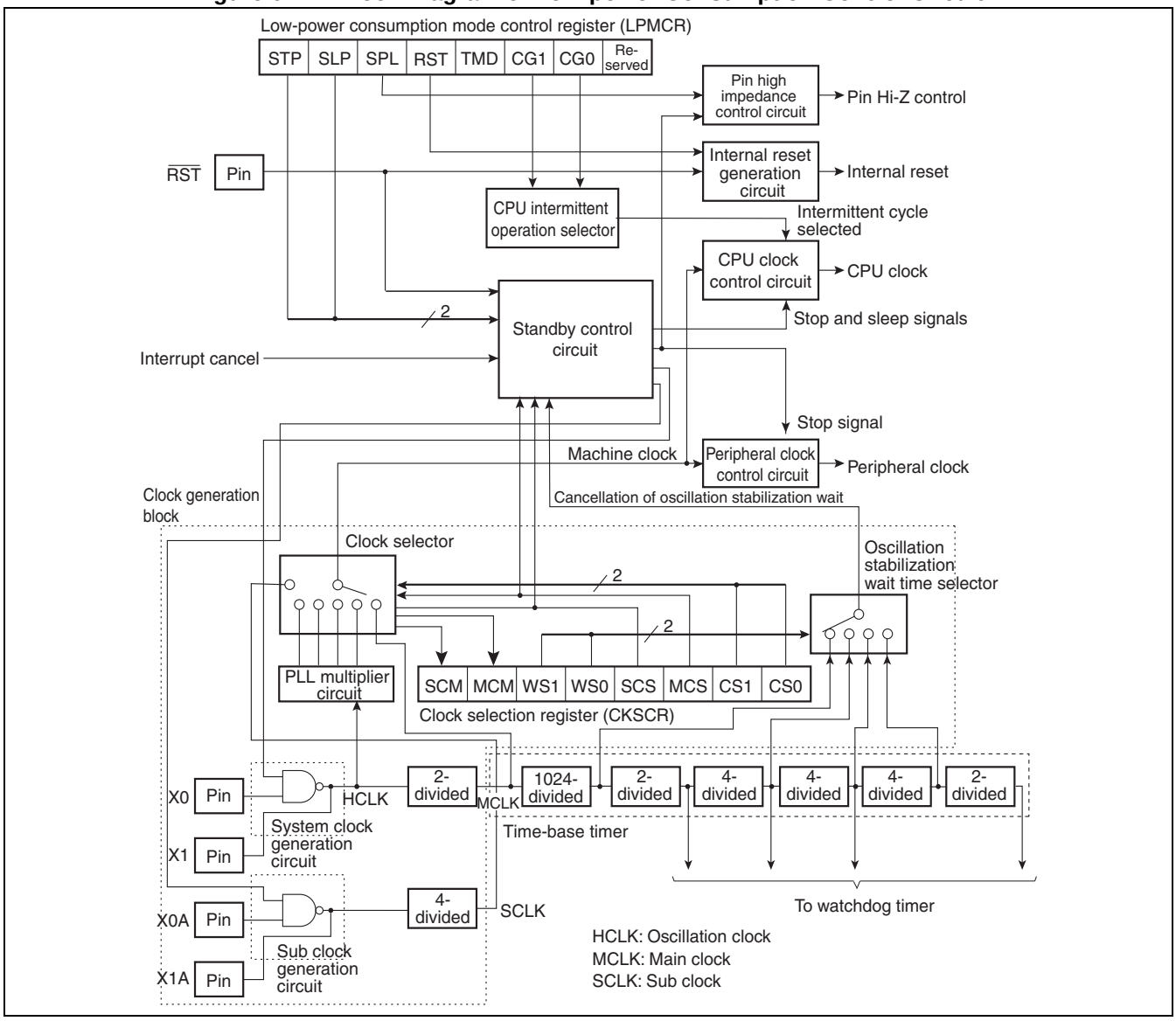
The low-power consumption control circuit consists of the following seven blocks:

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

■ Block Diagram of Low-power Consumption Control Circuit

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

Figure 6.2-1 Block Diagram of Low-power Consumption Control Circuit



- CPU intermittent operation selector

Selects the number of clocks for pauses in the CPU intermittent operation mode.

- Standby control circuit

Controls the CPU clock control circuit and peripheral clock control circuit, and makes transition to and cancellation of the low-power consumption mode.

- CPU clock control circuit

Controls clocks supplied to the CPU and peripheral clock control circuit peripheral functions.

- Peripheral clock control circuit

Controls clocks supplied to peripheral functions.

- Pin high impedance control circuit

Makes external pins high-impedance in the time-base timer mode and stop mode. The pull-up resistor is disconnected from the pin with the pull-up option selected in the stop mode.

- Internal reset generation circuit

Generates internal reset signals.

- Low-power consumption mode control register (LPMCR)

Makes the transition to and cancellation of the standby mode and configures the CPU intermittent operation function.

6.3 Low-power Consumption Mode Control Register (LPMCR)

The low-power consumption mode control register (LPMCR) makes the transition to and cancellation of the low-power consumption mode and configures the pause cycle count of the CPU clocks in the CPU intermittent operation mode.

■ Low-power Consumption Mode Control Register (LPMCR)

Figure 6.3-1 shows the configuration of the low-power consumption mode control register, and Table 6.3-1 lists the functions of each bit.

Figure 6.3-1 Configuration of Low-power Consumption Mode Control Register (LPMCR)

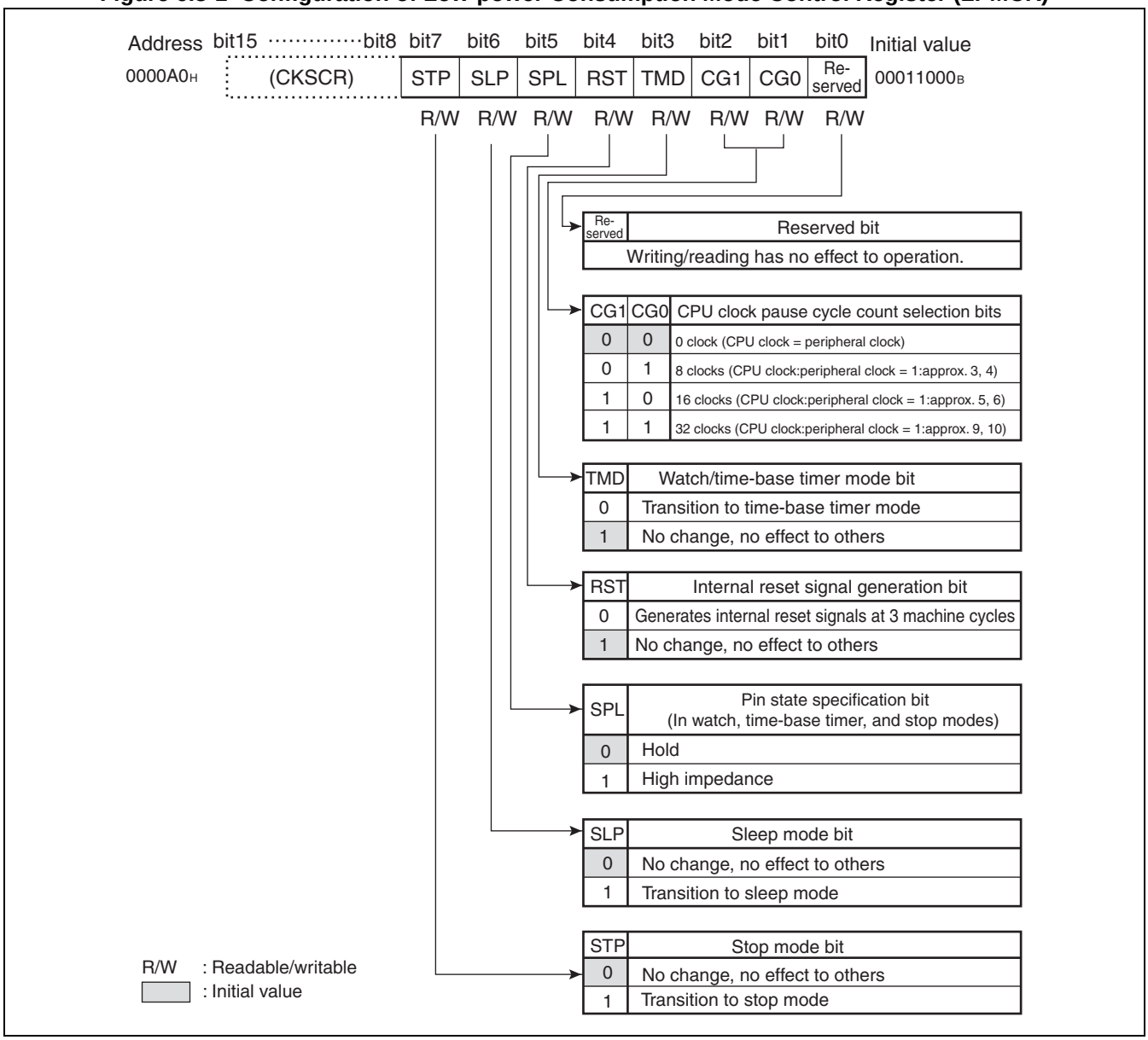


Table 6.3-1 Functions of Low-power Consumption Mode Control Register (LPMCR)

Bit name		Function
bit7	STP: Stop mode bit	• Directs the transition to the stop mode. • When this bit is set to "1", the transition to the stop mode is made. • Writing "0" to this bit has no effect. • Reset, time-base timer cancellation, or stop cancellation clears the bit to "0". • Reading this bit always returns "0".
bit6	SLP: Sleep mode bit	• Directs the transition to the sleep mode. • When this bit is set to "1", the transition to the sleep mode is made. • Writing "0" to this bit has no effect. • Reset, sleep cancellation, or stop cancellation clears the bit to "0". When the STP bit and SLP bit are set to "1" at the same time, the transition to the stop mode is made. • Reading this bit always returns "0".
bit5	SPL: Pin state specification bit (in watch, time-base timer, and stop modes)	• This bit is valid only in the time-base timer mode or stop mode. • When this bit is set to "0", the level of external pins is held. • When this bit is set to "1", the external pins are made to high impedance. • Reset initializes the bit to "0".
bit4	RST: Internal reset signal generation bit	• When this bit is set to "0", the internal reset signals are generated at 3 machine cycles. • Writing "1" to this bit has no effect. • Reading this bit always returns "1".
bit3	TMD: Watch/time-base timer mode bit	• Directs the transition to the watch mode or time-base timer mode. • In the main clock mode or PLL clock mode, when this bit is set to "0", the transition to the time-base timer mode is made. • In the sub clock mode, when this bit is set to "0", the transition to the watch mode is made. • Reset or interrupt request generation initializes the bit to "1". • Reading this bit always returns "1".
bit2, bit1	CG1, CG0: CPU clock pause cycle count selection bits	• Set the pause cycle count for CPU clock in the CPU intermittent operation function. • Specified cycle count CPU clock supply is stopped at each instruction. • One of four clock counts can be selected. • Any reset initializes the bits to 00_B.
bit0	Reserved: Reserved bit	Writing/reading has no effect.

Note:

To set a pin to high impedance when the pin is shared by a peripheral function and a port in the stop mode, watch mode, or time-base timer mode, disable the output of peripheral functions, and then set the STP bit to "1" or TMD bit to "0" in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/ PPG4,P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/

SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

■ Access to the Low-power Consumption Mode Control Register

The transition to the low-power consumption mode (stop mode, sleep mode, time-base timer mode, or watch mode) is executed with writing to the low-power consumption mode control register. The instructions listed in Table 6.3-2 should be used for the transition to the low-power consumption mode.

The following command string must be allocated immediately after using the low-power consumption mode transition instructions in Table 6.3-2.

```
    MOV LPMCR, #H'xx        ; Low-power consumption mode transition instruction in Table 6.3-2
    NOP
    NOP
    JMP $+3                 ; Jump to next instruction
    MOV A, #H'10            ; Any instruction
```

If instruction lines other than shown in are added, operations after canceling the standby mode will not be assured.

If the C language is used to access the low-power consumption mode control register, see Section "6.8 Notes on Using the Low-power Consumption Mode" and "■ Notes on Accessing the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode".

Use even addresses to write in the low-power consumption mode control register (LPMCR) in units of words. Writing to an odd address may cause malfunction.

When functions other than the functions listed in Table 6.3-1 are controlled, any instructions can be used.

● Priorities of STP, SLP, and TMD Bits

If a stop mode request, sleep mode request, and time-base timer mode request are executed at the same time, those requests are processed with the following order of priorities:

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

Table 6.3-2 List of Instructions Used for Transition to Low-power Consumption Mode

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

6.4 CPU Intermittent Operation Mode

The CPU intermittent operation mode is used to reduce power dissipation with intermittent operation of the CPU while external buses or peripheral functions are operated with high-speed.

■ CPU Intermittent Operation Mode

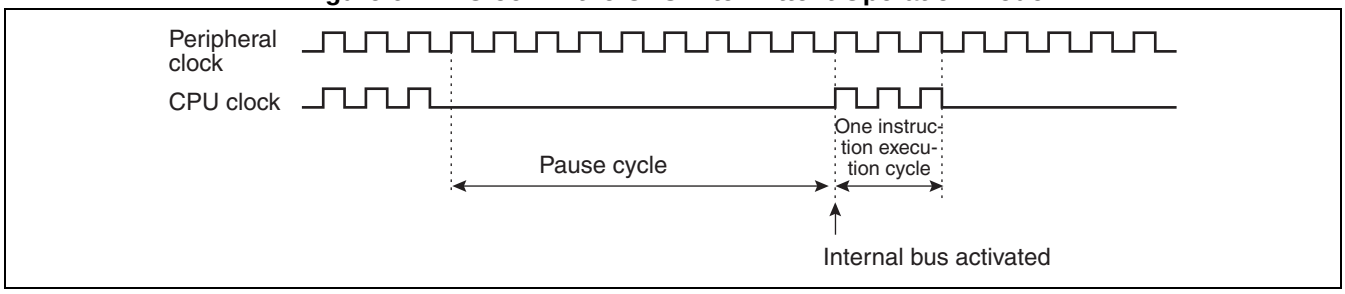
When the register, internal memory (ROM and RAM), I/O, peripheral functions, and external buses are accessed, the CPU intermittent operation mode holds the clock provided to the CPU every time an instruction is executed to delay the activation of an internal bus cycle. While high-speed peripheral clocks are supplied to peripheral functions, the execution speed of the CPU is reduced, the processing can be performed with low-power consumption.

The low-power consumption mode control register (LPMCR: CG1 and CG0) is used to select the number of cycles which halts the clock supplied to the CPU.

The operation of external buses itself uses the same clock as the one which is used in peripheral functions.

The instruction execution time in the CPU intermittent operation mode can be calculated by adding the normal execution time to the adjusted value that is the number of instruction executions for accesses to the register, internal memory, peripheral functions and external buses multiplied by the number of pause cycle count. Figure 6.4-1 shows the clock in the CPU intermittent operation mode.

Figure 6.4-1 Clock in the CPU Intermittent Operation Mode



6.5 Standby Mode

The standby mode includes the sleep mode (PLL sleep mode, main sleep mode, and sub sleep mode), watch mode, and stop mode.

■ Operation States in the Standby Mode

Table 6.5-1 shows the operation states in the standby mode.

Table 6.5-1 Operation States in the Standby Mode

Standby Mode		Transition conditions	Main clock	Sub clock	Machine clock	CPU	Peri-pheral	Pin	Release method
Sleep mode	PLL sleep mode	SCS=1 MCS=0 SLP=1	Operating	Operating	Operating	Stopped	Operating	Operating	Reset 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	Operating		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 operate.
*2 : The watch timer operates.
SPL : Pin state specification bit in the low-power consumption mode control register (LPMCR)
SLP : Sleep bit in the low-power consumption mode control register (LPMCR)
STP : Watch stop bit in the low-power consumption mode control register (LPMCR)
TMD : Watch/time-base timer mode bit in the low-power consumption mode control register (LPMCR)
MCS : Machine clock selection bit in the clock selection register (CKSCR)
SCS : Machine clock selection bit (sub) in the clock selection register (CKSCR)
Hi-Z : High impedance

Note:

To set a pin to high impedance when the pin shares a port with peripheral functions in the stop mode, watch mode, or time-base timer mode, disable the output of peripheral functions, and then set the STP bit to "1" or TMD bit to "0" in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/PPG4,P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

6.5.1 Sleep Mode

In the sleep mode, providing operation clock to a CPU is stopped and others than the CPU continue to operate. When the transition to the sleep mode is directed in the low-power consumption mode control register (LPMCR), if the PLL clock mode is set, the transition to the PLL sleep mode is made, if the main clock mode is set, the transition to the main sleep mode is made, and if the sub clock mode is set, the transition to the sub sleep mode is made.

■ Transition to Sleep Mode

When the low-power consumption mode control register is set (LPMCR: SLP=1, TMD=1, and STP=0), the transition to the sleep mode is made. When the transition to the sleep mode is made, if MCS bit is "0" and SCS bit is "1" in the clock selection register (CKSCR), the transition to the PLL sleep mode is made. If MCS bit is "1" and SCS bit is "1", the transition to the main sleep mode is made, and if SCS bit is "0", the transition to the sub sleep mode is made.

Note:

When the SLP bit and STP bit are set to "1" at the same time, the STP bit is preferred, and the transition to the stop mode is made. When the SLP bit is set to "1" and TMD bit is set to "0" at the same time, the TMD bit is preferred, and the transition to the time-base timer mode or watch mode is made.

- Data retaining function

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

- Operations while an interrupt request exists

When "1" is set to the SLP bit in the low-power consumption mode control register (LPMCR), if an interrupt request exists, the transition to the sleep mode is not made. Therefore, if the CPU does not accept the interrupt request, the CPU executes the next instruction, and if the CPU accepts the interrupt request, the CPU operation immediately branches to the interrupt processing routine.

- Pin state

During the sleep mode, pins keep the state before switching to the sleep mode.

■ Releasing Sleep Mode

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

● Return by reset

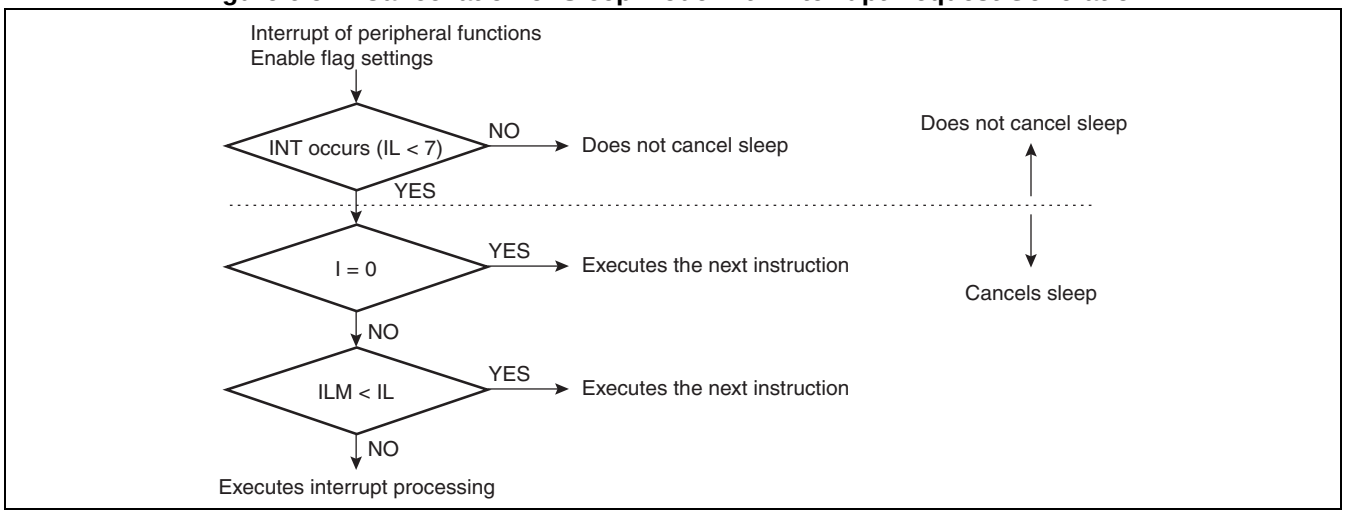
The mode is initialized to the main clock mode by reset.

● Return by interrupt

During the sleep mode, if an interrupt request with interrupt level higher than "7" is generated by a peripheral circuit or others, the sleep mode will be canceled. After the sleep mode is canceled, the interrupt request will be treated the same as the normal interrupt processing. If the interrupt request is accepted based on the setting of the I flag in the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR), the CPU will execute an interrupt processing. If the interrupt request is not accepted, the processing will be continued with the next instruction after the one which was specified to the sleep mode.

Figure 6.5-1 shows the cancellation of the sleep mode with the generation of the interrupt requests.

Figure 6.5-1 Cancellation of Sleep Mode with Interrupt Request Generation

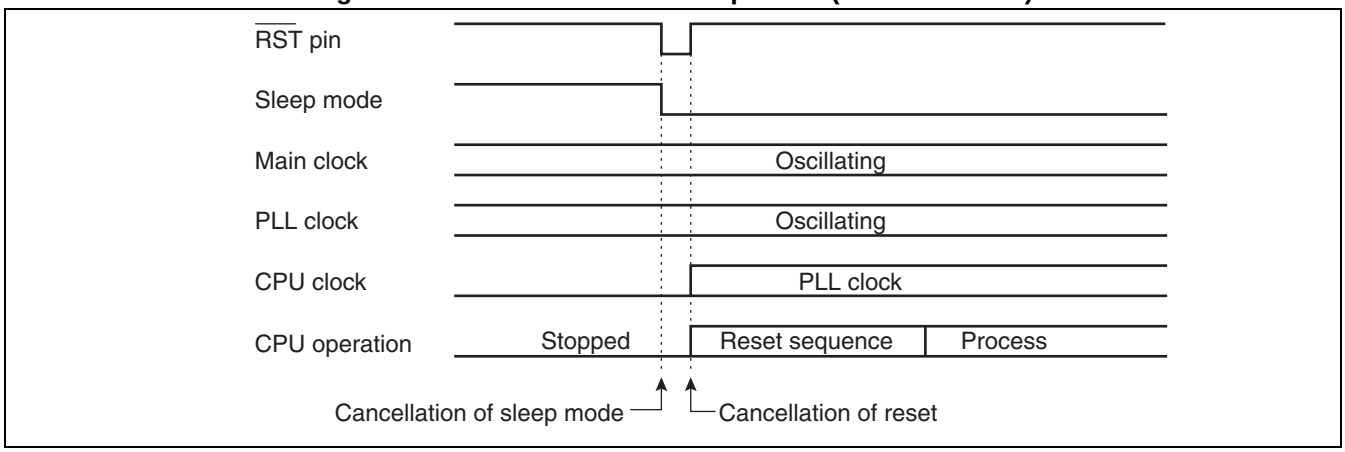


Note:

When an interrupt processing is executed, it is normal that an interrupt processing is executed after executing the next instruction coming after the one which is specified to the sleep mode.

Figure 6.5-2 shows the cancellation of the sleep mode (external reset).

Figure 6.5-2 Cancellation of Sleep Mode (External Reset)



6.5.2 Time-base Timer Mode

In the time-base timer mode, the other functions than the source oscillation, time-base timer, and watch timer are stopped. Therefore, all the functions except the time-base timer and watch timer are stopped.

■ Transition to Time-base Timer Mode

In the PLL clock mode or main clock mode (CKSCR: SCS=1), when "0" is written to the TMD bit of the low-power consumption mode control register (LPMCR), the transition to the time-base timer mode is made.

- Data retaining function

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

- Operations while an interrupt request occurs

When "0" is written to the TMD bit in the low-power consumption mode control register (LPMCR), if an interrupt request exists, the transition to the time-base timer mode is not made.

- Pin state

It can be controlled with the SPL bit in the low-power consumption mode control register (LPMCR) whether the external pins in the time-base timer mode are retained in the state they were right before the transition or go to the high-impedance state.

Note:

In the time-base timer mode, when the pin which shares a port with peripheral functions is set to high-impedance, disable the output of the peripheral functions, then set "0" the TMD bit in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/PPG4, P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

■ Releasing Time-base Timer Mode

The low-power consumption control circuit releases the time-base timer mode with reset input or interrupt generation.

- Return by reset

The mode is initialized to the main clock mode by reset.

- Return by interrupt

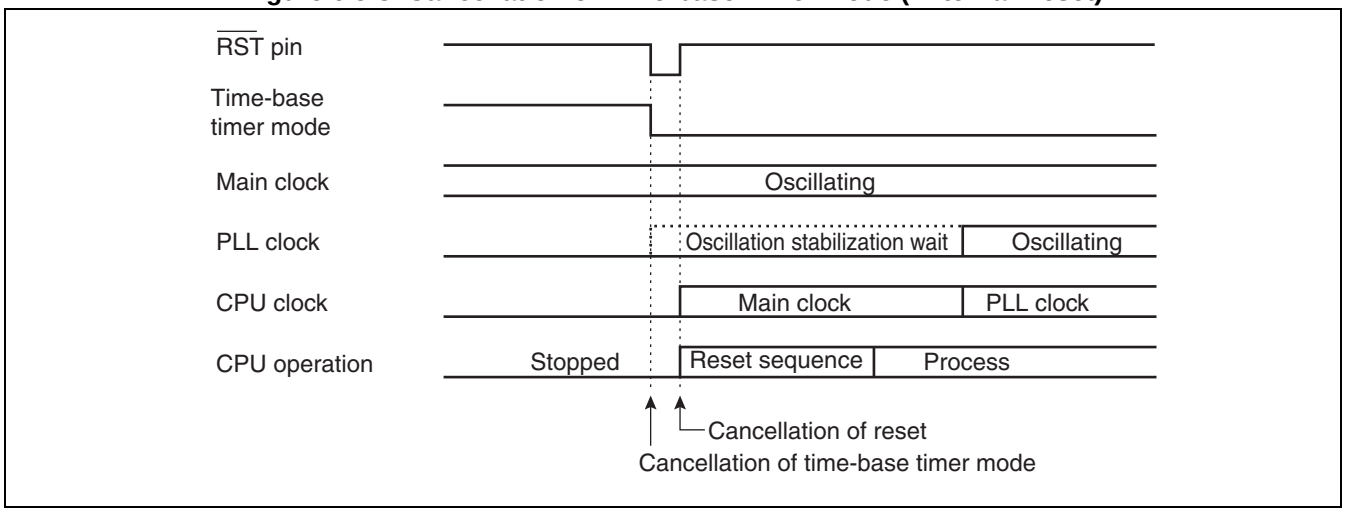
During the time-base timer mode, if an interrupt request with interrupt level higher than "7" is generated by peripheral circuits or others (the interrupt control register ICR: IL2, IL1, and IL0 are other than 111_B), the low-power consumption control circuit releases the time-base timer mode. After the time-base timer mode is canceled, the interrupt request will be treated the same as the normal interrupt processing. If an interrupt request is accepted based on the settings of the I flag in the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR), the interrupt processing is executed. If the interrupt request is not accepted, the processing will be continued with the next instruction before the transition to the time-base timer mode is made.

Note:

It is normal that an interrupt processing will be executed after executing the next instruction coming after the one which is specified to the time-base timer mode. However, if the transition to the time-base timer mode and acceptance of an external bus hold request occur simultaneously, the transition to the interrupt processing might be made before the next instruction is executed.

Figure 6.5-3 shows the operation for return from the time-base timer mode.

Figure 6.5-3 Cancellation of Time-base Timer Mode (External Reset)



6.5.3 Watch Mode

In the watch mode, operations other than the sub clock and watch timer are stopped. almost all the functions of the chip are stopped.
This mode can be used with the dual clock product.

■ Transition to Watch Mode

In the sub clock mode (CKSCR: SCS=0), when "0" is written to the TMD bit of the low-power consumption mode control register (LPMCR), the transition to the watch mode is made.

- Data retaining function

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

- Operations while an interrupt request occurs

When "0" is set to the TMD bit in the low-power consumption mode control register (LPMCR), if an interrupt request exists, the transition to the watch mode is not made.

- Pin state

It can be controlled with the SPL bit in the low-power consumption mode control register (LPMCR) whether the external pins in the watch mode are retained in the state they were right before the transition or go to the high-impedance state.

Note:

In the watch mode, when the pin which shares a port with peripheral functions is set to high-impedance, disable the output of the peripheral functions, then set "0" the TMD bit in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/PPG4,P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

■ Releasing Watch Mode

The low-power consumption control circuit releases the watch mode with reset input or interrupt generation.

● Return by reset

When the watch mode is released by a reset source, after the watch mode is released, the system enters oscillation stabilization wait reset state. The reset sequence is executed after the oscillation stabilization wait time has elapsed.

● Return by interrupt

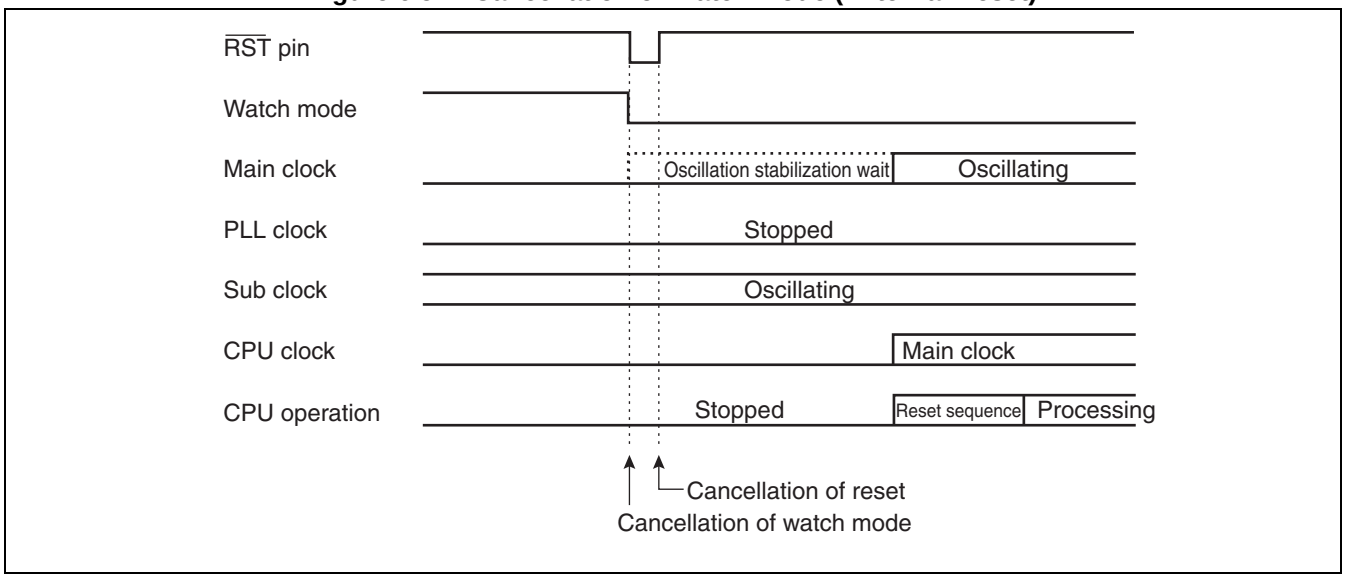
During the watch mode, if an interrupt request with interrupt level higher than "7" is generated by peripheral circuits or others (the interrupt control register ICR: IL2, IL1 and IL0 are other than 111_B), the low-power consumption control circuit releases the watch mode, and then immediately the transition to the sub clock mode is made. After the transition to the sub clock mode is made, the interrupt request will be treated the same as the normal interrupt processing. If the interrupt request is accepted with setting of the I flag in the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR), the CPU will execute an interrupt processing. If the interrupt request is not accepted, the processing will be continued with the next instruction before the transition to the watch mode is made.

Note:

When an interrupt processing is executed, it is normal that an interrupt processing is executed after executing the next instruction coming after the one which is specified to the sleep mode.

Figure 6.5-4 shows the cancellation of the watch mode (external reset).

Figure 6.5-4 Cancellation of Watch Mode (External Reset)



6.5.4 Stop Mode

In this mode, the source oscillation is stopped to stop all the functions. Therefore, data can be retained with the lowest-power dissipation.

■ Transition to Stop Mode

When the STP bit in the low-power consumption mode control register (LPMCR) is set to "1", the transition to the stop mode is made.

- Data retaining function

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

- Operations while an interrupt request occurs

When "1" is set to the STP bit in the low-power consumption mode control register (LPMCR), if an interrupt request exists, the transition to the stop mode is not made.

- Setting the pin state

It can be controlled with the SPL bit in the low-power consumption mode control register (LPMCR) whether the external pins in the stop mode are retained in the state they were right before the transition or go to the high-impedance state.

Note:

In the stop mode, when the pin which shares a port with peripheral functions is set to high-impedance, disable the output of the peripheral functions, then set "1" the STP bit in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/PPG4,P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

■ Releasing Stop Mode

The low-power consumption control circuit cancels the stop mode with a reset input or generation of an interrupt. When the CPU returns from the stop mode, the oscillation clock (HCLK) and sub clock (SCLK) are stopped. Therefore, after the main clock oscillation stabilization wait time or sub clock oscillation stabilization wait time passes,, the stop mode is released.

● Return by reset

When the stop mode is released by a reset source, after the stop mode is released, the system enters oscillation stabilization wait reset state. The reset sequence is executed after the oscillation stabilization wait time has elapsed.

● Return by interrupt

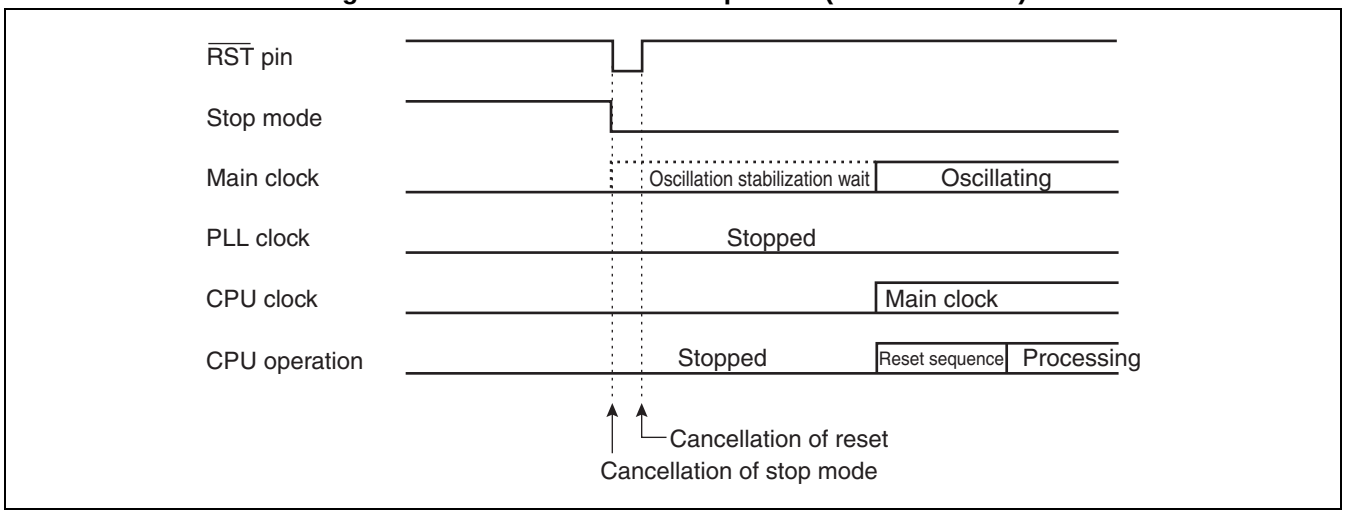
During the stop mode, if an interrupt request with interrupt level higher than "7" is generated by peripheral circuits or others (the interrupt control register ICR: IL2, IL1, and IL0 are other than 111_B), the low-power consumption control circuit releases the stop mode. When the stop mode is released, after the main clock oscillation stabilization wait time which was specified with the WS1 and WS0 bits in the clock selection register (CKSCR) passes,, the interrupt is treated the same as the normal interrupt processing. If the interrupt request is accepted with setting of the I flag in the condition code register (CCR), interrupt level mask register (ILM), and interrupt control register (ICR), the CPU will execute an interrupt processing. If the interrupt request is not accepted, the processing will be continued with the next instruction before the transition to the stop mode is made.

Notes:

- When an interrupt processing is executed, it is normal that an interrupt processing is executed after executing the next instruction coming after the one which is specified to the sleep mode. However, if the transition to the stop mode and acceptance of an external bus hold request occur simultaneously, before the next instruction is executed, the transition to the interrupt processing might be made.
- In PLL stop mode, the main clock and PLL multiplier circuit remain stopped. When the CPU returns from PLL stop mode, therefore, it is necessary to secure the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time. The oscillation stabilization wait time in this case bases upon the values set in the oscillation stabilization wait time selection bit in the clock selection register (CKSCR:WS1, WS0), and the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time will be counted together. Therefore, set a value to the "CKSCR:WS1, WS0" bit in line with the longest one of the oscillation stabilization wait times. However, because 2^{14}/HCLK or more is needed for the PLL clock oscillation stabilization wait time, set 10_B or 11_B to the "CKSCR: WS1, WS0" bit.

Figure 6.5-5 shows the cancellation of the stop mode (external reset).

Figure 6.5-5 Cancellation of Stop Mode (External Reset)

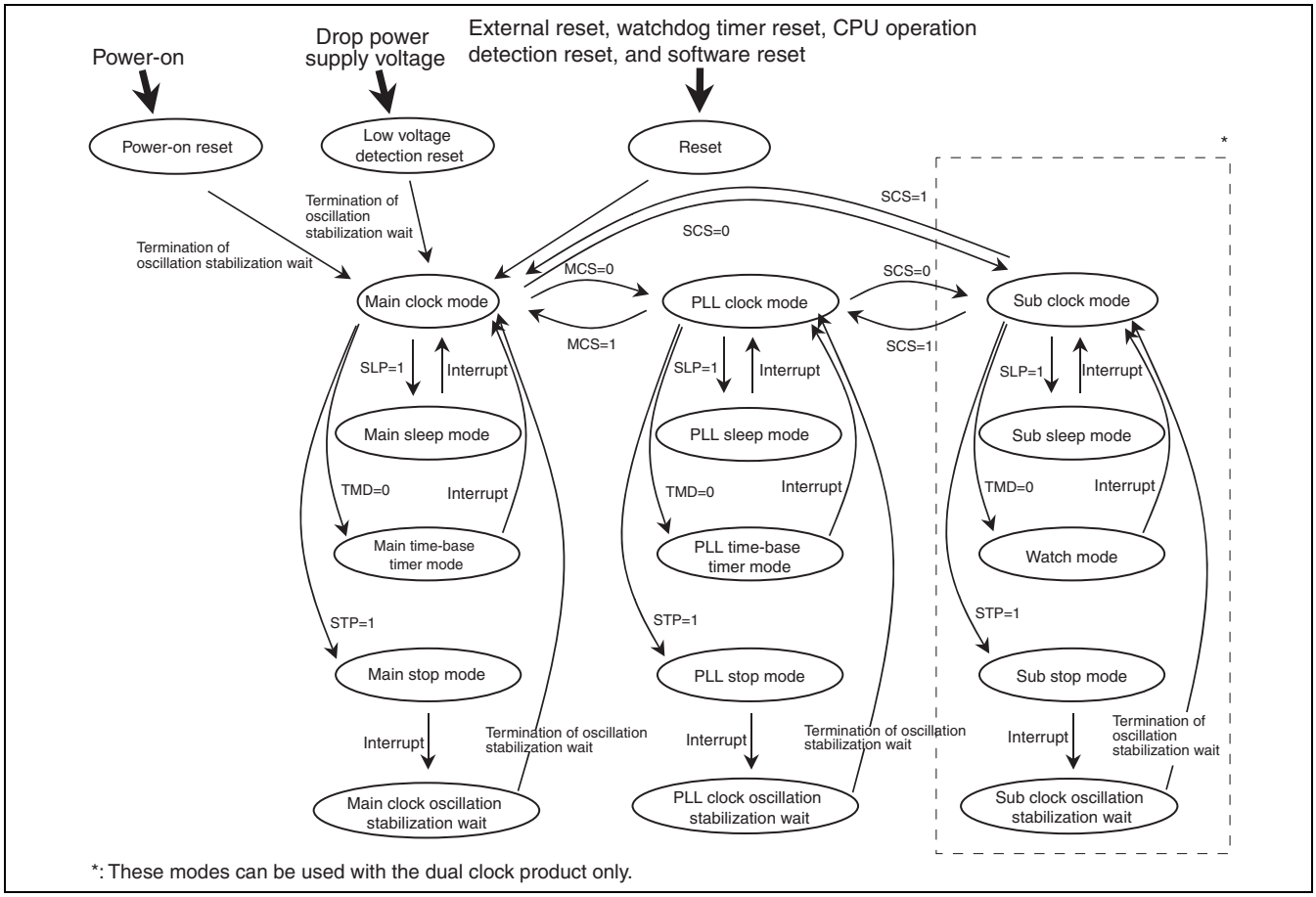


6.6 State Transition Diagram

Figure 6.6-1 shows the transition diagram and conditions of the operation status.

■ State Transition Diagram

Figure 6.6-1 State Transition Diagram



■ Operation States in the Low-power Consumption Mode

Table 6.6-1 shows the operation states in the low-power consumption mode.

Table 6.6-1 Operation States in the Low-power Consumption Mode

Operation state	Main clock	Sub clock	PLL clock	CPU	Peripheral	Watch	Time-base timer	Clock source
PLL	Operating	Operating	Operating	Operating	Operating	Operating	Operating	PLL clock
PLL sleep				Stopped				
PLL time-base timer*1					Stopped			
PLL stop	Stopped	Stopped	Stopped			Stopped	Stopped	
PLL oscillation stabilization wait	Operating	Operating	Operating			Operating	Operating	
Main	Operating	Operating	Stopped	Operating	Operating	Operating	Operating	Main clock
Main sleep				Stopped				
Main time-base timer*2					Stopped			
Main stop	Stopped	Stopped				Stopped	Stopped	
Main oscillation stabilization wait	Operating	Operating				Operating	Operating	
Sub	Stopped	Operating	Stopped	Operating	Operating	Operating	Stopped	Sub clock
Sub sleep				Stopped				
Watch					Stopped			
Sub stop		Stopped				Stopped		
Sub oscillation stabilization wait		Operating				Operating		
Power-on reset	Operating	Operating	Stopped	Stopped	Stopped	Operating	Operating	Main clock
Reset			Operating					

*1: In PLL clock mode

*2: In main clock mode

6.7 Pin State in the Standby Mode and at the Time of Reset

This section describes pin states in the standby mode and at the time of reset, for each access mode.

■ Pin States in Single Chip Mode

Table 6.7-1 shows each pin state in the single chip mode.

Table 6.7-1 Each Pin State in Single Chip Mode

Product name	Pin name	In standby mode			At reset
		In sleep mode	During stop/watch/time-base timer		
			SPL=0	SPL=1	
Evaluation chip, Mask ROM, and Flash Memory	P00 to P07 P10 to P15 P36 to P37 P40 to P47 P52/P54/P56/P57 P60 to P67	The previous state is kept.*2	The previous state is kept.*2	Input cut-off*3/Output Hi-Z	Input unavailable*4/ Output Hi-Z
	P22 to P27 P30 to P35 P70 to P77 P80 to P87				Input unavailable*4/ Output "L"
	P90 to P96 PC4 to PC7 PD0 to PD6 PE0 to PE2				Input unavailable*4/ Output Hi-Z
	P50/P51/P53/P55 PC0 to PC7		Input available*1 (External interrupt enabled)		

*1: "Input available" means that input functions can be used. Therefore, pull-up/pull-down processing or external input is needed. If the pin is used as an output port, its state is the same as other ports.

*2: "The previous state is kept" means that the output state right before the mode is kept as it is. However, note that if it was input state, input will be unavailable.
"Output state is kept as it is" means that if an internal peripheral with output is operated, its value is kept, and if the pin is outputting as a port, its value is kept.

*3: "Input cut-off" indicates the state in which an operation of an input gate near the pin is disabled, and "Output Hi-Z" means that the pin-driving transistor is prohibited from driving, and the pin is switched to high impedance state.

*4: "Input unavailable" means that the input value to the pin cannot be accepted internally because the internal circuit is stopped while the operation of input gate near the pin is allowed.

Note:

To set a pin to high impedance when the pin shares a port with peripheral functions in the stop mode, watch mode, or time-base timer mode, disable the output of peripheral functions, and then set the STP bit to "1" or TMD bit to "0" in the low-power consumption mode control register (LPMCR). Listed below are applicable pins.

Applicable pins: P00/SEG24 to P07/SEG31,P10/PPG2/IN5,P11/TOT0/PPG3/IN4,P12/TIN0/PPG4,P13/PPG5,P22/SEG00 to P27/SEG05,P30/SEG06 to P37/SEG13,P40/SEG14 to P47/SEG21,P90/SEG22 to P91/SEG23,PD1/SOT2,PD2/SCK2,PD4/SOT3,PD5/SCK3,PD6/TOT2,PE0/TOT3,P52/TX1/TX3,P54/TX0/TX2/SGA1,PE2/SGO1,P70/PWM1P0,P71/PWM1M0,P72/PWM2P0,P73/PWM2M0,P74/PWM1P1,P75/PWM1M1,P76/PWM2P1,P77/PWM2M1,P80/PWM1P2,P81/PWM1M2,P82/PWM2P2,P83/PWM2M2,P84/PWM1P3,P85/PWM1M3,P86/PWM2P3,P87/PWM2M3,P56/SGO0/FRCK,P57/SGA0,PC7/PPG1/TIN1/IN6,PC6/PPG0/TOT1/IN7,PC5/SCK1/TRG,PC4/SOT1,PC2/SCK0/INT6/IN2,PC1/SOT0/INT5/IN3

6.8 Notes on Using the Low-power Consumption Mode

Take notice of the following points when using the low-power consumption mode.

- **Transition to the standby mode and interrupts**
- **Notes on the transition to the standby mode**
- **Releasing the standby mode by an interrupt**
- **At releasing the stop mode**
- **Oscillation stabilization wait time**
- **Switching the clock mode**
- **Notes on accessing to the low-power consumption mode control register (LPMCR) for the transition to the standby mode.**

■ Transition to Standby Mode and Interrupts

Once an interrupt request is generated from a peripheral function to the CPU, the transition to any standby mode is not made because the setting to the low-power consumption mode control register, (LPMCR:STP=1, SLP=1) or (LPMCR:TMD=0), is ignored (and also even after interrupt processing, the transition to the standby mode is not made.). In this case, if the interrupt level is higher than "7", it is not a matter whether the interrupt request is accepted by the CPU.

Additionally, even when the CPU is processing an interrupt, if the interrupt request flag bit is cleared and there is no other interrupt request, the transition to the standby mode can be made.

■ Notes on Transition to Standby Mode

During the stop mode, watch mode, or time-base timer mode, if the pin which shares a port with a peripheral function is set to high-impedance, follow the procedure below.:

1) Disable the output of peripheral functions.
2) Set "1" to SPL/STP bits or "1" to TMD bit in the low-power consumption mode control register (LPMCR).

■ Releasing the Standby Mode by Interrupt

During the sleep, time-base timer, or stop mode, if the interrupt request with interrupt level higher than "7" is generated, the standby mode will be released. The releasing will be performed without regard to whether the CPU accepts the interrupt request or not.

After the standby mode is released, if the priority of the interrupt level setting bit (ICR: IL2, IL1, and IL0) for the interrupt request is higher than the interrupt level mask register (ILM) and the interrupt enable flag in the condition code register is enabled (CCR: I=1), the CPU will branch to the interrupt processing routine, as normal interrupt operation.

If the interrupt is not accepted, the operation will be restarted from the next instruction after the one which specified to the standby mode.

When an interrupt processing is executed, it is normal that an interrupt processing is executed after executing the next instruction coming after the one which is specified to the standby mode.

However, before the next instruction is executed, the interrupt processing might be performed depending on the conditions at the transition to the standby mode,

If the branch to the interrupt processing routine right after the return is prevented, measures such as disabling interrupts before setting the standby mode are necessary.

■ At Releasing the Stop Mode

Before the transition to the stop mode is made, the mode can be released with input according to the setting of the interrupt input sources of external interrupts "H" level, "L" level, and rising edge, and falling edge.

■ Oscillation Stabilization Wait Time

● Oscillation clock oscillation stabilization wait time

Since the oscillator for source oscillation is stopped during the stop mode, the oscillation stabilization wait time has to be secured. For the oscillation stabilization wait time, the time selected with the WS1 and WS0 bits in the clock selection register (CKSCR) is taken. Set 00_B to the WS1 and WS0 bits only in the main clock mode.

● PLL clock oscillation stabilization wait time

Since the PLL multiplier circuit is stopped during the main clock mode, if the transition to the PLL clock mode is made, a PLL clock oscillation stabilization wait time has to be secured. During the PLL clock oscillation stabilization wait time, the main clock operates.

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

Since the main clock and PLL multiplier circuit is stopped during the sub clock mode, if the transition to the PLL clock mode is made, the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time have to be secured. The oscillation stabilization wait time in this case bases upon the values set in the oscillation stabilization wait time selection bit in the clock selection register (CKSCR:WS1, WS0), and the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time will be counted together. Therefore, set a value to the "CKSCR:WS1, WS0" bit in line with the longest one of the oscillation stabilization wait times. However, because 2^{14}/HCLK or more is needed for the PLL clock oscillation stabilization wait time, set 10_B or 11_B to the "CKSCR: WS1, WS0" bit.

In PLL stop mode, the main clock and PLL multiplier circuit remain stopped. When the CPU returns from PLL stop mode, therefore, it is necessary to secure the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time. The oscillation stabilization wait time in this case bases upon the values set in the oscillation stabilization wait time selection bit in the clock selection register (CKSCR:WS1, WS0), and the main clock oscillation stabilization wait time and PLL clock oscillation stabilization wait time will be counted together. Therefore, set a value to the "CKSCR:WS1, WS0" bit in line with the longest one of the oscillation stabilization wait times. However, because 2^{14}/HCLK or more is needed for the PLL clock oscillation stabilization wait time, set 10_B or 11_B to the "CKSCR: WS1, WS0" bit.

■ Switching the Clock Mode

When switching the clock mode, do not switch to the other clock mode or low-power consumption mode until the switching is completed. The completion of switching can be checked by referring to the MCM bit and SCM bit in the clock selection register (CKSCR). Before the switching is completed, if a switching to the other clock mode or low-power consumption mode is performed, the switching may have no effect.

■ Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode

● When the low-power consumption mode control register (LPMCR) is accessed with the assembler language:

- When the setting for the transition to the standby mode is made in the low-power consumption mode control register (LPMCR), use the instructions in Table 6.3-2.
- The following command string must be allocated immediately after using the low-power consumption mode transition instructions in Table 6.3-2.

```
MOV LPMCR, #H'xx        ; Low-power Consumption Mode Transition Instruction in Table 6.3-2
NOP
NOP
JMP $+3                 ; Jump to next instruction
MOV A, #H'10            ; Any instruction
```

If a command string other than shown in is allocated, operations after the standby mode is canceled will not be assured.

● When the low-power consumption mode control register (LPMCR) is accessed with the C language.

When the setting for the transition to the standby mode is performed in the low-power consumption mode control register (LPMCR), access with one of the following method of (1) to (3):

(1) Making the instruction for the transition to the standby mode to a function, insert two of the built-in function of __wait_nop(). If there is a possibility that an interrupt other than the interrupt of the return to the standby is generated within the function, execute an optimization at compile time to prevent the generation of LINK/UNLINK instruction.

Example: Watch mode or time-base timer mode transition function

```
void enter_watch(){
    IO_LPMCR.byte = 0x10; /* Set "0" to the TMD bit in LPMCR */
    __wait_nop();
    __wait_nop();
}
```

(2) Write the instruction for the transition to the standby mode with __asm statement, and insert two NOP instructions and a JMP instruction after the instruction for the transition to the standby mode.

Example: The transition to the sleep mode

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

(3) Write the instruction for the transition to the standby mode between #pragma asm and #pragma endasm, and insert two NOP instructions and a JMP instruction after the instruction for the transition to the standby mode.

Example: The transition to the stop mode

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

7. Mode Setting

This chapter explains the operation mode and memory access mode.

7.1 Mode Setting

7.2 Mode Pins (MD2 to MD0)

7.3 Mode Data

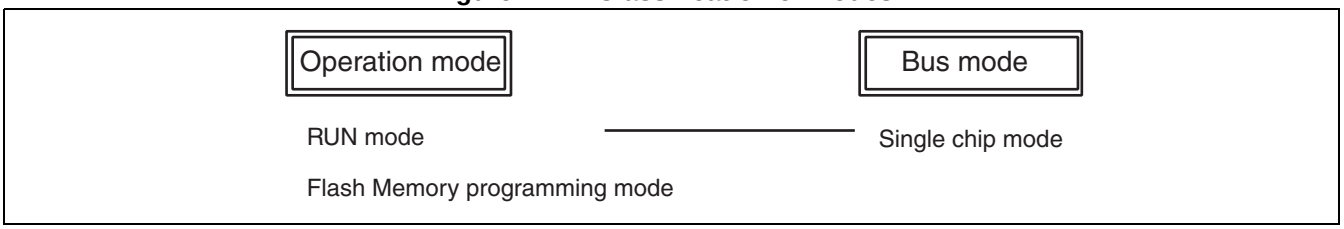
7.1 Mode Setting

F²MC-16LX has various modes for access methods and access areas.Each mode is set according to the setting of mode pins at reset and the mode-fetched mode data.

■ Mode Setting

In the F²MC-16LX, various modes are provided for access methods and access area, and in this module, those modes are classified as shown in Figure 7.1-1.

Figure 7.1-1 Classification of Modes



■ Operation Mode

The operation mode refers to a mode that controls the operation states of devices. It is specified with the contents of the mode pins (MD2 to MD0) and Mx bit in the mode data. A normal operation can be started with selecting an operation mode.

■ Bus Mode

The bus mode refers to a mode that controls the operations of internal ROM and external access function. It is specified with the contents of the mode pins (MD2 to MD0) and Mx bit in the mode data. The mode pins (MD2 to MD0) specifies the bus mode for reading the reset vector and mode data, and the Mx bit in mode data specifies the bus mode in normal operations.

■ RUN Mode

The RUN mode means the CPU operation mode. The RUN mode has the main clock mode, PLL clock mode, and various low-power consumption mode. See Section "6. Low-Power Consumption Mode" for details.

7.2 Mode Pins (MD2 to MD0)

Mode pins are three external pins of MD2 to MD0, and they specify load methods for reset vectors and mode data.

■ Mode Pins (MD2 to MD0)

The mode pins allow to select the external data bus or internal data bus for reading reset vectors and select the bus width upon the external data bus. For the internal Flash memory products, the mode pins specify the Flash memory write mode to write the internal Flash memory program and others. Table 7.2-1 shows the mode pin setting.

Table 7.2-1 Mode Pin Setting

MD2	MD1	MD0	Mode name	Reset vectors access area	External data bus width	Remarks
0	0	0	Setting prohibited			
0	0	1				
0	1	0				
0	1	1	Internal vector mode	Internal	Mode data	Operation after reset sequence is controlled with mode data.
1	0	0	Setting prohibited			
1	0	1				
1	1	0	Flash serial write mode	-	-	-
1	1	1	Flash writer write mode	-	-	-

Set MD2 to MD0: 0=Vss, 1=Vcc.

Note:

In the MB90920 series, the mode pins are used in the single chip mode only. Therefore, set the MD2 to MD0 to 011_B.

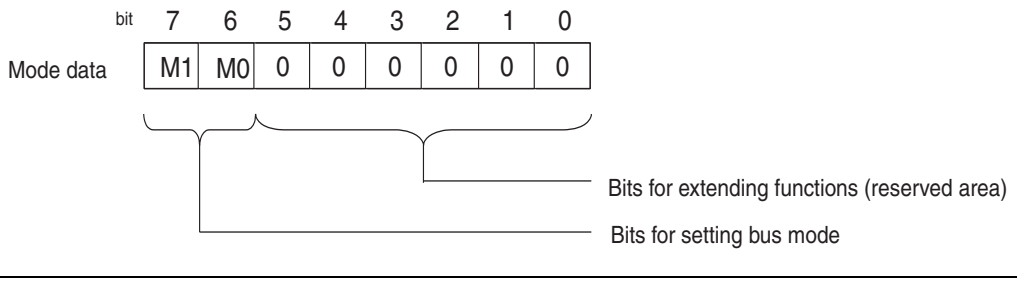
7.3 Mode Data

Mode data is stored at FFFFDF$_H$ of memory and specifies the operation after a reset sequence. The data is fetched to a CPU automatically.

■ Mode Data

While a reset sequence is executed, mode data in FFFFDF$_H$ address is fetched and stored in the mode register in the CPU core. The CPU set the memory access mode with this mode data. The mode register value can be changed by a reset sequence only. The setting of the mode data is valid after the reset sequence. Figure 7.3-1 shows the configuration of mode data.

Figure 7.3-1 Mode Data Configuration



■ Bus Mode Setting Bits

These bits specify the operation mode after the reset sequence is completed. Table 7.3-1 shows the relationship between each bit and the functions.

Table 7.3-1 Bus Mode Setting Bits and Functions

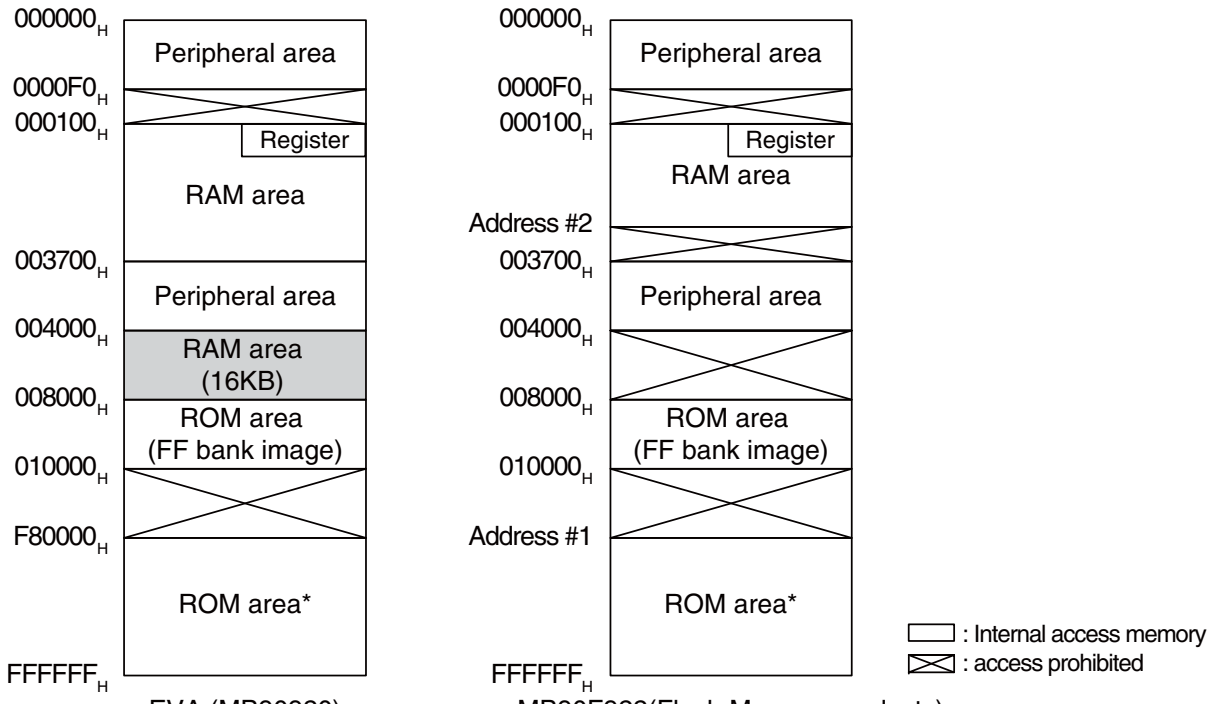
M1	M0	Function
0	0	Single chip mode
0	1	(Setting prohibited)
1	0	
1	1	

Note:

In the MB90920 series, the mode data is used in the single chip mode only. Therefore, set the M1 and MD0 to 00_B.

Figure 7.3-2 shows the correspondence between the access areas and physical addresses in the single chip mode.

Figure 7.3-2 Correspondence between Access Areas and Physical Addresses in Single Chip Mode



Product type	Address #1	Address #2
MB90922 (Mask ROM products 256KB)	$FC0000_H$	002900_H
MB90F922 (Flash Memory products 256KB)	$FC0000_H$	002900_H
MB90V920	–	003700_H

*: EVA product has no built-in ROM. This area should be as the ROM decode area of the tool.

■ Relationship between Mode Pins and Mode Data

Table 7.3-2 shows the relationship between mode pins and mode data.

Table 7.3-2 Relationship between Mode Pins and Mode Data

Mode	MD2	MD1	MD0	M1	M0
Single chip mode	0	1	1	0	0

Note:

In the MB90920 series, mode pins are used in the single chip mode only.

8. I/O Ports

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

8.1 I/O Ports
8.2 Assignment of Registers and Pins Shared with External Pins
8.3 Port 0
8.4 Port 1
8.5 Port 2
8.6 Port 3
8.7 Port 4
8.8 Port 5
8.9 Port 6
8.10 Port 7
8.11 Port 8
8.12 Port 9
8.13 Port C
8.14 Port D
8.15 Port E
8.16 Input Level Select Registers (PIL0 to PIL2)
8.17 Sample Program for I/O Ports

8.1 I/O Ports

The I/O ports can be used as general-purpose I/O ports (parallel I/O ports). The number of ports for the MB90920 series is 13 ports (93 pins). Each port is used both for peripheral functions and for providing input/output pins.

■ I/O Port Functions

The I/O ports use the port data register (PDR) to receive data from the CPU and then output it to the I/O pins or to obtain the input signals from the I/O pins and then write it to the CPU. The port also uses the port direction register (DDR) to set the I/O pin's input/output direction in units of individual bits. The functions of each port/peripheral function are listed below

- Port 0:
  Used as a general-purpose I/O port or for peripheral functions (LCD)
- Port 1:
  Used as a general-purpose I/O port or for peripheral functions (PPG/Reload timer/ICU)
- Port 2:
  Used as a general-purpose I/O port or for peripheral functions (LCD)
- Port 3:
  Used as a general-purpose I/O port or for peripheral functions (LCD)
- Port 4:
  Used as a general-purpose I/O port or for peripheral functions (LCD)
- Port 5:
  Used as a general-purpose I/O port or for peripheral functions (External interrupt input pin/CAN/Sound generator/ADC trigger input pin/Free-run timer clock input)
- Port 6:
  Used as a general-purpose I/O port or for peripheral functions (Analog input pin)
- Port 7:
  Used as a general-purpose I/O port or for peripheral functions (Stepping motor controller)
- Port 8:
  Used as a general-purpose I/O port or for peripheral functions (Stepping motor controller)
- Port 9:
  Used as a general-purpose I/O port or for peripheral functions (LCD)
- Port C:
  Used as a general-purpose I/O port or for peripheral functions (External interrupt input pin/LIN-UART/PPG)
- Port D:
  Used as a general-purpose I/O port or for peripheral functions (LIN-UART)
- Port E:
  Used as a general-purpose I/O port or for peripheral functions (Reload timer/Sound generator/ICU)

Table 8.1-1 shows the list of functions of each port.

Table 8.1-1 List of Functions of Each Port (Sheet 1 of 2)

Port name	Pin name	Input format	Output format	Function	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Port 0	P00/SEG24 to P07/SEG31	CMOS (Hysteresis) (Automotive level*)	CMOS	General-purpose I/O port	P07	P06	P05	P04	P03	P02	P01	P00
				Peripheral function	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24
					-	-	-	-	-	-	-	-
Port 1	P10/PPG2 to P15/IN0			General-purpose I/O port	-	-	P15	P14	P13	P12	P11	P10
				Peripheral function	-	-	IN0	IN1	PPG5	PPG4	PPG3	PPG2
					-	-	-	TIN2	-	TIN0	IN4	IN5
					-	-	-	-	-	-	TOT0	
Port 2	P22/SEG00 to P27/SEG05			General-purpose I/O port	P27	P26	P25	P24	P23	P22	-	-
				Peripheral function	SEG05	SEG04	SEG03	SEG02	SEG01	SEG00	-	-
Port 3	P30/SEG06 to P37/SEG13			General-purpose I/O port	P37	P36	P35	P34	P33	P32	P31	P30
				Peripheral function	SEG13	SEG12	SEG11	SEG10	SEG09	SEG08	SEG07	SEG06
Port 4	P40/SEG14 to P47/SEG21			General-purpose I/O port	P47	P46	P45	P44	P43	P42	P41	P40
				Peripheral function	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14
Port 5	P50/INT0 to P57/SGA0			General-purpose I/O port	P57	P56	P55	P54	P53	P52	P51	P50
				Peripheral function	SGA0	SGO0	INT2	TX0	INT3	TX1	INT1	INT0
					-	FRCK	RX0	TX2	-	TX3	RX1	ADTG
					-	-	RX2	SGA1	-	-	RX3	-
Port 6	P60/AN0 to P67/AN7	Analog CMOS (Hysteresis) (Automotive level*)		General-purpose I/O port	P67	P66	P65	P64	P63	P62	P61	P60
				Peripheral function	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

Table 8.1-1 List of Functions of Each Port (Sheet 2 of 2)

Port name	Pin name	Input format	Output format	Function	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Port 7	P70/PWM1P0 to P77/PWM2M1	CMOS (Hysteresis) (Automotive level*)	CMOS	General-purpose output port	P77	P76	P75	P74	P73	P72	P71	P70
				Peripheral function	PWM2M1	PWM2P1	PWM1M1	PWM1P1	PWM2M0	PWM2P0	PWM1M0	PWM1P0
Port 8	P80/PWM1P2 to P87/PWM2M3			General-purpose output port	P87	P86	P85	P84	P83	P82	P81	P80
				Peripheral function	PWM2M3	PWM2P3	PWM1M3	PWM1P3	PWM2M2	PWM2P2	PWM1M2	PWM1P2
Port 9	P90/SEG22 to P96/SEG23			General-purpose I/O port	-	P96	P95	P94	P93	P92	P91	P90
				Peripheral function	-	V2	V1	V0	(X1A)	(X0A)	SEG23	SEG22
Port C	PC0/SIN0 to PC7/PPG1			General-purpose I/O port	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
				Peripheral function	PPG1	PPG0	SCK1	SOT1	SIN1	SCK0	SOT0	SIN0
					TIN1	TOT1	TRG	-	INT7	INT6	INT5	INT4
					IN6	IN7	-	-	-	IN2	IN3	-
Port D	PD0/SIN2 to PD6/TOT2			General-purpose I/O port	-	PD6	PD5	PD4	PD3	PD2	PD1	PD0
				Peripheral function	-	TOT2	SCK3	SOT3	SIN3	SCK2	SOT2	SIN2
Port E	PE0/TOT3 to PE2/SGO1			General-purpose I/O port	-	-	-	-	-	PE2	PE1	PE0
				Peripheral function	-	-	-	-	-	SGO1	TIN3	TOT3

: Function enabled in the initial state

*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

Note:

Port 6 is also used as an analog input pin. If using the port as a general-purpose port, always set the corresponding analog input enable register (ADER6) bit to "0". At a reset, the ADER6 bit is initialized to "1".

P22 to P27 of port 2 and P30 to P35 of port 3 have segment output features SEG00 to SEG11 enabled in the initial state.

P94 to P96 of port 9 has LCD controller/driver reference power supply pins V0 to V2 enabled in the initial state.

To use these ports as general-purpose ports, set the corresponding bits in the LCD output control register (LOCR1/LOCR3) to "0".

8.2 Assignment of Registers and Pins Shared with External Pins

The registers related to I/O port setting are listed

■ List of I/O Port Registers

Table 8.2-1 shows the list of registers of each port.

Table 8.2-1 List of Registers of Each Port (Sheet 1 of 2)

Register name	Read/write	Address	Initial value
Port 0 data register (PDR0)	R/W	000000_H	XXXXXXXX_B
Port 1 data register (PDR1)	R/W	000001_H	$--XXXXXX_B$
Port 2 data register (PDR2)	R/W	000002_H	$XXXXXX--_B$
Port 3 data register (PDR3)	R/W	000003_H	XXXXXXXX_B
Port 4 data register (PDR4)	R/W	000004_H	XXXXXXXX_B
Port 5 data register (PDR5)	R/W	000005_H	XXXXXXXX_B
Port 6 data register (PDR6)	R/W	000006_H	XXXXXXXX_B
Port 7 data register (PDR7)	R/W	000007_H	XXXXXXXX_B
Port 8 data register (PDR8)	R/W	000008_H	XXXXXXXX_B
Port 9 data register (PDR9)	R/W	000009_H	$-XXXXXXX_B$
Port C data register (PDRC)	R/W	$00000C_H$	XXXXXXXX_B
Port D data register (PDRD)	R/W	$00000D_H$	$-XXXXXXX_B$
Port E data register (PDRE)	R/W	$00000E_H$	$-----XXX_B$
Port 0 direction register (DDR0)	R/W	000010_H	00000000_B
Port 1 direction register (DDR1)	R/W	000011_H	$--000000_B$
Port 2 direction register (DDR2)	R/W	000012_H	00000000_B
Port 3 direction register (DDR3)	R/W	000013_H	00000000_B
Port 4 direction register (DDR4)	R/W	000014_H	00000000_B
Port 5 direction register (DDR5)	R/W	000015_H	00000000_B
Port 6 direction register (DDR6)	R/W	000016_H	00000000_B
Port 7 direction register (DDR7)	R/W	000017_H	00000000_B
Port 8 direction register (DDR8)	R/W	000018_H	00000000_B
Port 9 direction register (DDR9)	R/W	000019_H	-0000000_B
Port C direction register (DDRC)	R/W	$00001C_H$	00000000_B
Port D direction register (DDRD)	R/W	$00001D_H$	-0000000_B

Table 8.2-1 List of Registers of Each Port (Sheet 2 of 2)

Register name	Read/write	Address	Initial value
Port E direction register (DDRE)	R/W	$00001E_H$	$-----000_B$
Analog input enabling register (ADER6)	R/W	$00001A_H$	11111111_B

R/W: Readable/Writable
X: Undefined value
-: Undefined

8.3 Port 0

Port 0 is a general-purpose I/O port that is also used for peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 0.

■ Port 0 Configuration

Port 0 consists of the following three elements:

- General-purpose I/O pins and peripheral function input/output pins (P00/SEG24 to P07/SEG31)
- Port 0 data register (PDR0)
- Port 0 direction register (DDR0)

■ Port 0 Pins

The I/O pins of Port 0 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.3-1 shows the pins of Port 0.

Table 8.3-1 Port 0 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 0	P00/SEG24	P00	General-purpose I/O	SEG24	LCDC	CMOS Hysteresis/Automotive level*	CMOS	F
	P01/SEG25	P01		SEG25				
	P02/SEG26	P02		SEG26				
	P03/SEG27	P03		SEG27				
	P04/SEG28	P04		SEG28				
	P05/SEG29	P05		SEG29				
	P06/SEG30	P06		SEG30				
	P07/SEG31	P07		SEG31				

: Function enabled in the initial state

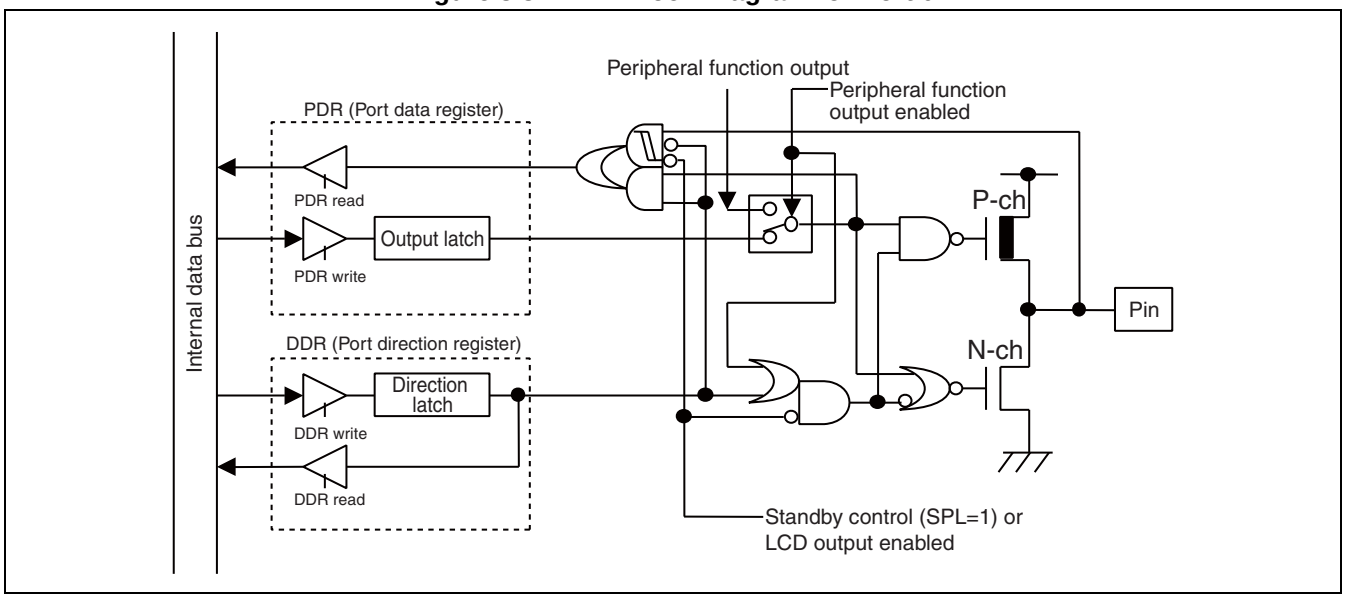
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 0

Figure 8.3-1 shows the pin block diagram for Port 0.

Figure 8.3-1 Pin Block Diagram for Port 0



Note:

If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR0 register value.

■ Registers for Port 0

The Port 0 registers are PDR0 and DDR0. The register bits have a one-to-one correspondence with the Port 0 pins. Table 8.3-2 shows the correspondence between registers and pins for Port 0.

Table 8.3-2 Correspondence between Registers and Pins for Port 0

Port name	Bit of related register and its corresponding pin								
Port 0	PDR0, DDR0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P07	P06	P05	P04	P03	P02	P01	P00

8.3.1 Port 0 Registers (PDR0, DDR0)

This section describes the registers for Port 0.

■ Functions of Port 0 Registers

- Port 0 data register (PDR0)

The PDR0 register indicates the pin states.

- Port 0 direction register (DDR0)

The DDR0 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0"

Note:

When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR0 register's setting.

Table 8.3-3 Functions of Port 0 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 0 data register (PDR0)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000000_H	XXXXXXXX_B
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 0 direction register (DDR0)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000010_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value

8.3.2 Description of Port 0 Operation

This section describes the operation of Port 0.

■ Operation of Port 0

● Operation as an output port

With the corresponding DDR0 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR0 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR0 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR0 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR0 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR0 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR0 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the value of the DDR0 register is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR0 register is not initialized. Therefore, if used as an output port, write the output data into the PDR0 register and then set the corresponding DDR0 register to "1".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR0 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.3-4 shows the pin states of Port 0.

Table 8.3-4 Pin States of Port 0

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P00/SEG24 to P07/SEG31	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High-impedance

Table 8.3-5 Priority of Pin Output of Port 0

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P00/SEG24	SEG24	P00	-	-
P01/SEG25	SEG25	P01	-	-
P02/SEG26	SEG26	P02	-	-
P03/SEG27	SEG27	P03	-	-
P04/SEG28	SEG28	P04	-	-
P05/SEG29	SEG29	P05	-	-
P06/SEG30	SEG30	P06	-	-
P07/SEG31	SEG31	P07	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.4 Port 1

Port 1 is a general-purpose I/O port that is also used for a peripheral function input port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 1.

■ Port 1 Configuration

Port 1 consists of the following three elements:

- General-purpose I/O pins and external interrupt input pins (P10/PPG2/IN5 to P15/IN0)
- Port 1 data register (PDR1)
- Port 1 direction register (DDR1)

■ Port 1 Pins

The pins of Port 1 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.4-1 shows the pins of Port 1.

Table 8.4-1 Port1 Pins

Port name	Pin name	Port function		Peripheral function				Type of input/output		Circuit type
Port 1	P10/PPG2/IN5	P10	General-purpose I/O	PPG2	PPG	IN5	ICU	CMOS Hysteresis/Automotive level*	CMOS	I
	P11/TOT0/PPG3/IN4	P11		TOT0	RLT	IN4	ICU	PPG3	PPG	
	P12/TIN0/PPG4	P12		TIN0				PPG4	PPG	
	P13/PPG5	P13		PPG5	PPG					
	P14/TIN2/IN1	P14		TIN2	RLT	IN1	ICU			
	P15/IN0	P15				IN0				

: Function enabled in the initial state

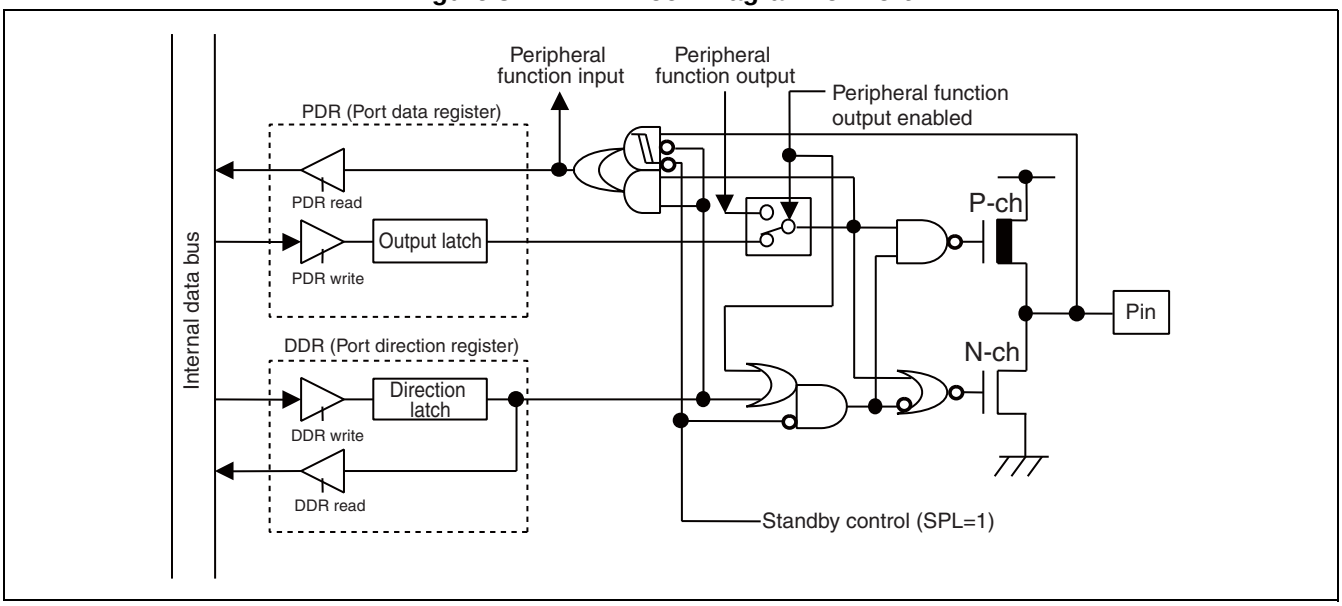
* : Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 1

Figure 8.4-1 shows the pin block diagram for Port 1.

Figure 8.4-1 Pin Block Diagram for Port 1



■ Registers for Port 1

The Port 1 registers are PDR1 and DDR1. The register bits have a one-to-one correspondence with the Port 1 pins. Table 8.4-2 shows the correspondence between registers and pins for Port 1.

Table 8.4-2 Correspondence between Registers and Pins for Port 1

Port name	Bit of related register and its corresponding pin								
Port 1	PDR1, DDR1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	-	-	P15	P14	P13	P12	P11	P10

8.4.1 Port 1 Registers (PDR1, DDR1)

This section describes the registers for Port 1.

■ Functions of Port 1 Registers

● Port 1 data register (PDR1)

The PDR1 register indicates the pin states.

● Port 1 direction register (DDR1)

The DDR1 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Notes:

- When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR1 register's setting.
- If a peripheral function with an input pin is used, set the DDR1 register bit corresponding to each peripheral function's input pin to "0" to make it work as an input port.

Table 8.4-3 Functions of Port 1 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 1 data register (PDR1)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000001_H	$--XXXXXX_B$
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 1 direction register (DDR1)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000011_H	$--000000_B$
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value
- : Undefined

8.4.2 Description of Port 1 Operation

This section describes the operation of Port 1.

■ Operation of Port 1

● Operation as an output port

With the corresponding DDR1 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR1 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR1 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR1 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR1 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR1 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR1 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Operation as a peripheral function input

For a port that is also used for peripheral function input, the pin value is always used as input to the peripheral function. To use an external signal as input for the peripheral function, set the DDR1 register for the input port to "0".

● Reset operation

At CPU reset, the DDR1 register value is cleared. Thus, all the output buffers are set to "OFF" (input port), and also as the pull-up resistor is cut, the pins are set to "high-impedance".

In a reset operation, the PDR1 register is not initialized. Therefore, if used as an output port, set the output data in the PDR1 register and then set the corresponding DDR1 register to "1".

● Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR1 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.4-4 shows the pin states of Port 1.

Table 8.4-4 Pin States of Port 1

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P10/PPG2 /IN5 to P15/IN0	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High impedance

Note:

To set a pin to high impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode, or time-base timer mode, disable the output of peripheral functions, and then set the STP bit to "1" or TMD bit to "0" in the low-power consumption mode control register (LPMCR).

Table 8.4-5 Priority of Pin Output of Port 1

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P10/PPG2/IN5	PPG2	P10	-	-
P11/TOT0/PPG3/IN4	PPG3	TOT0	P11	-
P12/TIN0/PPG4	PPG4	P12	-	-
P13/PPG5	PPG5	P13	-	-
P14/TIN2/IN1	P14	-	-	-
P15/IN0	P15	-	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.5 Port 2

Port 2 is a general-purpose I/O port that is also used for peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 2.

■ Port 2 Configuration

Port 2 consists of the following three elements:

- General-purpose I/O pins and peripheral function input/output pins (P22/SEG00 to P27/SEG05)
- Port 2 data register (PDR2)
- Port 2 direction register (DDR2)

■ Port 2 Pins

The I/O pins of Port 2 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.5-1 shows the pins of Port 2.

Table 8.5-1 Port 2 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 2	P22/SEG00	P22	General-purpose I/O	SEG00	LCDC	CMOS Hysteresis/Automotive level*	CMOS	F
	P23/SEG01	P23		SEG01				
	P24/SEG02	P24		SEG02				
	P25/SEG03	P25		SEG03				
	P26/SEG04	P26		SEG04				
	P27/SEG05	P27		SEG05				

: Function enabled in the initial state

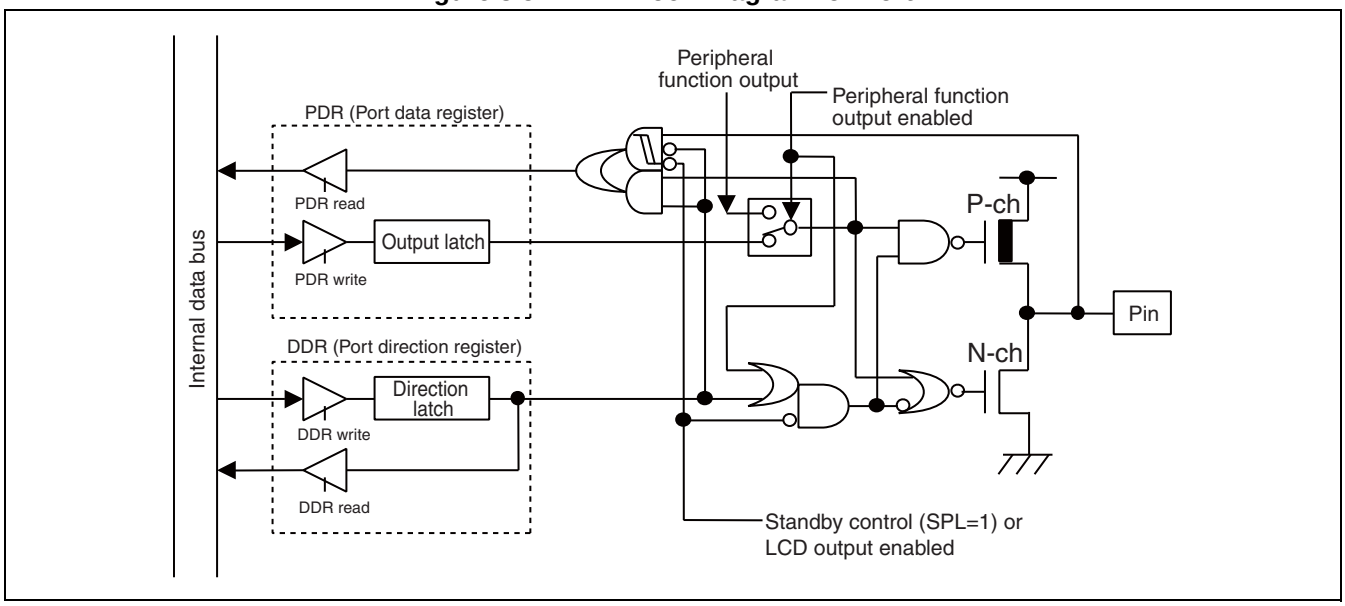
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 2

Figure 8.5-1 shows the pin block diagram for Port 2.

Figure 8.5-1 Pin Block Diagram for Port 2



Note:

If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR2 register value.

■ Registers for Port 2

The Port 2 registers are PDR2 and DDR2. The register bits have a one-to-one correspondence with the Port 2 pins. Table 8.5-2 shows the correspondence between registers and pins for Port 2.

Table 8.5-2 Correspondence between Registers and Pins for Port 2

Port name	Bit of related register and its corresponding pin								
Port 2	PDR2, DDR2	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P27	P26	P25	P24	P23	P22	-	-

8.5.1 Port 2 Data Register (PDR2, DDR2)

This section describes the registers for Port 2.

■ Functions of Port 2 Registers

- Port 2 data register (PDR2)

The PDR2 register indicates the pin states.

- Port 2 direction register (DDR2)

The DDR2 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR2 register's setting.

Table 8.5-3 Functions of Port 2 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 2 data register (PDR2)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000002_H	XXXXXX--$_B$
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 2 direction register (DDR2)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000012_H	000000--$_B$
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value
- : Undefined

8.5.2 Description of Port 2 Operation

This section describes the operation of Port 2.

■ Operations of Port 2

● Operation as an output port

With the corresponding DDR2 register bit set to "1", the port works as an output port.

When used as an output port, if data is written to the PDR2 register, it is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR2 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR2 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR2 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR2 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR2 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDR2 register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR2 register is not initialized. Therefore, if used as an output port, set the output data in the PDR2 register and then set the corresponding DDR2 register to "1".

Port 2 (P22 to P27) has segment output features SEG00 to SEG05 enabled in the initial state. To use it as a general-purpose

port, set the corresponding bit in the LCD output control register (LOCR1) to "0".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR2 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.5-4 shows the pin states of Port 2.

Table 8.5-4 Pin States of Port 2

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P22/SEG00 to P27/SEG05	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

SPL: Pin state specification bit of low-power consumption mode control register (SPL in LPMCR)
Hi-Z: High impedance

Table 8.5-5 Priority of Pin Output of Port 2

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P22/SEG00	SEG00	P22	-	-
P23/SEG01	SEG01	P23	-	-
P24/SEG02	SEG02	P24	-	-
P25/SEG03	SEG03	P25	-	-
P26/SEG04	SEG04	P26	-	-
P27/SEG05	SEG05	P27	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.6 Port 3

Port 3 is a general-purpose I/O port that is also used for peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 3.

■ Port 3 Configuration

Port 3 consists of the following three elements:

- General-purpose I/O pins and peripheral function input/output pins (P30/SEG06 to P37/SEG13)
- Port 3 data register (PDR3)
- Port 3 direction register (DDR3)

■ Port 3 Pins

The I/O pins of Port 3 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.6-1 shows the pins of Port 3.

Table 8.6-1 Port 3 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 3	P30/SEG06	P30	General-purpose I/O	SEG06	LCDC	CMOS Hysteresis/Automotive level*	CMOS	F
	P31/SEG07	P31		SEG07				
	P32/SEG08	P32		SEG08				
	P33/SEG09	P33		SEG09				
	P34/SEG10	P34		SEG10				
	P35/SEG11	P35		SEG11				
	P36/SEG12	P36		SEG12				
	P37/SEG13	P37		SEG13				

: Function enabled in the initial state

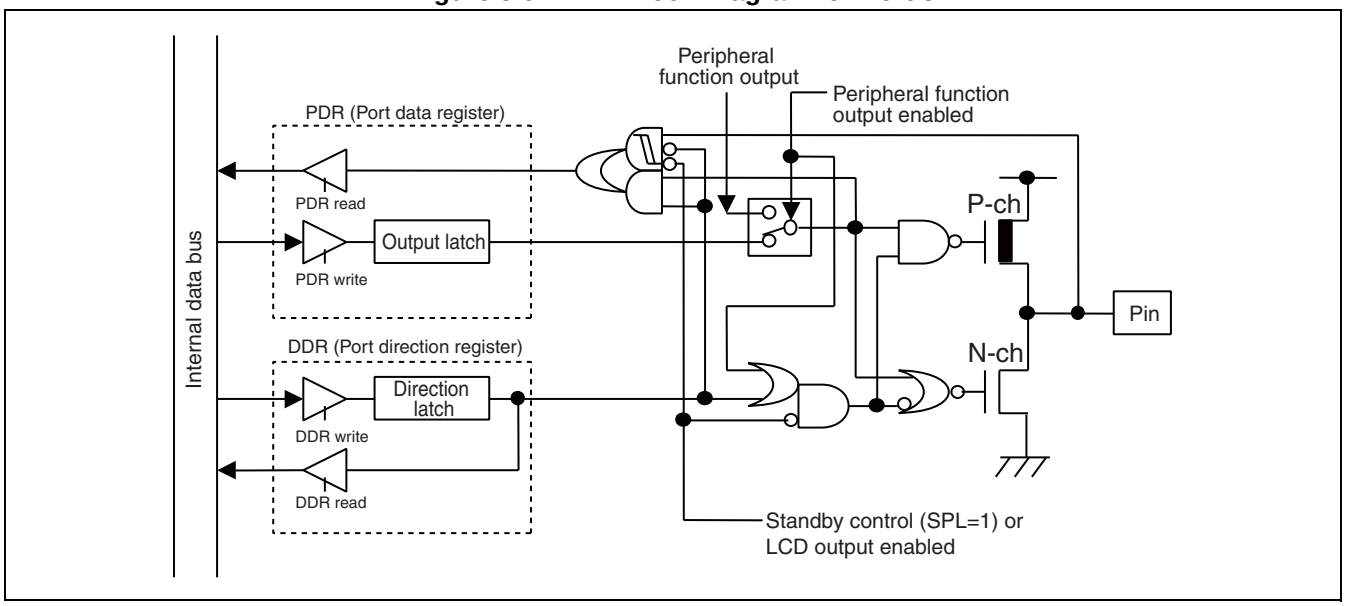
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 3

Figure 8.6-1 shows the pin block diagram for Port 3.

Figure 8.6-1 Pin Block Diagram for Port 3



Note:

If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR3 register value.

■ Registers for Port 3

The Port 3 registers are PDR3 and DDR3. The register bits have a one-to-one correspondence with the Port 3 pins. Table 8.6-2 shows the correspondence between registers and pins for Port 3.

Table 8.6-2 Correspondence between Registers and Pins for Port 3

Port name	Bit of related register and its corresponding pin								
Port 3	PDR3, DDR3	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P37	P36	P35	P34	P33	P32	P31	P30

8.6.1 Port 3 Registers (PDR3, DDR3)

This section describes the registers for Port 3.

■ Functions of Port 3 Registers

- Port 3 data register (PDR3)

The PDR3 register indicates the pin states.

- Port 3 direction register (DDR3)

The DDR3 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR3 register's setting.

Table 8.6-3 Functions of Port 3 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 3 data register (PDR3)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000003_H	XXXXXXXX_B
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 3 direction register (DDR3)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000013_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value

8.6.2 Description of Port 3 Operation

This section describes the operation of Port 3.

■ Operation of Port 3

● Operation as an output port

With the corresponding DDR3 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR3 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR3 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR3 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR3 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR3 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR3 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDR3 register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR3 register is not initialized. Therefore, if used as an output port, set the output data in the PDR3 register and then set the corresponding DDR3 register to "1".

Port 2 (P30 to P35) has segment output features SEG06 to SEG11 enabled in the initial state. To use it as a general-purpose

port, set the corresponding bit in the LCD output control register (LOCR1) to "0".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR3 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.6-4 shows the pin states of Port 3.

Table 8.6-4 Pin States of Port 3

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P30/SEG06 to P37/SEG13	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

SPL: Pin state specification bit of low-power consumption mode control register (SPL in LPMCR)
Hi-Z: High impedance

Table 8.6-5 Priority of Pin Output of Port 3

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P30/SEG06	SEG06	P30	-	-
P31/SEG07	SEG07	P31	-	-
P32/SEG08	SEG08	P32	-	-
P33/SEG09	SEG09	P33	-	-
P34/SEG10	SEG10	P34	-	-
P35/SEG11	SEG11	P35	-	-
P36/SEG12	SEG12	P36	-	-
P37/SEG13	SEG13	P37	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.7 Port 4

Port 4 is a general-purpose I/O port that is also used for a peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 4.

■ Port 4 Configuration

Port 4 consists of the following three elements:

- General-purpose I/O pins and peripheral function input/output pins (P40/SEG14 to P47/SEG21)
- Port 4 data register (PDR4)
- Port 4 direction register (DDR4)

■ Port 4 Pins

The I/O pins of Port 4 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.7-1 shows the pins of Port 4.

Table 8.7-1 Port 4 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 4	P40/SEG14	P40	General-purpose I/O	SEG14	LCDC	CMOS Hysteresis/Automotive level*	CMOS	F
	P41/SEG15	P41		SEG15				
	P42/SEG16	P42		SEG16				
	P43/SEG17	P43		SEG17				
	P44/SEG18	P44		SEG18				
	P45/SEG19	P45		SEG19				
	P46/SEG20	P46		SEG20				
	P47/SEG21	P47		SEG21				

: Function enabled in the initial state

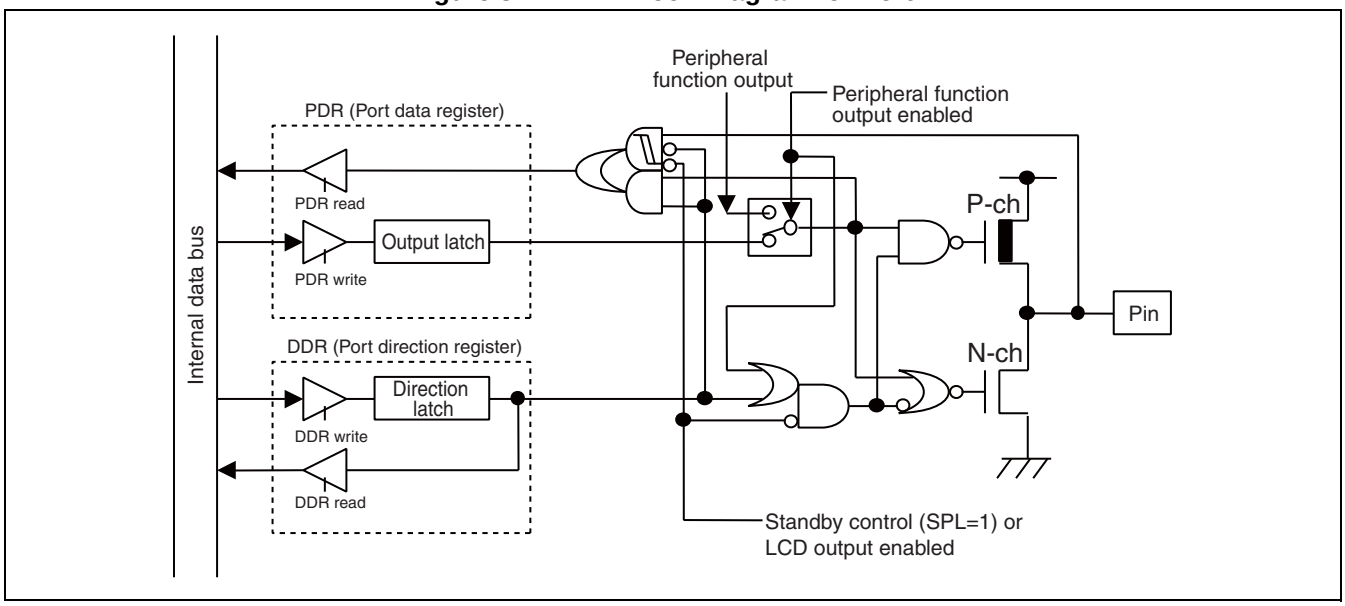
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 4

Figure 8.7-1 shows the pin block diagram for Port 4.

Figure 8.7-1 Pin Block Diagram for Port 4



Note:

If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR4 register value.

■ Registers for Port 4

The Port 4 registers are PDR4 and DDR4. The register bits have a one-to-one correspondence with the Port 4 pins. Table 8.7-2 shows the correspondence between registers and pins for Port 4.

Table 8.7-2 Correspondence between Registers and Pins for Port 4

Port name	Bit of related register and its corresponding pin								
Port 4	PDR4, DDR4	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P47	P46	P45	P44	P43	P42	P41	P40

8.7.1 Port 4 Registers (PDR4, DDR4)

This section describes the registers for Port 4.

■ Functions of Port 4 Registers

- Port 4 data register (PDR4)

The PDR4 register indicates the pin states.

- Port 4 direction register (DDR4)

The DDR4 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR4 register's setting.

Table 8.7-3 Functions of Port 4 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 4 data register (PDR4)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000004_H	XXXXXXXX_B
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 4 direction register (DDR4)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000014_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X : Undefined value

8.7.2 Description of Port 4 Operation

This section describes the operation of Port 4.

■ Operation of Port 4

● Operation as an output port

With the corresponding DDR4 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR4 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR4 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR4 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR4 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR4 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR4 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDR4 register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR4 register is not initialized. Therefore, if used as an output port, set the output data in the PDR4 register and then set the corresponding DDR4 register to "1".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR4 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.7-4 shows the pin states of Port 4.

Table 8.7-4 Pin States of Port 4

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P40/SEG14 to P47/SEG21	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High impedance

Table 8.7-5 Priority of Pin Output of Port 4

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P40/SEG14	SEG14	P40	-	-
P41/SEG15	SEG15	P41	-	-
P42/SEG16	SEG16	P42	-	-
P43/SEG17	SEG17	P43	-	-
P44/SEG18	SEG18	P44	-	-
P45/SEG19	SEG19	P45	-	-
P46/SEG20	SEG20	P46	-	-
P47/SEG21	SEG21	P47	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.8 Port 5

Port 5 is a general-purpose I/O port that is also used for a peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 5.

■ Port 5 Configuration

Port 5 consists of the following three elements:

- General-purpose I/O pins and peripheral function input pins (P50/INT0/ADTG to P57/SGA0)
- Port 5 data register (PDR5)
- Port 5 direction register (DDR5)

■ Port 5 Pins

The I/O pins of Port 5 are also used for peripheral function input pins. Therefore, if used as peripheral function input/output pins, they must not be used as general-purpose I/O ports. Table 8.8-1 shows the pins of Port 5.

Table 8.8-1 Port 5 Pins

Port name	Pin name	Port function		Peripheral function				Type of input/output		Circuit type
								Input	Output	
Port 5	P50/INT0/ ADTG	P50	General-purpose I/O	INT0	External interrupt	ADTG	A/D	CMOS Hysteresis/Automotive level*	CMOS	I
	P51/INT1/ RX1/RX3	P51		INT1		RX1 (RX3)	CAN1 CAN3			
	P52/TX1/ TX3	P52		-		TX1 (TX3)				
	P53/INT3	P53		INT3		-	-			
	P54/TX0/ TX2/SGA1	P54		TX0 (TX2)	CAN0 CAN2	SGA1	SG			
	P55/RX0/ RX2/INT2	P55		RX0 (RX2)		INT2	External interrupt			
	P56/SGO0/ FRCK	P56		SGO0	SG	FRCK	FRT			
	P57/SGA0	P57		SGA0		-	-			

: Function enabled in the initial state

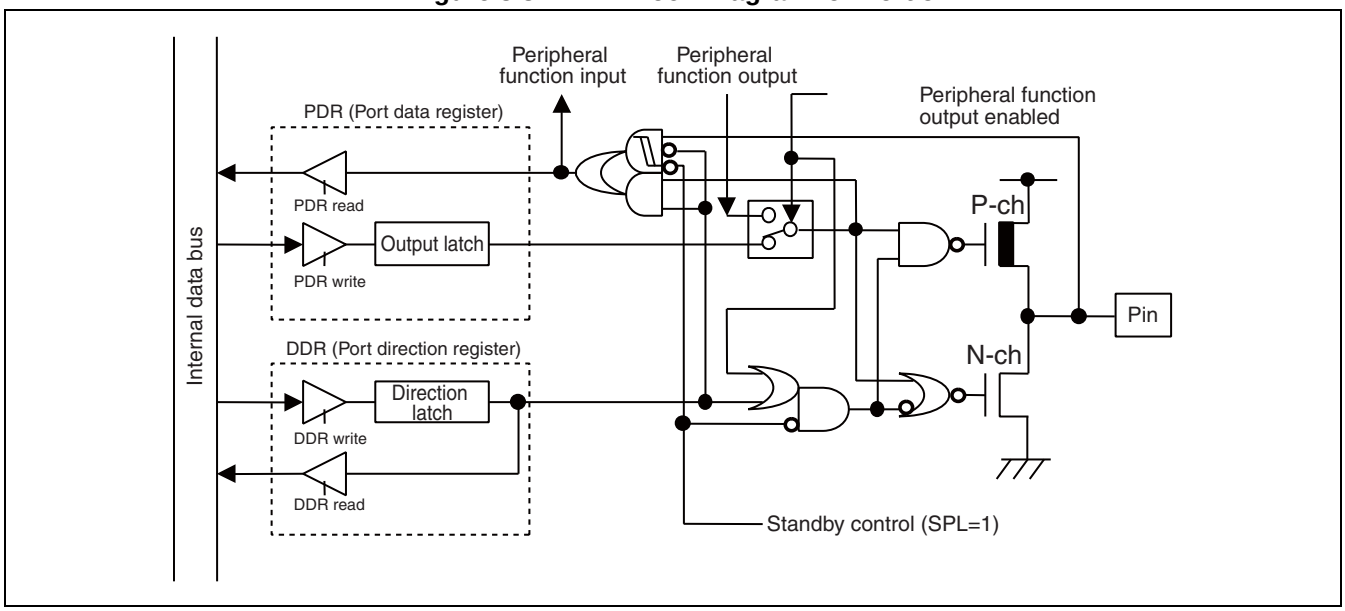
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 5

Figure 8.8-1 shows the pin block diagram for Port 5.

Figure 8.8-1 Pin Block Diagram for Port 5



■ Registers for Port 5

The Port 5 registers are PDR5 and DDR5. The register bits have a one-to-one correspondence with the Port 5 pins. Table 8.8-2 shows the correspondence between registers and pins for Port 5.

Table 8.8-2 Correspondence between Registers and Pins for Port 5

Port name	Bit of related register and its corresponding pin								
Port 5	PDR5, DDR5	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P57	P56	P55	P54	P53	P52	P51	P50

8.8.1 Port 5 Registers (PDR5, DDR5)

This section describes the registers for Port 5.

■ Functions of Port 5 Registers

- Port 5 data register (PDR5)

The PDR5 register indicates the pin states.

- Port 5 direction register (DDR5)

The DDR5 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Notes:

- When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR5 register's setting.
- If a peripheral function with an input pin is used, set the DDR5 register bit corresponding to each peripheral function's input pin to "0" to make it work as an input port.

Table 8.8-3 Functions of Port 5 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 5 data register (PDR5)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000005_H	XXXXXXXX_B
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 5 direction register (DDR5)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000015_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value

8.8.2 Description of Port 5 Operation

This section describes the operation of Port 5.

■ Operation of Port 5

● Operation as an output port

With the corresponding DDR5 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR5 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR5 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDR5 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR5 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR5 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR5 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Operation as a peripheral function input

For a port that is also used for a peripheral function input, the pin value is always set. To use an external signal as input for the peripheral function, set the DDR5 register for the input port to "0".

● Reset operation

At CPU reset, the DDR5 register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR5 register is not initialized. Therefore, if used as an output port, set the output data in the PDR5 register and then set the corresponding DDR5 register to "1".

● Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR5 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.8-4 shows the pin states of Port 5.

Table 8.8-4 Pin States of Port 5

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P50/INT0/ADTG to P57/SGA0	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z
P50/INT0, P51/INT1, P53/ INT3, P55/INT2 (External interrupt being set)	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input enabled/output Hi-Z

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

Hi-Z: High impedance

Table 8.8-5 Priority of Pin Output of Port 5

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P50/INT0/ADTG	P50	-	-	-
P51/INT1/RX1/RX3	P51	-	-	-
P52/TX1/TX3	TX1/TX3	P52	-	-
P53/INT3	P53	-	-	-
P54/TX0/TX2/SGA1	TX0/TX2	SGA1	P54	-
P55/RX0/RX2/INT2	P55	-	-	-
P56/SGO0/FRCK	SGO0	P56	-	-
P57/SGA0	SGA0	P57	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.9 Port 6

Port 6 is a general-purpose I/O port that is also used as an analog input of A/D converter. The use of each pin can be switched per bit between the analog input and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 6.

■ Port 6 Configuration

Port 6 consists of the following four elements:

- General-purpose I/O pins and analog input pins (P60/AN0 to P67/AN7)
- Port 6 data register (PDR6)
- Port 6 direction register (DDR6)
- Analog input enabling register (ADER6)

■ Port 6 Pins

The Port 6 input/output pins are also used as analog input pins. If a pin is used for an analog input, it cannot be used as a general-purpose I/O port. In addition, if a pin is used for a general-purpose port, do not use it as an analog input pin. Table 8.9-1 shows the Port 6 pins.

Table 8.9-1 Port 6 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 6	P60/AN0	P60	General-purpose I/O	AN0	Analog input 0	CMOS Hysteresis/Automotive level*	CMOS	H
	P61/AN1	P61		AN1	Analog input 1			
	P62/AN2	P62		AN2	Analog input 2			
	P63/AN3	P63		AN3	Analog input 3			
	P64/AN4	P64		AN4	Analog input 4			
	P65/AN5	P65		AN5	Analog input 5			
	P66/AN6	P66		AN6	Analog input 6			
	P67/AN7	P67		AN7	Analog input 7			

: Function enabled in the initial state

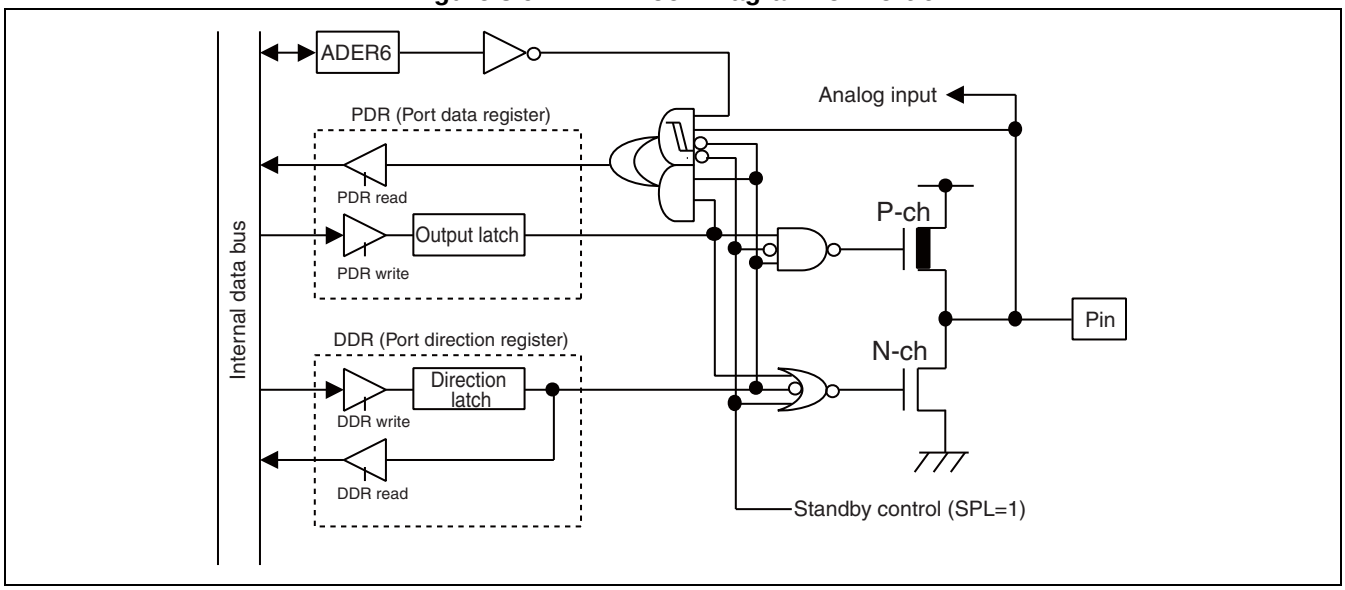
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 6

Figure 8.9-1 shows the pin block diagram of Port 6.

Figure 8.9-1 Pin Block Diagram for Port 6



Note:

For pins that are to be used as input ports, set the corresponding DDR6 register bit to "0", and the corresponding ADER6 register bit to "0".
For pins to be used as analog input pins, set the corresponding DDR6 register bit to "0" and the corresponding ADER6 register bit to "1". In this case, reading the PDR6 register returns "0".

■ Registers for Port 6

The Port 6 registers are PDR6, DDR6, and ADER6. The register bits have a one-to-one correspondence with the Port 6 pins. Table 8.9-2 shows the correspondence between registers and pins for Port 6.

Table 8.9-2 Correspondence between Registers and Pins for Port 6

Port name	Bit of related register and its corresponding pin								
Port 6	PDR6, DDR6, ADER6	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P67	P66	P65	P64	P63	P62	P61	P60

8.9.1 Port 6 Registers (PDR6, DDR6, ADER6)

This section describes the registers for Port 6.

■ Functions of Port 6 Registers

- Port 6 data register (PDR6)

The PDR6 register indicates the pin states.

- Port 6 direction register (DDR6)

The DDR6 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

- ADER6 register (ADER6)

The ADER6 register specifies whether a pin is to be used as a port or as an analog input in units of individual bits. The pin works as an analog input if the bit corresponding to the pin is set to "1" or as an input/output port if the bit is set to "0".

Notes:

- When used as port input/output, if a medium level signal is input, an input leak current flows. Therefore, pins used for analog input must have their corresponding ADER6 bits set to "1".
- At a reset, the DDR6 register is cleared and the ADER6 register is set to become an analog input.

Table 8.9-3 Functions of Port 6 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 6 data register (PDR6)	0	Pin state is "L" level	With DDR6=0, the state is high impedance. With DDR6=1, "L" level is output.	R/W	000006_H	XXXXXXXX_B
	1	Pin state is "H" level	With DDR6=0, the state is high impedance. With DDR=1, "H" level is output.			
Port 6 direction register (DDR6)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000016_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			
Analog input enable register (ADER6)	0	Port input/output mode		R/W	$00001A_H$	11111111_B
	1	Analog input mode				

R/W: Readable/Writable
X: Undefined value

8.9.2 Description of Port 6 Operation

This section describes the operation of Port 6.

■ Operation of Port 6

● Operation as an output port

With the corresponding DDR6 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR6 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR6 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then write "1" to the DDR register.

● Operation as an input port

With the corresponding DDR6 register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR6 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR6 register.

● Operation as an analog input

If the port is to be used for analog input, set the ADER6 register bit corresponding to the analog input pin to "1". Then, general-purpose port operation is prohibited and the pin will work as analog input pin. In this state, "0" is read from the PDR6.

● Reset operation

At CPU reset, the DDR6 register value is cleared and ADER6 register value is set to operate the port in analog input mode. For use as a general-purpose port, set the ADER6 register to "0" in advance to operate the port in input/output mode.

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.
Table 8.9-4 shows the pin states of Port 6.

Table 8.9-4 Pin States of Port 6

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P60/AN0 to P67/AN7	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Table 8.9-5 Priority of Pin Output of Port 6

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P60/AN0	P60	-	-	-
P61/AN1	P61	-	-	-
P62/AN2	P62	-	-	-
P63/AN3	P63	-	-	-
P64/AN4	P64	-	-	-
P65/AN5	P65	-	-	-
P66/AN6	P66	-	-	-
P67/AN7	P67	-	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.10 Port 7

Port 7 is a general-purpose output-only port that is also used for a peripheral function output port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the functions of this port as a general-purpose output-only port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 7.

■ Port 7 Configuration

Port 7 consists of the following three elements:

- General-purpose output pins and peripheral function output pins (P70/PWM1P0 to P77/PWM2M1)
- Port 7 data register (PDR7)
- Port 7 direction register (DDR7)

■ Port 7 Pins

The pins of Port 7 are also used for peripheral function output pins. If used as peripheral function output pins, they must not be used as general purpose I/O ports. Table 8.10-1 shows the pins of Port 7.

Table 8.10-1 Port 7 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 7	P70/ PWM1P0	P70	General-purpose output	PWM1P0	Stepping motor controller	-	-	L
	P71/ PWM1M0	P71		PWM1M0				
	P72/ PWM2P0	P72		PWM2P0				
	P73/ PWM2M0	P73		PWM2M0				
	P74/ PWM1P1	P74		PWM1P1				
	P75/ PWM1M1	P75		PWM1M1				
	P76/ PWM2P1	P76		PWM2P1				
	P77/ PWM2M1	P77		PWM2M1				

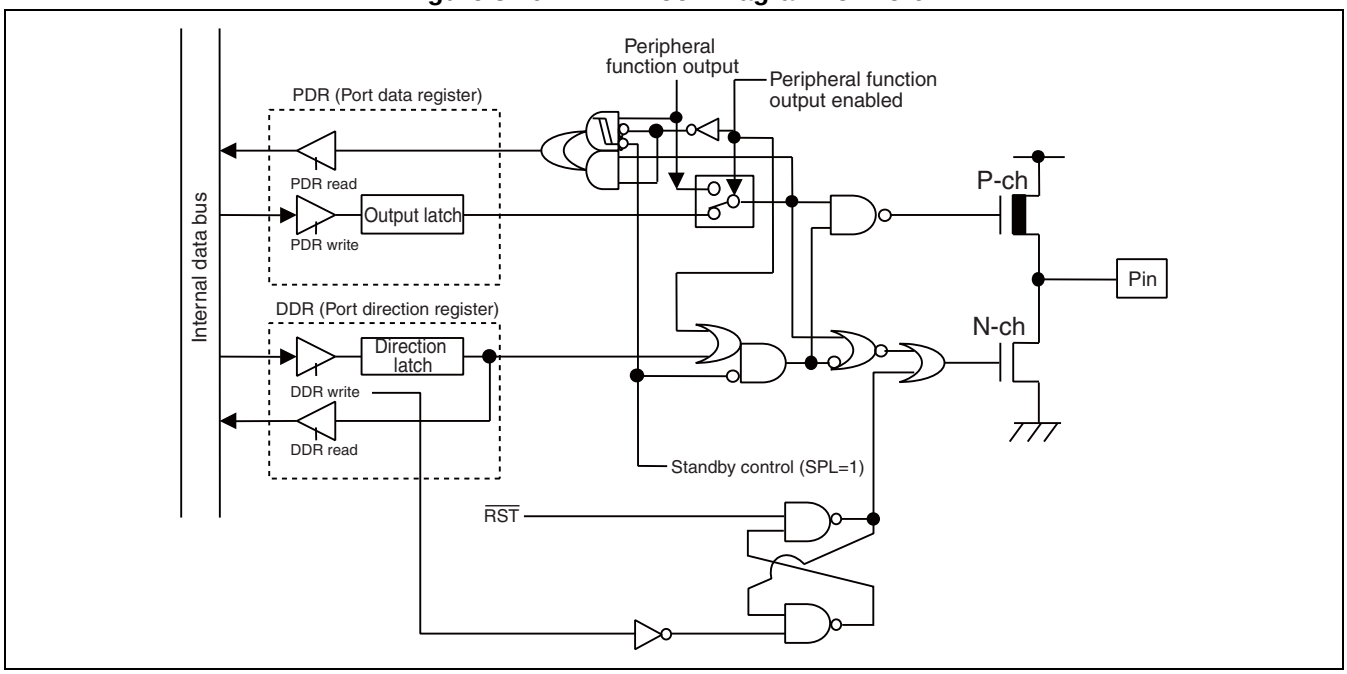
: Function enabled in the initial state

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 7

Figure 8.10-1 shows the pin block diagram for Port 7.

Figure 8.10-1 Pin Block Diagram for Port 7



If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR7 register value.

■ Registers for Port 7

The Port 7 registers are PDR7 and DDR7. The register bits have a one-to-one correspondence with the Port 7 pins. Table 8.10-2 shows the correspondence between registers and pins for Port 7.

Table 8.10-2 Correspondence between Registers and Pins for Port 7

Port name	Bit of related register and its corresponding pin								
Port 7	PDR7, DDR7	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P77	P76	P75	P74	P73	P72	P71	P70

8.10.1 Port 7 Registers (PDR7, DDR7)

This section describes the registers for Port 7.

■ Functions of Port 7 Registers

- Port 7 data register (PDR7)

The PDR7 register indicates the pin states.

- Port 7 direction register (DDR7)

The DDR7 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or becomes the high impedance state if the bit is set to "0".

Note:

When the output enable bit of the peripheral function (stepping motor controller) corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR7 register's setting.

Table 8.10-3 Functions of Port 7 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 7 data register (PDR7)	0	Flash Memory/Mask ROM product When PWM output enabled: Output value of peripheral function is "L" When PWM output disabled: PDR value is "0" EVA product When PWM output enabled: Output value of peripheral function is "L" When peripheral function output disabled, and DDR=0: Pin level is "L" When peripheral function output disabled, and DDR=1: PDR value is "0"	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000007_H	XXXXXXXX_B
	1	Flash Memory/Mask ROM product When PWM output enabled: Output value of peripheral function is "H" When PWM output disabled: PDR value is "1" EVA product When PWM output enabled: Output value of peripheral function is "H" When PWM output disabled, and DDR=0: Pin level is "H" When PWM output disabled, and DDR=1: PDR value is "1"	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 7 direction register (DDR7)	0	Direction latch is "0"	Sets the output buffer to "OFF". Releases the "L" output by a reset.	R/W	000017_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port. Releases the "L" output by a reset.			

R/W: Readable/Writable

X: Undefined value

8.10.2 Description of Port 7 Operation

This section describes the operation of Port 7.

■ Operation of Port 7

- Operation as an output port

Any data written to the PDR7 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR7 register allows the pin values (same values as in the PDR output latch) to be read.

- Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Reading the PDR7 register allows the pin values (peripheral function's output values) to be read.

- Reset operation

The initial states of pins become "L" output.

Note:

The pin states are initialized to "L" output by reset, but this output is performed regardless of the PDR7 register. If this "L" output is not cleared, it cannot be used as a resource output or general-purpose output port. When using it as a resource output or general-purpose output port, must write "0" or "1" into the DDR7 register (P70 to P77) in advance.

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because the output buffer is forcibly set to "OFF".

Table 8.10-4 shows the pin states of Port 7.

Table 8.10-4 Pin States of Port 7

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P70/ PWM1P0 to P77/PWM2M1	General-purpose output port	General-purpose output port	General-purpose output port	Output Hi-Z

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

Hi-Z: High impedance

Table 8.10-5 Priority of Pin Output of Port 7

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P70/ PWM1P0	PWM1P0	P70	-	-
P71/ PWM1M0	PWM1M0	P71	-	-
P72/ PWM2P0	PWM2P0	P72	-	-
P73/ PWM2M0	PWM2M0	P73	-	-
P74/ PWM1P1	PWM1P1	P74	-	-
P75/ PWM1M1	PWM1M1	P75	-	-
P76/PWM2P1	PWM2P1	P76	-	-
P77/PWM2M1	PWM2M1	P77	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

■ Output Driver Driving Power Supply for Port 7

The output driver driving power supply for port 7 is the power supply (DVcc/DVss) for high-current output buffer pin.

- Flash Memory/Mask ROM products

As the Flash Memory/Mask ROM products have DVcc and Vcc isolated from each other, DVcc can be set to a potential higher than Vcc.

- EVA product

As the EVA product does not have DVcc and Vcc isolated from each other, DVcc must be set to a potential equal to or lower than Vcc.

Note:

If DVcc is turned on with Vcc inactive on a Flash Memory/Mask ROM product, port 7 may momentarily output an "H" or "L" level signal at the rise of DVcc. To prevent this, it is advisable to turn on Vcc and DVcc at the same time or to turn on Vcc prior to DVcc.

8.11 Port 8

Port 8 is a general-purpose output-only port that is also used for a peripheral function output port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the functions of this port as a general-purpose output-only port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 8.

■ Port 8 Configuration

Port 8 consists of the following three elements:

- General-purpose output pins and peripheral function output pins (P80/PWM1P2 to P87/PWM2M3)
- Port 8 data register (PDR8)
- Port 8 direction register (DDR8)

■ Port 8 Pins

The pins of Port 8 are also used for peripheral function output pins. If used as peripheral function output pins, they must not be used as general purpose I/O ports. Table 8.11-1 shows the pins of Port 8.

Table 8.11-1 Port 8 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/ output		Circuit type
						Input	Output	
Port 8	P80/PWM1P2	P80	General-purpose output	PWM1P2	Stepping motor controller	-	-	L
	P81/PWM1M2	P81		PWM1M2				
	P82/PWM2P2	P82		PWM2P2				
	P83/PWM2M2	P83		PWM2M2				
	P84/PWM1P3	P84		PWM1P3				
	P85/PWM1M3	P85		PWM1M3				
	P86/PWM2P3	P86		PWM2P3				
	P87/PWM2M3	P87		PWM2M3				

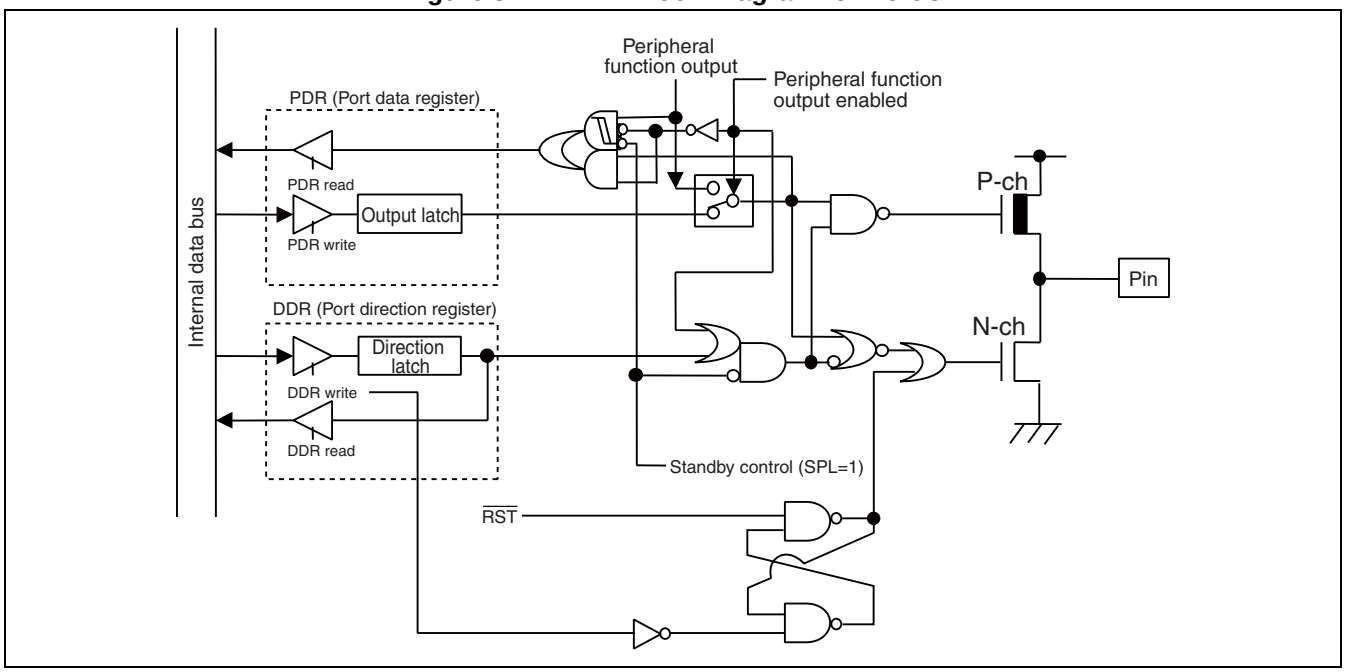
: Function enabled in the initial state

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 8

Figure 8.11-1 shows the pin block diagram for Port 8.

Figure 8.11-1 Pin Block Diagram for Port 8



If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDR8 register value.

■ Registers for Port 8

The Port 8 registers are PDR8 and DDR8. The register bits have a one-to-one correspondence with the Port 8 pins. Table 8.11-2 shows the correspondence between registers and pins for Port 8.

Table 8.11-2 Correspondence between Registers and Pins for Port 8

Port name	Bit of related register and its corresponding pin								
Port 8	PDR8, DDR8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	P87	P86	P85	P84	P83	P82	P81	P80

8.11.1 Port 8 Registers (PDR8, DDR8)

This section describes the registers for Port 8.

■ Functions of Port 8 Registers

- Port 8 data register (PDR8)

The PDR8 register indicates the pin states.

- Port 8 direction register (DDR8)

The DDR8 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or becomes the high impedance state if the bit is set to "0".

Note:

When the output enable bit of the peripheral function (stepping motor controller) corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR8 register's setting.

Table 8.11-3 Functions of Port 8 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 8 data register (PDR8)	0	Flash Memory/Mask ROM product When PWM output enabled: Output value of peripheral function is "L" When PWM output disabled: PDR value is "0" EVA product When PWM output enabled: Output value of peripheral function is "L" When peripheral function output disabled, and DDR=0: Pin level is "L" When peripheral function output disabled, and DDR=1: PDR value is "0"	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000008_H	XXXXXXXX_B
	1	Flash Memory/Mask ROM product When PWM output enabled: Output value of peripheral function is "H" When PWM output disabled: PDR value is "1" EVA product When PWM output enabled: Output value of peripheral function is "H" When PWM output disabled, and DDR=0: Pin level is "H" When PWM output disabled, and DDR=1: PDR value is "1"	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 8 direction register (DDR8)	0	Direction latch is "0"	Sets the output buffer to "OFF". Releases the "L" output by a reset.	R/W	000018_H	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port. Releases the "L" output by a reset.			

R/W: Readable/Writable

X: Undefined value

8.11.2 Description of Port 8 Operation

This section describes the operation of Port 8.

■ Operation of Port 8

● Operation as an output port

With the corresponding DDR8 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR8 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR8 register allows the pin values (same values as in the PDR output latch) to be read.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function.

Reading the PDR8 register allows the pin values (peripheral function's output values) to be read.

● Reset operation

The initial states of pins become "L" output.

Note:

The pin states are initialized to "L" output by reset, but this output is performed regardless of the PDR8 register. If this "L" output is not cleared, it cannot be used as a resource output or general-purpose output port. When using it as a resource output or general-purpose output port, must write "0" or "1" into the DDR8 register (P80 to P87) in advance.

● Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because the output buffer is forcibly set to "OFF".

Table 8.11-4 shows the pin states of Port 8.

Table 8.11-4 Pin States of Port 8

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P80/PWM1P2 to P87/PWM2M3	General-purpose output port	General-purpose output port	General-purpose output port	Output Hi-Z

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

Hi-Z: High impedance

Table 8.11-5 Priority of Pin Output of Port 8

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P80/PWM1P2	PWM1P2	P80	-	-
P81/PWM1M2	PWM1M2	P81	-	-
P82/PWM2P2	PWM2P2	P82	-	-
P83/PWM2M2	PWM2M2	P83	-	-
P84/PWM1P3	PWM1P3	P84	-	-
P85/PWM1M3	PWM1M3	P85	-	-
P86/PWM2P3	PWM2P3	P86	-	-
P87/PWM2M3	PWM2M3	P87	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

■ Output Driver Driving Power Supply for Port 8

The output driver driving power supply for port 8 is the power supply (DVcc/DVss) for high-current output buffer pin.

- Flash Memory/Mask ROM products

As the Flash Memory/Mask ROM products have DVcc and Vcc isolated from each other, DVcc can be set to a potential higher than Vcc.

- EVA product

As the EVA product does not have DVcc and Vcc isolated from each other, DVcc must be set to a potential equal to or lower than Vcc.

Note:

If DVcc is turned on with Vcc inactive on a Flash Memory/Mask ROM product, port 8 may momentarily output an "H" or "L" level signal at the rise of DVcc. To prevent this, it is advisable to turn on Vcc and DVcc at the same time or to turn on Vcc prior to DVcc.

8.12 Port 9

Port 9 is a general-purpose I/O port that is also used for a peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port 9.

■ Port 9 Configuration

Port 9 consists of the following four elements:

- General-purpose I/O pins and peripheral function input/output pins (P90/SEG22, P91/SEG23)
- General-purpose I/O pins and peripheral function input/output pins (P94/V0 to P96/V2)
- Port 9 data register (PDR9)
- Port 9 direction register (DDR9)

■ Port 9 Pins

The pins of Port 9 are also used for peripheral function I/O pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.12-1 shows the pins of Port 9.

Table 8.12-1 Port 9 Pins

Port name	Pin name	Port function		Peripheral function		Type of input/output		Circuit type
						Input	Output	
Port 9	P90/SEG22	P90	General-purpose I/O	SEG22	LCD Controller	CMOS Hysteresis/Automotive level*	CMOS	F
	P91/SEG23	P91		SEG23				
	P92*1 system	P92						I
	P93*1 system	P93						
	P94/V0	P94		V0	External divide			G
	P95/V1	P95		V1				
	P96/V2	P96		V2				

: Function enabled in the initial state

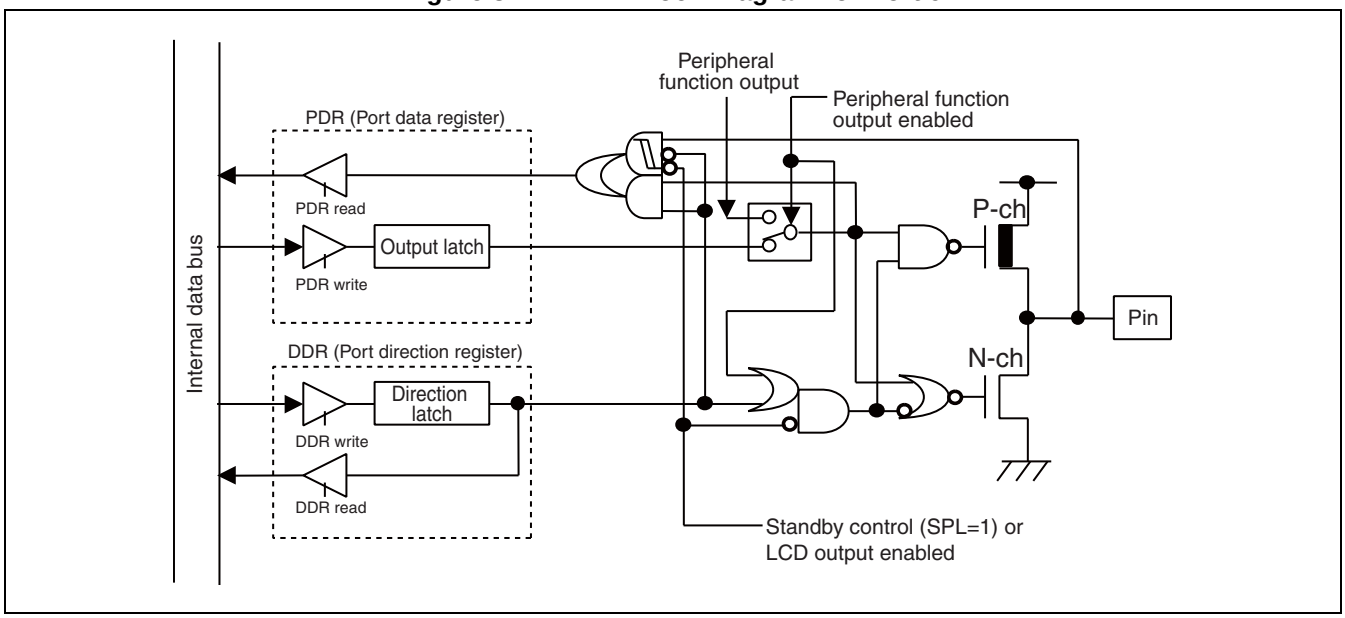
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port 9

Figure 8.12-1 shows the pin block diagram for Port 9.

Figure 8.12-1 Pin Block Diagram for Port 9



If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function's output irrespective of the DDR9 register value.

■ Registers for Port 9

The Port 9 registers are PDR9 and DDR9. The register bits have a one-to-one correspondence with Port 9 pins. Table 8.12-2 shows the correspondence between registers and pins for Port 9.

Table 8.12-2 Correspondence between Registers and Pins for Port 9

Port name	Bit of related register and its corresponding pin								
Port 9	PDR9, DDR9	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	-	P96	P95	P94	P93	P92	P91	P90

8.12.1 Registers for Port 9 (PDR9, DDR9)

This section describes the registers for Port 9.

■ Functions of Port 9 Registers

- Port 9 data register (PDR9)

The PDR9 register indicates the pin states.

- Port 9 direction register (DDR9)

The DDR9 register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDR9 register's setting.

Table 8.12-3 shows functions of Port 9 registers.

Table 8.12-3 Functions of Port 9 Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port 9 data register (PDR9)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	000009_H	$-XXXXXXX_B$
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port 9 direction register (DDR9)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	000019_H	-0000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value
- : Undefined

8.12.2 Description of Port 9 Operation

This section describes the operation of Port 9.

■ Operation of Port 9

● Operation as an output port

With the corresponding DDR9 register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDR9 register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDR9 register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then write "1" to the DDR register.

● Operation as an input port

With the corresponding DDR9 register bit set to "0", the port works as an input port.

In working as an input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDR9 register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDR9 register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDR9 register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDR9 register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDR9 register is not initialized. Therefore, if used as an output port, set the output data in the PDR9 register and then set the corresponding DDR9 register to "1".

Port 9 has its feature as the LCD controller/driver reference power supply (V0 to V2) enabled in the initial state. To use it

as a general-purpose port, set the corresponding bit in the LCD output control register (LOCR3) to "0".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDR9 register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.12-4 shows the pin states of Port 9.

Table 8.12-4 Pin States of Port 9

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
P90/SEG22 to P96/V2	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

SPL: Pin state specification bit of low-power consumption mode control register (SPL in LPMCR)
Hi-Z: High impedance

Table 8.12-5 Priority of Pin Output of Port 9

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
P90/SEG22	SEG22	P90	-	-
P91/SEG23	SEG23	P91	-	-
P92	P92	-	-	-
P93	P93	-	-	-
P94/V0	P94	-	-	-
P95/V1	P95	-	-	-
P96/V2	P96	-	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.13 Port C

Port C is a general-purpose I/O port that is also used for a peripheral function input port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port C.

■ Port C Configuration

Port C consists of the following three elements.

- General-purpose I/O pins and external interrupt input pins (PC0/SIN0/INT4 to PC7/PPG1/TIN1/IN6)
- Port C data register (PDRC)
- Port C direction register (DDRC)

■ Port C Pins

The pins of Port C pins are also used for peripheral function input/output pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.13-1 shows the pins of Port C.

Table 8.13-1 Port C Pins

Port name	Pin name	Port function		Peripheral function						Type of input/output		Circuit type
										Input	Output	
Port C	PC0/ SIN0/ INT4	PC0	General-purpose I/O	SIN0	UART			INT4	External interrupt	CMOS CMOS Hysteresis/Automotive level*	CMOS	J
	PC1/ SOT0/ INT5/ IN3	PC1		SOT0		IN3	ICU	INT5		CMOS Hysteresis/ Automotive level*		I
	PC2/ SCK0/ INT6/ IN2	PC2		SCK0		IN2		INT6				
	PC3/ SIN1/ INT7	PC3		SIN1				INT7		CMOS CMOS Hysteresis/ Automotive level*		J
	PC4/ SOT1	PC4		SOT1						CMOS Hysteresis/ Automotive level*		I
	PC5/ SCK1/ TRG	PC5		SCK1				TRG	RLT			
	PC6/ PPG0/ TOT1/ IN7	PC6		PPG0	PPG	IN7	ICU	TOT1				
	PC7/ PPG1/ TIN1/ IN6	PC7		PPG1		IN6		TIN1				

: Function enabled in the initial state

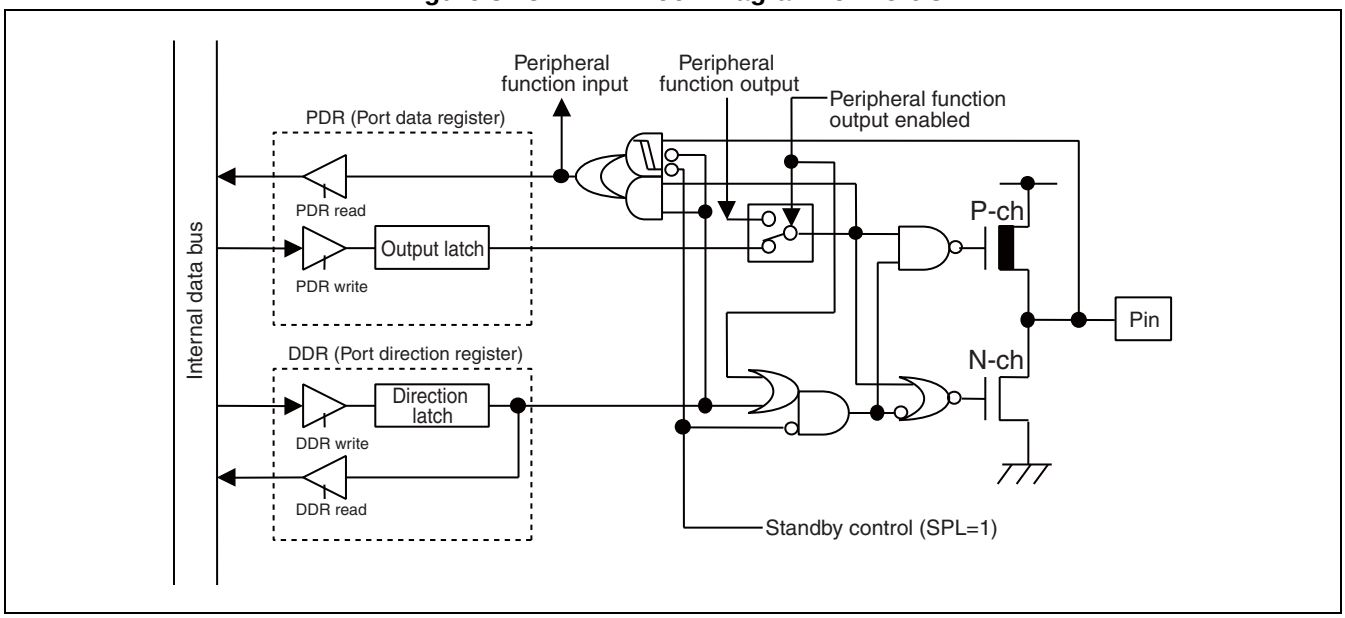
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").
For SIN0 and SIN1 pins, CMOS hysteresis 0.7Vcc/0.3Vcc can also be selected.

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port C

Figure 8.13-1 shows the pin block diagram for Port C.

Figure 8.13-1 Pin Block Diagram for Port C



■ Registers for Port C

The Port C registers are PDRC and DDRC. The register bits have a one-to-one correspondence with the Port C pins. Table 8.13-2 shows the correspondence between the registers and pins for Port C.

Table 8.13-2 Correspondence between Registers and Pins for Port C

Port name	Bit of related register and its corresponding pin								
Port C	PDRC, DDRC	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

8.13.1 Registers for Port C (PDRC, DDRC)

This section describes the registers for Port C.

■ Functions of Port C Registers

● Port C data register (PDRC)

The PDRC register indicates the pin states.

● Port C direction register (DDRC)

The DDRC register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Notes:

- When a peripheral function with an output pin is used, and the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDRC register's setting.
- If a peripheral function with an input pin is used, set the DDRC register bit corresponding to each peripheral function's input pin to "0" to make it work as an input port.

Table 8.13-3 Functions of Port C Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port C data register (PDRC)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	$00000C_H$	XXXXXXXX_B
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port C direction register (DDRC)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	$00001C_H$	00000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X : Undefined value

8.13.2 Description of Port C Operation

This section describes the operation of Port C.

■ Operation of Port C

● Operation as an output port

With the corresponding DDRC register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDRC register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDRC register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDRC register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDRC register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDRC register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDRC register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Operation as a peripheral function input

For a port that is also used for a peripheral function input, the pin value is always set. To use an external signal as input for the peripheral function, set the DDRC register for the input port to "0".

- Reset operation

At CPU reset, the DDRC register value is cleared. Thus, all the output buffers are set to "OFF" (input port), and, the pins are set to high impedance.

In a reset operation, the PDRC register is not initialized. Therefore, if used as an output port, set the output data in the PDRC register and then set the corresponding DDRC register to "1".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDRC register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.13-4 shows the pin states of Port C.

Table 8.13-4 Pin States of Port C

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
PC0/SIN0/INT4 to PC7/PPG1/TIN1/IN6	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High impedance

Note:

To set a pin to high-impedance when the pin is shared by a peripheral function and a port in stop mode, watch mode, or time-base timer mode, disable the output of peripheral functions, and set the STP bit to "1" or TMD bit to "0" in the low-power consumption mode control register (LPMCR).

Table 8.13-5 Priority of Pin Output of Port C

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
PC0/SIN0/INT4	PC0	-	-	-
PC1/SOT0/INT5/IN3	SOT0	PC1	-	-
PC2/SCK0/INT6/IN2	SCK0	PC2	-	-
PC3/SIN1/INT7	PC3	-	-	-
PC4/SOT1	SOT1	PC4	-	-
PC5/SCK1/TRG	SCK1	PC5	-	-
PC6/PPG0/TOT1/IN7	PPG0	TOT1	PC6	-
PC7/PPG1/TIN1/IN6	PPG1	PC7	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.14 Port D

Port D is a general-purpose I/O port that is also used for a peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port D.

■ Port D Configuration

Port D consists of the following three elements.

- General-purpose I/O pins and peripheral function input/output pins (PD0/SIN2 to PD6/TOT2)
- Port D data register (PDRD)
- Port D direction register (DDRD)

■ Pins of Port D

The I/O pins of Port D are also used for peripheral function input/output pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.14-1 shows the Port D pins.

Table 8.14-1 Port D Pins

Port name	Pin name	Port function		Peripheral function		Type of Input/Output		Circuit type
						Input	Output	
Port D	PD0/SIN2	PD0	General-purpose I/O	SIN2	UART	CMOS/ CMOS Hysteresis/Automotive level*	CMOS	J
	PD1/SOT2	PD1		SOT2		CMOS Hysteresis/ Automotive level*		I
	PD2/SCK2	PD2		SCK2				
	PD3/SIN3	PD3		SIN3		CMOS/ CMOS Hysteresis/ Automotive level*		J
	PD4/SOT3	PD4		SOT3		CMOS Hysteresis/ Automotive level*		I
	PD5/SCK3	PD5		SCK3				
	PD6/TOT2	PD6		TOT2	RLT			

: Function enabled in the initial state

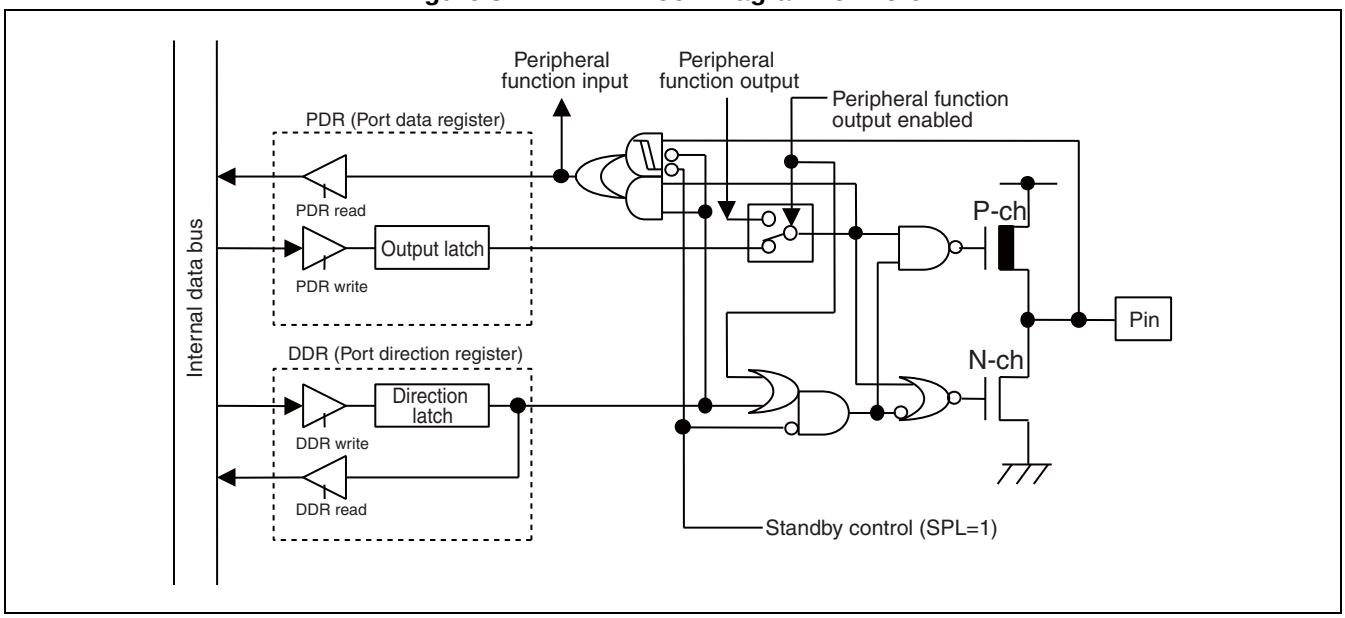
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").
SIN2 and SIN3 pins also select CMOS Hysteresis 0.7Vcc/0.3Vcc.

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port D

Figure 8.14-1 shows the pin block diagram for Port D.

Figure 8.14-1 Pin Block Diagram for Port D



■ Registers for Port D

The Port D registers are PDRD and DDRD. The register bits have a one-to-one correspondence with the Port D pins. Table 8.14-2 shows the correspondence between the registers and pins for Port D.

Table 8.14-2 Correspondence between Registers and Pins for Port D

Port name	Bit of related register and its corresponding pin								
Port D	PDRD, DDRD	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	-	PD6	PD5	PD4	PD3	PD2	PD1	PD0

8.14.1 Registers for Port D (PDRD, DDRD)

This section describes the registers for Port D.

■ Functions of Port D Registers

- Port D data register (PDRD)

The PDRD register indicates the pin states.

- Port D direction register (DDRD)

The DDRD register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, if the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDRD register's setting.

Table 8.14-3 Functions of Port D Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port D data register (PDRD)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	$00000D_H$	$-XXXXXXX_B$
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port D direction register (DDRD)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	$00001D_H$	-0000000_B
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value
- : Undefined

8.14.2 Description of Port D Operation

This section describes the operation of Port D.

■ Operation of Port D

● Operation as an output port

With the corresponding DDRD register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDRD register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDRD register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDRD register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDRD register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDRD register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDRD register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDRD register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDRD register is not initialized. Therefore, if used as an output port, set the output data in the PDRD register and then set the corresponding DDRD register to "1".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDRD register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.14-4 shows the pin states of Port D.

Table 8.14-4 Pin States of Port D

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
PD0/SIN2 to PD6/TOT2	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High impedance

Table 8.14-5 Priority of Pin Output of Port D

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
PD0/SIN2	PD0	-	-	-
PD1/SOT2	SOT2	PD1	-	-
PD2/SCK2	SCK2	PD2	-	-
PD3/SIN3	PD3	-	-	-
PD4/SOT3	SOT3	PD4	-	-
PD5/SCK3	SCK3	PD5	-	-
PD6/TOT2	TOT2	PD6	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.15 Port E

Port E is a general-purpose I/O port that is also used for a peripheral function I/O port. The use of each pin can be switched per bit between the peripheral function and the port. This section mainly describes the function of this port as a general-purpose I/O port. It indicates the configuration, pins, block diagrams of pins, and registers for Port E.

■ Port E Configuration

Port E consists of the following three elements.

- General-purpose I/O pins and peripheral function input/output pins (PE0/TOT3 to PE2/SGO1)
- Port E data register (PDRE)
- Port E direction register (DDRE)

■ Port E Pins

The I/O pins of Port E are also used for peripheral function input/output pins. If used as peripheral function I/O pins, they must not be used as general purpose I/O ports. Table 8.15-1 shows the pins of Port E.

Table 8.15-1 Port E Pins

Port name	Pin name	Port function		Peripheral function		Type of Input/Output		Circuit type
						Input	Output	
Port E	PE0/TOT3	PE0	General-purpose I/O	TOT3	RLT	CMOS Hysteresis/Automotive level*	CMOS	I
	PE1/TIN3	PE1		TIN3				
	PE2/SGO1	PE2		SGO1	SG			

: Function enabled in the initial state

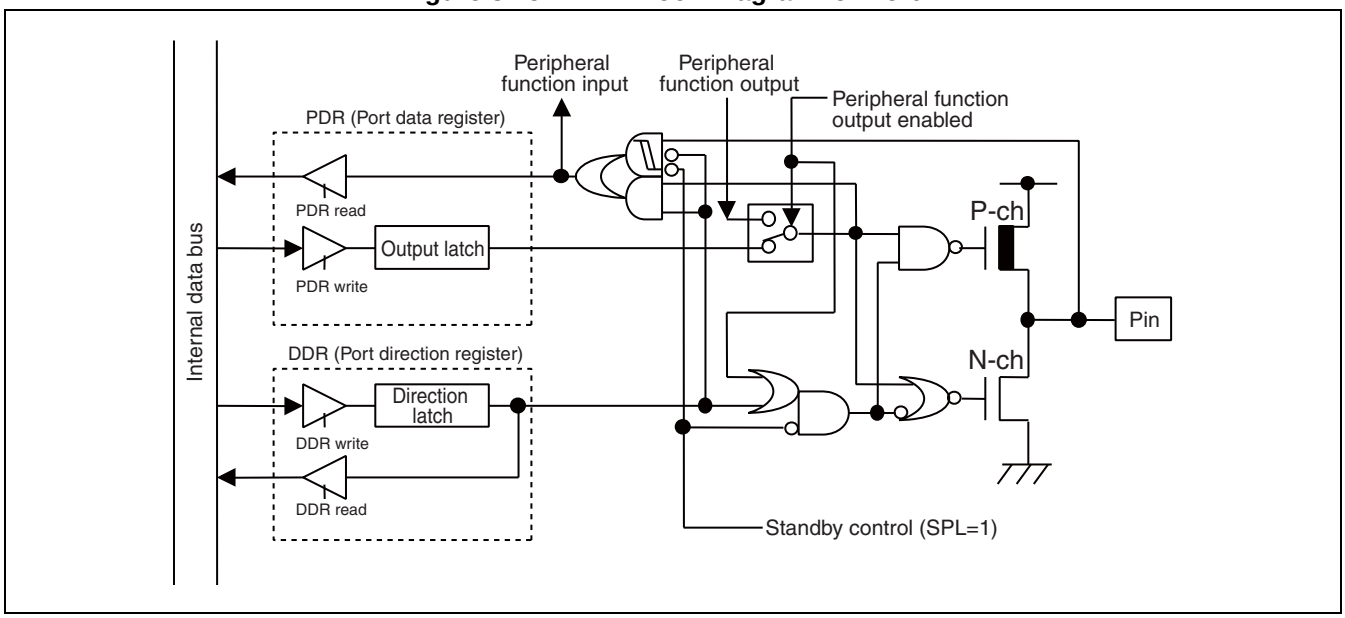
*:Automotive level is a standard for input voltage. For the standard values, please refer to the data sheet ("3. DC Characteristics" in "■ ELECTRICAL CHARACTERISTICS").

For the circuit type, refer to Section "1.7 I/O Circuit Types".

■ Pin Block Diagram for Port E

Figure 8.15-1 shows the pin block diagram for Port E.

Figure 8.15-1 Pin Block Diagram for Port E



Note:

If the peripheral function's output enable bit is set to "enabled", the corresponding pin is forced to work as a peripheral function output irrespective of the DDRE register value.

■ Registers for Port E

The Port E registers are PDRE and DDRE. The register bits have a one-to-one correspondence with the Port E pins. Table 8.15-2 shows the correspondence between the registers and pins for Port E.

Table 8.15-2 Correspondence between Registers and Pins for Port E

Port name	Bit of related register and its corresponding pin								
Port E	PDRE, DDRE	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	Corresponding pin	-	-	-	-	-	PE2	PE1	PE0

8.15.1 Registers for Port E (PDRE, DDRE)

This section describes the registers for Port E.

■ Functions of Port E Registers

- Port E data register (PDRE)

The PDRE register indicates the pin states.

- Port E direction register (DDRE)

The DDRE register is used to set the pin input/output direction for each bit. The pin works as an output port if the bit corresponding to the port (pin) is set to "1" or as an input port if the bit is set to "0".

Note:

When a peripheral function with an output pin is used, if the output enable bit of each peripheral function corresponding to the pin is set to "enabled", the pin works as a peripheral function output pin irrespective of the DDRE register's setting.

Table 8.15-3 Functions of Port E Registers

Register name	Data	When reading	When writing	R/W	Address	Initial value
Port E data register (PDRE)	0	Pin state is "L" level	Sets the output latch to "0", and outputs "L" level when used as an output port	R/W	$00000E_H$	$-----XXX_B$
	1	Pin state is "H" level	Sets the output latch to "1", and outputs "H" level when used as an output port			
Port E direction register (DDRE)	0	Direction latch is "0"	Sets the output buffer to "OFF" to be an input port	R/W	$00001E_H$	$-----000_B$
	1	Direction latch is "1"	Sets the output buffer to "ON" to be an output port			

R/W: Readable/Writable
X: Undefined value
- : Undefined

8.15.2 Description of Port E Operation

This section describes the operation of Port E

■ Operation of Port E

● Operation as an output port

With the corresponding DDRE register bit set to "1", the port works as an output port.

When used as an output port, any data written to the PDRE register is retained in the PDR output latch and then output to the pins as it is.

Reading the PDRE register allows the pin values (same values as in the PDR output latch) to be read.

Note:

If the port data register uses a read-modify-write (RMW) instruction (e.g., a bit set instruction), the target bit is set to the value specified and the output bit specified by the DDR register is not affected. However, the input bit specified by the DDR register writes the input value from the pin to the output latch, and the value is output. Therefore, to switch the input bit to the output bit, write the output data to the PDR register and then set the DDR register as the output.

● Operation as an input port

With the corresponding DDRE register bit set to "0", the port works as an input port.

In working as input port, the output buffer is set to "OFF", and the pins to high impedance.

If data is written to the PDRE register, it is retained in the PDR output latch, but not output to the pins.

The pin levels ("L" or "H") are read from the PDRE register.

● Operation as a peripheral function output

To use the port as a peripheral function output, set it with the output enable bit of the peripheral function. Switching the input/output has priority over the peripheral function's output enable bit. Therefore, even if the DDRE register bit is set to "0", the port is used as a peripheral function output as long as the corresponding peripheral function is output enabled. Even if a peripheral function is output enabled, the pin value can be read, allowing the peripheral function's output value to be detected.

● Reset operation

At CPU reset, the DDRE register value is cleared. Thus, all the output buffers are set to "OFF" (input port) and the pins are set to high impedance.

In a reset operation, the PDRE register is not initialized. Therefore, if used as an output port, set the output data in the PDRE register and then set the corresponding DDRE register to "1".

- Operation for the stop and time-base timer modes

If the pin state specification bit of the low-power consumption mode control register (SPL in LPMCR) is "1" when a transition to the stop mode or time-base timer mode occurs, the pins are set to high impedance. This is because, irrespective of the DDRE register value, the output buffer is forcibly set to "OFF". To prevent leakage due to the input open, the input is fixed.

Table 8.15-4 shows the pin states of Port E.

Table 8.15-4 Pin States of Port E

Pin name	Normal operation	Sleep mode	Stop mode, time-base timer mode (SPL=0)	Stop mode, time-base timer mode (SPL=1)
PE0/TOT3 to PE2/SGO1	General-purpose I/O port	General-purpose I/O port	General-purpose I/O port	Input cut-off/output Hi-Z

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

Hi-Z: High impedance

Table 8.15-5 Priority of Pin Output of Port E

Pin name	Priority 1	Priority 2	Priority 3	Priority 4
PE0/TOT3	TOT3	PE0	-	-
PE1/TIN3	PE1	-	-	-
PE2/SGO1	SGO1	PE2	-	-

Note: Priority 1 has the highest priority and Priority 4 has the lowest priority.

8.16 Input Level Select Registers (PIL0 to PIL2)

The input level select registers allow to switch from Automotive input levels (V_{IH}/V_{IL}= $0.8V_{CC}/0.5V_{CC}$) to CMOS Hysteresis input levels ($V_{IH}/V_{IL}=0.8V_{CC}/0.2V_{CC}$). In addition, the serial input pin (SIN) can select CMOS input levels ($V_{IH}/V_{IL}=0.7V_{CC}/0.3V_{CC}$).

■ Input Level Select Register 0 (PIL0)

	Address bit	7	6	5	4	3	2	1	0
PIL0	$00008C_H$	Reserved	ILP6	ILP5	ILP4	ILP3	ILP2	ILP1	ILP0
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Initial value	0	0	0	0	0	0	0	0

R/W: Readable/Writable

- bit7: Reserved bit

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

- bit6 to bit0:ILP6 to ILP0

These bits set the input level of corresponding ports.
ILP6 to ILP0 correspond to Port 6 to Port 0 respectively.

"When set to "0": Sets to Automotive input level.

"When set to 1": Sets to CMOS hysteresis input level ($V_{IH}/V_{IL}=0.8V_{CC}/0.2V_{CC}$).

■ Input Level Selection Register 1 (PIL1)

	Address bit	15	14	13	12	11	10	9	8
PIL1	$00008D_H$	-	-	-	Reserved	ILSIN1	ILSIN0	ILP9	Reserved
	Read/Write	-	-	-	R/W	R/W	R/W	R/W	R/W
	Initial value	X	X	X	0	0	0	0	0

R/W: Readable/Writable
- : Undefined
X: Undefined value

- bit15 to bit13: Undefined bits

These are undefined bits. Writing has no effect on operation. Read value is undefined.

- bit12: Reserved bit

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

- bit11:ILSIN1

Selects the input level of serial input pin of UART1(SIN1).

When set to "0": Sets to the level specified in PIL0 register.

When set to "1": Sets to CMOS input ($0.7V_{CC}/0.3V_{CC}$) level.

- bit10: ILSIN0

Selects the input level of serial input pin of UART0(SIN0).

When set to "0": Sets to the level specified in PIL0 register.

When set to "1": Sets to CMOS input ($0.7V_{CC}/0.3V_{CC}$) level.

- bit9: ILP9

Selects the input level of Port 9.

"When set to "0": Sets to Automotive input level.

"When set to 1": Sets to CMOS hysteresis input level ($V_{IH}/V_{IL}=0.8V_{CC}/0.2V_{CC}$).

- bit8: Reserved bit

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

■ Input Level Selection Register 2 (PIL2)

	Address bit	7	6	5	4	3	2	1	0
PIL2	$00008E_H$	-	-	-	ILSIN3	ILSIN2	ILPE	ILPD	ILPC
	Read/Write	-	-	-	R/W	R/W	R/W	R/W	R/W
	Initial value	X	X	X	0	0	0	0	0

R/W: Readable/Writable
- : Undefined
X: Undefined value

- bit7 to bit5: Undefined bits

These are undefined bits. Writing has no effect on operation. Read value is undefined.

- bit4: ILSIN3

These bits are used to select the input level for the serial input pin (SIN3) of UART3.

When set to "0": Sets to the level specified in PIL2 register.

When set to "1": Sets to CMOS input ($0.7V_{CC}/0.3V_{CC}$) level.

- bit3: ILSIN2

These bits are used to select the input level for the serial input pin (SIN2) of UART2.

When set to "0": Sets to the level specified in PIL2 register.

When set to "1": Sets to CMOS input ($0.7V_{CC}/0.3V_{CC}$) level.

- bit2 to bit0:ILPE to ILPC

These bits set the input levels of corresponding ports.

ILPE to ILPC correspond to Port E to Port C respectively.

"When set to "0": Sets to Automotive input level.

"When set to 1": Sets to CMOS input level ($V_{IH}/V_{IL}=0.8V_{CC}/0.2V_{CC}$).

Note:

The threshold of the corresponding input pin varies immediately after the setting of the input level select register is changed. Therefore, do not use the read value from the pin until 2 machine cycles are elapsed after the setting is changed.
When the setting is changed, be sure to disable the corresponding resource.

8.17 Sample Program for I/O Ports

A sample program that uses the I/O port is shown below.

■ Sample Program for I/O Ports

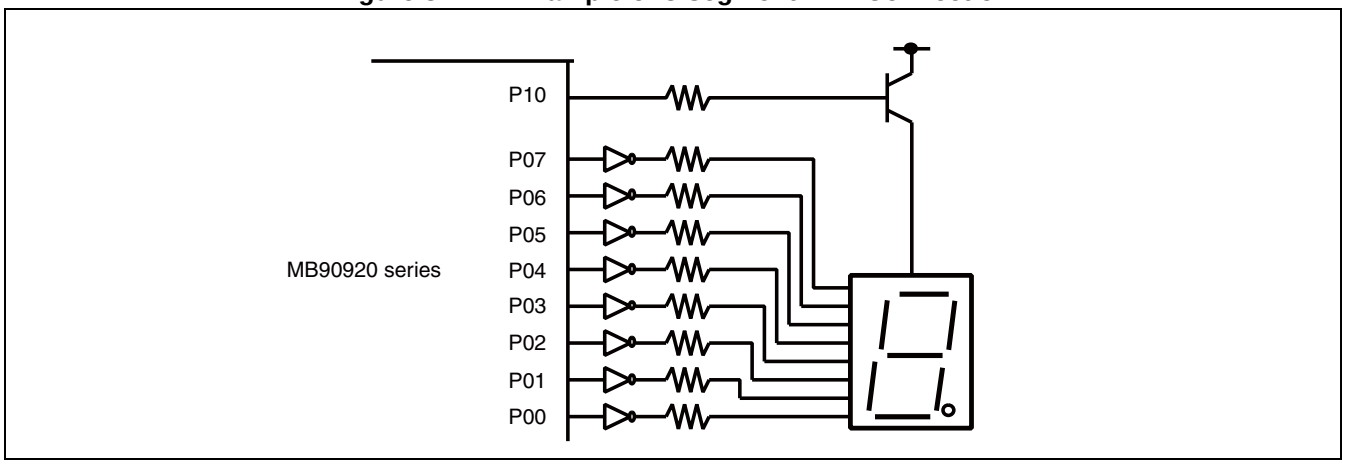
● Specification of processing

On ports 0 and 1, all 7-segment LEDs (8-segment if Dp is included) are on.

The P10 pin corresponds to the LED's common anode pin, and the P00 to P07 pins correspond to the segment pins.

Figure 8.17-1 shows an example of 8-segment LED connection.

Figure 8.17-1 Example of 8-segment LED Connection



[Coding example]

```
PDR0    EQU     000000H
PDR1    EQU     000001H
DDR0    EQU     000010H
DDR1    EQU     000011H
;---------Main program-----------------------------------------------------------
CODE            CSEG
START:
                                        ;Initial setting completed
        MOV     I:PDR1,  #00000000B     ;P10 set to "L" level,
        MOV     I:DDR1,  #11111111B     ;All bits in Port 1 set as outputs
        MOV     I:PDR0,  #11111111B     ;All bits in Port 0 set to "1"
        MOV     I:DDR0,  #11111111B     ;All bits in Port 0 set as outputs
CODE            ENDS
;-------------------------------------------------------------------------------
        END     START
```

9. Watchdog Timer/Time-Base Timer/Watch Timer (Used as Sub Clock)

This chapter describes the functions and operations of the watchdog timer, time-base timer, and watch timer (used as sub clock).

9.1 Outline of Watchdog Timer/Time-base Timer/Watch Timer

9.2 Block Diagrams of Watchdog Timer/Time-base Timer/Watch Timer

9.3 List of Registers for Watchdog Timer/Time-base Timer/Watch Timer

9.4 Operation of Watchdog Timer/Time-base Timer/Watch Timer

9.5 Notes on Using the Watchdog Timer/Time-base Timer

9.6 Program Example for Watchdog Timer/Time-base Timer

9.1 Outline of Watchdog Timer/Time-base Timer/Watch Timer

The circuit configurations of the watchdog timer, time-base timer, and watch timer are shown below respectively.

- **Watchdog timer: watchdog counter, control register, and watchdog reset circuit**
- **Time-base timer: 18-bit timer, circuit to control interval interrupts**
- **Watch timer: 15-bit timer, circuit to control interval interrupts**

■ Functions of the Watchdog Timer

The watchdog timer consists of a 2-bit watchdog counter that has a clock source using the carry-over signal from a 18-bit time-base timer or 15-bit watch timer, control register, and watchdog reset control section. If the timer is not cleared within a certain time after the startup, the CPU will be reset.

■ Functions of Time-base Timer

The time-base timer is an 18-bit free-run counter (time-base counter) that counts up synchronously with the main clock (in divide-by-2 of oscillation clock). It has an interval timer function to select one of four interval times. It also has a function to supply operation clocks of the oscillation stabilization wait time timer output, and the watchdog timer, etc. The time-base timer uses the main clock irrespective of the MCS and SCS bits in CKSCR.

■ Watch Timer Function

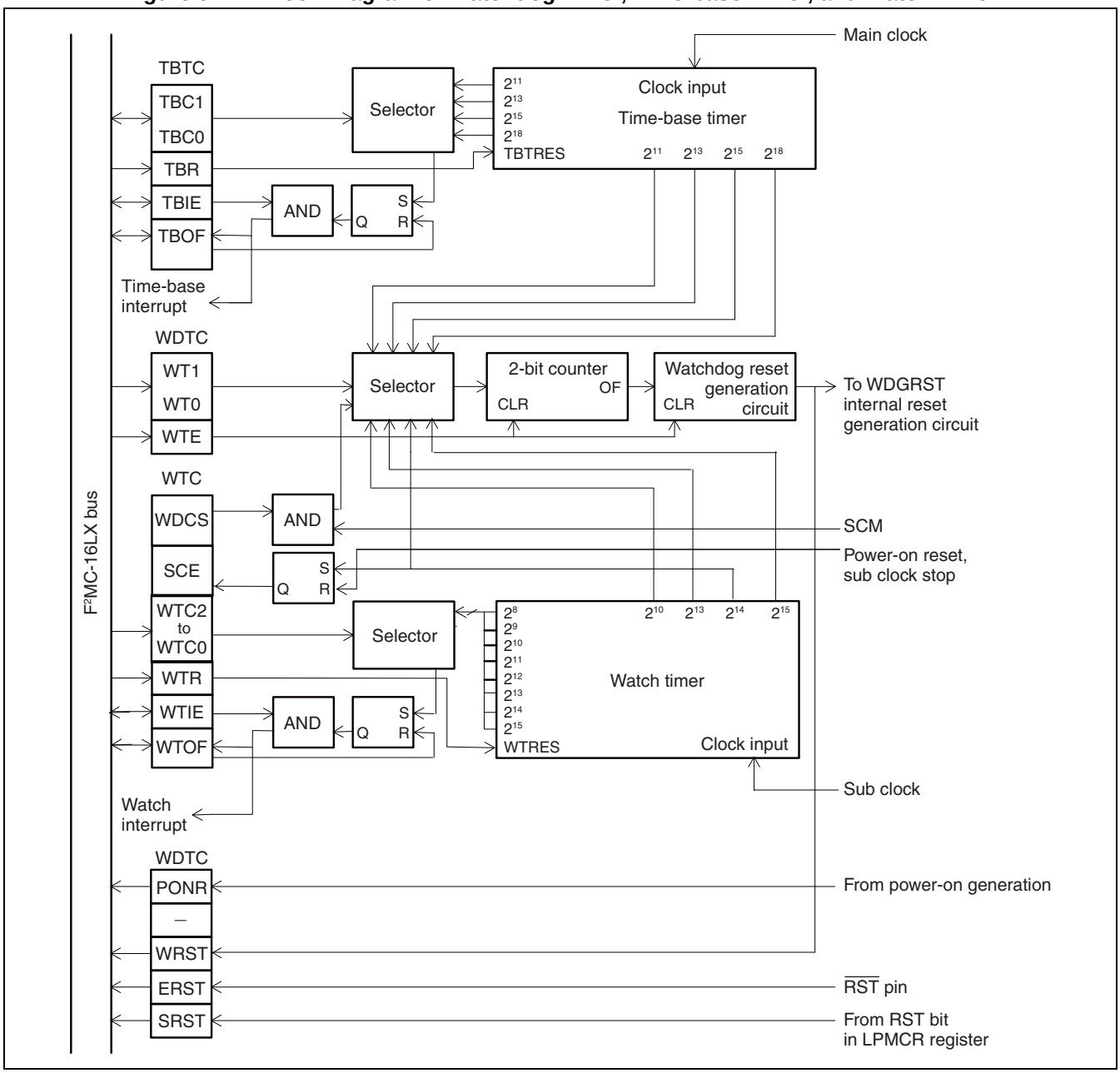
The watch timer, which is a timer for the watchdog timer's clock source and sub clock oscillation stabilization time waiting, has a function of an interval timer by regularly generating an interrupt. In addition, it uses the sub clock irrespective of the MCS and SCS bits in CKSCR.

9.2 Block Diagrams of Watchdog Timer/Time-base Timer/ Watch Timer

The block diagrams of the watchdog timer/time-base timer/watch timer are shown below.

■ Block Diagram of Watchdog Timer/Time-base Timer/Watch Timer

Figure 9.2-1 Block Diagram of Watchdog Timer, Time-base Timer, and Watch Timer



9.3 List of Registers for Watchdog Timer/Time-base Timer/ Watch Timer

This section describes the list of registers for the watchdog timer, time-base timer, and watch timer.

■ List of Registers of Watchdog Timer/Time-base Timer/Watch Timer

Figure 9.3-1 lists the registers used for the watchdog timer, time-base timer, and watch timer.

Figure 9.3-1 List of Registers for Watchdog Timer/Time-base Timer/ Watch Timer

Watchdog Control Register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $0000A8_H$	PONR	–	WRST	ERST	SRST	WTE	WT1	WT0	WDTC
Read/Write →	R	–	R	R	R	W	W	W	
Initial value →	X	-	X	X	X	1	1	1	

Time-base timer control register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $0000A9_H$	Reserved	–	–	TBIE	TBOE	TBR	TBC1	TBC0	TBTC
Read/Write →	–	–	–	R/W	R/W	W	R/W	R/W	
Initial value →	1	-	-	0	0	1	0	0	

Watch timer control register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $0000AA_H$	WDCS	SCE	WTIE	WTOF	WTR	WTC2	WTC1	WTC0	WTC
Read/Write →	R/W	R	R/W	R/W	W	R/W	R/W	R/W	
Initial value →	1	0	0	0	1	0	0	0	

9.3.1 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) indicates the watchdog timer start, clear, and reset sources.

■ Bit Configuration of the Watchdog Timer Control Register (WDTC)

Figure 9.3-2 shows the bit configuration of the watchdog timer control register (WDTC).

Figure 9.3-2 Bit Configuration of the Watchdog Timer Control Register (WDTC)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $0000A8_H$	PONR	–	WRST	ERST	SRST	WTE	WT1	WT0	WDTC
Read/Write →	R	–	R	R	R	W	W	W	
Initial value →	X	-	X	X	X	1	1	1	

Note:

Do not access using read-modify-write (RMW) instructions, since they may cause an error in operation.

[bit7, bit5 to bit3] PONR, WRST, ERST, SRST

PONR, WRST, ERST, and SRST are flags indicating reset sources. They are set as shown in Table 9.3-1 by reset. All bits are cleared after reading the WDTC register.

These bits are read-only register.

Table 9.3-1 PONR, WRST, ERST, and SRST (reset Source Bits)

Reset source	PONR	WRST	ERST	SRST
Power-on	1	-	-	-
Watchdog Timer	*	1	*	*
External pin ($\overline{RST}$ input) CPU operation detection reset	*	*	1	*
Low-voltage detection reset	1	*	1	*
RST bit (software reset)	*	*	*	1

*: Previous value is retained.

-: Undefined

[bit2] WTE

While the watchdog timer is in the stop state, if WTE is written with "0", the watchdog timer starts operating. By writing "0" twice or more, the watchdog timer counter is cleared. Writing "1" has no effect.

The watchdog timer stops if any reset source is generated. "1" is output in read operations.

[bit1, bit0] WT1, WT0

WT1 and WT0 are bits used to select the watchdog timer's interval time. Data is valid only when it is written during the startup of the watchdog timer. Data written during other than the startup of the watchdog timer is ignored. These bits can only be written.

The clock input to the watchdog timer is selected with the 3 bits: the WDCS bit of the watch timer control register WTC, the result of a logical AND of the SCM bit in the low-power consumption control circuit clock selection register LPMCR, and the WT1 and WT0 bits.

This is, with WDCS set to "1", if the main clock and PLL clock are selected as the machine clock, the output of the time-base timer can be selected as the input clock of the watchdog timer. Alternatively, if the sub clock timer is selected, the output of the watch timer can be selected as the input clock of the watchdog timer.

Table 9.3-2 shows setting the interval time with the WT1/WT0 bits.

Table 9.3-2 WT1, WT0 (Interval Time Selection Bits)

Logical AND	WT1	WT0	Interval time (4MHz oscillation)	
			Minimum	Maximum*
1	0	0	Approx. 3.58 ms	Approx. 4.61 ms
1	0	1	Approx. 14.33 ms	Approx. 18.43 ms
1	1	0	Approx. 57.23 ms	Approx. 73.73 ms
1	1	1	Approx. 458.75 ms	Approx. 589.82 ms
0	0	0	Approx. 436 ms	Approx. 563 ms
0	0	1	Approx. 3.50 s	Approx. 4.50 s
0	1	0	Approx. 7.0 s	Approx. 9.0 s
0	1	1	Approx. 14.0 s	Approx. 18.0s

* : The maximum interval time is the value available when the watchdog timer is operating and the time-base timer or watch timer is not reset.

9.3.2 Time-base Timer Control Register (TBTC)

The time-base timer control register (TBTC) is used to control interrupts by the time-base timer and clear the time-base counter.

■ Bit Configuration of the Time-base Timer Control Register (TBTC)

Figure 9.3-3 shows the bit configuration of the time-base timer control register (TBTC).

Figure 9.3-3 Bit Configuration of the Time-base Timer Control Register (TBTC)

Time-base timer control register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 0000A9$_H$	Reserved	–	–	TBIE	TBOE	TBR	TBC1	TBC0	TBTC
Read/Write →	–	–	–	R/W	R/W	W	R/W	R/W	
Initial value →	1	-	-	0	0	1	0	0	

[bit15] Reserved

bit15 is a reserved bit. Always set it to "1".

[bit12] TBIE

The TBIE bit is used to enable interval interrupts by the time-base timer. If set to "1", interrupts are enabled, if set to "0", interrupts are disabled. This bit is cleared by reset. This bit can be read and written.

[bit11] TBOF

TBOF is an interrupt request flag of the time-base timer. With the TBIE bit set to "1", if the TBOF bit is set to "1", an interrupt request is generated. It is set to "1" in intervals specified by the bits TBC1 and TBC0.

The TBOF bit is cleared by the conditions listed below.

- Writing "0"
- Transition to main stop mode
- Transition to PLL stop mode
- Transition from sub clock mode to main clock mode
- Transition from sub clock mode to PLL clock mode
- Transition from main clock mode to PLL clock mode
- Writing "0" to the TBR bit
- Reset

Writing "1" has no effect.

Reading by read-modify-write (RMW) instructions always reads "1".

Note:

Clear the TBOF bit, after the time-base timer interrupts are disabled by the TBIE bit or the interrupt level mask register (ILM) in the processor status (PS).

[bit10] TBR

The TBR bit is used to clear all bits in the time-base timer's counter to "0". Writing "0" will clear the time-base counter. Writing "1" has no effect. "1" is always in read.

[bit9, bit8] TBC1, TBC 0

TBC1 and TBC0 are bits used to set the time-base timer's interval.

At a reset, they will be initialized to 00_B. These bits can be read and written.

Table 9.3-3 Interval Time and Cycle Count of TBC1 and TBC0

TBC1	TBC0	Interval time for source oscillation: 4 MHz	Oscillation clock (source oscillation) cycle count
0	0	1.024 ms	2^{12}cycles
0	1	4.096 ms	2^{14}cycles
1	0	16.384 ms	2^{16}cycles
1	1	131.072 ms	2^{19}cycles

9.3.3 Watch Timer Control Register (WTC)

The watch timer control register (WTC) is used to select the clock signal, control interrupts and intervals, and clear the counter.

■ Watch Timer Control Register (WTC)

Figure 9.3-4 shows the bit configuration of the watch timer control register (WTC).

Figure 9.3-4 Bit Configuration of the Watch Timer Control Register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000AA$_H$	WDCS	SCE	WTIE	WTOF	WTR	WTC2	WTC1	WTC0	WTC
Read/Write →	R/W	R	R/W	R/W	W	R/W	R/W	R/W	
Initial value →	1	0	0	0	1	0	0	0	

[bit7] WDCS

The WDCS bit is used to select the watchdog timer's clock source. If set to "0", the watch timer's clock output is selected as the watchdog timer's clock source. If set to "1", the time-base timer's clock output is selected as the watchdog timer's clock source.

This bit is initialized to "1" by reset.

Note:

If WDCS is modified, as the time-base timer and watch timer operate asynchronously with each other, the watchdog count may be shorter by 1 count. Therefore, to modify WDCS, clear the watchdog timer immediately before the clock mode is modified.

[bit6] SCE

This bit indicates that the sub clock's oscillation stabilization wait time has elapsed. This bit is set to "0" while the oscillation stabilization wait time is in progress. The oscillation stabilization wait time is fixed to 2^{14} cycles (sub clock). This bit is initialized to "0" at power-on reset and stop.

[bit5] WTIE

The WTIE bit enables interval interrupts by the watch wait timer. This bit is set to "1" to enable interrupts or set to "0" to disable them. This bit is initialized to "0" by reset. This bit can be read and written.

[bit4] WTOF

The WTOF bit is the watch timer's interrupt request flag. With the WTIE bit set to "1", if the WTOF bit is set to "1", an interrupt request is generated. The WTOF bit is set to "1" at intervals set by bits WTC2 to WTC0.

The WTOF bit is cleared by the following conditions.

- Writing "0"
- Transition to stop mode
- Reset

Writing "1" has no effect.

Reading by a read-modify-write (RMW) instruction always reads "1".

[bit3] WTR

The WTR bit is used to clear all bits in the watch timer's counter to "0". If the WTR bit is set to "0", the clock counter is cleared. Writing "1" has no effect. "1" is always read.

[bit2 to bit0] WTC2, WTC1, WTC0

The WTC2, WTC1, and WTC0 bits are used to set the watch timer's interval. Table 9.3-4 shows the settings for the interval. Reset initializes bits WTC2 to WTC0 to 000_B. These bits can be read and written.

Setting bits WTC2 to WTC0 requires setting the WTOF bit to "0".

Table 9.3-4 Selection of the Watch Timer Interval

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	1.00 s
1	1	0	2.00 s
1	1	1	4.00 s

*: The value of the interval time applies to a sub clock oscillation of 32.768 kHz.

9.4 Operation of Watchdog Timer/Time-base Timer/Watch Timer

This section describes the operation of the watchdog timer, time-base timer, and watch timer.

■ Operation of Watchdog Timer/Time-base Timer/Watch Timer

● Watchdog Timer

The watchdog timer issues a reset request if, for example, due to the program running out of control, the WTE bit in the WDTC register is not set to "0" within the specified time.

● Time-base Timer

The time-base timer provides such timer functions as acting the watchdog timer's clock source, and providing the oscillation stabilization wait time for main clock and the PLL clock. It also provides an interval interrupt function by generating interrupts at regular intervals.

● Watch Timer

The watch timer functions as a clock source for the watchdog timer, a timer for sub clock oscillation stabilization wait time, and an interval interrupt function that generates an interrupt at regular intervals.

9.4.1 Watchdog Timer Operation

The watchdog timer issues a reset request if, for example due to a program running out of control, the WTE bit in the WDTC register is not set to "0" within the specified time.

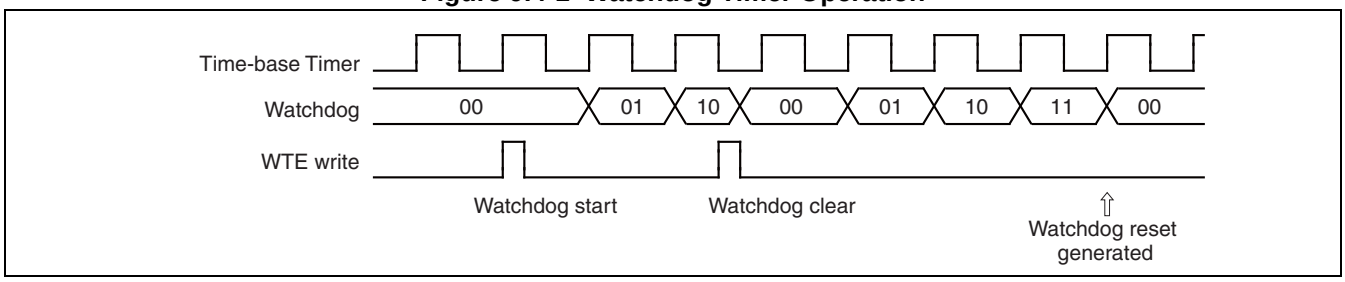
■ Method of Starting the Watchdog Timer

The watchdog timer can be started by setting the WTE bit in the WDTC register to "0" when it is stopped. At the same time, the reset generation interval of the watchdog timer is specified in the bits WT1 and WT0. Only the data at the time of the startup of the watchdog timer is valid to set the interval.

■ Prohibiting Watchdog Timer Reset

After the startup of the watchdog timer, the 2-bit watchdog counter must be regularly cleared by the program. Concretely, the WTE bit in the WDTC register must be regularly set to "0". The watchdog counter is a 2-bit counter that uses the time-base timer's carry-over signal as the clock source. Therefore, if the time-base timer is cleared, the time before generation of a watchdog reset may become longer than according to the original settings.

Figure 9.4-1 Watchdog Timer Operation



■ Watchdog Stop

The watchdog timer can be stopped by various reset sources.

■ Clearing the Watchdog Timer

The watchdog timer is cleared by writing the WTE bit as well as by reset, transition to sleep mode, stop mode, or watch mode.

In watch mode, the watchdog timer's counter is cleared and then the counting stops.

■ Checking Reset Sources

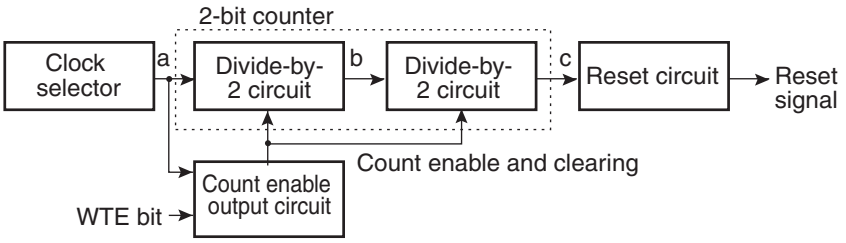
After a reset, the reset source can be determined by checking the PONR, WRST, ERST, and SRST bits in the watchdog timer control register (WDTC).

■ Interval Time of the Watchdog Timer

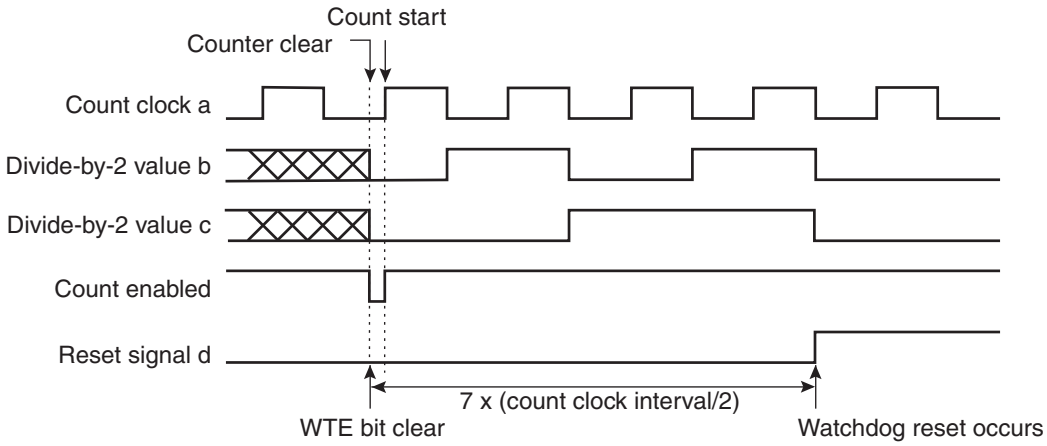
Figure 9.4-2 shows the relationship between the timing when the watchdog timer is cleared and its interval time. The interval time varies depending on the timing for clearing the watchdog timer, which requires 3.5 to 4.5 times the count clock interval.

Figure 9.4-2 Clear Timing and Interval Time of the Watchdog Timer

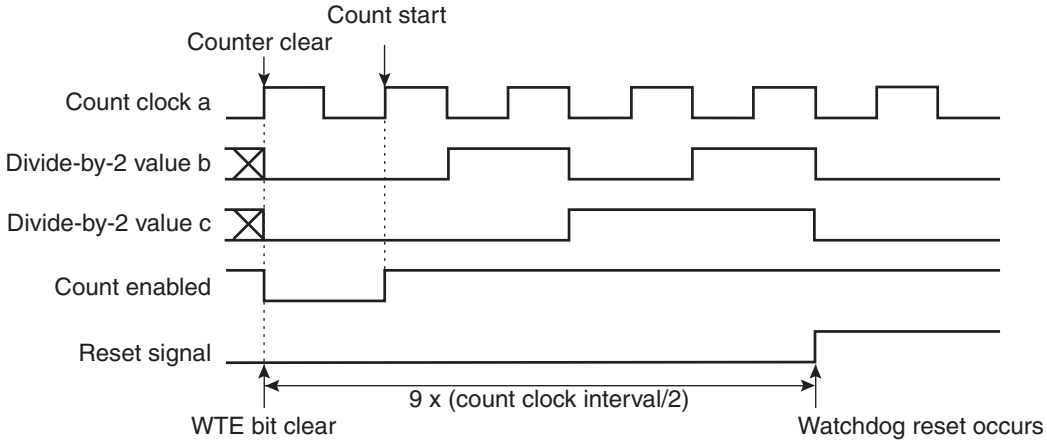
[WDG timer block diagram]



[Minimum interval time] If the WTE bit is cleared immediately before the rise of count clock



[Maximum interval time] If WTE bit is cleared immediately after the rise of count clock



9.4.2 Operation of Time-base Timer

The time-base timer provides such timer functions as acting the watchdog timer's clock source, and providing the oscillation stabilization wait time for main clock and the PLL clock. It also provides an interval interrupt function by generating interrupts at regular intervals.

■ Operation of Time-base Timer

The time-base timer consists of an 18-bit counter and uses a main clock as the count clock. While a main clock is input, count operation continues.

Time-base counter is cleared by the following conditions:

- Power-on reset
- Transition to main stop mode
- Transition to PLL stop mode
- Transition from main clock mode to PLL clock mode
- Transition from sub clock mode to main clock mode
- Transition from sub clock mode to PLL clock mode
- Setting the TBR bit in the TBTC register to "0".

The watchdog timer and interval interrupt functions, which use the output of the time-base timer, are affected by clearing the time-base timer.

■ Interval Interrupt Function

This function is used to generate interrupts at regular intervals using the time-base counter's carry-over signal. It sets the TBOF flag each time the interval set by the bits TBC0 and TBC1 in the TBTC register elapses. This flag is set based on the time when the time-base timer is finally cleared.

If a transition from main clock mode to PLL clock mode occurs, the time-base timer is cleared, since the time-base timer is used as a timer for waiting for a stabilization of the PLL clock's oscillation.

If a transition to stop mode occurs, the time-base timer is used as a timer for waiting for oscillation stabilization when operation resumes. Therefore, the TBOF flag is cleared at the same time as the mode transition.

■ Interrupts of the Time-base Timer

If the time-base timer counter counts up with the internal count clock and the bit of the selected interval timer overflows, the interrupt request flag bit (TBOF bit of the TBTC register) is set to "1". Then, if the interrupt request enable bit is set to enabled (TBIE in the TBTC register =1), an interrupt request (#34) is sent to the CPU. In the interrupt handling routine, set the TBOF bit to "0" to clear the interrupt request. In addition, the TBOF bit is set if the specified bit overflows, irrespective of the TBIE bit value.

When the TBOF bit is set to "1" and the TBIE bit is switched from "disabled" to "enabled" ("0" to "1"), an interrupt request is generated immediately.

Note:

Clear the interrupt request flag bit (TBTC:TBOF) after the time-base timer interrupts are disabled by the TBIE bit or the Interrupt level mask register (ILM) in the processor status (PS).

■ Interrupts and EI^{2}OS of Time-base Timer

Table 9.4-1 shows the time-base timer's interrupts and EI^{2}OS.

Table 9.4-1 Time-base Timer Interrupts and EI^{2}OS

Interrupt No.	Interrupt level setting register		Vector table address			EI^{2}OS
	Register name	Address	Lower	Upper	Bank	
#34 (22_H)	ICR11	$0000BC_H$	$FFFF70_H$	$FFFF71_H$	$FFFF72_H$	X

X: Not available

Notes:

- ICR11 is commonly used by time-base timer interrupts and watch timer (for sub clock) interrupts.
  The interrupt is therefore used for 2 purposes, but the interrupt level is the same.
- The time-base timer cannot use the extended intelligent I/O service (EI^{2}OS).

■ Timer Function for the Oscillation Stabilization Wait Time

The time-base timer is used as a timer for the oscillation stabilization wait time of the main clock and the PLL clock. Oscillation stabilization wait time is counted starting from a counter set to "0" (count clear) and ends when the oscillation stabilization wait time bit overflows. If, however, a return from the time-base timer mode to the PLL clock mode occurs, the time-base timer counter is not cleared and indicates a time on the way of counting. Table 9.4-2 shows the clearing of the time-base counter and the oscillation stabilization wait time.

Table 9.4-2 Clearing the Time-base Timer Counter and Oscillation Stabilization Wait Time

Operation	Counter clear	TBOF clear	Oscillation stabilization wait time
Writing "0" to TBR bit in TBTC	Y	Y	-
Power-on reset	Y	Y	Main clock oscillation stabilization wait time
Releasing main stop mode	Y	Y	Main clock oscillation stabilization wait time
Releasing PLL stop mode			
Releasing sub stop mode	N	N	Sub clock oscillation stabilization wait time
Transition from main clock mode to PLL clock mode (MCS=1 → 0)	Y	Y	PLL clock oscillation stabilization wait time
Transition from sub clock mode to main clock mode (SCS=0 → 1)	Y	Y	Main clock oscillation stabilization wait time
Transition from sub clock mode to PLL clock mode (MCS= 0, SCS= 0 → 1)	Y	Y	Main clock oscillation stabilization wait time
Releasing time-base timer mode	N	N	None
Releasing sleep mode	x	x	None

Y: Used

N: Not used

9.4.3 Operation of Watch Timer

The watch timer acts as the watchdog timer's clock source, provides a timer for the sub clock's oscillation stabilization wait time, and also provides an interval interrupt function by generating interrupts at regular intervals.

■ Operation of Watch Timer

The watch timer consists of a 15-bit counter that uses a sub clock as the count clock. While a sub oscillation clock is input, count operation continues. The watch timer is cleared by power-on reset, by a transition to stop mode, and by the writing WTR bit in the WTC register to "0".

Note:

The watchdog timer and interval interrupt, which use the watch timer's output, are affected by clearing the watch timer.
When setting the watch timer clear bit (WTR) of the watch timer control register (WTC) to "0" to clear the watch timer, perform such an operation while the interrupts of the watch timer are disabled with the WTC overflow interrupt enable bit (WTIE) set to "0".
In addition, prior to enabling the interrupts, clear the interrupt requests by setting "0" to the overflow bit (WTOF) of WTC to "0".

■ Interval Interrupt Function of Watch Timer

The interval interrupt function generates interrupts at regular intervals based on carry-over signals of the watch timer. It sets the WTOF flag at regular intervals that are specified by the WTC2 to WTC0 bits of the WTC register. The timing when the flag is set is based on the time when the watch timer is finally cleared.

In a transition to stop mode, the watch timer is used to provide a timer for the oscillation stabilization wait time before operation resumes. Therefore, the WTOF flag is also cleared at a mode transition.

■ Watch Timer Interrupts and EI^{2}OS

Table 9.4-3 shows the watch timer's interrupts and EI^{2}OS

Table 9.4-3 Watch Timer Interrupts and EI^{2}OS

Interrupt No.	Interrupt level setting register		Vector table address			EI^{2}OS
	Register name	Address	Lower	Upper	Bank	
#30 ($1E_H$)	ICR09	$0000B9_H$	$FFFF84_H$	$FFFF85_H$	$FFFF86_H$	x

X: Not available

Notes:

- The watch timer and the real-time watch timer share the same interrupt vector.
- In addition, ICR09 is commonly used by real-time watch timer interrupts and PPG timer 2 to PPG timer 5 interrupts, and the interrupt level is the same.
- The watch timer cannot use the extended intelligent I/O service (EI^{2}OS).

9.5 Notes on Using the Watchdog Timer/Time-base Timer

This section provides the notes on using the watchdog timer and the effects on peripheral functions of clearing an interrupt request when using the time-base timer and clearing the time-base timer.

■ Notes on Using the Watchdog Timer

● Stopping the watchdog timer

The watchdog timer is stopped by any of the reset sources.

● Interval time

The interval time is determined by a count clock that uses a carry-over signal of the time-base timer or watch timer. For this reason, the watchdog timer's interval time may be longer than according to the original setting when the counter as the clock source is cleared respectively. As the time-base timer can be cleared by the transition from the main clock mode to the PLL clock mode, the transition from the sub clock mode to the main clock mode, the transition from the sub clock mode to the PLL clock mode, as well as writing "0" to the time-base timer clear bit (TBR) of the time-base timer control register (TBTC), special attention is needed.

● Selection of interval time

The interval time can be set at the startup of the watchdog timer. Data written during other than the startup is ignored.

● Notes on programming

To make a program that repeatedly clears the watchdog timer in the main loop, the processing time of the main loop, including interrupt handling, cannot exceed the minimum interval time of the watchdog timer.

● Watchdog timer operation in time-base timer mode

In time-base timer mode, the watchdog timer stops while the time-base timer operates.

■ Notes on Using the Time-base Timer

● Clearing an interrupt request

Clearing the TBOF bit of the time-base timer control register must be performed via the TBIE bit or the interrupt level mask register (ILM) of the processor status register (PS) while time-base timer interrupts are masked.

● Effects of clearing the time-base timer

The following operations are affected by clearing the time-base timer's counter.

- When an interval timer function (interval interrupt) by the time-base timer is used
- When the watchdog timer is used.

- Using the oscillation stabilization wait time timer

In main stop mode, the main clock oscillation stops at power-on. Therefore, the main clock, which uses the operation clock provided from the time-base timer, requires an oscillation stabilization wait time after the oscillator has started operation. Select an appropriate oscillation stabilization wait time for the type of resonator connected to the main clock's oscillator (clock generation block). For details, see Section "5.6 Oscillation Stabilization Wait Time".

- Notes on peripheral functions whose clock is provided by the time-base timer

In the mode in which main clock oscillation stops, the counter is cleared and the time-base timer stops. In addition, the clock is provided from the time-base timer in the state after initialization if the time-base timer's counter is cleared. Therefore, the "H" level may be shortened or the "L" level may be prolonged by 1/2 an interval at maximum. Although the watchdog timer's clock is also provided in the state after initialization, the watchdog timer's counter is cleared as well, causing the watchdog timer to operate with the standard interval.

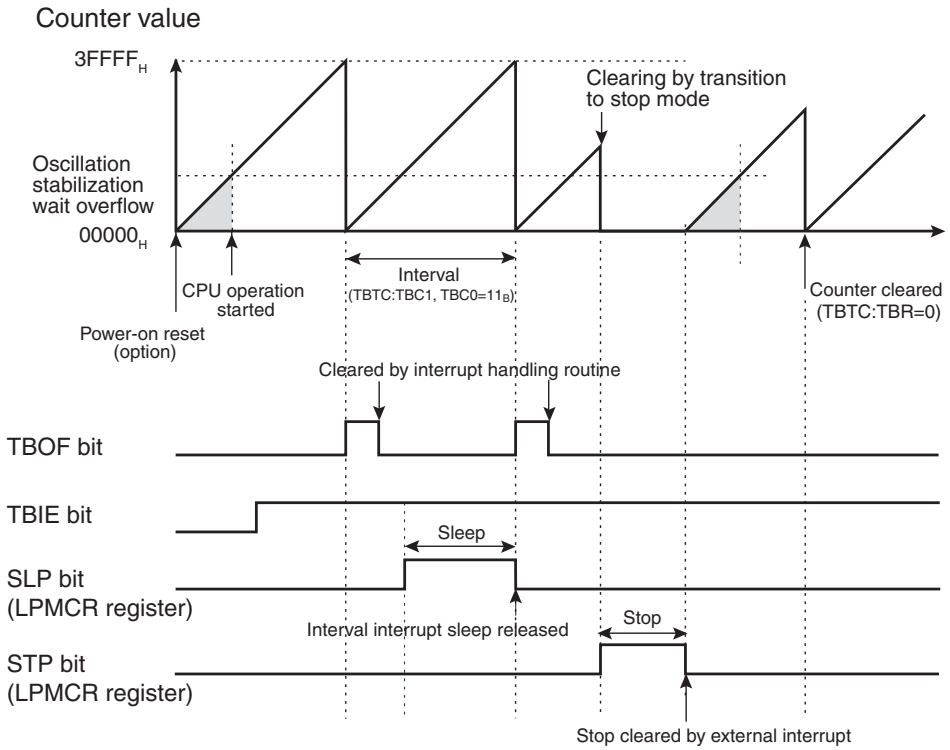
■ Operation of Time-base Timer

Figure 9.5-1 shows the operation in the following states:

- When a power-on reset occurs
- When transition to sleep mode occurs during operation of the interval timer function.
- When transition to stop mode occurs.
- When a counter clear request occurs.

Transition to stop mode clears the time-base timer, and operation stops. At the return from the stop mode, the time-base timer counts the oscillation stabilization wait time.

Figure 9.5-1 Operation of Time-base Timer



If the interval time selection bits (TBTC: TBC1, TBC0) in the time-base timer control register is set to 11_B (2^{19}/HCLK).

: Oscillation stabilization wait time

9.6 Program Example for Watchdog Timer/Time-base Timer

A program example for the watchdog timer/time-base timer is shown below.

■ Program Example for Watchdog Timer

● Specification of processing

The watchdog timer is cleared with every loop in the main program.

The main program has to go around once within the minimum interval time of the watchdog timer.

[Coding example]

```
WDTC    EQU     0000A8H             ;Watchdog timer control register
WTE     EQU     WDTC:2              ;Watchdog control bit
;---------Main program-------------------------------------------------------
CODE    CSEG
START:
;       :                           ;Stack pointer (SP) should be initialized

WDG_START:
        MOV     WDTC,  #00000011B   ;Watchdog timer started
                                    ;Interval time of 221±218 cycles selected
;---------Main loop---------------------------------------------------------
MAIN:   CLRB    I:WTE               ;Watchdog timer cleared
;       :                                   2 bits regularly cleared
;       User process
;       :
        JMP     MAIN                ;Loop by the period shorter than the
                                    ;interval time of the watchdog timer
CODE    ENDS
;---------Vector setting----------------------------------------------------
VECT    CSEG    ABS=0FFH
        ORG     0FFDCH              ;Reset vector setting
        DSL     START
        DB      OOH                 ;Set to single-chip mode
VECT    ENDS
        ENDS    START
```

■ Program Example for Time-base Timer

● Specification of processing

This program repeatedly generates interval interrupt at 2^{12}/HCLK (HCLK: oscillation clock). The interval time at this point of time is approx. 1.0 ms (at 4 MHz operation).

[Coding example]

```
ICR12   EQU     0000BCH             ;Interrupt control register for the
                                    ;time-base timer
TBTC    EQU     0000A9H             ;Time base-timer control register
TBOF    EQU     TBTC:3              ;Interrupt request flag bit
;---------Main program-------------------------------------------------------
CODE    CSEG
START:
;       :                           ;Stack pointer (SP) should be initialized
        AND     CCR,  #0BFH         ;Interrupt disabled
        MOV     I:ICR12,  #00H      ;Interrupt level 0 (highest)
        MOV     I:TBTC,  #10010000B ;Upper 3 bits are fixed
                                    ;Interrupts enabled, TBOF clear
                                    ;Counter clear
                                    ;Selection of 2^12/HCLK interval time
        MOV     ILM,  #07H          ;ILM in PS is set to level 7
        OR      CCR,  #40H          ;Interrupt enabled
LOOP:   MOV     A, #00H             ;Infinite loop
        MOV     A, #01H
        BRA     LOOP
;---------Interrupt program---------------------------------------------------
WARI:
        CLRB    I:TBOF              ;Clear interrupt request flag
;       :
;       User process
;       :
        RETI                        ;Return from interrupt
CODE    ENDS
;---------Vector setting------------------------------------------------------
VECT    CSEG    ABS=0FFH
        ORG     0FF70H              ;Setting vector to interrupt #35(23H)
        DSL     WARI
        ORG     0FFDCH              ;Reset vector setting
        DSL     START
        DB      00H                 ;Setting to single-chip mode
VECT    ENDS
        ENDS    START
```

10. Input Capture

This chapter describes the input capture operation.

10.1 Outline of Input Capture
10.2 List of Input Capture Registers
10.3 Description of Operations

10.1 Outline of Input Capture

The input capture unit consists of 1 16-bit free-run timer and 8 16-bit input captures.

■ Configuration

● Input capture (× 8)

The input capture consists of 8 independent external input pins, their corresponding capture registers and control registers. Based on detection of any edges of signals input from external input pins, the 16-bit free-run timer value can be stored in the capture register and an interrupt can be generated at the same time.

Valid edges (rising edge, falling edge and both edges) of the external input signal can be selected.

8 input captures can operate independently.

Interrupts can be generated at any valid edge of the external input signal.

● 16-bit free-run timer (× 1)

The 16-bit free-run timer consists of a 16-bit up-counter, a control register, a 16-bit compare clear register, and a prescaler. The output value of this counter is used to provide the basic time (base timer) for the input capture.

The counter operation clock can be selected from among eight types.
There are eight internal clocks: ϕ, ϕ/2, ϕ/4, ϕ/8, ϕ/16, ϕ/32, ϕ/64, ϕ/128 (ϕ : Machine clock).

An interrupt can be generated by overflows of the counter value and compare matching with the compare clear register (compare matching requires a mode setting).

The counter value is initialized to 0000_H by reset, program clear, and compare matching with the compare clear register.

10.1.1 Block Diagram of Input Capture

Input capture consists of the following blocks.

- **Block Diagram of Input Capture**

Figure 10.1-1 Block Diagram of Input Capture Unit 0

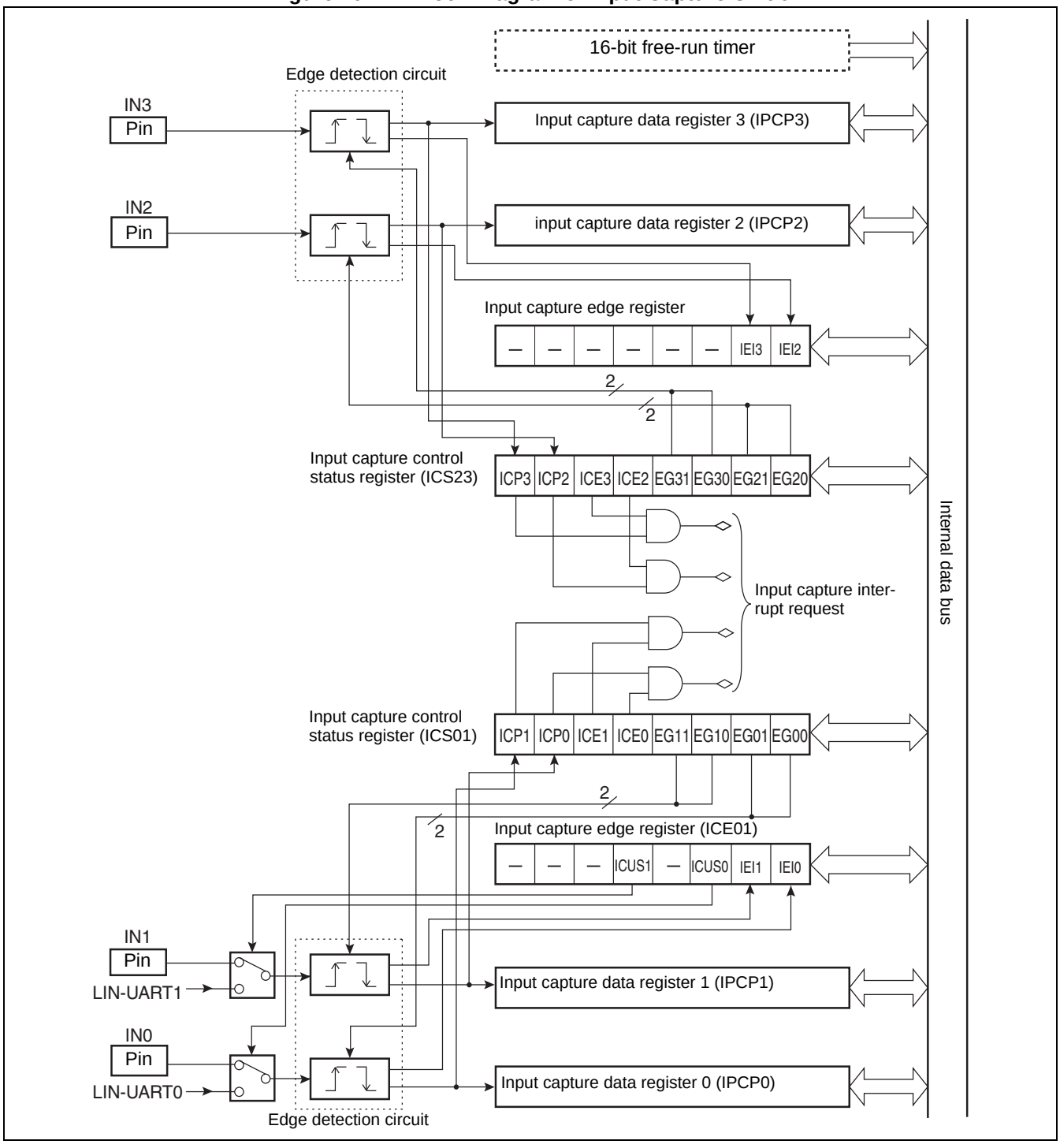
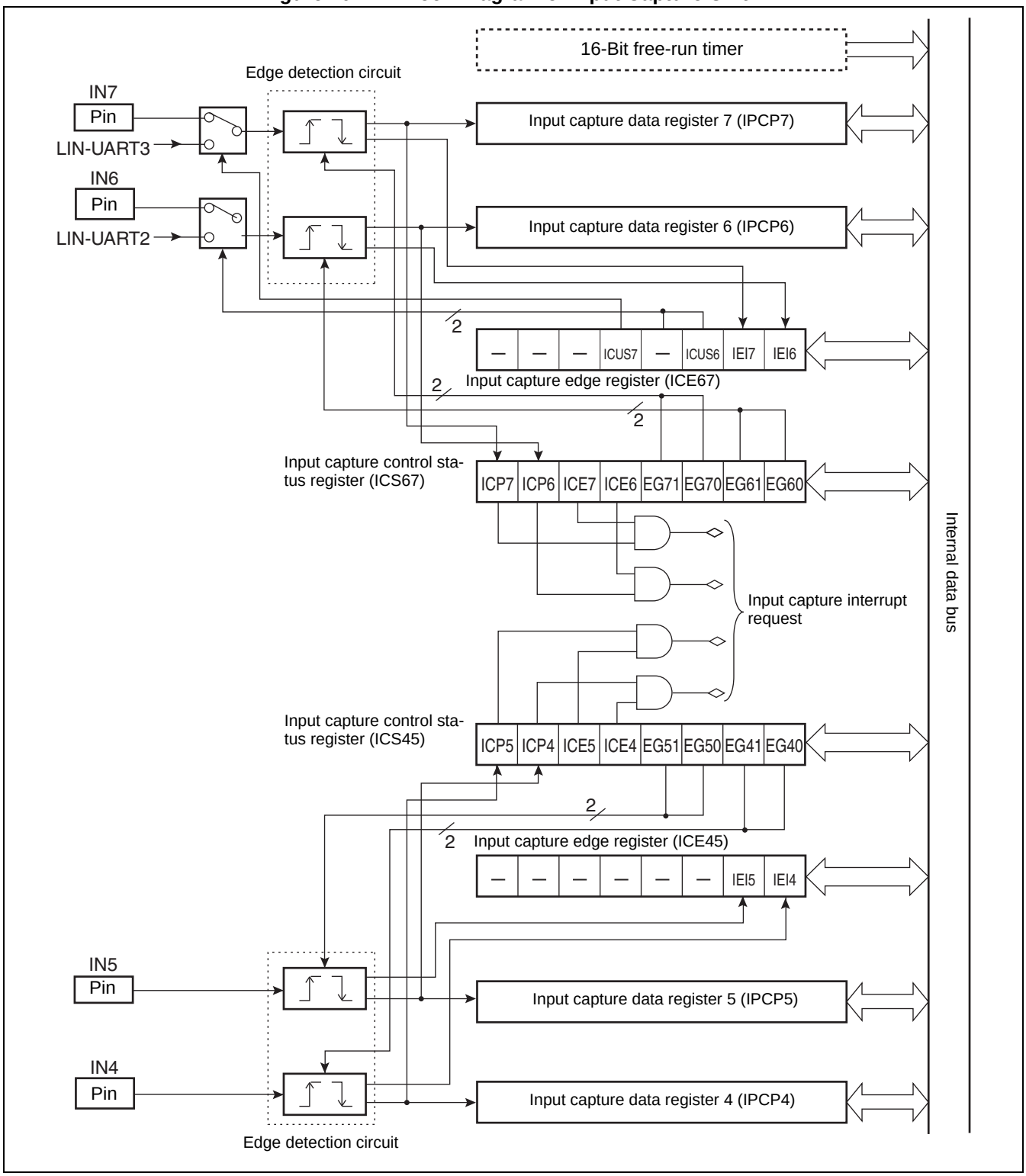


Figure 10.1-2 Block Diagram of Input Capture Unit 1



10.2 List of Input Capture Registers

This section lists the input capture registers.

■ List of Registers for 16-bit Free-run Timer Section

Figure 10.2-1 lists registers for the 16-bit free-run timer section.

Figure 10.2-1 List of Registers for 16-bit Free-run Timer Section

Compare clear register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000025_H	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	CPCLR
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Compare clear register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000024_H	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	CPCLR
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Timer data register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000027_H	T15	T14	T13	T12	T11	T10	T09	T08	TCDT
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Timer data register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000026_H	T07	T06	T05	T04	T03	T02	T01	T00	TCDT
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Timer control status register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000029_H	ECKE	Reserved	-	MSI2	MSI1	MSI0	ICLR	ICRE	TCCSH
Read/Write →	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	1	-	0	0	0	0	0	

Timer control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000028_H	IVF	IVFE	STOP	MODE	SCLR	CLK2	CLK1	CLK0	TCCSL
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

R/W: Readable/writable
X: Undefined value
-: Undefined

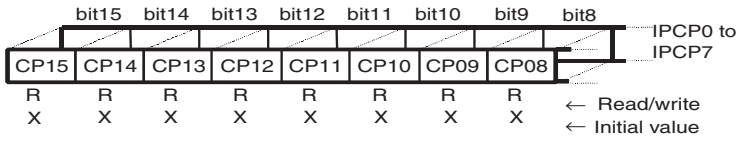
■ List of Registers for Input Capture Section

Figure 10.2-2 lists of registers for the input capture section.

Figure 10.2-2 List of Registers for Capture Section

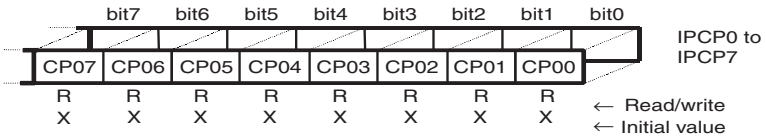
Input capture data register (Upper)

Address: ch.0 000061_H
Address: ch.1 000063_H
Address: ch.2 000065_H
Address: ch.3 000067_H
Address: ch.4 003941_H
Address: ch.5 003943_H
Address: ch.6 003945_H
Address: ch.7 003947_H



Input capture data register (Lower)

Address: ch.0 000060_H
Address: ch.1 000062_H
Address: ch.2 000064_H
Address: ch.3 000066_H
Address: ch.4 003940_H
Address: ch.5 003942_H
Address: ch.6 003944_H
Address: ch.7 003946_H



Input capture control status register (Upper)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $00006A_H$	ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20	ICS23
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000068_H	ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	ICS01
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Upper)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $0000D2_H$	ICP7	ICP6	ICE7	ICE6	EG71	EG70	EG61	EG60	ICS67
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $0000D0_H$	ICP5	ICP4	ICE5	ICE4	EG51	EG50	EG41	EG40	ICS45
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

(Continued)

(Continued)

Input capture edge register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $00006B_H$	-	-	-	-	-	-	IEI3	IEI2	ICE23
Read/Write →	-	-	-	-	-	-	R	R	
Initial value →	-	-	-	-	-	-	X	X	

Input capture edge register (Lower)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000069_H	-	-	-	ICUS1	-	ICUS0	IEI1	IEI0	ICE01
Read/Write →	-	-	-	R/W	-	R/W	R	R	
Initial value →	-	-	-	0	-	0	X	X	

Input capture edge register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $0000D3_H$	-	-	-	ICUS7	-	ICUS6	IEI7	IEI6	ICE67
Read/Write →	-	-	-	R/W	-	R/W	R	R	
Initial value →	-	-	-	0	-	0	X	X	

Input capture edge register (Lower)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $0000D1_H$	-	-	-	-	-	-	IEI5	IEI4	ICE45
Read/Write →	-	-	-	-	-	-	R	R	
Initial value →	-	-	-	-	-	-	X	X	

R/W: Readable/writable
R: Read only
W: Write only
X: Undefined value
-: Undefined

10.2.1 Detailed Description of the Input Capture Registers

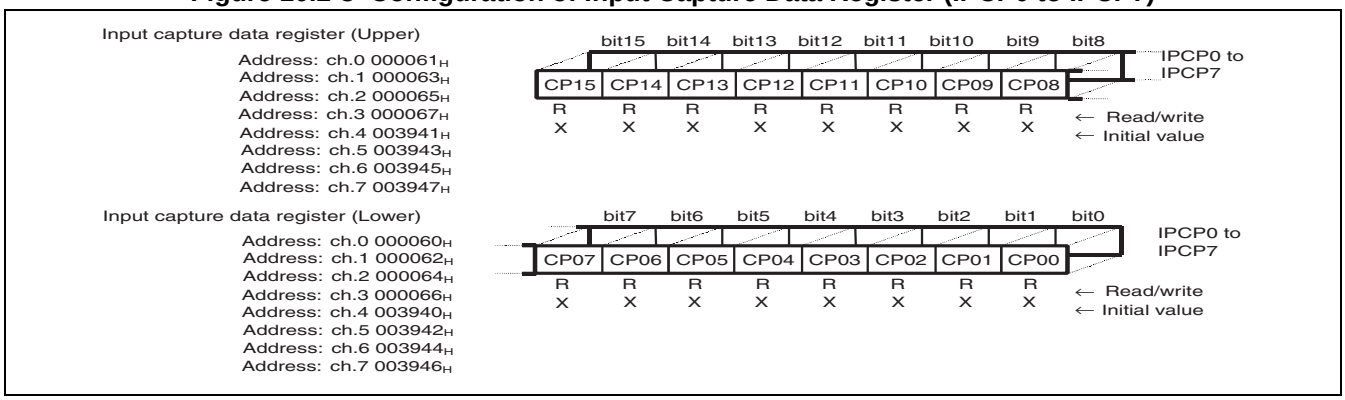
There are two types of input capture registers:

- **Input capture register (IPCP0 to IPCP7)**
- **Input capture control registers (ICS01/23/45/67)**

■ Input Capture Register (IPCP0 to IPCP7)

The IPCP register is used to store the value of the 16-bit free-run timer at detection of a valid edge of the corresponding external pin input waveform (word access is required. Writing is unavailable).

Figure 10.2-3 Configuration of Input Capture Data Register (IPCP0 to IPCP7)



■ Input Capture Control Status Register (ICS01/23/45/67)

Figure 10.2-4 Configuration of Input Capture Control Status Register (ICS01/23/45/67)

Input capture control status register (Upper)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 00006A$_H$	ICP3	ICP2	ICE3	ICE2	EG31	EG30	EG21	EG20	ICS23
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000068$_H$	ICP1	ICP0	ICE1	ICE0	EG11	EG10	EG01	EG00	ICS01
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Upper)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000D2$_H$	ICP7	ICP6	ICE7	ICE6	EG71	EG70	EG61	EG60	ICS67
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Input capture control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000D0$_H$	ICP5	ICP4	ICE5	ICE4	EG51	EG50	EG41	EG40	ICS45
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

[bit7, bit6]: ICP7 to ICP0

These are input capture interrupt flags. When any valid edge of external input pins is detected, these bits are set to "1". With the interrupt enable bits (ICE7 to ICE0) set, an interrupt can be generated at detection of a valid edge. The corresponding bits are cleared by writing "0". Writing "1" has no effect. In read-modify-write (RMW) instructions, "1" is read.

0	No valid edge detected (initial value)
1	Valid edge detected

ICPn: n corresponds to the channel number of the input capture.

[bit5, bit4]: ICE7 to ICE0

These bits are input capture interrupt enable bits. With the ICE bits set to "1", an input capture interrupt is generated if the corresponding interrupt flags (ICP7 to ICP0) are set to "1".

0	Interrupt disabled (Initial value)
1	Interrupt enabled

ICEn: n corresponds to the channel number of the input capture.

[bit3 to bit0]: EG71/EG70 to EG01/EG00

These bits are used to select the polarity of a valid edge from the external input. They are also used for enabling input capture operation.

EGn1	EGn0	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 ↑ & ↓

EGn1/EGn0: n corresponds to the channel number of the input capture.

10.2.2 Input Capture Edge Register (ICE)

The input capture edge register has a function to indicate the selected edge direction and to select whether the input signal is input from either external pin or LIN-UART. By cooperating with the LIN-UART, the baud rate measurement at the LIN slave operation can be performed.

The correspondence between ICE01 to ICE67 / channel name and input pin (UART) name is shown as follows.

ICE01: input capture ch.0, ch.1	**IN0 (/UART0)**	**IN1 (/UART1)**
ICE23: input capture ch.2, ch.3	**IN2**	**IN3**
ICE45: input capture ch.4, ch.5	**IN4**	**IN5**
ICE67: input capture ch.6, ch.7	**IN6 (/UART2)**	**IN7 (/UART3)**

■ Input Capture Edge Register (ICE)

Figure 10.2-5 Input Capture Edge Register (ICE)

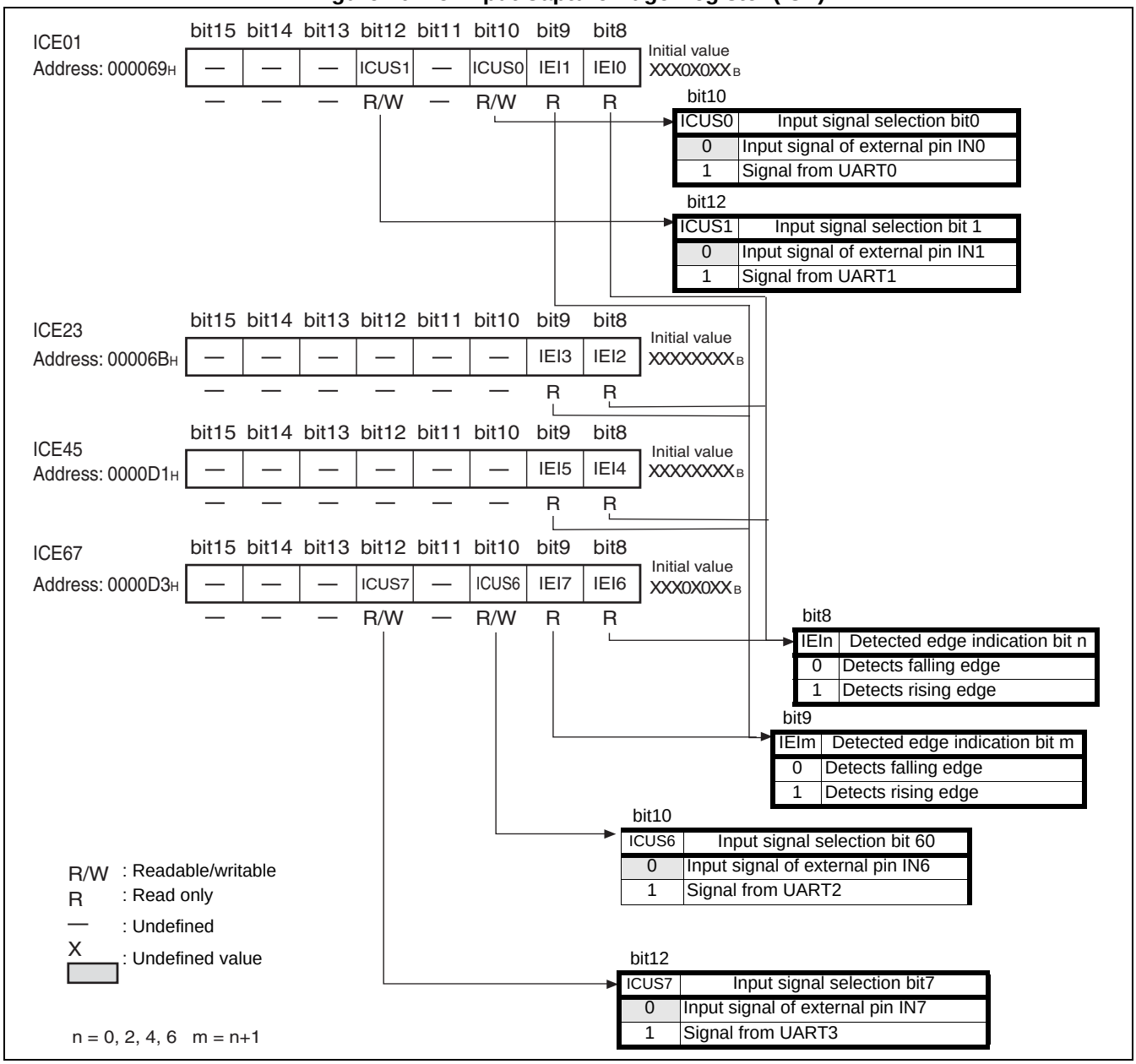


Table 10.2-1 Functions of Input Capture Edge Register 01(ICE01)

Bit name		Function
bit15 to bit13	Undefined bits	**When reading:** The value is undefined. **When writing:** No effect.
bit12	ICUS1: Input signal selection bit 1	This bit selects the input signal used as the trigger of input capture 1. **When set to "0":** Select the external pin IN1. **When set to "1":** Select the LIN-UART1
bit11	Undefined bit	**When reading:** The value is undefined. **When writing:** No effect.
bit10	ICUS0: Input signal selection bit 0	This bit selects the input signal used as the trigger of input capture 0. **When set to "0":** Select the external pin IN0. **When set to "1":** Select the LIN-UART0.
bit9	IEI1: Detected edge indication bit 1	This bit indicates the type of edges detected (rising/falling) by input capture 1. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: The value of this bit is disabled when the capture operation is stopped (ICS01: EG11, EG10=00_B).
bit8	IEI0: Detected edge indication bit0	This bit indicates the type of edge (rising/falling) detected by input capture 0. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: The value of this bit is disabled when the capture operation is stopped (ICS01: EG01, EG00=00_B).

Table 10.2-2 Functions of Input Capture Edge Register 23, 45(ICE23, ICE45)

Bit name		Function
bit15 to bit10	Undefined bits	**When reading:** The value is undefined. **When writing:** No effect.
bit9	IEI3, IEI5: Detected edge indication bit 3, 5	This bit indicates the type of edge (rising/falling) detected by input capture 3, 5. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: This bit value is disabled when the capture operation is stopped (ICSnm:EGm1, EGm0=00_B). (n = 2, 4 m = n+1)

Table 10.2-2 Functions of Input Capture Edge Register 23, 45(ICE23, ICE45)

Bit name		Function
bit8	IEI2, IEI4: Detected edge indication bit 2, 4	This bit indicates the type of edge (rising/falling) detected by input capture 2, 4. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: This bit is disabled when the capture operation is stopped. (ICSnm:EGn1, EGn0=00_B). (n = 2, 4 m = n+1)

Table 10.2-3 Functions of Input Capture Edge Register 67(ICE67)

Bit name		Function
bit15 to bit13	Undefined bits	**When reading:** The value is undefined. **When writing:** No effect.
bit12	ICUS7: Input signal selection bit 7	This bit selects the input signal used as the trigger of the input capture 7. **When set to "0":** Select the external pin IN7. **When set to "1":** Select the LIN-UART3
bit11	Undefined bit	**When reading:** The value is undefined. **When writing:** No effect.
bit10	ICUS6: Input signal selection bit 60	This bit selects the input signal used as the trigger of input capture 6. **When set to "0":** Select the external pin IN6 **When set to "1": Select the LIN-UART2**
bit9	IEI7: Detected edge indication bit 7	This bit indicates the type of edge (rising/falling) detected by input capture 7. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: The value of this bit is disabled when the capture operation is stopped (ICS67: EG71, EG70=00_B)
bit8	IEI6: Detected edge indication bit 6	This bit indicates the type of edge (rising/falling) detected by input capture 6. • This bit is read-only. **"0":** Indicates that falling edge has been detected. **"1":** Indicates that rising edge has been detected. Note: The value of this bit is disabled when the capture operation is stopped (ICS67: EG61, EG60=00_B)

Note:

For input captures 0,1,6, and 7, if the input signal is selected to the LIN-UART (ICEnm:ICUS), the

input capture is used to calculate the baud rate when the LIN-UART operates the LIN slave. In this case, it must be set to the input capture interrupt enable (ICSnm:ICEn=1 or ICEm=1) and to the detection of both edges (ICSnm:EGn1, EGn0=11_B or EGm1, EGm0=11_B). See Section "17.7.3 Operation with LIN Function (Operation Mode 3)" for details of the baud rate calculation.

n = 0, 2, 4, 6 m = n+1

10.2.3 Detailed Description of 16-Bit Free-run Timer Register

There are three types of 16-bit free-run timer registers:

- **Timer data register (TCDT)**
- **Compare clear register (CPCLR)**
- **Timer control status register (TCCSH, TCCSL)**

■ Timer Data Register (TCDT)

Figure 10.2-6 Configuration of the Timer Data Register (TCDT)

Timer data register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000027_H	T15	T14	T13	T12	T11	T10	T09	T08	TCDT
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Timer data register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000026_H	T07	T06	T05	T04	T03	T02	T01	T00	TCDT
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

This is a register that can read the count value for the 16-bit free-run timer. The counter value is cleared to 0000_H at a reset. Be sure the write operation is always performed in stop (STOP=1) state. The timer value can be set when writing to this register only in the stop (STOP=1) state. This register requires word access. The 16-bit free-run timer is initialized by the following:

- Reset
- Clearing (CLR) the timer control/status registers
- Matching the compare clear register value and timer counter value (this requires to set a mode)

■ Compare Clear Register (CPCLR)

Figure 10.2-7 Configuration of the Compare Clear Register (CPCLR)

Compare clear register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000025_H	CL15	CL14	CL13	CL12	CL11	CL10	CL09	CL08	CPCLR
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Compare clear register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000024_H	CL07	CL06	CL05	CL04	CL03	CL02	CL01	CL00	CPCLR
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

This compare clear register is a 16-bit compare register used for comparison with a 16-bit free-run timer. As the register value at initialization is not specified, enable its operation after a value has been set. This register requires word access. When "1" has been set to the MODE bit of timer control status register (TCCSH, TCCSL), if the register value matches the value of the 16-bit free-run timer value, the 16-bit free-run timer value is cleared to 0000_H. In addition, when the register value matches the value of the 16-bit free run timer, the compare clear interrupt flag is set. When the compare clear interrupt flag is "1", if interrupts are enabled, an interrupt request is issued to the CPU.

■ Timer Control Status Register (TCCSH, TCCSL)

Figure 10.2-8 Configuration of the Timer Control Status Register (TCCSH, TCCSL)

Timer control status register (Upper)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 000029_H	ECKE	Reserved	-	MSI2	MSI1	MSI0	ICLR	ICRE	TCCS
Read/Write →	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	1	-	0	0	0	0	0	

Timer control status register (Lower)

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 000028_H	IVF	IVFE	STOP	MODE	SCLR	CLK2	CLK1	CLK0	TCCS
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

[bit15]: ECKE

This bit is used to select whether to use an internal or external source for the count clock of the 16-bit free-run timer. As the clock is updated immediately after writing the ECKE bit, be sure to change this bit only while output compare and input capture are stopped.

0	Internal clock source is selected (initial value)
1	Clock is input from the external pin (FRCK)

Note:

With the internal clock selected, specify the count clock in bit2 to bit0 (CLK2 to CLK0). This count clock works as a base clock. If the clock is input from FRCK, set bit6 of DDR5 to "0".

[bit14]: Reserved bit

"1" is always set at writing. "1" is always read.

[bit13]: Undefined bit

Read value is undefined. Writing has no effect on operation.

[bit12 to bit10]: MSI2 to MSI0

These bits are used to set the count for masking compare clear interrupts. They are set with the 3-bit reload counter: Every time the counter value is set to 000_B, the count value is reloaded. In addition, the counter value is loaded when writing to this register. The masking count = the set count (For example, for masking twice and throwing an interrupt the third time, set these bits to 010_B). However, note that setting these bits to 000_B does not mask interrupt sources.

[bit9]: ICLR

This bit is an interrupt request flag for compare clear. The ICLR bit is set to "1" if the compare clear register value and the value of the 16-bit free-run timer value match. An interrupt occurs if the interrupt request enable bit (bit8: ICRE) is set. The ICLR bit is cleared by writing "0". Writing "1" has no effect. "1" is always read by a read-modify-write (RMW) instruction.

0	No interrupt request (Initial value)
1	Interrupt requested

[bit8]: ICRE

This bit is an interrupt enable bit for compare clear. If, with the ICRE bit set to "1", the interrupt flag (bit9: ICLR) is set to "1", an interrupt is generated.

0	Interrupt disabled (Initial value)
1	Interrupt enabled

[bit7]: IVF

This bit is an interrupt request flag for the 16-bit free-run timer. If the 16-bit free-run overflows, the IVF bit is set to "1". If the interrupt request enable bit (bit6: IVFE) is set, an interrupt is generated. The IVF bit is cleared by writing "1". Writing "1" has no effect. "1" is always read by a read-modify-write (RMW) instruction.

0	No interrupt request (Initial value)
1	Interrupt requested

[bit6]: IVFE

This bit is an interrupt enable bit for the 16-bit free-run timer. If, with the IVFE bit set to "1", the interrupt flag (bit7: IVF) is set to "1", an interrupt is generated.

0	Interrupt disabled (Initial value)
1	Interrupt enabled

[bit5]: STOP

This bit is used to stop counting the 16-bit free-run timer. Writing "1" will stop counting the timer. Writing "0" starts counting the timer.

If the 16-bit free-run timer stops, the output compare operation also stops.

0	Count enabled (operation) (initial value)
1	Count disabled (stopped)

[bit4]: MODE

This bit is used to set the initialization condition of the 16-bit free-run timer. If this bit is set to "0", the counter is initialized by reset and the clear bit (bit3:SCLR). If this bit is set to "1", the counter can be initialized by matching with the compare clear register (CPCLR) value in addition to reset and the clear bit (bit3:SCLR).

The counter value is initialized when the count value changes.

0	Initialized by reset or the clear bit (initial value)
1	Initialized by reset, the clear bit, or the compare clear register

[bit3]: SCLR

This bit is used to initialize the 16-bit free-run timer to 0000_H during operation. Writing "1" initializes the counter to 0000_H. Writing "0" has no effect. The read value is always "0". The counter value is initialized when the count value changes.

To initialize in timer stop mode, write 0000_H to the data register.

SCLR	Meaning of flag
0	No meaning (initial value)
1	Counter initialized to 0000_H

Note:

Writing "0" to this bit before the next count clock cycle after writing "1" prevents the counter value value from being initialized.

[bit2 to bit0]: CLK2 to CLK0

These bits are used to select a count clock for the 16-bit free-run timer. Because the clock changes immediately after setting the CLK bit, change the bit only when output compare and input capture are stopped.

CLK2	CLK1	CLK0	Count clock	ϕ=32MHz	ϕ= 8MHz	ϕ= 4MHz	ϕ= 1MHz
0	0	0	ϕ	31.25 ns	125 ns	0.25 µs	1 µs
0	0	1	ϕ/2	62.5 ns	0.25 µs	0.5 µs	2 µs
0	1	0	ϕ/4	0.125 µs	0.5 µs	1 µs	4 µs
0	1	1	ϕ/8	0.25 µs	1 µs	2 µs	8 µs
1	0	0	ϕ/16	0.5 µs	2 µs	4 µs	16 µs
1	0	1	ϕ/32	1 µs	4 µs	8 µs	32 µs
1	1	0	ϕ/64	2 µs	8 µs	16 µs	64 µs
1	1	1	ϕ/128	4 µs	16 µs	32 µs	128 µs

ϕ=Machine clock

10.3 Description of Operations

This section describes the operations of the input capture.

■ Description of Operations

● 16-Bit Free-run Timer

The 16-bit free-run timer starts counting the counter value from 0000_H when a reset has been released. This counter value is used as a reference time for the 16-bit output compare and 16-bit input capture.

● 16-bit input capture

The 16-bit input capture can generate an interrupt after fetching a 16-bit free-run timer value into the capture register upon detection of the specified valid edge.

■ Input Capture Interrupts and EI^{2}OS

Table 10.3-1 shows input capture interrupts and EI^{2}OS

Table 10.3-1 Input Capture Interrupts and EI^{2}OS

Channel	Interrupt No.	Interrupt level setting register		Vector table address			EI^{2}OS
		Register Name	Address	Lower	Upper	Bank	
Input capture 0	#15 ($0F_H$)	ICR02	$0000B2_H$	$FFFFC0_H$	$FFFFC1_H$	$FFFFC2_H$	*
Input capture 1	#19 (13_H)	ICR04	$0000B4_H$	$FFFFB0_H$	$FFFFB1_H$	$FFFFB2_H$	*
Input capture 2	#21 (15_H)	ICR05	$0000B5_H$	$FFFFA8_H$	$FFFFA9_H$	$FFFFAA_H$	*
Input Capture 3 to 7	#23 (17_H)	ICR06	$0000B6_H$	$FFFFA0_H$	$FFFFA1_H$	$FFFFA2_H$	*
Free-run timer Over-flow Free-run timer clear	#31 ($1F_H$)	ICR10	$0000BA_H$	$FFFF80_H$	$FFFF81_H$	$FFFF82_H$	×

× : Not available

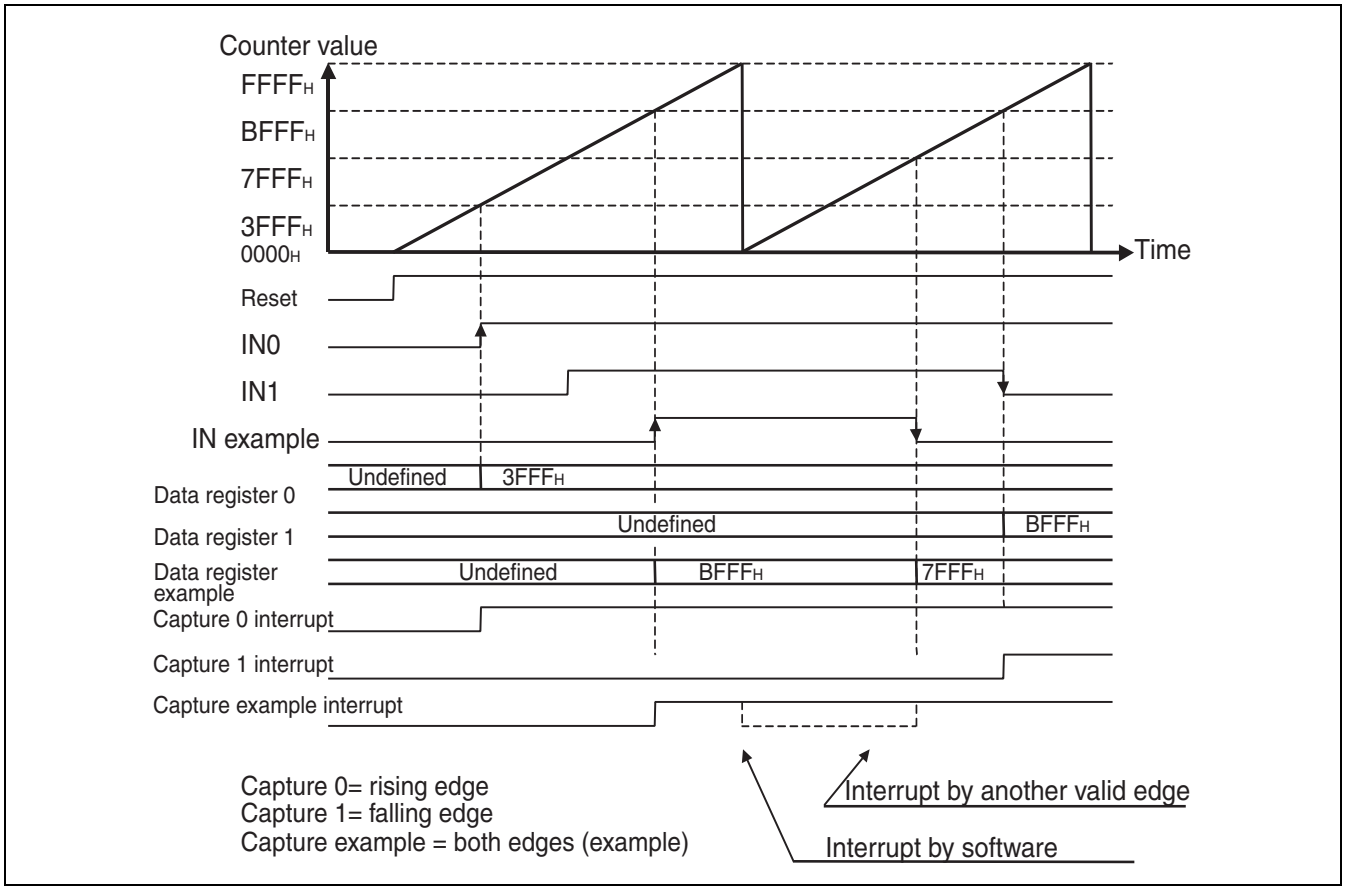
* : Available when not using interrupt sources sharing ICR02, ICR04, ICR05, ICR06, and the interrupt vector.

10.3.1 16-bit Input Capture

The 16-bit input capture can generate an interrupt after fetching a 16-bit free-run timer value into the capture register upon detection of the specified valid edge.

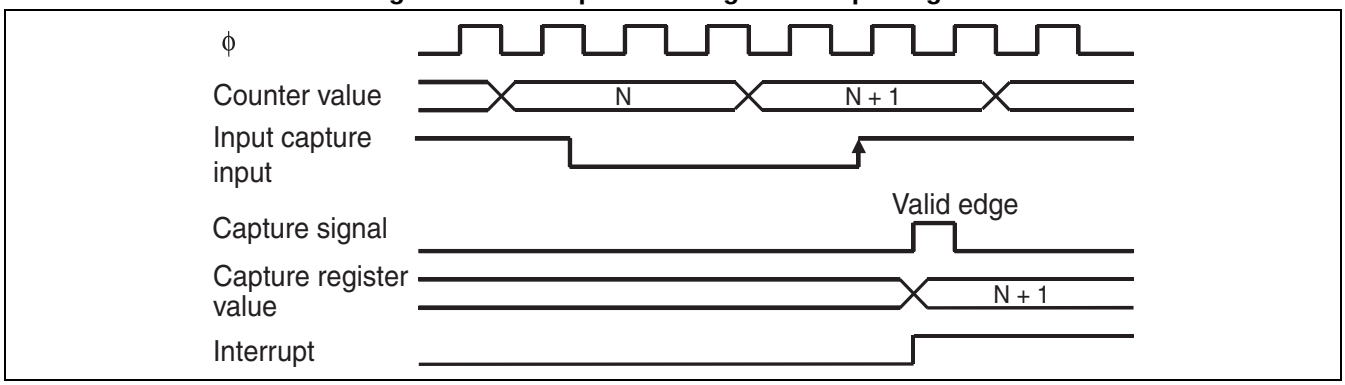
■ Operation of 16-bit Input Capture

Figure 10.3-1 Example of Timing for Fetching Input Captures



■ Input Timing for a 16-bit Input Capture

Figure 10.3-2 Capture Timing for an Input Signal



10.3.2 16-bit Free-run Timer

The 16-bit free-run timer starts counting the counter value from 0000_H when a reset has been released. This counter value is used as a reference time for the 16-bit output compare and 16-bit input capture.

■ Operations of 16-bit Free-run Timer

The counter value is cleared in the following conditions:

- Overflow occurs
- Compare-match with compare clear register value successful (this requires to set a mode)
- Setting the SCLR bit of the TCCSH, TCCSL registers to "1" during operation
- Writing 0000_H to TCDT in timer stop mode

An interrupt may occur if an overflow is generated or if the compare clear register value is compared and matched with the free-run timer (for a compare match interrupt, a mode setting is required).

Figure 10.3-3 Clearing the Counter at an Overflow

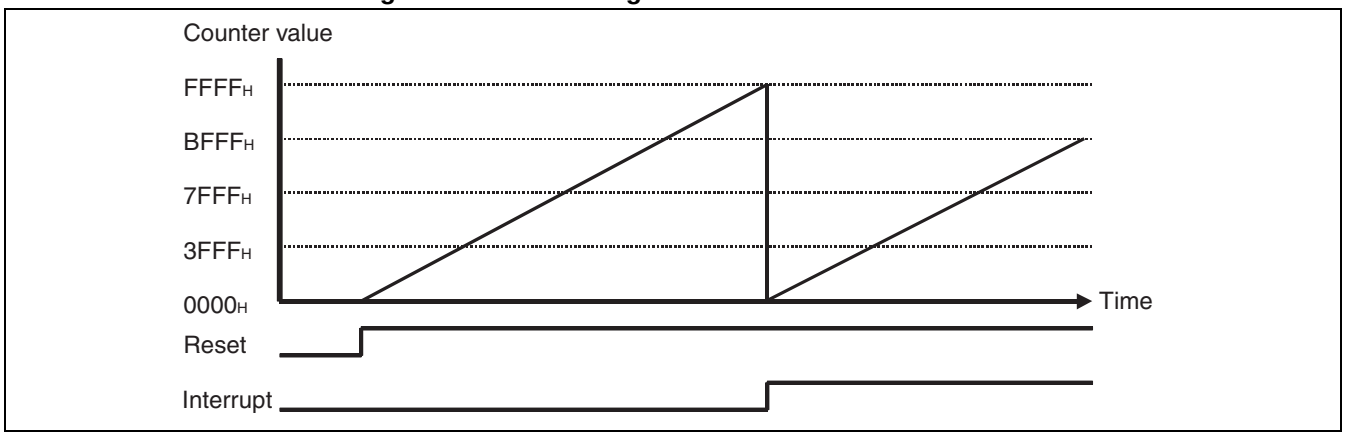
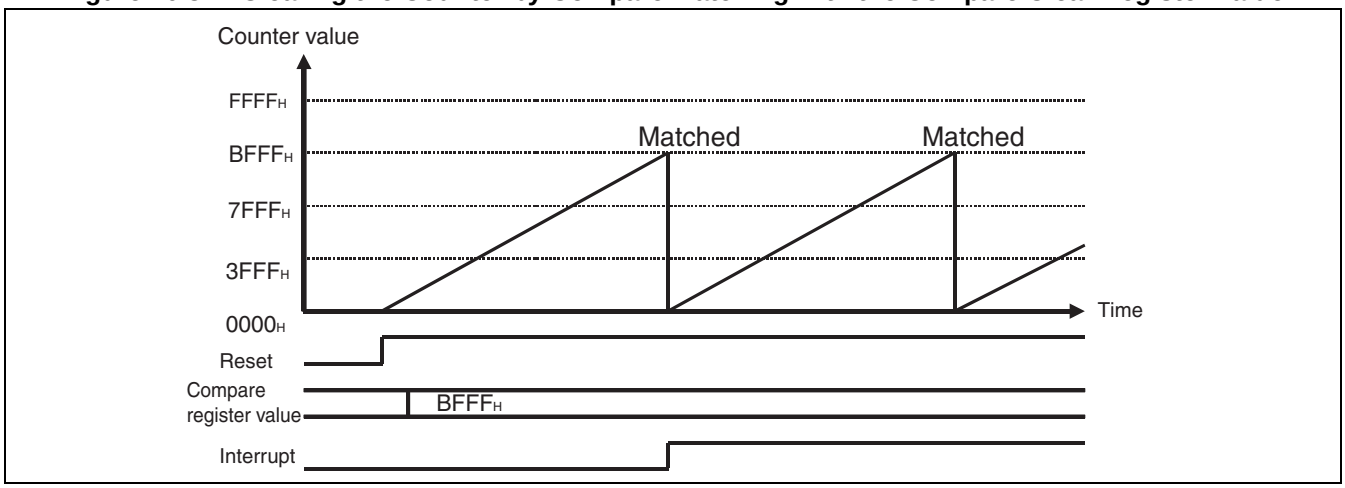


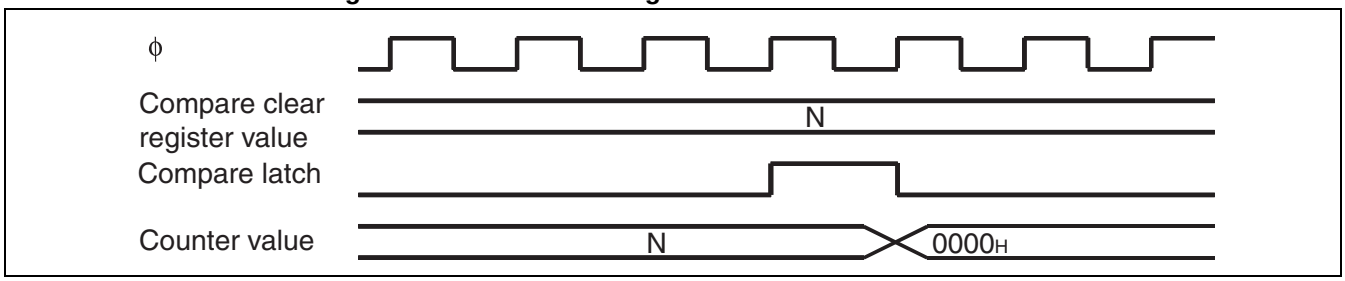
Figure 10.3-4 Clearing the Counter by Compare Matching with the Compare Clear Register Value



■ Clear Timing for 16-bit Free-run Timer

The counter is cleared by reset, by software, and by matching with the compare clear register. Counter clearing by reset is performed as soon as the clear source occurs, while counter clearing by matching with the compare clear register or by software is performed after synchronizing with the count timing.

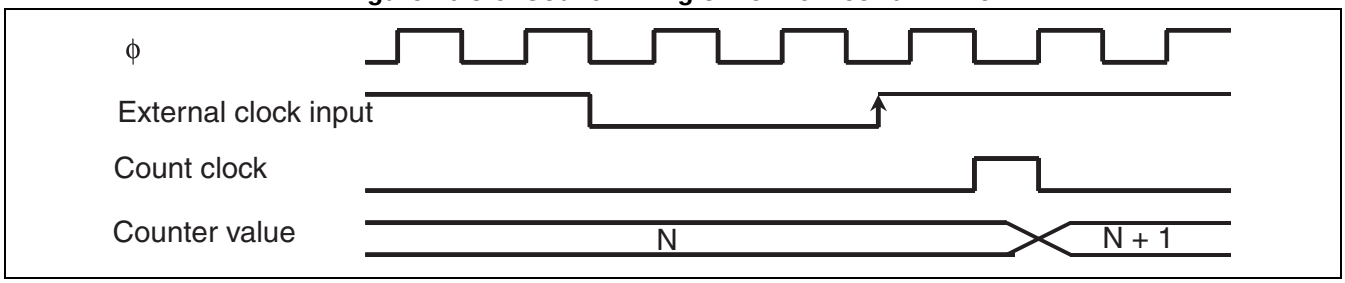
Figure 10.3-5 Clear Timing for the 16-bit Free-run Timer



■ Count Timing of 16-bit Free-run Timer

The 16-bit free-run timer counts up using the clock (internal or external clock) input. If an external clock is selected, counting occurs at the rising edge.

Figure 10.3-6 Count Timing of 16-Bit Free-run Timer



11. 16-Bit Reload Timer

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

11.1 Overview of 16-bit Reload Timer
11.2 Configuration of 16-bit Reload Timer
11.3 Pins of 16-bit Reload Timer
11.4 Registers of 16-bit Reload Timer
11.5 Interrupts of 16-bit Reload Timer
11.6 Operation of 16-bit Reload Timer
11.7 Notes on Using 16-bit Reload Timer
11.8 Sample Program for 16-bit Reload Timer

11.1 Overview of 16-bit Reload Timer

The 16-bit reload timer has two modes: Internal clock mode (with countdown performed in synchronization with three types of internal clock), and event count mode (with countdown performed by detecting any pulse edge input to the external pin. Either mode may be selected. This timer defines an underflow when the counter value is in the range from 0000_H to $FFFF_H$. Therefore, an underflow occurs at a count of [reload register's setting value + 1].
Counter operation can be selected either reload mode in which an underflow causes the count set value to be reloaded for repeated counting, or one shot mode in which counting is stopped when an underflow occurs. Counter underflow may generate an interrupt and supports the extended intelligent I/O service (EI^2OS).

■ Operation Mode of 16-bit Reload Timer

Table 11.1-1 lists the operation modes of the 16-bit reload timer.

Table 11.1-1 Operation Mode of 16-bit Reload Timer

Clock mode	Counter operation	16-bit reload timer operation
Internal clock mode	Reload mode	Software trigger operation External trigger operation External gate input operation
	One shot mode	
Event count mode (External clock mode)	Reload mode	Software trigger operation
	One shot mode	

■ Internal Clock Mode

One type of count clock can be selected among three types of internal clock to operate as follows.

- Software trigger operation

Writing "1" to TRG bit of the timer control status register (TMCSR0 to TMCSR3) starts count operation. Trigger input by the TRG bit is also enabled for external trigger input and external gate input.

- External trigger operation

Starts count operation when the selected edge (rising, falling, or both) is input to the TIN0/TIN1/TIN2/TIN3 pin.

- External gate input operation

Continues counting while the signal level selected ("L" or "H") is being input to the TIN0/TIN1/TIN2/TIN3 pin.

■ Event Count Mode (External Clock Mode)

Event count mode is a function to start countdown at the selected valid edge (rising, falling, or both) when the edge is input to the TIN0/TIN1/TIN2/TIN3 pin. It is also used as an interval timer when using an external clock with a constant interval.

■ Count Operation

● Reload mode

If the countdown causes an underflow (0000_H –> $FFFF_H$), the setting value for counting is reloaded and count operation is continued. Since an underflow can generate an interrupt request, it can be used as an interval timer. Also, a toggled waveform, which reverses itself at every underflow, is output from the TOT0/TOT1/TOT2/TOT3 pin. Table 11.1-2 lists the interval time for the 16-bit reload timer.

Table 11.1-2 Interval Time of 16-bit Reload Timer

Count clock	Count clock cycle	Interval time
Internal clock	$2^1/\phi$ (0.0625 μs)	0.0625 μs to 4.096 ms
	$2^3/\phi$ (0.25 μs)	0.25 μs to 16.384 ms
	$2^5/\phi$ (1.0 μs)	10.0 μs to 65.54 ms
External clock	$2^3/\phi$ or more (0.5 μs)	0.25 μs or more

ϕ: Machine clock () indicates the value for 32MHz machine clock operation.

● One shot mode

If countdown leads to an underflow (0000_H –> $FFFF_H$), the count operation will be stopped. An underflow can generate an interrupt. During counter operation, the square wave that indicates counting can be output from the TOT0, TOT1, TOT2, and TOT3 pins.

Reference:

- The 16-bit reload timer can be used for creating the UART baud rate.
- The 16-bit reload timer can be used as an activation trigger for A/D converter.

■ Interrupts and EI^{2}OS of 16-bit Reload Timer

Table 11.1-3 lists the interrupts and EI^{2}OS from the 16-bit reload timer.

Table 11.1-3 Interrupts and EI^{2}OS of 16-bit Reload Timer

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
16-bit reload timer 0	#17 (11_H)	ICR03	$0000B3_H$	$FFFFB8_H$	$FFFFB9_H$	$FFFFBA_H$	△
16-bit reload timer 1	#28 ($1C_H$)	ICR08	$0000B8_H$	$FFFF8C_H$	$FFFF8D_H$	$FFFF8E_H$	△
16-bit reload timer 2	#18 (12_H)	ICR03	$0000B3_H$	$FFFFB4_H$	$FFFFB5_H$	$FFFFB6_H$	△
16-bit reload timer 3	#22 (16_H)	ICR05	$0000B5_H$	$FFFFA4_H$	$FFFFA5_H$	$FFFFA6_H$	△

: Available when not using interrupt source sharing ICR03, ICR08, ICR05, and interrupt vector.

11.2 Configuration of 16-bit Reload Timer

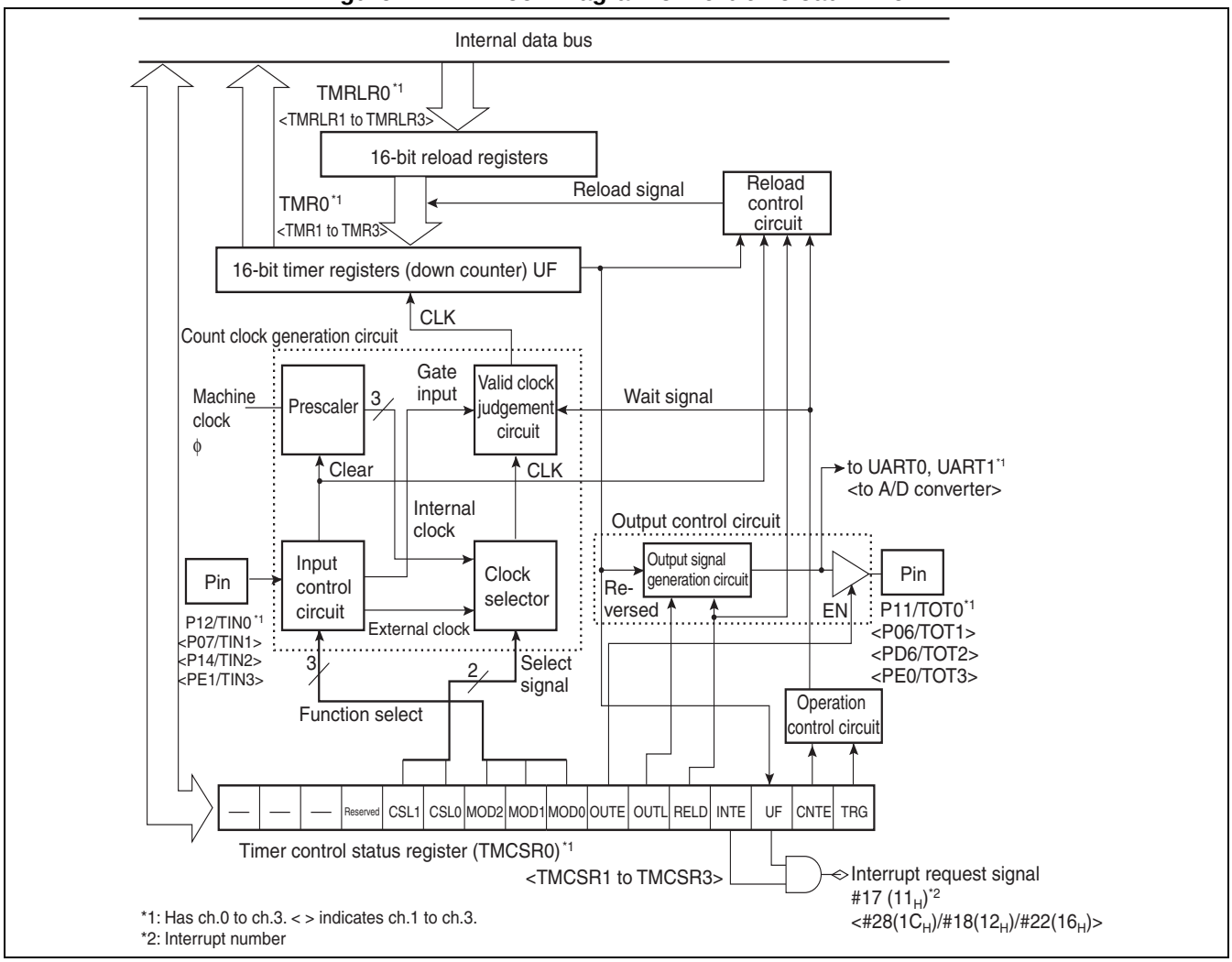
The 16-bit reload timer consists of the following 7 blocks:

- **Count clock generator circuit**
- **Reload control circuit**
- **Output control circuit**
- **Operation control circuit**
- **16-bit timer registers (TMR0 to TMR3)**
- **16-bit reload registers (TMRLR0 to TMRLR3)**
- **Timer control status registers (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H)**

■ Block Diagram of 16-bit Reload Timer

Figure 11.2-1 shows a block diagram of the 16-bit reload timer.

Figure 11.2-1 Block Diagram of 16-bit Reload Timer



- Count clock generator circuit

The count clock generator circuit generates the count clock for the 16-bit reload timer from the machine clock or external input clock.

- Reload control circuit

Controls reload operation when the timer starts and when underflow occurs.

- Output control circuit

Controls the reversal of TOT pin output due to 16-bit timer register underflow and the enable/disable state of TOT pin output.

- Operation control circuit

Controls starting/stopping of the 16-bit reload timer.

- 16-bit timer registers (TMR0 to TMR3)

These are the 16-bit down counter. The current counter value is read from these registers.

- 16-bit reload registers (TMRLR0 to TMRLR3)

These registers are used to set the interval time of the 16-bit reload timer. The set value in these registers is loaded into the 16-bit timer registers for countdown.

- Timer control status registers (TMCSR0L to TMCSR3L/TMCSR0H to TMCSR3H)

These registers are used to select the count clock and operation mode, set operating conditions, activate a trigger by software, enable/disable count operation, select reload/one shot mode, select the pin output level, enable/disable timer output, control interrupts, and check the state of operation of the 16-bit reload timer.

11.3 Pins of 16-bit Reload Timer

This section shows the pins of the 16-bit reload timer and their block diagram.

■ Pins of 16-bit Reload Timer

The pins of 16-bit reload timer can also be used for general-purpose ports. Table 11.3-1 shows the pin functions, I/O type, and settings for using the 16-bit reload timer.

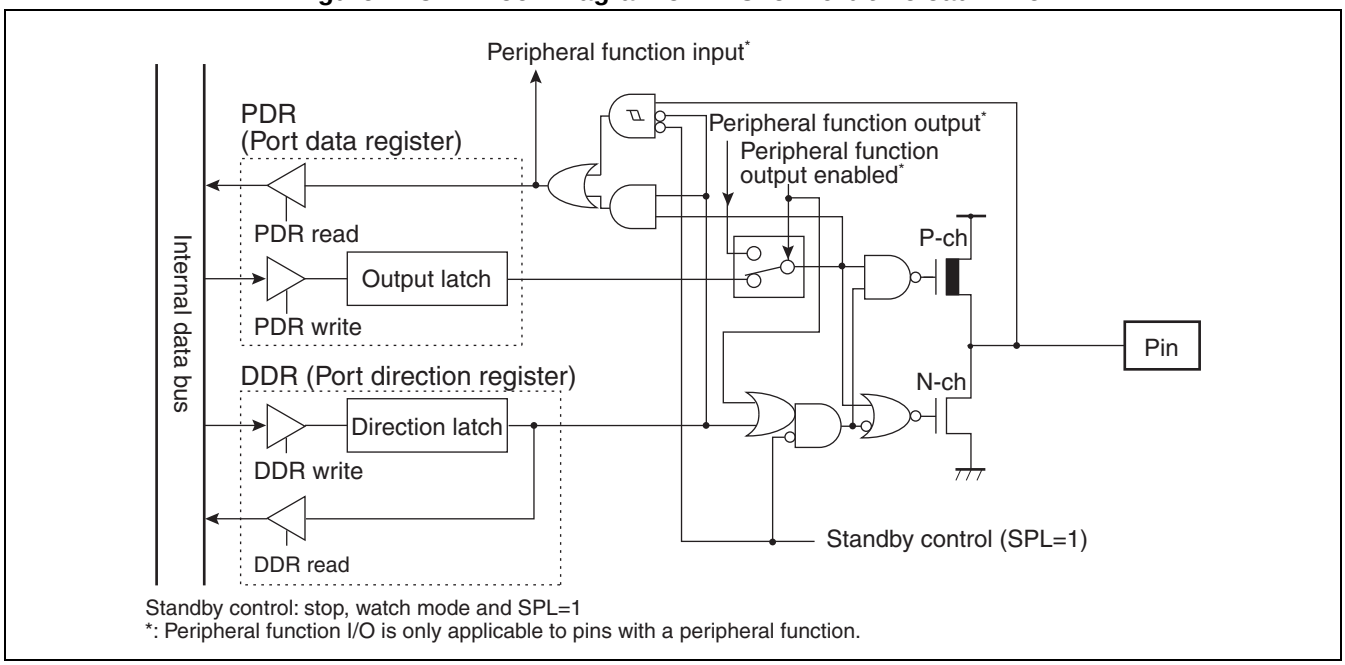
Table 11.3-1 Pins of 16-bit Reload Timer

Pin name	Pin function	I/O type	Pull-up selection	Standby control	Setting to use pin
P12/TIN0/PPG4	I/O of Port 1/Timer input	CMOS output/ CMOS hysteresis input	No	Yes	Sets to input port (DDR1:bit2=0) PPG4 output disabled
P11/TOT0/PPG3/IN4	I/O of Port 1/Timer output				Sets to timer output enabled (TMCSR0L:OUTE=1) PPG3 output disabled
PC7/PPG1/TIN1/IN6	I/O of Port C/Timer input				Sets to input port (DDRC:bit7=0) PPG1 output disabled
PC6/PPG0/TOT1/IN7	I/O of Port C/Timer output				Sets to timer output enabled (TMCSR1L:OUTE=1) PPG0 output disabled
P14/TIN2/IN1	I/O of Port 1/Timer input				Sets to input port (DDR1:bit4=0)
PD6/TOT2	I/O of Port D/Timer output				Sets to timer output enabled (TMCSR2L:OUTE=1)
PE1/TIN3	I/O of Port E/Timer input				Sets to input port (DDRE:bit1=0)
PE0/TOT3	I/O of Port E/Timer output				Sets to timer output enabled (TMCSR3L:OUTE=1)

■ Block Diagram of Pins for 16-bit Reload Timer

Figure 11.3-1 shows a block diagram of the pins for the 16-bit reload timer.

Figure 11.3-1 Block Diagram of Pins for 16-bit Reload Timer



11.4 Registers of 16-bit Reload Timer

This section lists the registers of the 16-bit reload timer.

■ Register List of 16-bit Reload Timer

Figure 11.4-1 lists the registers of the 16-bit reload timer.

Figure 11.4-1 Register List of 16-Bit Reload Timer

	Address	bit15 ····· bit8	bit7 ····· bit0
16-bit reload timer 0	000051H,000050H	TMCSR0L, TMCSR0H (Timer control status register)	
	000053H,000052H	TMR0/TMRLR0 (16-bit timer register/16-bit reload register)*	
16-bit reload timer 1	000055H,000054H	TMCSR1L, TMCSR1H (Timer control status register)	
	000057H,000056H	TMR1/TMRLR1 (16-bit timer register/16-bit reload register)*	

	Address	bit15 ····· bit8	bit7 ····· bit0
16-bit reload timer 2	0000D5H,0000D4H	TMCSR2L, TMCSR2H (Timer control status register)	
	003951H,003950H	TMR2/TMRLR2 (16-bit timer register/16-bit reload register)*	
16-bit reload timer 3	0000D7H,0000D6H	TMCSR3L, TMCSR3H (Timer control status register)	
	003953H,003952H	TMR3/TMRLR3 (16-bit timer register/16-bit reload register)*	

*: Functions as a 16-bit timer register (TMR) for reading, and as a 16-bit reload register (TMRLR) for writing.

11.4.1 Timer Control Status Registers (Upper) (TMCSR0H to TMCSR3H)

Upper bit11 to bit8 and lower bit7 in the timer control status registers (TMCSR0 to TMCSR3) have functions to select the operation mode and set the operating condition of the 16-bit reload timer. The lower bit7 (MOD0 bit) is described here.

■ Timer Control Status Registers, Upper (TMCSR0H to TMCSR3H)

Figure 11.4-2 Timer Control Status Registers, Upper (TMCSR0H to TMCSR3H)

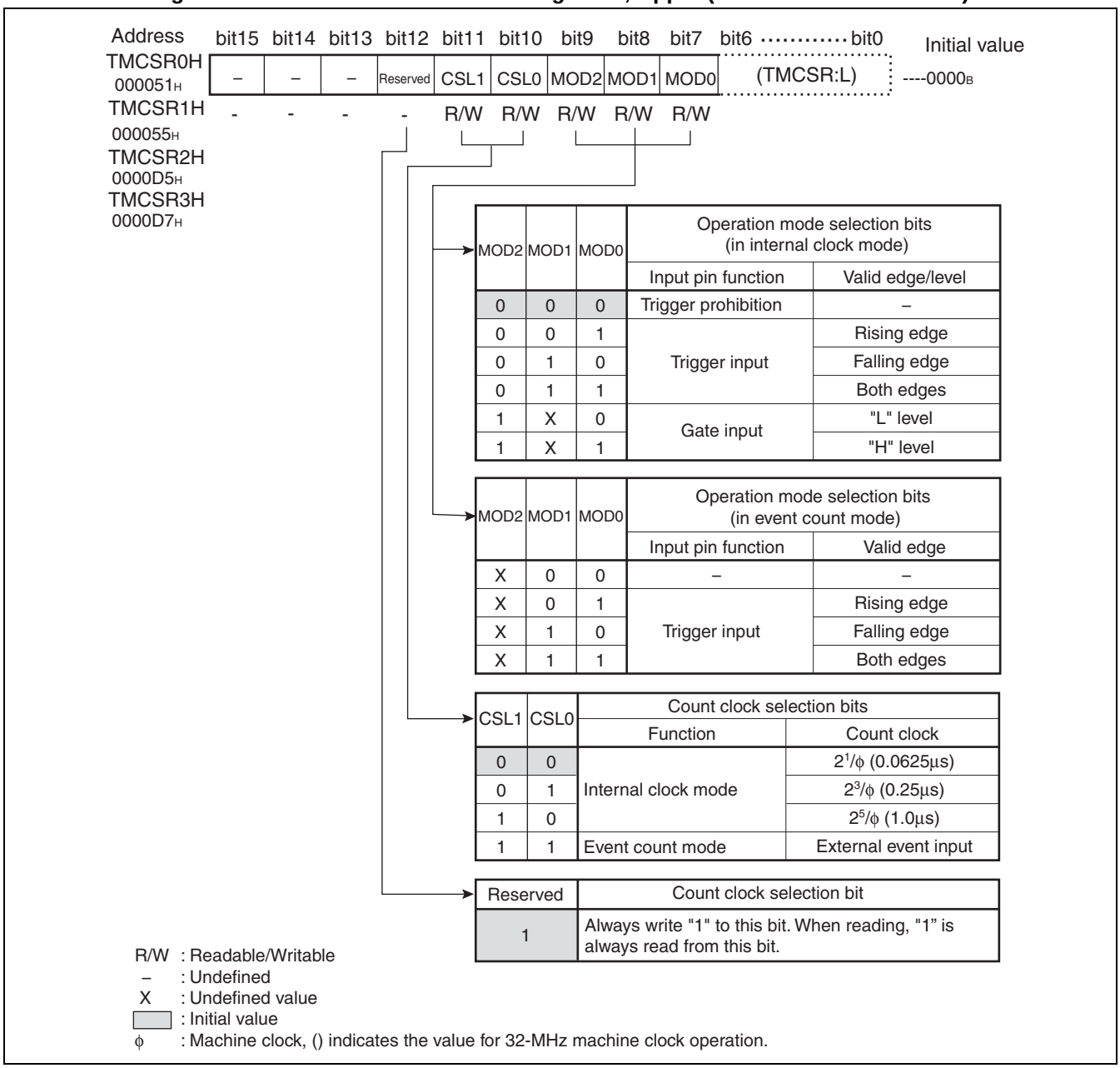


Table 11.4-1 Functions of Upper Bits of Timer Control Status Registers (TMCSR0H to TMCSR3H)

Bit name		Function
bit15 to bit13	Undefined bits	• Value at reading is not defined. • Writing does not affect operation.
bit12	Reserved bit	When writing, "1" is always set to this bit. When reading, "1" is always read from this bit.
bit11, bit10	CSL1, CSL0: Count clock selection bits	• Select the count clock. • When these bits are values other than 11_B, internal clock mode is set to count the internal clock. • When these bits are 11_B, event count mode is set to count the external clock edge.
bit9 to bit7	MOD2, MOD1, MOD0: Operation mode selection bits	Internal clock mode: • The MOD2 bit is used to select the function of the input pin. • When the MOD2 bit is "0", the input pin is used as trigger input pin. When a valid edge is input, the value of the reload register is loaded into the counter to continue with count operation. The MOD1 and MOD0 bits are used to select the type of the valid edge. • When the MOD2 bit is "1", the input pin is set as a gate input and counts only while a valid level is being input. • Since the value of the MOD1 bit has no effect, any value ("0" or "1") can be set. • The MOD0 is used to select the valid level. Event count mode: • Since the value of the MOD2 bit has no effect, any value ("0" or "1") can be set. • The input pin is used as a trigger input pin for event input. The MOD1 and MOD0 bits are used to select a valid edge.

11.4.2 Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L)

Lower 7 bits of the timer control status registers (TMCSR0 to TMCSR3) have functions to set the operating condition, enable/disable the operation, control interrupts, and check the status of the 16-bit reload timer.

■ Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L)

Figure 11.4-3 Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L)

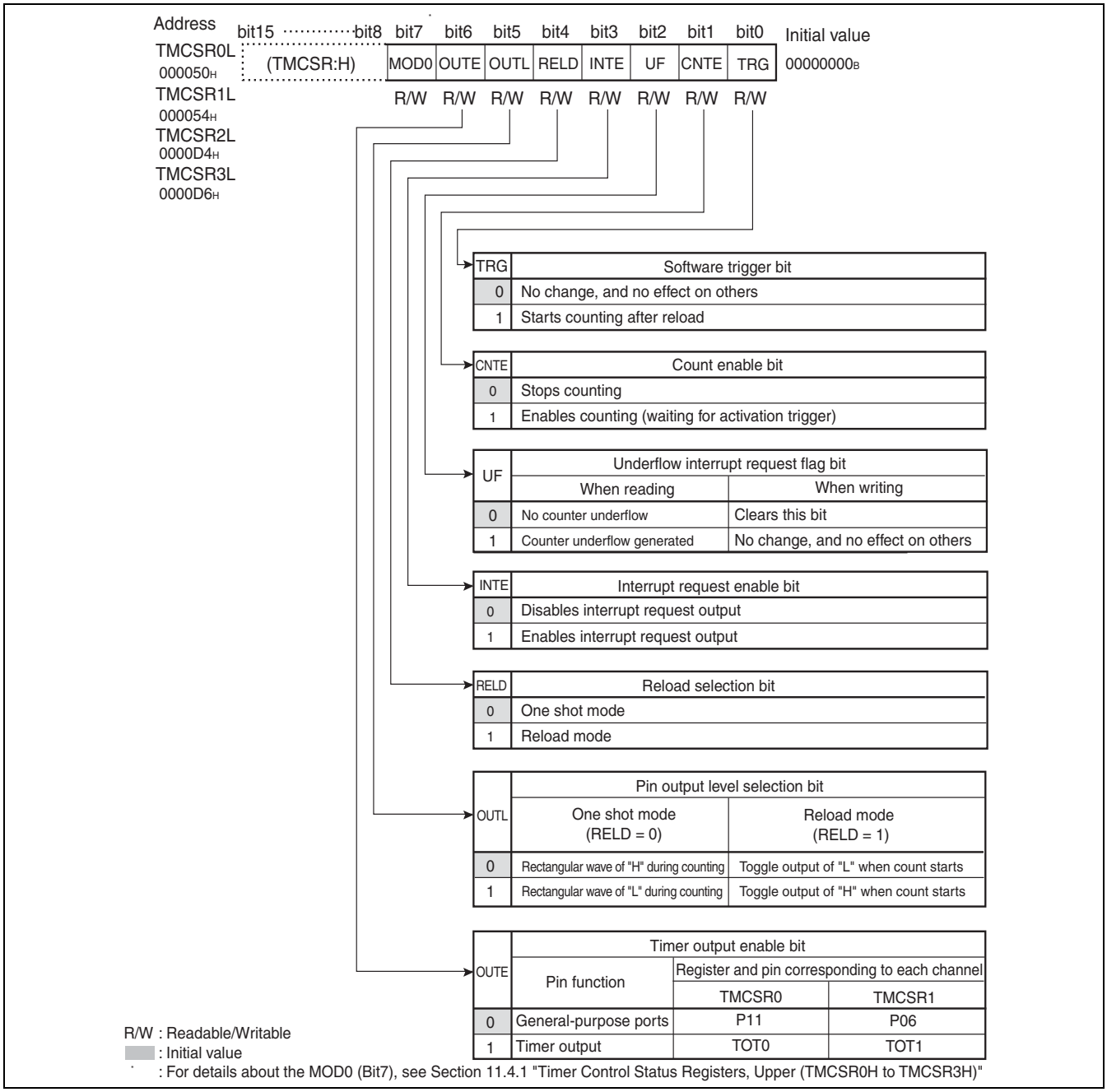


Table 11.4-2 Functions of Lower Bits of Timer Control Status Registers (TMCSR0L to TMCSR3L)

Bit name		Function
bit6	OUTE: Timer output enable bit	• Enables/disables outputs from the timer output pin. • When this bit is "0", the pin is used as general-purpose port; When this bit is "1", the pin is set as a timer output pin. • The output waveform from the timer output pin becomes toggle output in reload mode and square wave output in one shot mode, which indicates that counting is in progress.
bit5	OUTL: Pin output level selection bit	• A register used to select the output level of the timer output pin. • Toggle this bit to "0" or "1" to reverse the pin level.
bit4	RELD: Reload selection bit	• Enables reload operation. • Setting this bit "1" selects the reload mode, loads the value of the reload register into the counter when underflow occurs, and continues the count operation. • Setting this bit to "0" selects the one shot mode, and stops the count operation when underflow occurs.
bit3	INTE: Interrupt request enable bit	• Enables/disables output of interrupt requests to the CPU. • When this bit and the interrupt request flag bit (UF) are set to "1", an interrupt request is output.
bit2	UF: Underflow interrupt request flag bit	• This bit is set to "1" when a counter underflow occurs. • Writing "0" clears this bit. Writing "1" has no change and no effect on others. • "1" is always read from this bit when reading by the read-modify-write (RMW) instruction. • This bit is also cleared when EI^2OS occurs.
bit1	CNTE: Count enable bit	• Enables/disables count operation. • When this bit is set to "1", activation trigger wait state is entered. When activation trigger occurs, the actual counting starts.
bit0	TRG: Software trigger bit	• Used to activate the interval timer function or counter function by software. • Writing this bit to "1" activates the software trigger, loads the value of the reload register into the counter, and starts the count operation. Writing "0" has no effect. • When reading, "0" is always read from this bit. • When CNTE=1, the trigger input by this bit is always valid regardless of the operation mode.

11.4.3 16-bit Timer Registers (TMR0 to TMR3)

The 16-bit timer registers (TMR0 to TMR3) can always read the count value of the 16-bit down counter.

■ 16-bit Timer Registers (TMR0 to TMR3)

Figure 11.4-4 shows the bit configuration of the 16-bit timer registers (TMR0 to TMR3).

Figure 11.4-4 Bit Configuration of 16-bit Timer Registers (TMR0 to TMR3)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
TMR0: 000053_H	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX_B
TMR1: 000057_H	R	R	R	R	R	R	R	R	

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
TMR0: 000052_H	D7	D7	D7	D7	D7	D2	D1	D0	XXXXXXXX_B
TMR1: 000056_H	R	R	R	R	R	R	R	R	

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
TMR2: 003951_H	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX_B
TMR3: 003953_H	R	R	R	R	R	R	R	R	

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
TMR2: 003950_H	D7	D7	D7	D7	D7	D2	D1	D0	XXXXXXXX_B
TMR3: 003952_H	R	R	R	R	R	R	R	R	

R : Read-only
X : Undefined

These registers can read the counter value of the 16-bit down counter. When the counter operation is enabled (TMCSR0 to TMCSR3:CNTE=1) and the count is started, the value written to the 16-bit reload register is loaded into these registers and countdown is started. In counter stop state (TMCSR0 to TMCSR3:CNTE=0), the values of these register are retained.

Notes:

- Although these registers can be read during the counter operation, a word transfer instruction (e.g. MOVW A, 003AH) must be used.
- Be sure to use word access.
- The 16-bit timer registers (TMR0 to TMR3) are read-only registers, and assigned the same address as the write-only 16-bit reload registers (TMRLR0 to TMRLR3). Therefore, writing does not affect TMR value, but is performed to TMRLR0 to TMRLR3.

11.4.4 16-bit Reload Registers (TMRLR0 to TMRLR3)

The 16-bit reload registers (TMRLR0 to TMRLR3) are used to set a reload value to the 16-bit down counter. The value written to these registers are loaded to the down counter and counted down.

■ 16-bit Reload Registers (TMRLR0 to TMRLR3)

Figure 11.4-5 shows the bit configuration of the 16-bit reload registers (TMRLR0 to TMRLR3).

Figure 11.4-5 Bit Configuration of 16-bit Reload Register (TMRLR0 to TMRLR3)

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
TMRLR0H: 000053_H	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX_B
TMRLR1H: 000057_H	W	W	W	W	W	W	W	W	

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
TMRLR0L: 000052_H	D7	D7	D7	D7	D7	D2	D1	D0	XXXXXXXX_B
TMRLR1L: 000056_H	W	W	W	W	W	W	W	W	

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
TMRLR2H: 003951_H	D15	D14	D13	D12	D11	D10	D9	D8	XXXXXXXX_B
TMRLR3H: 003953_H	W	W	W	W	W	W	W	W	

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
TMRLR2L: 003950_H	D7	D7	D7	D7	D7	D2	D1	D0	XXXXXXXX_B
TMRLR3L: 003952_H	W	W	W	W	W	W	W	W	

R : Read only
X : Undefined value

Regardless of the operation mode of the 16-bit reload timer, these registers are set the initial value of the counter in the state the counter operation is disabled (TMCSR0 to TMCSR3:CNTE=0). When the counter operation is enabled (TMCSR0 to TMCSR3:CNTE=1) and the counter is started, the countdown is started from the value written to these registers.

In reload mode, the value set in the 16-bit reload registers (TMRLR0 to TMRLR3) is reloaded into the counter and the countdown continues when underflow occurs. In one shot mode, the counter stops at $FFFF_H$ when underflow occurs.

Notes:

- Writing to these registers must be performed with the counter stopped (TMCSR0 to TMCSR3: CNTE=0). And, a word transfer instruction (e.g. MOVW 003AH, A) must be used to write to these registers.
- Be sure to use word access.
- The 16-bit reload registers (TMRLR0 to TMRLR3) are functionally write-only registers, and assigned the same address as the read-only 16-bit timer registers (TMR0 to TMR3). Therefore, since the read value is the value of TMR0 to TMR3, instructions to perform the read-modify-write (RMW) operations, such as INC/DEC, cannot be used.

11.5 Interrupts of 16-bit Reload Timer

The 16-bit reload timer can generate an interrupt request due to counter underflow. The timer also supports the extended intelligent I/O service (EI^{2}OS).

■ Interrupts of 16-bit Reload Timer

Table 11.5-1 lists the interrupt control bits and interrupt sources of the 16-bit reload timer.

Table 11.5-1 Interrupt Control Bits and Interrupt Sources of 16-bit Reload Timer

	16-bit Reload Timer
Interrupt request flag bit	TMCSR0 to TMCSR3:UF
Interrupt request enable bit	TMCSR0 to TMCSR3:INTE
Interrupt source	Underflow of 16-bit down counter (TMR0 to TMR3)

In the 16-bit reload timer, when the underflow of the down counter (0000_H –> $FFFF_H$) occurs, UF bit in the timer control status registers (TMCSR0L to TMCSR3L,TMCSR0H to TMCSR3H) is set to "1". When interrupt requests are enabled (TMCSR0 to TMCSR3:INTE=1), the interrupt request is output to the interrupt controller.

■ Interrupts and EI^{2}OS of 16-bit Reload Timer

Table 11.5-2 lists the interrupts and EI^{2}OS of the 16-bit reload timer.

Table 11.5-2 Interrupts and EI^{2}OS of 16-bit Reload Timer

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
16-bit reload timer 0	#17 (11_H)	ICR03	$0000B3_H$	$FFFFB8_H$	$FFFFB9_H$	$FFFFBA_H$	△
16-bit reload timer 1	#28 ($1C_H$)	ICR08	$0000B8_H$	$FFFF8C_H$	$FFFF8D_H$	$FFFF8E_H$	△
16-bit reload timer 2	#18 (12_H)	ICR03	$0000B3_H$	$FFFFB4_H$	$FFFFB5_H$	$FFFFB6_H$	△
16-bit reload timer 3	#22 (16_H)	ICR05	$0000B5_H$	$FFFFA4_H$	$FFFFA5_H$	$FFFFA6_H$	△

△: Available when not using interrupt sources sharing ICR03, ICR05, ICR08 or the interrupt vector.

■ EI^{2}OS Function of 16-bit Reload Timer

The 16-bit reload timer has a circuit supporting EI^{2}OS. Therefore, a counter underflow can start EI^{2}OS. However, EI^{2}OS is available only when no other peripheral function that shares the interrupt control register (ICR) does not use the interrupt. To use the EI^{2}OS on the 16-bit reload timer 0, DTP/external interrupt 1 must be disabled. To use EI^{2}OS on the 16-bit reload timer 1, PPG timer 1 interrupt must be disabled.

11.6 Operation of 16-bit Reload Timer

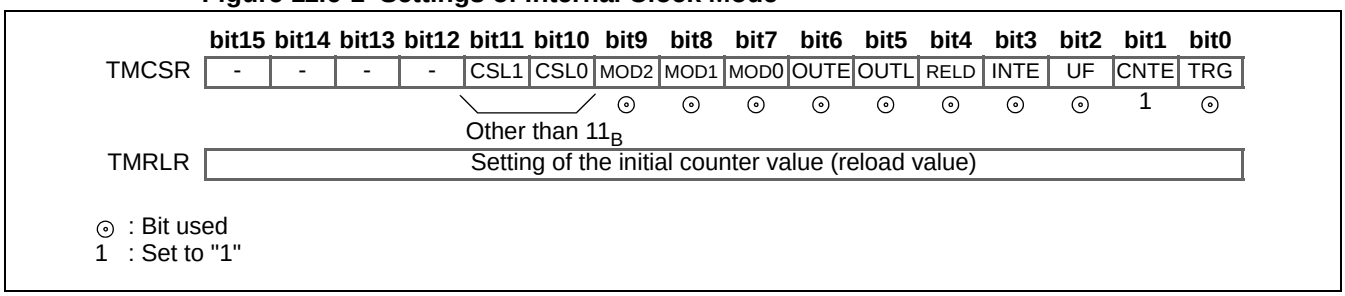
This section describes the 16-bit reload timer settings and counter operation states.

■ 16-bit Reload Timer Settings

● Settings of internal clock mode

To operate as an interval timer, the settings shown in Figure 11.6-1 are required.

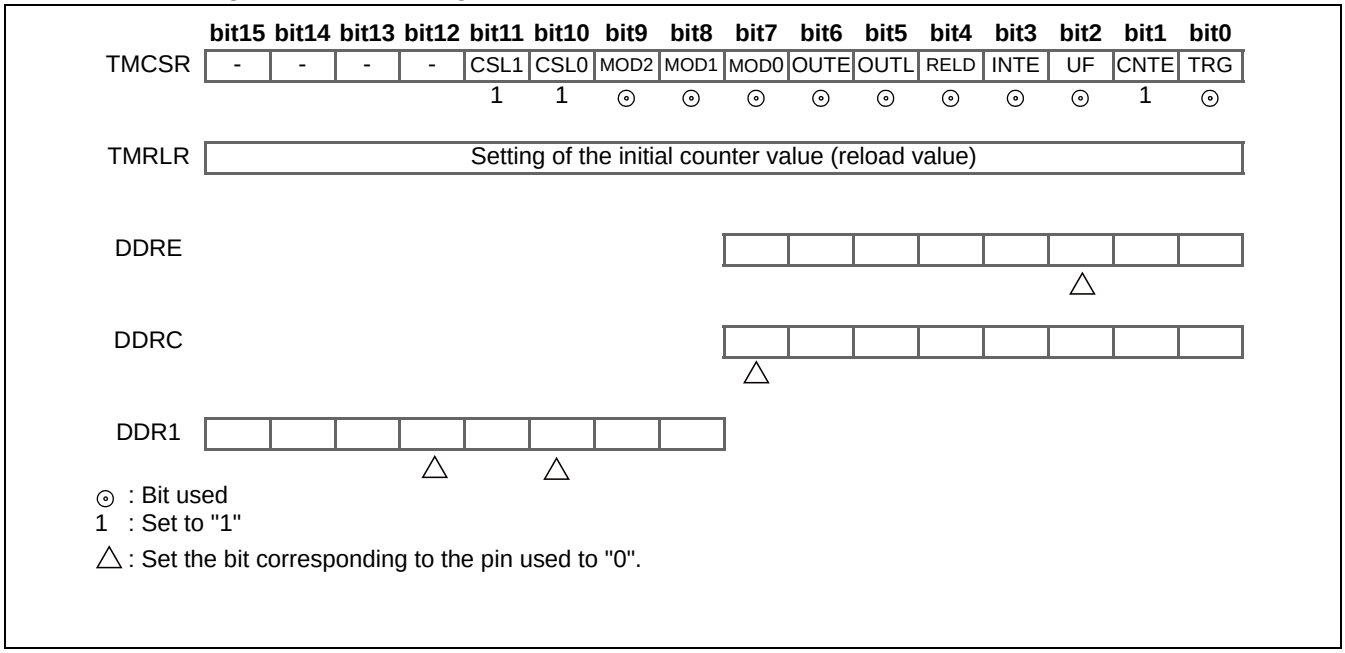
Figure 11.6-1 Settings of Internal Clock Mode



● Settings of event count mode

To operate as an event counter, the settings shown in Figure 11.6-2 are required.

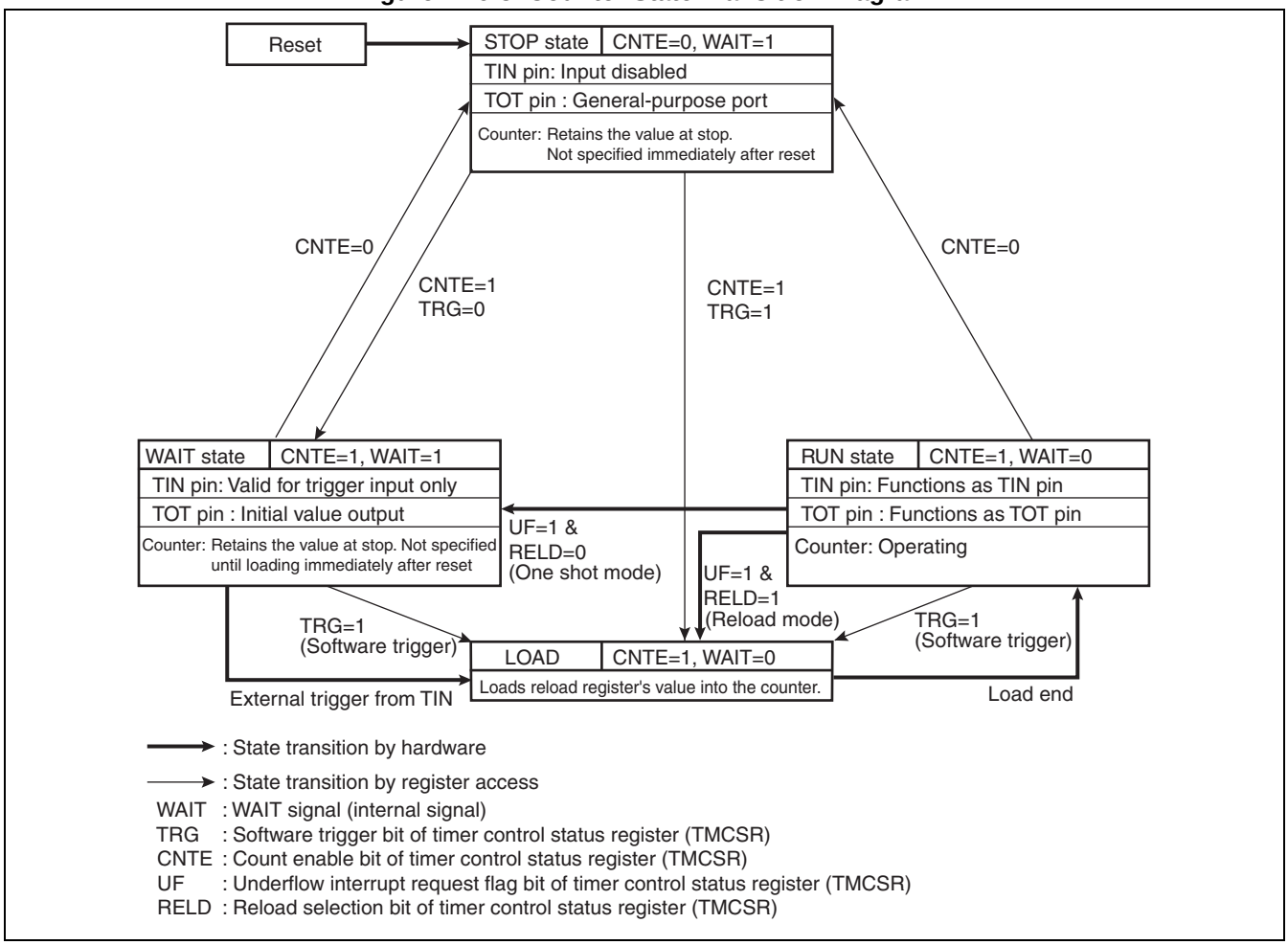
Figure 11.6-2 Settings of Event Count Mode



■ Counter Operation States

The state of the counter is determined by the CNTE bit of the timer control status register (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H) and the internal WAIT signal. States that can be set include the stop state (STOP state), activation trigger wait state (WAIT state), and operation state (RUN state). Figure 11.6-3 shows a counter state transition diagram.

Figure 11.6-3 Counter State Transition Diagram



11.6.1 Internal Clock Mode (Reload Mode)

The counter operates in sync with the internal count clock to count down the 16-bit counter and generates the interrupt request to CPU with the counter underflow. The counter also can output a toggle waveform from the timer output pin.

■ Operation in Internal Clock Mode (Reload Mode)

When the count operation is enabled (TMCSR0 to TMCSR3:CNTE=1) and the timer is activated by the software trigger bit (TMCSR:TRG) or the external trigger, the value of the 16-bit reload registers (TMRLR0 to TMRLR3) is loaded into the counter and the count operation is started. When both the count enable bit and the software trigger bit are set to "1", the count is enabled and starts.

If the counter value causes underflow (0000_H –> $FFFF_H$), the value of the 16-bit reload register (TMRLR0 to TMRLR3) is loaded to the counter, and the count operation is continued. At this moment, the underflow interrupt request flag bit (UF) is set to "1", and the interrupt request occurs if the interrupt request enable bit (INTE) is set to "1".

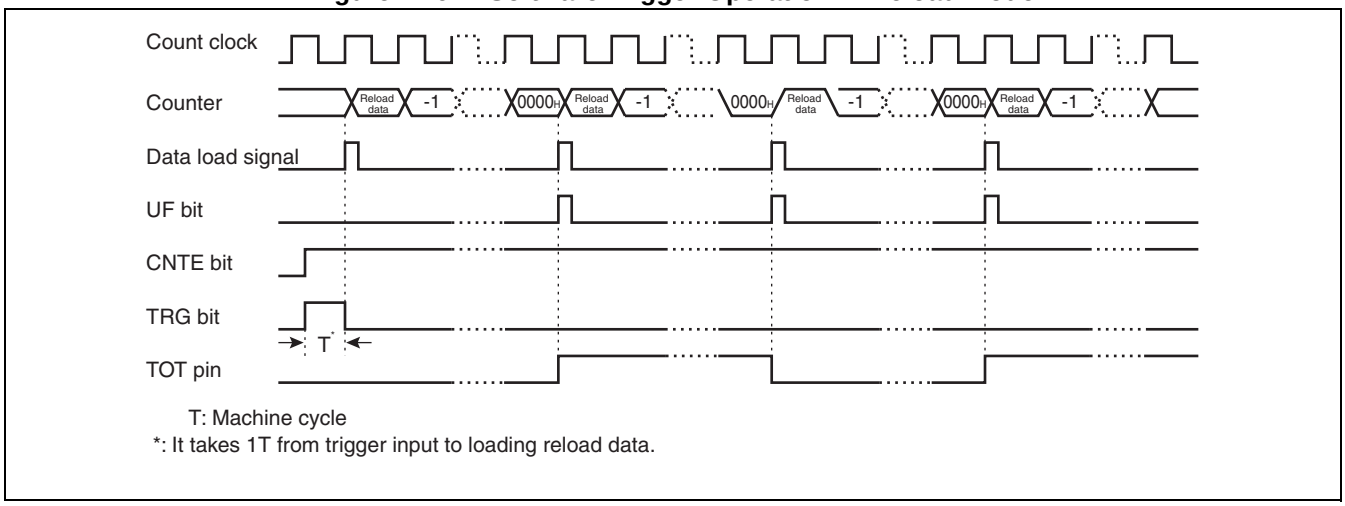
And, the TOT pin outputs a toggle waveform that is reversed at every underflow.

- Software trigger operation

Writing "1" to the TRG bit of the timer control status registers (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H) starts the counter.

Figure 11.6-4 shows the software trigger operation in reload mode.

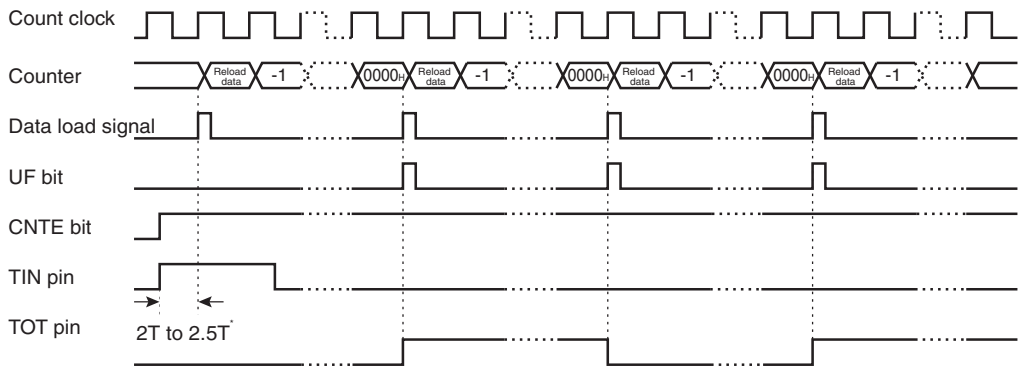
Figure 11.6-4 Software Trigger Operation in Reload Mode



- External trigger operation

When the valid edge (rising, falling, or both can be selectable) is input to the TIN pin, the counter is activated. Figure 11.6-5 shows the external trigger operation in reload mode.

Figure 11.6-5 External Trigger Operation in Reload Mode



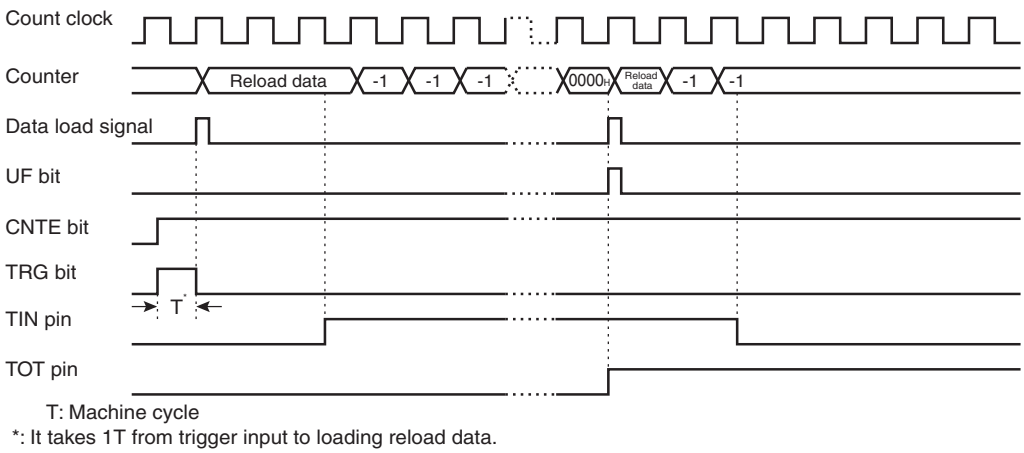
Note:

The pulse width of the trigger input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

- Gate input operation

While the valid level ("H" level or "L" level can be selected) is being input to the TIN pin, the count operation is performed. Figure 11.6-6 shows the gate input operation in reload mode.

Figure 11.6-6 Gate Input Operation in Reload Mode



Note:

The pulse width of the gate input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

11.6.2 Internal Clock Mode (One Shot Mode)

The counter operates in sync with the internal count clock to count down the 16-bit counter and generates the interrupt request to CPU with the counter underflow. The counter also output a square wave from TOT0 to TOT3 pins, which indicates that counting is in progress.

■ Operation of Internal Clock Mode (One Shot Mode)

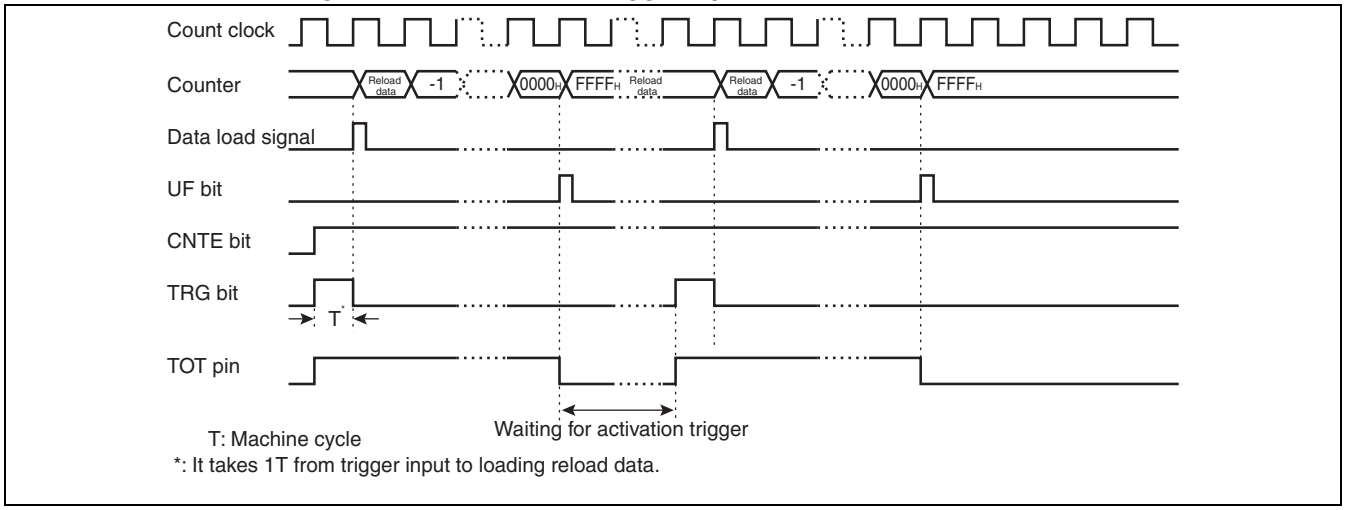
When the count operation is enabled (TMCSR0 to TMCSR3:CNTE=1) and the timer is activated by the software trigger bit (TMCSR0 to TMCSR3:TRG) or the external trigger, the count operation is started. When both the count enable bit and the software trigger bit are set to "1", the count is enabled and starts. If the counter value causes underflow (0000_H –> $FFFF_H$), the counter stops at $FFFF_H$. At this moment, the underflow interrupt request flag bit (UF) is set to "1", and the interrupt request occurs if the interrupt request enable bit (INTE) is set to "1".

The counter can also output a square wave from TOT pin, which indicates that counting is in progress.

● Software trigger operation

Writing "1" to the TRG bit of the timer control status registers (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H) starts the counter. Figure 11.6-7 shows the software trigger operation in on shot mode.

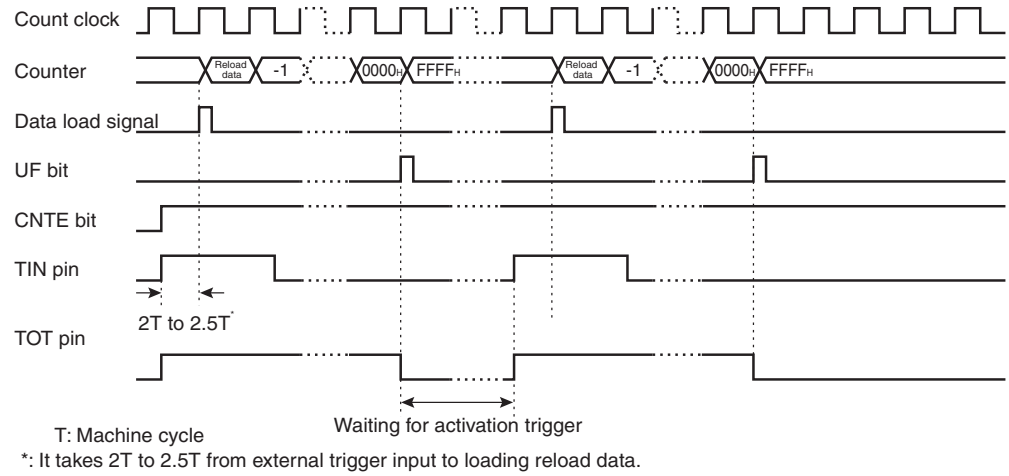
Figure 11.6-7 Software Trigger Operation in One Shot Mode



● External trigger operation

When the valid edge (rising, falling, or both can be selectable) is input to the TIN0 to TIN3 pins, the counter is activated. Figure 11.6-8 shows the external trigger operation in one shot mode.

Figure 11.6-8 External Trigger Operation in One Shot Mode



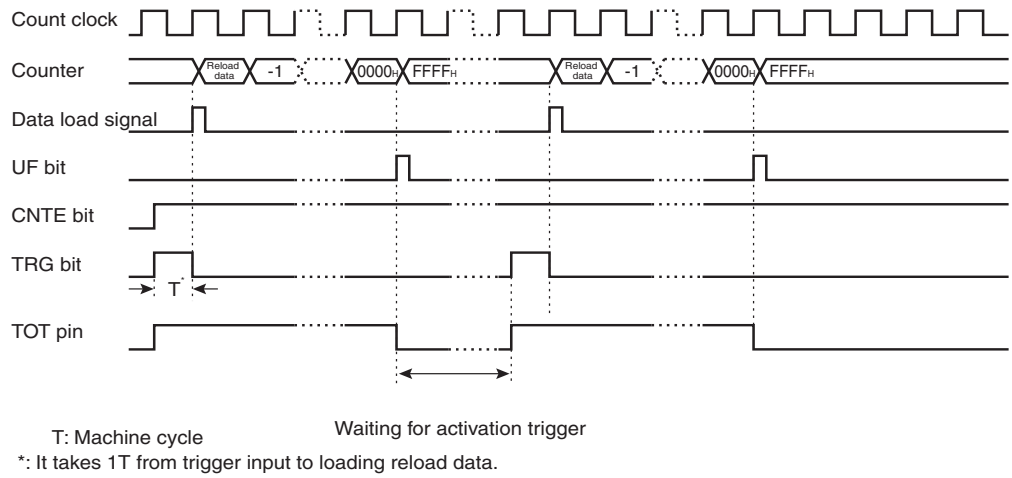
Note:

The pulse width of the trigger input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

- Gate input operation

While the valid level ("H" level or "L" level can be selected) is being input to the TIN0 to TIN3 pins, the count operation is performed. Figure 11.6-9 shows the gate input operation in one shot mode.

Figure 11.6-9 Gate Input Operation in One Shot Mode



Note:

The pulse width of the gate input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

11.6.3 Event Count Mode

The counter counts input edges from the TIN pin to count down the 16-bit counter and generates the interrupt request to CPU with the counter underflow. The TOT0 to TOT3 pins can also output either a toggle waveform or square wave.

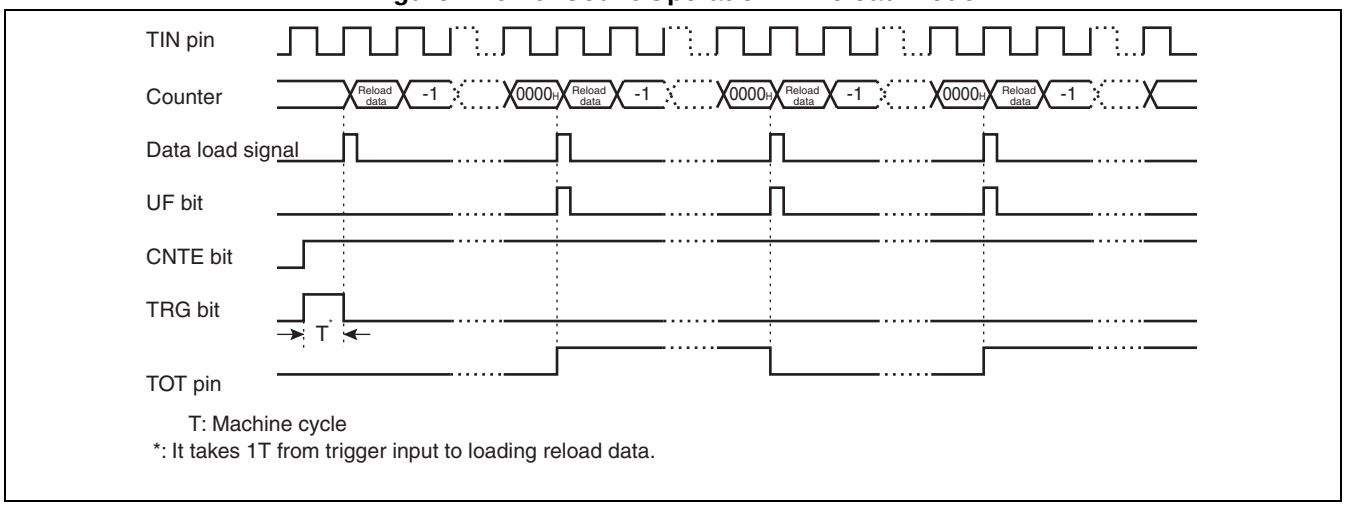
■ Event Count Mode

When the count operation is enabled (TMCSR0 to TMCSR3:CNTE=1) and the counter is activated (TMCSR0 to TMCSR3:TRG=1), the value of the 16-bit reload register (TMRLR0 to TMRLR3) is loaded into the counter and counted down whenever the valid edge (rising, falling, or both can be selected) of pulses (external count clock) input to the TIN0 to TIN3 pins is detected. When both the count enable bit and the software trigger bit are set to "1", the count is enabled and starts.

● Operation in reload mode

If the counter value causes underflow (0000_H –> $FFFF_H$), the value of the 16-bit reload register (TMRLR0 to TMRLR3) is loaded to the counter, and the count operation is continued. At this moment, the underflow interrupt request flag bit (UF) is set to "1", and the interrupt request occurs if the interrupt request enable bit (TMCSR0 to TMCSR3:INTE) is set to "1". And, the TOT0 to TOT3 pins output a toggle waveform that is reversed at every underflow. Figure 11.6-10 shows the count operation in reload mode.

Figure 11.6-10 Count Operation in Reload Mode



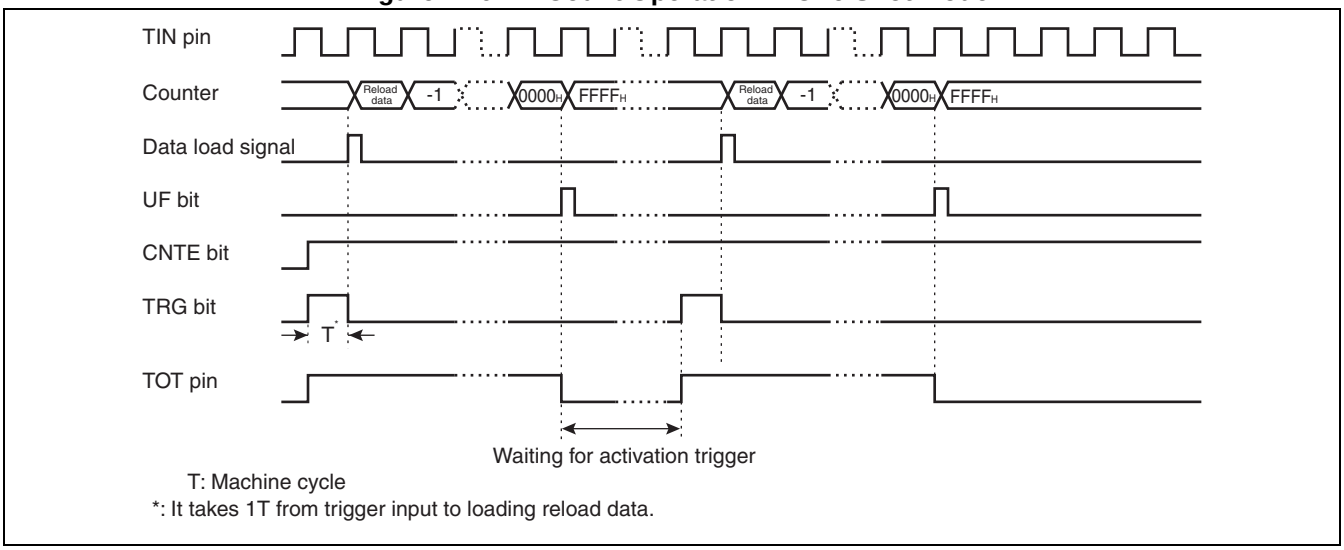
Note:

The "H" and "L" widths of the clock input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

- Operation in one shot mode

If the counter value causes underflow (0000_H –> $FFFF_H$), the counter stops at $FFFF_H$. At this moment, the underflow request flag bit (UF) is set to "1", and the interrupt request occurs if the interrupt request output enable bit (INTE) is set to "1". The counter can also output a square wave from TOT0 to TOT3 pins, which indicates that counting is in progress. Figure 11.6-11 shows the count operation in one shot mode.

Figure 11.6-11 Count Operation in One Shot Mode



Note:

The "H" and "L" widths of the clock input to the TIN pins (TIN0 to TIN3) must conform to the rating in the data sheet.

11.7 Notes on Using 16-bit Reload Timer

This section provides notes on using the 16-bit reload timer.

■ Notes on Using 16-bit Reload Timer

● Notes on setup by program

Writing to the 16-bit reload registers (TMRLR0 to TMRLR3) must be performed with the counter stopped (TMCSR0 to TMCSR3:CNTE=0). Although the 16-bit timer registers (TMR0 to TMR3) can be read during the counter operation, a word transfer instruction (e.g. MOVW A, dir) must be used.

Changing the CSL1 and CSL0 bits of the timer control status registers (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H) must be performed with the counter stopped (TMCSR0 to TMCSR3:CNTE=0).

● Notes on interrupts

If the UF bit of the timer control status registers (TMCSR0L to TMCSR3L, TMCSR0H to TMCSR3H) is set to "1" and the interrupt request is enabled (TMCSR:INTE=1), return from interrupt processing cannot be performed. Be sure to clear the UF bit.

Interrupt control register is shared between the 16-bit reload timers 0 and 2, the 16-bit reload timer 1 and the PPG timer 1, and the 16-bit reload timer 3 and the input capture 2. When the 16-bit reload timer uses EI^2OS, interrupts by the resource sharing the interrupt control register must be disabled.

● Notes in standby mode

When an evaluation product transfers to the standby mode or returns from it, the external interrupt request flag bit in external interrupt source register is set (EIRR:ER) no matter whether the external interrupt is set enable or disable (ENIR:EN=1/0).

11.8 Sample Program for 16-bit Reload Timer

This section provides sample programs for internal clock mode and event count mode of the 16-bit reload timer.

■ Sample Program for Internal Clock Mode

● Processing specifications

Use the 16-bit reload timer to generate a 12.5ms interval timer interrupt.

Use reload mode to generate interrupts repeatedly.

Use no external trigger, but use a software trigger to start the timer.

Does not use EI^{2}OS.

Use 32-MHz machine clock and 1 μs count clock.

[Coding example]

```
ICR03 EQU   0000B3H       ;Interrupt control register 03
TMCSR EQU   000050H       ;Timer control status register
TMR   EQU   000052H       ;16-bit timer register
TMRLR EQU   000052H       ;16-bit reload register
UF    EQU   TMCSR:2       ;Interrupt request flag bit
CNTE  EQU   TMCSR:1       ;Counter operation enable bit
TRG   EQU   TMCSR:0       ;Software trigger bit
;----------Main program-----------------------------------------------
CODE        CSEG
START:
;     :                   ;Stack pointer (SP) already initialized
      AND   CCR, #0BFH    ;Interrupt disabled
      MOV   I:ICR03, #00H ;Interrupt level 0 (highest)
      CLRB  I:CNTE        ;Counter paused
      MOVW  I:TMRLR, #30D3H ;Sets the data for the 12.5ms timer
      MOVW  I:TMCSR, #0001000000011011B
                          ;Interval timer operation, clock 2 μs
                          ;Disables external trigger, disables
                          ;external output
                          ;Selects reload mode, enables interrupt
                          ;Clears the interrupt flag, and starts
                          ;counter
      MOV   ILM, #07H     ;Sets ILM in PS to level 7.
      OR    CCR, #40H     ;Interrupt enabled
LOOP: MOV   A, #00H       ;Infinite loop
      MOV   A, #01H       ;
      BRA   LOOP          ;
```

```
;----------Interrupt program-----------------------------------------
WARI:
      CLRB  I:UF            ;Clears the interrupt request flag
;     :
;     User processing;      :
      RETI                  ; Return from interrupt.

CODE  ENDS
;----------Vector setting--------------------------------------------
VECT  CSEG  ABS=0FFH
      ORG   0FFB8H          ;Set the vector to interrupt #17(11H)
      DSL   WARI

ORG   0FFDCH                ;Reset vector setting
      DSL   START
      DB    00H             ;Set to single-chip mode.
VECT  ENDS
      END   START
```

■ Sample Program for Event Count Mode

● Processing specifications

When the rising edge in the pulses input to the external event input pin is counted the 10,000th time by the 16-bit reload timer/counter, an interrupt occurs.

The device operates in one shot mode.

The rising edge is selected for external trigger input.

Does not use EI^{2}OS.

[Coding example]

```
ICR03 EQU   0000B3H         ;Interrupt control register for 16-bit
                            ;reload timer
TMCSR EQU   000050H         ;Timer control status register
TMR   EQU   000052H         ;16-bit timer register
TMRLR EQU   000052H         ;16-bit reload register
DDR1  EQU   000011H         ;Port data register
UF    EQU   TMCSR:2         ;Interrupt request flag bit
CNTE  EQU   TMCSR:1         ;Counter operation enable bit
TRG   EQU   TMCSR:0         ;Software trigger bit
;----------Main program ----------------------------------------------CSEG
START:
;     :                     ;Stack pointer (SP) already initialized
      AND   CCR, #0BFH      ;Interrupt disabled
      MOV   I:ICR03, #00H   ;Interrupt level 0 (highest)
      MOV   I:DDR1, #00H    ;Sets P12/TIN0 pin to input
      CLRB  I:CNTE          ;Counter paused
```

```
      MOVW  I:TMRLR, #2710H ;Sets reload value to 10000
      MOVW  I:TMCSR, #0000110010001011B
                            ;Counter operation, external trigger, rising
                            ;edge, external output disabled
                            ;Selects one shot mode, enables interrupt
                            ;Clears the interrupt flag, and starts
                            ;counter
      MOV   ILM, #07H       ;Set ILM in PS to level 7
      OR    CCR, #40H       ;Interrupt enabled
LOOP: MOV   A, #00H         ;Infinite loop
      MOV   A, #01H         ;
      BRA   LOOP            ;
;----------Interrupt program-----------------------------------------
WARI:
      CLRB  I:UF            ;Clears the interrupt request flag
;     :
;     User processing
;     :
      RETI                  ; Return from interrupt

CODE  ENDS
;----------Vector setting--------------------------------------------
VECT  CSEG  ABS=0FFH
      ORG   0FF84H          ;Set the vector to interrupt #30(1E_H)
      DSL   WARI
      ORG   0FFDCH          ; Reset vector setting
      DSL   START
      DB    00H             ; Set to single-chip mode
      VECT  ENDS
      END   START
```

12. PPG Timer

This chapter describes the operations of PPG timer.

12.1 Overview of PPG Timer

12.2 Block Diagram of PPG Timer

12.3 Registers of PPG Timer

12.4 Interrupt of PPG Timer

12.5 PPG Timer Operation

12.1 Overview of PPG Timer

The PPG timer consists of the prescaler, 16-bit down counter (x1), 16-bit data register with a buffer for setting cycle, 16-bit compare register with a buffer for setting a duty, and a pin control section.
The PPG timer can output pulses synchronized to the external or software trigger.
The cycle and duty of the output pulse can be changed freely by updating two 16-bit register values.

■ Overview of PPG Timer

● PWM function

The PPG timer can output pulses programmably by updating the values of the registers above in synchronization to the trigger.

It can also be used as a D/A converter by an external circuit.

● One shot function

This function can detect an edge of the trigger input, and output a single pulse.

● Pin control

- Set to "1" after duty is matched (priority).
- Reset to "0" by counter borrow.
- Has output value fixed mode to simply output all "L" (or all "H").
- Polarity can be specified.

● 16-bit down counter

The counter operation clock can be selected among 10 types. 10 types of internal clock
(ϕ, ϕ/2, ϕ/4, ϕ/8, ϕ/16, ϕ/32, ϕ/64, ϕ/128, ϕ/256, ϕ/512)
ϕ: Machine Clock

● Interrupt request

- Timer activation
- Counter borrow (cycle match)
- Duty match
- Counter borrow (cycle match) or duty match
- Multiple channels can be set to activate simultaneously and restart during operation by an external trigger.

12.2 Block Diagram of PPG Timer

This section shows a block diagram of PPG timer.

■ Block Diagram

Figure 12.2-1 Block Diagram

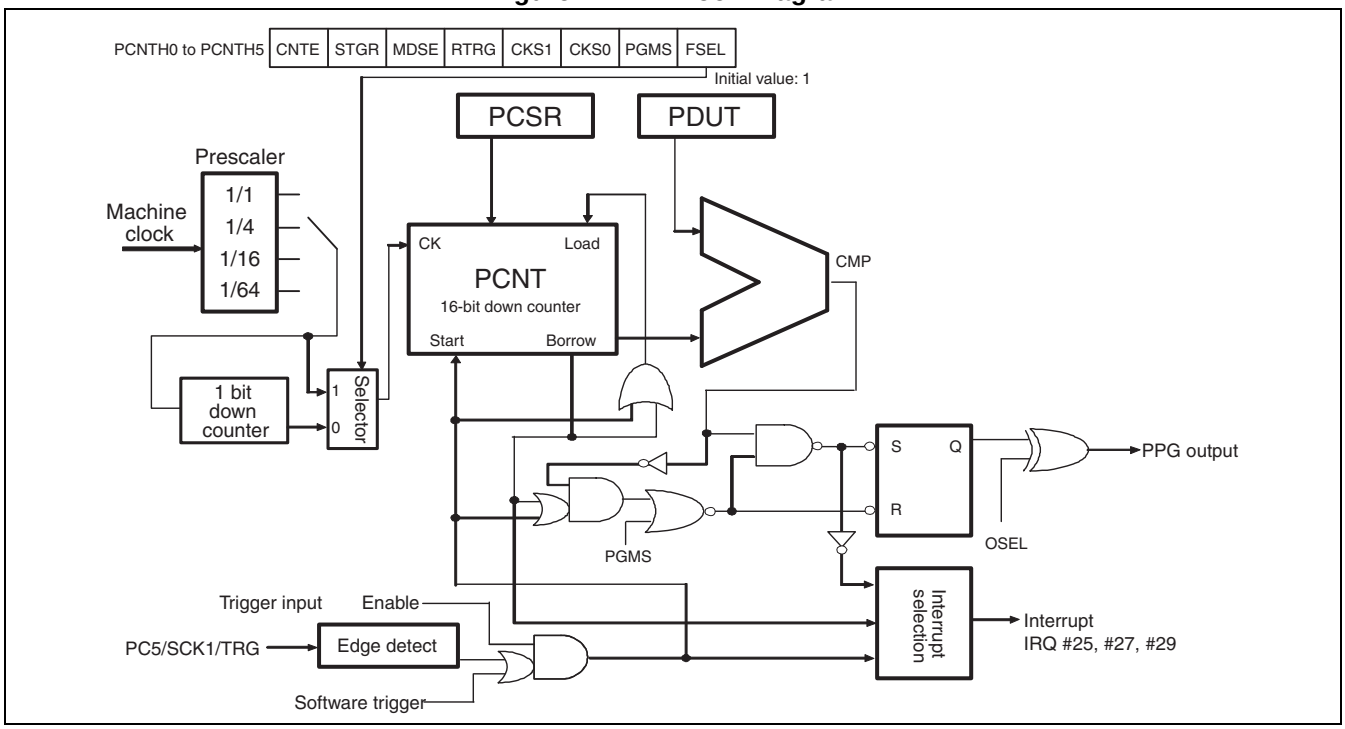
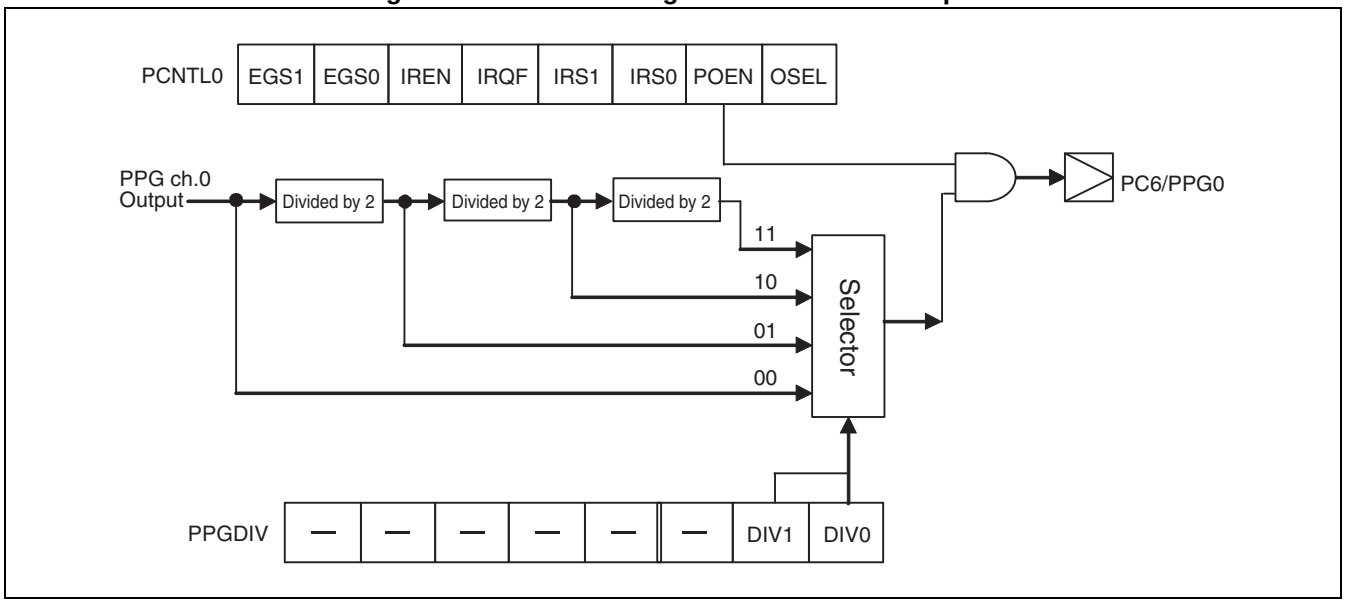


Figure 12.2-2 Block Diagram of PPG Ch.0 Output



12.3 Registers of PPG Timer

This section describes the registers of the PPG timer.

■ Registers of PPG Timer

Registers of PPG timer are described in detail next.

12.3.1 List of PPG Timer Registers

This section describes the list of the PPG timer registers.

■ List of PPG Timer Registers

PPG Control Status Register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 00002B$_H$ ch.1 00002D$_H$ ch.2 00002F$_H$ ch.3 0000DB$_H$ ch.4 0000DD$_H$ ch.5 0000DF$_H$	CNTE	STGR	MOSE	CKST	CKS1	CKS0	PGMS	FSEL	PCNTH0 to PCNTH5
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	0	0	0	0	0	0	0	1	← Initial value

PPG Control Status Register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 00002A$_H$ ch.1 00002C$_H$ ch.2 00002E$_H$ ch.3 0000DA$_H$ ch.4 0000DC$_H$ ch.5 0000DE$_H$	EGS1	EGS0	IREN	IREQF	IRS1	IRS0	POEN	OSEL	PCNTL0 to PCNTL5
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

PPG Down Counter Register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003921$_H$ ch.1 003929$_H$ ch.2 003931$_H$ ch.3 0039E1$_H$ ch.4 0039E9$_H$ ch.5 0039F1$_H$	DC15	DC14	DC13	DC12	DC11	DC10	DC09	DC08	PDCRH0 to PDCRH5
	R	R	R	R	R	R	R	R	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG Down Counter Register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003920$_H$ ch.1 003928$_H$ ch.2 003930$_H$ ch.3 0039E0$_H$ ch.4 0039E8$_H$ ch.5 0000F0$_H$	DC07	DC06	DC05	DC04	DC03	DC02	DC01	DC00	PDCRL0 to PDCRL5
	R	R	R	R	R	R	R	R	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG Cycle Setting Register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003923$_H$ ch.1 00392B$_H$ ch.2 003933$_H$ ch.3 0039E3$_H$ ch.4 0039EB$_H$ ch.5 0039F3$_H$	CS15	CS14	CS13	CS12	CS11	CS10	CS09	CS08	PCSRH0 to PCSRH5
	W	W	W	W	W	W	W	W	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG Cycle Setting Register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003922$_H$ ch.1 00392A$_H$ ch.2 003932$_H$ ch.3 0039E2$_H$ ch.4 0039EA$_H$ ch.5 0000F2$_H$	CS07	CS06	CS05	CS04	CS03	CS02	CS01	CS00	PCSRL0 to PCSRL5
	W	W	W	W	W	W	W	W	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG Duty Setting Register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003925_H ch.1 003920_H ch.2 003935_H ch.3 $0039E5_H$ ch.4 $0039E0_H$ ch.5 $0039F5_H$	DU15	DU14	DU13	DU12	DU11	DU10	DU09	DU08	PDURH0 to PDURH5
	W	W	W	W	W	W	W	W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

PPG Duty Setting Register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003924_H ch.1 $00392C_H$ ch.2 003934_H ch.3 $0039E4_H$ ch.4 $0039EC_H$ ch.5 $0000F4_H$	DU07	DU06	DU05	DU04	DU03	DU02	DU01	DU00	PDURL0 to PDURL5
	W	W	W	W	W	W	W	W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

PPG0 Output Division Setting Register

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003926_H ch.1 $00392E_H$ ch.2 003936_H ch.3 $0039E6_H$ ch.4 $0039EE_H$ ch.5 $0000F6_H$	-	-	-	-	-	-	DIV1	DIV0	PPGDIV0 to PPGDIV5
	R	R	R	R	R	R	R/W	R/W	← Read/Write
	1	1	1	1	1	1	0	0	← Initial value

12.3.2 Detailed Description of PPG Timer

The PPG timer has the following 5 registers.

- **PPG control status register (PCNT0 to PCNT5)**
- **PPG down counter register (PDCR0 to PDCR5)**
- **PPG cycle setting register (PCSR0 to PCSR5)**
- **PPG duty setting register (PDUT0 to PDUT5)**
- **PPG output division setting register (PPGDIV0 to PPGDIV5)**

■ PPG Control Status Register (PCNT)

Upper bits of PPG control status register

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 $00002B_H$ ch.1 $00002D_H$ ch.2 $00002F_H$ ch.3 $0000DB_H$ ch.4 $0000DD_H$ ch.5 $0000DF_H$	CNTE	STGR	MOSE	CKST	CKS1	CKS0	PGMS	FSEL	PCNTH0 to PCNTH5
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	0	0	0	0	0	0	0	1	← Initial value

Lower bits of PPG control status register

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 $00002A_H$ ch.1 $00002C_H$ ch.2 $00002E_H$ ch.3 $0000DA_H$ ch.4 $0000DC_H$ ch.5 $0000DE_H$	EGS1	EGS0	IREN	IREQF	IRS1	IRS0	POEN	OSEL	PCNTL0 to PCNTL5
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

[bit15] CNTE: Timer enable bit

This bit enables the operation of the 16-bit down counter.

0	Stop (initial value)
1	Enabled

[bit14] STGR: Software trigger bit

Writing "1" to the STGR bit issues a software trigger.

The read value of the STGR bit is always "0".

[bit13] MDSE: Mode selection bit

This bit selects either the PWM operation to output pulses continuously or the one shot operation to output a single pulse. This bit cannot be rewritten during operation.

0	PWM operation (initial value)
1	One shot operation

[bit12] RTRG: Restart enable bit

This bit enables to restart based on a trigger (software/external). This bit cannot be rewritten during operation.

0	Restart disabled (initial value)
1	Restart enable

[bit11, bit10] CKS1,CKS0: Count clock selection bits

These bits are used to select a counter clock for the 16-bit count down. These bits cannot be rewritten 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

By writing "1" into PGMS bit, PPG output can be masked to either "0" or "1", regardless of mode setting, cycle setting value, and duty setting value.

0	No output mask (initial value)
1	Output mask

PPG output when writing "1" to PGMS

Polarity	PPG output
Normal polarity	L
Reverse polarity	H

To output all "H" at normal polarity or all "L" at reverse polarity, write the same value into the cycle setting register and duty setting register. This enables the reverse of the above mask values to be output.

[bit8] FSEL: Count clock division control bit

0	Divided by 2
1	Divided by 1 (initial value)

FSEL bit sets the division ratio of the count clock.
If "0" is written to FSEL, the cycle set by the counter clock selection bits (CKS1 and CKS0) is further divided by 2.

[bit7, bit6] EGS1,EGS0: Trigger input edge selection bits

These bits select a valid edge polarity of the external trigger. Writing "1" to the software trigger bit enables the software triggers in any mode selected.

EGS1	EGS0	Edge selection
0	0	Invalid (initial value)
0	1	Rising edge
1	0	Falling edge
1	1	Both edges

[bit5] IREN: Interrupt request enable bit

This bit enables to interrupt the PPG timer. When the IREN bit is "1", setting the interrupt flag (bit4:IRQF) to "1" generates an interrupt.

0	Interrupt disabled (initial value)
1	Interrupt enabled

[bit4] IRQF: Interrupt request flag

When the interrupt source selected by the interrupt source selection bit ((bit3, bit2:IRS1, IRS0) occurs, IRQF is set to "1". If the interrupt request enable bit (bit5:IREN) is enabled the interrupt request is issued to the CPU.

The IRQF bit is readable and writable. It can be cleared by writing "0"; writing "1" will not change the bit value. The read value is "1" regardless of the bit value when using a read-modify-write (RMW) instruction.

If setting "1" and writing "0" occur simultaneously, writing "0" is preferred.

0	No interrupt request (initial value)
1	Interrupt requested

[bit3, bit2] IRS1,IRS0: Interrupt source selection bits

These bits select the source to set the bit4:IRQF. These bits cannot be rewritten 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 reverse polarity PPG↓ (duty match)
1	1	Counter borrow or normal polarity PPG↑, or reverse polarity PPG↓

[bit1] POEN: PPG output enable bit

Setting this bit to "1" enables PPG to output from the pin.

0	General-purpose port (initial value)
1	PPG output pin

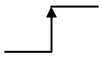
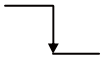
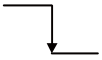
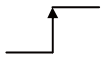
[bit0] OSEL: PPG output polarity specification bit

This bit sets the polarity of the PPG output.

0	Normal polarity (initial value)
1	Reverse polarity

The following operation can be performed in combination with the bit9: PGMS.

PGMS	OSEL	PPG output
0	0	Normal polarity (initial value)
0	1	Reverse polarity
1	0	Output fixed to "L"
1	1	Output fixed to "H"

Polarity	Stop state	Duty matched	Counter matched
Normal polarity	"L" output		
Reverse polarity	"H" output		

■ PPG Down Counter Register (PDCR)

PDCR register can read the value of the 16-bit down counter.

Access the PDCR register in words.

PPG down counter register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003921_H ch.1 003929_H ch.2 003931_H ch.3 $0039E1_H$ ch.4 $0039E9_H$ ch.5 $0039F1_H$	DC15	DC14	DC13	DC12	DC11	DC10	DC09	DC08	PDCRH0 to PDCRH5
	R	R	R	R	R	R	R	R	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG down counter register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003920_H ch.1 003928_H ch.2 003930_H ch.3 $0039E0_H$ ch.4 $0039E8_H$ ch.5 $0000F0_H$	DC07	DC06	DC05	DC04	DC03	DC02	DC01	DC00	PDCRL0 to PDCRL5
	R	R	R	R	R	R	R	R	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

■ PPG Cycle Setting Register (PCSR)

The PCSR register sets the cycle with a buffer. Transfer from the buffer is executed by a counter borrow or activation.

Write to the cycle setting register, then to the duty setting register at the time of initializing the cycle setting register or for changing the setting.

Access the PCSR register in words. Only writing is available, and reading results in an undefined value.

PPG cycle setting register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003923_H ch.1 $00392B_H$ ch.2 003933_H ch.3 $0039E3_H$ ch.4 $0039EB_H$ ch.5 $0039F3_H$	CS15	CS14	CS13	CS12	CS11	CS10	CS09	CS08	PCSRH0 to PCSRH5
	W	W	W	W	W	W	W	W	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

PPG cycle setting register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003922_H ch.1 $00392A_H$ ch.2 003932_H ch.3 $0039E2_H$ ch.4 $0039EA_H$ ch.5 $0000F2_H$	CS07	CS06	CS05	CS04	CS03	CS02	CS01	CS00	PCSRL0 to PCSRL5
	W	W	W	W	W	W	W	W	← Read/Write
	1	1	1	1	1	1	1	1	← Initial value

■ PPG Duty Set Register (PDUT)

The PDUT register sets the duty with a buffer. The transfer from the buffer is executed by a counter borrow or activation.

If both values are the same in the cycle setting register and the duty setting register, all "H" for normal polarity or all "L" for reverse polarity is output.

Do not set the value which causes PCSR<PDUT. PPG output will be undefined.

Access the PDUT register in words. Only writing is available, and reading results in an undefined value.

PPG duty setting register (upper)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
ch.0 003925_H ch.1 003920_H ch.2 003935_H ch.3 $0039E5_H$ ch.4 $0039E0_H$ ch.5 $0039F5_H$	DU15	DU14	DU13	DU12	DU11	DU10	DU09	DU08	PDURH0 to PDURH5
	W	W	W	W	W	W	W	W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

PPG duty setting register (lower)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003924_H ch.1 $00392C_H$ ch.2 003934_H ch.3 $0039E4_H$ ch.4 $0039EC_H$ ch.5 $0000F4_H$	DU07	DU06	DU05	DU04	DU03	DU02	DU01	DU00	PDURL0 to PDURL5
	W	W	W	W	W	W	W	W	← Read/Write
	0	0	0	0	0	0	0	0	← Initial value

■ PPG0 Output Division Set Register (PPGDIV)

PPG0 Output Division Setting Register

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ch.0 003926_H ch.1 $00392E_H$ ch.2 003936_H ch.3 $0039E6_H$ ch.4 $0039EE_H$ ch.5 $0000F6_H$	-	-	-	-	-	-	DIV1	DIV0	PPGDIV0 to PPGDIV5
	R	R	R	R	R	R	R/W	R/W	← Read/Write
	1	1	1	1	1	1	0	0	← Initial value

[bit7 to bit2] Undefined bits

These are undefined bits.

Writing has no effect on operation.

Read value is always "1".

[bit1, bit0] DIV1, DIV0: Division setting bits

These bits set the division ratio of PPG ch.0.

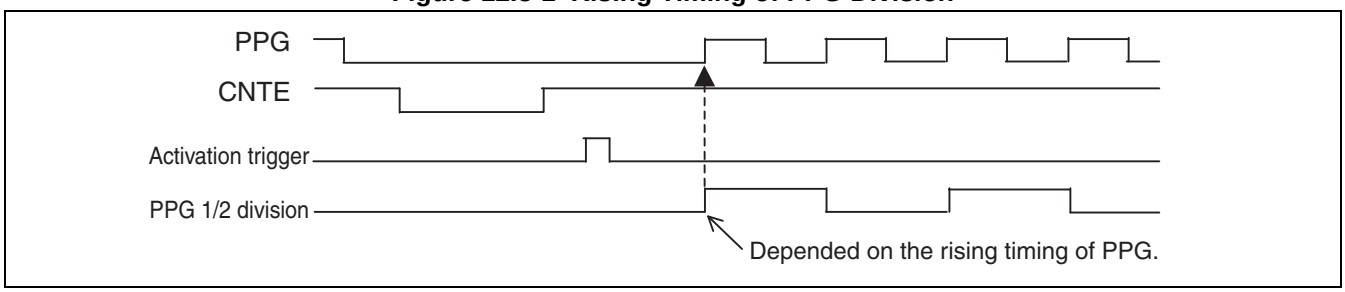
CS1	CS0	Division ratio
0	0	1/1 (initial value)
0	1	1/2
1	0	1/4
1	1	1/8

Note:

There are following limitations when the 1/2, 1/4, and 1/8 division setting is used.

- The output waveform is fixed to 50% of the duty.
- The one shot operation (PCNT:MDSE=1) is prohibited.
- The PPG output reverse function (PCNT:OSEL=1) is prohibited.
- The state of the PPG output fixed (PCNT:PGMS, OSEL=01_B, 10_B, 11_B) is prohibited.
- When PCSR=PDUT, the setting is prohibited.

Figure 12.3-1 Rising Timing of PPG Division



The PPG division is depended on the rising timing of PPG.

12.4 Interrupt of PPG Timer

The PPG timer can generate an interrupt request by timer activation, counter borrow (cycle match), and duty match. The timer also supports the extended intelligent I/O service (EI^{2}OS).

■ Interrupt of PPG Timer

Table 12.4-1 shows the interrupt control bits and interrupt source of the PPG timer.

Table 12.4-1 Interrupt Control Bits and Interrupt Source of PPG Timer

Interrupt source	Lower bits of PPG control status register (PCNTL5 to PCNTL0)		
	Interrupt flag bit	Interrupt enabled bit	Clear interrupt flag
• Timer activation • Counter borrow (cycle match) • Duty match	IRQF	IREN	• Write "0" to IRQF bit • Clear by EI^{2}OS • Reset

The interrupt flag bit of the PPG timer is set to "1" by the interrupt source listed in Table 12.4-1. If the interrupt enable bit is set to "1" at this time, an interrupt request is output to an interrupt controller.

■ Interrupt of PPG Timer and EI^{2}OS

Table 12.4-2 shows the interrupt of PPG and EI^{2}OS.

Table 12.4-2 Interrupt of PPG Timer and EI^{2}OS

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
PPG timer 0	#25(19_H)	ICR07	$0000B7_H$	$FFFF98_H$	$FFFF99_H$	$FFFF9A_H$	△
PPG timer 1	#27($1B_H$)	ICR08	$0000B8_H$	$FFFF90_H$	$FFFF91_H$	$FFFF92_H$	△
PPG timer 2	#29($1D_H$)	ICR09	$0000B9_H$	$FFFF88_H$	$FFFF89_H$	$FFFF8A_H$	△
PPG timer 3							
PPG timer 4							
PPG timer 5							

△ : Available when ICR07, ICR08, ICR09, or interrupt sources sharing an interrupt vector are not used

■ EI^{2}OS Functions of PPG Timer

The PPG timer has a circuit supporting EI^{2}OS. Thus, the PPG timer can activate EI^{2}OS by timer activation, counter borrow (cycle match), and duty match. However, EI^{2}OS for ICR07, ICR08, ICR09 is available only when no other peripheral function that shares the interrupt control register (ICR) or the interrupt vector is used. For example, when the EI^{2}OS is activated with the PPG timer 0, external interrupts of the channel 6/7 and the interrupt of UART3 transmission must be prohibited.

12.5 PPG Timer Operation

This section describes the PPG Timer Operation.

■ PPG Timer Operation

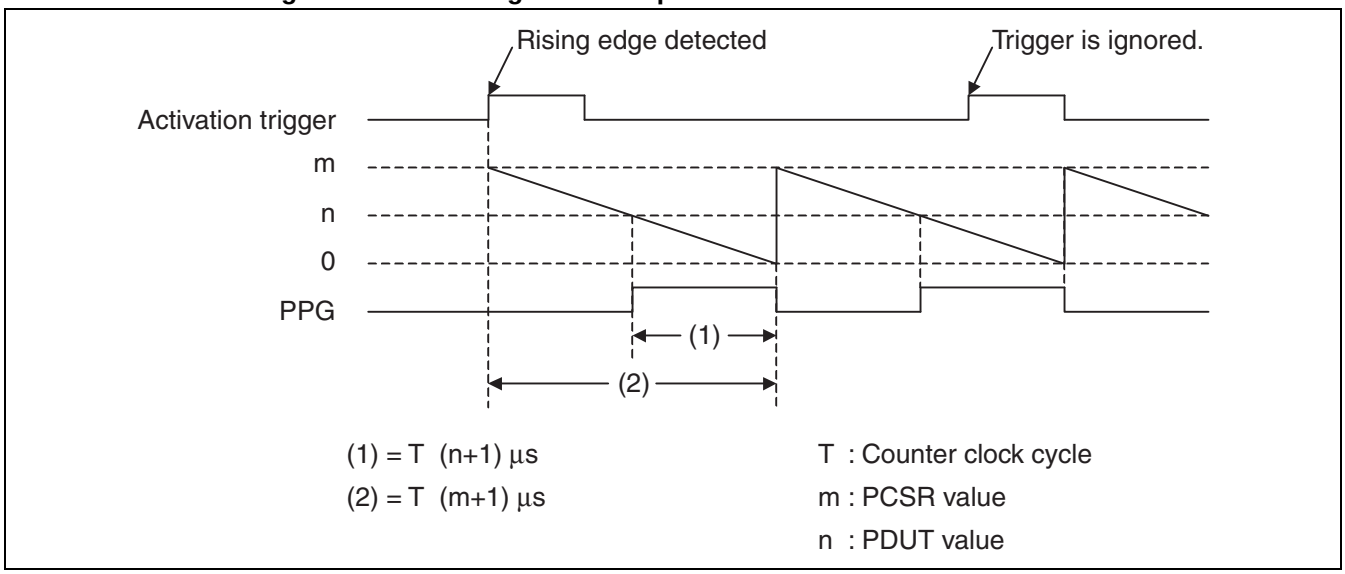
PWM operation, one shot operation, interrupt source and timing are described next.

12.5.1 PWM Operation

In PWM operation, pulses can be output continuously after an activation trigger was detected. The cycle of the output pulse can be controlled by changing the PCSR value, the duty ratio can be controlled by changing the PDUT value.

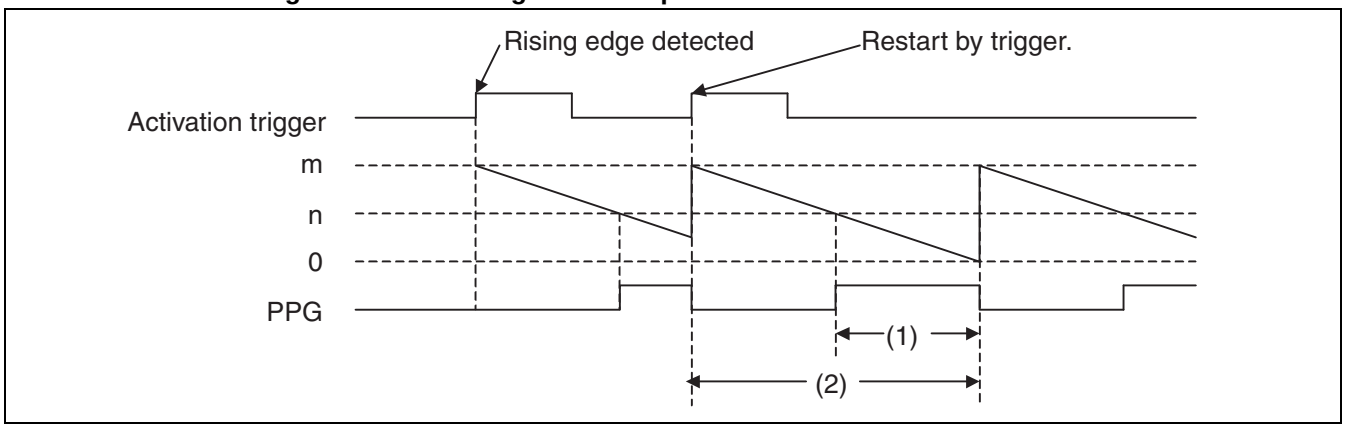
■ When Disabling Restart

Figure 12.5-1 Timing of PWM Operation When Restart is Disabled



■ When Enabling Restart

Figure 12.5-2 Timing of PWM Operation When Restart is Enabled



Note:

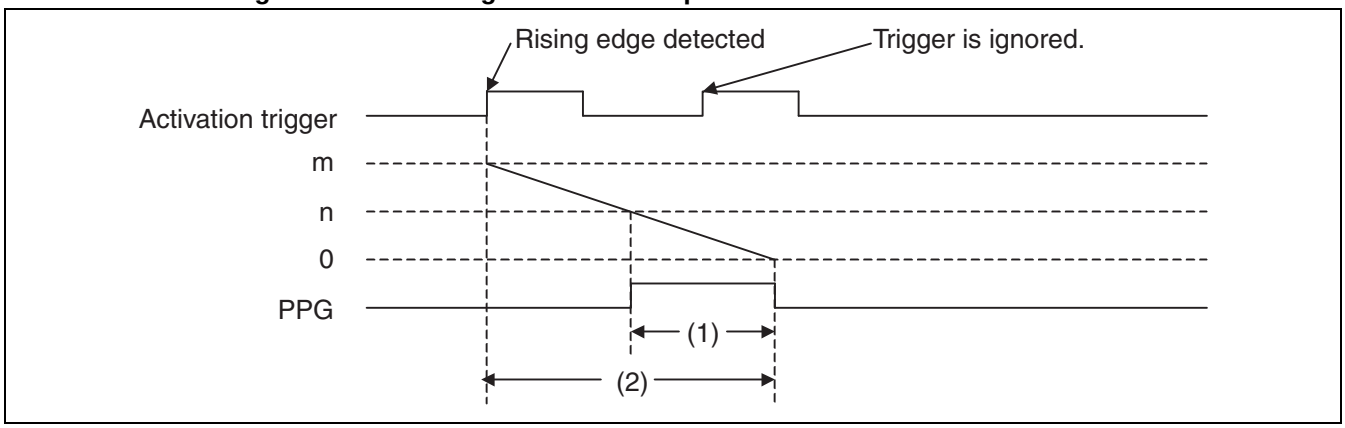
After writing data into PCSR, be sure to write to PDUT.
Refer to the data sheet for the minimum pulse width of the external TRG input.
However, even if a pulse with a smaller pulse width than quoted above is input, it may be recognized as valid. Also, since there is no filter functions for the external TRG input, add an external filter as required.

12.5.2 One Shot Operation

In one shot operation, a single pulse with an arbitrary width can be output by a trigger. When restart is enabled, the counter value is reloaded if an activation trigger is detected during operation.

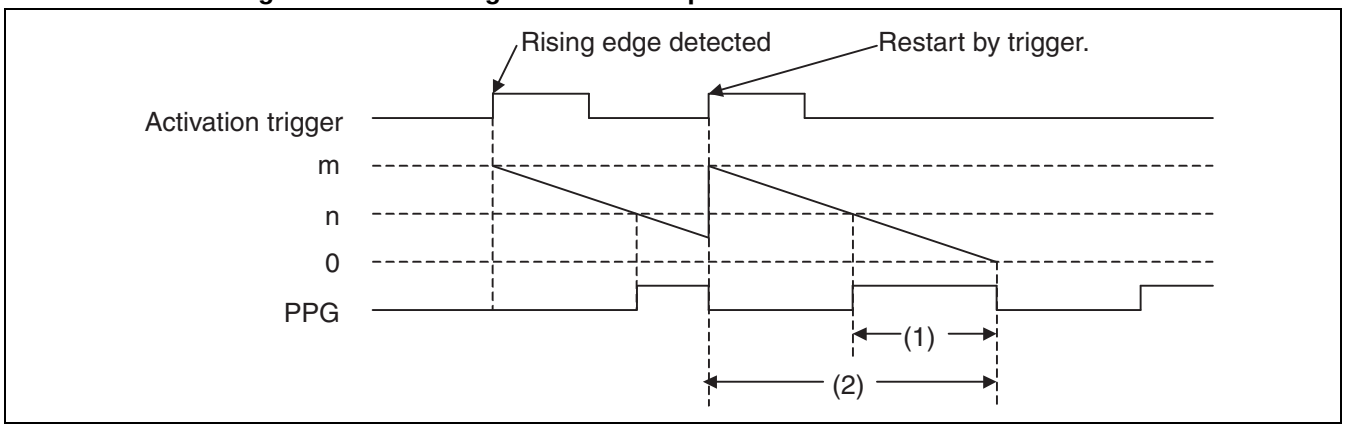
■ When Disabling Restart

Figure 12.5-3 Timing of One Shot Operation When Restart is Disabled



■ When Enabling Restart

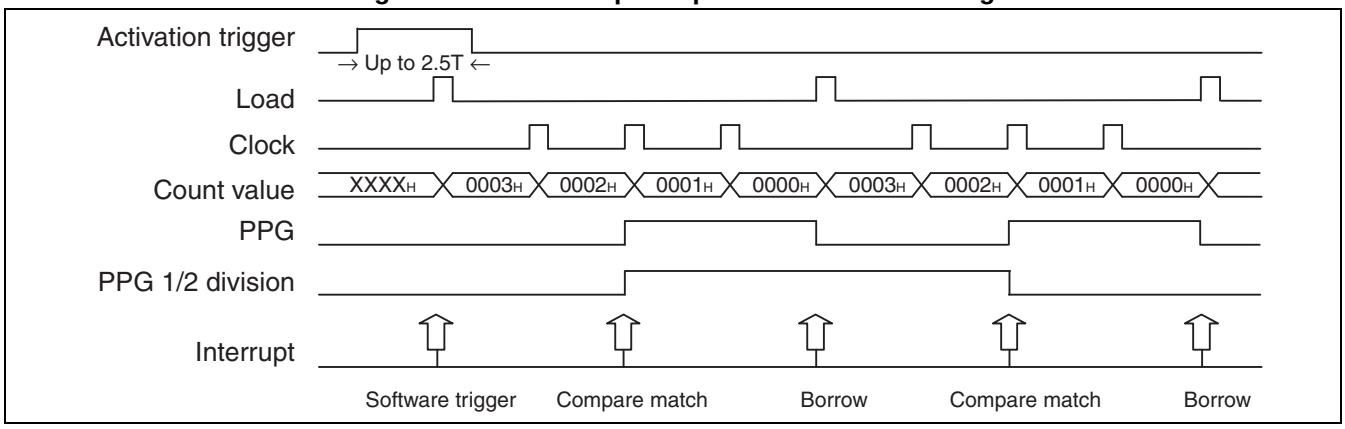
Figure 12.5-4 Timing of One Shot Operation When Restart is Enabled



12.5.3 Interrupt Source and Timing

2.5T (T: count clock cycle) is required at most for loading the count value after an activation trigger.

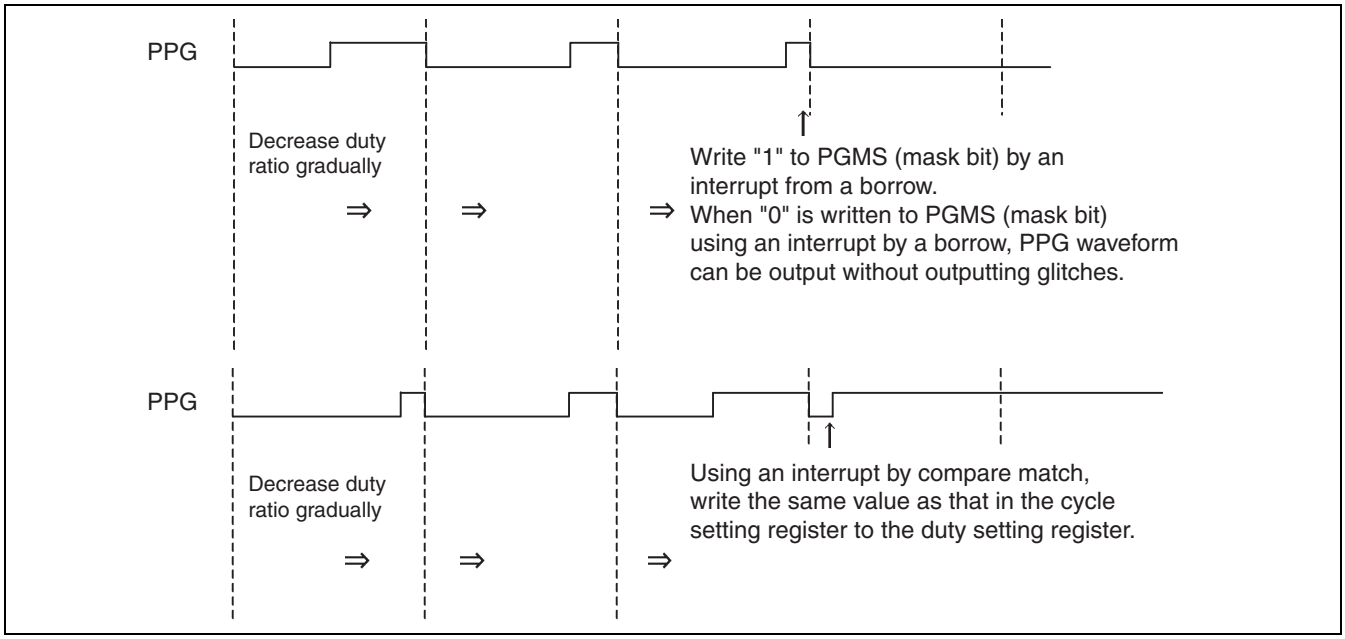
Figure 12.5-5 Interrupt Output Sources and Timing



Note:

An interrupt is generated in the timing of 1/1 division when the 1/2, 1/4, and 1/8 division is used.

■ Example of PWM Output for All "L" or All "H"



13. Real-Time Watch Timer

This chapter describes the functions and operations of the real-time watch timer.

13.1 Overview of Real-time Watch Timer
13.2 Registers of Real-time Watch Timer
13.3 Interrupt of Real-time Watch Timer

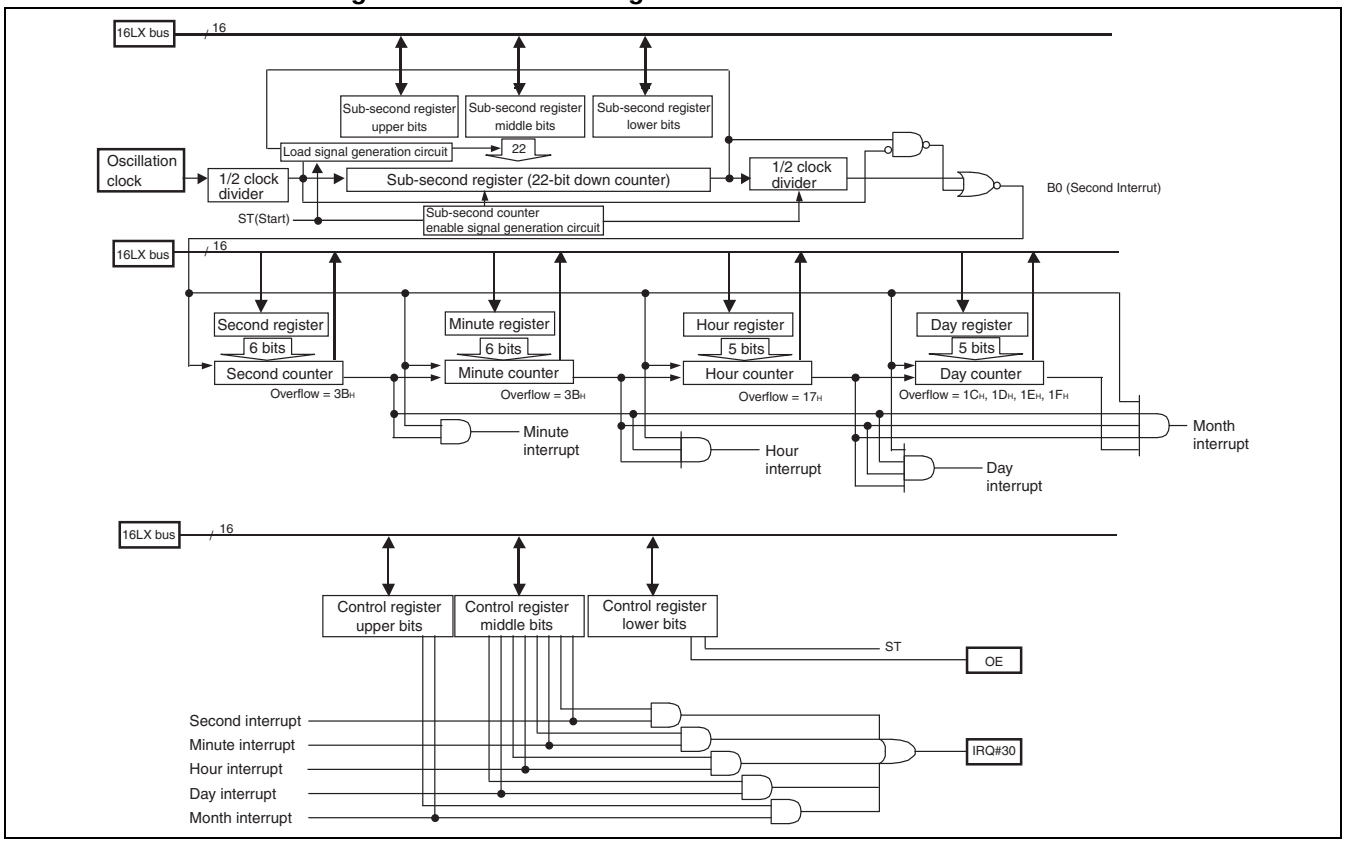
13.1 Overview of Real-time Watch Timer

The real-time watch timer consists of the real-time watch timer control register, sub-second data register, second/minute/hour/day registers, 1/2 clock divider, 21-bit prescaler, and second/minute/hour/day counters. The oscillation frequency of the MCU is assumed to be 4MHz in order to perform a specified operation of the real-time watch timer. The real-time watch timer operates as a real-world timer, providing information on real-world time.

■ Block Diagram of Real-time Watch Timer

Figure 13.1-1 shows a block diagram of the real-time watch timer.

Figure 13.1-1 Block Diagram of Real-time Watch Timer



13.2 Registers of Real-time Watch Timer

The 6 types of registers of the real-time watch timer are as follows:

- **Real-time watch timer control register (WTCR)**
- **Sub-second data register (WTBR)**
- **Second data register (WTSR)**
- **Minute data register (WTMR)**
- **Hour data register (WTHR)**
- **Day data register (WTDR)**

■ List of Registers of Real-time Watch Timer

Figure 13.2-1 lists the registers of the real-time watch timer.

Figure 13.2-1 Registers of Real-time Watch Timer

Upper bits of real-time watch timer control register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000CE$_H$	–	–	–	–	Reserved	Reserved	INTE4	INT4	WTCRH
Read/Write →	–	–	–	–	R/W	R/W	R/W	R/W	
Initial value →	–	–	–	–	X	X	0	0	

Middle bits of real-time watch timer control register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 0000CD$_H$	INTE3	INT3	INTE2	INT2	INTE1	INT1	INTE0	INT0	WTCRM
Read/Write →	–	–	–	R/W	R/W	W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Lower bits of real-time watch timer control register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000CC$_H$	Reserved	Reserved	Reserved	–	–	Reserved	Reserved	ST	WTCRL
Read/Write →	R/W	R/W	R/W	–	–	R/W	R/W	R/W	
Initial value →	0	0	0	–	–	X	X	0	

Sub-second data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 00395A$_H$	–	–	D21	D20	D19	D18	D17	D16	WTBRH
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	X	X	X	X	X	X	

Sub-second data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 003959$_H$	D15	D14	D13	D12	D11	D10	D19	D18	WTBRM
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Sub-second data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 003958$_H$	D7	D6	D5	D4	D3	D2	D1	D0	WTBRL
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Second data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 00395B$_H$	–	–	S5	S4	S3	S2	S1	S0	WTSR
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	0	0	0	0	0	0	

Minute data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 00395C$_H$	–	–	M5	M4	M3	M2	M1	M0	WTMR
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	X	X	X	X	X	X	

(Continued)

(Continued)

Minute data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 00395C$_H$	**–**	**–**	**M5**	**M4**	**M3**	**M2**	**M1**	**M0**	WTMR
Read/Write →	**–**	**–**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	
Initial value →	–	–	X	X	X	X	X	X	

Hour data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 00395D$_H$	**–**	**–**	**–**	**H4**	**H3**	**H2**	**H1**	**H0**	WTHR
Read/Write →	**–**	**–**	**–**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	
Initial value →	–	–	–	0	0	0	0	0	

Day data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 00395E$_H$	**MS1**	**MS0**	**–**	**N4**	**N3**	**N2**	**N1**	**N0**	WTDR
Read/Write →	**R/W**	**R/W**	**–**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	
Initial value →	0	0	–	0	0	0	0	0	

13.2.1 Real-Time Watch Timer Control Register

The real-time watch timer control register activates and stops the real-time watch timer, controls interrupts, and sets external output pins.

■ Bit Configuration of Timer Control Register

Figure 13.2-2 shows the bit configuration of the timer control register.

Figure 13.2-2 Bit Configuration of Timer Control Register

Upper bits of real-time watch timer control register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000CE$_H$	–	–	–	–	Reserved	Reserved	INTE4	INT4	WTCRH
Read/Write →	–	–	–	–	R/W	R/W	R/W	R/W	
Initial value →	–	–	–	–	X	X	0	0	

Middle bits of real-time watch timer control register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 0000CD$_H$	INTE3	INT3	INTE2	INT2	INTE1	INT1	INTE0	INT0	WTCRM
Read/Write →	–	–	–	R/W	R/W	W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Lower bits of real-time watch timer control register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 0000CC$_H$	Reserved	Reserved	Reserved	–	–	Reserved	Reserved	ST	WTCRL
Read/Write →	R/W	R/W	R/W	–	–	R/W	R/W	R/W	
Initial value →	0	0	0	–	–	X	X	0	

[bit15 to bit8,bit1, bit0] WTCRH, WTCRM, INT4 to INT0, INTE4 to INTE0: Interrupt flags and interrupt enable flags

INT4 to INT0 are interrupt flags. Theses flags are set in case of 1 second overflow (second interrupt), and when an overflow of the second counter (minute interrupt), minute counter (hour interrupt), hour counter (day interrupt), or day counter (month interrupt) occurs. If the INT bit is set when the corresponding INTE bit is "1", the real-time watch timer issues an interrupt signal. These flags are designed to issue an interrupt signal at every second (INT0)/ minute (INT1)/hour (INT2)/day (INT3)/month (INT4).

Writing "0" to the INT bits clears each flag, writing "1" has no effect. When a read-modify-write (RMW) instruction is executed on the INT bit, "1" is read.

[bit7 to bit4] WTCRH: Undefined bits

The read value is "1".

Writing has no effect on operation.

[bit7 to bit5, bit2, bit1] WRCRL: Reserved bits

Be sure to set these bits to "0".

[bit4, bit3] WTCRL: Undefined bits

The read value is "1".

Writing has no effect on operation.

[bit2, bit1] WTCRH: Reserved bits

Be sure to set these bits to 00_B.

[bit0] WRCRL: ST: Start bit

When the ST bit is set to "1", the real-time watch timer loads the second/minute/hour/day values from each register to start the operation.

When the ST bit is set to "0", the 22-bit prescaler and 1/2 clock division circuit are reset to their initial value and stop the operation. Day, hour, minute, and second counters stop with retaining data.

When the ST bit is reset to "0" by an internal reset, all counters and prescalers are reset to the initial value "0" and stop the operation.

To operate $\rightarrow$ stop $\rightarrow$ operate the ST bit, the intervals from the stop (writing "0" to the ST bit) to the operation (rewriting "1" to the ST bit) must be at least (oscillation clock $\times$ 2) seconds or more.

13.2.2 Sub-Second Data Register

The sub-second register stores reload value for the 22-bit prescaler used for dividing the frequency of the oscillation clock. The reload values are generally specified in such a way that 22-bit prescaler output becomes just 0.5 seconds cycle.

■ Bit Configuration of Sub-second Data Register

Figure 13.2-3 shows the bit configuration of the sub-second data register.

Figure 13.2-3 Bit Configuration of Sub-second Data Register

Sub-second data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 00395A$_H$	–	–	D21	D20	D19	D18	D17	D16	WTBRH
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	X	X	X	X	X	X	

Sub-second data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: 003959$_H$	D15	D14	D13	D12	D11	D10	D9	D8	WTBRM
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Sub-second data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: 003958$_H$	D7	D6	D5	D4	D3	D2	D1	D0	WTBRL
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

[bit7, bit6] WTBRH: Undefined bits

The read value is "1".

Writing has no effect on operation.

[bit5 to bit0] WTBRH: D21 to D16

[bit15 to bit8] WTBRM: D15 to D8

[bit7 to bit0] WTBRL: D7 to D0

The sub-second data register is used to store reload values for the 22-bit prescaler. These values are reloaded after the reload counter reaches "0". Note that the reload operation is not executed between write instructions when changing all 3 bytes. Otherwise, the 22-bit prescaler may load an incorrect value that consists of new and old data bytes. In general, it is recommended that the sub-second register is rewritten while the start bit is "0". When the sub-second register is set to "0", the 22-bit prescaler dose not operate at all.

The input clock uses the oscillation clock, and its frequency is designed to be 4MHz. Setting 0F423F (hexadecimal) to the reload value of the 22-bit prescaler provides the accurate clock signal of 0.5 second.

13.2.3 Second/Minute/Hour/Day Data Registers

The second/minute/hour/day data registers are used to store time information. They indicate the seconds, minutes, hours, and days in binary.
When these registers are read, the counter value is simply returned. However, the setting value is read form the MS1 and MS0 bit of the WTDR register.

■ Bit Configuration of Second/Minute/Hour/Day Data Registers

Figure 13.2-4 shows the configurations of the second/minute/hour/day data registers.

Figure 13.2-4 Bit Configuration of Second/Minute/Hour/Day Data Registers

Second data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $00395B_H$	–	–	S5	S4	S3	S2	S1	S0	WTSR
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	0	0	0	0	0	0	

Minute data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $00395C_H$	–	–	M5	M4	M3	M2	M1	M0	WTMR
Read/Write →	–	–	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	X	X	X	X	X	X	

Hour data register

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
Address: $00395D_H$	–	–	–	H4	H3	H2	H1	H0	WTHR
Read/Write →	–	–	–	R/W	R/W	R/W	R/W	R/W	
Initial value →	–	–	–	0	0	0	0	0	

Day data register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
Address: $00395E_H$	MS1	MS0	–	N4	N3	N2	N1	N0	WTDR
Read/Write →	R/W	R/W	–	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	–	0	0	0	0	0	

[bit13 to bit8] WTSR: S5 to S0 second data bits

[bit5 to bit0] WTMR: M5 to M0 minute data bits

[bit12 to bit8] WTHR: H4 to H0 hour data bits

[bit4 to bit0] WTDR: N4 to N0 day data bits

Since the second/minute/hour/day data registers have 4 byte registers, please check that the value obtained from the registers is consistent.

For example, if the value of "1 day, 1 hour, 59 minutes, 59 seconds" is obtained, that value may represent "1 day, 0 hour, 59 minutes,

59 seconds", "1 day, 1 hour, 0 minutes, 0 seconds", or "1 day, 2 hours, 0 minutes, 0 seconds".

Similarly, the operation clock of the MCU is one half of the oscillation clock (when the PLL stops), the read value from these registers may be incorrect. This is caused by synchronization between the read operation and the count operation. Therefore, it is recommended that the read instruction is triggered by using the second interrupt.

Note:

Do not set impossible data (e.g. 60 seconds) to the WTSR, WTMR, WTHR, and WTDR registers. Also, do not set the inconsistent data with the MS1 and MS0 bits to the N4 to N1 bits.

WTDR [bit7, bit6] MS1, MS0 day count setting bits

These bits are used to set the count value of the day counter.

MS1	MS0	Operation
0	0	Counts up to 31 days (initial value).
0	1	Counts up to 30 days.
1	0	Counts up to 29 days.
1	1	Counts up to 28 days.

[bit15, bit14] WTSR: Undefined bits

[bit7, bit6] WTMR: Undefined bits

[bit15 to bit13] WTHR: Undefined bits

[bit5] WTDR: Undefined bit

- The read value is "1".
- Writing has no effect on operation.

13.3 Interrupt of Real-time Watch Timer

The real-time watch timer can generate an interrupt request when an 1 second overflow occurs, and each of the second/minute/hour/day counter overflows.

■ Interrupt of Real-time Watch Timer

Table 13.3-1 shows the interrupt control bits and interrupt sources of the real-time watch timer.

Table 13.3-1 Interrupt of Real-time Watch Timer

Interrupt source	Real-time watch timer control register (WTCR)		
	Interrupt flag bit	Interrupt enable bit	Clearing the interrupt flag bit
Second interrupt (1 second overflow)	INT0	INTE0	• Writing "0" to the INT0 to INT4 bits • Reset
Minute interrupt (second counter overflow)	INT1	INTE1	
Hour interrupt (minute counter overflow)	INT2	INTE2	
Day interrupt (hour counter overflow)	INT3	INTE3	
Month interrupt (day counter overflow)	INT4	INTE4	

In the real-time watch timer, the interrupt flag bit is set to "1" by the interrupt source shown in Table 13.3-1. If the interrupt enable bit is set to "1" at this time, the interrupt request is output to the interrupt controller.

■ Interrupts of Real-time Watch Timer and EI^{2}OS

Table 13.3-2 shows the interrupts of the real-time watch timer, EI^{2}OS.

Table 13.3-2 Interrupts of Real-time Watch Timer and EI^{2}OS

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
Real-time watch timer	#30($1E_H$)	ICR09	$0000B9_H$	$FFFF84_H$	$FFFF85_H$	$FFFF86_H$	X

X: Not available

14. Delay Interrupt Generation Module

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

14.1 Overview of Delay Interrupt Generation Module
14.2 Operation of Delay Interrupt Generation Module

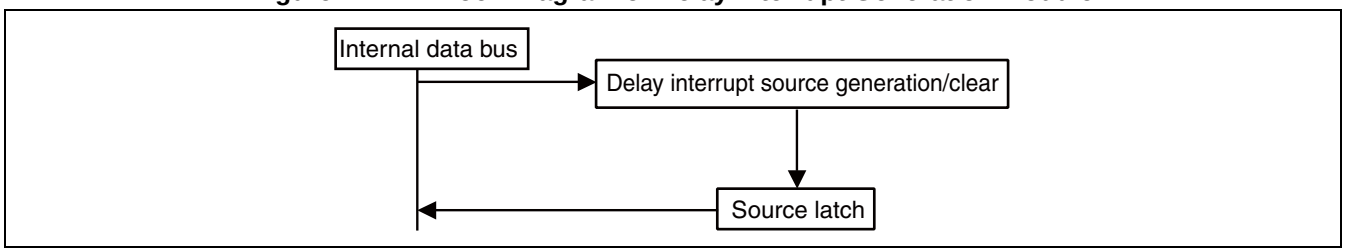
14.1 Overview of Delay Interrupt Generation Module

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 14.1-1 shows a block diagram of the delay interrupt generation module.

Figure 14.1-1 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 14.1-2 Register Configuration of Delay Interrupt Generation Module

DIRR Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
00009F$_H$								D8	XXXXXXX0$_B$
								R/W	

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 14.1-1 lists the interrupts of the delay interrupt generation module and EI^{2}OS.

Table 14.1-1 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

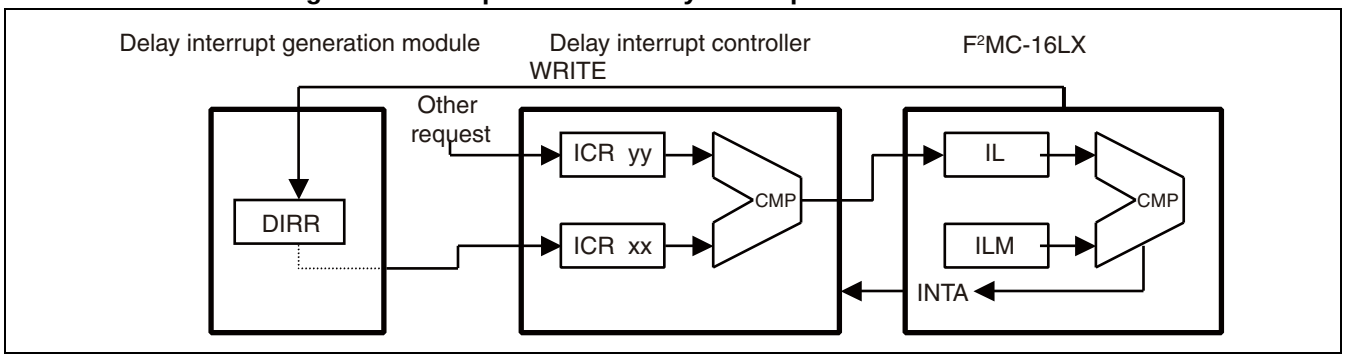
14.2 Operation of Delay Interrupt Generation Module

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 14.2-1 shows the operation of the delay interrupt generation module.

Figure 14.2-1 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.

15. DTP/External Interrupt Circuit

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

15.1 Overview of DTP/External Interrupt Circuit

15.2 Configuration of DTP/External Interrupt Circuit

15.3 Pins of DTP/External Interrupt Circuit

15.4 Registers of DTP/External Interrupt Circuit

15.5 Operations of DTP/External Interrupt Circuit

15.6 Notes on Using the DTP/External Interrupt Circuit

15.7 Sample Programs of DTP/External Interrupt Circuit

15.1 Overview of DTP/External Interrupt Circuit

The DTP (Data Transfer Peripheral)/external interrupt circuit is located between externally connected peripheral units and the F^{2}MC-16LX CPU. This circuit is used to transfer an interrupt request or a data transfer request generated by the peripheral unit to the CPU, generate an external interrupt request, or start the extended intelligent I/O service (EI^{2}OS).

■ DTP/External Interrupt Function

The DTP/external interrupt function uses a signal input to the DTP/external interrupt pin as a start source, which is received by the CPU according to the same procedure as for normal hardware interrupt. This function is used to generate an external interrupt and to start the extended intelligent I/O service (EI^{2}OS).

When an interrupt request is received by the CPU, and the corresponding extended intelligent I/O service (EI^{2}OS) is disabled, the DTP/external interrupt function operates as an external input function and branches to an interrupt routine. When the EI^{2}OS is enabled, the function operates as the DTP function, transfers data automatically via the EI^{2}OS, and branches to an interrupt routine when the specified numbers of data transfer is completed. Table 15.1-1 shows an overview of the DTP/external interrupt.

Table 15.1-1 Overview of DTP/External Interrupt

	External interrupt	DTP function
Input pin	× 8 (P50/INT0, P51/INT1, P53/INT3, P55/INT2, PC0/INT4 to PC3/INT7)	
Interrupt source	Selects the detection level or edge for each pin with the request level register (ELVRH/ELVRL)	
	Inputs either "H" level/"L" level/rising edge/falling edge	Inputs "H" level/"L" level
Interrupt No.	#16 (10_H), #20 (14_H), #24 (18_H), #26 ($1A_H$)	
Interrupt control	Enables or disables an interrupt request output by external interrupt enable register (ENIR)	
Interrupt flag	Retains the interrupt source in the external interrupt source register (EIRR)	
Selection of processing	Sets EI^{2}OS to "disabled" (ICR:ISE=0)	Sets the EI^{2}OS to "enabled" (ICR:ISE=1)
Processing	Branches to the external interrupt processing routine	Branches to the interrupt routine after the specified number of automatic data transfers via the EI^{2}OS is completed

ICR: Interrupt control register

■ Interrupts of DTP/External Interrupt Circuit and EI^{2}OS

Table 15.1-2 lists interrupts of the DTP/external interrupt circuit and the EI^{2}OS.

Table 15.1-2 Interrupts of DTP/External Interrupt Circuit and EI^{2}OS

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
INT0 INT1	#16 (10_H)	ICR02	$0000B2_H$	$FFFFBC_H$	$FFFFBD_H$	$FFFFBE_H$	△
INT2 INT3	#20 (14_H)	ICR04	$0000B4_H$	$FFFFAC_H$	$FFFFAD_H$	$FFFFAE_H$	
INT4	#24 (18_H)	ICR06	$0000B6_H$	$FFFF9C_H$	$FFFF9D_H$	$FFFF9E_H$	
INT5							
INT6	#26($1A_H$)	ICR07	$0000B7_H$	$FFFF94_H$	$FFFF95_H$	$FFFF96_H$	
INT7							

△ : Available if the interrupt requests shared with the registers ICR02 to ICR07 are not used.

15.2 Configuration of DTP/External Interrupt Circuit

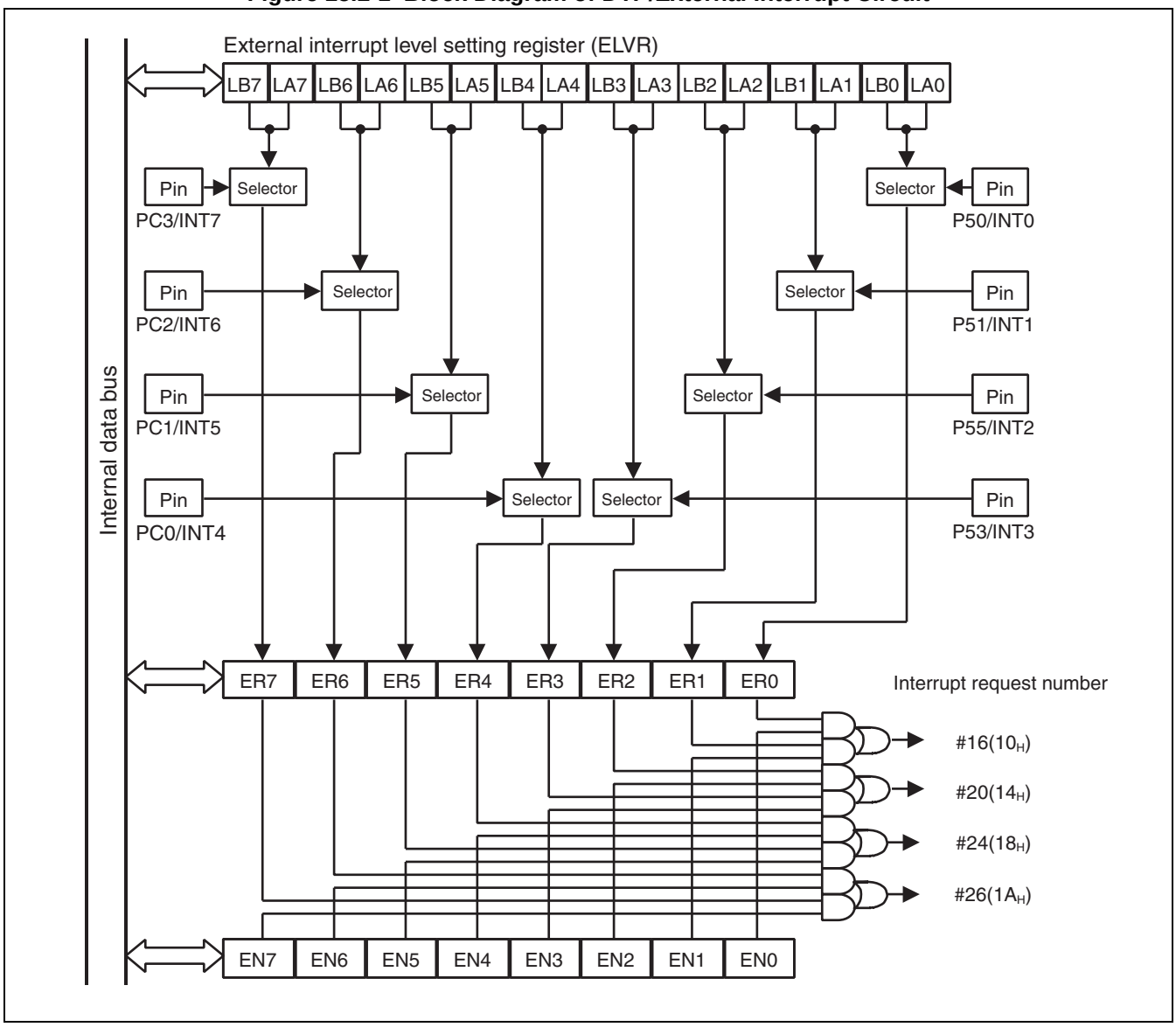
The DTP/external interrupt circuit consists of the following four blocks.

- **DTP/Interrupt input detection circuit**
- **External interrupt level setting register (ELVRL/ELVRH)**
- **External interrupt source register**
- **External interrupt enable register (ENIR)**

■ Block Diagram of DTP/External Interrupt Circuit

Figure 15.2-1 shows a block diagram of the DTP/external interrupt circuit.

Figure 15.2-1 Block Diagram of DTP/External Interrupt Circuit



- DTP/external interrupt input detection circuit

When the level or edge selected for each pin by the external interrupt level setting register (ELVRH/ELVRL) is detected from a pin input signal and then the valid signal is detected. The IR bit of the external interrupt source register (EIRR) corresponding to the pin is set to "1".

- External Interrupt Level Setting Register (ELVRH/ELVRL)

Selects a valid level or edge for each pin.

- External Interrupt Source Register (EIRR)

Retains the DTP/external interrupt source. When there is an external interrupt request flag bit corresponding to each pin and the pin has the valid signal, this register is set to "1".

- External Interrupt Enable Register (ENIR)

Enables/disables an external interrupt for each pin.

15.3 Pins of DTP/External Interrupt Circuit

This section describes the pins of the DTP/external interrupt circuit and their block diagram.

■ Pins of DTP/External Interrupt Circuit

The pins of the DTP/external interrupt circuit are used as both general-purpose ports and other peripheral functions. Table 15.3-1 shows the pin functions, I/O type, and settings for using the DTP/external interrupt circuit.

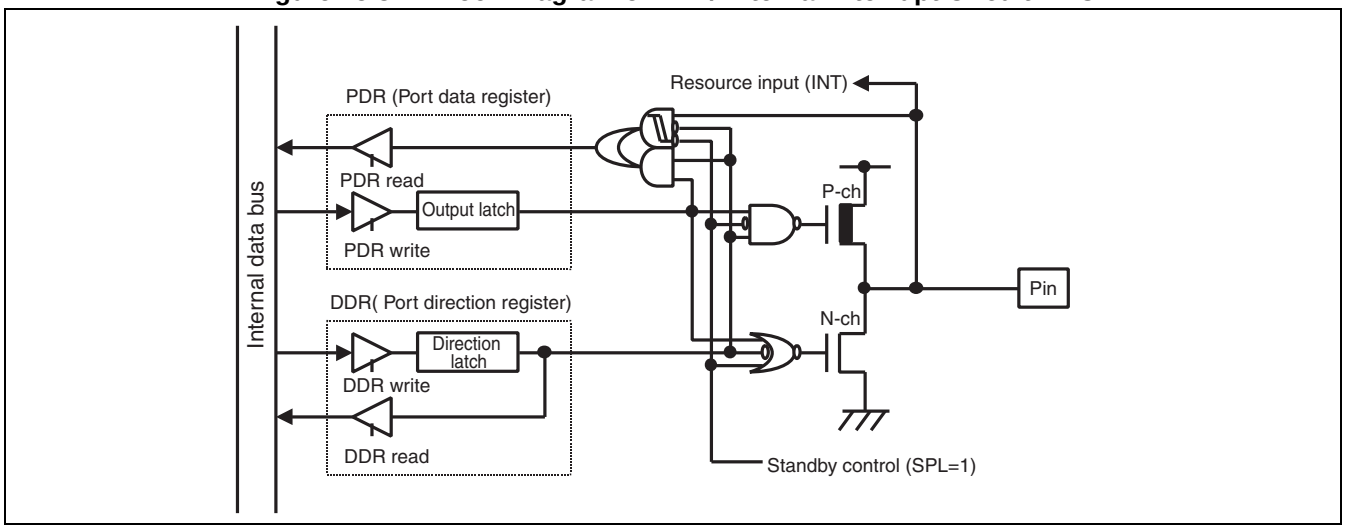
Table 15.3-1 Pins of DTP/External Interrupt Circuit

Pin name	Pin function	I/O type	Pull-up resistor	Standby control	Setting required to use pins
P50/INT0/ ADTG	I/O of Port 5/ External interrupt input	CMOS output/ CMOS hysteresis	None	None	Sets to input port (DDR5: bit0=0)
P51/INT1/ RX1/RX3					Sets to input port (DDR5: bit1=0)
P55/INT2/ RX0/RX2					Sets to input port (DDR5: bit5=0)
P53/INT3					Sets to input port (DDR5: bit3=0)
PC0/INT4/ SIN0	I/O of Port C/ External interrupt input				Sets to input port (DDRC: bit0=0)
PC1/INT5/IN3/ SOT0					Sets to input port (DDRC: bit1=0) UART0 data output disabled
PC2/INT6/IN2/ SCK0					Sets to input port (DDRC: bit2=0) UART0 clock output disabled
PC3/INT7/ SIN1					Sets to input port (DDRC: bit3=0)

■ Block Diagram of DTP/External Interrupt Circuit Pins

Figure 15.3-1 shows a block diagram of the pins of the DTP/external interrupt circuit.

Figure 15.3-1 Block Diagram of DTP/External Interrupt Circuit Pins



15.4 Registers of DTP/External Interrupt Circuit

This section lists the registers of the DTP/external interrupt circuit.

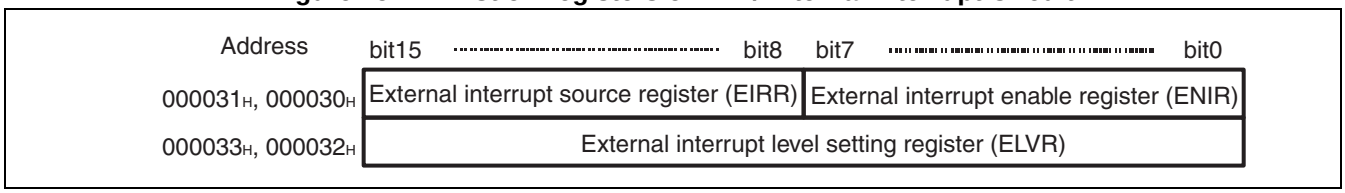
■ Registers of DTP/External Interrupt Circuit

The DTP/external interrupt circuit consists of the following three types.

- External interrupt source register (EIRR)
- External interrupt enable register (ENIR)
- External interrupt level setting register (ELVRH/ELVRL)

Figure 15.4-1 lists the registers of the DTP/external interrupt circuit.

Figure 15.4-1 List of Registers of DTP/External Interrupt Circuit



15.4.1 External Interrupt Source Register (EIRR)

The external interrupt source register (EIRR) is used to retain and clear interrupt sources.

■ External Interrupt Source Register (EIRR)

Figure 15.4-2 shows the configuration of the external interrupt source register. Table 15.4-1 lists the functions of each bit.

Figure 15.4-2 External Interrupt Source Register (EIRR)

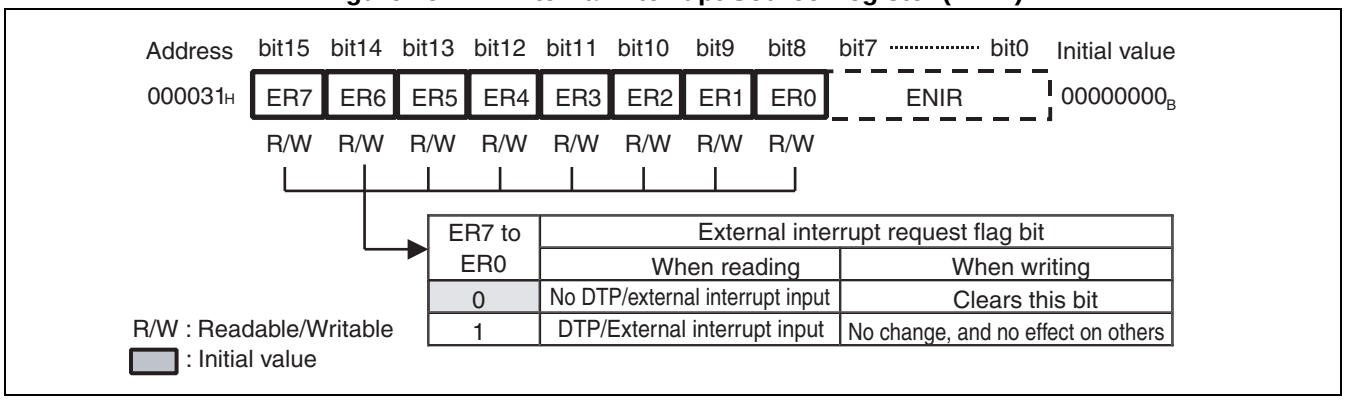


Table 15.4-1 Functions of Each Bit of External Interrupt Source Register (EIRR)

Bit name		Function
bit15 to bit8	ER7 to ER0: External interrupt request flag bit	• Sets to "1" when the edge or level signal selected by the bits (LB7, LA7 to LB0, LA0) of the external interrupt level setting register (ELVRH/ELVRL) is input to the DTP/ external interrupt pin (retaining the interrupt source). • Outputs an interrupt request to the CPU when this bit and the corresponding bits (EN7 to EN0) of the external interrupt enable register (ENIR) are set to "1". • This bit is cleared by writing "0". Writing "1" has no effect on this bit, and the bit remains unchanged. • If the extended intelligent I/O service (EI^{2}OS) is activated, the corresponding external interrupt request flag bit is automatically cleared when one data transfer is completed.

Notes: •When a read-modify-write (RMW) instruction is used, "1" is read.
•When multiple external interrupt request outputs are enabled (ENIR:EN7 to EN0=1), clear only the bits which the CPU accepts an interrupt (the bits set to "1" among ER7 to ER0). No other bits must be cleared unconditionally. The initial value is 00_H. However, the value after reset cancellation depends on the state of the DTP/external interrupt pin.

15.4.2 External Interrupt Enable Register (ENIR)

The external interrupt enable register (ENIR) is used to enable/disable interrupt request output to the CPU.

■ External Interrupt Enable Register (ENIR)

Figure 15.4-3 shows the configuration of the external interrupt enable register (ENIR). Table 15.4-2 lists the functions of each bit.

Also, Table 15.4-3 shows the correspondence among the external interrupt source register (EIRR), the external interrupt enable register (ENIR), and each channel.

Figure 15.4-3 External Interrupt Enable Register (ENIR)

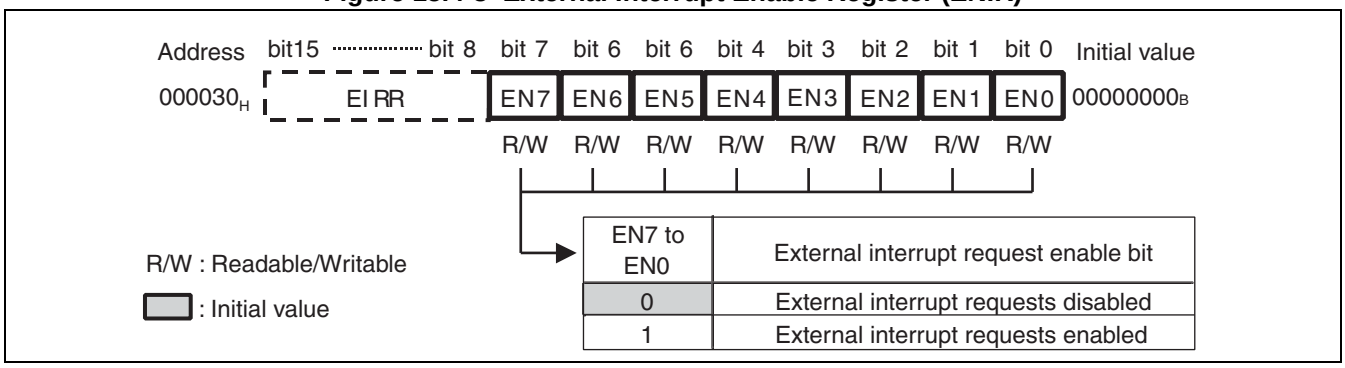


Table 15.4-2 Functions of Each Bit of External Interrupt Enable Register (ENIR)

Bit name		Function
bit7 to bit0	EN7 to EN0: External interrupt request enable bits	Bits to enable/disable interrupt request output to the CPU. When these bits and the corresponding bits (ER7 to ER0) of the external interrupt source register (EIRR) are "1", an interrupt request is output to the CPU. • The state of the DTP/external interrupt pin can be read from the port data register, regardless of the state of the external interrupt request enable bits. • The bits (ER7 to ER0) of the external interrupt source register (EIRR) is set to "1" when an interrupt source is detected, regardless of the value of the external interrupt request enable bits.

Note: To use the DTP/external interrupt pin, write "0" to the corresponding port direction register bit and set the pin as "input".

Table 15.4-3 Correspondence between Channels and DTP/Interrupt Control Register (EIRR, ENIR)

DTP/External interrupt pin	Interrupt No.	External interrupt request flag bit	External interrupt request enable bit
PC3/INT7	#26 ($1A_H$)	ER7	EN7
PC2/INT6	#26 ($1A_H$)	ER6	EN6
PC1/INT5	#24 (18_H)	ER5	EN5
PC0/INT4	#24 (18_H)	ER4	EN4
P53/INT3	#20 (14_H)	ER3	EN3
P55/INT2	#20 (14_H)	ER2	EN2
P51/INT1	#16 (10_H)	ER1	EN1
P50/INT0	#16 (10_H)	ER0	EN0

15.4.3 External Interrupt Level Setting Register (ELVRH/ELVRL)

The external interrupt level setting register (ELVRH/ELVRL) is used to select a signal level or edge type for each pin for detecting that an signal input to the DTP/external interrupt pin is a DTP/external interrupt source.

■ External Interrupt Level Setting Register (ELVRH/ELVRL)

Figure 15.4-4 shows the configuration of the external interrupt level setting register (ELVRH/ELVRL). Table 15.4-4 lists the functions of each bit.

Also, Table 15.4-5 shows the correspondence between each bit of the external interrupt level setting register (ELVRH/ELVRL) and each channel.

Figure 15.4-4 Configuration of External Interrupt Level Setting Register (ELVRH/ELVRL)

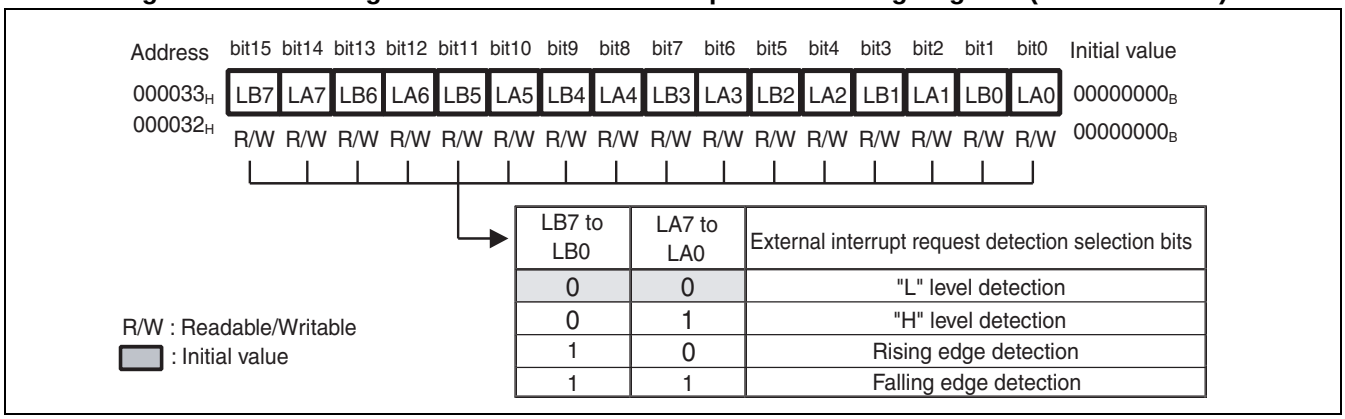


Table 15.4-4 Functions of Each Bit of External Interrupt Level Setting Register (ELVRH/ELVRL)

Bit name		Function
bit15 to bit0	LB7 to LB0, LA7 to LA0: External interrupt request detection selection bits	• Bits to select the signal level or edge type which is input to the DTP/external interrupt pin and used as DTP/external interrupt source. • Two bits are assigned to one pin.

Note: When a selected detection signal is input to the DTP/external interrupt pin, the external interrupt request flag bit is set to "1", regardless of the setting of the external interrupt enable register (ENIR).

Table 15.4-5 Correspondence between Channels and Bits of External Interrupt Level Setting Register (ELVRH/ELVRL)

DTP/external interrupt pin	Interrupt No.	Bit name
PC3/INT7	#26($1A_H$)	LB7, LA7
PC2/INT6	#26($1A_H$)	LB6, LA6
PC1/INT5	#24 (18_H)	LB5, LA5
PC0/INT4	#24 (18_H)	LB4, LA4
P53/INT3	#20 (14_H)	LB3, LA3
P55/INT2	#20 (14_H)	LB2, LA2

Table 15.4-5 Correspondence between Channels and Bits of External Interrupt Level Setting Register (ELVRH/ELVRL)

DTP/external interrupt pin	Interrupt No.	Bit name
P51/INT1	#16 (10_H)	LB1, LA1
P50/INT0	#16 (10_H)	LB0, LA0

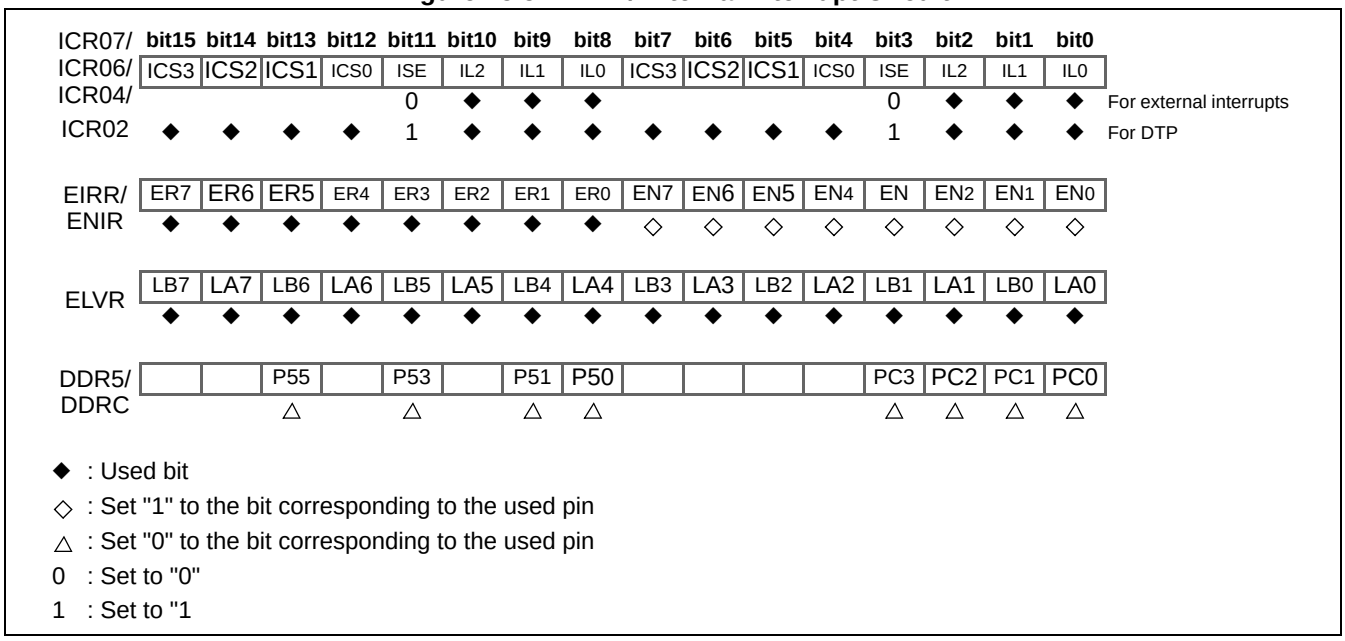
15.5 Operations of DTP/External Interrupt Circuit

The DTP/external interrupt circuit has an external interrupt function and a DTP function. This section describes the settings and operations of each function.

■ Settings of the DTP/External Interrupt Circuit

To operate the DTP/external interrupt circuit, the settings shown in Figure 15.5-1 are required.

Figure 15.5-1 DTP/External Interrupt Circuit



Set the registers of the DTP/external interrupt circuit as follows:

1) Set the input port to the general-purpose I/O port, which is shared with the pin to be used as external interrupt input.
2) Disable the target bit for the external interrupt enable register (ENIR).
3) Set the target bit for the external interrupt level setting register (ELVRH/ELVRL).
4) Clear the target bit for the external interrupt source register (EIRR).
5) Enable the target bit for the external interrupt enable register (ENIR).

Before setting the registers of the DTP/external interrupt circuit, disable the external interrupt request output (ENIR: EN7 to EN0=0) first. To enable the external interrupt request output (ENIR: EN7 to EN0=1), the corresponding interrupt request flag bit must be cleared (EIRR: ER7 to ER0=0) in advance.

This is to avoid accidentally generating an interrupt request when setting the register.

- Switching between external interrupt function and DTP function

To switch between the external interrupt function and the DTP function, set the ISE bit of the corresponding interrupt control register (ICR). When the ISE bit is "1", the extended intelligent service (EI^2OS) is enabled and the DTP function operates. When the ISE bit is "0", the EI^2OS is disabled and only the external interrupt function can operate.

Note:

When multiple interrupts are assigned to one ICR register, all of these interrupt request has the same interrupt level (IL2 to IL0). Using EI^2OS with one interrupt request will generally disable the use of other interrupt requests.

■ DTP/External Interrupt Operation

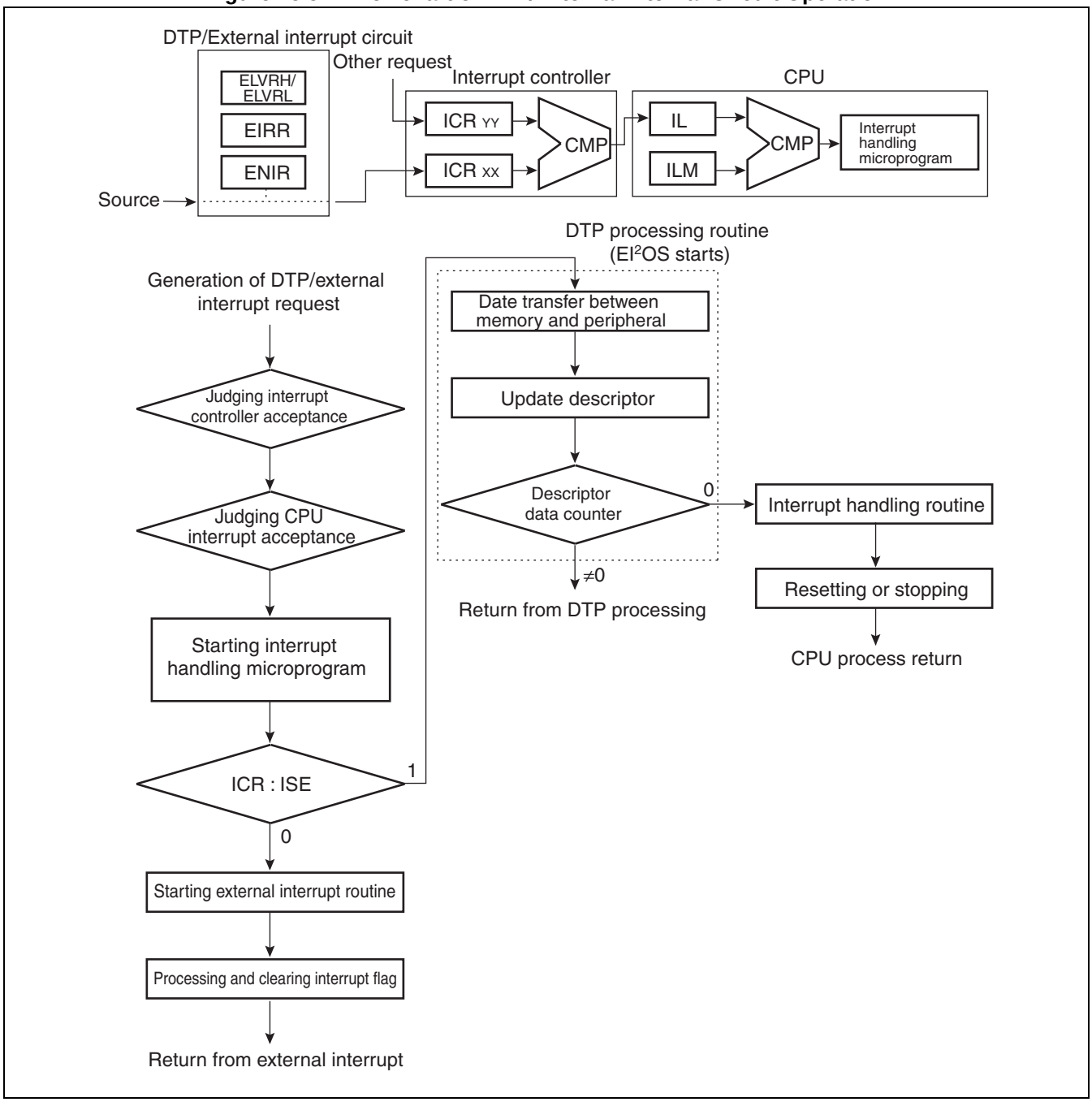
Table 15.5-1 lists the control bits and interrupt sources of the DTP/external interrupt circuit.

Table 15.5-1 Control Bits and Interrupt Sources of DTP/External Interrupt Circuit

	DTP/external interrupt circuit
Interrupt request flag bit	EIRR: ER7 to ER0
Interrupt request enable bit	ENIR: EN7 to EN0
Interrupt source	Valid edge/level input to pins (INT7 to INT0)

If a source specified by the external interrupt level setting register (ELVRH/ELVRL) is input to the corresponding pin after a DTP/external interrupt request is set, this resource generates an interrupt request signal to the interrupt controller. When the ISE bit is "0", an interrupt handling microprogram is executed. When the ISE bit is "1", an extended intelligent I/O service (EI^2OS) processing (DTP processing) microprogram is executed. Figure 15.5-2 shows a flowchart of the DTP/external interrupt circuit operation.

Figure 15.5-2 Flowchart of DTP/External Internal Circuit Operation



15.5.1 External Interrupt Function

The DTP/external interrupt circuit has an external interrupt function that issues an interrupt request by an input to the DTP external interrupt pin with a signal level selected.

■ External Interrupt Function

When a signal (edge or level) selected by the external interrupt level setting register (ELVRH/ELVRL) is detected at the DTP/external interrupt pin, the bits (ER7 to ER0) of the external interrupt source register (EIRR) are set to "1". In such case, when the interrupt request enable bits of the external interrupt enable register (ENIR) is set to "enable" (ENIR: EN7 to EN0=1), generation of an interrupt source is reported to the interrupt controller. The interrupt controller determines the priority of the interrupt level (ICR: IL2 to IL0) for interrupt requests from other peripheral functions and the interrupt priority when multiple interrupt is issued at the same time. The CPU determines the interrupt level mask register (PS: ILM2 to ILM0), the priority of the interrupt level, and the interrupt enable bit (PS: CCR=1). When the interrupt request is accepted by the CPU, an interrupt handling (microprogram) by an internal CPU operation is executed to branch to an interrupt handling routine. Clear the interrupt request by writing "0" to the interrupt request flag bit processed in the interrupt handling routine.

Notes:

- The ER bit is set to "1" when the DTP/external interrupt start source, regardless of the state of the corresponding EN bit.
- When the interrupt routine starts, clear the ER bit that is the start source. While the ER bit is set to "1", the process cannot return from the interrupt. In such case, be sure not to clear any flag bit other than that for the interrupt source unconditionally.

15.5.2 DTP Function

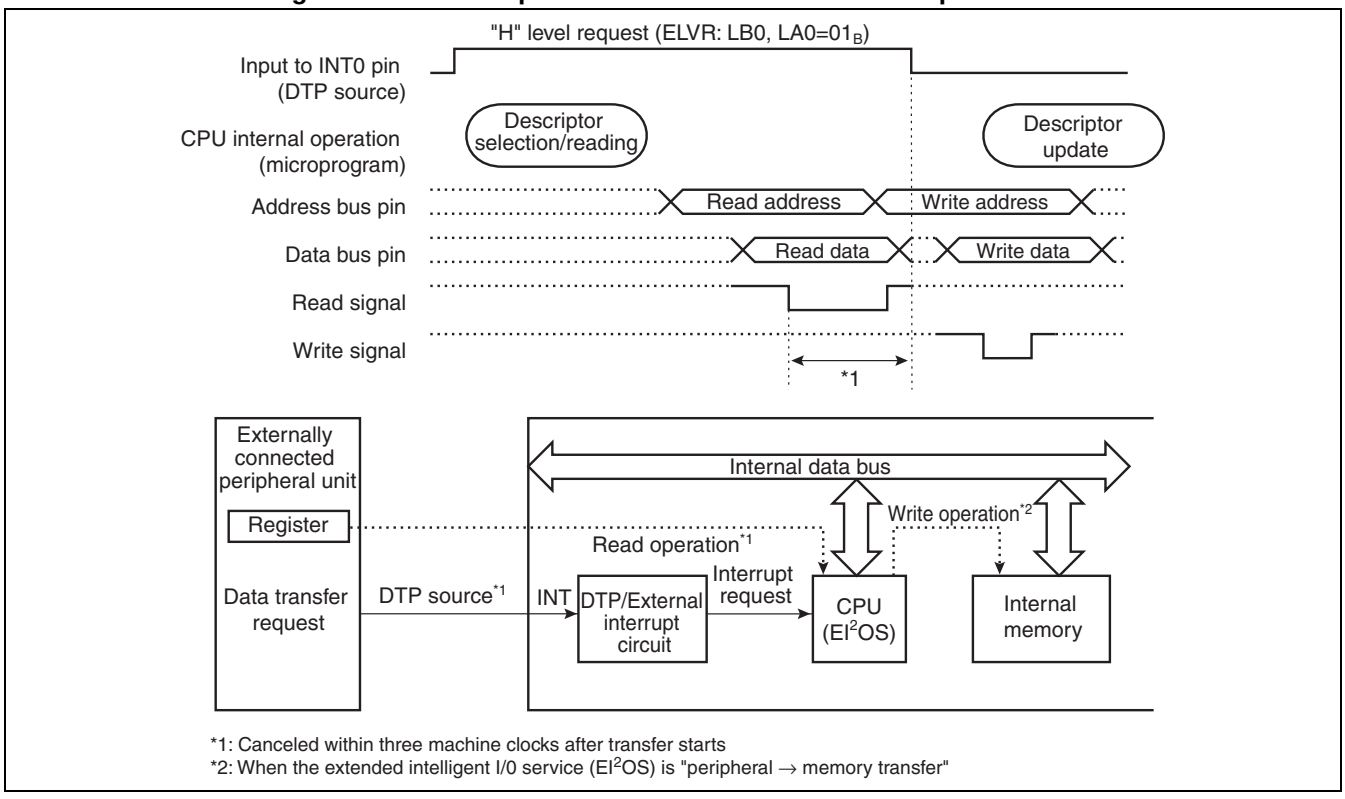
The DTP/external interrupt circuit has a DTP function that detects a signal from the external peripheral unit at the DTP/external interrupt pin to start an extended intelligent I/O service (EI^{2}OS).

■ Operation of DTP Function

The DTP function detects a data transfer request signal from an external peripheral unit, then automatically transfers data between memory and the peripheral unit. An external interrupt function for level detection will start the extended intelligent I/O service (EI^{2}OS). This function operates in the same way as an external interrupt function before the CPU accepts an interrupt request. When EI^{2}OS operation is enabled (ICR: ISE=1) and the interrupt request is accepted, EI^{2}OS is activated, and data transfer is started. When one data transfer is completed, the function updates the descriptor clears the interrupt request, and prepares for the next request from the pin. When all transfer by EI^{2}OS is completed, the process is branched to the interrupt handling routine.

Figure 15.5-3 shows an example of the interface between memory and an external peripheral unit.

Figure 15.5-3 Example of Interface with External Peripheral Unit



Note:

The external peripheral unit must cancel only the level of the data transfer request signal (DTP

source) within three machine clocks after initial transfer starts.

15.6 Notes on Using the DTP/External Interrupt Circuit

This section provides notes on the signals input to the DTP/external interrupt circuit, and explains how to clear standby mode and interrupts.

■ Notes on Using the DTP/External Interrupt Circuit

● Conditions for peripheral units externally connected when using the DTP function

The externally connected peripheral units where the DTP function can be support must be able to automatically clear a data transfer request when the transfer starts. Unless a transfer request is canceled within three machine clock cycles after transfer operation starts, the DTP/external interrupt circuit assumes that a new transfer request has been generated.

● Input polarity of external interrupts

When the external interrupt level setting register (ELVRH/ELVRL) is set to "edge detection", the pulse width must be at least 3 machine clocks in order to detect that the edge has been input.

When the request input level is set to "level setting", the pulse width must be at least 3 machine clock cycles. Even if the external interrupt source register (EIRR) is cleared, the request to the interrupt controller is continuously generated as long as the interrupt input pin remains active level.

When the "level detection" is set and a level to be an interrupt source is input, the source F/F in the external interrupt source register (EIRR) is set to "1", and the source is retained as shown in Figure 15.6-1. Therefore, even if the source is canceled, the request to the interrupt controller remains active as long as the interrupt request output is enabled. To cancel the request to the interrupt controller, clear the external interrupt request flag bit to clear the source F/F as shown in Figure 15.6-2.

Figure 15.6-1 Clearing the Source Hold Circuit When a Level is Set

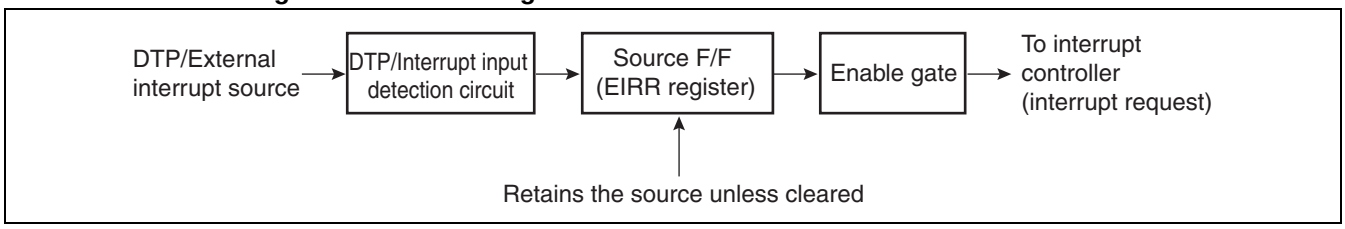
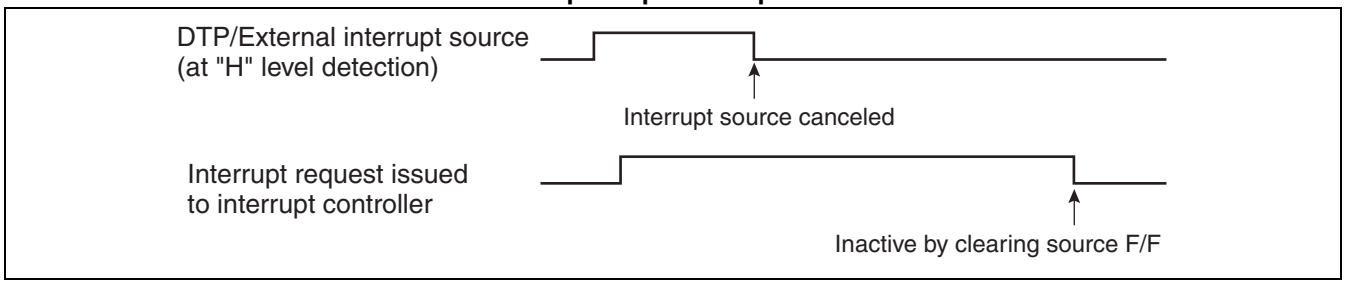


Figure 15.6-2 Relationship between DTP/External Interrupt Source and Interrupt Request when Interrupt Request Output is Enabled



- Notes on interrupts

When the external interrupt function is active and the interrupt request output is enabled with the request flag bit "1", the process cannot return from the interrupt handling. Be sure to clear the external interrupt request flag bit in the interrupt handing routine. When the DTP function is active, EI^{2}OS automatically clears this flag. When a level detection is active, the external interrupt request flag bit is immediately set again, even if the bit was cleared, as long as the level to be an interrupt source is retained. Disable the interrupt request output or cancel the interrupt source as required.

- Notes in standby mode

When an evaluation product transfers to the standby mode or returns from it, the DTP/ external interrupt request flag bit in external interrupt source register is set (EIRR: ER) no matter whether the external interrupt is set enable or disable (ENIR: EN=1/0).

15.7 Sample Programs of DTP/External Interrupt Circuit

This section provides sample programs of the external interrupt function and the DTP function.

■ Sample Program of External Interrupt Function

● Processing specifications

The program detects the rising edge of pulses input to the INT0 pin, and generates an external interrupt.

[Coding example]

```
ICR02 EQU   0000B2H            ; Interrupt control register for
                               ; External external interrupt circuit
DDR5  EQU   000015H            ; Port 5 direction register
ENIR  EQU   000030H            ; External interrupt enable register
EIRR  EQU   000031H            ; External interrupt source register
ELVRL EQU   000032H            ; External interrupt level setting register
ELVRH EQU   000033H            ; External interrupt level setting register
ER0   EQU   EIRR:0             ; INT0 interrupt flag bit
EN0   EQU   ENIR:0             ; INT0 interrupt enable bit
;----------Main program----------------------------------------------
CODE  CSEG
START:
;     :                        ; Stack pointer (SP) already initialized
      MOV   I:DDR5, #00000000B ; Set DDR5 to "input"
      AND   CCR, #0BFH         ; Interrupt disabled
      MOV   I:ICR02, #00H      ; Interrupt level 0 (highest), EI2OS
                               ; disabled
      CLRB  EN0                ; Disable INT0 with ENIR
      MOV   I:ELVR, #00000010B ; Select rising edge for INT0
      CLRB  I:ER0              ; Clear the source of INT0 with EIRR
      SETB  I:EN0              ; Enable INT0 with ENIR
      MOV   ILM, #07H          ; Set ILM in PS to level 7
      OR    CCR, #40H          ; Interrupt enabled
LOOP: MOV   A, #00H            ; Infinite loop
      MOV   A, #01H
      BRA   LOOP
;----------Interrupt program-----------------------------------------
WARI:
      CLRB  ER0                ; Clear the interrupt request flag
;     :
;     User processing
;     :
```

```
      RETI                    ; Return from interrupt.
CODE  ENDS
;----------Vector setting----------------------------------------------
VECT  CSEG  ABS=0FFH
      ORG   0FFBCH            ; Set a vector for the interrupt
                              ; #16 (10H)
      DSL   WARI
      ORG   0FFDCH            ; Reset vector setting
      DSL   START
      DB    00H               ; Set to single-chip mode
VECT  ENDS
      END   START
```

■ Sample Program of DTP Function

● Processing specifications

The program detects the "H" level of signal input to the INT0 pin and activate ch.0 of the extended intelligent I/O service (EI^2OS).

The DTP processing (EI^2OS) outputs the data in RAM to port 0.

[Coding example]

```
ICR02 EQU   0000B2H           ; Interrupt control register for
                              ; DTP/external interrupt circuit
DDR0  EQU   000010H           ; Port 0 direction register
DDR5  EQU   000015H           ; Port 5 direction register
ENIR  EQU   000030H           ; External interrupt enable register
EIRR  EQU   000031H           ; External interrupt source register
ELVRL EQU   000032H           ; External interrupt level setting register
ELVRH EQU   000033H           ; External interrupt level setting register
ER0   EQU   EIRR:0            ; INT0 interrupt flag bit
EN0   EQU   ENIR:0            ; INT0 interrupt enable bit
BAPL  EQU   000100H           ; Lower bits of buffer address pointer
BAPM  EQU   000101H           ; Middle bits of buffer address pointer
BAPH  EQU   000102H           ; Upper bits of buffer address pointer
ISCS  EQU   000103H           ; EI2OS status register
IOAL  EQU   000104H           ; Lower bits of I/O address register
IOAH  EQU   000105H           ; Upper bits of I/O address register
DCTL  EQU   000106H           ; Lower bits of data counter
DCTH  EQU   000107H           ; Upper bits of data counter
;----------Main program-----------------------------------------------
CODE  CSEG
START:
;     :                       ; Stack pointer (SP) already initialized
      MOV   I:DDR0, #11111111B ; Set DDR0 to "output"
      MOV   I:DDR5, #00000000B ; Set DDR5 to "input"
```

```
      AND   CCR, #0BFH          ;Interrupt disabled
      MOV   I:ICR02, #08H       ; Interrupt level 0 (highest)
                                ; EI2OS enable, ch.0
      MOV   BAPL, #00H          ; Set output data address
      MOV   BAPM, #06H          ;
      MOV   BAPH, #00H          ;
MOV   ISCS, #12H                ; Byte transfer, I/O address fixed,
                                ; buffer address + 1, memory → I/O
      MOV   IOAL, #00H          ; As transfer destination address
      MOV   IOAH, #00H          ; pointer, specify port 0 (PDR0)
      MOV   DCTL, #0AH          ; Transfer count: 10 times
      MOV   DCTH, #00H          ;
      CLRB  I:EN0               ; Disable INT0 with ENIR
      MOV   I:ELVR, #00000001B ; Select "H" level for INT 0
      CLRB  I:ER0               ; Clear the source of INT0 with EIRR
      SETB  I:EN0               ; Enable INT0 with ENIR
      MOV   ILM, #07H           ; Set ILM in PS to level 7
      OR    CCR, #40H           ; Interrupt enabled
LOOP: MOV   A, #00H             ; Infinite loop
      MOV   A, #01H             ;
      BRA   LOOP                ;
;----------Interrupt program-----------------------------------------
WARI:
      CLRB  I:ER0               ; Clear the interrupt request flag
;     :                         ; Switch the channel, change the transfer
                                ; address as required
;     User processing           ; Set the processing again, such as the
                                ; termination of EI2OS
                                ; To terminate the processing,
;     :                         ; interrupt must be disabled
      RETI                      ; Return from interrupt
CODE  ENDS
;----------Vector setting--------------------------------------------
VECT  CSEG  ABS=0FFH
      ORG   0FFBCH              ; Set a vector for the interrupt
                                ; #16 (10H)
      DSL   WARI
      ORG   0FFDCH              ; Reset vector setting
      DSL   START
      DB    00H                 ; Set to single-chip mode
VECT  ENDS
      END   START
```

16. 8-/10-Bit A/D Converter

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

16.1 Overview of 8-/10-bit A/D Converter

16.2 Block Diagram of 8-/10-bit A/D Converter

16.3 Configuration of 8-/10-bit A/D Converter

16.4 Interrupts of 8-/10-bit A/D Converter

16.5 Explanation of 8-/10-bit A/D Converter Operations

16.6 Precautions when Using 8-/10-bit A/D Converter

16.1 Overview of 8-/10-bit A/D Converter

The 8-/10-bit A/D converter converts analog input voltages to 8-bit or 10-bit digital values in the RC successive approximation conversion method.

- **Up to 8 channels of analog input pins can be selected for input signals.**
- **The software trigger, internal timer output or external trigger can be selected as a start trigger.**

■ Functions of 8-/10-bit A/D Converter

This converter converts an analog voltage (input voltage) which is input to the analog input pin to an 8-bit or 10-bit digital value (A/D conversion).

The 8-/10-bit A/D converter has the following functions:

- The minimum A/D conversion time is 1.4 µs * per channel, including the sampling time.
- The minimum sampling time is 0.5 µs * per channel.
- The RC successive approximation conversion method with a sample and hold circuit is used as the conversion method.
- 8-bit or 10-bit resolution can be specified.
- Up to 8 channels can be used as analog input pins.
- Interrupt requests can be generated by storing A/D conversion results in the A/D data register.
- When an interrupt request is generated, EI^2OS can be started.
- The software, internal timer output or external trigger (falling edge) can be selected as a start trigger.

*: When the frequency of the machine clock is 32MHz, and $AV_{CC} \geq 4.5$ V.

■ Conversion Modes of 8-/10-bit A/D Converter

The 8-/10-bit A/D converter has the following conversion modes:

Table 16.1-1 Conversion Modes of 8-/10-bit A/D Converter

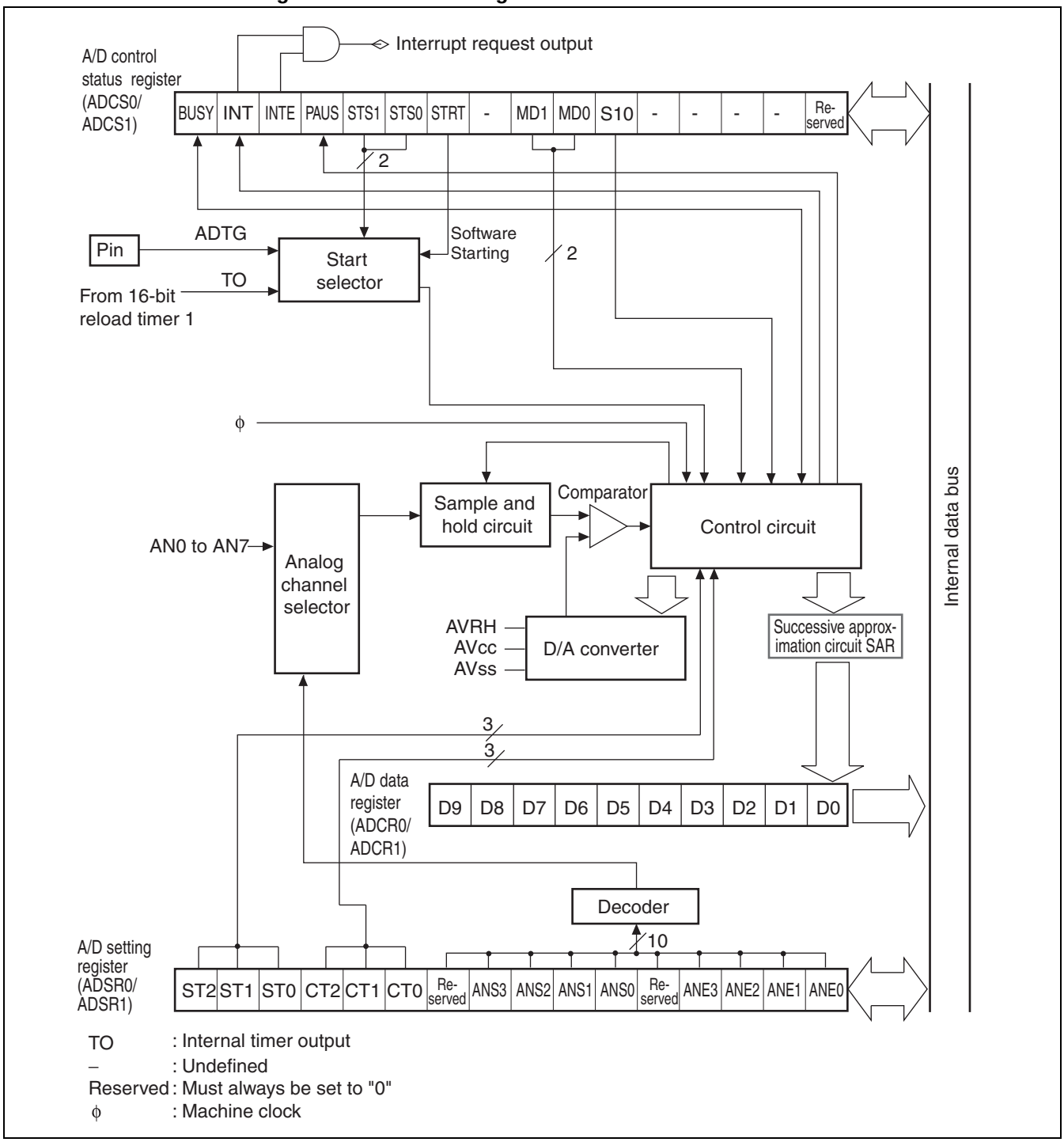
Conversion mode	Description
Single conversion mode	A/D conversion is performed sequentially from the start channel to the end channel. Once the A/D conversion is completed for the end channel, the A/D conversion function is stopped.
Continuous conversion mode	A/D conversion is performed sequentially from the start channel to the end channel. When the A/D conversion for the end channel is completed, the conversion sequence returns to the start channel to continue the A/D conversion operation.
Stop conversion mode	A/D conversion is performed while stopping after each channel is processed. Once the A/D conversion is completed for the end channel, the conversion sequence returns to the start channel to repeat the A/D conversion and stop operation.

16.2 Block Diagram of 8-/10-bit A/D Converter

The 8-/10-bit A/D converter consists of the following blocks.

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

Figure 16.2-1 Block Diagram of 8-/10-bit A/D Converter



- Detailed descriptions of pins and other items in the block diagram

Table 16.2-1 lists the actual pin names and interrupt request numbers of the 8-/10-bit A/D converter.

Table 16.2-1 Pins and Interrupt Request Numbers in Block Diagram

Pin name/Interrupt request number in block diagram		Actual pin name/interrupt request number
ADTG	Trigger input pin	P50/ADTG
TO	Internal timer output	Output of 16-bit reload timer 1
AN0 to AN7	Analog input pins ch.0 to ch.7	P60/AN0 to P67/AN7
AVRH	Vref+ Input pin	AVRH
AV_{CC}	V_{CC} Input pin	AV_{CC}
AV_{SS}	V_{SS} Input pin	AV_{SS}
Interrupt request output	Interrupt request output	#32 (20_H)

- A/D control status register (ADCS0, ADCS1)

This register starts A/D conversion by software, selects the start trigger for the A/D conversion, selects the conversion mode, enables or disables interrupt requests, checks and clears the interrupt request flag, pauses A/D conversion operation, checks the state during the conversion, and selects the resolution type.

- Successive approximation circuit (SAR)

Successive approximation is performed for each bit and the conversion results are stored. When the next A/D conversion starts, the results of the current A/D conversion stored in the circuit are destroyed.

- A/D data register (ADCR0, ADCR1)

The A/D conversion results are stored in the successive approximation circuit for each bit separately during the conversion and then stored in this register when the A/D conversion is completed, and the results are confirmed. This register can be used to read the A/D conversion results.

- A/D setting register (ADSR0, ADSR1)

This register sets up the start channel and end channel for A/D conversion as well as sets the compare time and sampling time for A/D conversion.

- Start selector

This selects the type of triggers used to start A/D conversion. The internal timer output or external pin input can be specified as the start trigger.

- Decoder

The decoder selects an analog input pin to be used for A/D conversion from the setting of the A/D conversion start channel select bits (ADSR: ANS3 to ANS0) and A/D conversion end channel select bits (ADSR: ANE3 to ANE0) in the A/D setting register.

- Analog channel selector

This selector selects a pin to be used for A/D conversion from the 8 channels of analog input pins after receiving a signal from the decoder.

- Sample and hold circuit

This circuit holds the input voltage selected by the analog channel selector. As the circuit maintains the input voltage generated immediately after start of A/D conversion, the conversion can be performed without being affected by input voltage changes during the A/D conversion process.

- D/A converter

This converter generates a reference voltage which is compared with the input voltage held in the sample and hold circuit.

- Comparator

The comparator compares the input voltage held in the sample and hold circuit with the output voltage of the D/A converter to determine which is higher/lower.

- Control circuit

This circuit determines an A/D conversion value after receiving the higher/lower signal from the comparator. Once the conversion results are confirmed, the data from these conversion results is stored in the A/D data register. When the output of interrupt requests is enabled, an interrupt is generated.

16.3 Configuration of 8-/10-bit A/D Converter

The pins, registers and interrupt sources of the A/D converter are shown below.

■ Pins of 8-/10-bit A/D Converter

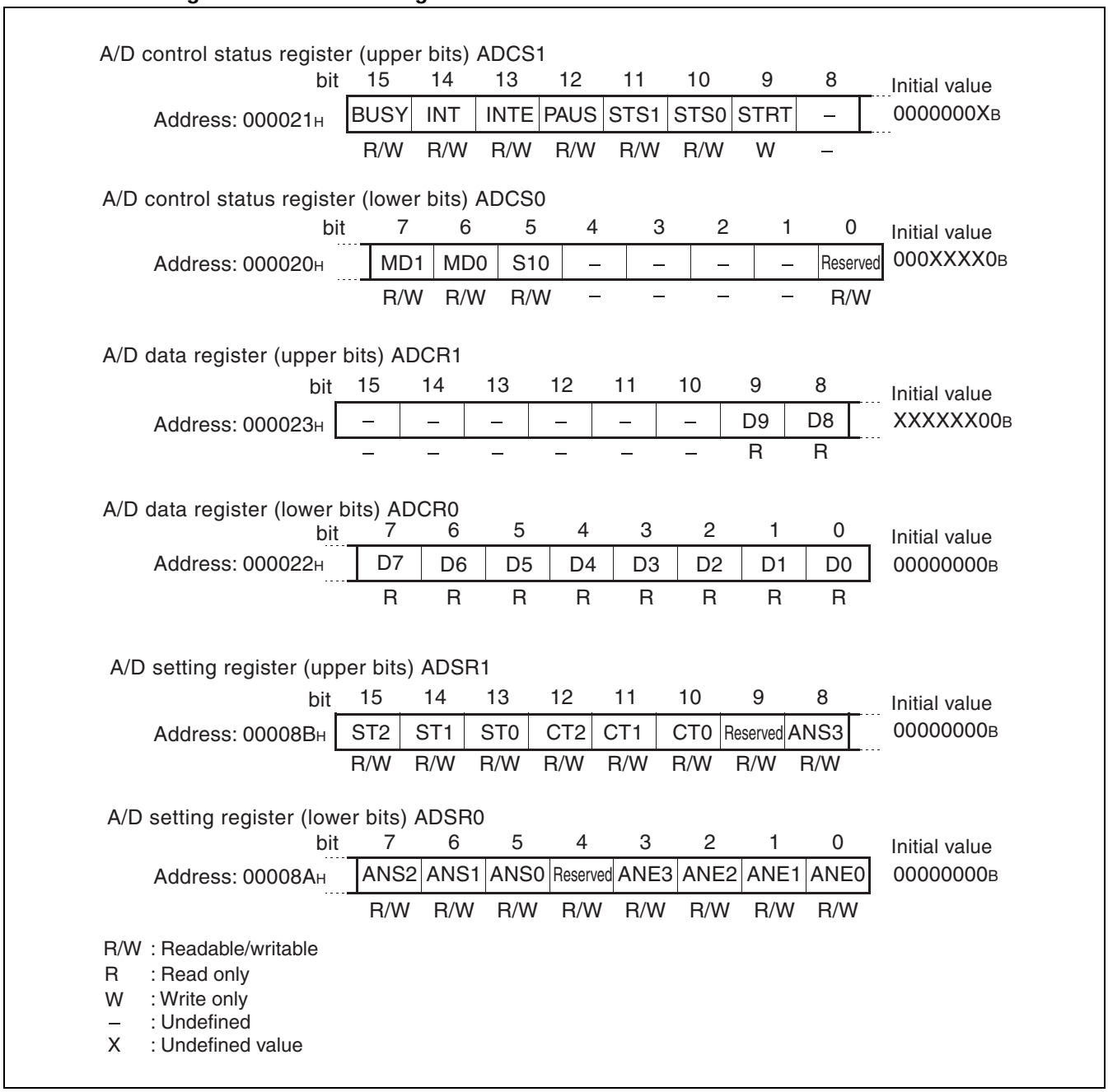
The pins of the 8-/10-bit A/D converter are also used as general-purpose I/O ports. Table 16.3-1 lists the functions of each pin and settings for when using the 8-/10-bit A/D converter.

Table 16.3-1 Pins of 8-/10-bit A/D Converter

Function name	Pin name	Pin function	Setting for when using 8-/10-bit A/D converter
Trigger input	P50/ADTG/INT0	General-purpose I/O port/ External trigger input/ External interrupt 0 input	Set as input port by port direction register (DDR5)
ch.0	P60/AN0	General-purpose I/O port/ Analog inputs/ PPG output	Enable analog signal input (ADER6: Corresponding bits of ADE7 to ADE0 set to "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		

■ List of Registers and Initial Values of 8-/10-bit A/D Converter

Figure 16.3-1 List of Registers and Initial Values of 8-/10-bit A/D Converter



16.3.1 Upper bits in the A/D control status register (ADCS1)

The upper bits in the A/D control status register (ADCS1) enable the following settings:

- **Starting A/D conversion by software**
- **Selecting the start trigger for A/D conversion**
- **Enabling or disabling the interrupt requests by storing A/D conversion results into the A/D data register**
- **Checking and clearing the interrupt request flag by storing A/D conversion results into the A/D data register**
- **Pausing A/D conversion operation and checking the state during the conversion**

■ Upper Bits in the A/D Control Status Register (ADCS1)

Figure 16.3-2 Upper Bits in the A/D Control Status Register (ADCS1)

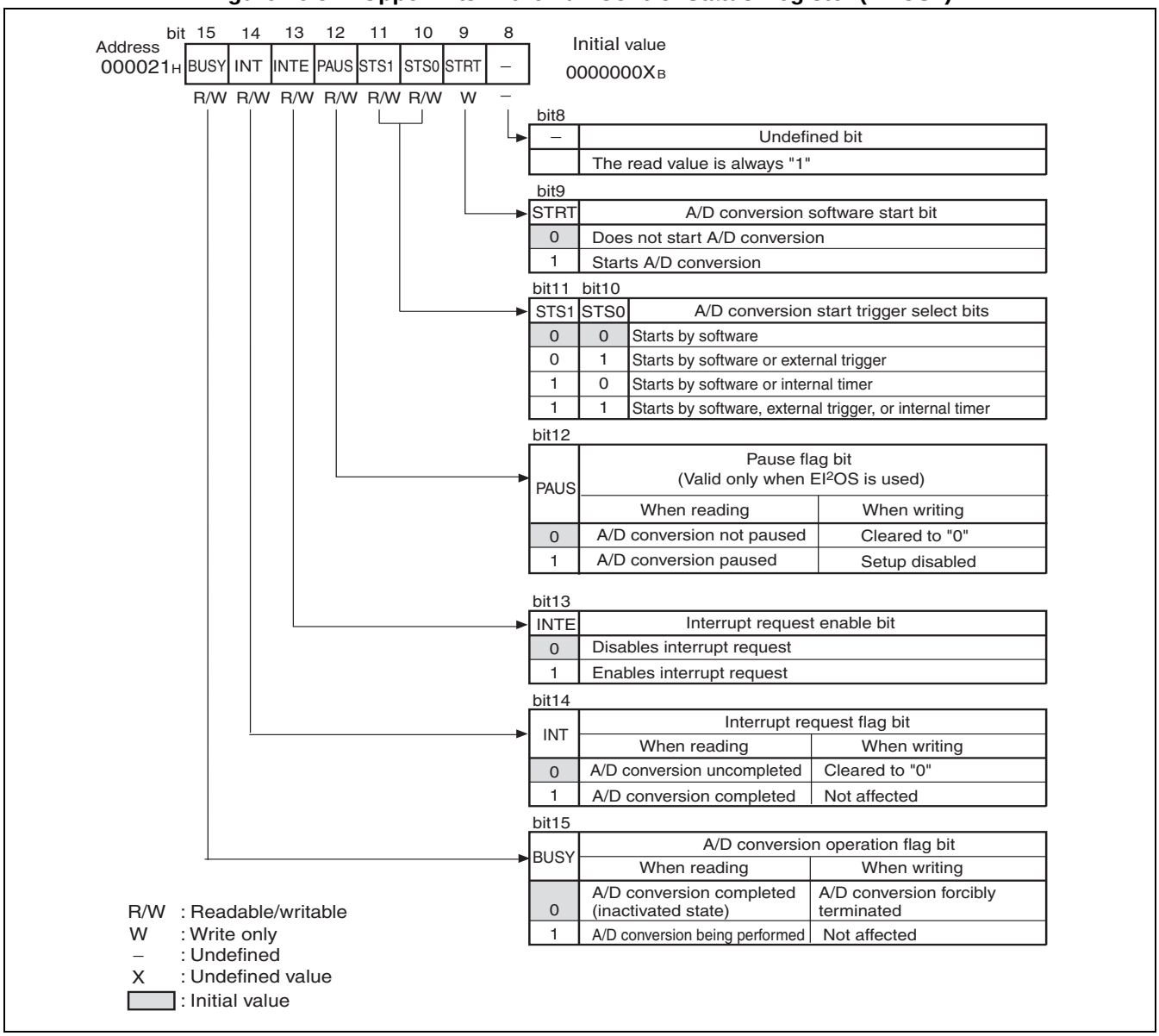


Table 16.3-2 Functions of Upper Bits in the A/D Control Status Register (ADCS1) (Sheet 1 of 3)

Bit name		Function
bit15	BUSY: A/D conversion operation flag bit	This bit forcibly terminates the 8-/10-bit A/D converter. When read, it indicates whether the 8-/10-bit A/D converter is running or stopped. **When set to "0":** Forcibly terminates the 8-/10-bit A/D converter. **When set to "1":** Does not affect the operation. **When read:** "1" is read if the 8-/10-bit A/D converter is running, while "0" is read if the converter is stopped. "1" is also read when in "halt state" in the stop conversion mode. Notes: • "1" is read by the read-modify-write (RMW) instruction. • In the single conversion mode, this bit is cleared upon the completion of A/D conversion. • In the continuous conversion and the stop conversion modes, this bit is not cleared until stopped by writing "0". • Do not forcibly terminate A/D converter operation (BUSY=0) and start it (by software (STRT = 1)/ external trigger/ timer) at the same time.
bit14	INT: Interrupt request flag bit	This bit indicates the generation of an interrupt request. • When A/D conversion is completed and the A/D conversion results are stored in the A/D data register (ADCR), the INT bit is set to "1". • When interrupt requests are enabled (INTE = 1) and the interrupt request flag bit is set (INT = 1), an interrupt request is generated. • This bit is cleared when "0" is written to it. Alternatively, it is automatically cleared when the data transfer of the A/D conversion results is completed by EI^{2}OS. **When set to "0":** Cleared **When set to "1":** Not affected Note: "1" is read by the read-modify-write (RMW) instruction.
bit13	INTE: Interrupt request enable bit	This bit enables or disables the output of interrupt requests. When interrupt requests are enabled (INTE = 1) and the interrupt request flag bit is set (INT = 1), an interrupt request is generated. Note: When transferring the A/D conversion results by EI^{2}OS, always set the bit to "1".

Table 16.3-2 Functions of Upper Bits in the A/D Control Status Register (ADCS1) (Sheet 2 of 3)

Bit name		Function
bit12	PAUS: Pause flag bit	The PAUS bit indicates that the A/D conversion data protection function has been activated. The PAUS bit is valid only when the output of interrupt requests is enabled (ADCS: INTE = 1). **When the A/D conversion data protection function is activated:** Set to "1" **When set to "0":** Cleared to"0" **When set to "1":** Set to"1" • When the output of interrupt requests is enabled (ADCS: INTE = 1) and A/D conversion is performed, an interrupt request is generated upon setup of the interrupt request flag bit (ADCS: INT) every time the A/D conversion session is completed. If the next A/D conversion session is completed without clearing the interrupt request flag bit (ADCS: INT) beforehand, the A/D conversion operation is paused to prevent the previous data from being overwritten (A/D conversion data protection function). When A/D conversion operation 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 status and restarts the A/D conversion operation. • The interrupt request flag bit (ADCS: INT) is cleared by writing "0". Alternatively, when the A/D data register is set to transfer the A/D conversion results by EI^2OS, the interrupt request flag bit (ADCS: INT) is cleared by EI^2OS upon the completion of transfer of the A/D conversion results. Notes: • For the A/D conversion data protection function, see Section "16.5.5 A/D Conversion Data Protection Function". • Even when the pause status is released, the PAUS bit is not cleared automatically. To clear the PAUS bit, write "0".
bit11, bit10	STS1, STS0: A/D conversion start trigger select bits	These bits select the type of triggers used to start the 8-/10-bit A/D converter (start trigger). • 00_B : Starting by software • 01_B : Starting by external pin trigger/software • 10_B : Starting by 16-bit reload timer/software • 11_B : Starting by external pin trigger/16-bit reload timer/software Notes: • When external pin trigger is selected (01_B, 11_B): A/D conversion starts when the falling edge is detected at the ADTG pin. • When 16-bit reload timer is selected (10_B, 11_B): A/D conversion starts when the output of the 16-bit reload timer 1 becomes "1". • When more than one start triggers are selected (Other than STS1, STS0 = 00_B): the 8-/10-bit A/D converter is started by the first start trigger generated. • Change start trigger settings, when the operations of the peripheral functions used to generate a start trigger are stopped (inactive trigger state).
bit9	STRT: A/D conversion software start bit	This bit starts the 8-/10-bit A/D converter by software. **When set to "1":** 8-/10-bit A/D converter starts. • When A/D conversion operation is paused in the stop conversion mode, the A/D conversion operation is restarted by writing "1" to the STRT bit. **When set to "0":** Invalid. No changes Notes: • "0" is read by the read-modify-write (RMW) instruction. • When read by an instructions other than read-modify-write (RMW) instructions, "1" is read rather than the written value. • Do not forcibly terminate the 8-/10-bit A/D converter (BUSY = 0) and start it by software (STRT = 1) at the same time.

Table 16.3-2 Functions of Upper Bits in the A/D Control Status Register (ADCS1) (Sheet 3 of 3)

Bit name		Function
bit8	Undefined bit	• When read: "1" is always read. • When write: Not affected.

16.3.2 Lower Bits in the A/D Control Status Register (ADCS0)

The lower bits in the A/D control status register enable the following settings:

- **Selecting the A/D conversion mode**
- **Selecting the start and end channels of the A/D conversion**

■ Lower Bits in the A/D Control Status Register (ADCS0)

Figure 16.3-3 Lower Bits in the A/D Control Status Register (ADCS0)

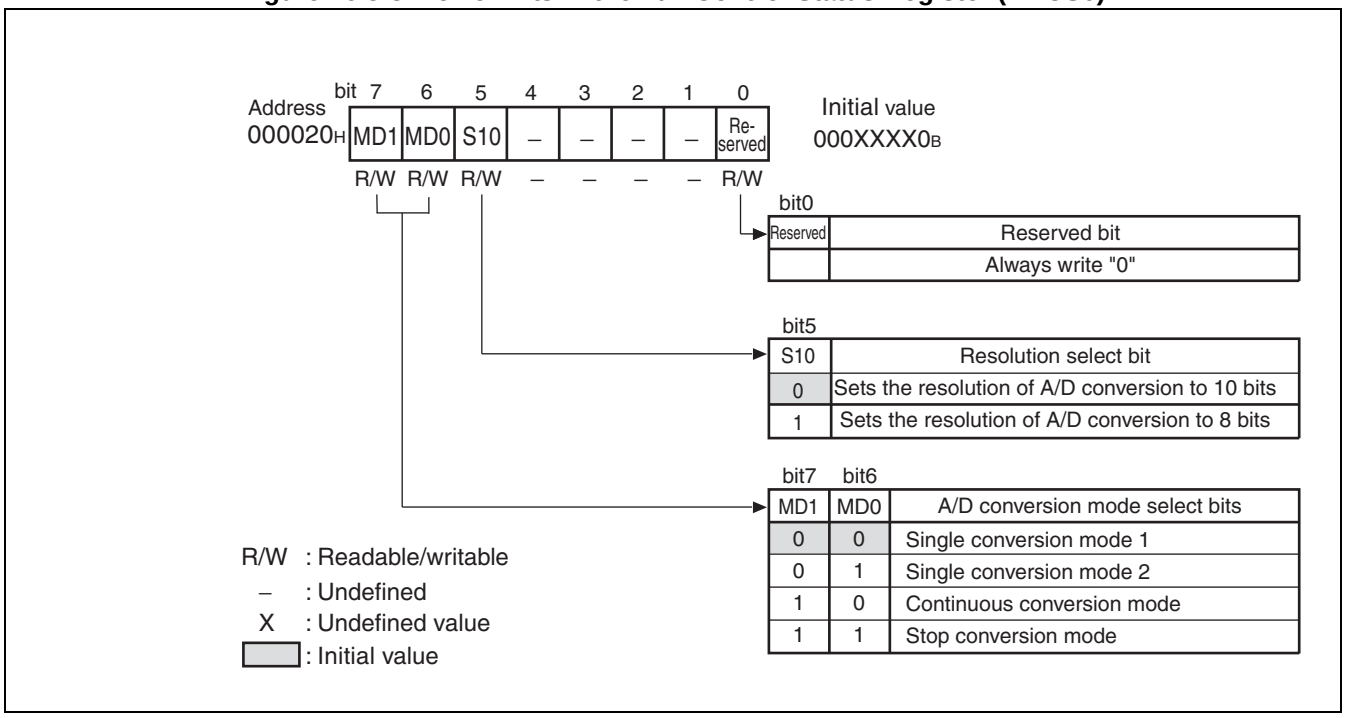


Table 16.3-3 Functions of Lower Bits in the A/D Control Status Register (ADCS0)

Bit name		Function
bit7, bit6	MD1, MD0: A/D conversion mode select bits	These bits set the A/D conversion mode. For detailed information about how to use each mode, see Section "16.5 Explanation of 8-/10-bit A/D Converter Operations". **In single conversion modes 1 and 2:** • A/D conversion is performed on the analog input from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) in succession. • The A/D conversion operation will stop when the A/D conversion for the end channel is completed. • For the difference between single conversion modes 1 and 2, see Section "16.5 Explanation of 8-/10-bit A/D Converter Operations". **In continuous conversion mode:** • A/D conversion is performed on the analog input from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) in succession. • When the A/D conversion is completed for the end channel, the conversion sequence returns to the analog input of the start channel to continue the A/D conversion. **In stop conversion mode:** • A/D conversion starts from the start channel (ADSR: ANS3 to ANS0). The A/D conversion operation will stop every time the A/D conversion for a channel is completed. If a start trigger is entered while the A/D conversion operation is still stopped, A/D conversion for the next channel will be performed. • The A/D conversion operation will stop when the A/D conversion for the end channel is completed. If a start trigger is entered while the conversion operation is still stopped, the conversion sequence will return to the analog input of the start channel to continue the A/D conversion. Note: When changing the conversion mode, make sure that A/D conversion has not already started (stop state).
bit5	S10: Resolution select bit	This bit sets the resolution of A/D conversion. **When set to "0":** Resolution of A/D conversion is set to A/D conversion data bits D9 to D0 (10 bits) **When set to "1":** Resolution of A/D conversion is set to A/D conversion data bits D7 to D0 (8 bits) Note: When changing bit S10, make sure that A/D conversion has not already started (stop state). If bit S10 is changed after the A/D conversion has already started, the conversion results stored in the A/D conversion data bits (D9 to D0) will become invalid.

16.3.3 A/D Data Registers (ADCR0/ADCR1)

A/D data registers (ADCR0/ADCR1) are used to record the digital values generated from the conversion results. While ADCR0 records the lower 8 bits of the conversion results, ADCR1 records the highest 2 bits. Every time conversion is completed, these registers are rewritten, and in general, store the last conversion values.

■ A/D Data Registers (ADCR0/ADCR1)

Figure 16.3-4 A/D Data Registers (ADCR0/ADCR1)

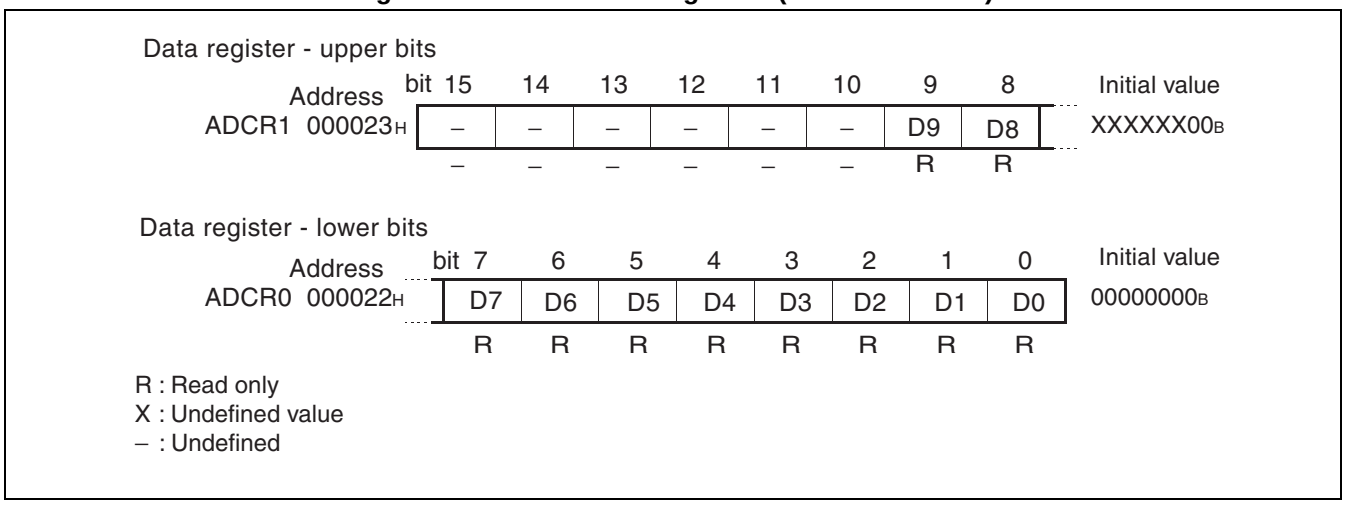


Table 16.3-4 Functions of A/D Data Registers (ADCR0/ADCR1)

Bit name		Function
bit15 to bit10	Undefined bits	When reading, "1" is always read.
bit9 to bit0	D9 to D0: A/D conversion data bits	These bits store A/D conversion results. **When resolution is set to 10 bits (S10 = 0):** Conversion data is stored in D9 to D0 (10 bits). **When resolution is set to 8 bits (S10 = 1):** Conversion data is stored in D7 to D0 (8 bits). Then, the read value of D9, D8 becomes "1". Notes: • Writing to these registers is prohibited. • To read the conversion results stored in the A/D conversion data bits (D9 to D0), a word instruction (MOVW) must be used.

16.3.4 A/D Setting Registers (ADSR0/ADSR1)

The A/D setting registers (ADSR0/ADSR1) enable the following settings:

- **Setting A/D conversion times (sampling time and compare time)**
- **Setting sampling channels (start channel and end channel)**
- **Displaying the current sampling channels**

■ A/D Setting Registers (ADSR0/ADSR1)

Figure 16.3-5 A/D Setting Registers (ADSR0/ADSR1)

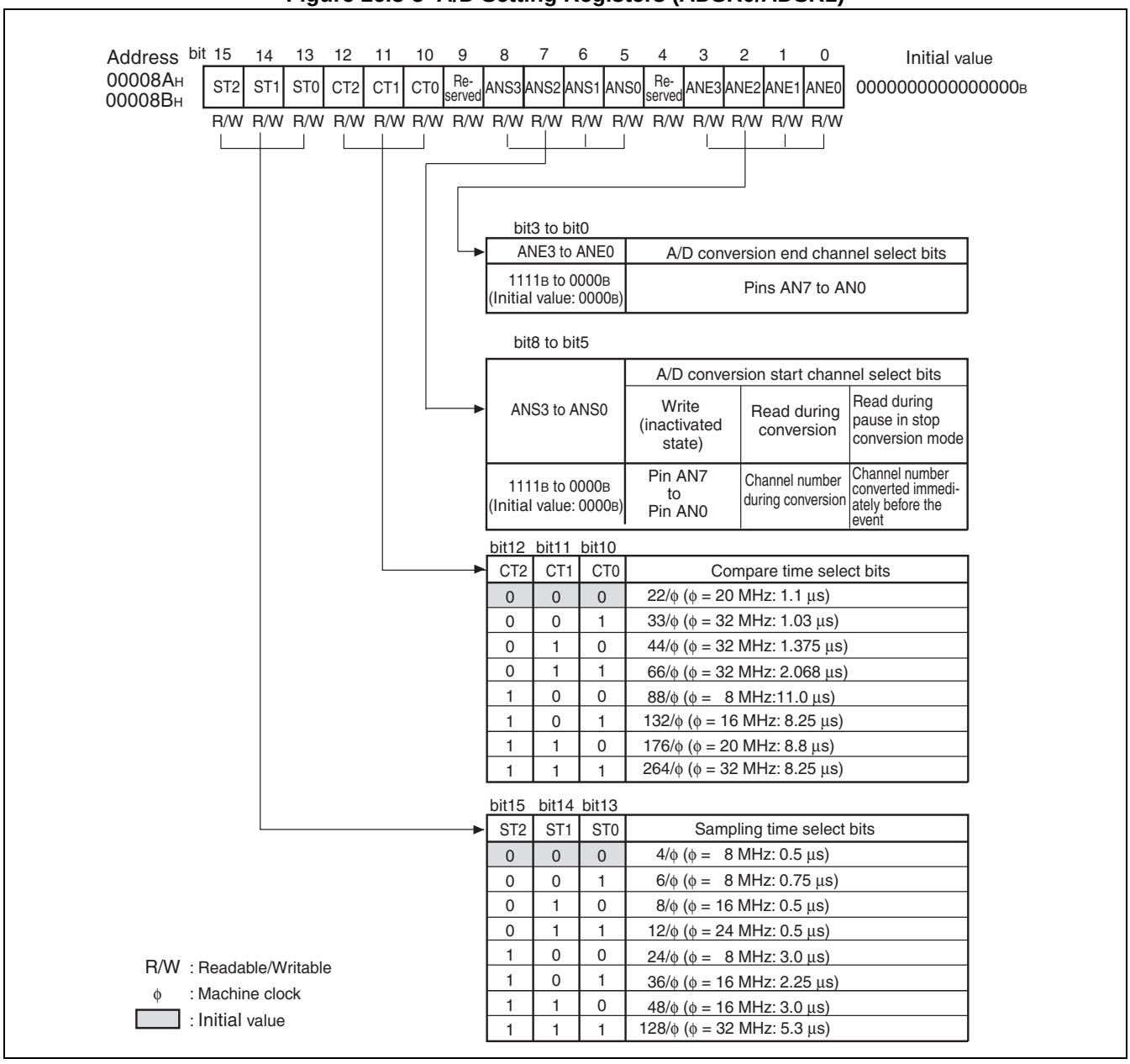


Table 16.3-5 Functions of A/D Setting Registers (ADSR0/ADSR1) (Sheet 1 of 2)

Bit name		Function
bit15 to bit13	ST2, ST1, ST0: Sampling time select bits	These bits set the sampling time for A/D conversion. • They set the time spent from start of A/D conversion to when the input analog voltage is sampled and then held in the sample and hold circuit. • For information about the settings of these bits, see Table 16.3-6.
bit12 to bit10	CT2, CT1, CT0: Compare time select bits	These bits set the compare time for A/D conversion. • They set the time spent from when A/D conversion is performed on the analog input to when the results are stored in the data bits (D9 to D0). • For information about the settings of these bits, see Table 16.3-7.
bit8 to bit5	ANS3 to ANS0: A/D conversion start channel select bits	These bits set the channel with which A/D conversion is started. When reading, you can identify the channel number currently being converted when the A/D conversion is in progress, and you can identify the last channel number on which the A/D conversion was performed when the A/D conversion is completed or stopped. In addition, even when a particular value is set to these bits, the channel number of the channel on which the A/D conversion was performed previously is read rather than that set value until the A/D conversion starts. At a reset, the value is initialized to 0000_B. **Start channel < End channel:** A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) and ends at the channel set by the A/D conversion end channel select bits (ANE3 to ANE0). **Start channel = End channel:** A/D conversion is performed for the only one channel that is set by the A/D conversion start (= end) channel select bits (ANS3 to ANS0 = ANE3 to ANE0). **Start channel > End channel:** A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) to AN7, and then another A/D conversion starts up to the channel set by the A/D conversion end channel select bits (ANE3 to ANE0). **In continuous conversion mode, stop conversion mode:** Once A/D conversion is completed at the channel set by the A/D conversion end channel select bits (ANE3 to ANE0), the conversion sequence returns to the channel set by A/D conversion start channel select bits (ANS3 to ANS0). **Read (in mode other than stop conversion mode):** The channel number of the channel (7 to 0) for which A/D conversion is currently being performed is read. **Read (in stop conversion mode):** When a read is performed in the stop mode, the channel number of the last channel for which A/D conversion was performed immediately before it was stopped is read. Notes: Do not set the A/D conversion start channel bits (ANS3 to ANS0) during A/D conversion. Use the word access method when writing to these bits. If a byte-access or bit operation is performed, A/D conversion may start from an unintended channel.

Table 16.3-5 Functions of A/D Setting Registers (ADSR0/ADSR1) (Sheet 2 of 2)

Bit name		Function
bit3 to bit0	ANE3 to ANE0: A/D conversion end channel select bits	These bits set the channel at which A/D conversion ends. **Start channel < End channel:** A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) and ends at the channel set by the A/D conversion end channel select bits (ANE3 to ANE0). **Start channel = End channel:** A/D conversion is performed for the only one channel that is set by the A/D conversion start (= end) channel select bits (ANS3 to ANS0 = ANE3 to ANE0). **Start channel > End channel:** A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) to AN7, and then another A/D conversion starts up to the channel set by the A/D conversion end channel select bits (ANE3 to ANE0). **In continuous conversion mode, stop conversion mode:** Once A/D conversion is completed at the channel set by the A/D conversion end channel select bits (ANE3 to ANE0), the conversion sequence returns to the channel set by A/D conversion start channel select bits (ANS3 to ANS0). Notes: • Do not set the A/D conversion end channel bits (ANE3 to ANE0) during A/D conversion. • Do not use a read-modify-write (RMW) instruction to set the sampling time select bits (ST2, ST1 and ST0), the compare time select bits (CT2, CT1 and CT0), or the A/D conversion end channel select bits (ANE3, ANE2, ANE1 and ANE0) after setting the A/D conversion start channel select bits (ANS3, ANS2, ANS1 and ANS0). As the previously converted channel is read by bits ANS3, ANS2, ANS1 and ANS0 until A/D conversion operation starts, the bit values of ANS3, ANS2, ANS1 and ANS0 may be rewritten when a read-modify-write (RMW) instruction is used to set bits ST2, ST1, ST0, bits CT2, CT1, CT0, and bits ANE3, ANE2, ANE1, ANE0 after bits ANS3, ANS2, ANS1 and ANS0 are set.

■ Setup for Sampling Time (Bits ST2 to ST0)

Table 16.3-6 Correlation between Bit ST2 to ST0 and Sampling Times

ST2	ST1	ST0	Sampling time setting	Example setting (ϕ: Internal operating frequency)
0	0	0	4 machine cycles	ϕ = 8MHz: 0.5μs
0	0	1	6 machine cycles	ϕ = 8MHz: 0.75μs
0	1	0	8 machine cycles	ϕ = 16MHz: 0.5μs
0	1	1	12 machine cycles	ϕ = 24MHz: 0.5μs
1	0	0	24 machine cycles	ϕ = 8MHz: 3 μs
1	0	1	36 machine cycles	ϕ = 16MHz: 2.25μs
1	1	0	48 machine cycles	ϕ = 16MHz: 3.0 μs
1	1	1	128 machine cycles	ϕ = 32MHz: 4.0μs

The sampling time must be set in accordance with the drive impedance (R_{ext}), which is connected to the analog input. The conversion accuracy is not guaranteed unless the following conditions are satisfied:

- $R_{ext} \leq 1.5k\Omega$:
  - $4.5V \leq AV_{CC} < 5.5V$: Set the sampling time to 0.5μs or higher.
  - $4.0V \leq AV_{CC} < 4.5V$: Set the sampling time to 1.2μs or higher.
- $R_{ext} > 1.5k\Omega$: Set the sampling time to T_{samp} shown in the following expression, or higher.
  - $4.5V \leq AV_{CC} < 5.5V$: $T_{samp} = (2.52k\Omega + R_{ext}) \times 10.7pF \times 7$
  - $4.0V \leq AV_{CC} < 4.5V$: $T_{samp} = (13.6k\Omega + R_{ext}) \times 10.7pF \times 7$

■ Setup for Compare Time (Bits CT2 to CT0)

Table 16.3-7 Correlation between Bits CT2 to CT0 and Compare Times

CT2	CT1	CT0	Compare time setting	Example setting (ϕ: Internal operating frequency)
0	0	0	22 machine cycles	ϕ = 20MHz: 1.1μs
0	0	1	33 machine cycles	ϕ = 32MHz: 1.03μs
0	1	0	44 machine cycles	ϕ = 32MHz: 1.375μs
0	1	1	66 machine cycles	ϕ = 32MHz: 2.063μs
1	0	0	88 machine cycles	ϕ = 8MHz: 11.0 μs
1	0	1	132 machine cycles	ϕ = 16MHz: 8.25μs
1	1	0	176 machine cycles	ϕ = 20MHz: 8.8μs
1	1	1	264 machine cycles	ϕ = 32MHz: 8.25μs

The compare time must be set in accordance with the analog power supply voltage (AV_{CC}). The conversion accuracy is not guaranteed unless the following conditions are satisfied:

- $4.5V \leq AV_{CC} < 5.5$ V: Set the compare time to 1.00μs or higher.
- $4.0V \leq AV_{CC} < 4.5$ V: Set the compare time to 2.00μs or higher.

16.3.5 Analog Input Enable Register (ADER6)

This register enables or disables the analog input pins used for the 8-/10-bit A/D converter.

■ Analog Input Enable Register (ADER6)

Figure 16.3-6 Analog Input Enable Register (ADER6)

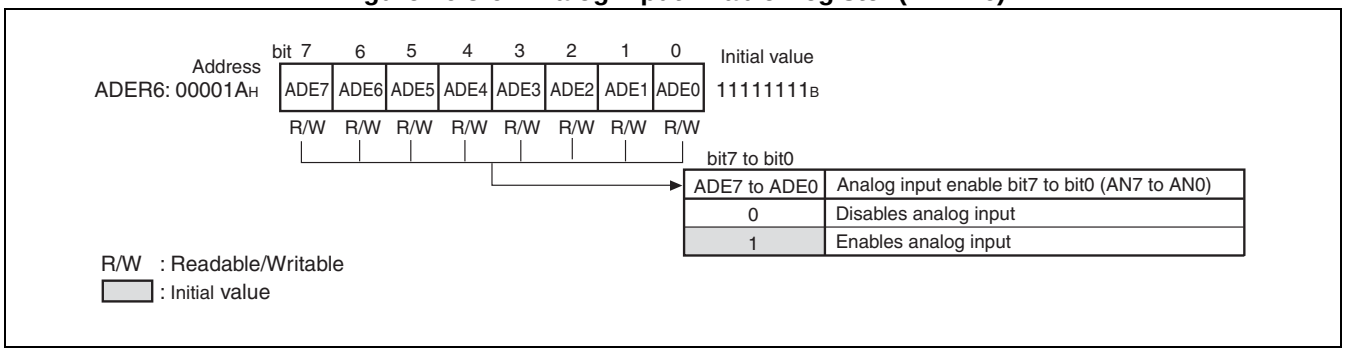


Table 16.3-8 Function of Port 6 Analog Input Enable Register (ADER6)

Bit name		Function
bit7 to bit0	ADE7 to ADE0: Analog input enable bits	These bits enable or disable analog input from A/D conversion analog input pins (AN7 to AN0), located on port 6. **When set to "0":** Analog input disabled **When set to "1":** Analog input enabled

Notes:

- To use the register as an analog input pin, write "1" to the bits in the analog input enable register (ADER6) corresponding to the pin to be used in order to set it as analog input.
- Setting the analog input pin as ADEx = 0 is prohibited. The pin must always be set as ADEx = 1.
- Each analog input pin is also used as a general-purpose I/O port and input/output of a peripheral function. When set as ADEx = 1, the pin is forced to become an analog input pin regardless of the settings of the port direction register (DDR6) and I/O settings of peripheral functions, therefore it cannot be used otherwise.

16.4 Interrupts of 8-/10-bit A/D Converter

In the 8-/10-bit A/D converter, an interrupt request is generated when the conversion results are stored in the A/D data register (ADCR) upon the completion of A/D conversion. The extended intelligent I/O service (EI^{2}OS) can be used.

■ Interrupts of A/D Converter

When the A/D conversion results are stored in the A/D data register (ADCR) upon the completion of A/D conversion of analog input voltage, the interrupt request flag bit in the A/D control status register (ADCS: INT) is set to "1". An interrupt request is generated when the output of interrupt requests is enabled
(ADCS: INTE = 1) and the interrupt request flag bit is set (ADCS: INT = 1).

■ Interrupts and EI^{2}OS of 8-/10-bit A/D Converter

Reference:

For information about the interrupt number, interrupt control register, and interrupt vector address, see "CHAPTER 3 INTERRUPTS".

■ EI^{2}OS of 8-/10-bit A/D Converter

In the 8-/10-bit A/D converter, EI^{2}OS can be used to transfer the A/D conversion results from the A/D data register (ADCR) to memory. For information about how to use the EI^{2}OS functions, see Section "16.5.4 Conversion Operation by EI^{2}OS Function" as well as Section "16.5.5 A/D Conversion Data Protection Function".

16.5 Explanation of 8-/10-bit A/D Converter Operations

The 8-/10-bit A/D converter performs A/D conversion operation in the following conversion modes. Each mode is specified by setting the A/D conversion mode select bits in the A/D control status register (ADCS: MD1, MD0):

- **Single conversion mode**
- **Continuous conversion mode**
- **Stop conversion mode**

■ Single Conversion Mode (ADCS: MD1, MD0 = 00_B or 01_B)

- When a start trigger is entered, A/D conversion is performed on the analog input from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) in succession.
- The A/D conversion operation will stop when the A/D conversion for the end channel is completed.

Notes:

- In single conversion mode1 (ADCS: MD1, MD0 = 00_B), the 8-/10-bit A/D converter may be restarted when a start trigger is entered while A/D conversion is being performed or paused*. Therefore, do not enter the start trigger while A/D conversion is being performed or paused.
  In single conversion mode2 (ADCS: MD1, MD0 = 01_B), the 8-/10-bit A/D converter is not restarted even when a start trigger is entered while A/D conversion is being performed or paused*.
- When restarting the converter in single conversion mode 1 or 2, follow the procedure shown Section "16.5.1 Single Conversion Mode".

*: In pause state, the A/D conversion protection function operates, and the conversion is temporarily stopped. For more information, see Section "16.5.5 A/D Conversion Data Protection Function".

■ Continuous Conversion Mode (ADCS: MD1, MD0 = 10_B)

- When a start trigger is entered, A/D conversion is performed on the analog input from the start channel (ADSR: ANS3 to ANS0) to the end channel (ADSR: ANE3 to ANE0) in succession.
- When the A/D conversion is completed for the end channel, the conversion sequence returns to the analog input of the start channel to continue the A/D conversion.

■ Stop Conversion Mode (ADCS: MD1, MD0 = 11_B)

- When a start trigger is entered, A/D conversion starts at the start channel (ADSR: ANS3 to ANS0). The A/D conversion operation will stop every time the A/D conversion for a channel is completed. This state is called "pause state". If a start trigger is entered while the A/D conversion operation is still stopped,
  A/D conversion for the next channel will be performed.
- The A/D conversion operation will stop when the A/D conversion for the end channel is completed. If a start trigger is

entered while the conversion operation is still stopped, the conversion sequence will return to the analog input of the start channel to continue the A/D conversion.

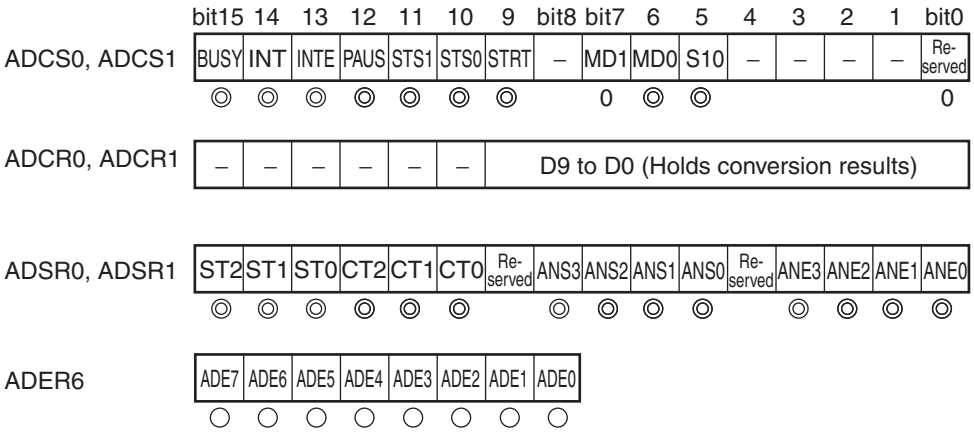
16.5.1 Single Conversion Mode

In single conversion mode, A/D conversion is performed sequentially from the start channel to the end channel. The A/D conversion operation will stop when the A/D conversion for the end channel is completed.

■ Setup for Single Conversion Mode

To operate the 8-/10-bit A/D converter in the single conversion mode, the settings described in Figure 16.5-1 are required.

Figure 16.5-1 Setup for Single Conversion Mode



– : Undefined
◎ : Used bit
○ : Specify "1" to the bit corresponding to the pin to be used as an analog input
0 : Specify "0"

■ Operations and Applications of Single Conversion Mode

- When a start trigger is entered, an A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0), and continues through to the channel set by the A/D conversion end channel select bits (ANE3 to ANE0).
- When the A/D conversion is completed for the channel set by the A/D conversion end channel select bits (ANE3 to ANE0), the A/D conversion operation is stopped.
- To force the termination of an A/D conversion, write "0" to the A/D conversion operation flag bit in the A/D control status register (ADCS: BUSY).

[When the start channel and end channel are the same]

When the channel number of the start and end channels are set to the same number (ADSR: ANS3 to ANS0 = ADSR: ANE3 to ANE0), A/D conversion is terminated after performing conversion for only one channel once, which is specified as the start channel (= end channel).

[Conversion sequence in single conversion mode]

Table 16.5-1 shows examples of the conversion sequence in single conversion mode.

Table 16.5-1 Conversion Sequence in Single Conversion Mode

Start channel	End channel	Conversion sequence in single conversion mode
Pin AN0 (ADSR: ANS3 to ANS0 = 0000_B)	Pin AN3 (ADSR: ANE3 to ANE0 = 0011_B)	AN0 –> AN1 –> AN2 –> AN3 –> Terminated
Pin AN5 (ADSR: ANS3 to ANS0 = 0101_B)	Pin AN2 (ADSR: ANE3 to ANE0 = 0010_B)	AN5 –> AN6 –> AN7 –> AN8 –> … AN30 –> AN31 –> AN0 –> AN1 –> AN2 –> Terminated
Pin AN3 (ADSR: ANS3 to ANS0 = 0011_B)	Pin AN3 (ADSR: ANE3 to ANE0 = 0011_B)	AN3 –> Terminated

Note: Start channel > End channel:
Starts sampling on the channels that do not have any analog input pin (AN8 to AN31), therefore, we do not recommend this setting.

[Restart]

If you want to restart the A/D conversion while the A/D conversion is in execution or pause state, the conversion is required to forcibly terminate before restarting. Follow the procedure below:

1) Clear the A/D conversion operation flag bit (ADCS: BUSY)

2) Clear the interrupt request flag bit (ADCS: INT)

3) Set the A/D conversion software start bit (ADCS: STRT)

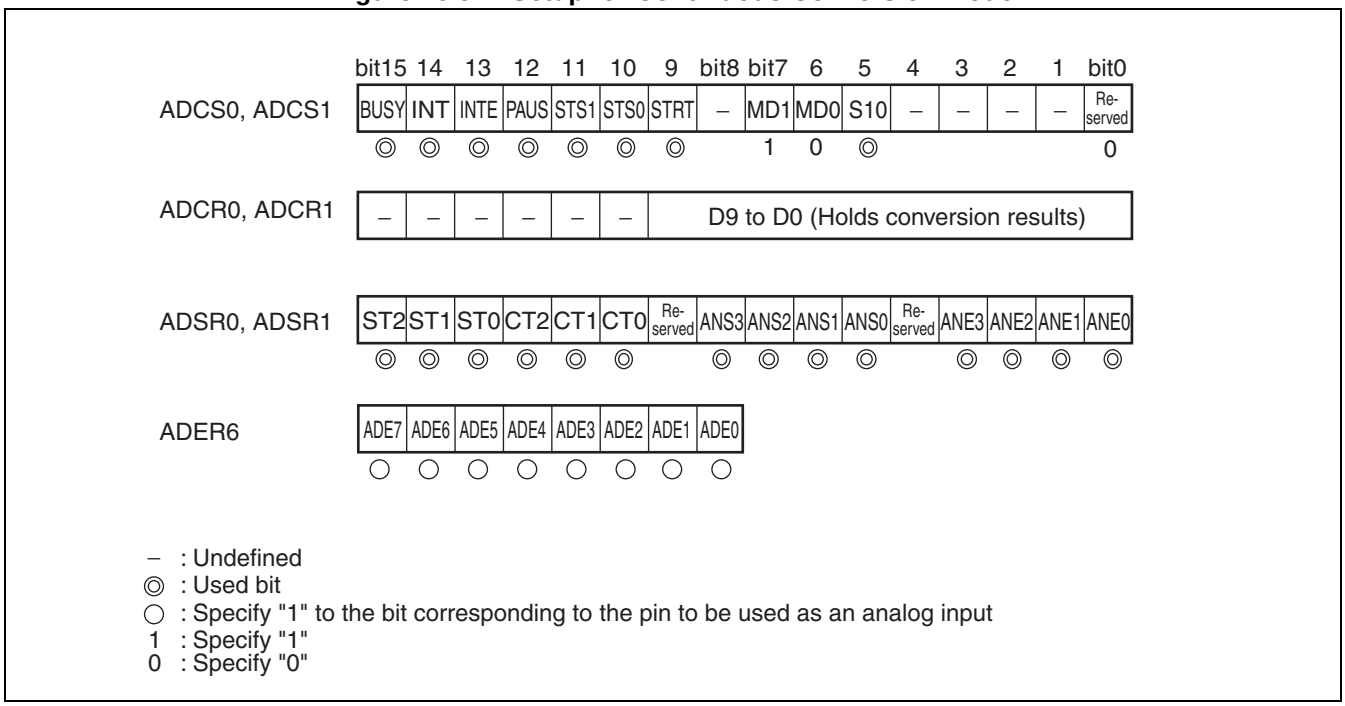
16.5.2 Continuous Conversion Mode

In continuous conversion mode, A/D conversion is performed sequentially from the start channel to the end channel. When the A/D conversion for the end channel is completed, the conversion sequence returns to the start channel to continue the A/D conversion operation.

■ Setup for Continuous Conversion Mode

To operate the 8-/10-bit A/D converter in the continuous conversion mode, the settings described in Figure 16.5-2 are required.

Figure 16.5-2 Setup for Continuous Conversion Mode



– : Undefined
◎ : Used bit
○ : Specify "1" to the bit corresponding to the pin to be used as an analog input
1 : Specify "1"
0 : Specify "0"

■ Operations and Applications of Continuous Conversion Mode

- When a start trigger is entered, an A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0), and continues through to the channel set by the A/D conversion end channel select bits (ANE3 to ANE0).
- When the A/D conversion is completed for the channel set by the A/D conversion end channel select bits (ANE3 to ANE0), the conversion sequence returns to the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) to continue the A/D conversion.
- To force the termination of an A/D conversion, write "0" to the A/D conversion operation flag bit in the A/D control status register (ADCS: BUSY).

[When the start channel and end channel are the same]

When the start channel is set to the same channel as the end channel (ADSR: ANS3 to ANS0 = ADSR: ANE3 to ANE0), A/D conversion is performed at the one channel set as the start channel (= end channel) repeatedly.

[Conversion sequence in continuous conversion mode]

Table 16.5-2 shows examples of the conversion sequence in continuous conversion mode.

Table 16.5-2 Conversion Sequence in Continuous Conversion Mode

Start channel	End channel	Conversion sequence in continuous conversion mode
Pin AN0 (ADSR: ANS3 to ANS0 = 0000_B)	Pin AN3 (ADSR: ANE3 to ANE0 = 0011_B)	AN0 –> AN1 –> AN2 –> AN3 –> AN0 –> Repeat
Pin AN5 (ADSR: ANS3 to ANS0 = 0101_B)	Pin AN2 (ADSR: ANE3 to ANE0 = 0010_B)	AN5 –> AN6 –> AN7 –> AN8 –> … AN30 –> AN31 –> AN0 –> AN1 –> AN2 –> AN5 –> Repeat
Pin AN3 (ADSR: ANS3 to ANS0 = 0011_B)	Pin AN3 (ADSR: ANE3 to ANE0 = 0011_B)	AN3 –> AN3 –> Repeat

Note: Start channel > End channel:
Starts sampling on the channels that do not have any analog input pin (AN8 to AN31), therefore, we do not recommend this setting.

[Restart]

If you want to restart the A/D conversion while the A/D conversion is in execution or pause state, the conversion is required to forcibly terminate before restarting. Follow the procedure below:

1) Clear the A/D conversion operation flag bit (ADCS: BUSY)

2) Clear the interrupt request flag bit (ADCS: INT)

3) Set the A/D conversion software start bit (ADCS: STRT)

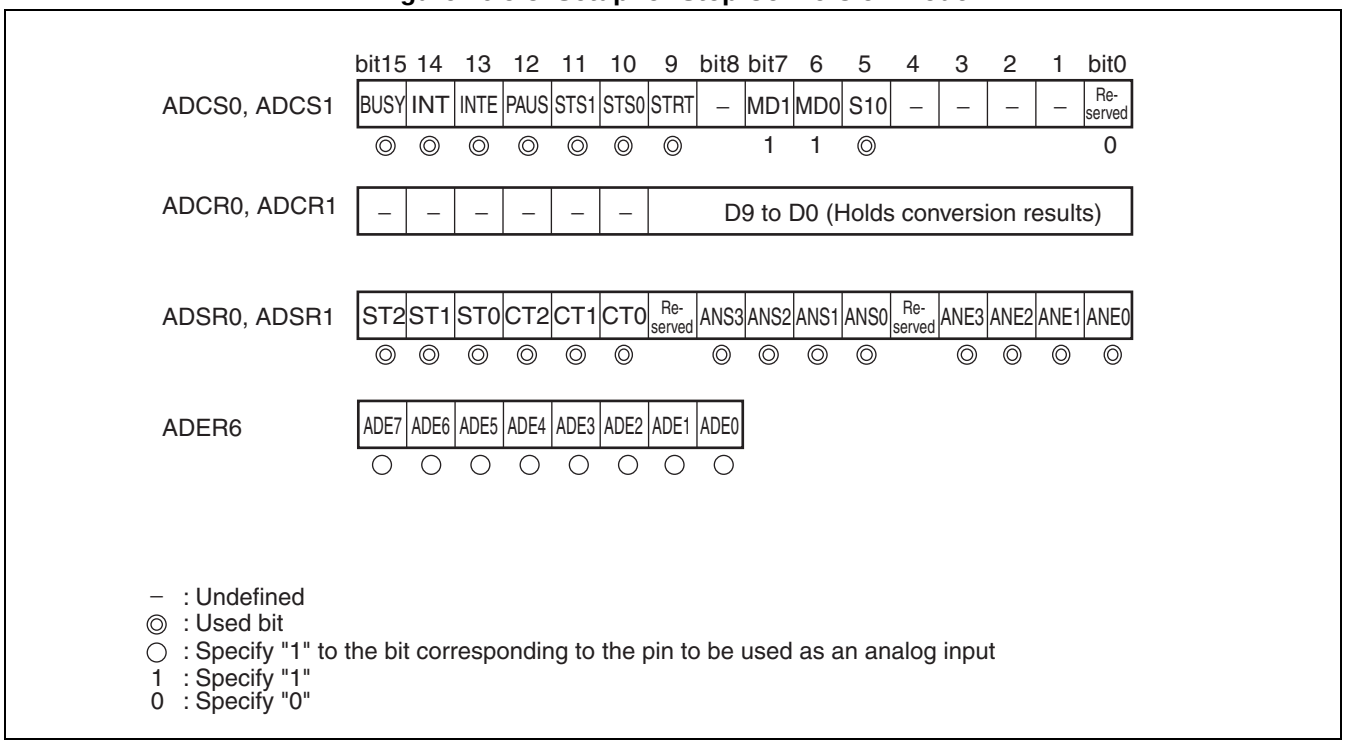
16.5.3 Stop Conversion Mode

In the stop conversion mode, A/D conversion is performed by repeatedly stopping and starting at each channel. The A/D conversion sequence returns to the start channel to continue the A/D conversion, when a start trigger is entered after the A/D conversion is completed for the end channel and the A/D conversion operation is stopped.

■ Setup for Stop Conversion Mode

To operate the 8-/10-bit A/D converter in the stop conversion mode, the settings described in Figure 16.5-3 are required.

Figure 16.5-3 Setup for Stop Conversion Mode



■ Operations and Applications of Stop Conversion Mode

- When a start trigger is entered, A/D conversion starts from the channel set by the A/D conversion start channel select bits (ANS3 to ANS0). The A/D conversion operation will stop every time the A/D conversion for a channel is completed. When the start trigger is entered while the A/D conversion operation is stopped, A/D conversion is performed for the next channel.
- When the A/D conversion is completed for the channel set by the A/D conversion end channel select bits (ANE3 to ANE0), the A/D conversion operation is stopped. When the start trigger is entered while the A/D conversion operation is stopped, the conversion sequence returns to the channel set by the A/D conversion start channel select bits (ANS3 to ANS0) to continue the A/D conversion.
- To force the termination of an A/D conversion, write "0" to the A/D conversion operation flag bit in the A/D control

status register (ADCS: BUSY).

[When the start channel and end channel are the same]

When the start channel is set to the same channel as the end channel (ADSR: ANS3 to ANS0 = ADSR: ANE3 to ANE0), A/D conversion is performed and then stopped repeatedly at the only one channel specified as the start channel (= end channel).

[Conversion sequence in stop conversion mode]

Table 16.5-3 shows example conversion sequences for the stop conversion mode.

Table 16.5-3 Conversion Sequence in Stop Conversion Mode

Start channel	End channel	Conversion sequence in stop conversion mode
Pin AN0 (ADSR: ANS3 to ANS0 = 0000_B)	Pin AN3 (ADSR:ANE3 to ANE0 = 0011_B)	AN0 –> Stop/Start –> AN1 –> Stop/Start –> AN2 –> Stop/Start –> AN3 –> Stop/Start –> AN0 –> Repeat
Pin AN5 (ADSR: ANS3 to ANS0 = 0101_B)	Pin AN2 (ADSR: ANE3 to ANE0 = 0010_B)	AN5 –> Stop/Start –> AN6 –> Stop/Start –> AN7 –> Stop/Start –> AN8 –> Stop/Start –> … AN30 –> Stop/Start –> AN31 –> Stop/Start –> AN0 –> Stop/Start –> AN1 –> Stop/Start –> AN2 –> Stop/Start –> AN5 –> Repeat*
Pin AN3 (ADSR: ANS3 to ANS0 = 0011_B)	Pin AN3 (ADSR: ANE3 to ANE0 = 0011_B)	AN3 –> Stop/Start –> AN3 –> Stop/Start –> Repeat

Note: Start channel > End channel:
Starts sampling on the channels that do not have any analog input pin (AN8 to AN31), therefore, we do not recommend this setting.

[Restart]

If you want to restart the A/D conversion while the A/D conversion is in execution or pause state, the conversion is required to forcibly terminate before restarting. Follow the procedure below:

1) Clear the A/D conversion operation flag bit (ADCS: BUSY)

2) Clear the interrupt request flag bit (ADCS: INT)

3) Set the A/D conversion software start bit (ADCS: STRT)

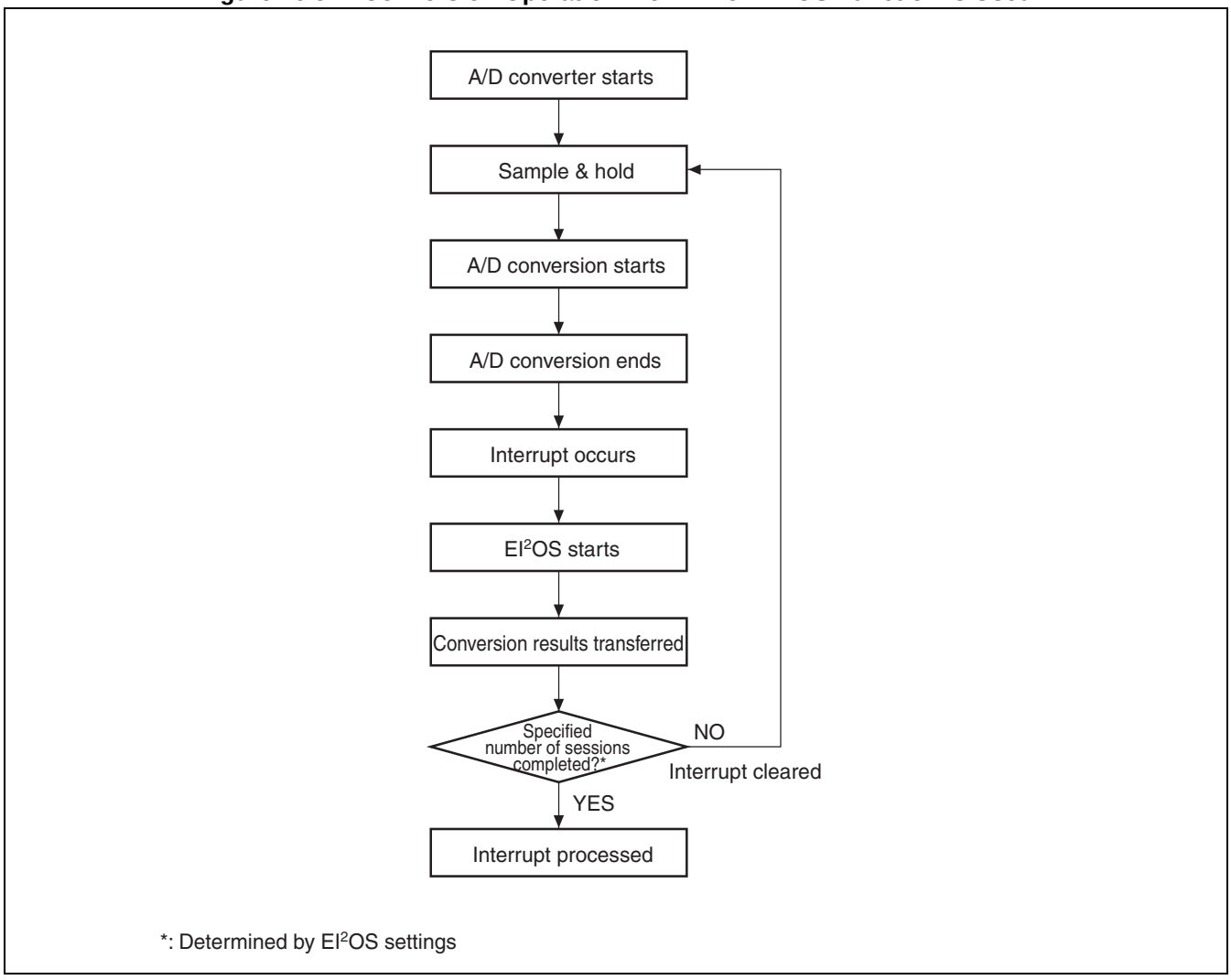
16.5.4 Conversion Operation by EI^{2}OS Function

In the 8-/10-bit A/D converter, the EI^{2}OS function can be used to transfer the A/D conversion results to the memory.

■ Conversion Operation by EI^{2}OS Function

Figure 16.5-4 shows the flow of the conversion operation when the EI^{2}OS function is used.

Figure 16.5-4 Conversion Operation Flow When EI^{2}OS Function is Used



16.5.5 A/D Conversion Data Protection Function

The data protection function is activated when A/D conversion is performed while the output of interrupt requests is enabled.

■ Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter

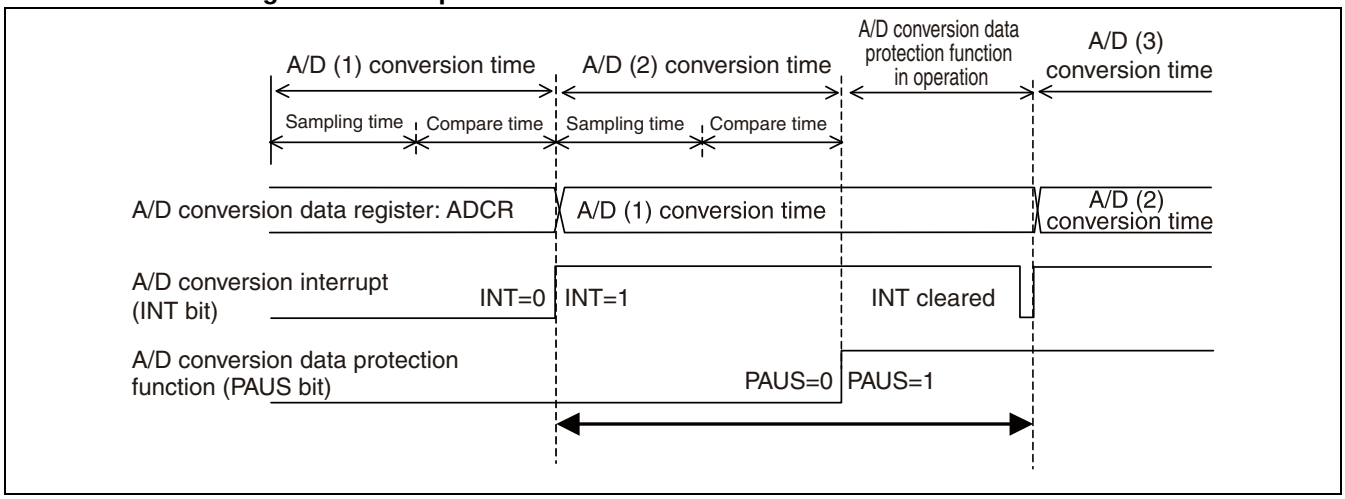
The A/D conversion data protection function prevents A/D conversion data from being unretrieved.

The 8-/10-bit A/D converter has one A/D data register (ADCR1/ADCR0) for storing conversion data and one successive approximation circuit for storing the data on which A/D conversion is currently being performed. During the A/D conversion, the 8-/10-bit A/D converter stores the conversion data for each bit separately in the successive approximation circuit. Once the A/D conversion is completed, the A/D conversion results are stored in the A/D data register.

Depending on whether or not to use the A/D conversion data protection function, the 8-/10-bit A/D converter operates differently, as described below.

- If you set the interrupt request enable bit (ADCS: INTE) to "0", the data protection function is disabled. In this case, when A/D conversions are performed successively, the conversion results are stored in the A/D data register every time the 8-/10-bit A/D converter completes each conversion (Consequently, the latest conversion data is always stored).
- If you set the interrupt request enable bit (ADCS: INTE) to "1", the data protection function is enabled. When A/D conversions are performed successively in this state, the interrupt request flag bit becomes "1" (ADCS: INT = 1) upon the completion of the first conversion session. If the next A/D conversion is performed and completed while INT = 1, the 8-/10-bit A/D converter enters to the "pause state" just before the conversion results are transferred from the successive approximation circuit to the A/D data register, preventing the conversion data from being overwritten. At this point, "1" is set to the pause flag bit in the A/D control status register (ADCS: PAUS). If you clear the interrupt request flag bit (ADCS: INT) to "0" in the pause state, the data stored in the successive approximation circuit is transferred to the A/D data register (see Figure 16.5-5).

Figure 16.5-5 Operation of A/D Conversion Data Protection Function



● A/D conversion data protection function when reading the A/D conversion results by CPU

- When the A/D conversion results are stored in the A/D data register (ADCR) after A/D conversion is performed on analog input, "1" is set to the interrupt request flag bit in the A/D control status register (ADCS: INT).
- When the next A/D conversion session is completed, if the interrupt request flag bit (ADCS: INT), which was set upon the completion of the previous A/D conversion session, is still set, and interrupt requests are enabled (ADCS: INTE = 1), the A/D conversion operation will be paused just before the new data overwrites the current data in the A/D data register in order to protect the data.
- As the output of interrupt requests is enabled by the A/D control status register (ADCS: INTE = 1), an interrupt request is generated when the INT bit is set. When the INT bit is cleared, the A/D conversion operation is released from the pause.
- When performing A/D conversion sessions successively, the 8-/10-bit A/D converter starts the next conversion session. At this point, the pause flag bit (ADCS: PAUS) is not cleared to "0" automatically. To clear, you must write "0" to this bit.

Notes:

- If the output of interrupt requests is disabled during the pause state (ADCS: INTE = 0), A/D conversion may start, causing the data in the A/D data register to be rewritten.
- When performing multiple A/D conversion sessions successively, always read the data stored in 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 the data stored in the A/D data register is read while the A/D conversion is paused, the firstly-stored conversion data will be overwritten by the next conversion data and therefore destroyed.

● A/D conversion data protection function when transferring the A/D conversion results by EI^{2}OS

If the next A/D conversion session is completed while the EI^{2}OS function is used to transfer the A/D conversion result from the A/D data register to the memory, the A/D conversion operation is paused just before the new data overwrites the current data in the A/D data register for data protection purposes. When the A/D conversion operation stops, the pause flag bit of the A/D control status register (ADCS: PAUS) is set to "1".

Once the transfer of the A/D conversion results to the memory is completed using the EI^{2}OS function, the A/D conversion is released from the pause state. When performing A/D conversion sessions successively, the A/D conversion operation is resumed. At this point, the pause flag bit (ADCS: PAUS) is not cleared to "0" automatically. To clear, you must write "0" to the PAUS bit.

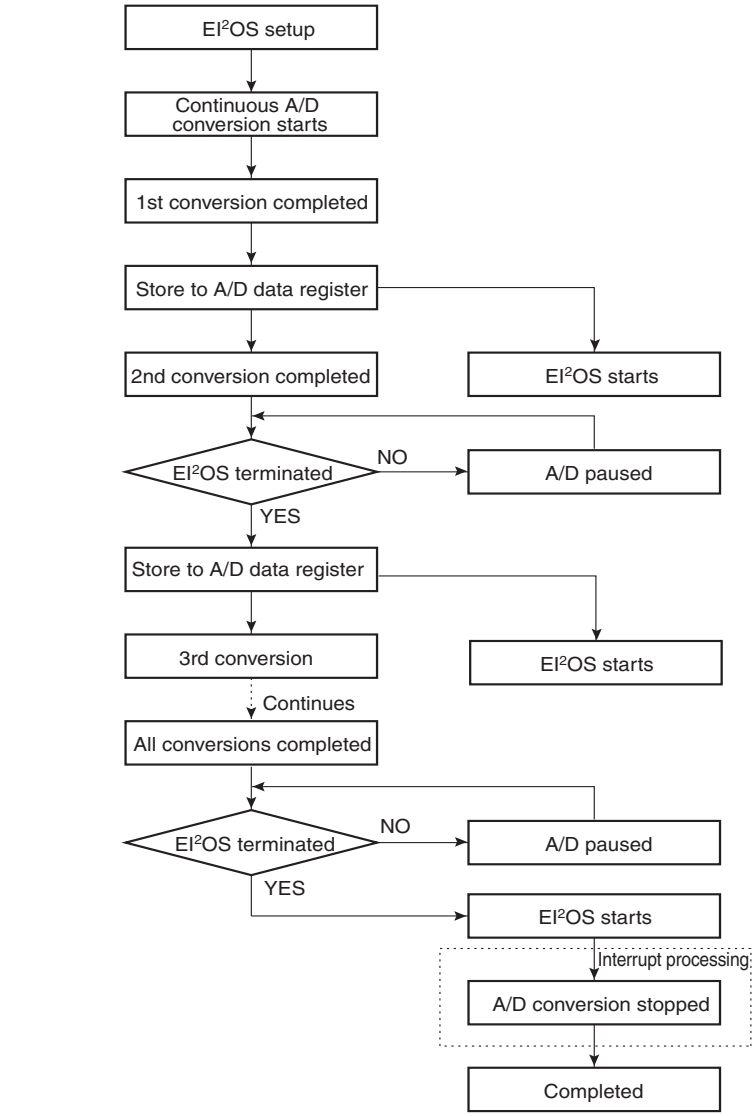
Notes:

- Do not clear the interrupt request flag bit from CPU (ADCS: INT = 0), when the EI^{2}OS function is used to transfer the A/D conversion results to the memory, or the data in the A/D data register, which is being transferred, may be rewritten.
- Do not disable the output of interrupt requests, when the EI^{2}OS function is used to transfer the A/D conversion results to the memory. If the output of interrupt requests is disabled during the pause state (ADCS: INTE = 0), A/D conversion may start, causing the data in the A/D data register, which is currently being transferred, to be rewritten.
- Do not restart while the EI^{2}OS function is used to transfer the A/D conversion results to the memory. If restarted while the A/D conversion is paused, the conversion results may be destroyed.

- Processing flow of A/D conversion data protection function when EI^{2}OS is used

Figure 16.5-6 shows the processing flow of the A/D conversion data protection function when EI^{2}OS is used.

Figure 16.5-6 Processing Flow of A/D Conversion Data Protection Function When EI^{2}OS is Used



Note: The flow for when the A/D converter operation is stopped is omitted.

16.6 Precautions when Using 8-/10-bit A/D Converter

Precautions must be taken for the following points when using the 8-/10-bit A/D converter.

■ Precautions When Using 8-/10-bit A/D Converter

● Analog input pins

- The analog input pins are also used as general-purpose I/O ports for port 6. When using them as analog input pins, set up the analog input enable register (ADER6) to switch them to analog input pins.
- When using the pin as analog input pin, write "1" to the bit in the analog input enable register (ADER6) which corresponds to the pin to be used in order to enable the analog input.
- If a medium-level signal is input while the pin remains set as a general-purpose I/O port, input leakage current is flowed to the gate. When using it as an analog input pin, always enable the analog input before use.

● Precaution when using the internal timer or external trigger to start the converter

When setting the A/D start trigger select bits of the A/D control status register (ADCS: STS1 and STS0) to start the 8-/10-bit A/D converter by internal timer output or external trigger, set the level of the timer output and the external trigger to the inactive side ("H" side for the external trigger). If the input value of the start trigger is set to the active side, the operation may start as soon as the A/D start trigger select bits of the A/D control status register (ADCS: STS1 and STS0) are set.

● Power application and analog input sequence of the 8-/10-bit A/D converter

- Always apply power to the 8-/10-bit A/D converter and analog inputs after the digital power supply (V_{CC}) is turned on.
- When power-off, turn off the 8-/10-bit A/D converter and the analog inputs before turning off the digital power supply.
- Do not allow AVRH to exceed AV_{CC} when turning on and off the power.

● Power supply voltage of the 8-/10-bit A/D converter

Care must be taken to ensure that the power supply of the 8-/10-bit A/D converter (AV_{CC}) does not exceed the digital power supply (V_{CC}) voltage to prevent latch-up.

17. LIN-UART

This chapter explains the functions and operations of the LIN-UART.

17.1 Overview of LIN-UART
17.2 Configuration of LIN-UART
17.3 Pins of LIN-UART
17.4 LIN-UART Registers
17.5 Interrupts of LIN-UART
17.6 LIN-UART Baud Rates
17.7 Operation of LIN-UART
17.8 Notes on Using LIN-UART

17.1 Overview of LIN-UART

The LIN (Local Interconnect Network)-UART is a general-purpose serial data communication interface for performing synchronous or asynchronous communication (start-stop synchronization) with external devices. LIN-UART provides bidirectional communication function (normal mode), master-slave communication function (multiprocessor mode in master/slave systems), and special features for LIN bus systems.

■ Functions of LIN-UART

● Functions of LIN-UART

LIN-UART is a general-purpose serial data communication interface for transmitting serial data to and receiving data from another CPU and peripheral devices. It has the functions listed in Table 17.1-1.

Table 17.1-1 Functions of LIN-UART

	Function
Data buffer	Full-duplicate double-buffer
Serial input	Perform oversampling 5 times and determine the received value by majority decision of sampling value (asynchronous mode only)
Transfer mode	• Clock synchronous (selecting start/stop synchronous or start/stop bit) • Clock asynchronous (start/stop bits can be used.)
Baud rate	• Dedicated baud rate generator (The baud rate is consisted of 15-bit reload counter.) • An external clock can be inputted and also be adjusted by reload counter.
Data length	• 7 bits (other than synchronous or LIN mode) • 8 bits
Signaling	NRZ (non return to zero)
Start bit timing	Synchronization to the falling edge of the start bit in the asynchronous mode
Detection of receive error	• Framing error • Overrun error • Parity error (not supported for operation mode 1)
Interrupt request	• Receive interrupt (receive termination, detection of receive error, LIN synch break detection) • Transmit interrupt (transmit data empty) • Interrupt request to ICU (LIN synch field detection: LSYN) • Both the transmission and reception support the extended intelligent I/O service (EI^{2}OS) (except UART3)
Master/Slave type communication function (multiprocessor mode)	This function enables communication between 1 (only use master) and n (slave) (This function supports both of master and slave system.)
Synchronous mode	Master or slave function
Pin access	Capable of reading the state of serial I/O pin directly
LIN bus option	• Master device operation • Slave device operation • LIN synch break detection • LIN synch break generation • Detection of start/stop edges in LIN synch field connected to input capture 0, 1, 6, and 7
Synchronous serial clock	Synchronous serial clock can be continuously outputted to SCK pin for synchronous communication with start/stop bits.
Clock delay option	Special synchronous clock mode for delaying clock (enabled to SPI)

The LIN-UART operates in four different modes, which are determined by the MD0 and the MD1 bits of the LIN-UART serial mode register (SMR). Mode 0 and 2 are used for bidirectional serial communication, mode 1 for master/slave communication and mode 3 for LIN master/slave communication.

Table 17.1-2 LIN-UART Operation Modes

Operation mode		Data length		Synchronous method	Stop bit length	Data bit format
		Without Parity	With Parity			
0	Normal mode	7 bits or 8 bits		Asynchronous	1 bit or 2 bits	LSB first MSB first
1	Multi processor mode	7 bits or 8 bits +1*	-	Asynchronous		
2	Normal mode	8 bits		Synchronous	None, 1 bit, 2 bits	
3	LIN mode	8 bits	-	Asynchronous	1 bit	LSB first

- : Setting disabled

* : +1 is the address/data select bit (AD) used for controlling communication in multiprocessor mode.

The MD1 and MD0 bits of the LIN-UART serial mode register (SMR) determine the operation mode of LIN-UART as shown in the following table:

Table 17.1-3 LIN-UART Operation Modes

MD1	MD0	Mode	Type
0	0	0	Asynchronous (normal mode)
0	1	1	Asynchronous (multiprocessor mode)
1	0	2	Synchronous (normal mode)
1	1	3	Asynchronous (LIN mode)

Notes:

- Mode 1 operation is supported both for master and slave operation of LIN-UART in a master/ slave connection system.
- In Mode 3, the LIN-UART function is fixed to the communication format 8N1-Format, LSB first.
- If the mode is changed, LIN-UART cuts off all possible transmission or reception and awaits then new action.

■ LIN-UART Interrupt and EI^{2}OS

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
LIN-UART0 reception	#39(27H)$_H$	ICR14	0000BE$_H$	FFFF60$_H$	FFFF61$_H$	FFFF62$_H$	*1
LIN-UART0 transmission	#40(28H)$_H$	ICR14	0000BE$_H$	FFFF5C$_H$	FFFF5D$_H$	FFFF5E$_H$	*2
LIN-UART1 reception	#37(25H)$_H$	ICR13	0000BD$_H$	FFFF68$_H$	FFFF69$_H$	FFFF6A$_H$	*1
LIN-UART1 transmission	#38(26H)$_H$	ICR13	0000BD$_H$	FFFF64$_H$	FFFF65$_H$	FFFF66$_H$	*2
LIN-UART2 reception	#35(23H)$_H$	ICR12	0000BC$_H$	FFFF70$_H$	FFFF71$_H$	FFFF72$_H$	*1
LIN-UART2 transmission	#36(24H)$_H$	ICR12	0000BC$_H$	FFFF6C$_H$	FFFF6D$_H$	FFFF6E$_H$	*2
LIN-UART3 reception	#24(18H)$_H$	ICR06	0000B6$_H$	FFFF9C$_H$	FFFF9D$_H$	FFFF9E$_H$	*3
LIN-UART3 transmission	#26(1AH)$_H$	ICR07	0000B7$_H$	FFFF94$_H$	FFFF95$_H$	FFFF96$_H$	*2

*1: EI^{2}OS can only be used when the interrupt source that shares ICR12 to ICR14, and the interrupt vector is not used. Reception error can be detected. EI^{2}OS stop function is available.

*2: EI^{2}OS can only be used when the interrupt source that shares ICR07, ICR13, ICR14, and the interrupt vector is not used.

*3: EI^{2}OS can only be used when the interrupt source that shares ICR06 and the interrupt vector is not used. Reception error can be detected. EI^{2}OS stop function is available.

17.2 Configuration of LIN-UART

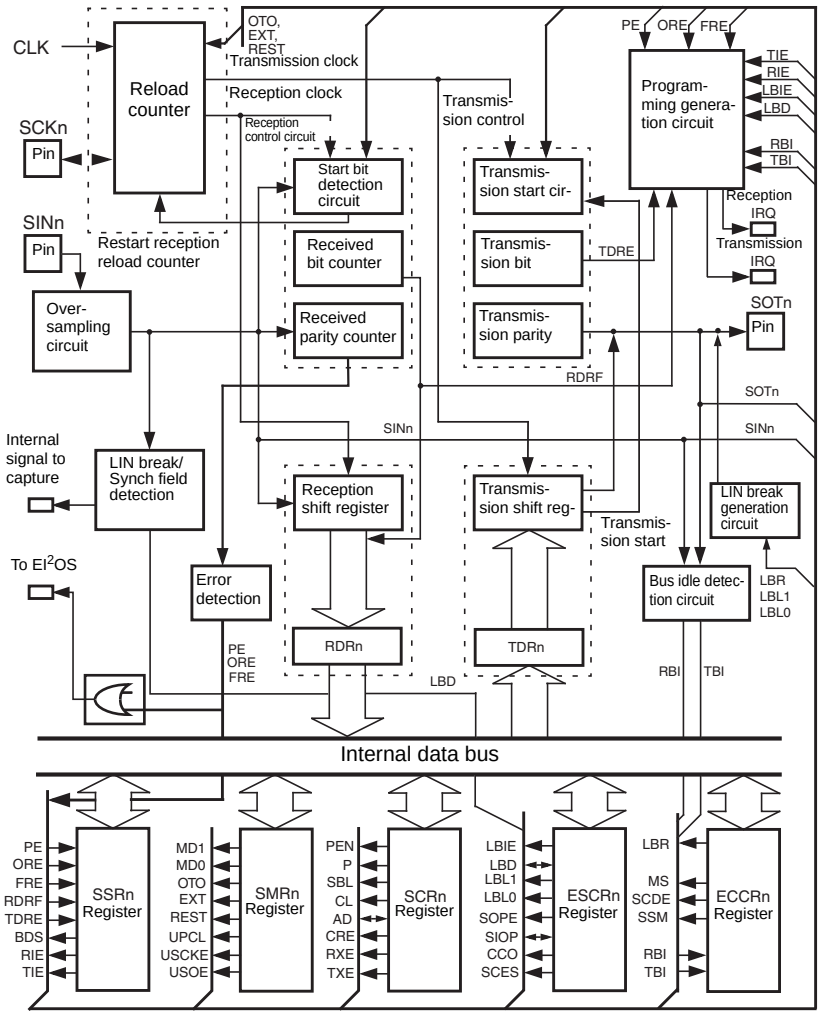
This section briefly outlines the blocks of the LIN-UART.

■ LIN-UART Consists of the Following Blocks.

- Reload counter
- Reception control circuit
- Reception shift register
- Reception data register (RDR)
- Transmission control circuit
- Transmission shift register
- Transmission data register (TDR)
- Error detection circuit
- Oversampling circuit
- Interrupt generation circuit
- LIN synch Break/synch field detection circuit
- Bus idle detection circuit
- LIN-UART serial mode register (SMR)
- Serial control register (SCR)
- Serial status register (SSR)
- Extended communication control register (ECCR)
- Extended status control register (ESCR)

■ Block Diagram of LIN-UART

Figure 17.2-1 Block Diagram of LIN-UART



n = 0, 1, 2, 3

■ Explanation of the Blocks

● Reload counter

15-bit reload counter that functions as the dedicated baud rate generator. The reload counter consists of a 15-bit register for the reload value. It generates the transmitting and receiving clocks with the external clock or the internal clock. The count value of the transmission reload counter can be read via the BGRn1/BGRn0.

● Reception control circuit

The reception control circuit consists of a received bit counter, start bit detection circuit, and received parity counter. The received bit counter counts reception data bits. When reception of one data item for the specified data length is completed, the received bit counter sets the flag in the LIN-UART reception data register. In this case, if the reception interrupt is enabled, the reception interrupt request is generated. The start bit detection circuit detects start bits from the serial input signal and sends a signal to the reload counter to synchronize it to the falling edge of these start bits. The received parity counter calculates the parity of the reception data.

● Reception shift register

The reception shift register fetches reception data input from the SINn pin, shifting the data bit by bit. When reception is completed, the reception shift register transfers receive data to the RDR register.

● Reception data register (RDR)

This register retains reception data. Serial input data is converted and stored in this register.

● Transmission control circuit

The transmission control circuit consists of a transmission bit counter, transmission start circuit, and transmission parity counter. The transmission bit counter counts transmission data bits. When the transmission of one data item of the specified data length is completed, the transmission bit counter sets the flag in the transmission data register. In this case, if the transmission interrupt is enabled, the transmission interrupt request is generated. The transmission start circuit starts transmission when data is written to TDR register. The transmission parity counter generates a parity bit for data to be transmitted if parity is enabled.

● Transmission shift register

The transmission shift register transfers data written to the TDR register to itself and outputs the data to the SOTn pin, shifting the data bit by bit.

● Transmission data register (TDR)

This register sets transmission data. Data written to this register is converted to serial data and outputted.

● Error detection circuit

The error detection circuit checks if there was any error at the end of reception. If an error has occurred, it sets the corresponding error flags.

Oversampling circuit

The oversampling circuit oversamples for five times in the asynchronous mode. The received value is determined by majority decision of sampling value. It is switched off in synchronous operation mode.

Interrupt generation circuit

The interrupt generation circuit controls all interrupt sources. If a corresponding interrupt enable bit is set, the interrupt will be generated immediately.

LIN synch break/synch field detection circuit

The LIN synch break/synch field detection circuit detects a LIN synch break when the LIN master node transmits a message header. The LBD flag is set when the LIN synch break is detected. An internal signal is output to the capture in order to detect the 1st and 5th falling edges of the LIN synch field and to measure the actual serial clock synchronization transmitted by the master node.

LIN synch break generation circuit

The LIN synch break generation circuit generates a LIN synch break of a determined length.

Bus idle detection circuit

The bus idle detection circuit detects if neither reception nor transmission is going on. In this case, the circuit generates the flag bits TBI and RBI.

LIN-UART serial mode register (SMR)

Operating functions are as follows:

- Selecting the LIN-UART operation mode
- Selecting a clock input source
- Selecting if an external clock is connected "one-to-one" or connected to the reload counter
- Resetting dedicated reload timer
- Resetting the LIN-UART software (preserving the settings of the registers)
- Specifying whether to enable/disable serial data output to the corresponding pin
- Specifying whether to enable/disable clock output to the corresponding pin

Serial control register (SCR)

Operating functions are as follows:

- Specifying whether to provide parity bits
- Selecting parity bits
- Specifying a stop bit length
- Specifying a data length
- Selecting a frame data format in mode 1
- Clearing the error flags

- Specifying whether to enable/disable transmission
- Specifying whether to enable/disable reception

● Serial status register (SSR)

Operating functions are as follows:

- Indicating status of receive/transmit operations and errors
- Specifying LSB first or MSB first as transfer direction
- Receive interrupt enable/disable
- Transmit interrupt enable/disable

● Extended status control register (ESCR)

- LIN synch break interrupt enable/disable
- Indicating LIN synch break detection
- Specifying LIN synch break length
- Directly accessing SINn and SOTn pins
- Specifying continuous clock output operation in LIN-UART synchronous clock mode
- Specifying sampling clock edge

● Extended communication control register (ECCR)

- Bus idle detection
- Setting synchronous clock
- LIN synch break generation

17.3 Pins of LIN-UART

This section lists and details the pins, interrupt sources, and registers of the LIN-UART.

■ Pins of LIN-UART

The LIN-UART pins also serve as general-purpose ports. Table 17.3-1 lists the pin functions, I/O formats, and settings required to use LIN-UART.

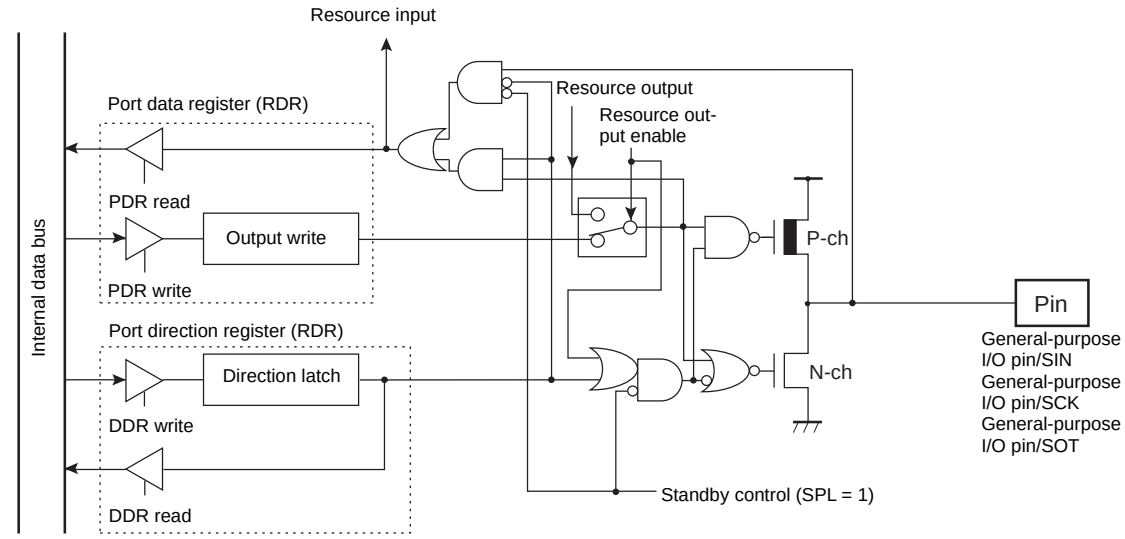
Table 17.3-1 Pins of LIN-UART

Pin name	Pin function	I/O type	Pull-up select	Standby control	Setting to use pin
PC0/SIN0 PC3/SIN1 PD0/SIN2 PD3/SIN3	Port I/O, serial data input	CMOS output, CMOS/CMOS hysteresis/ Automotive input	None	Yes	Sets to input port (DDR: corresponding bit = 0)
PC1/SOT0 PC4/SOT1 PD1/SOT2 PD4/SOT3	Port I/O, serial data output	CMOS output, CMOS hysteresis/ Automotive input			Set to output enable mode (SMRn: SOE = 1)
PC2/SCK0 PC5/SCK1 PD2/SCK2 PD5/SCK3	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)

See "3. DC Characteristics in ■ ELECTRICAL CHARACTERISTICS" in the data sheet for the standard value.

■ Block Diagram of LIN-UART Pins

Figure 17.3-1 Block Diagram of LIN-UART Pins



Standby control: stop mode (SPL =1), watch mode (SPL = 1), time-base timer mode (SPL = 1)

Note: Resource I/O signals are inputted or outputted from pins including resource functions.

17.4 LIN-UART Registers

This section lists LIN-UART registers.

■ List of LIN-UART Registers

Figure 17.4-1 List of LIN-UART Registers

• LIN-UART0

Address:		bit15 bit8	bit7 bit0
000035_H,	000034_H	SCR0 (serial control register)	SMR0 (serial mode register)
000037_H,	000036_H	SSR0 (serial status register)	RDR0/TDR0 (reception data register/transmission data register)
000039_H,	000038_H	ESCR0 (extended status control register)	ECCR0 (extended communication control register)
$00003B_H$,	$00003A_H$	BGR01 (baud rate generator register)	BGR00 (baud rate generator register)

• LIN-UART1

Address:		bit15 bit8	bit7 bit0
$0000C5_H$,	$0000C4_H$	SCR1 (serial control register)	SMR1 (serial mode register)
$0000C7_H$,	$0000C6_H$	SSR1 (serial status register)	RDR1/TDR1 (reception data register/transmission data register)
$0000C9_H$,	$0000C8_H$	ESCR1 (extended status control register)	ECCR1 (extended communication control register)
$0000CB_H$,	$0000CA_H$	BGR11 (baud rate generator register)	BGR10 (baud rate generator register)

• LIN-UART2

Address:		bit15 bit8	bit7 bit0
$0000E1_H$,	$0000E0_H$	SCR2 (serial control register)	SMR2 (serial mode register)
$0000E3_H$,	$0000E2_H$	SSR2 (serial status register)	RDR2/TDR2 (reception data register/transmission data register)
$0000E5_H$,	$0000E4_H$	ESCR2 (extended status control register)	ECCR2 (extended communication control register)
$0000E7_H$,	$0000E6_H$	BGR21 (baud rate generator register)	BGR20 (baud rate generator register)

• LIN-UART3

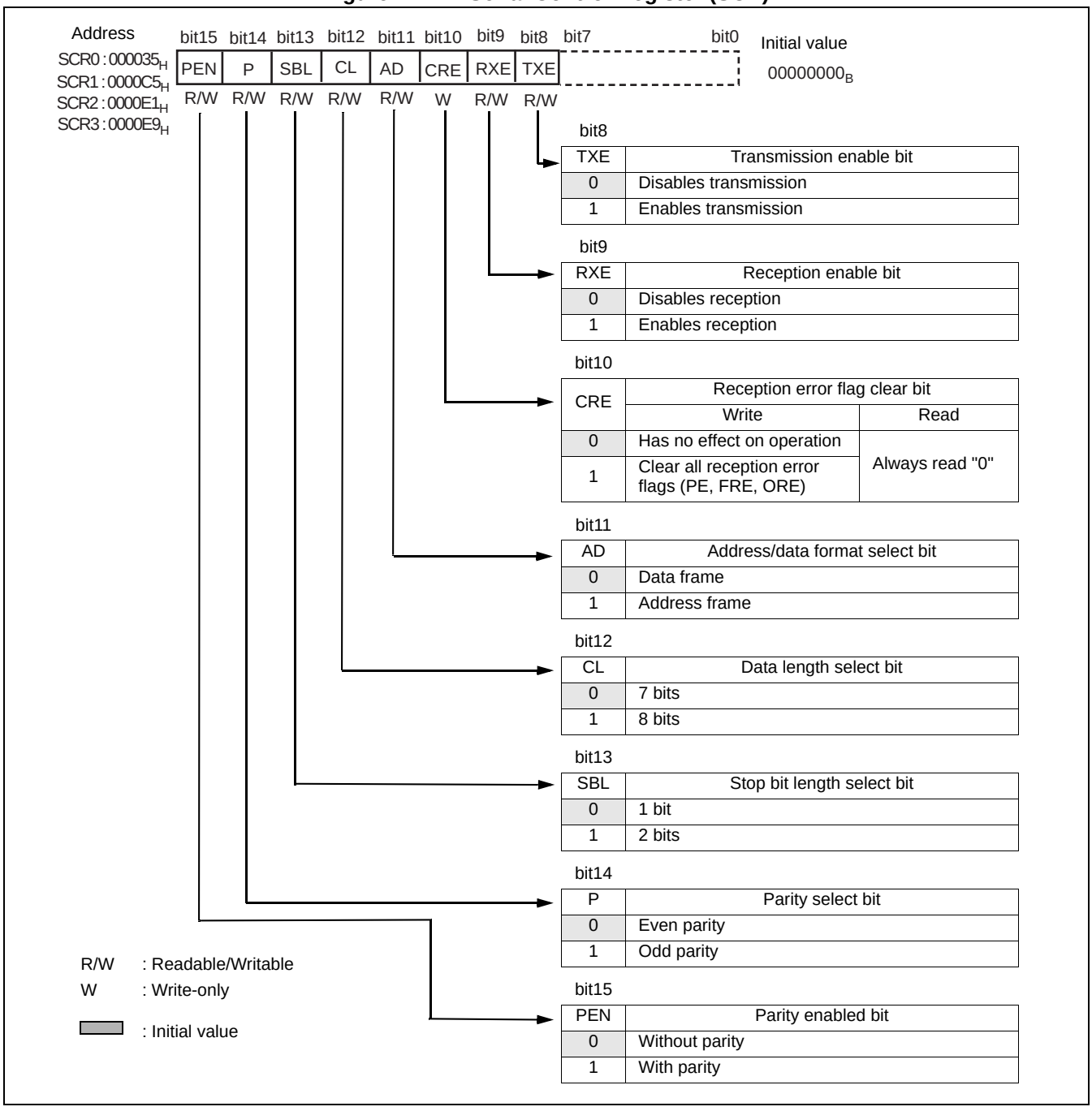
Address:		bit15 bit8	bit7 bit0
$0000E9_H$,	$0000E8_H$	SCR3 (serial control register)	SMR3 (serial mode register)
$0000EB_H$,	$0000EA_H$	SSR3 (serial status register)	RDR3/TDR3 (reception data register/transmission data register)
$0000ED_H$,	$0000EC_H$	ESCR3 (extended status control register)	ECCR3 (extended communication control register)
$0000EF_H$,	$0000EE_H$	BGR31 (baud rate generator register)	BGR30 (baud rate generator register)

17.4.1 Serial Control Register (SCR)

The serial control register (SCR) specifies parity, selects the stop bit and data lengths, selects a frame data format in mode 1, clears the reception error flag, and specifies whether to enable/disable transmission and reception.

■ Serial Control Register (SCR)

Figure 17.4-2 Serial Control Register (SCR)



bit8

TXE	Transmission enable bit
0	Disables transmission
1	Enables transmission

bit9

RXE	Reception enable bit
0	Disables reception
1	Enables reception

bit10

CRE	Reception error flag clear bit	
	Write	Read
0	Has no effect on operation	Always read "0"
1	Clear all reception error flags (PE, FRE, ORE)	

bit11

AD	Address/data format select bit
0	Data frame
1	Address frame

bit12

CL	Data length select bit
0	7 bits
1	8 bits

bit13

SBL	Stop bit length select bit
0	1 bit
1	2 bits

bit14

P	Parity select bit
0	Even parity
1	Odd parity

bit15

PEN	Parity enabled bit
0	Without parity
1	With parity

Table 17.4-1 Functions of Each Bit in Serial Control Register (SCR)

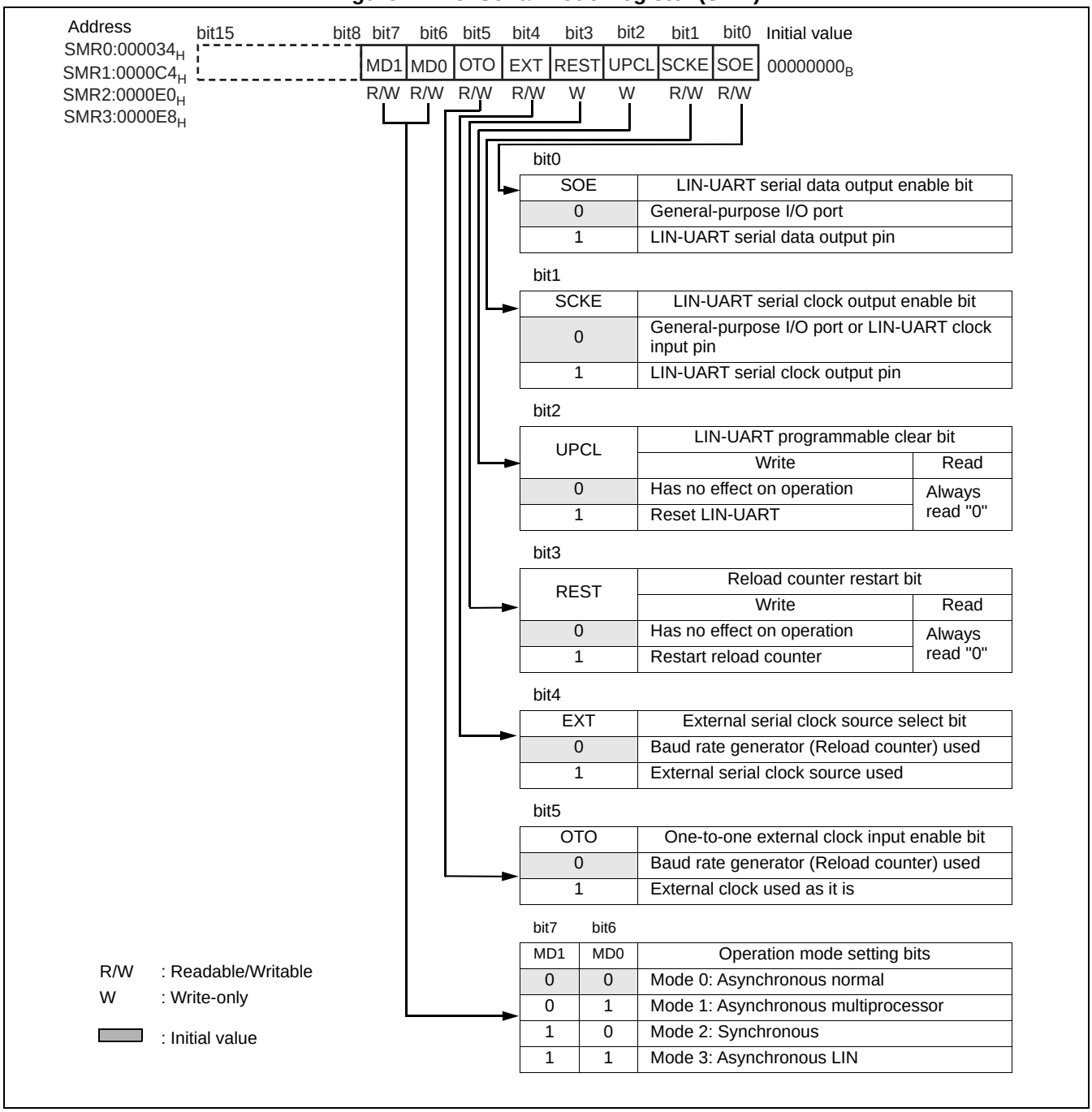
Bit name		Function
bit15	PEN: Parity enable bit	This bit selects whether to add a parity bit (during transmission) and whether to detect parity (during reception). Note: Parity bit is only provided in operation mode 0 and in operation mode 2 with start/stop bit (ECCR: SSM=1). This bit is fixed to "0" in mode 3 (LIN).
bit14	P: Parity selection bit	When parity is provided (SCR: PEN=1), this bit selects even (0) or odd (1) parity.
bit13	SBL: Stop bit length selection bit	This bit selects the length of the stop bit (frame end mark of transmission data) in operation modes 0 and 1 (asynchronous) or in operation mode 2 (synchronous) with start/stop bit (ECCR: SSM=1). This bit is fixed to "0" in mode 3 (LIN). Note: At reception, always first stop bit is detected.
bit12	CL: Data length selection bit	This bit specifies the length of transmission or reception data. This bit is fixed to "1" in modes 2 and 3.
bit11	AD: Address/data format selection bit	This bit specifies the frame data format to be transmitted and received in multiprocessor mode (mode 1). Writing to this bit is provided for a master CPU, reading from this bit for slave CPU. • When set to 0: data frame • When set to 1: address data frame The reading value is a value of last received data format. Note: For using this bit, see Section "17.8 Notes on Using LIN-UART".
bit10	CRE: Reception error flag clear bit	This bit clears the FRE, ORE, and PE flags of the serial status register (SSR). • Writing "1" to this bit clears the error flag. • Writing "0" has no effect. "0" is always read. Note: Clear reception error flags after the receive operation. When the reception error flag is cleared without disabling the reception, the reception is interrupted once at that timing and then it restarts. Therefore, when the reception is restarted, incorrect data might be received.
bit9	RXE: Reception operation enable bit	This bit enables/disables LIN-UART reception. • If this bit is set to "0", LIN-UART disables the reception of data frames. • If this bit is set to "1", LIN-UART enables the reception of data frames. The LIN synch break detection in mode 3 remains unaffected. Note: If reception is disabled (RXE=0) during receiving, it is stopped immediately. In this case, the data is not guaranteed.
bit8	TXE: Transmission operation enable bit	This bit enables/disables LIN-UART transmission. • If the bit is set to "0", LIN-UART disables the transmission of data frames. • If the bit is set to "1", LIN-UART enables the transmission of data frames. Note: If transmission is disabled (TXE=0) during transmitting, it is stopped immediately. In this case, the data is not guaranteed.

17.4.2 LIN-UART Serial Mode Register (SMR)

LIN-UART serial mode register (SMR) selects an operation mode and baud rate clock and specifies whether to enable/disable output of serial data and clocks to the corresponding pin.

■ LIN-UART Serial Mode Register (SMR)

Figure 17.4-3 Serial Mode Register (SMR)



bit0

SOE	LIN-UART serial data output enable bit
0	General-purpose I/O port
1	LIN-UART serial data output pin

bit1

SCKE	LIN-UART serial clock output enable bit
0	General-purpose I/O port or LIN-UART clock input pin
1	LIN-UART serial clock output pin

bit2

UPCL	LIN-UART programmable clear bit	
	Write	Read
0	Has no effect on operation	Always read "0"
1	Reset LIN-UART	

bit3

REST	Reload counter restart bit	
	Write	Read
0	Has no effect on operation	Always read "0"
1	Restart reload counter	

bit4

EXT	External serial clock source select bit
0	Baud rate generator (Reload counter) used
1	External serial clock source used

bit5

OTO	One-to-one external clock input enable bit
0	Baud rate generator (Reload counter) used
1	External clock used as it is

bit7	bit6	
MD1	MD0	Operation mode setting bits
0	0	Mode 0: Asynchronous normal
0	1	Mode 1: Asynchronous multiprocessor
1	0	Mode 2: Synchronous
1	1	Mode 3: Asynchronous LIN

Table 17.4-2 Functions of Each Bit in Serial Mode Register (SMR)

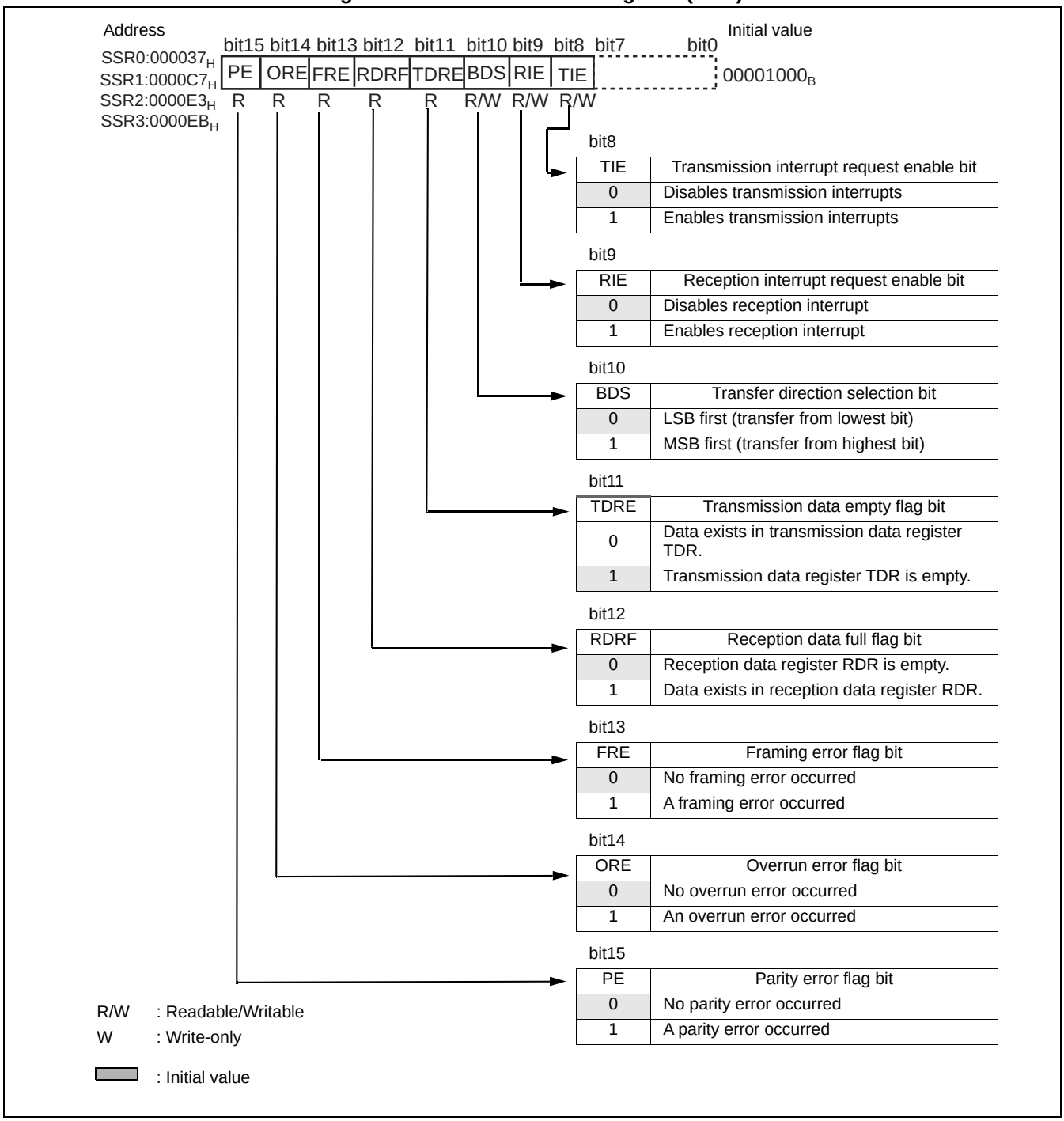
Bit name		Function
bit7, bit6	MD1, MD0: Operation mode setting bits	These two bits set the LIN-UART operation mode.
bit5	OTO: One-to-one external clock input enable bit	This bit sets an external clock directly to the LIN-UART serial clock by writing "1". This function is used for operation mode 2 (synchronous) slave operation (ECCR: MS=1). When EXT=0, this bit is fixed to "0".
bit4	EXT: External serial clock source select	This bit selects the clock input. When "0" is set to this bit, it selects the clock of internal baud rate generator (reload counter). When "1" is set to it, it selects the external serial clock source.
bit3	REST: Reload counter restart bit	If "1" is written to this bit, the reload counter is restarted. Writing "0" to this bit has no effect. "0" is always read.
bit2	UPCL: LIN-UART programmable clear bit (LIN-UART software reset)	Writing "1" to this bit resets LIN-UART immediately (LIN-UART software reset). The register settings are preserved. Possible reception or transmission will cut off. All of transmission/reception interrupt sources (TDRE, RDRF, LBD, PE, ORE, FRE) are cleared. Reset the LIN-UART after disabling the interrupt and transmission. Also, when the reception data register is cleared (RDR = 00_H), the reload counter restarts. Writing "0" to this bit has no effect. "0" is always read.
bit1	SCKE: LIN-UART serial clock output enable bit	This bit controls the serial clock I/O ports. When this bit is "0", SCKn pin operates as general-purpose I/O port or serial clock input pin. When this bit is "1", the pin operates as serial clock output pin and outputs clock in operation mode 2 (synchronous). Note: When using SCKn pin as serial clock input (SCKE=0) pin, set the corresponding DDR bit of general-purpose I/O port as input port. Also, select external clock (EXT = 1) using the clock selection bit. Reference: When the SCKn pin is assigned to serial clock output (SCKE=1), it functions as the serial clock output pin regardless of the status of the general-purpose I/O ports.
bit0	SOE: LIN-UART serial data output enable bit	This bit enables or disables the output of serial data. When this bit is "0", SOTn pin operates as general-purpose I/O port. When this bit is "1", SOTn pin operates as serial data output pins (SOTn). Reference: When the output of serial data is enabled (SOE=1), SOTn pin functions as SOTn regardless of the status of general-purpose I/O ports.

17.4.3 Serial Status Register (SSR)

The serial status register (SSR) checks the transmission and reception status and error status, and enables or disables the interrupts.

■ Serial Status Register (SSR)

Figure 17.4-4 Serial Status Register (SSR)



bit8

TIE	Transmission interrupt request enable bit
0	Disables transmission interrupts
1	Enables transmission interrupts

bit9

RIE	Reception interrupt request enable bit
0	Disables reception interrupt
1	Enables reception interrupt

bit10

BDS	Transfer direction selection bit
0	LSB first (transfer from lowest bit)
1	MSB first (transfer from highest bit)

bit11

TDRE	Transmission data empty flag bit
0	Data exists in transmission data register TDR.
1	Transmission data register TDR is empty.

bit12

RDRF	Reception data full flag bit
0	Reception data register RDR is empty.
1	Data exists in reception data register RDR.

bit13

FRE	Framing error flag bit
0	No framing error occurred
1	A framing error occurred

bit14

ORE	Overrun error flag bit
0	No overrun error occurred
1	An overrun error occurred

bit15

PE	Parity error flag bit
0	No parity error occurred
1	A parity error occurred

Table 17.4-3 Functions of Each Bit in Serial Status Register (SSR) (Sheet 1 of 2)

Bit name		Function
bit15	PE: Parity error flag bit	• This bit is set to "1" when a parity error occurs during reception at PE=1 and is cleared when "1" is written to the CRE bit of the LIN-UART serial mode register (SMR). • A reception interrupt request is outputted when this bit and the RIE bit are "1". • When this flag is set, the data in the reception data register (RDR) is invalid.
bit14	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 CRE bit of the LIN-UART serial mode register (SMR). • A reception interrupt request is outputted when this bit and the RIE bit are "1". • When this flag is set, the data in the reception data register (RDR) is invalid.
bit13	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 CRE bit of the LIN-UART serial mode register (SMR). • A reception interrupt request is outputted when this bit and the RIE bit are "1". • When this flag is set, the data in the reception data register (RDR) is invalid.
bit12	RDRF: Receive 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 RDR and cleared to "0" when the reception data register (RDR) is read. • A reception interrupt request is outputted when this bit and the RIE bit are "1".
bit11	TDRE: Transmission data empty flag bit	• This flag indicates the status of the transmission data register (TDR). • This bit is cleared to "0" when transmission data is written to TDR and indicates that valid data exists in TDR. This bit is set to "1" when data is loaded into the transmission shift register and transmission starts and indicates that no valid data exists in TDR. • A transmission interrupt request is generated if both this bit and the TIE bit are "1". • If the LBR bit in the extended communication control register (ECCR) register is set to "1" while the TDRE bit is "1", then this bit once changes to "0". After the completion of LIN synch break generation, the TDRE bit changes back to "1". Note: This bit is set to "1" as its initial value.
bit10	BDS: Transfer direction selection bit	This bit selects whether to transfer serial data from the least significant bit (LSB first, BDS=0) or the most significant bit (MSB first, BDS=1). Note: The high-order and low-order sides of serial data are interchanged with each other during reading from or writing to the serial data register. If this bit is set to another value after the data is written to the RDR register, the data becomes invalid. This bit is fixed to "0" in mode 3 (LIN).
bit9	RIE: Reception interrupt request enable bit	• This bit enables or disables the reception interrupt request output to the CPU. • When both this bit and the receive data flag bit (RDRF) are "1", or when one or more error flag bits (PE, ORE, FRE) is "1", then a reception interrupt request is outputted.

Table 17.4-3 Functions of Each Bit in Serial Status Register (SSR) (Sheet 2 of 2)

Bit name		Function
bit8	TIE: Transmission interrupt request enable bit	• This bit enables or disables the transmission interrupt request output to the CPU. • A transmission interrupt request is outputted when this bit and the TDRE bit are "1".

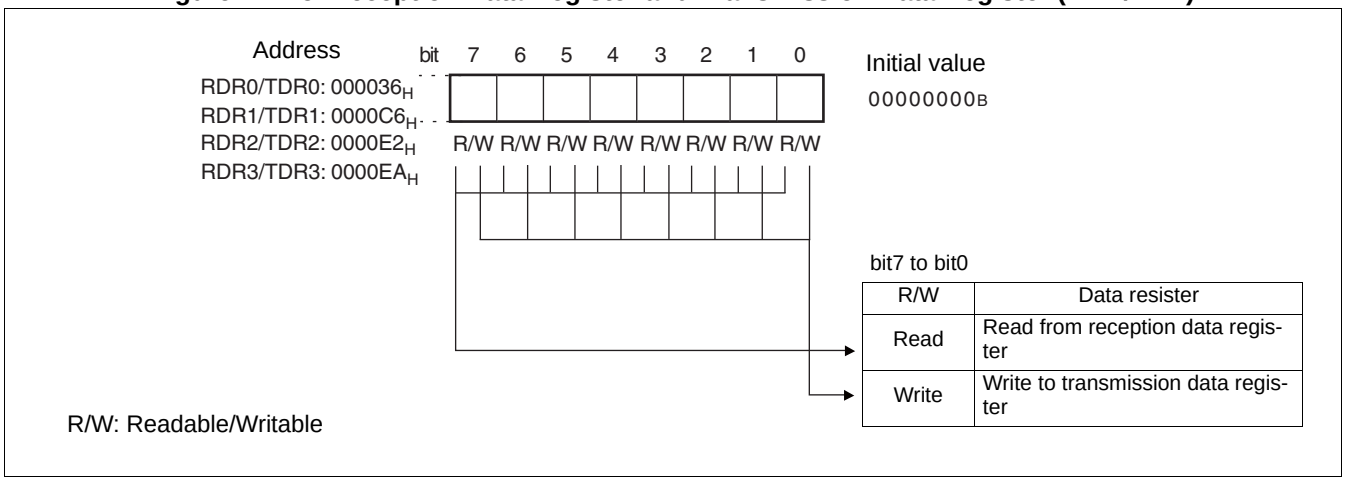
17.4.4 Reception Data Register and Transmission Data Register (RDR/TDR)

Both RDR and TDR registers are located at the same address. At reading, it functions as the reception data register. At writing, it functions as the transmission data register.

■ Reception Data Register and Transmission Data Register (RDR/TDR)

Figure 17.4-5 shows the bit configuration of reception data register and transmission data register (RDR/TDR).

Figure 17.4-5 Reception Data Register and Transmission Data Register (RDR/TDR)



Reception data register (RDR) is the data buffer register for serial data reception.

The serial data signal transmitted to the serial input pin (SINn pin) is converted in the shift register and stored in the reception data register (RDR).

When the data length is 7 bits, the upper bit (RDR: D7) contains "0".

When the reception data is stored in this register (RDR), the reception data full flag bit (SSR: RDRF) is set to "1". If a reception interrupt is enabled (SSR: RIE=1) at this point, generates a reception interrupt request.

Read the reception data register (RDR) when the reception data full flag bit (SSR:RDRF) is "1". The RDRF bit is cleared automatically to "0" when RDR is read.

Also the reception interrupt is cleared if it is enabled and no error has occurred.

Data in RDR is invalid when a reception error occurs (SSR: PE, ORE, or FRE = 1).

■ Transmission Data Register (TDR)

Transmission data register (TDR) is the data buffer register for serial data transmission.

When data to be transmitted is written to the transmission data register (TDR) in transmission enable state (SCR: TXE=1), it is transferred to the transmission shift register, then converted to serial data, and transmitted from the serial data output pin (SOTn pin).

If the data length is 7 bits, the upper bit (TDR: D7) is invalid data.

When transmission data is written to this register (TDR), the transmission data empty flag bit (SSR: TDRE) is cleared to "0".

When transfer to the transmission shift register is completed and transmission starts, the transmission data empty flag bit (SSR: TDRE) is set to "1".

When the transmission data empty flag bit (SSR: TDRE) is "1", the next part of transmission data can be written. If transmission interrupt has been enabled, a transmission interrupt is generated. Write the next part of transmission data when a transmission interrupt is generated or the transmission data empty flag bit (SSR: TDRE) is "1".

Note:

The transmission data register is a write-only register and the reception data register is a read-only register. These registers are located in the same address, so the read value is different from the write value. Therefore, instructions that perform a read-modify-write (RMW) operation, such as the INC/DEC instruction, cannot be used.

17.4.5 Extended Status Control Register (ESCR)

Extended status control register (ESCR) provides several LIN functions (enabling/ disabling LIN synch break interrupt, selecting LIN synch break length, and detecting LIN synch break), direct access to the SINn and SOTn pins and setting of continuous clock output in LIN-UART synchronous clock mode and sampling clock edge.

■ Bit Configuration of Extended Status Control Register (ESCR)

Figure 17.4-6 shows the bit configuration of the extended status control register (ESCR), and Table 17.4-4 shows the function of each bit.

Figure 17.4-6 Bit Configuration of Extended Status Control Register (ESCR)

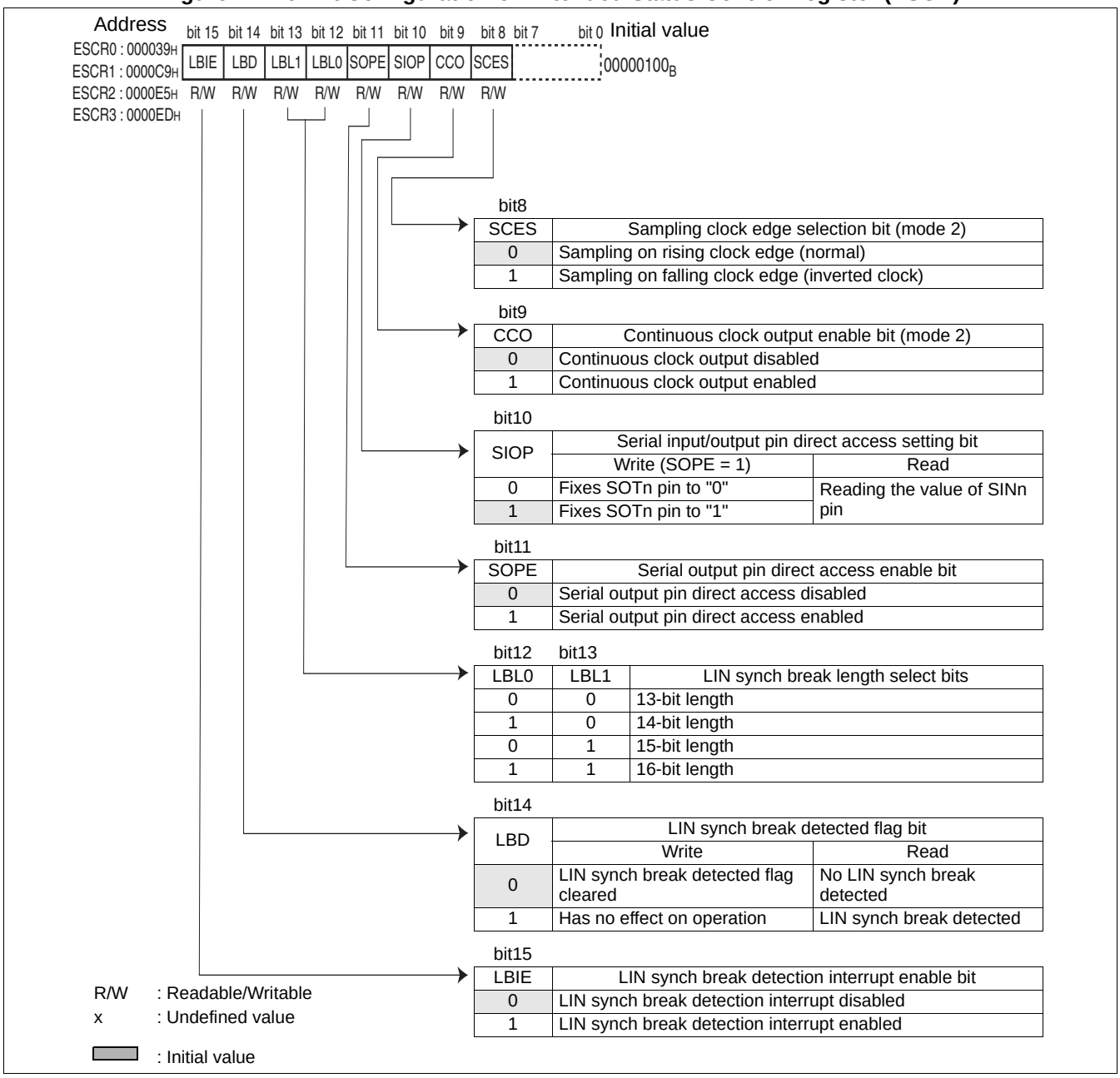


Table 17.4-4 Functions in Each Bit of the Extended Status Control Register (ESCR)

Bit name		Function
bit15	LBIE: LIN synch break detection interrupt enable bit	This bit enables/disables LIN synch break detection interrupt. An interrupt is generated when the LIN synch break detected flag (LBD) is "1" and the interrupt is enabled (LBIE = 1). This bit is fixed to "0" in operation modes 1 and 2.

Table 17.4-4 Functions in Each Bit of the Extended Status Control Register (ESCR)

Bit name		Function
bit14	LBD: LIN synch break detected flag bit	This bit is set to "1" if a LIN synch break was detected in operation mode 3 (the serial input is "0" when bit width is 11 bits or more). Writing "0" to it clears this bit and the corresponding interrupt, if it is enabled. Read-modify-write (RMW) instructions always read "1". Note that this dose not indicate a LIN synch break detection. Note: When LIN synch break detection is performed, disable reception (SCR:RXE=0) after enabling LIN synch break detection interrupt (LBIE=1).
bit13, bit12	LBL1/0: LIN synch break length selection bits	These bits specify the bit length for the LIN synch break generation time. Receiving a LIN synch break is always fixed to 11 bit times.
bit11	SOPE: Serial output pin direct access enable bit*	Setting this bit to "1" when serial data output is enabled (SMR:SOE = 1) enables direct writing to the SOTn pin.* Note: Setting value of this bit is valid only when the TXE bit of the serial control register (SCR) is "0".
bit10	SIOP: Serial input/output pin direct access bit*	Normal read instructions always return the value of the SINn pin. If writing this bit when direct access to the serial output pin is enabled (SOPE=1), the written value is reflected to the SOTn pin. Note: Setting value of this bit is valid only when the TXE bit of the serial control register (SCR) is "0". For the bit operation instruction, the bit value of the SOTn in the read cycle is returned.*
bit9	CCO: Continuous clock output enable bit	This bit enables a continuous serial clock output at the SCKn pin if LIN-UART operates in operation mode 2 (synchronous) and master setting and the SCKn pin is configured as a clock output. Note: When CCO bit is "1", use SSM bit of ECCR as setting to "1".
bit8	SCES: Sampling clock edge selection bit	Setting this bit to "1" inverts the sampling edge from the rising edge to the falling edge in operation mode 2 (synchronous) with the slave setting. If the SCKn pin is clock output in operation mode 2 with the master setting (ECCR: MS=0), the internal serial clock and the output clock signal are inverted. During operation modes 0, 1, 3, set this bit to "0".

* :

Table 17.4-5 Interaction of SOPE and SIOP

SOPE	SIOP	Writing to SIOP	Reading from SIOP
0	R/W	Has no effect (However, the written value is held)	Return the SINn value
1	R/W	Write "0" or "1" to SOTn	Return the SINn value
1	RMW	Reads current value of SOTn and write "0" or "1" to SIOP	

17.4.6 Extended Communication Control Register (ECCR)

The extended communication control register (ECCR) provides bus idle detection, synchronous clock settings, and the LIN synch break generation.

■ Bit Configuration of Extended Communication Control Register (ECCR)

Figure 17.4-7 shows the bit configuration of the extended communication control register (ECCR), and Table 17.4-6 shows the function of each bit.

Figure 17.4-7 Bit Configuration of Extended Communication Control Register (ECCR)

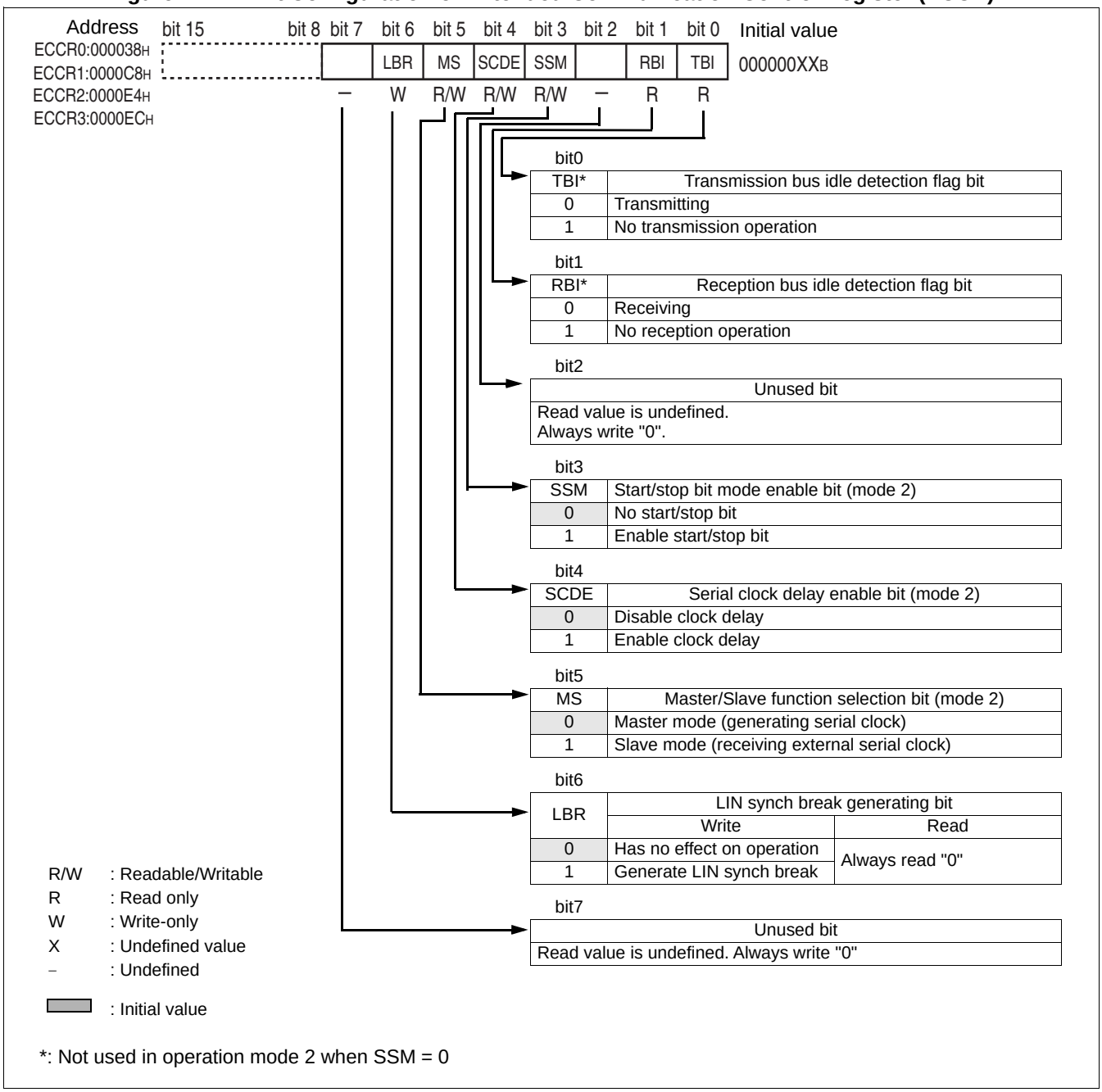


Table 17.4-6 Functions of Each Bit in the Extended Communication Control Register (ECCR)

Bit name		Function
bit7	Unused bit	This bit is unused. Read value is undefined. Always write "0"
bit6	LBR: LIN synch break generating bit	Writing "1" to this bit generates a LIN synch break of the length selected by the LBL0/LBL1 bits of the ESCR, if operation mode 3 is selected. Setting "0" to it in operation mode 0.

Table 17.4-6 Functions of Each Bit in the Extended Communication Control Register (ECCR)

Bit name		Function
bit5	MS: Master/Slave mode selection bit	This bit selects master or slave mode in mode 2. If master mode ("0") is selected, LIN-UART generates the synchronous clock. If slave mode ("1") is selected, LIN-UART receives external serial clock. This bit is fixed to "0" in modes 0, 1 and 3. Change this bit, when the SCR: TXE bit is "0". Note: If slave mode is selected, the clock source must be external and enabled the external clock input (SMR: SCKE = 0, EXT = 1, OTO = 1).
bit4	SCDE: Serial clock delay enable bit	Setting the SCDE bit to "1" in the master mode operation during mode 2 outputs a delayed serial clock as shown in Figure 17.7-5. This bit is enabled to SPI. This bit is fixed to "0" in modes 0, 1 and 3.
bit3	SSM: Start/Stop bit mode enable bit	This bit adds start and stop bits to the synchronous data format when set to "1" mode 2. This bit is fixed to "0" in modes 0, 1 and 3.
bit2	Unused bit	This bit is unused. Read value is undefined. Always write to "0".
bit1	RBI: Reception bus idle detection flag bit	This bit is "1" if there is no reception operation on the SIN pin and it is kept at "H". Do not use this bit in mode 2 when SSM=0.
bit0	TBI: Transmission bus idle detection flag bit	This bit is "1" if there is no transmission operation on the SOTn pin. Do not use this bit in mode 2 when SSM=0.

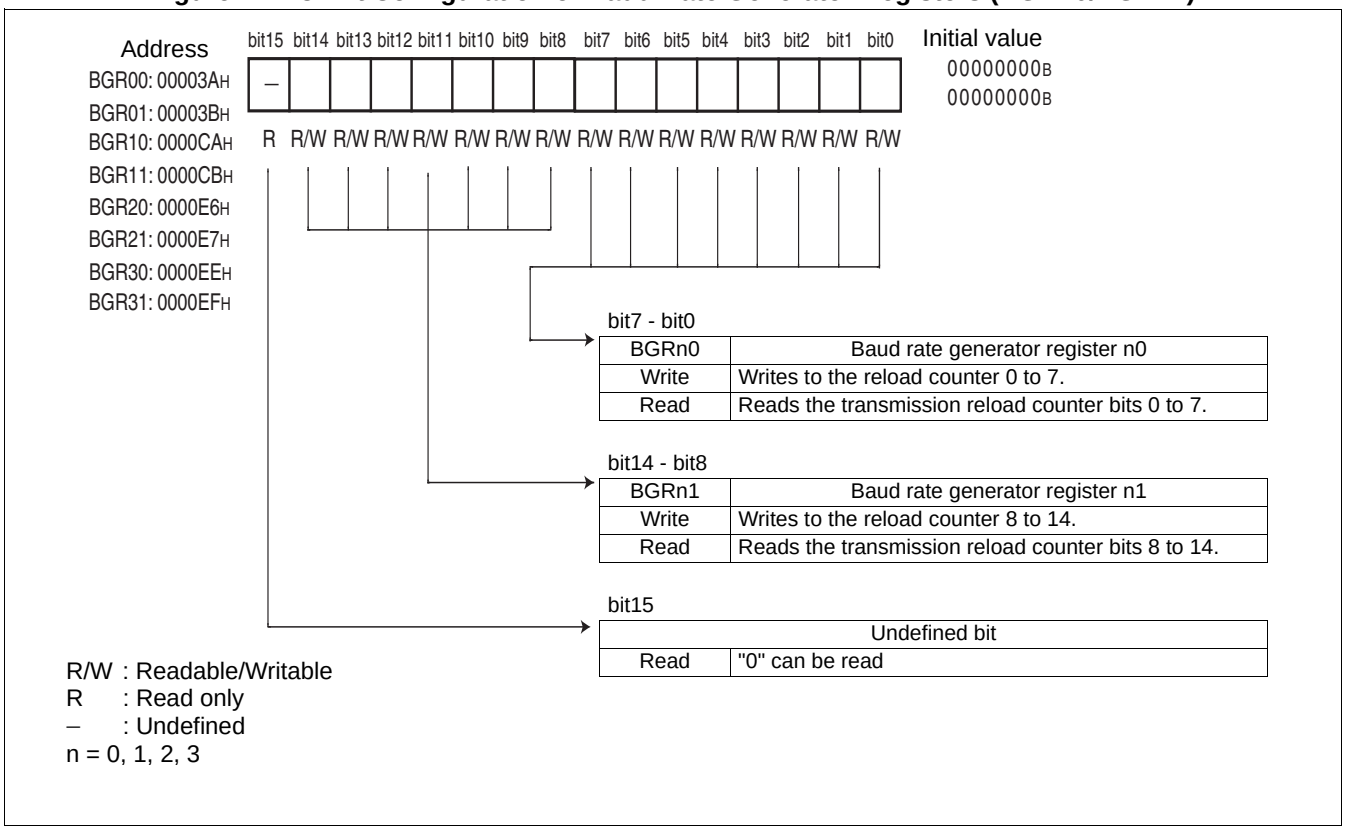
17.4.7 Baud Rate Generator Registers 0 and 1 (BGRn0/BGRn1)

The baud rate generator registers 0 and 1 (BGRn0/BGRn1) set the division ratio for the clock value. Also, the count value of the transmission reload counter can be read.

■ Bit Configuration of Baud Rate Generator Registers (BGRn0/BGRn1)

Figure 17.4-8 shows the bit configuration of the baud rate generator registers (BGRn0/BGRn1).

Figure 17.4-8 Bit Configuration of Baud Rate Generator Registers (BGRn0/BGRn1)



The baud rate generator registers determine the division ratio for the serial clock.

The BGRn1 corresponds to the upper bit and BGRn0 to lower bit, writing of the reload value to counter and reading of the transmission reload counter value are allowed. Also, byte and word access are enabled.

When writing the reload value to the baud rate generator registers, the reload counter starts counting.

17.5 Interrupts of LIN-UART

LIN-UART uses both reception and transmission interrupts. An interrupt request can be generated for either of the following causes:

- **Receive data is set in the reception data register (RDR), or a reception error occurs.**
- **Transmission data is transferred from the transmission data register (TDR) to the transmission shift register and transmission is started.**
- **LIN synch break is detected.**

The extended intelligent I/O service (EI^{2}OS) is available for these interrupts.

■ Interrupts of LIN-UART

Table 17.5-1 shows the interrupt control bits and interrupt cause of the LIN-UART.

Table 17.5-1 Interrupt Control Bits and Interrupt Cause of LIN-UART

Reception and transmission/ capture	Interrupt request flag bit	Flag register	Operation mode				Interrupt cause	Interrupt cause enable bit	Clearing interrupt request flag
			0	1	2	3			
Reception	RDRF	SSR	❍	❍	❍	❍	Receive data is written to RDR	SSR:RIE	Receive data is read
	ORE	SSR	❍	❍	❍	❍	Overrun error		"1" is written to reception error flag clear bit (SCR: CRE)
	FRE	SSR	❍	❍	△	❍	Framing error		
	PE	SSR	❍	×	△	×	Parity error		
	LBD	ESCR	×	×	×	❍	LIN synch break detection	ESCR:LBIE	"0" is written to ESCR: LBD.
Transmission	TDRE	SSR	❍	❍	❍	❍	Transmission register is empty.	SSR:TIE	Writing transmis-sion data
Input capture	ICP0/ ICP1/ ICP6/ ICP7	ICS01/ ICS67	×	×	×	❍	1st falling edge of LIN synch field	ICS01:ICE0/ICE1, ICS67:ICE6/ICE7	Disable ICP0/ICP1/ ICP6/ICP7
	ICP0 ICP1/ ICP6/ ICP7	ICS01/ ICS67	×	×	×	❍	5th falling edge of LIN synch field		

❍ : Bit used

× : Unused bit

△ : Availabe only when ECCR: SSM=1

- Reception interrupt

If one of the following events occurs in reception mode, the corresponding flag bit of the serial status register (SSR) is set to "1":

Data reception is completed

The received data was transferred from the serial input shift register to the reception data register (RDR) and data can be read (RDRF=1).

Overrun error

RDRF = 1 and RDR was not read by the CPU and next serial data is received (ORE=1).

Framing error

Stop bit reception error (FRE=1)

Parity error

Parity detection error (PE=1)

If at least one of these flag bits above is "1" and the reception interrupt is enabled (SSR: RIE = 1), a reception interrupt request is generated.

If the reception data register (RDR) is read, the RDRF flag is automatically cleared to "0". The error flags are cleared to "0", if "1" is written to the reception error flag clear bit (CRE) of the serial control register (SCR).

Note:

The CRE flag is write only and held "1" for one clock cycle when "1" is written to it.

- Transmission interrupt

If transmission data is transferred from the transmission data register (TDR) to the transfer shift register and transfer is started, the transmission data register empty flag bit (TDRE) of the serial status register (SSR) is set to "1". In this case, a transmission interrupt request is generated, if the transmission interrupt is enabled (SSR: TIE=1).

Note:

The initial value of TDRE (after hardware or software reset) is "1". Therefore, an interrupt is generated immediately then, if the TIE bit is set to "1". Also note, that the only way to clear the TDRE flag is writing data to the transmission data register (TDR).

- LIN synch break interrupt

This works for LIN slave operation in operation mode 3.

If the bus (serial input) is "0" for more than 11 bit times, the LIN synch break detected flag bit (LBD) of the extended status control register (ESCR) is set to "1". The LIN synch break interrupt and the LBD flag are cleared after writing "0" to the LBD flag. The LBD flag has to be cleared before input capture interrupt for LIN synch field.

When LIN synch break detection is performed, it is necessary to disable the reception (SCR: RXE=0).

- LIN synch field edge detection interrupts

This works for LIN slave operation in operation mode 3.

After LIN synch break detection, the internal signal is set to "1" at first falling edge of the LIN synch field and to the "0" after fifth falling edge. When the internal signal is set in the capture side to be inputted to capture (ICU0/1/6/7) and to be detected both edges, the interrupt occurs if the capture interrupt is enabled.

The difference of the count values detected in the capture is serial clock 8 bits for master, and new baud rate can be calculated.

When the falling edge of the start bit is detected, the reload counter restarts automatically.

■ LIN-UART Interrupts and EI^{2}OS

Table 17.5-2 LIN-UART Interrupts and EI^{2}OS

Channel	Interrupt No.	Interrupt control register		Vector table address			EI^{2}OS
		Register name	Address	Lower	Upper	Bank	
LIN-UART0 reception	#39(27_H)	ICR14	$0000BE_H$	$FFFF60_H$	$FFFF61_H$	$FFFF62_H$	*1
LIN-UART0 transmission	#40(28_H)	ICR14	$0000BE_H$	$FFFF5C_H$	$FFFF5D_H$	$FFFF5E_H$	*2
LIN-UART1 reception	#37(25_H)	ICR13	$0000BD_H$	$FFFF68_H$	$FFFF69_H$	$FFFF6A_H$	*1
LIN-UART1 transmission	#38(26_H)	ICR13	$0000BD_H$	$FFFF64_H$	$FFFF65_H$	$FFFF66_H$	*2
LIN-UART2 reception	#35(23_H)	ICR12	$0000BC_H$	$FFFF70_H$	$FFFF71_H$	$FFFF72_H$	*1
LIN-UART2 transmission	#36(24_H)	ICR12	$0000BC_H$	$FFFF6C_H$	$FFFF6D_H$	$FFFF6E_H$	*2
LIN-UART3 reception	#24(18_H)	ICR06	$0000B6_H$	$FFFF9C_H$	$FFFF9D_H$	$FFFF9E_H$	*3
LIN-UART3 transmission	#26($1A_H$)	ICR07	$0000B7_H$	$FFFF94_H$	$FFFF95_H$	$FFFF96_H$	*2

*1: EI^{2}OS can only be used when the ICR12 to ICR14 and interrupt source that shares the interrupt vector are not used. Reception error can be detected. EI^{2}OS stop function is available.

*2: EI^{2}OS can only be used when the ICR07, ICR13, ICR14, and interrupt source that shares the interrupt vector are not used.

*3: EI^{2}OS can only be used when the ICR06 and the interrupt source that shares the interrupt vector are not used. Reception error can be detected. EI^{2}OS stop function is available.

■ LIN-UART EI^{2}OS Functions

LIN-UART has a circuit for operating EI^{2}OS, which can be started up for either reception or transmission interrupts.

- For reception

EI^{2}OS can be used if other interrupt is not enabled because the UART shares the interrupt control registers with transmission interrupt and other UART.

- For transmission

The interrupt control register shares with the transmission interrupt or other UART. Therefore, EI^{2}OS can be used if other interrupt is not enabled.

17.5.1 Timing of Reception Interrupt Generation and Flag Set

The following are the reception interrupt causes: completion of reception (SSR: RDRF) and occurrence of a reception error (SSR: PE, ORE, or FRE).

■ Timing of Reception Interrupt Generation and Flag Set

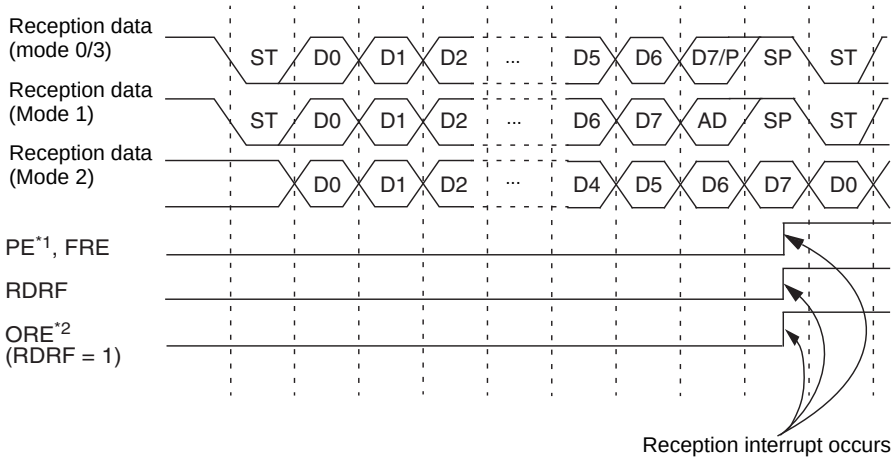
The received data is stored in the RDR register if the first stop bit is detected in mode 0, 1, 2 (if SSM = 1), 3, or the last data bit was detected in mode 2 (if SSM = 0). Each flag is set if the reception is completed (SSR:RDRF = 1) and the reception error occurs (SSR: PE, ORE, or FRE=1). In this case, if the reception interrupt is enabled (SSR:RIE=1), reception interrupt occurs.

Note:

If a reception error has occurred in each mode, the reception data register (RDR) contains invalid data.

Figure 17.5-2 shows the reception operation and flag set timing.

Figure 17.5-1 Reception Operation and Flag Set Timing

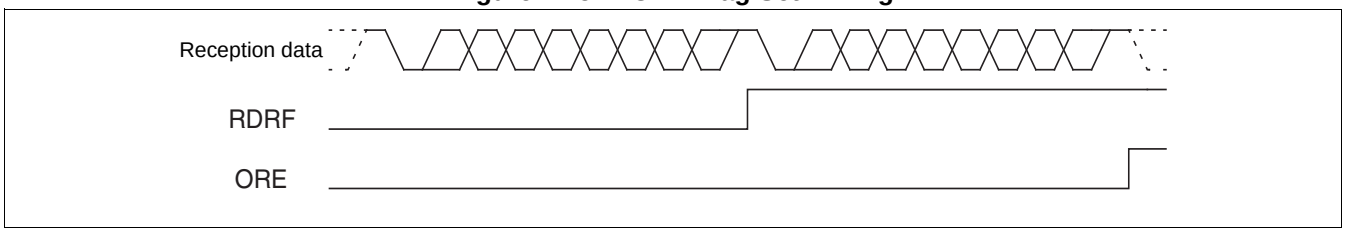


*1: The PE flag will always remain "0" in mode 1 or 3.
*2: An overrun error occurs, if next data is transferred before the reception data is read (RDRF=1).
ST: Start Bit SP: Stop Bit AD: Mode 1 (multiprocessor) address/data selection bit

Note:

Figure 17.5-2 does not show all possible reception options for mode 0. This is an example for "7P1" and "8N1" (P = "even parity" or "odd parity").

Figure 17.5-2 ORE Flag Set Timing



17.5.2 Timing of Transmission Interrupt Generation and Flag Set

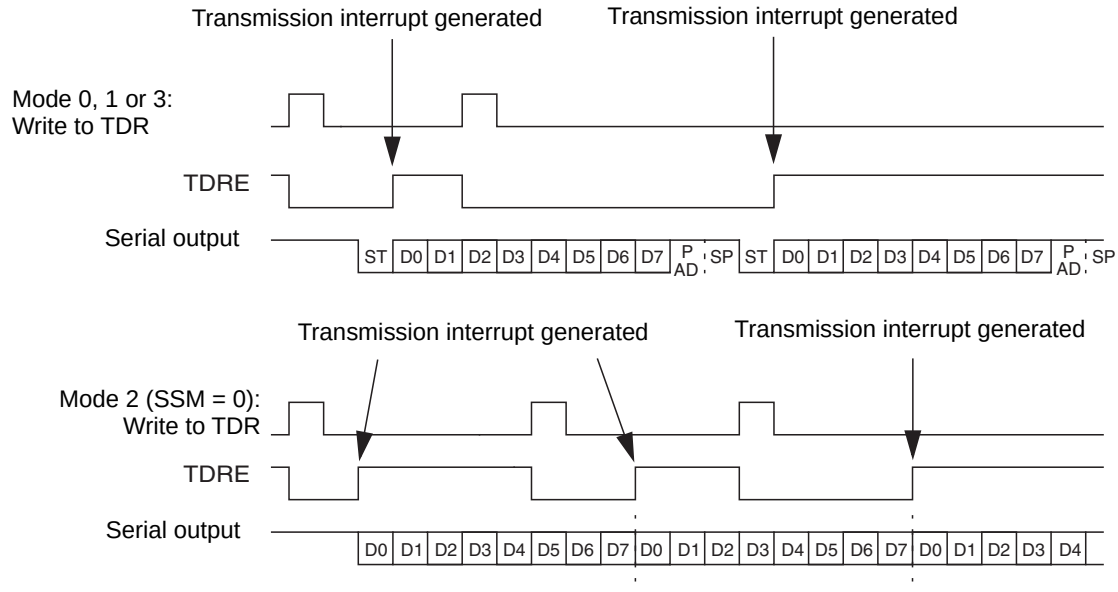
A transmission interrupt is generated when the transmission data is transferred from transmission data register (TDR) to transmission shift register and transmission is started.

■ Timing of Transmission Interrupt Generation and Flag Set

When the data written to the TDR register is transferred to the transmission shift register and the transmission is started, next data to be written is enabled (SSR: TDRE=1). Then, if transmission interrupt is enabled (SSR: TIE=1), the transmission interrupt occurs. Because the TDRE bit is "read only", it only can be cleared to "0" by writing data into TDR.

Figure 17.5-3 shows the transmission operation and flag set timing for the each modes of LIN-UART.

Figure 17.5-3 Transmission Operation and Flag Set Timing



ST: Start bit D0 to D7: data bits P: Parity SP: Stop bit
AD: Address/data selection bit (mode1)

Note:

Figure 17.5-3 does not show all possible transmission options for mode 0. This is an example for only "8p1" (p = "even parity or "odd parity").

Parity bit is not provided in mode 3 or mode 2, if SSM = 0.

■ Transmission Interrupt Request Generation Timing

If the TDRE flag is set to "1" when a transmission interrupt is enabled (SSR: TIE=1), transmission interrupt is generated.

Note:

A transmission interrupt is generated immediately after the transmission interrupt is enabled (SSR:TIE=1) because the TDRE bit is set to "1" as its initial value. TDRE can be cleared only by writing new data to the transmission data register (TDR). Carefully specify the transmission interrupt enable timing.

17.6 LIN-UART Baud Rates

One of the following can be selected for the LIN-UART transmission/reception clock source:

- **Dedicated baud rate generator (Reload Counter)**
- **Input external clock to baud rate generator (Reload Counter)**
- **External clock (directly use SCKn pin input clock)**

■ UART Baud Rate Selection

One of the following three types of baud rates can be selected. The baud rate selection circuit is designed as shown in Figure 17.6-1.

● Baud rates determined using the dedicated baud rate generator (reload counter) with internal clock divided

LIN-UART has two independent internal reload counters for transmission and reception serial clock. The baud rate can be selected via the 15-bit reload value determined by the baud rate generator registers 1, 0 (BGRn1, BGRn0).

The reload counter divides the internal clock by set value.

It is used in asynchronous or synchronous (master) mode.

Internal clock and baud rate generator clock is selected for the setting of clock source (SMR: EXT=0, OTO=0).

● Baud rates determined using the dedicated baud rate generator (reload counter) with external clock divided

An external clock is used for the clock source of the reload counter.

The baud rate can be selected via the 15-bit reload value determined by the baud rate generator registers 1, 0 (BGRn1, BGRn0).

The reload counter divides the external clock by set value.

It is used in asynchronous mode.

Select external clock and baud rate generator clock for the setting of the clock source (SMR: EXT=1, OTO=0).

This was designed to use the oscillators with special frequencies and having the possibility to divide them.

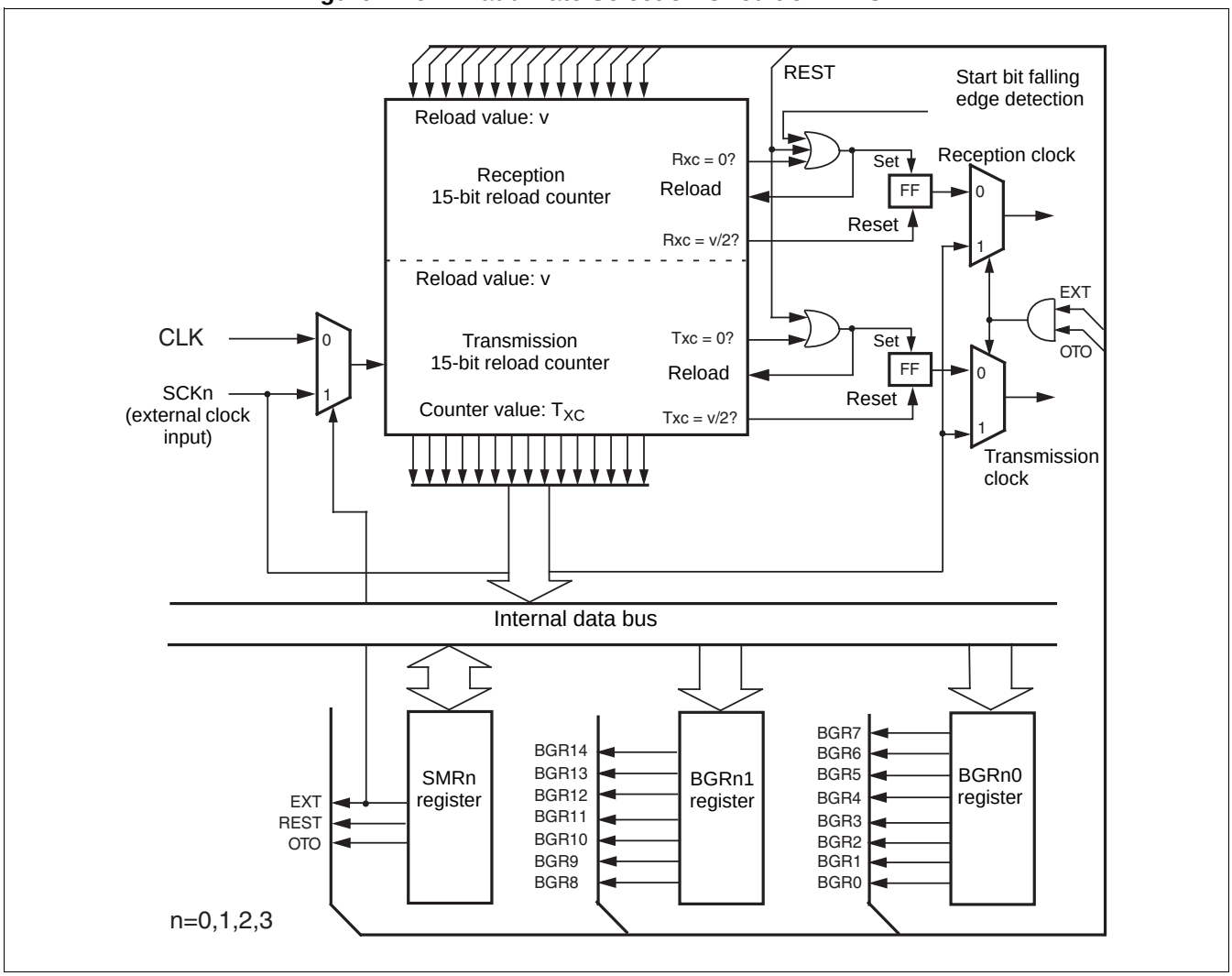
● Baud rates using external clock (one-to-one mode)

The clock input from LIN-UART clock pulse input pins (SCKn) is used as it is (synchronous mode 2 slave operation (ECCR: MS=1)).

It is used in synchronous mode (slave).

Select external clock and direct use of external clock for the setting of clock source (SMR: EXT=1, OTO=1).

Figure 17.6-1 Baud Rate Selection Circuit of LIN-UART



17.6.1 Setting the Baud Rate

This section describes how the baud rates are set and the resulting serial clock frequency is calculated.

■ Calculating the Baud Rate

The both 15-bit reload counters are programmed by the baud rate generator registers 1, 0 (BGRn1/BGRn0).

The calculation formula for the baud rate is as follows.

Reload value::

$$v = \left(\frac{\phi}{b} \right) - 1$$

v: reload value b: baud rate ϕ: machine clock, external clock frequency

Example of calculation

If the machine clock is 32 MHz, the internal clock is used, and the desired baud rate is 19200 bps, then the reload value v is calculated as shown below.

Reload value:

$$v = \left(\frac{32 \times 10^6}{19200} \right) - 1 = 1665$$

The exact baud rate can then be calculated:

$$b = \frac{\phi}{(v + 1)} = \frac{16 \times 10^6}{1666} = 19207.6831$$

Note:

Setting the reload value to "0" stops the reload counter. For this reason, the minimum division ratio is 2.

For transmission/reception in asynchronous mode, the reload value must be greater than or equal to 4 because 5 times over-sampling is performed to determine the reception value.

■ Reload Value and Baud Rate for Each Clock Speed

Reload value and baud rate for each clock speed is shown in Table 17.6-1.

Table 17.6-1 Reload Value and Baud Rate

Baud rate	8 MHz		10 MHz		16 MHz		20 MHz		32 MHz	
	Reload value	dev.	Reload value	dev.	Reload value	dev.	Reload value	dev.	Reload value	dev.
4 M	-	-	-	-	-	-	4	0	7	0
2 M	-	-	4	0	7	0	9	0	15	0
1 M	7	0	9	0	15	0	19	0	31	0
500000	15	0	19	0	31	0	39	0	63	0
460800	-	-	-	-	-	-	-	-	68	-0.16
250000	31	0	39	0	63	0	79	0	127	0
230400	-	-	-	-	-	-	-	-	138	0.08
153600	51	-0.16	64	-0.16	103	-0.16	129	-0.16	207	-0.16
125000	63	0	79	0	127	0	159	0	255	0
115200	68	-0.64	86	0.22	138	0.08	173	0.22	277	0.08
76800	103	-0.16	129	-0.16	207	-0.16	259	-0.16	416	0.08
57600	138	0.08	173	0.22	277	0.08	346	-0.06	554	-0.01
38400	207	-0.16	259	-0.16	416	0.08	520	0.03	832	-0.030
28800	277	0.08	346	<0.01	554	-0.01	693	-0.06	1110	<0.01
19200	416	0.08	520	0.03	832	-0.03	1041	0.03	1665	0.02
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	3070	<0.01
9600	832	0.04	1041	0.03	1666	0.02	2083	0.03	3332	<0.01
7200	1110	<0.01	1388	<0.01	2221	<0.01	2777	<0.01	4443	<0.01
4800	1666	0.02	2082	-0.02	3332	<0.01	4166	<0.01	6666	<0.01
2400	3332	<0.01	4166	<0.01	6666	<0.01	8332	<0.01	13332	<0.01
1200	6666	<0.01	8334	0.02	13332	<0.01	16666	<0.01	26666	<0.01
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-

The unit of frequency deviation (dev.) is %.

Note:

The maximum baud rate for synchronous mode is 1/5 of the machine clock.

■ External Clock

If the EXT bit of the SMR is set to "1", an external clock is selected. In the baud rate generator, the external clock can be used in the same way as the internal clock.

When slave operation is used in synchronous mode 2, select the one-to-one external clock input mode (SMR:OTO=1). In this mode, the external clock input to SCKn is input directly to the UART serial clock.

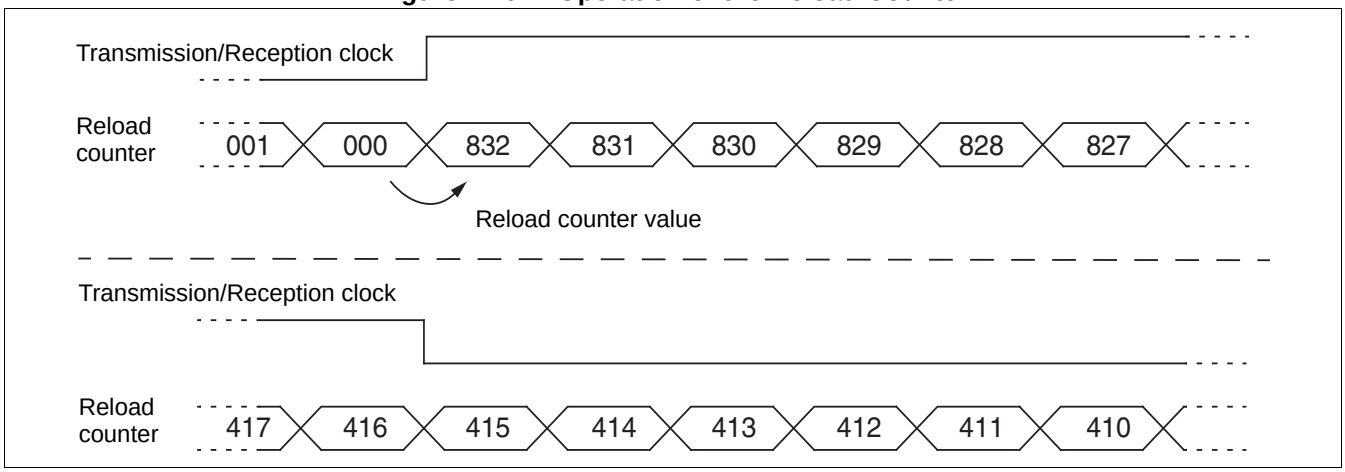
Note:

The external clock signal is synchronized to the internal clock in the LIN-UART. This means that indivisible external clock rates will result in phase unstable signals.

■ Operation of the Reload Counter

Figure 17.6-2 shows the operation of the 2 reload counters when the reload value is 832.

Figure 17.6-2 Operation of the Reload Counter



Note:

The falling edge of the serial clock signal is generated after the reload value divided by 2 ((v+1)/2) is counted.

17.6.2 Reload counter

The reload counter is a 15-bit reload counter that functions as dedicated baud rate generator. The transmission/reception clock is generated by the external or internal clock.
The count value in the transmission reload counter can be read from the baud rate generator registers (BGR1, BGR0).

■ Function of Reload Counter

The reload counter has the transmission and reception reload counters and functions as dedicated baud rate generator. It consists of a 15-bit register for the reload value and generates the transmission/reception clocks by the external or internal clock. The count value in the transmission reload counter can be read from the baud rate generator registers (BGR1, BGR0).

● Count start

When the reload value is written to the baud rate generator registers (BGR1, BGR0), the reload counter starts counting.

● Restart

The reload counter restarts under the following conditions.

Common for transmission/reception reload counter

- UART programmable reset (SMR:UPCL bit)
- Programmable restart (SMR:REST bit)

Reception reload counter

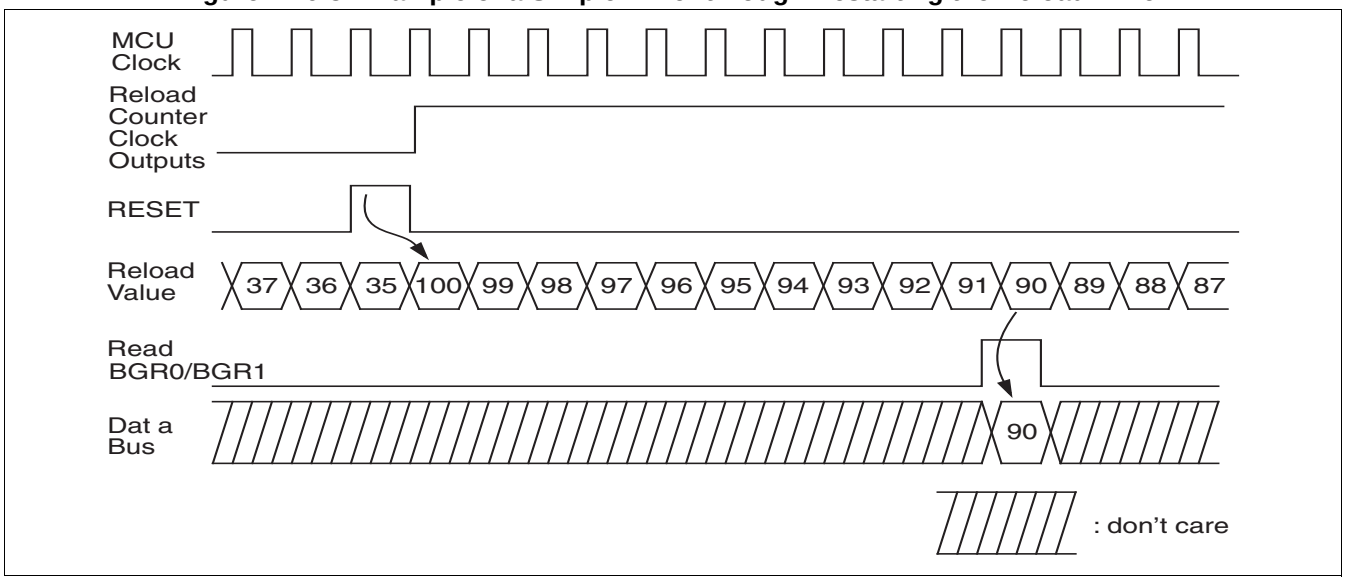
- Falling edge of start bit is detected in asynchronous mode

  If the REST bit of the serial mode register (SMR) is set to "1", both reload counters are restarted at the next clock cycle.

  This feature is intended to use the transmission reload counter as a simple timer.

  Figure 17.6-3 shows a possible usage of this feature (assume that the reload value is 100).

Figure 17.6-3 Example of a Simple Timer through Restarting the Reload Timer



In this case, machine cycle after restart cyc is calculated as follows:

cyc = v - c + 1 = 100 - 90 + 1 = 11

v: reload value, c: reload counter value

Note:

The reload counter restarts also when UART is reset by writing "1" to SMR:UPCL bit.

- Automatic restart (reception reload counter only)

  In asynchronous mode, if a falling edge of a start bit is detected, the reception reload counter is restarted. This is intended to synchronize the reception shift register to the reception data.

● Clearing counters

The reload value of the baud rate generator registers (BGR1, BGR0) and the reload counters are cleared to 00_H by the reset and the counters stop.

Although the counter value is temporarily cleared to 00_H by the LIN-UART reset (writing "1" to SMR:UPCL), the reload counter restarts since the reload value is retained. The counter value is not cleared to 00_H by the restart setting (writing "1" to SMR:REST), and the reload counter restarts.

17.7 Operation of LIN-UART

LIN-UART operates in operation mode 0 for bidirectional serial communication, in mode 1 as master or slave in multiprocessor communication, and in mode 2 and 3 in bidirectional communication as master or slave.

■ Operation of LIN-UART

● Operation mode

There are four LIN-UART operation modes: modes 0 to 3. As listed in Table 17.7-1, an operation mode can be selected according to the inter-CPU communication method and data transfer method.

Table 17.7-1 LIN-UART Operation Modes

Operation mode		Data length		Synchronous method	Stop bit length	Data bit format
		Without parity	With parity			
0	Normal mode	7 bits or 8 bits		Asynchronous	1 bit or 2 bits	LSB-first MSB-first
1	Multiprocessor mode	7 bits or 8 bits + 1*	-	Asynchronous		
2	Normal mode	8 bits		Synchronous	None, 1 bit, 2 bits	
3	LIN mode	8 bits	-	Asynchronous	1 bit	LSB-first

-: Setting disabled

*: "+1" means the address/data selection bit (AD) used for controlling communication in the multiprocessor mode

Note:

Mode 1 operation is supported both for master or slave operation of LIN-UART in a master/slave connection system. In Mode 3, the communication format is fixed to 8N1, LSB first.

If the mode is changed, LIN-UART cuts off all possible transmission or reception and awaits then new action.

■ Inter-CPU Connection Method

External clock one-to-one connection (normal mode) or master/slave connection (multiprocessor mode) can be selected. For either connection method, the data length, whether to enable parity, and the synchronization method must be common to all CPUs. Select an operation mode as follows:

- In the one-to-one connection method, operation mode 0 or 2 must be used in the two CPUs. Select operation mode 0 for asynchronous mode and operation mode 2 for synchronous mode. Note, that one CPU has to set to the master and the other to the slave in mode 2.
- Select operation mode 1 for the master/slave connection method and use it either for the master or slave system.

■ Synchronous Method

In asynchronous operation, LIN-UART reception clock is automatically synchronized to the falling edge of a received start bit. In synchronous mode, the synchronization is performed either by the clock signal of the master device or by the clock signal if operating as master.

■ Signaling

NRZ (Non Return to Zero)

■ Enabling Transmission/Reception

LIN-UART controls both transmission and reception using SCR: TXE bit and SCR: RXE bit respectively.

To disable the transmission or reception, follow the procedure described below.

- If reception operation is disabled during reception, finish reception and read the reception data register (RDR). Then stop the reception operation.
- If the transmission operation is disabled during transmission, wait until transmission finishes and then stop the transmission operation.

17.7.1 Operation in Asynchronous Mode (Operation Modes 0 and 1)

When LIN-UART is used in operation mode 0 (normal mode) or operation mode 1 (multiprocessor mode), the asynchronous transfer mode is selected.

■ Operation in Asynchronous Mode

- Transmission/Reception data format

Transmit/reception data always begins with a start bit ("L" level) followed by transmission/reception for a specified data bit length, and ends with at least one stop bit ("H" level).

The bit transfer direction (LSB first or MSB first) is determined by the BDS bit of the serial status register (SSR). The parity bit (if enabled) is always placed between the last data bit and the first stop bit.

In operation mode 0, the length of the data frame can be 7 bits or 8 bits, with or without parity, and the stop bit length (1 or 2).

In operation mode 1, the length of the data frame can be 7 bits or 8 bits with a following address-/data-bit instead of a parity bit. The stop bit length (1 or 2) can be selected.

The calculation formula for the bit length of a transmission/reception frame is:

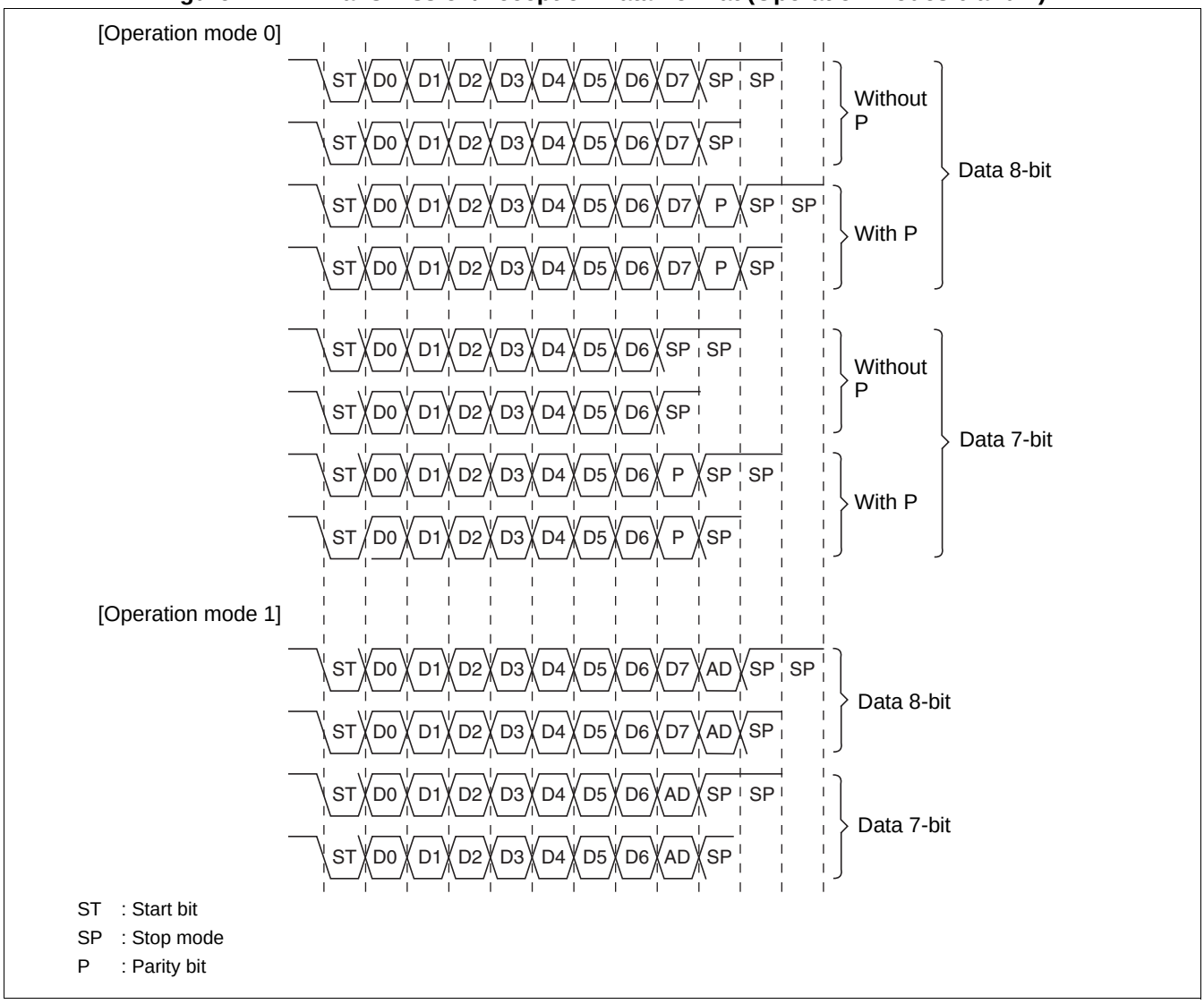
Length = 1 + d + p + s

(d = number of data bits [7 or 8], p = parity [0 or 1],

s = number of stop bits [1 or 2])

Figure 17.7-1 shows the data format in the asynchronous mode.

Figure 17.7-1 Transmission/Reception Data Format (Operation Modes 0 and 1)



Note:

If BDS bit of the serial status register (SSR) is set to "1" (MSB first), the bit stream processes in the following order: D7, D6, ..., D1, D0 (P).

● Transmission operation

If the transmission data register empty flag bit (TDRE) of the serial status register (SSR) is "1", transmission data is allowed to be written to the transmission data register (TDR). When data is written, the TDRE flag goes "0". If the transmission operation is enabled (TXE of the serial control register (SCR)=1), the data is written to the transmission shift register and the transmission starts at the next clock cycle of the serial clock, beginning with the start bit.

If transmission interrupt is enabled (TIE = 1), the interrupt is generated by the TDRE flag. Note, that the initial value of the TDRE flag is "1", an interrupt will occur immediately after "1" is written to TIE in this case.

When data length is set to 7 bits (CL=0), MSB bit of TDR becomes unused bit regardless of setting for transfer direction selection bit (BDS) (LSB first or MSB first).

Note:

As the initial value of transmission data empty flag bit (SSR: TDRE) is "1" if the transmission interrupt is enabled (SSR: TIE=1), the interrupt occurs immediately.

- Reception operation

Reception operation is performed when it is enabled (SCR: RXE=1). If a start bit is detected, a data frame is received according to the data format specified by the SCR. In case of errors, the corresponding error flags are set (SSR: PE, ORE, FRE). After the reception of the data frame, the data is transferred from the reception shift register to the reception data register (RDR) and the receive data register full flag bit (SSR: RDRF) is set to "1". In this case, if the reception interrupt request is enabled (SSR: RIE=1), the reception interrupt request is occurred.

When reading data after reception of one frame data, check the error flag state and read reception data from the RDR register if the reception is performed normally. If the reception error occurs, perform error processing.

When the reception data is read, the receive data register full flag bit (SSR: RDRF) is cleared to "0".

When data length is set to 7 bits (CL=0), MSB bit of TDR becomes unused bit regardless of setting for transfer direction selection bit (BDS) (LSB first or MSB first).

Note:

Only when the receive data register full flag bit (SSR: RDRF) is set to "1" and no errors have occurred (SSR: PE, ORE, FRE=0), the reception data register (RDR) contains valid data.

- Used clock

Use the internal clock or external clock. Select the baud rate generator (SMR: EXT = 0 or 1, OTO = 0) for desired baud rate.

- Stop bit

1- or 2-stop bit can be selected at the transmission. When 2-stop bit is selected, both stop bits are detected at the reception.

When first stop bit is detected, the receive data register full flag bit (SSR: RDRF) is "1". Then, when the start bit is not detected, the reception bus idle flag (ECCR:RBI) is set to "1", indicating no reception operation.

- Error detection

In mode 0, the parity, overrun, and framing errors can be detected.

In mode 1, the overrun and framing errors can be detected, and the parity error cannot be detected.

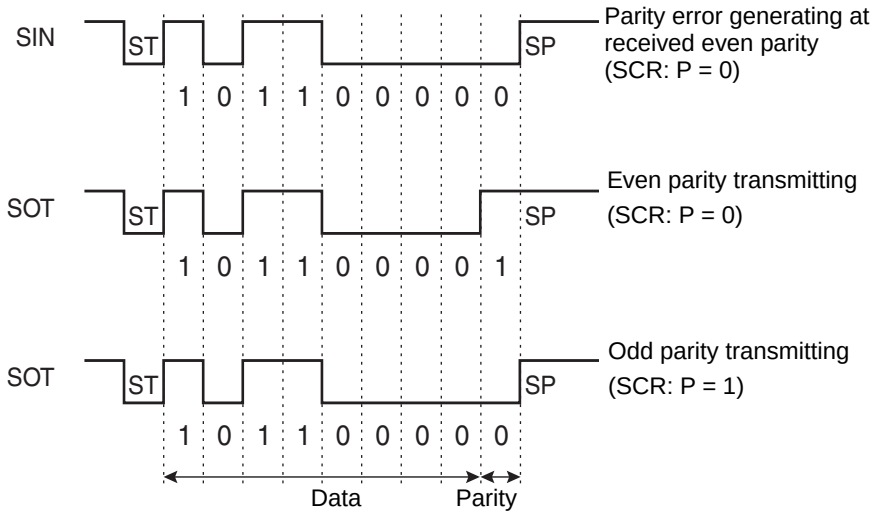
- Parity

Parity can set to add (transmission) or detect (reception) the parity bit.

The parity enable bit (SCR: PEN) specifies whether parity is enabled or disabled, and parity selection bit (SCR: P) selects the even/odd parity.

In operation mode 1, the parity cannot be used.

Figure 17.7-2 Transmission Data when Parity Enabled



ST: Start bit, SP: Stop bit at parity ON (PEN=1)

Note: Parity cannot be used at operation mode 1.

- Data signal type

The data signal type is NRZ data format.

- Data transition method

The direction of the bit stream (LSB first or MSB first) is determined by BDS bit of the Serial Status Register (SSR).

17.7.2 Operation in Synchronous Mode (Operating Mode 2)

The clock synchronous transfer method is used for LIN-UART operation mode 2 (normal mode).

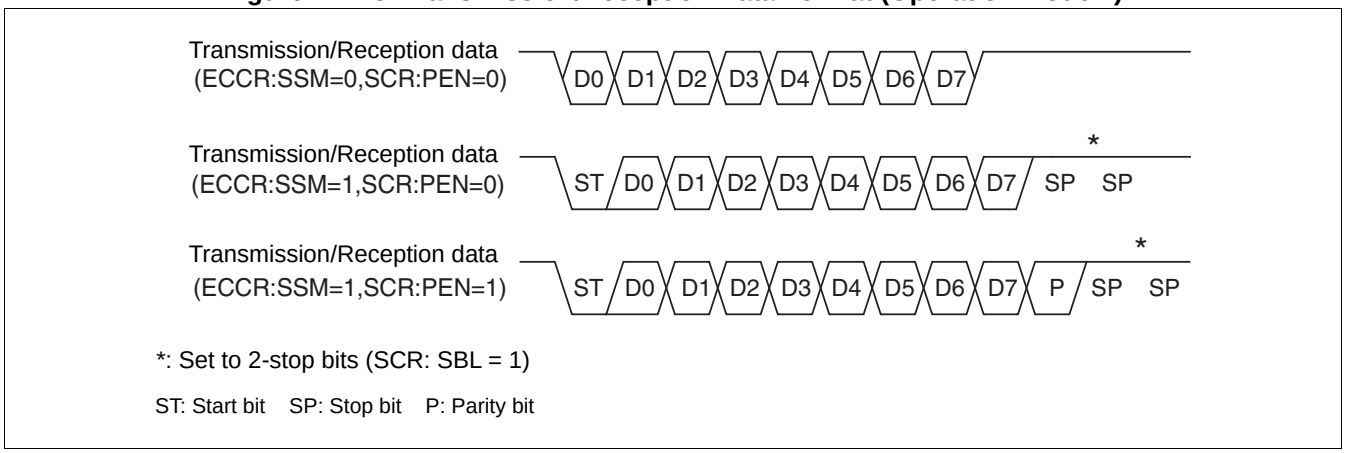
■ Operation in Synchronous Mode (Operation Mode 2)

● Transmission/Reception data format

In the synchronous mode, 8-bit data is transmitted/received with or without start/stop bits depending the selection (ECCR: SSM). Also, when the start/stop bit is provided (ECCR: SSM = 1), presence or absence of the parity bit can be selected (SCR: PEN).

Figure 17.7-3 illustrates the data format in the synchronous operation mode.

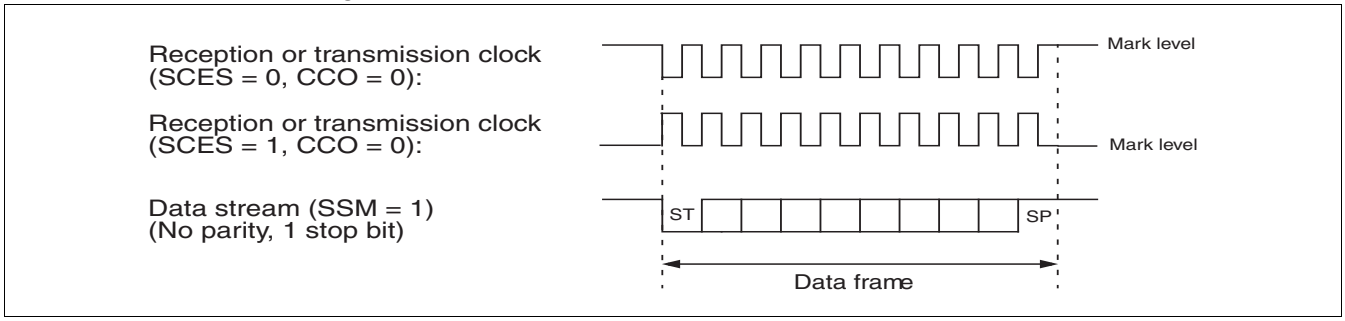
Figure 17.7-3 Transmission/Reception Data Format (Operation Mode 2)



● Clock inversion function

If the SCES bit of the extended status control register (ESCR) is set to "1", the serial clock is inverted. Therefore, in slave mode LIN-UART samples the data at the falling edge of the received serial clock. Note, that in master mode if SCES is set to "1", the mark level is "0".

Figure 17.7-4 Transfer Data Format with Clock Inversion



- Start/stop bits

If the SSM bit of the extended communication control register (ECCR) is set to "1", the start and stop bits are added as in the asynchronous mode.

- Clock supply

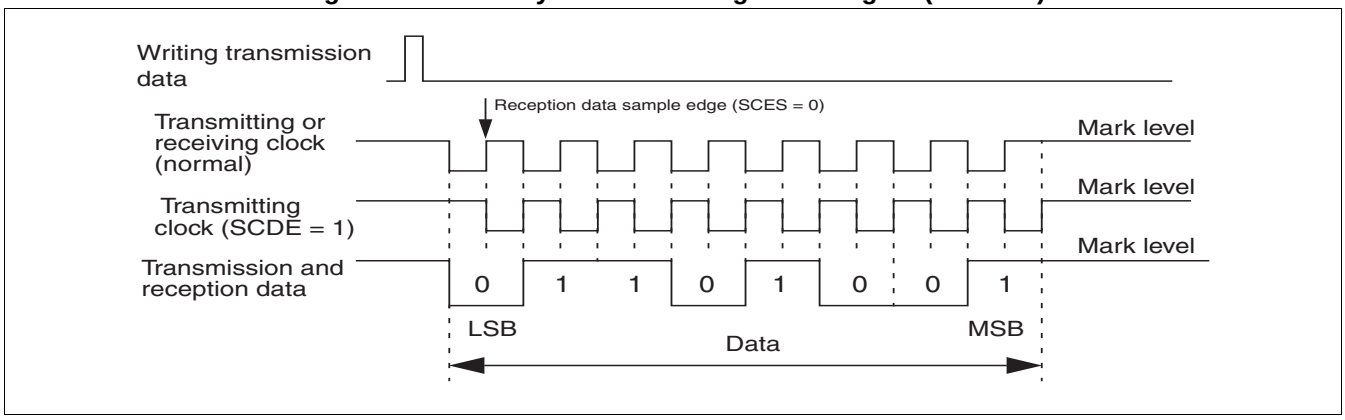
In clock synchronous mode (normal), the number of the transmit/reception bits must be equal to the number of the clock cycles. When the start/stop bit is enabled, the number of the added start/stop bits must be equal, as well.

When the serial clock output is enabled (SMR:SCKE=1) in master mode (ECCR:MS=0), a synchronous clock is output automatically at transmission/reception. When the serial clock output is disabled (SMR:SCKE=0) or the slave mode is selected (ECCR:MS=1), the clock for each bit of transmit/reception data must be supplied from the outside.

While there is no transmission/reception, the clock signal must be kept at "H" as the mark level.

If the SCDE bit of the ECCR register is "1", a delayed transmit clock is output as shown in Figure 17.7-5. This function is required when the receiving device samples data at the rising or falling edge of the clock.

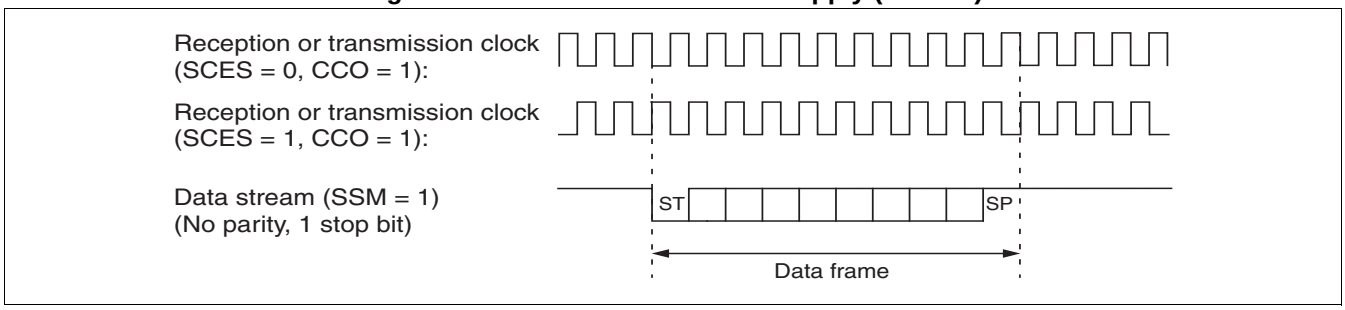
Figure 17.7-5 Delayed Transmitting Clock Signal (SCDE=1)



If the SCES bit of the ESCR register is "1", the UART clock signal is inverted. Receiving data is sampled at the falling edge of the clock. In this case, the serial data must be valid value at the falling edge of the clock.

If the CCO bit of ESCR register is "1", the serial clock output of the SCKn pin is continuously supplied in the master mode. In this mode, be sure to add the start/stop bits (SSM = 1) to identify the start and end of data frame. Figure 17.7-6 shows the operation of this function.

Figure 17.7-6 Continuous Clock Supply (Mode 2)



● Error detection

If no start/stop bits are selected (ECCR: SSM = 0), only overrun errors are detected.

● Communication settings for synchronous mode

For communication in the synchronous mode, following settings have to be done:

- Baud rate generator registers (BGR0/BGR1)

  Set the desired value for the dedicated baud rate reload counter.

- Serial mode register (SMR)

  MD1, MD0 : 10_B (Mode 2)

  SCKE : "1" . . . . . .for the dedicated baud rate reload counter

  "0" . . . . . .for external clock input

  SOE : "1" . . . . . .for transmission and reception

  "0" . . . . . .for reception only

- Serial control register (SCR)

  RXE, TXE : set one of these bits to "1"

  AD : no address/data selection - don't care

  CL : automatically fixed to 8-bit data - don't care

  CRE : "1" to clear receive error flags. Suspend transmission and reception.

  -- when SSM=0:

  PEN, P, SBL: don't care

  -- when SSM=1:

  PEN : "1" . . . . . .if parity bit is added/detected, "0" . . . . . if not

  P : "0" . . . . . .for even parity, "1" . . . . . odd parity

  SBL : "1" . . . . . .for 2 stop bits, "0" . . . . . for 1 stop bit.

- Serial status register (SSR)

  BDS : "0" . . . . . .for LSB first, "1" . . . . . for MSB first

  RIE : "1" . . . . . .if reception interrupts are used; "0" . . . . . reception interrupts are disabled.

  TIE : "1" . . . . . .if transmission interrupts are used;"0" . . . . . transmission interrupts are disabled.

- Extended communication control register (ECCR)

  SSM : "0" . . . . . .if no start/stop bits are desired (normal) "

  "1" . . . . . .for adding start/stop bits (extended function)

  MS : "0" . . . . . .for master mode (LIN-UART generates the serial clock)

  "1" . . . . . .for slave mode (LIN-UART receives serial clock from the master device)

Note:

To start the communication, write data to TDR register.

Only when receiving the data, disable the serial output (SMR: SOE = 0) and write the dummy data

to TDR.

By enabling the continuous clock and start/stop bits, the bi-directional communication the same as the asynchronous mode is allowed.

17.7.3 Operation with LIN Function (Operation Mode 3)

LIN-UART can be used either as LIN-Master or LIN-Slave in operation mode 3. For this LIN function, setting the LIN-UART to mode 3 configures the data format to 8N1-LSB-first format.

■ Operation in Asynchronous LIN Mode

● Operation as LIN master

In LIN mode, the master determines the baud rate of the whole bus, therefore slaves devices have to synchronize to the master. Therefore, the desired baud rate remains fixed in master operation after initialization.

Writing "1" into the LBR bit of the extended communication control register (ECCR) generates a 13 bits to 16 bits time "L" level on the SOTn pin, which is the LIN synch break and the start of a LIN message.

Thereby the TDRE flag of the serial status register (SSR) goes "0" and is reset to "1" (initial value) after the break, and generates a transmission interrupt for the CPU (if TIE of SSR is "1").

The length of the LIN break to be sent can be determined by the LBL1/LBL0 bits of the ESCR as follows:

Table 17.7-2 LIN Break Length

LBL0	LBL1	Break length
0	0	13 bits
1	0	14 bits
0	1	15 bits
1	1	16 bits

The synch field is sent as byte data of 55_H after the LIN break. To prevent a transmission interrupt, the 55_H can be written to the TDR just after writing the "1" to the LBR bit, although the TDRE flag is "0".

● Operation as LIN slave

In LIN slave mode, LIN-UART has to synchronize to the master's baud rate. If reception is disabled (RXE = 0), however, LIN break interrupt is enabled (LBIE = 1) LIN-UART will generate a reception interrupt. In this case, the LBD bit of ESCR is set to "1".

Writing "0" to this bit clears the reception interrupt request flag.

For the calculation of the baud rate, the UART0 operation is explained as an example. When UART0 detects first falling edge of synch field, set the internal signal inputted to the input capture (ICU0) to "H" and start ICU0. This internal signal is set to "L" at fifth falling edge, ICU0 must be set to the LIN mode (ICE01). Also, the ICU0 interrupt must be set to enable and to detect both edges (ICS01). The time which the ICU0 input signal is "1" is the value multiplied the baud rate by 8.

Therefore, baud rate setting value is summarized as follows:

without free-run timer overflow : BGR value = {(b-a) × Fe/(8 × ϕ)} -1

with free-run timer overflow : BGR value = {(max + b-a) × Fe/(8 × ϕ)}-1

max : the free-run timer maximum value

a : ICU data register value after the 1st interrupt

b : ICU data register value after the 2nd interrupt

ϕ is the machine clock frequency (MHz).

Fe is the external clock frequency (MHz).

When the internal baud rate generator is used(EXT=0), it calculates as Fe=ϕ.

Note:

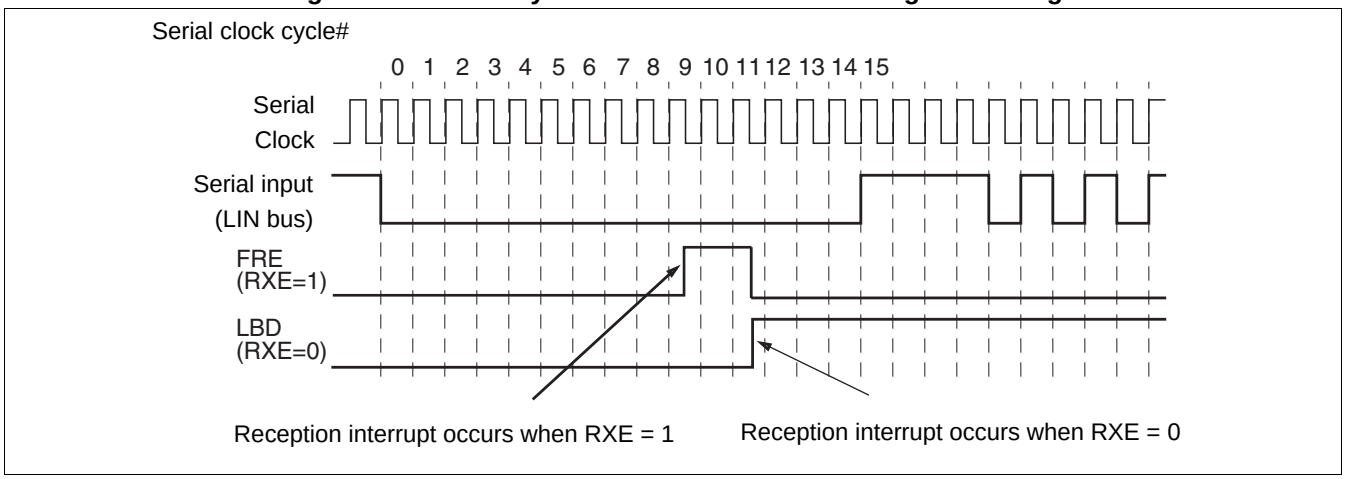
As shown in the LIN slave mode, when the BGR value newly calculated by sync field generates ±15% or more baud rate error, do not set the baud rate.

For the correspondence between other LIN-UARTs and ICUs, see Section "10.3.1 16-bit Input Capture" and "10.3.2 16-bit Free-run Timer".

- LIN Synch Break detection interrupt and flag

If a LIN synch break is detected in the slave mode, the LIN break detected flag (LBD) of the ESCR is set to "1". This causes an interrupt, if the LIN break interrupt is enabled (LBIE=1).

Figure 17.7-7 LIN Synch Break Detection and Flag Set Timing



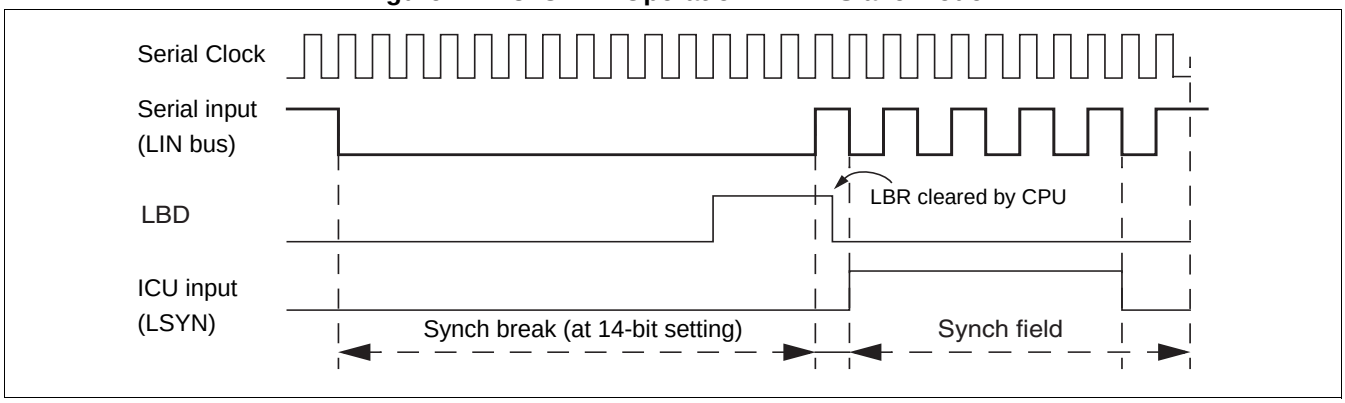
The figure above demonstrates the LIN synch break detection and flag set timing.

Note that the data framing error (FRE) flag bit of the SSR will cause a reception interrupt 2 bits times earlier than the LIN break interrupt ("8N1"), so that it is recommended to set RXE to "0" if a LIN break is expected.

LIN synch break detection is only supported in operation mode 3.

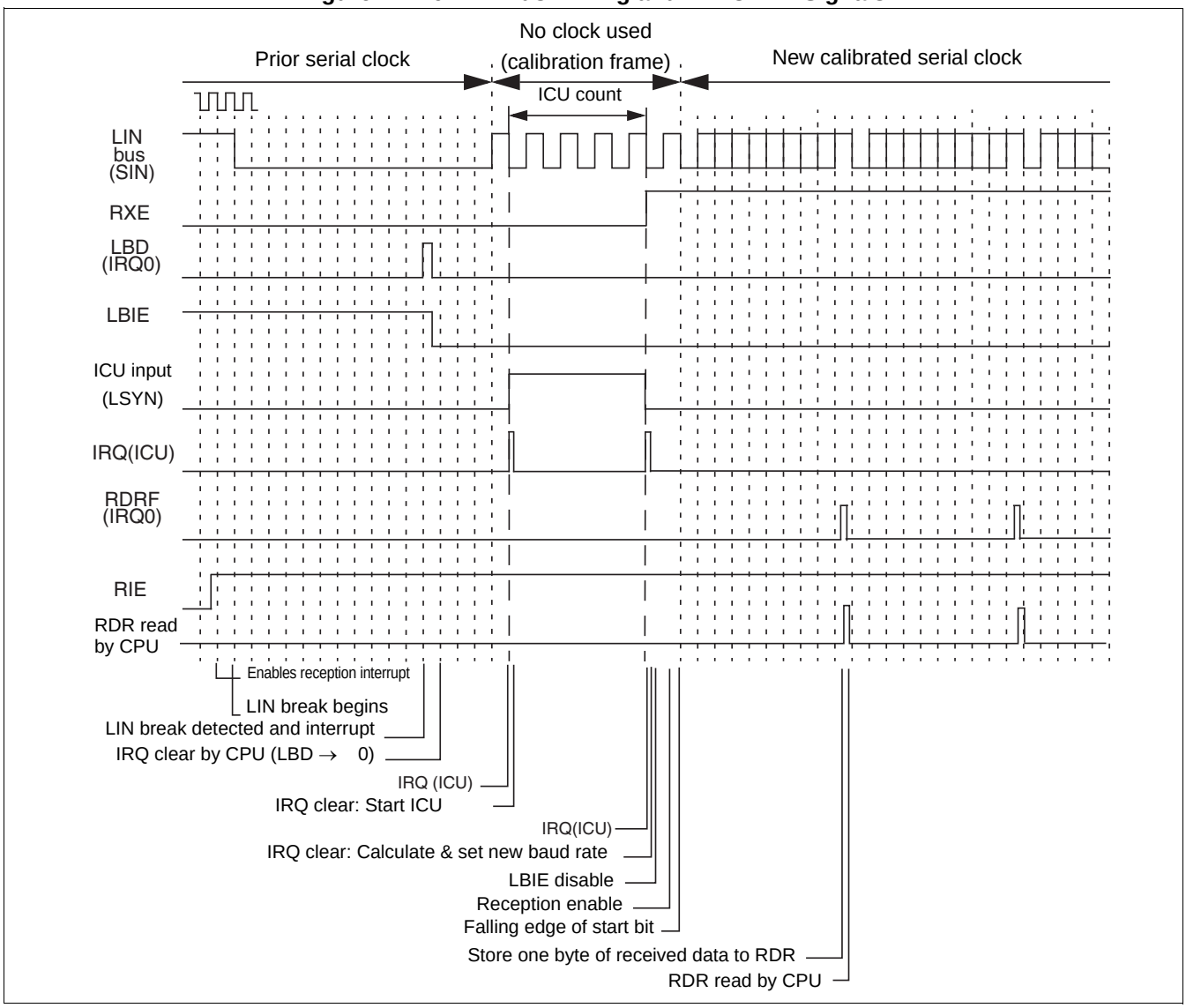
Figure 17.7-8 shows a typical start of a LIN message and the operation of the LIN-UART.

Figure 17.7-8 UART Operation in LIN Slave Mode



- LIN bus timing

Figure 17.7-9 LIN Bus Timing and LIN-UART Signals



17.7.4 Serial Pin Direct Access

LIN-UART allows the user to directly access to the transmission pin (SOTn) or the reception pin (SINn).

■ LIN-UART Pin Direct Access

The LIN-UART provides the ability for the software to access directly to serial input or output pin.

The status of the serial input pin (SINn) can be read by the serial input/output pin direct access bit (ESCR:SIOP).

If direct write to the serial output pin (SOTn) is allowed (ESCR: SOPE = 1) when the serial output is enabled (SMR: SOE = 1) after "0" or "1" is written to the SIOP bit of the ESCR register, the SOTn value can be set arbitrarily.

In LIN mode, this function can be used for reading the transmitted data or for error handling when the LIN bus line signal is physically incorrect.

Notes:

- Direct access is enabled only when the transmission is not ongoing (transmission shift register is empty).
- Write a value to the serial output pin direct access bit (ESCR:SIOP) before enabling the transmission (SMR: SOE = 1). This prevents the signal of the unexpected level from being outputted because the SIOP bit retains the previous value.
- During a Read-Modify-Write (RMW) instruction, the SIOP bit returns the actual value of the SOTn pin in the read cycle instead of the value of SINn during a normal read instruction.

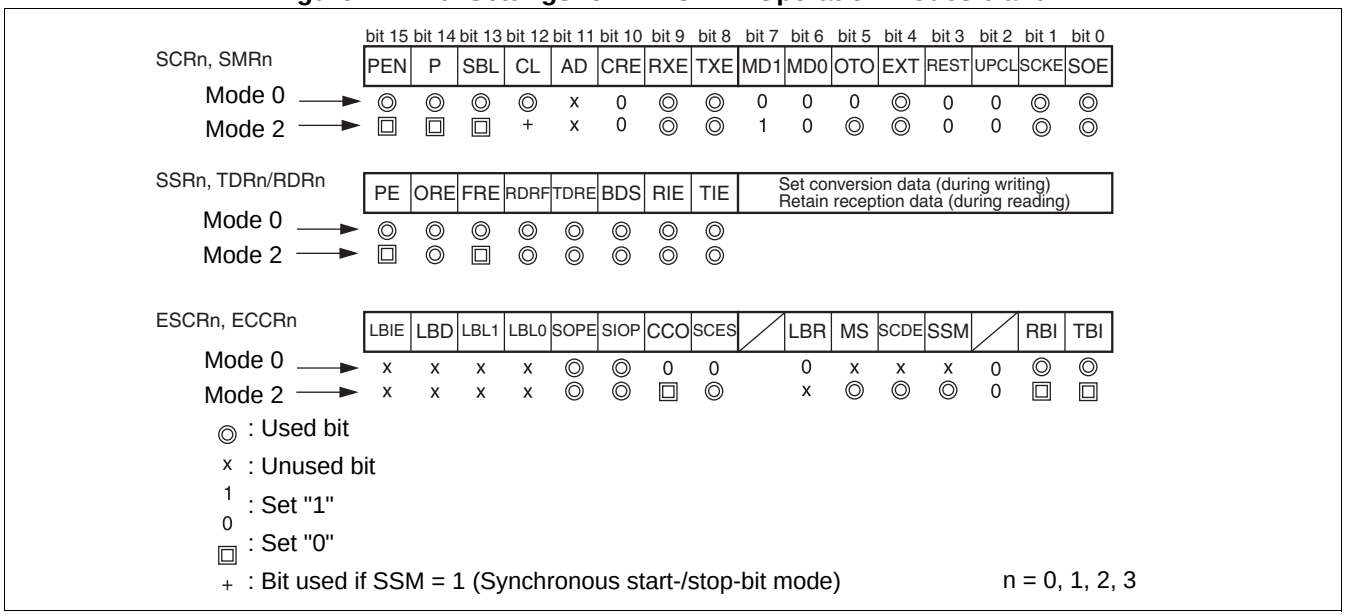
17.7.5 Bidirectional Communication Function (Normal Mode)

In operation mode 0 or 2, normal serial bidirectional communication is available. Select operation mode 0 for asynchronous communication and operation mode 2 for synchronous communication.

■ Bidirectional Communication Function

Figure 17.7-10 shows the settings to operate LIN-UART in normal mode (operation mode 0 or 2).

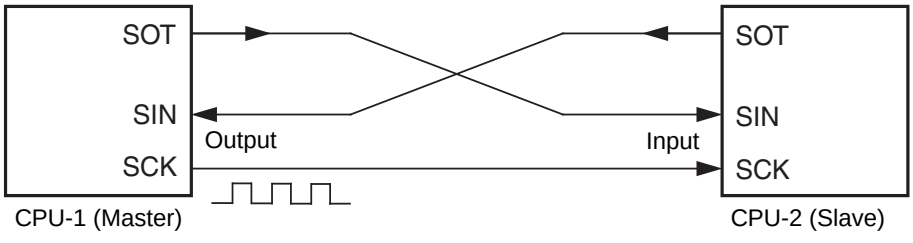
Figure 17.7-10 Settings for LIN-UART Operation Modes 0 and 2



● Inter-CPU connection

Two CPUs are interconnected as shown in Figure 17.7-11.

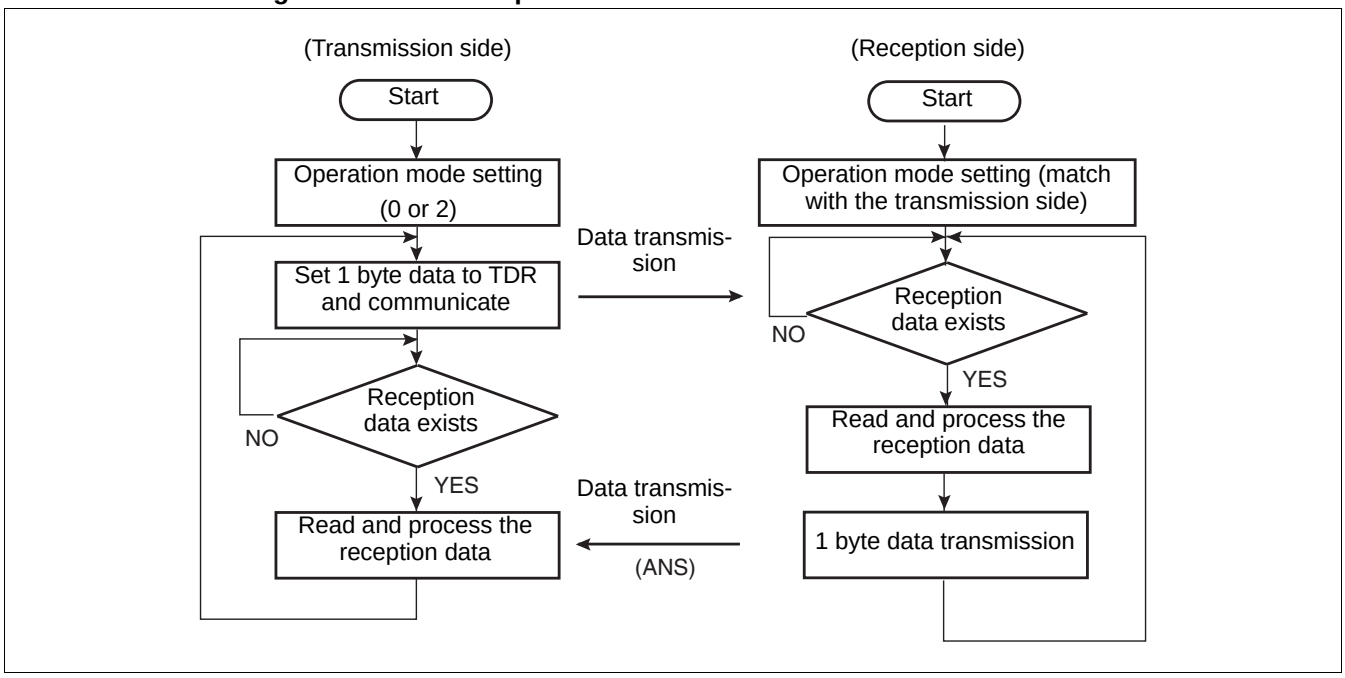
Figure 17.7-11 Connection Example of LIN-UART Mode 2 Bidirectional Communication



- Communication procedure

Communication starts at arbitrary timing from the transmission side when the transmission data is provided. When the transmission data is received at the reception side, ANS (per one byte in example) is returned periodically. Figure 17.7-12 shows an example of the bidirectional communication flowchart.

Figure 17.7-12 Example of Bidirectional Communication Flowchart



17.7.6 Master/Slave Type Communication Function (Multiprocessor Mode)

LIN-UART communication with multiple CPUs connected in master/slave mode is available for both master or slave systems in the operation mode 1.

■ Master/Slave Type Communication Function

Figure 17.7-13 shows the settings to operate LIN-UART in multiprocessor mode (operation mode 1).

Figure 17.7-13 Settings for LIN-UART Operation Mode 1

	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
SCRn, SMRn	PEN	P	SBL	CL	AD	CRE	RXE	TXE	MD1	MD0	OTO	EXT	REST	UPCL	SCKE	SOE
Mode 1 →	+	x	◎	◎	◎	0	◎	◎	0	1	0	◎	0	0	◎	◎
SSRn, TDRn/RDRn	PE	ORE	FRE	RDRF	TDRE	BDS	RIE	TIE	Set conversion data (during writing) Retain reception data (during reading)							
Mode 1 →	x	◎	◎	◎	◎	◎	◎	◎								
ESCRn, ECCRn	LBIE	LBD	LBL1	LBL0	SOPE	SIOP	CCO	SCES	/	LBR	MS	SCDE	SSM	/	RBI	TBI
Mode 1 →	x	x	x	x	◎	◎	0	0		x	x	x	x	0	◎	◎

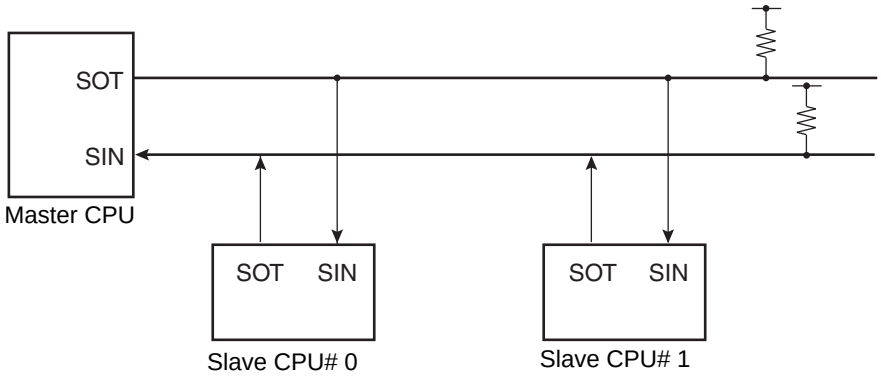
◎ : Used bit
x : Unused bit
1 : Set to "1"
0 : Set to "0"
+ : Bit automatically set correctly

n = 0, 1, 2, 3

● Inter-CPU connection

As shown in Figure 17.7-14, a communication system consists of one master CPU and multiple slave CPUs connected to two communication lines. LIN-UART can be used for the master or slave CPU.

Figure 17.7-14 Connection Example of LIN-UART Master/Slave Communication



● Function selection

Select the operation mode and data transfer mode for master-slave communication as shown in Table 17.7-3.

Table 17.7-3 Selection of the Master/Slave Communication Function

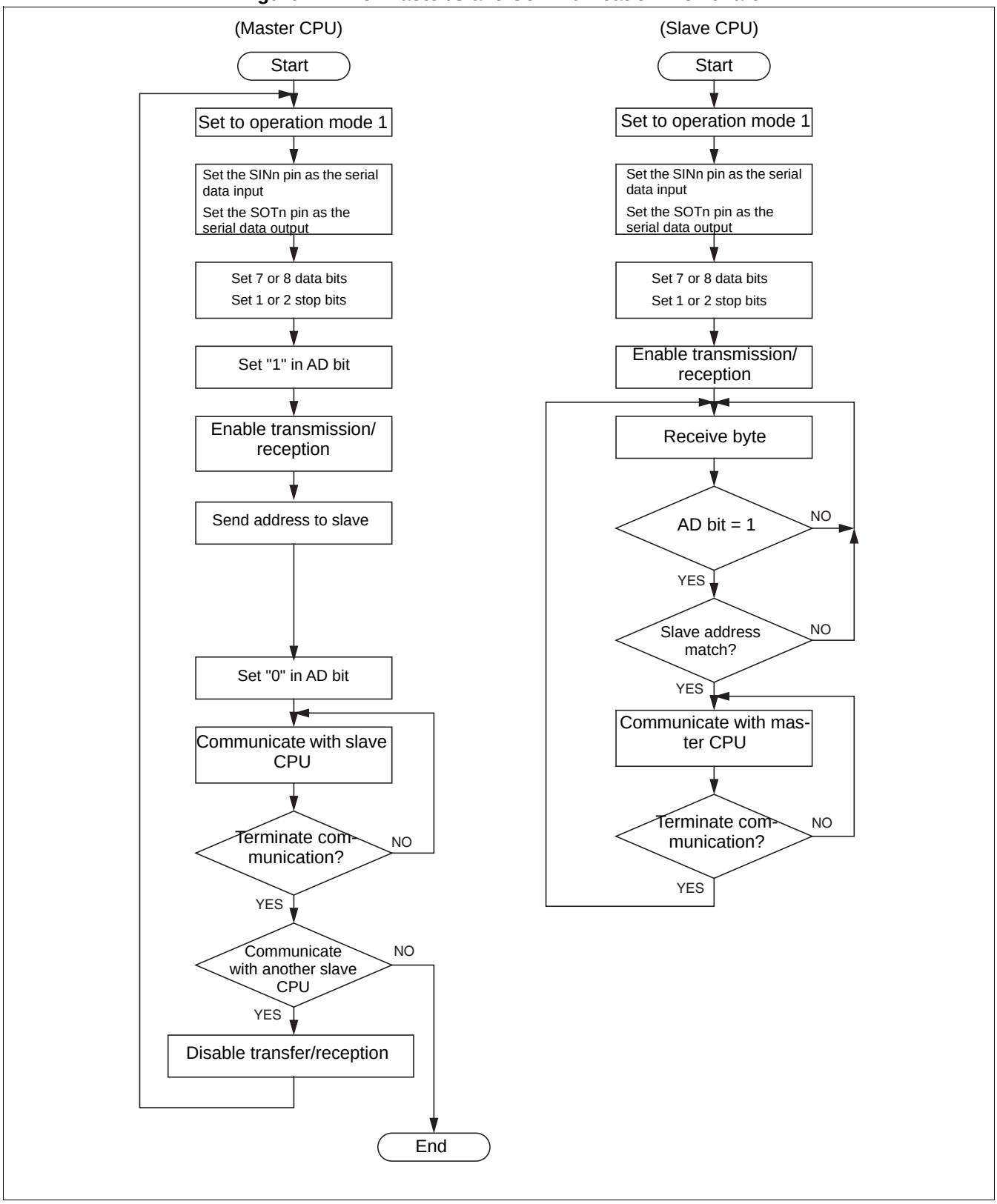
	Operation mode		Data	Parity	Synchronous method	Stop Bit	Bit direction
	Master CPU	**Slave CPU**					
Address transmis-sion/ reception	Mode 1 (AD bit Transmission/ Reception)	Mode 1 (AD bit Transmission/ Reception)	AD=1 + 7 or 8-bit address	None	Asynchronous	1 bit or 2 bits	LSB first or MSB-first
Data transmission/ reception			AD=0 + 7 or 8-bit data				

● Communication procedure

When the master CPU transmits address data, communication starts. The AD bit in the address data is set to "1", and the communication destination slave CPU is selected. Each slave CPU checks the address data using a program. When the address data indicates the address assigned to a slave CPU, the slave CPU communicates with the master CPU.

Figure 17.7-15 shows a flowchart of master-slave communication (multiprocessor mode)

Figure 17.7-15 Master/Slave Communication Flowchart



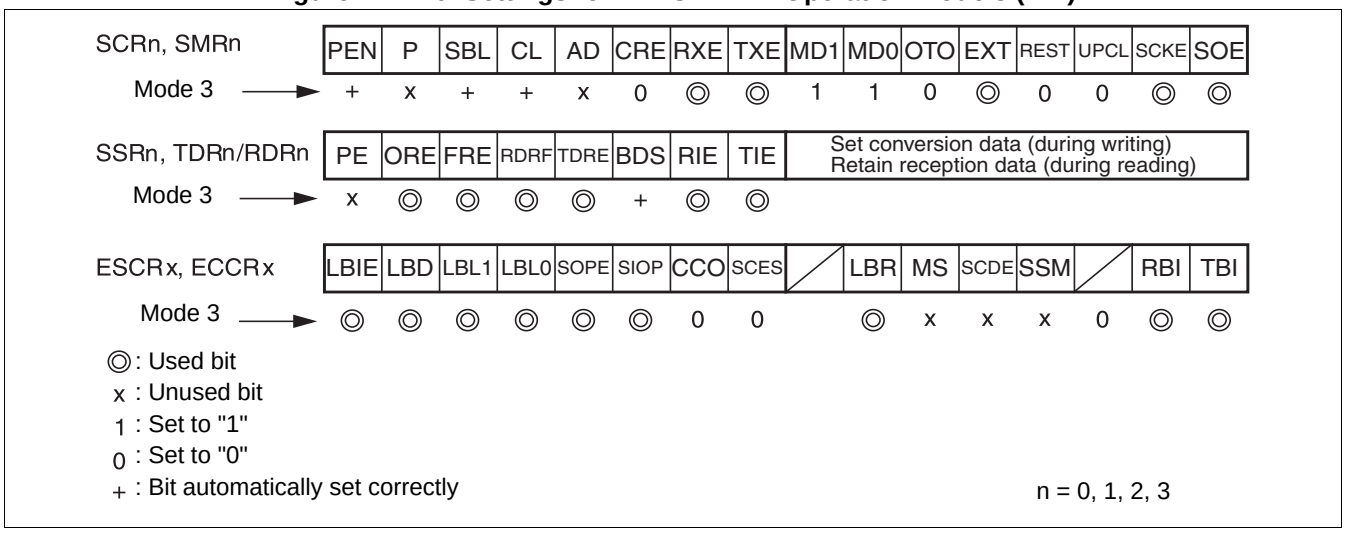
17.7.7 LIN Communication Function

LIN-UART communication with LIN devices is available for both LIN master or LIN slave systems.

■ LIN Master/Slave Type Communication Function

The settings shown in Figure 17.7-16 are required to operate LIN-UART in LIN communication mode (operation mode 3).

Figure 17.7-16 Settings for LIN-UART in Operation Mode 3 (LIN)

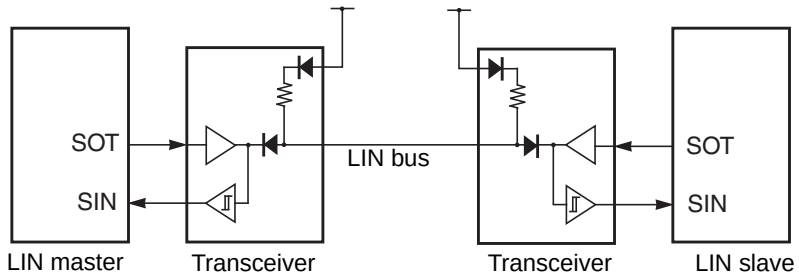


● LIN device connection

Figure 17.7-17 shows a communication system of one LIN-master device and a LIN-slave device.

LIN-UART can operate both as LIN-master or LIN-slave.

Figure 17.7-17 Connection Example of LIN-bus System

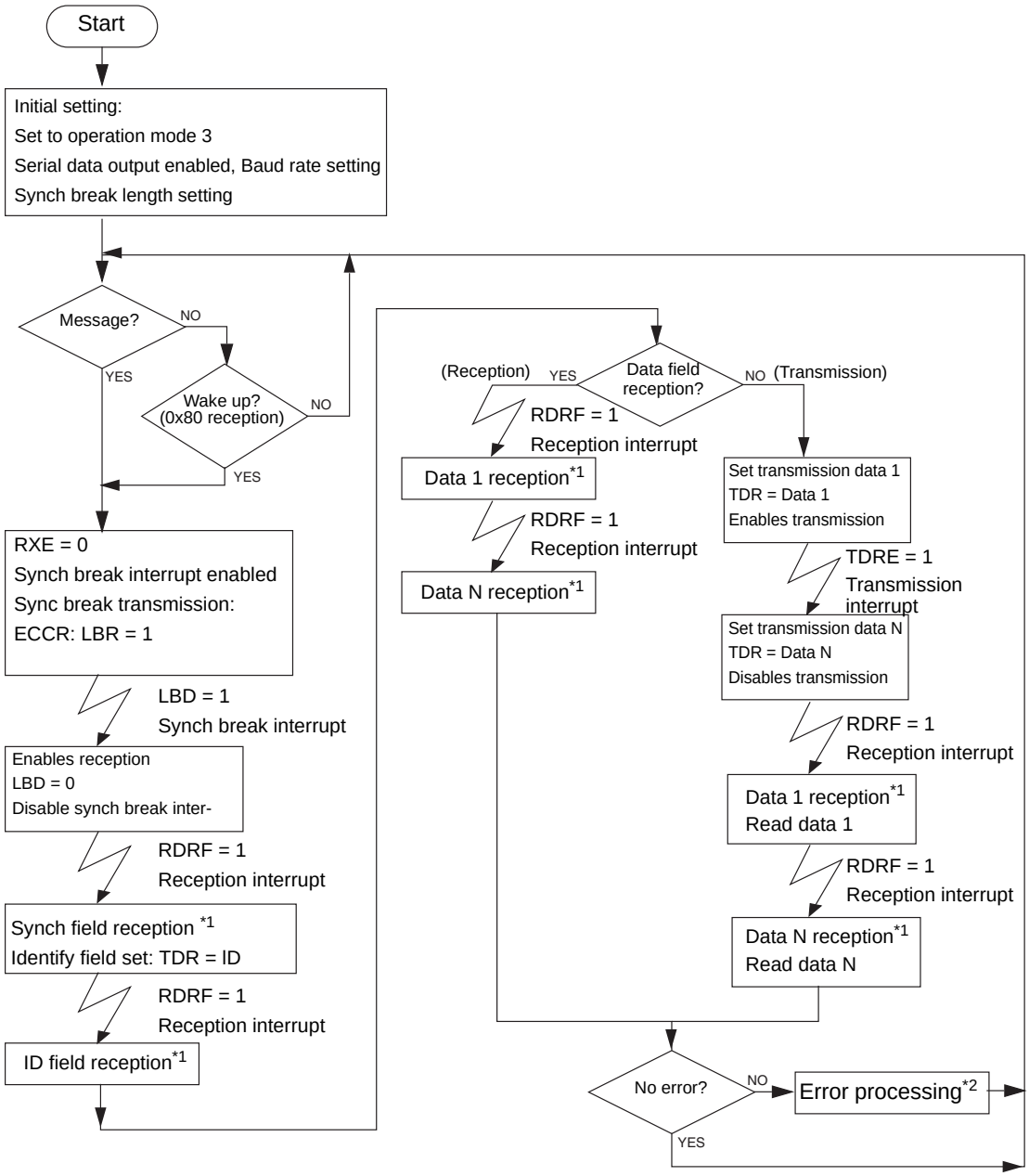


17.7.8 Sample Flowcharts for LIN-UART in LIN Communication (Operation Mode 3)

This section shows sample flowcharts for LIN-UART in LIN communication.

■ LIN Master Device

Figure 17.7-18 LIN Master Flowchart



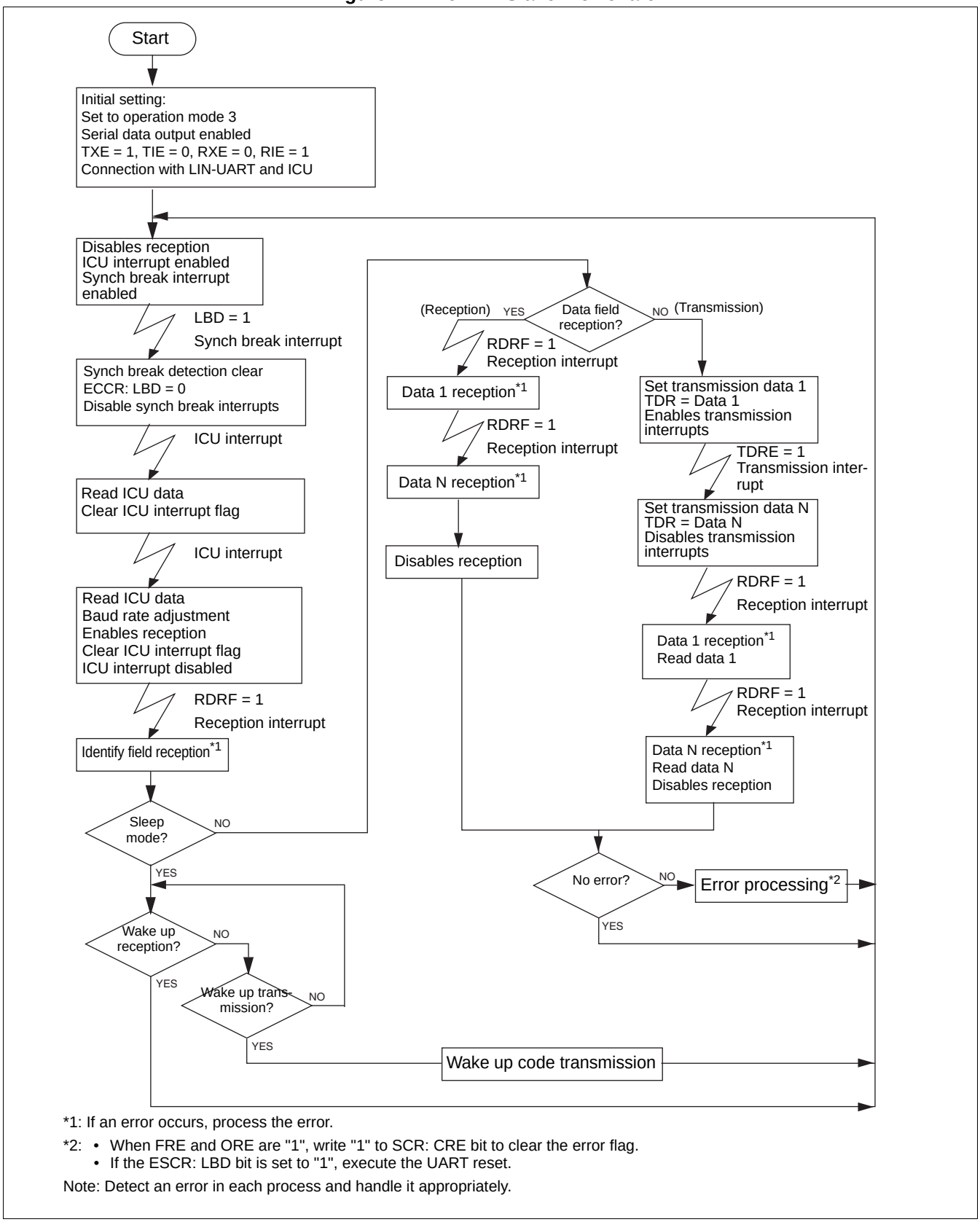
*1: If an error occurs, process the error.

*2:
- When FRE and ORE are "1", write "1" to SCR: CRE bit to clear the error flag.
- If the ESCR: LBD bit is set to "1", execute the UART reset.

Note: Detect an error in each process and handle it appropriately.

■ LIN Slave Device

Figure 17.7-19 LIN Slave Flowchart



*1: If an error occurs, process the error.

*2:
- When FRE and ORE are "1", write "1" to SCR: CRE bit to clear the error flag.
- If the ESCR: LBD bit is set to "1", execute the UART reset.

Note: Detect an error in each process and handle it appropriately.

17.8 Notes on Using LIN-UART

Notes on using LIN-UART are given below.

■ Notes on Using LIN-UART

● Enabling operations

The LIN-UART has the TXE (transmission) and RXE (reception) enable bit in the serial control register (SCR) for transmission and reception, respectively. Both, transmission and reception operations, must be enabled before the transfer starts because they have been disabled as the default value (initial value). Also, you can disable these operations to stop transfer as required.

● Communication mode setting

Set the communication mode while the LIN-UART is not operating. If the mode is set during transmission/reception, the transmitted/received data is not guaranteed.

● Transmission interrupt enabling timing

The default (initial value) of the transmission data empty flag bit (SSR: TDRE) is "1" (no transmission data and transmission data write enable state). A transmission interrupt request is generated as soon as the transmission interrupt request is enabled (SSR: TIE=1). Be sure to set the TIE flag to "1" after setting the transmission data to avoid an immediate interrupt.

● Changing operation settings

It is recommended to reset LIN-UART after changing operation settings, particularly if (for example) start-/stop-bits added to or removed from the data format.

The correct operation settings are not guaranteed even if you reset the LIN-UART (SMR:UPCL=1) at the same time as setting the LIN-UART serial mode register (SMR). Therefore, it is recommended to reset the LIN-UART (SMR:UPCL=1) once again, after setting the bit in LIN-UART serial mode register (SMR).

● Using LIN functions

The LIN features are available in mode 3, but using mode 3 sets the UART data format automatically to LIN format (8-bit length, no parity, one stop bit, and LSB first).

While the length of LIN break transmit bit is variable, the detection bit length is fixed to 11 bits.

● LIN slave settings

When starting LIN slave mode, be sure to set the baud rate before receiving the LIN synch break in order to ensure that at least 13 bits of the LIN synch break is detected.

- Software compatibility

Although this LIN-UART is similar to the old Cypress-UART, it is not software compatible to them. The programming models may be the same, but the structure of the registers differ. Furthermore, the setting of the baud rate is now determined by a reload value instead of selecting a predefined value.

- Bus idle function

The bus idle function cannot be used in synchronous mode 2.

- AD bit (serial control register (SCR): address/data type select bit)

Special care has to be taken when using the AD bit.

The AD bit is used to select the address/data for transmission in write operation, and to read the AD bit received last in read operation. Internally, the received and the transmitted value are stored in different registers.

With Read-Modify-Write (RMW) instructions, the transmitted value is read. This can lead to a wrong value in the AD bit, when one of the other bits in the same register is accessed by an instruction of this kind.

Therefore, this bit should be written by the last register access before transmission. Alternatively, using byte wise access and writing the correct values for all bits at once avoids this problem.

- Software reset of LIN-UART

When TXE bit of serial control register (SCR) is "0", execute LIN-UART software reset (SMR: UPCL = 1).

- Synch break detection

In mode 3 (LIN operation), the LBR bit in the ESCR register is set to "1" if the serial input is kept at "0" for more than 11-bit time (synch break detection). Then the LIN-UART waits for the following synch field to be received. If the LIN-UART is set into this state for other reasons than the synch break, it recognizes that synch break is inputted (LBD = 1) and waits for synch field.

In this case, execute the LIN-UART reset (SMR: UPCL = 1).

18. CAN Controller

This chapter describes an overview of the CAN controller and its functions.

18.1 Features of CAN Controller
18.2 Block Diagram of CAN Controller
18.3 Classification of CAN Controller Registers
18.4 CAN Controller Transmission
18.5 CAN Controller Reception
18.6 Using CAN Controller
18.7 Procedure of Transmission via Message Buffer (x)
18.8 Procedure of Reception Via Message Buffer (x)
18.9 Specifying the Multi-level Message Buffer Configuration
18.10 CAN WAKE UP Function
18.11 Precautions When Using CAN Controller
18.12 Sample Program of CAN

18.1 Features of CAN Controller

The CAN controller is a module that is integrated into a 16-bit microcomputer (F^2MC-16LX). CAN (Controller Area Network) is the standard protocol used for serial communication between controllers in automobiles, and is widely applied in the industrial fields.

■ Features of CAN Controller

The CAN controller has the following features:

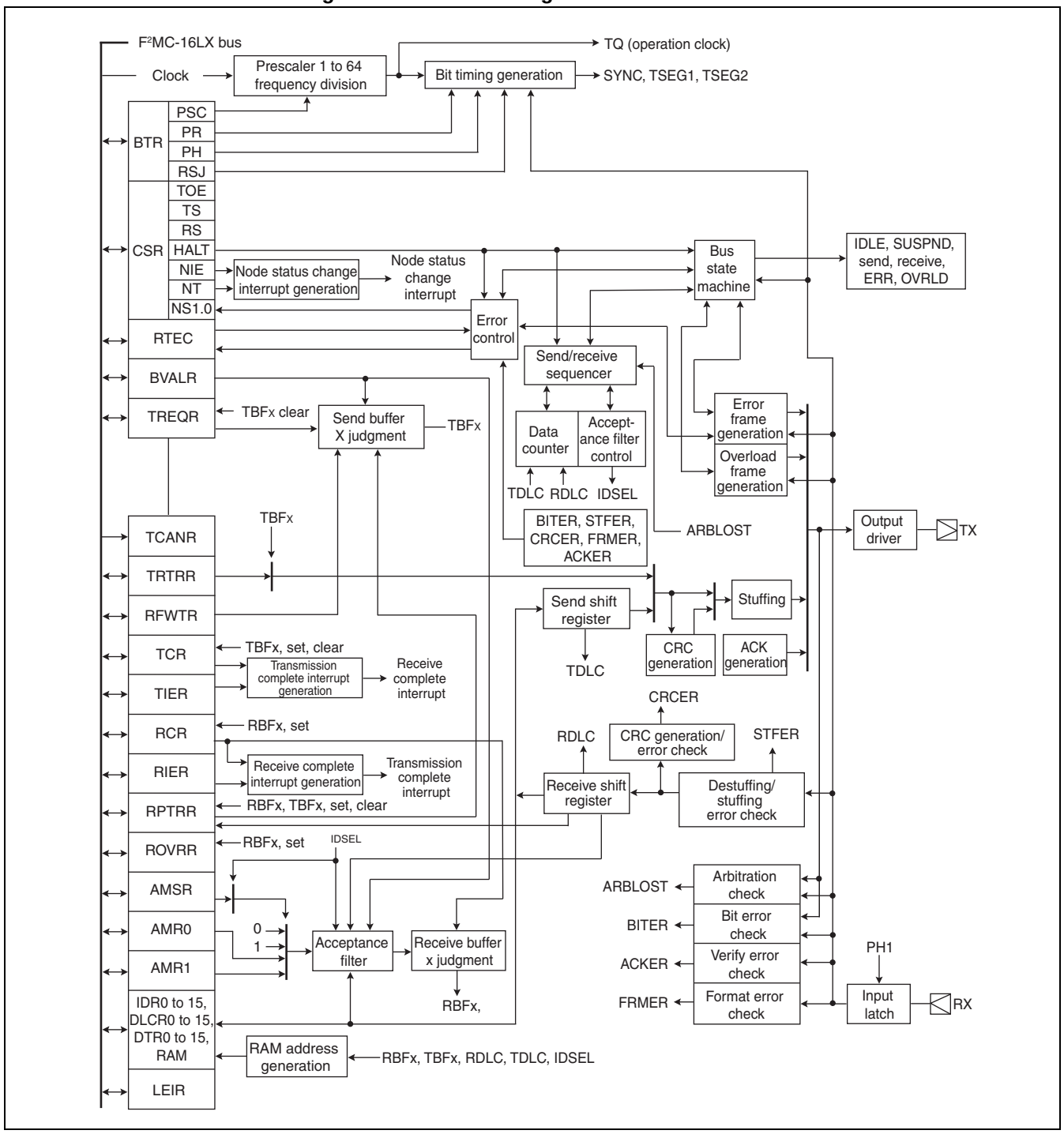
- Conforms to CAN specifications version 2.0, Parts A and B.
  Supports send/receive operations in the standard and extended frame formats.
- Supports data frame transmission based on remote frame reception.
- 16 send/receive message buffers
  29-bit ID and 8-byte data
  Multi-level message buffer structure
- Supports full-bit compare, full-bit mask, and partial-bit mask filtering.
  Provides two acceptance mask registers in either the standard or extended frame format.
- Bit speed is programmable between 10 kbps to 1 Mbps (Using 1 Mbps requires a minimum 8 MHz machine clock).
- CAN WAKE UP function

18.2 Block Diagram of CAN Controller

Figure 18.2-1 shows a block diagram of the CAN controller.

■ Block Diagram of CAN Controller

Figure 18.2-1 Block Diagram of CAN Controller



18.3 Classification of CAN Controller Registers

The CAN controller registers can be classified into the following 3 types:

- **General control register**
- **Message buffer control register**
- **Message buffer**

■ General Control Registers

The following 4 types of general control registers are provided:

- Control status register (CSR)
- Last event indication register (LEIR)
- Receive and transmit error counters (RTEC)
- Bit timing register (BTR)

Table 18.3-1 lists the general control registers.

Table 18.3-1 List of General Control Registers

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003C00_H$	$003D00_H$	$003E00_H$	$003F00_H$	Control status register	CSR	(R/W, R)	$00\text{---}000_B$ $0\text{----}0\text{-}1_B$
$003C01_H$	$003D01_H$	$003E01_H$	$003F01_H$				
$003C02_H$	$003D02_H$	$003E02_H$	$003F02_H$	Last event indication register	LEIR	(R/W)	-------- $000\text{-}0000_B$
$003C03_H$	$003D03_H$	$003E03_H$	$003F03_H$				
$003C04_H$	$003D04_H$	$003E04_H$	$003F04_H$	Receive and transmit error counters	RTEC	(R)	00000000_B 00000000_B
$003C05_H$	$003D05_H$	$003E05_H$	$003F05_H$				
$003C06_H$	$003D06_H$	$003E06_H$	$003F06_H$	Bit timing register	BTR	(R/W)	-1111111_B 11111111_B
$003C07_H$	$003D07_H$	$003E07_H$	$003F07_H$				

■ Message Buffer Control Registers

The following 14 types of message buffer control registers are provided:

- Message buffer valid register (BVALR)
- IDE register (IDER)
- Transmission request register (TREQR)
- Transmission RTR register (TRTRR)
- Remote frame receive wait register (RFWTR)
- Transmission cancel register (TCANR)
- Transmission complete register (TCR)
- Transmission interrupt enable register (TIER)
- Receive complete register (RCR)
- Remote request receive register (RRTRR)
- Receive overrun register (ROVRR)
- Receive interrupt enable register (RIER)
- Acceptance mask selection register (AMSR)
- Acceptance mask registers 0 and 1 (AMR0, AMR1)

Table 18.3-2 lists message buffer control registers.

Table 18.3-2 List of Message Buffer Control Registers (Sheet 1 of 2)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
000040_H	000070_H	$0039C0_H$	$0039D0_H$	Message buffer valid register	BVALR	(R/W)	00000000_B 00000000_B
000041_H	000071_H	$0039C1_H$	$0039D1_H$				
000042_H	000072_H	$0039C2_H$	$0039D2_H$	Transmission request register	TREQR	(R/W)	00000000_B 00000000_B
000043_H	000073_H	$0039C3_H$	$0039D3_H$				
000044_H	000074_H	$0039C4_H$	$0039D4_H$	Transmission cancel register	TCANR	(W)	00000000_B 00000000_B
000045_H	000075_H	$0039C5_H$	$0039D5_H$				
000046_H	000076_H	$0039C6_H$	$0039D6_H$	Transmission complete register	TCR	(R/W)	00000000_B 00000000_B
000047_H	000077_H	$0039C7_H$	$0039D7_H$				
000048_H	000078_H	$0039C8_H$	$0039D8_H$	Receive complete register	RCR	(R/W)	00000000_B 00000000_B
000049_H	000079_H	$0039C9_H$	$0039D9_H$				
$00004A_H$	$00007A_H$	$0039CA_H$	$0039DA_H$	Remote request receive register	RRTRR	(R/W)	00000000_B 00000000_B
$00004B_H$	$00007B_H$	$0039CB_H$	$0039DB_H$				
$00004C_H$	$00007C_H$	$0039CC_H$	$0039DC_H$	Receive overrun register	ROVRR	(R/W)	00000000_B 00000000_B
$00004D_H$	$00007D_H$	$0039CD_H$	$0039DD_H$				
$00004E_H$	$00007E_H$	$0039CE_H$	$0039DE_H$	Receive interrupt enable register	RIER	(R/W)	00000000_B 00000000_B
$00004F_H$	$00007F_H$	$0039CF_H$	$0039DF_H$				

Table 18.3-2 List of Message Buffer Control Registers (Sheet 2 of 2)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003C08_H$	$003D08_H$	$003E08_H$	$003F08_H$	IDE register	IDER	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003C09_H$	$003D09_H$	$003E09_H$	$003F09_H$				
$003C0A_H$	$003D0A_H$	$003E0A_H$	$003F0A_H$	Transmission RTR register	TRTRR	(R/W)	00000000_B 00000000_B
$003C0B_H$	$003D0B_H$	$003E0B_H$	$003F0B_H$				
$003C0C_H$	$003D0C_H$	$003E0C_H$	$003F0C_H$	Remote frame receive wait register	RFWTR	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003C0D_H$	$003D0D_H$	$003E0D_H$	$003F0D_H$				
$003C0E_H$	$003D0E_H$	$003E0E_H$	$003F0E_H$	Transmission interrupt enable register	TIER	(R/W)	00000000_B 00000000_B
$003C0F_H$	$003D0F_H$	$003E0F_H$	$003F0F_H$				
$003C10_H$	$003D10_H$	$003E10_H$	$003F10_H$	Acceptance mask selection register	AMSR	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003C11_H$	$003D11_H$	$003E11_H$	$003F11_H$				
$003C12_H$	$003D12_H$	$003E12_H$	$003F12_H$				XXXXXXXX_B XXXXXXXX_B
$003C13_H$	$003D13_H$	$003E13_H$	$003F13_H$				
$003C14_H$	$003D14_H$	$003E14_H$	$003F14_H$	Acceptance mask register 0	AMR0	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003C15_H$	$003D15_H$	$003E15_H$	$003F15_H$				
$003C16_H$	$003D16_H$	$003E16_H$	$003F16_H$				$XXXXX---_B$ XXXXXXXX_B
$003C17_H$	$003D17_H$	$003E17_H$	$003F17_H$				
$003C18_H$	$003D18_H$	$003E18_H$	$003F18_H$	Acceptance mask register 1	AMR1	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003C19_H$	$003D19_H$	$003E19_H$	$003F19_H$				
$003C1A_H$	$003D1A_H$	$003E1A_H$	$003F1A_H$				$XXXXX---_B$ XXXXXXXX_B
$003C1B_H$	$003D1B_H$	$003E1B_H$	$003F1B_H$				

■ Message Buffers

The following 3 types of message buffers are provided:

- ID register x (x = 0 to 15) (IDRx)
- DLC register x (x = 0 to 15) (DLCRx)
- Data register x (x = 0 to 15) (DTRx)

Table 18.3-3, Table 18.3-4, and Table 18.3-5 list message buffers: ID register, DLC register, and data register respectively.

Table 18.3-3 List of Message Buffers (ID Register) (Sheet 1 of 3)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A00_H$ to $003A1F_H$	$003B00_H$ to $003B1F_H$	003700_H to $00371F_H$	003800_H to $00381F_H$	General-purpose RAM	--	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003A20_H$	$003B20_H$	003720_H	003820_H	ID register 0	IDR0	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A21_H$	$003B21_H$	003721_H	003821_H				
$003A22_H$	$003B22_H$	003722_H	003822_H				$XXXXX---_B$ XXXXXXXX_B
$003A23_H$	$003B23_H$	003723_H	003823_H				
$003A24_H$	$003B24_H$	003724_H	003824_H	ID register 1	IDR1	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A25_H$	$003B25_H$	003725_H	003825_H				
$003A26_H$	$003B26_H$	003726_H	003826_H				$XXXXX---_B$ XXXXXXXX_B
$003A27_H$	$003B27_H$	003727_H	003827_H				
$003A28_H$	$003B28_H$	003728_H	003828_H	ID register 2	IDR2	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A29_H$	$003B29_H$	003729_H	003829_H				
$003A2A_H$	$003B2A_H$	$00372A_H$	$00382A_H$				$XXXXX---_B$ XXXXXXXX_B
$003A2B_H$	$003B2B_H$	$00372B_H$	$00382B_H$				
$003A2C_H$	$003B2C_H$	$00372C_H$	$00382C_H$	ID register 3	IDR3	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A2D_H$	$003B2D_H$	$00372D_H$	$00382D_H$				
$003A2E_H$	$003B2E_H$	$00372E_H$	$00382E_H$				$XXXXX---_B$ XXXXXXXX_B
$003A2F_H$	$003B2F_H$	$00372F_H$	$00382F_H$				
$003A30_H$	$003B30_H$	003730_H	003830_H	ID register 4	IDR4	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A31_H$	$003B31_H$	003731_H	003831_H				
$003A32_H$	$003B32_H$	003732_H	003832_H				$XXXXX---_B$ XXXXXXXX_B
$003A33_H$	$003B33_H$	003733_H	003833_H				
$003A34_H$	$003B34_H$	003734_H	003834_H	ID register 5	IDR5	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A35_H$	$003B35_H$	003735_H	003835_H				
$003A36_H$	$003B36_H$	003736_H	003836_H				$XXXXX---_B$ XXXXXXXX_B
$003A37_H$	$003B37_H$	003737_H	003837_H				

Table 18.3-3 List of Message Buffers (ID Register) (Sheet 2 of 3)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A38_H$	$003B38_H$	003738_H	003838_H	ID register 6	IDR6	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A39_H$	$003B39_H$	003739_H	003839_H				
$003A3A_H$	$003B3A_H$	$00373A_H$	$00383A_H$				$XXXXX---_B$ XXXXXXXX_B
$003A3B_H$	$003B3B_H$	$00373B_H$	$00383B_H$				
$003A3C_H$	$003B3C_H$	$00373C_H$	$00383C_H$	ID register 7	IDR7	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A3D_H$	$003B3D_H$	$00373D_H$	$00383D_H$				
$003A3E_H$	$003B3E_H$	$00373E_H$	$00383E_H$				$XXXXX---_B$ XXXXXXXX_B
$003A3F_H$	$003B3F_H$	$00373F_H$	$00383F_H$				
$003A40_H$	$003B40_H$	003740_H	003840_H	ID register 8	IDR8	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A41_H$	$003B41_H$	003741_H	003841_H				
$003A42_H$	$003B42_H$	003742_H	003842_H				$XXXXX---_B$ XXXXXXXX_B
$003A43_H$	$003B43_H$	003743_H	003843_H				
$003A44_H$	$003B44_H$	003744_H	003844_H	ID register 9	IDR9	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A45_H$	$003B45_H$	003745_H	003845_H				
$003A46_H$	$003B46_H$	003746_H	003846_H				$XXXXX---_B$ XXXXXXXX_B
$003A47_H$	$003B47_H$	003747_H	003847_H				
$003A48_H$	$003B48_H$	003748_H	003848_H	ID register 10	IDR10	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A49_H$	$003B49_H$	003749_H	003849_H				
$003A4A_H$	$003B4A_H$	$00374A_H$	$00384A_H$				$XXXXX---_B$ XXXXXXXX_B
$003A4B_H$	$003B4B_H$	$00374B_H$	$00384B_H$				
$003A4C_H$	$003B4C_H$	$00374C_H$	$00384C_H$	ID register 11	IDR11	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A4D_H$	$003B4D_H$	$00374D_H$	$00384D_H$				
$003A4E_H$	$003B4E_H$	$00374E_H$	$00384E_H$				$XXXXX---_B$ XXXXXXXX_B
$003A4F_H$	$003B4F_H$	$00374F_H$	$00384F_H$				
$003A50_H$	$003B50_H$	003750_H	003850_H	ID register 12	IDR12	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A51_H$	$003B51_H$	003751_H	003851_H				
$003A52_H$	$003B52_H$	003752_H	003852_H				$XXXXX---_B$ XXXXXXXX_B
$003A53_H$	$003B53_H$	003753_H	003853_H				
$003A54_H$	$003B54_H$	003754_H	003854_H	ID register 13	IDR13	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A55_H$	$003B55_H$	003755_H	003855_H				
$003A56_H$	$003B56_H$	003756_H	003856_H				$XXXXX---_B$ XXXXXXXX_B
$003A57_H$	$003B57_H$	003757_H	003857_H				

Table 18.3-3 List of Message Buffers (ID Register) (Sheet 3 of 3)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A58_H$	$003B58_H$	003758_H	003858_H	ID register 14	IDR14	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A59_H$	$003B59_H$	003759_H	003859_H				
$003A5A_H$	$003B5A_H$	$00375A_H$	$00385A_H$				$XXXXX---_B$ XXXXXXXX_B
$003A5B_H$	$003B5B_H$	$00375B_H$	$00385B_H$				
$003A5C_H$	$003B5C_H$	$00375C_H$	$00385C_H$	ID register 15	IDR15	(R/W)	XXXXXXXX_B XXXXXXXX_B
$003A5D_H$	$003B5D_H$	$00375D_H$	$00385D_H$				
$003A5E_H$	$003B5E_H$	$00375E_H$	$00385E_H$				$XXXXX---_B$ XXXXXXXX_B
$003A5F_H$	$003B5F_H$	$00375F_H$	$00385F_H$				

Table 18.3-4 List of Message Buffers (DLC Register) (Sheet 1 of 2)

Address				Register	Abbreviation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A60_H$	$003B60_H$	003760_H	003860_H	DLC register 0	DLCR0	(R/W)	---- XXXX_B
$003A61_H$	$003B61_H$	003761_H	003861_H				
$003A62_H$	$003B62_H$	003762_H	003862_H	DLC register 1	DLCR1	(R/W)	---- XXXX_B
$003A63_H$	$003B63_H$	003763_H	003863_H				
$003A64_H$	$003B64_H$	003764_H	003864_H	DLC register 2	DLCR2	(R/W)	---- XXXX_B
$003A65_H$	$003B65_H$	003765_H	003865_H				
$003A66_H$	$003B66_H$	003766_H	003866_H	DLC register 3	DLCR3	(R/W)	---- XXXX_B
$003A67_H$	$003B67_H$	003767_H	003867_H				
$003A68_H$	$003B68_H$	003768_H	003868_H	DLC register 4	DLCR4	(R/W)	---- XXXX_B
$003A69_H$	$003B69_H$	003769_H	003869_H				
$003A6A_H$	$003B6A_H$	$00376A_H$	$00386A_H$	DLC register 5	DLCR5	(R/W)	---- XXXX_B
$003A6B_H$	$003B6B_H$	$00376B_H$	$00386B_H$				
$003A6C_H$	$003B6C_H$	$00376C_H$	$00386C_H$	DLC register 6	DLCR6	(R/W)	---- XXXX_B
$003A6D_H$	$003B6D_H$	$00376D_H$	$00386D_H$				
$003A6E_H$	$003B6E_H$	$00376E_H$	$00386E_H$	DLC register 7	DLCR7	(R/W)	---- XXXX_B
$003A6F_H$	$003B6F_H$	$00376F_H$	$00386F_H$				
$003A70_H$	$003B70_H$	003770_H	003870_H	DLC register 8	DLCR8	(R/W)	---- XXXX_B
$003A71_H$	$003B71_H$	003771_H	003871_H				
$003A72_H$	$003B72_H$	003772_H	003872_H	DLC register 9	DLCR9	(R/W)	---- XXXX_B
$003A73_H$	$003B73_H$	003773_H	003873_H				
$003A74_H$	$003B74_H$	003774_H	003874_H	DLC register 10	DLCR10	(R/W)	---- XXXX_B
$003A75_H$	$003B75_H$	003775_H	003875_H				
$003A76_H$	$003B76_H$	003776_H	003876_H	DLC register 11	DLCR11	(R/W)	---- XXXX_B
$003A77_H$	$003B77_H$	003777_H	003877_H				
$003A78_H$	$003B78_H$	003778_H	003878_H	DLC register 12	DLCR12	(R/W)	---- XXXX_B
$003A79_H$	$003B79_H$	003779_H	003879_H				
$003A7A_H$	$003B7A_H$	$00377A_H$	$00387A_H$	DLC register 13	DLCR13	(R/W)	---- XXXX_B
$003A7B_H$	$003B7B_H$	$00377B_H$	$00387B_H$				
$003A7C_H$	$003B7C_H$	$00377C_H$	$00387C_H$	DLC register 14	DLCR14	(R/W)	---- XXXX_B
$003A7D_H$	$003B7D_H$	$00377D_H$	$00387D_H$				

Table 18.3-4 List of Message Buffers (DLC Register) (Sheet 2 of 2)

Address				Register	Abbreviation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A7E_H$	$003B7E_H$	$00377E_H$	$00387E_H$	DLC register 15	DLCR15	(R/W)	---- XXXX_B
$003A7F_H$	$003B7F_H$	$00377F_H$	$00387F_H$				

Table 18.3-5 List of Message Buffers (Data Register)

Address				Register	Abbre-viation	Access	Initial Value
CAN0	CAN1	CAN2	CAN3				
$003A80_H$ to $003A87_H$	$003B80_H$ to $003B87_H$	003780_H to 003787_H	003880_H to 003887_H	Data register 0 (8 bytes)	DTR0	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003A88_H$ to $003A8F_H$	$003B88_H$ to $003B8F_H$	003788_H to $00378F_H$	003888_H to $00388F_H$	Data register 1 (8 bytes)	DTR1	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003A90_H$ to $003A97_H$	$003B90_H$ to $003B97_H$	003790_H to 003797_H	003890_H to 003897_H	Data register 2 (8 bytes)	DTR2	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003A98_H$ to $003A9F_H$	$003B98_H$ to $003B9F_H$	003798_H to $00379F_H$	003898_H to $00389F_H$	Data register 3 (8 bytes)	DTR3	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AA0_H$ to $003AA7_H$	$003BA0_H$ to $003BA7_H$	$0037A0_H$ to $0037A7_H$	$0038A0_H$ to $0038A7_H$	Data register 4 (8 bytes)	DTR4	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AA8_H$ to $003AAF_H$	$003BA8_H$ to $003BAF_H$	$0037A8_H$ to $0037AF_H$	$0038A8_H$ to $0038AF_H$	Data register 5 (8 bytes)	DTR5	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AB0_H$ to $003AB7_H$	$003BB0_H$ to $003BB7_H$	$0037B0_H$ to $0037B7_H$	$0038B0_H$ to $0038B7_H$	Data register 6 (8 bytes)	DTR6	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AB8_H$ to $003ABF_H$	$003BB8_H$ to $003BBF_H$	$0037B8_H$ to $0037BF_H$	$0038B8_H$ to $0038BF_H$	Data register 7 (8 bytes)	DTR7	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AC0_H$ to $003AC7_H$	$003BC0_H$ to $003BC7_H$	$0037C0_H$ to $0037C7_H$	$0038C0_H$ to $0038C7_H$	Data register 8 (8 bytes)	DTR8	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AC8_H$ to $003ACF_H$	$003BC8_H$ to $003BCF_H$	$0037C8_H$ to $0037CF_H$	$0038C8_H$ to $0038CF_H$	Data register 9 (8 bytes)	DTR9	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AD0_H$ to $003AD7_H$	$003BD0_H$ to $003BD7_H$	$0037D0_H$ to $0037D7_H$	$0038D0_H$ to $0038D7_H$	Data register 10 (8 bytes)	DTR10	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AD8_H$ to $003ADF_H$	$003BD8_H$ to $003BDF_H$	$0037D8_H$ to $0037DF_H$	$0038D8_H$ to $0038DF_H$	Data register 11 (8 bytes)	DTR11	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AE0_H$ to $003AE7_H$	$003BE0_H$ to $003BE7_H$	$0037E0_H$ to $0037E7_H$	$0038E0_H$ to $0038E7_H$	Data register 12 (8 bytes)	DTR12	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AE8_H$ to $003AEF_H$	$003BE8_H$ to $003BEF_H$	$0037E8_H$ to $0037EF_H$	$0038E8_H$ to $0038EF_H$	Data register 13 (8 bytes)	DTR13	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AF0_H$ to $003AF7_H$	$003BF0_H$ to $003BF7_H$	$0037F0_H$ to $0037F7_H$	$0038F0_H$ to $0038F7_H$	Data register 14 (8 bytes)	DTR14	(R/W)	XXXXXXXX_B to XXXXXXXX_B
$003AF8_H$ to $003AFF_H$	$003BF8_H$ to $003BFF_H$	$0037F8_H$ to $0037FF_H$	$0038F8_H$ to $0038FF_H$	Data register 15 (8 bytes)	DTR15	(R/W)	XXXXXXXX_B to XXXXXXXX_B

18.3.1 Control Status Register (CSR)

Bit operation instructions (read-modify-write instructions) cannot be used for the control status register (CSR).

■ Bit Configuration of Control Status Register (CSR)

Figure 18.3-1 shows the bit configuration of the control status register (CSR).

Figure 18.3-1 Bit Configuration of Control Status Register (CSR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C01_H$ (CAN0) $003D01_H$ (CAN1) $003E01_H$ (CAN2) $003F01_H$ (CAN3)	TS	RS	–	–	–	NT	NS1	NS0	
	R	R	–	–	–	R/W	R	R	<- Read/Write
	0	0	–	–	–	0	0	1	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C00_H$ (CAN0) $003D00_H$ (CAN1) $003E00_H$ (CAN2) $003F00_H$ (CAN3)	TOE	–	–	–	–	NIE	–	HALT	
	R/W	–	–	–	–	R/W	–	R/W	<- Read/Write
	0	–	–	–	–	0	–	1	<- Initial value

[bit15] TS: Transmission status bit

This bit indicates whether a message is being sent.

- 0: No message being sent.
- 1: Message being sent.

This bit is also "0" when an error frame and an overload frame are being sent.

[bit14] RS: Receive status bit

This bit indicates whether a message is being received.

- 0: No message being received.
- 1: Message being received.

This bit is "1" while a message exists on the bus. Therefore, this bit is also "1" when a message is being sent. This bit does not necessarily indicate whether a receive message passed through an acceptance filter. Consequently, when this bit is "0", it indicates that the bus operation is stopped (HALT = 0), the bus is in intermission/bus idle state, or an error/overload frame exists on the bus.

[bit10] NT: Node status transition flag

This bit becomes "1" when the node status changes incrementally, and also when it changes from bus-off to error active.

In other words, the NT bit is set to "1" when the node status changes as follows. The values in parentheses show the values of the NS1 and NS0 bits.

- From error active (00_B) to warning (01_B)
- From warning (01_B) to error passive (10_B)
- From error passive (10_B) to bus-off (11_B)
- From bus-off (11_B) to error active (00_B)

When the node status transition interrupt enable bit (NIE) is "1", an interrupt is generated.

Writing "0" to the NT bit sets the bit to "0". Writing "1" to this bit is ignored. "1" is read from this bit when a read-modify-write (RMW) instruction is executed.

[bit9, bit8] NS1, NS0: Bit1 and bit0 of node status

The NS1 and NS0 bits indicate the current node status.

Table 18.3-6 shows this relationship.

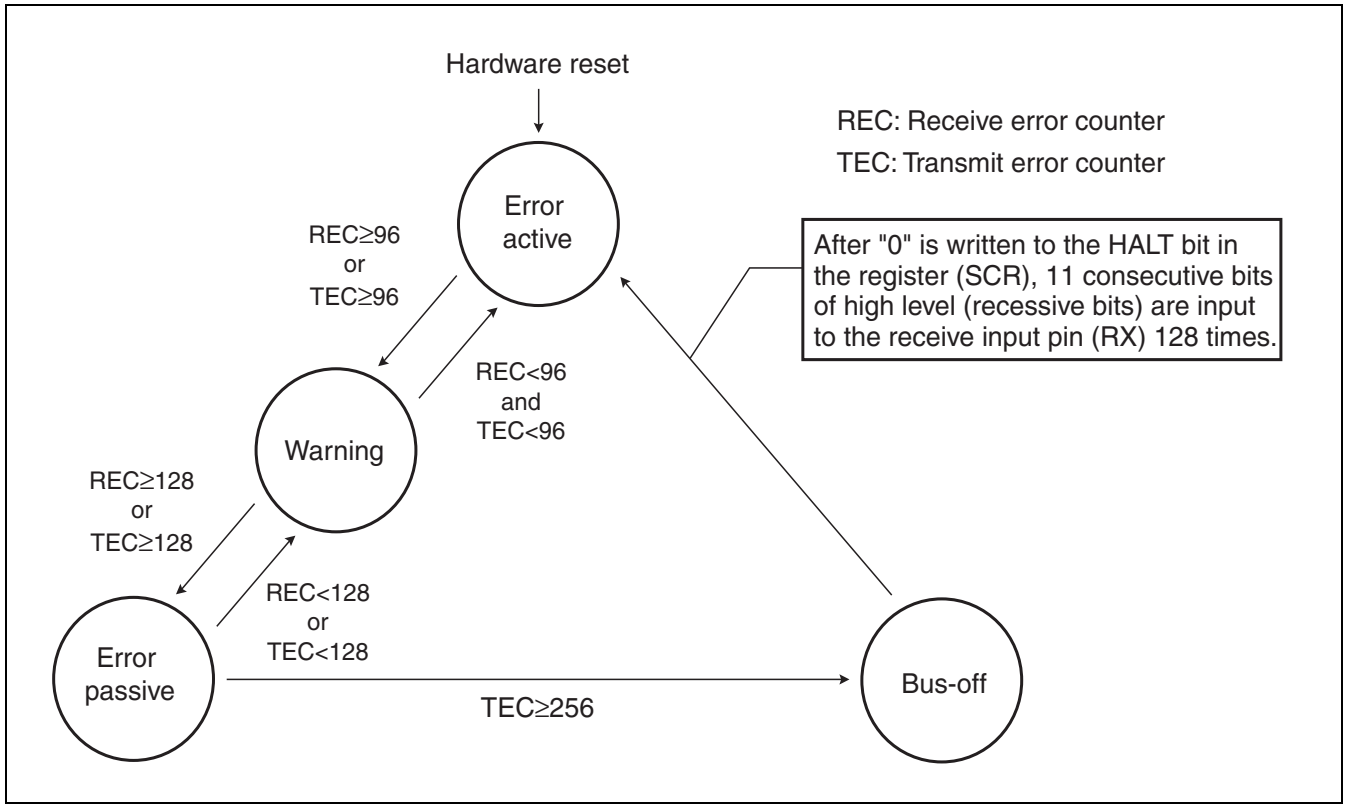
Table 18.3-6 Relationship Between NS1/NS0 and Node Status

NS1	NS0	Node Status
0	0	Error active
0	1	Warning (Error active)
1	0	Error passive
1	1	Bus-off

Note:

Warning (error active) is considered as part of error active by the explanation of node status in CAN Specifications 2.0B, but it indicates that the receive or transmit error counter exceeded 96. Figure 18.3-2 shows a diagram of node status transition.

Figure 18.3-2 Node Status Transition Diagram



[bit7] TOE: Transmission output enable bit

Writing "1" to the TOE bit switches the general-purpose port pin to the CAN controller send pin.

- 0: General-purpose port pin
- 1: CAN controller send pin

[bit2] NIE: Node status transition interrupt enable bit

The NIE bit enables or disables node status transition interrupts (when NT = 1).

- 0: Disables node status transition interrupts.
- 1: Enables node status transition interrupts.

[bit0] HALT: Bus operation stop bit

The HALT bit sets or releases bus operation stop, or indicates the state of the bus operation.

When reading

- 0: Indicates that the bus is in operation.
- 1: Indicates that bus operation is stopped.

When writing

- 0: Releases bus operation stop.
- 1: Sets bus operation stop.

Note:

Make sure that the HALT bit is "1" before you write "0" to this bit during bus-off.
Reference program example:

```
switch(IO_CANCT0.CSR.bit.NS)
    {
        case0:/*error active*/
                break;
        case1:/*warning*/
                break;
         case2:/*error passive*/
                break;
        default:/*bus off*/
                for(i=0;(i<=500)&&(IO_CANCT0.CSR.bit.HALT==0);i++);
                IO_CANCT0.CSR.word=0x0084;/*HALT=0*/
                break;
    }
```

Note: Variable i is used as a fail-safe measure.

■ Bus Operation Stop Bit (HALT = 1)

The bus operation stop bit sets or releases bus operation stop, or indicates the state of the bus operation.

● Condition to set bus operation stop (HALT = 1)

Bus operation stop (HALT = 1) is set under the following three conditions:

- After hardware reset
- When the node status changes to bus-off
- When "1" is written to HALT

Notes:

- Bus operation must be stopped by writing "1" to HALT before F^{2}MC-16LX enters the low-power consumption mode (stop mode, clock mode, or hardware standby mode). If "1" is written to HALT during transmission, bus operation will stop after the transmission is completed (HALT = 1). If "1" is written to HALT during reception, bus operation stops immediately (HALT = 1). If a receive message is being stored in message buffer (x), bus operation will stop after the message is stored (HALT = 1).
- Always read HALT bit to check whether bus operation has stopped.

● Condition to release bus operation stop (HALT = 0)

Writing "0" to the HALT bit releases bus operation stop.

Notes:

- The release of bus operation stop which was set after hardware reset or by writing "1" to HALT will be performed after "0" is written to HALT and then 11 consecutive bits of high level (recessive bits) are input to the receive input pin (RX).
- The release of bus operation stop which was set by the change in the node status to bus-off will be performed after "0" is written to HALT and then 11 consecutive bits of high level (recessive bits) are input to the receive input pin (RX) 128 times.
  Then, the values of both the transmit and receive error counters reach "0", and the node status changes to error active.
- Make sure that the HALT bit is "1" before you write "0" to this bit during bus-off.

■ State Between Bus Operation Stops (HALT = 1)

While bus operation is stopped:

- All bus operation is disabled including sending and receiving.
- The transmission output pin (TX) outputs high level (recessive bit).
- The values of other registers and error counters do not change.

Note:

The bit timing register (BTR) must be set while bus operation is stopped (HALT = 1).

18.3.2 Last Event Indication Register (LEIR)

The last event indication register (LEIR) indicates the last event.
NTE, TCE and RCE are exclusive. When one of the last event bits is set to "1", other bits are set to "0".

■ Bit Configuration of Last Event Indication Register (LEIR)

Figure 18.3-3 shows the bit configuration of the last event indication register (LEIR).

Figure 18.3-3 Bit Configuration of Last Event Indication Register (LEIR)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C02_H$ (CAN0) $003D02_H$ (CAN1) $003E02_H$ (CAN2) $003F02_H$ (CAN3)	NTE	TCE	RCE	–	MBP3	MBP2	MBP1	MBP0	
	R/W	R/W	R/W	–	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	–	0	0	0	0	<- Initial value

[bit7] NTE: Node status transition event bit

When the NTE bit is "1", it indicates that node status transition was the last event.

The NTE bit is set to "1" simultaneously with the NT bit in the control status register (CSR).

The NTE bit is set to "1" independently of the setting of the node status transition interrupt enable (NIE) bit in CSR.

Writing "0" to the NTE bit sets the bit to "0". Writing "1" to the NTE bit is ignored.

"1" is read from this bit when a read-modify-write (RMW) instruction is executed.

[bit6] TCE: Transmission complete event bit

When the TCE bit is "1", it indicates that transmission completion was the last event.

The TCE bit is set to "1" simultaneously with any 1 bit in the transmission complete register (TCR).The TCE bit is set to "1" independently of the bit setting of the transmission interrupt enable register (TIER).

Writing "0" to the TCE bit sets the bit to "0". Writing "1" to the TCE bit is ignored.

"1" is read from this bit when a read-modify-write (RMW) instruction is executed.

When this bit is set to "1", the MBP3 to MBP0 bits are used to indicate the number of the message buffer for which a send operation was completed.

[bit5] RCE: Receive complete event bit

When the RCE bit is "1", it indicates that reception completion was the last event.

The RCE bit is set to "1" simultaneously with any 1 bit in the receive complete register (RCR).The RCE bit is set to "1" independently of the bit setting of the receive interrupt enable register (RIER).

Writing "0" to the RCE bit sets the bit to "0". Writing "1" to the RCE bit is ignored.

"1" is read from this bit when a read-modify-write (RMW) instruction is executed.

When the RCE bit is set to "1", the MBP3 to MBP0 bits are used to indicate the number of the message buffer for which a receive operation was completed.

[bit3 to bit0] MBP3 to MBP0: Message buffer pointer bits

When the TCE or RCE bit is set to "1", the MBP3 to MBP0 bits indicate the number of the relevant message buffer (0 to 15).

When the NTE bit is set to "1", the MBP3 to MBP0 bits have no meaning.

Writing "0" to the MBP3 to MBP0 bits sets these bits to "0". Writing "1" to the MBP3 to MBP0 bits is ignored.

"1" is read from these bits when a read-modify-write (RMW) instruction is executed.

When LEIR is accessed within the CAN interrupt handler, the event that caused an interrupt is not necessarily the same as that indicated by LEIR. At the time when an interrupt to LEIR access is requested within the interrupt handler, another CAN event may occur.

18.3.3 Receive and Transmit Error Counters (RTEC)

The receive and transmit error counters (RTEC) indicate the transmit error count and receive error count defined by the CAN specifications.This register is read-only.

■ Bit Configuration of Receive and Transmit Error Counters (RTEC)

Figure 18.3-4 shows the bit configuration of the receive and transmit error counters (RTEC).

Figure 18.3-4 Bit Configuration of Receive and Transmit Error Counters (RTEC)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C05H (CAN0) 003D05H (CAN1) 003E05H (CAN2) 003F05H (CAN3)	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	
	R	R	R	R	R	R	R	R	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C04H (CAN0) 003D04H (CAN1) 003E04H (CAN2) 003F04H (CAN3)	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	
	R	R	R	R	R	R	R	R	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

[bit15 to bit8] TEC7 to TEC0: Transmit error counter

TEC7 to TEC0 are a transmit error counter.

TEC7 to TEC0 indicate 0 to 7 for a counter value greater than 256. The subsequent increments are not counted as a counter value. In such a case, the node status is shown as bus-off (NS1 and NS0 in the control status register (CSR) = 11_B).

[bit7 to bit0] REC7 to REC0: Receive error counter

REC7 to REC0 are a receive error counter.

REC7 to REC0 indicate 0 to 7 for a counter value greater than 256. The subsequent increments are not counted as a counter value. In such a case, the node status is shown as error passive (NS1 and NS0 in the control status register (CSR) = 10_B).

18.3.4 Bit Timing Register (BTR)

The bit timing register (BTR) sets the prescaler and bit timing.

■ Bit Configuration of Bit Timing Register (BTR)

Figure 18.3-5 shows the bit configuration of the bit timing register (BTR).

Figure 18.3-5 Bit Configuration of Bit Timing Register (BTR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C07_H$ (CAN0) $003D07_H$ (CAN1) $003E07_H$ (CAN2) $003F07_H$ (CAN3)	–	TS2.2	TS2.1	TS2.0	TS1.3	TS1.2	TS1.1	TS1.0	
	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	–	1	1	1	1	1	1	1	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C06_H$ (CAN0) $003D06_H$ (CAN1) $003E06_H$ (CAN2) $003F06_H$ (CAN3)	RSJ1	RSJ0	PSC5	PSC4	PSC3	PSC2	PSC1	PSC0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	1	1	1	1	1	1	1	1	<- Initial value

Note:

BTR must be set while the bus operation is stopped (HALT = 1).

[bit14 to bit12] TS2.2 to TS2.0: Time segment 2 setting bits 2 to 0

The TS2.2 to TS2.0 bits determine time segment 2 (TSEG2) by dividing the unit time (TQ) by [(TS2.2 to TS2.0)+1]. Time segment 2 is equivalent to phase buffer segment 2 (PHASE_SEG2) in the CAN specifications.

[bit11 to bit8] TS1.3 to TS1.0: Time segment 1 setting bits 3 to 0

The TS1.3 to TS1.0 bits determine time segment 1 (TSEG1) by dividing the unit time (TQ) by [(TS1.3 to TS1.0)+1]. Time segment 1 is equivalent to the propagation segment (PROP_SEG) + phase buffer segment 1 (PHASE_SEG1) in the CAN specifications.

[bit7, bit6] RSJ1, RSJ0: Re-synchronous jump width setting bits 1, 0

The RSJ1 and RSJ0 bits determine the re-synchronous jump width by dividing the unit time (TQ) by [(RSJ1, RSJ0)+1].

[bit5 to bit0] PSC5 to PSC0: Prescaler setting bits 5 to 0

The PSC5 to PSC0 bits determine the unit time (TQ) of the CAN controller by dividing the input clock by [(PSC5 to PSC0)+1].

Figure 18.3-6 and Figure 18.3-7 show the bit time segments in the CAN specifications and in the CAN controller respectively.

Figure 18.3-6 Bit Time Segments in CAN Specifications

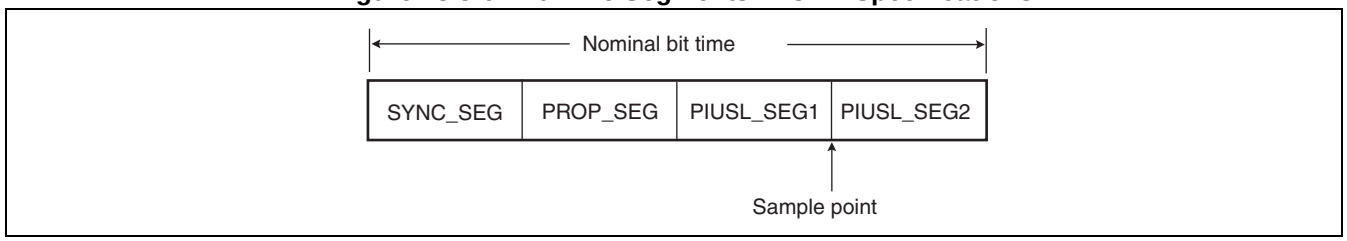
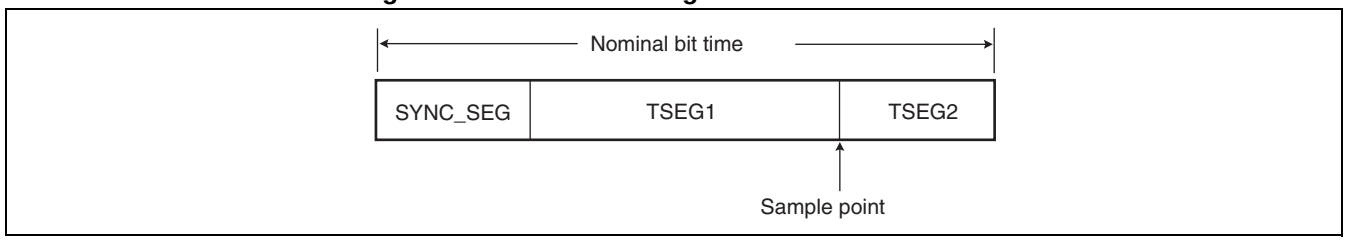


Figure 18.3-7 Bit Time Segments in CAN Controller



The following shows the relationship among PSC = PSC5 to PSC0, TS1 = TS1.3 to TS1.0, TS2 = TS2.2 to TS2.0, RSJ = RSJ1, and RSJ0 when the input clock (CLK), unit time (TQ), bit time (BT), synchronous segment (SYNC_SEG), time segment 1 and 2 (TSEG1, TSEG2), and re-synchronous jump width [(RSJ1+RSJ0)+1] are frequency-divided.

TQ = (PSC+1) × CLK

BT = SYNC_SEG + TSEG1 + TSEG2

= (1 + (TS1+1) + (TS2+1)) × TQ

= (3 + TS1+TS2) × TQ

RSJW = (RSJ + 1) × TQ

To ensure proper operation, the following conditions must be satisfied:

BT ≥ 8TQ

TSEG2 ≥ RSJW

TSEG2 ≥ 2TQ

When PSC = 0

TSEG1 ≥ 5TQ

When PSC ≥ 1

TSEG1 ≥ 2TQ

TSEG1 ≥ RSJW

■ Sample Setting of the Bit Timing Register

The following is the sample setting of the bit timing register.

● Operating conditions

- Communication speed (BT): 100 kbps (10µs)
- 1TQ: 0.5µs (1/20 of 1BT)
- Re-synchronous jump width (RSJW): 4TQ
- Delay time: 50 ns
- Internal operation frequency: 16 MHz (0.0625µs)

● Sample setting

The following procedure is used to specify the setting value of each bit.

1. TQ = (PSC+1) × CLK
   0.5 = (PSC+1) × 0.0625
   PSC = 0.5/0.0625-1 = 7
   Therefore, the setting value for PSC5 to PSC0 is as follows:
   PSC5 to PSC0 = 000111_B
2. RSJW = (RSJ+1) × TQ
   4TQ = (RSJ+1) × TQ
   RSJ = 4-1 = 3
   Therefore, the setting value for RSJ1 to RSJ0 is as follows:
   RSJ1 to RSJ0 = 11_B
3. TSEG2 ≥ RSJW
   (TS2+1) × TQ ≥ 4TQ
   TS2 ≥ 4-1 ≥ 3
   Therefore, the setting value for TS2.2 to TS2.0 is as follows:
   TS2.2 to TS2.0 ≥ 011_B
4. TSEG1 ≥ RSJW
   (TS1+1) × TQ ≥ 4TQ
   TS1 ≥ 3
   Therefore, the setting value for TS1.3 to TS1.0 is as follows:
   TS1.3 to TS1.0 ≥ 0100_B

Since the condition for the communication speed is 100 kbps, the following combinations are available for 3. and 4.

- TS1.3 to TS1.0 = 1010_B TS2.2 to TS2.0 = 111_B
- TS1.3 to TS1.0 = 1011_B TS2.2 to TS2.0 = 110_B
- TS1.3 to TS1.0 = 1100_B TS2.2 to TS2.0 = 101_B
- TS1.3 to TS1.0 = 1101_B TS2.2 to TS2.0 = 100_B
- TS1.3 to TS1.0 = 1110_B TS2.2 to TS2.0 = 011_B

18.3.5 Message Buffer Valid Register (BVALR)

The message buffer valid register (BVALR) sets the validity of message buffer (x) and indicates the state of the message buffer.

■ Bit Configuration of Message Buffer Valid Register (BVALR)

Figure 18.3-8 shows the bit configuration of the message buffer valid register (BVALR).

Figure 18.3-8 Bit Configuration of Message Buffer Valid Register (BVALR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
000041_H (CAN0) 000071_H (CAN1) $0039C1_H$ (CAN2) $0039D1_H$ (CAN3)	BVAL15	BVAL14	BVAL13	BVAL12	BVAL11	BVAL10	BVAL9	BVAL8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000040_H (CAN0) 000070_H (CAN1) $0039C0_H$ (CAN2) $0039D0_H$ (CAN3)	BVAL7	BVAL6	BVAL5	BVAL4	BVAL3	BVAL2	BVAL1	BVAL0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

The message buffer valid register (BVALR) consists of 16 bits, and each bit sets whether to validate or invalidate each of 16 message buffers.

- 0: Message buffer (x) invalid
- 1: Message buffer (x) valid

When message buffer (x) is set to invalid ("0"), no message is sent or received.

When the buffer is set to invalid ("0") during a send operation, it becomes invalid (BVALx = 0) after the transmission is completed or is ended by an error.

When the buffer is set to invalid ("0") during a receive operation, it immediately becomes invalid (BVALx = 0). If, however, the receive message is being stored in the message buffer (x), the message buffer (x) becomes invalid after the message is stored.

Notes:

- "x" indicates the message buffer number (x = 0 to 15).
- When message buffer (x) is invalidated by writing "0" to the corresponding bit (BVALx), the execution of bit operation instructions is disabled until the bit is set to "0".
- To invalidate the message buffer (BVAL in BVALR=0) while the CAN controller is participating in a CAN communication (the read value of the HALT bit in CSR is "0", and the CAN controller is ready to receive or transmit messages by participating in a CAN bus communication), follow the cautions in Section "18.11 Precautions When Using CAN Controller".

18.3.6 IDE Register (IDER)

The IDE register sets the frame format used by message buffer (x) during send/receive operations.

■ Bit Configuration of IDE Register (IDER)

Figure 18.3-9 shows the bit configuration of the IDE register (IDER).

Figure 18.3-9 Bit Configuration of IDE Register (IDER)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C09_H$ (CAN0) $003D09_H$ (CAN1)	IDE15	IDE14	IDE13	IDE12	IDE11	IDE10	IDE9	IDE8	
$003E09_H$ (CAN2)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
$003F09_H$ (CAN3)	X	X	X	X	X	X	X	X	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C08_H$ (CAN0) $003D08_H$ (CAN1)	IDE7	IDE6	IDE5	IDE4	IDE3	IDE2	IDE1	IDE0	
$003E08_H$ (CAN2)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
$003F08_H$ (CAN3)	X	X	X	X	X	X	X	X	<- Initial value

The IDE register (IDER) consists of 16 bits, and each bit specifies a frame format used for each of 16 message buffers.

- 0: Standard frame format (ID11 bits) is used for message buffer (x)
- 1: Extended frame format (ID29 bits) is used for message buffer (x)

Notes:

- This register must be set while message buffer (x) is invalid (BVALx in the message buffer valid register (BVALR) = 0). If this is set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.
- To invalidate the message buffer (BVAL in BVALR = 0) while the CAN controller is participating in a CAN communication (the read value of the HALT bit in CSR is "0", and the CAN controller is ready to receive or transmit messages by participating in a CAN bus communication), follow the cautions in Section "18.11 Precautions When Using CAN Controller".

18.3.7 Transmission Request Register (TREQR)

The transmission request register (TREQR) sets a transmission request for message buffer (x) and indicate the state of the buffer.

■ Bit Configuration of Transmission Request Register (TREQR)

Figure 18.3-10 shows the bit configuration of the transmission request register (TREQR).

Figure 18.3-10 Bit Configuration of Transmission Request Register (TREQR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
000043_H (CAN0) 000073_H (CAN1) $0039C3_H$ (CAN2) $0039D3_H$ (CAN3)	TREQ15	TREQ14	TREQ13	TREQ12	TREQ11	TREQ10	TREQ9	TREQ8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000042_H (CAN0) 000072_H (CAN1) $0039C2_H$ (CAN2) $0039D2_H$ (CAN3)	TREQ7	TREQ6	TREQ5	TREQ4	TREQ3	TREQ2	TREQ1	TREQ0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

When "1" is written to TREQx, transmission for message buffer (x) will start. If RFWTx in the remote frame receive wait register (RFWTR)*1 is "0", transmission starts immediately. If RFWTx = 1, transmission is delayed until a remote frame is received (the remote request receive register (RRTRR)*1 becomes "1"), and then starts. If "1" is written to TREQx when RRTRx is already "1", transmission starts immediately even when RFWTx = 1.*2

*1: For details about TRTRR and RFWTR, see Sections "18.3.8 Transmission RTR register (TRTRR)" and "18.3.9 Remote Frame Receive Wait Register (RFWTR)".

*2: For details of clearing transmission, see Sections "18.3.10 Transmission Cancel Register (TCANR)" and "18.3.11 Transmission Complete Register (TCR)".

Writing "0" to TREQx is ignored.

"0" is read from this bit when a read-modify-write (RMW) instruction is executed.

If the attempts to clear this bit (to "0") when a send operation is completed and to set the bit by writing "1" occur at the same time, clearing the bit has priority.

When "1" is written to more than one bit, transmission starts from the message buffer (x) with the lowest number.

TREQx remains "1" in transmission wait state, and becomes "0" when transmission is completed or cleared.

18.3.8 Transmission RTR register (TRTRR)

The transmission RTR register (TRTRR) sets the transmission RTR (remote transmission request) bit via message buffer (x).

■ Bit Configuration of Transmission RTR Register (TRTRR)

Figure 18.3-11 shows the bit configuration of the transmission RTR register (TRTRR).

Figure 18.3-11 Bit Configuration of Transmission RTR Register (TRTRR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C0B_H$ (CAN0) $003D0B_H$ (CAN1) $003E0B_H$ (CAN2) $003F0B_H$ (CAN3)	TRTR15	TRTR14	TRTR13	TRTR12	TRTR11	TRTR10	TRTR9	TRTR8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C0A_H$ (CAN0) $003D0A_H$ (CAN1) $003E0A_H$ (CAN2) $003F0A_H$ (CAN3)	TRTR7	TRTR6	TRTR5	TRTR4	TRTR3	TRTR2	TRTR1	TRTR0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

The transmission RTR register (TRTRR) consists of 16 bits, and each bit sets the remote transmission request for each of 16 message buffers.

- 0: Sends a data frame.
- 1: Sends a remote frame.

18.3.9 Remote Frame Receive Wait Register (RFWTR)

The remote frame receive wait register (RFWTR) sets the condition for starting transmission when a request for data frame transmission is set (TREQx in the transmission request register (TREQR) is "1" and TRTRx in the transmission RTR register (TRTRR) is "0").

■ Bit Configuration of Remote Frame Receive Wait Register (RFWTR)

Figure 18.3-12 shows the bit configuration of the remote frame receive wait register (RFWTR).

Figure 18.3-12 Bit Configuration of Remote Frame Receive Wait Register (RFWTR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C0D$_H$ (CAN0) 003D0D$_H$ (CAN1) 003E0D$_H$ (CAN2) 003F0D$_H$ (CAN3)	RFWT15	RFWT14	RFWT13	RFWT12	RFWT11	RFWT10	RFWT9	RFWT8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C0C$_H$ (CAN0) 003D0C$_H$ (CAN1) 003E0C$_H$ (CAN2) 003F0C$_H$ (CAN3)	RFWT7	RFWT6	RFWT5	RFWT4	RFWT3	RFWT2	RFWT1	RFWT0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

The remote frame receive wait register (RFWTR) consists of 16 bits, and each bit sets the transmission start condition when the corresponding message buffer is set for data frame transmission.

- 0: Starts transmission immediately.
- 1: Waits until a remote frame is received (the remote request receive register (RRTRR) becomes "1"), and then starts transmission.

Notes:

- Transmission starts immediately if a request for transmission is set when RRTRx is already "1".
- Do not set RFWTx to "1" for remote frame transmission.

18.3.10 Transmission Cancel Register (TCANR)

The transmission cancel register (TCANR) cancels requests in wait state for transmission to message buffer (x) when "1" is written to TCANx.
When canceling is completed, TREQx in the transmission request register (TREQR) becomes "0". Writing "0" to TCANx is ignored.
The transmission cancel register (TCANR) is a write-only register, and its read value is always "0".

■ Bit Configuration of Transmission Cancel Register (TCANR)

Figure 18.3-13 shows the bit configuration of the transmission cancel register (TCANR).

Figure 18.3-13 Bit Configuration of Transmission Cancel Register (TCANR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
000045_H (CAN0) 000075_H (CAN1) $0039C5_H$ (CAN2) $0039D5_H$ (CAN3)	TCAN15	TCAN14	TCAN13	TCAN12	TCAN11	TCAN10	TCAN9	TCAN8	
	W	W	W	W	W	W	W	W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000044_H (CAN0) 000074_H (CAN1) $0039C4_H$ (CAN2) $0039D4_H$ (CAN3)	TCAN7	TCAN6	TCAN5	TCAN4	TCAN3	TCAN2	TCAN1	TCAN0	
	W	W	W	W	W	W	W	W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

18.3.11 Transmission Complete Register (TCR)

When transmission via message buffer (x) is completed, the corresponding TCx becomes "1".
When TIEx in the transmission interrupt enable register (TIER) is "1", an interrupt is generated.

■ Bit Configuration of Transmission Complete Register (TCR)

Figure 18.3-14 shows the bit configuration of the transmission complete register (TCR).

Figure 18.3-14 Bit Configuration of Transmission Complete Register (TCR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
000047_H (CAN0) 000077_H (CAN1) $0039C7_H$ (CAN2) $0039D7_H$ (CAN3)	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000046_H (CAN0) 000076_H (CAN1) $0039C6_H$ (CAN2) $0039D6_H$ (CAN3)	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

TCx = 0 is made under the following conditions:

- "0" is written to TCx.
- "1" is written to TREQx in the transmission request register (TREQR).
  When "0" is written to TCx after transmission is completed, TCx is set to "0". Writing "1" to TCx is ignored. "1" is read from this bit when a read-modify-write (RMW) instruction is executed.

Note:

If the attempts to set this bit to "1" when transmission is completed and to clear the bit by writing "0" occur at the same time, setting the bit to "1" has priority.

18.3.12 Transmission Interrupt Enable Register (TIER)

The transmission interrupt enable register (TIER) enables or disables transmission interrupts via message buffer (x). A transmission interrupt is generated when transmission is completed (when TCx in the transmission complete register (TCR) becomes "1").

■ Bit Configuration of Transmission Interrupt Enable Register (TIER)

Figure 18.3-15 shows the bit configuration of the transmission interrupt enable register (TIER).

Figure 18.3-15 Bit Configuration of Transmission Interrupt Enable Register (TIER)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C0F_H$ (CAN0) $003D0F_H$ (CAN1) $003E0F_H$ (CAN2) $003F0F_H$ (CAN3)	TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C0E_H$ (CAN0) $003D0E_H$ (CAN1) $003E0E_H$ (CAN2) $003F0E_H$ (CAN3)	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

The transmission interrupt enable register (TIER) consists of 16 bits, and each bit sets whether to enable or disable transmission interrupts to each of 16 message buffers.

- 0: Disables transmission interrupts.
- 1: Enables transmission interrupts.

18.3.13 Receive Complete Register (RCR)

When storing a receive message in message buffer (x) is completed, RCx becomes "1".
When RIEx in the receive complete interrupt enable register is "1", an interrupt is generated.

■ Bit Configuration of Receive Complete Register (RCR)

Figure 18.3-16 shows the bit configuration of the receive complete register (RCR).

Figure 18.3-16 Bit Configuration of Receive Complete Register (RCR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
000049_H (CAN0) 000079_H (CAN1) $0039C9_H$ (CAN2) $0039D9_H$ (CAN3)	RC15	RC14	RC13	RC12	RC11	RC10	RC9	RC8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
000048_H (CAN0) 000078_H (CAN1) $0039C8_H$ (CAN2) $0039D8_H$ (CAN3)	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

RCx = 0 is made under the following conditions:

- "0" is written to RCx.
- After the receive message is processed, write "0" to RCx. Otherwise, writing "1" to RCx is ignored.
  "1" is read from this bit when a read-modify-write (RMW) instruction is executed.

Note:

If the attempts to set this bit to "1" when a receive operation is completed and to clear the bit by writing "0" occur at the same time, setting the bit to "1" has priority.

18.3.14 Remote Request Receive Register (RRTRR)

When a received remote frame is stored in message buffer (x), RRTRx becomes "1" (at the same time when the RCx setting becomes "1").

■ Bit Configuration of Remote Request Receive Register (RRTRR)

Figure 18.3-17 shows the bit configuration of the remote request receive register (RRTRR).

Figure 18.3-17 Bit Configuration of Remote Request Receive Register (RRTRR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$00004B_H$ (CAN0) $00007B_H$ (CAN1) $0039CB_H$ (CAN2) $0039DB_H$ (CAN3)	RRTR15	RRTR14	RRTR13	RRTR12	RRTR11	RRTR10	RRTR9	RRTR8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$00004A_H$ (CAN0) $00007A_H$ (CAN1) $0039CA_H$ (CAN2) $0039DA_H$ (CAN3)	RRTR7	RRTR6	RRTR5	RRTR4	RRTR3	RRTR2	RRTR1	RRTR0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

RRTRx = 0 is made under the following conditions:

- "0" is written to RRTRx.
- After a received data frame is stored in message buffer (x) (at the same time when the RCx setting becomes "1").
- After transmission via message buffer (x) is completed (when TCx in the transmission complete register (TCR) is "1"). Writing "1" to RRTRx is ignored.
  "1" is read from this bit when a read-modify-write (RMW) instruction is executed.

Note:

If the attempts to set this bit to "1" and to clear the bit by writing "0" occur at the same time, setting the bit to "1" has priority.

18.3.15 Receive Overrun Register (ROVRR)

If the receive complete register (RCR) is already "1" when storing a receive message into message buffer (x) is completed, ROVRx becomes "1" to indicate that reception caused an overrun.

■ Bit Configuration of Receive Overrun Register (ROVRR)

Figure 18.3-18 shows the bit configuration of the receive overrun register (ROVRR).

Figure 18.3-18 Bit Configuration of Receive Overrun Register (ROVRR)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
00004D$_H$ (CAN0) 00007D$_H$ (CAN1) 0039CD$_H$ (CAN2) 0039DD$_H$ (CAN3)	ROVR15	ROVR14	ROVR13	ROVR12	ROVR11	ROVR10	ROVR9	ROVR8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
00004C$_H$ (CAN0) 00007C$_H$ (CAN1) 0039CC$_H$ (CAN2) 0039DC$_H$ (CAN3)	ROVR7	ROVR6	ROVR5	ROVR4	ROVR3	ROVR2	ROVR1	ROVR0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Writing "0" to ROVRx makes ROVRx = 0. Writing "1" to ROVRx is ignored. When "0" is written to ROVRx after a reception overrun is confirmed, ROVRx is set to "0".

"1" is read from this bit when a read-modify-write (RMW) instruction is executed.

Note:

If the attempts to set this bit to "1" and to clear the bit by writing "0" occur at the same time, setting the bit to "1" has priority.

18.3.16 Receive Interrupt Enable Register (RIER)

The receive interrupt enable register (RIER) enables or disables receive interrupts via message buffer (x).
A receive interrupt is generated when reception is completed (when RCx in the receive complete register (RCR) is "1").

■ Bit Configuration of Receive Interrupt Enable Register (RIER)

Figure 18.3-19 shows the bit configuration of the receive interrupt enable register (RIER).

Figure 18.3-19 Bit Configuration of Receive Interrupt Enable Register (RIER)

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
00004F$_H$ (CAN0) 00007F$_H$ (CAN1) 0039CF$_H$ (CAN2) 0039DF$_H$ (CAN3)	RIE15	RIE14	RIE13	RIE12	RIE11	RIE10	RIE9	RIE8	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
00004E$_H$ (CAN0) 00007E$_H$ (CAN1) 0039CE$_H$ (CAN2) 0039DE$_H$ (CAN3)	RIE7	RIE6	RIE5	RIE4	RIE3	RIE2	RIE1	RIE0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	0	0	0	0	0	0	0	0	<- Initial value

The receive interrupt enable register (RIER) consists of 16 bits, and each bit sets whether to enable or disable receive interrupts to each of 16 message buffers.

- 0: Disables receive interrupts.
- 1: Enables receive interrupts.

18.3.17 Acceptance Mask Selection Register (AMSR)

The acceptance mask selection register (AMSR) selects a mask (acceptance mask) for the comparison between the receive message ID and message buffer (x) ID.

■ Bit Configuration of Acceptance Mask Selection Register (AMSR)

Figure 18.3-20 shows the bit configuration of the acceptance mask selection register (AMSR).

As listed in Table 18.3-7, the acceptance mask for the corresponding message buffer is selected based on a combination of 2 bits.

Figure 18.3-20 Bit Configuration of Acceptance Mask Selection Register (AMSR)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C10_H$ (CAN0) $003D10_H$ (CAN1) $003E10_H$ (CAN2) $003F10_H$ (CAN3)	AMS3.1	AMS3.0	AMS2.1	AMS2.0	AMS1.1	AMS1.0	AMS0.1	AMS0.0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C11_H$ (CAN0) $003D11_H$ (CAN1) $003E11_H$ (CAN2) $003F11_H$ (CAN3)	AMS7.1	AMS7.0	AMS6.1	AMS6.0	AMS5.1	AMS5.0	AMS4.1	AMS4.0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003C12_H$ (CAN0) $003D12_H$ (CAN1) $003E12_H$ (CAN2) $003F12_H$ (CAN3)	AMS11.1	AMS11.0	AMS10.1	AMS10.0	AMS9.1	AMS9.0	AMS8.1	AMS8.0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003C13_H$ (CAN0) $003D13_H$ (CAN1) $003E13_H$ (CAN2) $003F13_H$ (CAN3)	AMS15.1	AMS15.0	AMS14.1	AMS14.0	AMS13.1	AMS13.0	AMS12.1	AMS12.0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

Table 18.3-7 Selection of the Acceptance Mask

AMSx.1	AMSx.0	Acceptance Mask
0	0	Full-bit compare
0	1	Full-bit mask
1	0	Acceptance mask register 0 (AMR0)
1	1	Acceptance mask register 1 (AMR1)

Notes:

- AMSx.1 and AMSx.0 must be set while message buffer (x) is invalid (when BVALx in the message buffer valid register (BVALR) is "0"). If these bits are set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.
- To invalidate the message buffer (BVAL in BVALR = 0) while the CAN controller is participating in a CAN communication (the read value of the HALT bit in CSR is "0", and the CAN controller is ready to receive or transmit messages by participating in a CAN bus communication), follow the cautions in Section "18.11 Precautions When Using CAN Controller".

18.3.18 Acceptance Mask Registers 0 and 1 (AMR0/AMR1)

There are 2 types of acceptance mask register: AMR0 and AMR1, and both can be used in either the standard frame format or extended frame format.
AM28 to AM18 (11 bits) are used for an acceptance mask in the standard frame format; AM28 to AM0 (29 bits) are used for an acceptance mask in the extended frame format.

■ Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1)

Figure 18.3-21 shows the bit configuration of the acceptance mask registers 0 and 1 (AMR0/AMR1).

Figure 18.3-21 Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1)

AMR0 BYTE0

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C14H (CAN0) 003D14H (CAN1) 003E14H (CAN2) 003F14H (CAN3)	AM28	AM27	AM26	AM25	AM24	AM23	AM22	AM21	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

AMR0 BYTE1

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C15H (CAN0) 003D15H (CAN1) 003E15H (CAN2) 003F15H (CAN3)	AM20	AM19	AM18	AM17	AM16	AM15	AM14	AM13	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

AMR0 BYTE2

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C16H (CAN0) 003D16H (CAN1) 003E16H (CAN2) 003F16H (CAN3)	AM12	AM11	AM10	AM9	AM8	AM7	AM6	AM5	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

AMR0 BYTE3

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C18H (CAN0) 003D18H (CAN1) 003E18H (CAN2) 003F18H (CAN3)	AM4	AM3	AM2	AM1	AM0	–	–	–	
	R/W	R/W	R/W	R/W	R/W	–	–	–	<- Read/Write
	X	X	X	X	X	–	–	–	<- Initial value

AMR1 BYTE0

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C14H (CAN0) 003D14H (CAN1) 003E14H (CAN2) 003F14H (CAN3)	AM28	AM27	AM26	AM25	AM24	AM23	AM22	AM21	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

AMR1 BYTE1

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C15H (CAN0) 003D15H (CAN1) 003E15H (CAN2) 003F15H (CAN3)	AM20	AM19	AM18	AM17	AM16	AM15	AM14	AM13	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

(Continued)

(Continued)

AMR1 BYTE2

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
003C1A$_H$ (CAN0) 003D1A$_H$ (CAN1) 003E1A$_H$ (CAN2) 003F1A$_H$ (CAN3)	AM12	AM11	AM10	AM9	AM8	AM7	AM6	AM5	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

AMR1 BYTE3

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
003C1B$_H$ (CAN0) 003D1B$_H$ (CAN1) 003E1B$_H$ (CAN2) 003F1B$_H$ (CAN3)	AM4	AM3	AM2	AM1	AM0	–	–	–	
	R/W	R/W	R/W	R/W	R/W	–	–	–	<- Read/Write
	X	X	X	X	X	–	–	–	<- Initial value

- 0: Compare

Compares the acceptance code corresponding to this bit (IDRx in the ID register to be compared with the receive message ID) with the bit of the receive message ID. If these bits do not match, the message is not received.

- 1: Mask

Masks the acceptance code ID register (IDRx) corresponding to this bit. The comparison with the bit of the receive message ID is not performed.

Notes:

- AMR0 and AMR1 must be set while all message buffers (x) selecting AMR0 and AMR1 are invalid (BVALx in the message buffer valid register (BVALR) is "0"). If these bits are set while the message buffers are valid (BVALx = 1), unnecessary receive messages may be stored.
- To invalidate the message buffer (BVAL in BVALR = 0) while the CAN controller is participating in a CAN communication (the read value of the HALT bit in CSR is "0", and the CAN controller is ready to receive or transmit messages by participating in a CAN bus communication), follow the cautions in Section "18.11 Precautions When Using CAN Controller".

18.3.19 Message Buffers

There are 16 message buffers being provided, and 1 message buffer x (x = 0 to 15) consists of an ID register (IDRx), a DLC register (DLCRx), and a data register (DTRx).

■ Message Buffers

Message buffer (x) is used for both send and receive operations.

Lower-numbered message buffers have higher priority.

When a transmission request is issued to more than one message buffer during transmission, transmission starts from the message buffer having the lowest number (See Section "18.4 CAN Controller Transmission".).

When a receive message ID passes through the acceptance filters (the mechanism to compare the acceptance mask ID of a receive message with the message buffer) of more than one message buffers during reception, the receive message is stored in the message buffer having the lowest number (See Section "18.5 CAN Controller Reception".).

If the same acceptance filter is specified in more than one message buffers, these message buffers can be used as a multi-level message buffer. This makes some time reserve for reception (See Section "18.8 Procedure of Reception Via Message Buffer (x)".).

Notes:

- Write operation to a message buffer and to the general-purpose RAM area must be performed in units of words by using even-numbered addresses. Note that the write operation in units of bytes may result in undefined data being written to the upper byte during writing to the lower byte.
- When the BVALx bit in the message buffer valid register (BVALR) is "0" (invalid), message buffer (x) (IDRx, DLCRx, and DTRx) can be used as general-purpose RAM. If, however, the CAN controller is sending or receiving data, it uses a message buffer, resulting in the possibility of the delay in the CPU access for up to 64 machine cycles. The same processing may occur in the general-purpose RAM area (CAN0: Addresses $003A00_H$ to $003A1F_H$ and addresses $003B00_H$ to $003B1F_H$).

18.3.20 ID Register x (x = 0 to 15) (IDRx)

ID register x (x = 0 to 15) (IDRx) is an ID register for message buffer (x).

■ Bit Configuration of ID Register x (x = 0 to 15) (IDRx)

Figure 18.3-22 shows the bit configuration of ID register x (x = 0 to 15) (IDRx).

Figure 18.3-22 Bit Configuration of ID Register x (x = 0 to 15) (IDRx)

BYTE0

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A20_H$ +4x (CAN0) $003B20_H$ +4x (CAN1) 003720_H +4x (CAN2) 003820_H +4x (CAN3)	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE1

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A21_H$ +4x (CAN0) $003B21_H$ +4x (CAN1) 003721_H +4x (CAN2) 003821_H +4x (CAN3)	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE2

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A22_H$ +4x (CAN0) $003B22_H$ +4x (CAN1) 003722_H +4x (CAN2) 003822_H +4x (CAN3)	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE3

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A23_H$ +4x (CAN0) $003B23_H$ +4x (CAN1) 003723_H +4x (CAN2) 003823_H +4x (CAN3)	ID4	ID3	ID2	ID1	ID0	–	–	–	
	R/W	R/W	R/W	R/W	R/W	–	–	–	<- Read/Write
	X	X	X	X	X	–	–	–	<- Initial value

When message buffer (x) is used in the standard frame format (IDEx in the IDE register (IDER) = 0), use 11 bits of ID28 to ID18. When the buffer is used in the extended frame format (IDEx = 1), use 29 bits of ID28 to ID0.

ID28 to ID0 have the following functions:

- Setting an acceptance code (ID to be compared with a receive message ID)
- Setting a send message ID
  In the standard frame format, it is prohibited to set all bits of ID28 to ID22 to "1".
- Storing a receive message ID
  A receive message ID is stored also in the bits masked by an acceptance mask. In the standard frame format, undefined values (part of a message received previously) are stored in ID17 to ID0.

Notes:

- Write operation to the ID register must be performed in units of words. Note that the write operation in units of bytes may result in undefined data being written to the upper byte during writing to the lower byte. Writing to the upper byte is ignored.
- The ID register must be set while message buffer (x) is invalid (BVALx in the message buffer valid register (BVALR) is "0"). If this register is set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.
- To invalidate the message buffer (BVAL in BVALR = 0) while the CAN controller is participating in a CAN communication (the read value of the HALT bit in CSR is "0", and the CAN controller is ready to receive or transmit messages by participating in a CAN bus communication), follow the cautions in Section "18.11 Precautions When Using CAN Controller".

■ Sample Setting of the ID Register

Table 18.3-8 and Table 18.3-9 show the sample setting of the ID register in the standard and extended frame formats.

Table 18.3-8 Sample Setting of ID Register in Standard Frame Format (Sheet 1 of 2)

ID (Decimal)	ID (Hexadecimal)	BYTE0	BYTE1
1	1	00	20
2	2	00	40
3	3	00	60
4	4	00	80
5	5	00	A0
6	6	00	C0
7	7	00	E0
8	8	01	00
9	9	01	20
10	A	01	40
• •		• •	
30	1E	03	C0
31	1F	03	C1
32	20	03	C2
• •		• •	
100	064	0C	80
101	065	0C	A0
• •		• •	
200	0C8	19	00

Table 18.3-8 Sample Setting of ID Register in Standard Frame Format (Sheet 2 of 2)

ID (Decimal)	ID (Hexadecimal)	BYTE0	BYTE1
• •		• •	
2043	7FB	FF	06
2044	7FC	FF	80
2045	7FD	FF	A0
2046	7FE	FF	C0
2047	7FF	FF	E0

Table 18.3-9 Sample Setting of ID Register in Extended Frame Format (Sheet 1 of 2)

ID (Decimal)	ID (Hexadecimal)	BYTE0	BYTE1	BYTE0	BYTE1
1	1	00	00	00	08
2	2	00	00	00	10
3	3	00	00	00	18
4	4	00	00	00	20
5	5	00	00	00	28
6	6	00	00	00	30
7	7	00	00	00	38
8	8	00	00	00	40
9	9	00	00	00	48
10	A	00	00	00	50
• •		• •			
30	1E	00	00	00	F0
31	1F	00	00	00	F8
32	20	00	00	01	00
• •		• •			
100	064	00	00	03	20
101	065	00	00	03	28
• •		• •			
200	0C8	00	00	06	40
• •		• •			
2043	7FB	00	00	3F	D8
2044	7FC	00	00	3F	E0

Table 18.3-9 Sample Setting of ID Register in Extended Frame Format (Sheet 2 of 2)

ID (Decimal)	ID (Hexadecimal)	BYTE0	BYTE1	BYTE0	BYTE1
2045	7FD	00	00	3F	E8
2046	7FE	00	00	3F	F0
2047	7FF	00	00	3F	F8
• •		• •			
8190	1FFE	00	00	FF	F0
8191	1FFF•	00	00	FF	F8
8192	2000	00	01	00	00
• •		• •			
536870905	1FFFFFF9	FF	FF	FC	80
536870906	1FFFFFFA	FF	FF	FD	00
536870907	1FFFFFFB	FF	FF	FD	80
536870908	1FFFFFFC	FF	FF	FE	00
536870909	1FFFFFFD	FF	FF	FE	80
536870910	1FFFFFFE	FF	FF	FF	00
536870911	1FFFFFFF	FF	FF	FF	80

18.3.21 DLC Register x (x = 0 to 15) (DLCRx)

DLC register x (x = 0 to 15) (DLCRx) stores DLC corresponding to message buffer x.

■ Bit Configuration of DLC Register x (x = 0 to 15) (DLCRx)

Figure 18.3-23 shows the bit configuration of DLC register x (x = 0 to 15) (DLCRx).

Figure 18.3-23 Bit Configuration of DLC Register x (x = 0 to 15) (DLCRx)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A60_H$ +2x (CAN0) $003B60_H$ +2x (CAN1) 003760_H +2x (CAN2) 003860_H +2x (CAN3)	–	–	–	–	DLC3	DLC2	DLC1	DLC0	
	–	–	–	–	R/W	R/W	R/W	R/W	<- Read/Write
	–	–	–	–	X	X	X	X	<- Initial value

● Transmission

- When a data frame is sent (TRTRx in the transmission RTR register (TRTRR) is "0"), the data length of the send message is specified (in units of bytes).
- When a remote frame is sent (TRTRx = 1), the data length of the request message is specified (in units of bytes).

Note:

Setting values other than 0000_B to 1000_B (0 to 8 bytes) is disabled.

● Reception

- When a data frame is received (RRTRx in the remote frame request receive register (RRTRR) is "0"), the data length of the receive message is stored (in units of bytes).
- When a remote frame is received (RRTRx = 1), the data length of the request message is stored (in units of bytes).

Note:

Write operation to the DLC register must be performed in units of words. Note that write operation in units of bytes may result in undefined data being written to the upper byte during writing to the lower byte. Writing to the upper byte is ignored.

18.3.22 Data Register x (x = 0 to 15) (DTRx)

Data register x (x = 0 to 15) (DTRx) is a data register for message buffer (x).
Data register x (x = 0 to 15) (DTRx) is used only to send or receive data frames; it is not used to send or receive remote frames.

■ Bit Configuration of Data Register x (x = 0 to 15) (DTRx)

Figure 18.3-24 shows the bit configuration of data register x (x = 0 to 15) (DTRx).

Figure 18.3-24 Bit Configuration of Data Register x (x = 0 to 15) (DTRx)

BYTE0

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A80_H$ +8x (CAN0) $003B80_H$ +8x (CAN1) 003780_H +8x (CAN2) 003880_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE1

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A81_H$ +8x (CAN0) $003B81_H$ +8x (CAN1) 003781_H +8x (CAN2) 003881_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE2

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A82_H$ +8x (CAN0) $003B82_H$ +8x (CAN1) 003782_H +8x (CAN2) 003882_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE3

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A83_H$ +8x (CAN0) $003B83_H$ +8x (CAN1) 003783_H +8x (CAN2) 003883_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE4

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A84_H$ +8x (CAN0) $003B84_H$ +8x (CAN1) 003784_H +8x (CAN2) 003884_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

BYTE5

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A85_H$ +8x (CAN0) $003B85_H$ +8x (CAN1) 003785_H +8x (CAN2) 003885_H +8x (CAN3)	D7	D6	D5	D4	D3	D2	D1	D0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
	X	X	X	X	X	X	X	X	<- Initial value

(Continued)

(Continued)

BYTE6

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
$003A86_H$ +8x (CAN0)	D7	D6	D5	D4	D3	D2	D1	D0	
$003B86_H$ +8x (CAN1)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
003786_H +8x (CAN2)	X	X	X	X	X	X	X	X	<- Initial value
003886_H +8x (CAN3)									

BYTE7

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
$003A87_H$ +8x (CAN0)	D7	D6	D5	D4	D3	D2	D1	D0	
$003B87_H$ +8x (CAN1)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	<- Read/Write
003787_H +8x (CAN2)	X	X	X	X	X	X	X	X	<- Initial value
003887_H +8x (CAN3)									

- Setting send message data (any length between 0 and 8 bytes)

Sending data begins with MSB, followed by BYTE0, BYTE1, to BYTE7 in this order.

- Receive message data

Storing data begins with MSB, followed by BYTE0, BYTE1, to BYTE7 in this order.

If receive message data is less than 8 bytes, the remaining bytes in the data register (DTRx) to which data is stored are undefined.

Note:

Write operation to the data register must be performed in units of words. Note that write operation in units of bytes may result in undefined data being written to the upper byte during writing to the lower byte. Writing to the upper byte is ignored.

18.4 CAN Controller Transmission

The CAN controller starts the transmission via message buffer (x) when "1" is written to TREQx in the transmission request register (TREQR). At this moment, TREQx becomes "1" and TCx in the transmission complete register (TCR) becomes "0".

■ Starting CAN Controller Transmission

When RFWTx in the remote frame receive wait register (RFWTR) becomes "0", transmission starts immediately. When RFWTx is "1", transmission is delayed until a remote frame is received (RRTRx in the remote request receive register (RRTRR) becomes "1") and then started.

When a request for transmission is issued to more than one message buffer (more than one TREQx is "1"), transmission starts from the message buffer with the lowest number.

Message transmission to the CAN bus (via transmission output pin TX) starts when the bus is idle.

When TRTRx in the transmission RTR register (TRTRR) is "0", a data frame is sent. When TRTRx is "1", a remote frame is sent.

If arbitration for transmission fails due to the conflict of the message buffer with another CAN controller on the CAN bus, or if an error occurs during transmission, the message buffer waits until the bus becomes idle and then repeats retransmission until the transmission is completed successfully.

■ Clearing a CAN Controller Transmission Request

● Clearing via the transmission cancel register (TCANR)

A transmission request issued to message buffer (x) for which transmission was not executed during send wait state can be cleared by writing "1" to TCANx in the transmission cancel register (TCANR). When the clearing is completed, TREQx becomes "0".

● Clearing by storing a receive message

Reception is performed even for the message buffer (x) for which a transmission request was issued but transmission was not executed.

If a request for data frame transmission was issued but transmission was not executed for message buffer (x) (TRTRx = 0 or TREQx = 1), the transmission request is cleared (TREQx = 0) after a receive data frame that passed the acceptance filter is stored. This transmission request is not cleared by storing a remote frame (TREQx = 1 remains unchanged).

If a request for remote frame transmission is issued but transmission was not executed for message buffer (x) (TRTRx = 1 or TREQx = 1), the transmission request is cleared (TREQx = 0) after a receive remote frame that passed the acceptance filter is stored. This transmission request is cleared by storing either of a data frame or a remote frame.

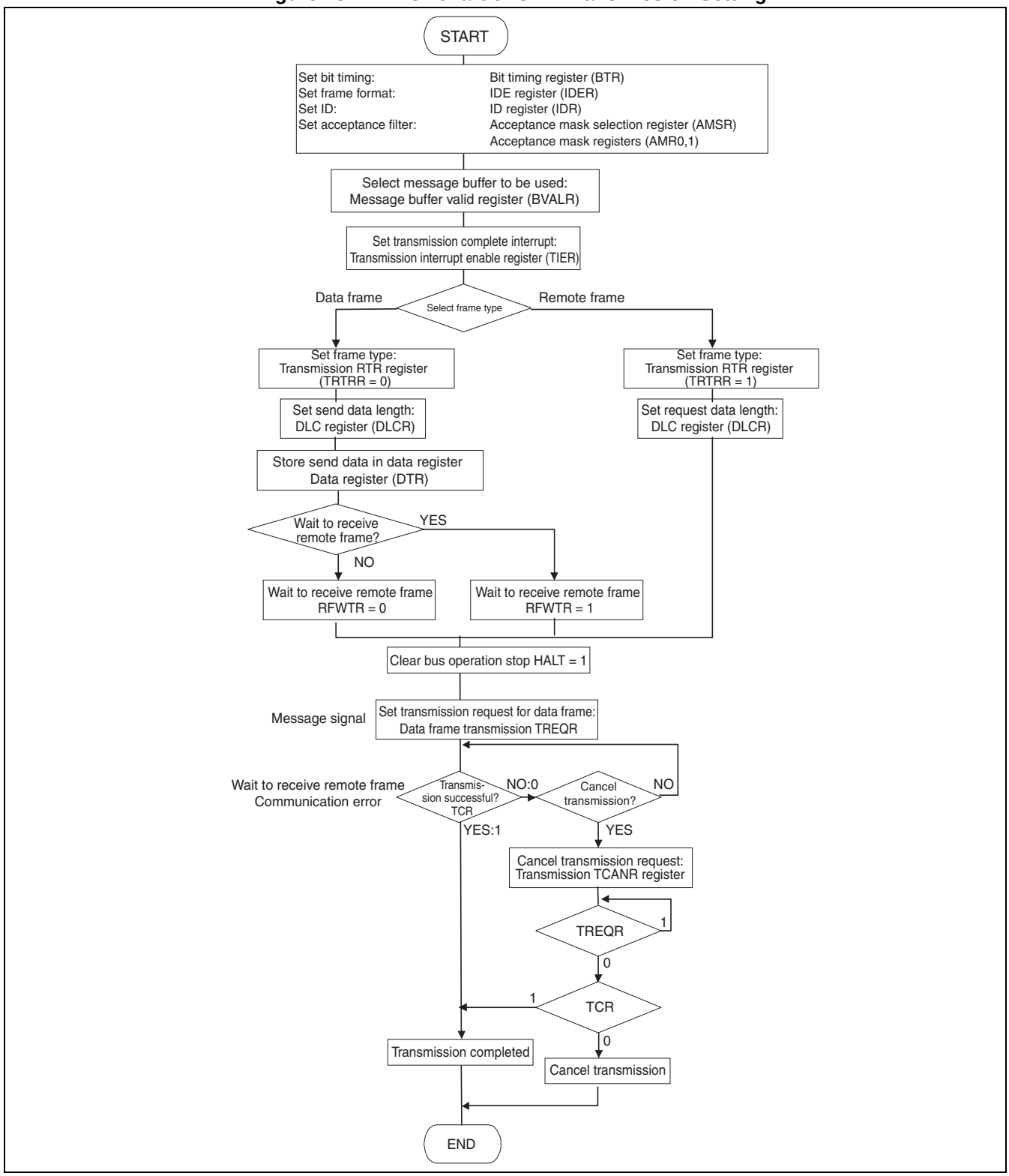
■ Completing CAN Controller Transmission

When transmission is successful, RRTRx and TREQx become "0" and TCx in the transmission complete register (TCR) becomes "1". When transmission complete interrupts are enabled (when TIEx in the transmission interrupt enable register (TIER) is "1"), an interrupt is generated.

■ Flowchart of CAN Transmission Setting

Figure 18.4-1 shows a flowchart of the CAN transmission setting.

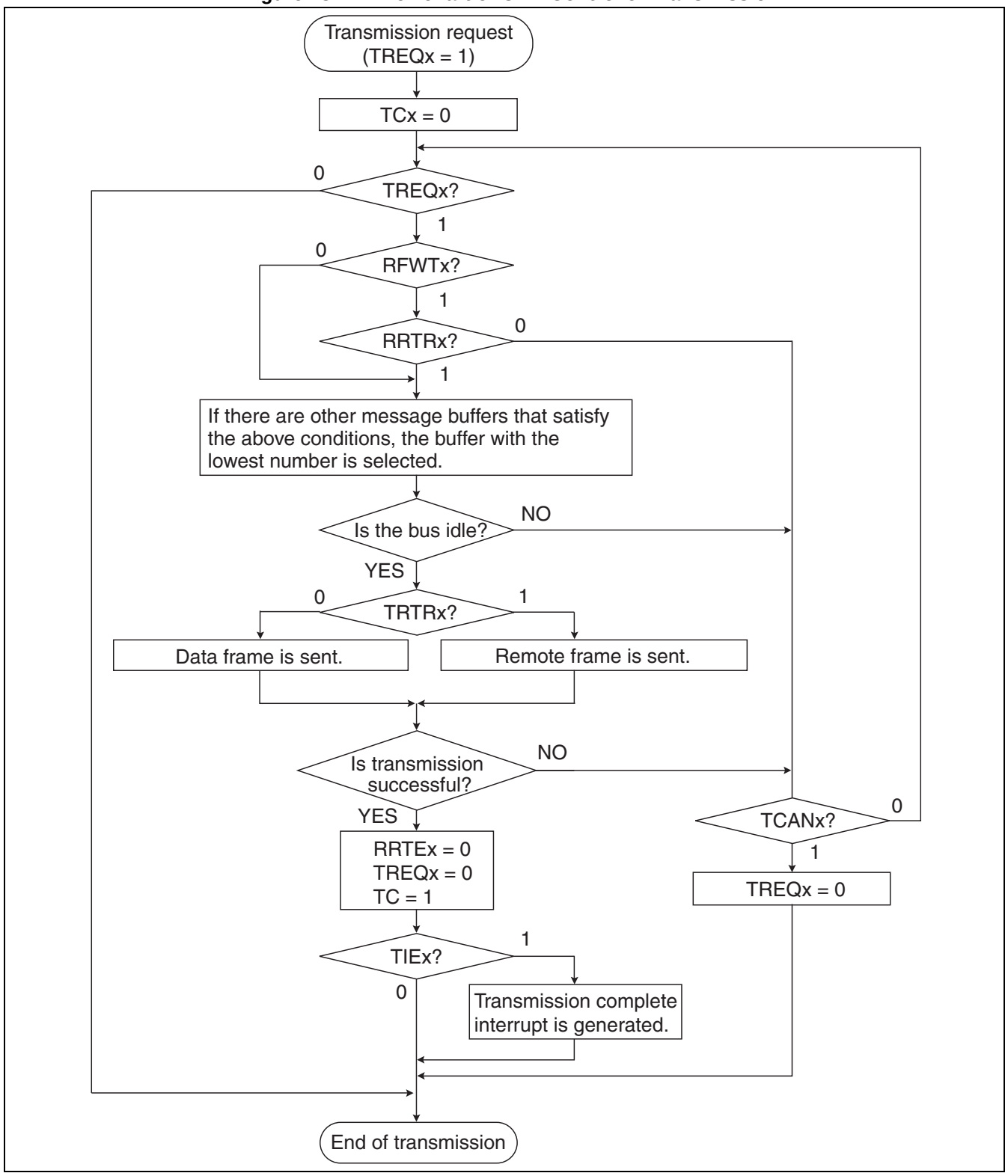
Figure 18.4-1 Flowchart of CAN Transmission Setting



■ Flowchart of CAN Controller Transmission

Figure 18.4-2 shows a flowchart of the CAN controller transmission.

Figure 18.4-2 Flowchart of CAN Controller Transmission



18.5 CAN Controller Reception

Reception starts when the start of a data frame or a remote frame (SOF) is detected on the CAN bus.

■ Acceptance Filtering

Receive messages in the standard frame format are compared with message buffer (x) set in the standard frame format (IDEx in the IDE register (IDER) is "0"). Receive messages in the extended frame format are compared with message buffer (x) set in the extended frame format (IDEx is "1").

When all bit sets to be compared with the acceptance mask match after the comparison of the receive message ID and acceptance code (ID register (IDRx) to be compared with the receive message ID), the receive message passes the acceptance filter for message buffer (x).

■ Storing the Receive Message

When a receive operation is successful, the receive message including the ID that passed the acceptance filter is stored in message buffer (x).

When a data frame is received, the receive message is stored in the ID register (IDRx), DLC register (DLCRx), and data register (DTRx).

Even if the data of the receive message is less than 8 bytes, a certain data is stored in the remaining bytes in DTRx with undefined values.

When a remote frame is received, the receive message is stored only in IDRx and DLCRx, and DTRx remains unchanged.

If more than one message buffer includes IDs that passed the acceptance filter, message buffer x which should store the receive message is determined by the following rule:

Message buffer x (x = 0 to 15) which has the lower number has higher priority. In other words, message buffer 0 has the highest priority, and message buffer 15 has the lowest priority.

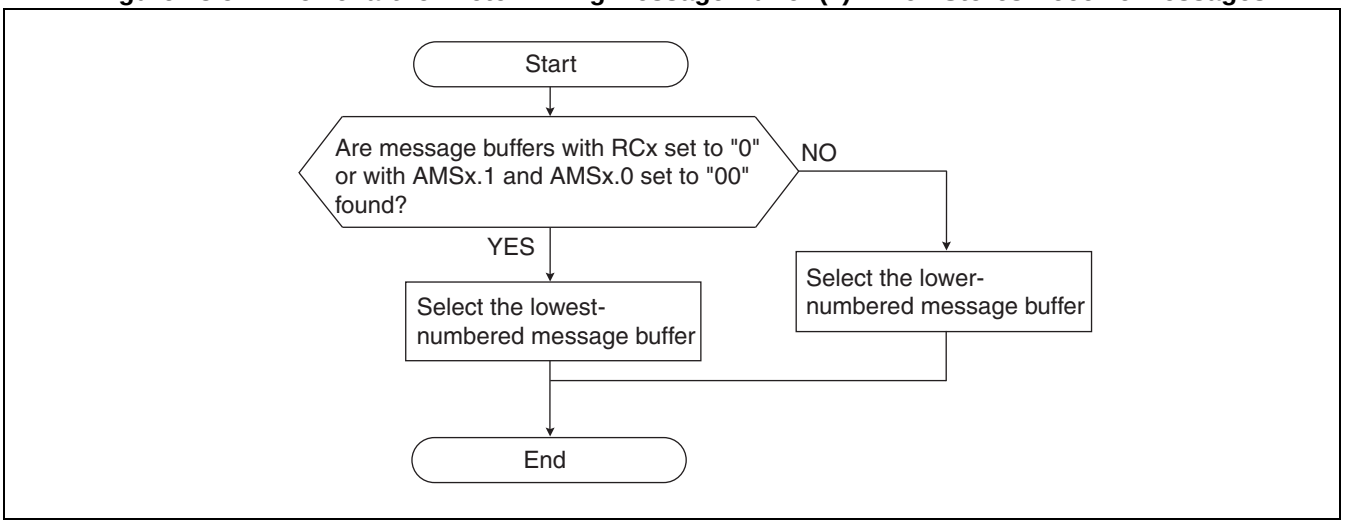
For storing a receive message, the message buffer for which RCx bit in the receive complete register (RCR) is set to "0" basically has priority.

If the bits in the acceptance mask selection register (AMSR) are set for a message buffer for which all-bit compare (AMSx.1 and AMSx.0 bits) are set to 00_B, the receive message is stored regardless of the value of the RCx bit in RCR.

When there are several buffers for which the RCx bit in RCR is set to "0" and the bits in AMSR are set to all-bit compare, the receive message is stored in message buffer x having the lowest number (highest priority). If no such message buffers exist, the receive message is stored in message buffer x having a lower number.

Figure 18.5-1 shows a flowchart of determining message buffer x which should store the receive message. It is recommended to set message buffers in order of buffers for which the bits in AMSR are set to all-bit compare, buffers that use AMR0 or AMR1, and buffers for which the bits in AMSR are set to all-bit mask; sequentially in order of buffer numbers within each group.

Figure 18.5-1 Flowchart for Determining Message Buffer (x) which Stores Receive Messages



■ Receive Overrun

When the RCx bit in the receive complete register (RCR) corresponding to message buffer x in which a receive message is to be stored has already been set to "1" and storing a receive message in message buffer x is completed, the ROVRx bit in the receive overrun register (ROVRR) is set to "1" to indicate a receive overrun.

■ Processing for Receiving Data Frame and Remote Frame

● Processing for receiving a data frame

When a data frame is received, RRTRx in the remote request receive register (RRTRR) becomes "0".

TREQx in the transmission request register (TREQR) becomes "0" immediately before a receive message is stored. Transmission requests to message buffer (x) for which transmission was not executed are cleared. The requests are cleared for both data and remote frame transmission.

● Processing for receiving a remote frame

When a remote frame is received, RRTRx becomes "1".

When TRTRx in the transmission RTR register (TRTRR) is "1", TREQx becomes "0". As a result, requests for remote frame transmission to the message buffer for which transmission was not executed are cleared.

Notes:

- Requests for data frame transmission are not cleared.
- For details of clearing a transmission request, see Section "18.4 CAN Controller Transmission".

■ Receive Complete

RCx in the receive complete register (RCR) becomes "1" after a receive message is stored.

When receive interrupts are enabled (RIEx in the receive interrupt enable register (RIER) is "1"), an interrupt is generated.

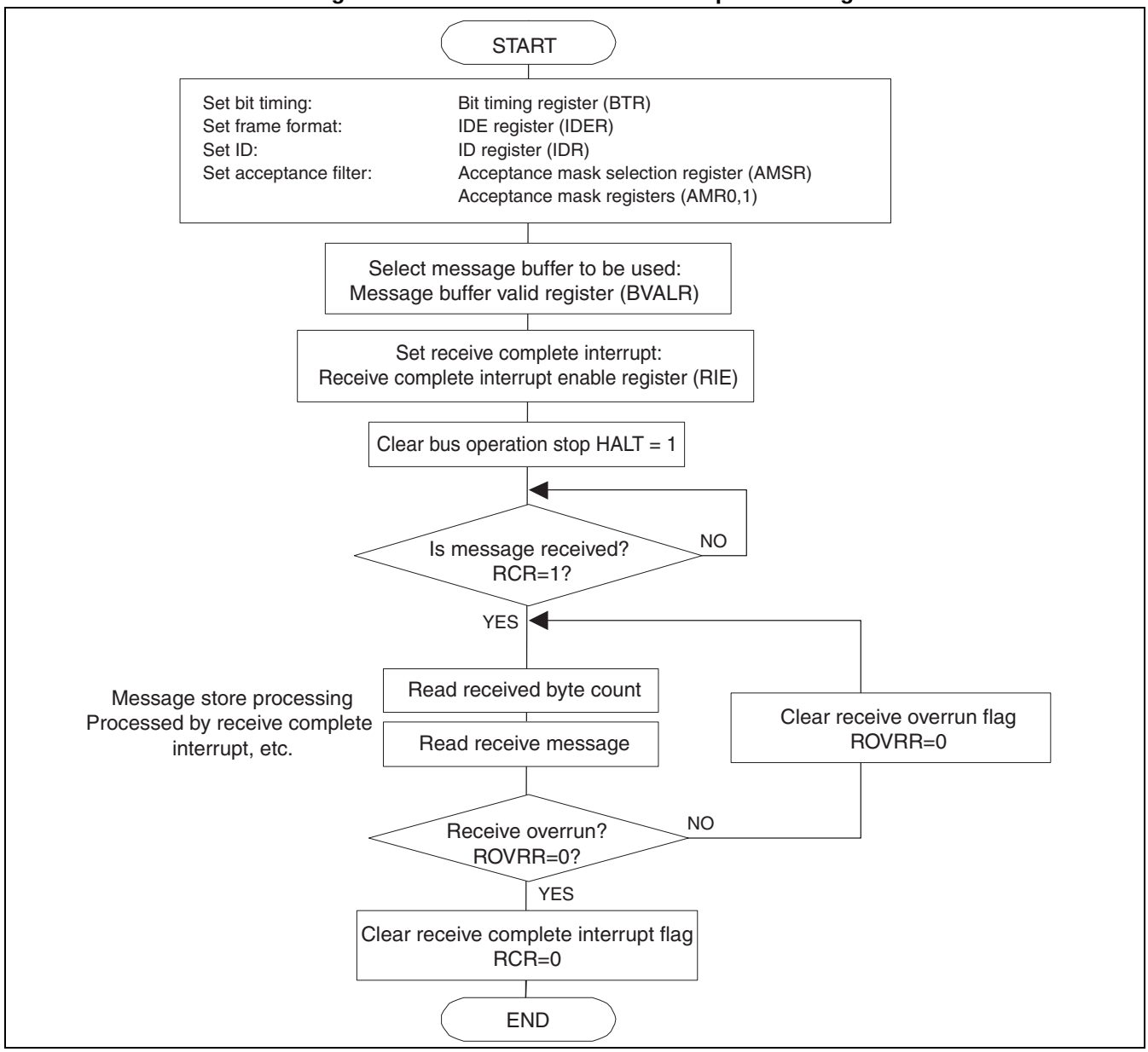
Note:

This CAN controller does not receive messages which were sent by itself.

■ Flowchart of CAN Reception Setting

Figure 18.5-2 shows the flowchart of the CAN reception setting.

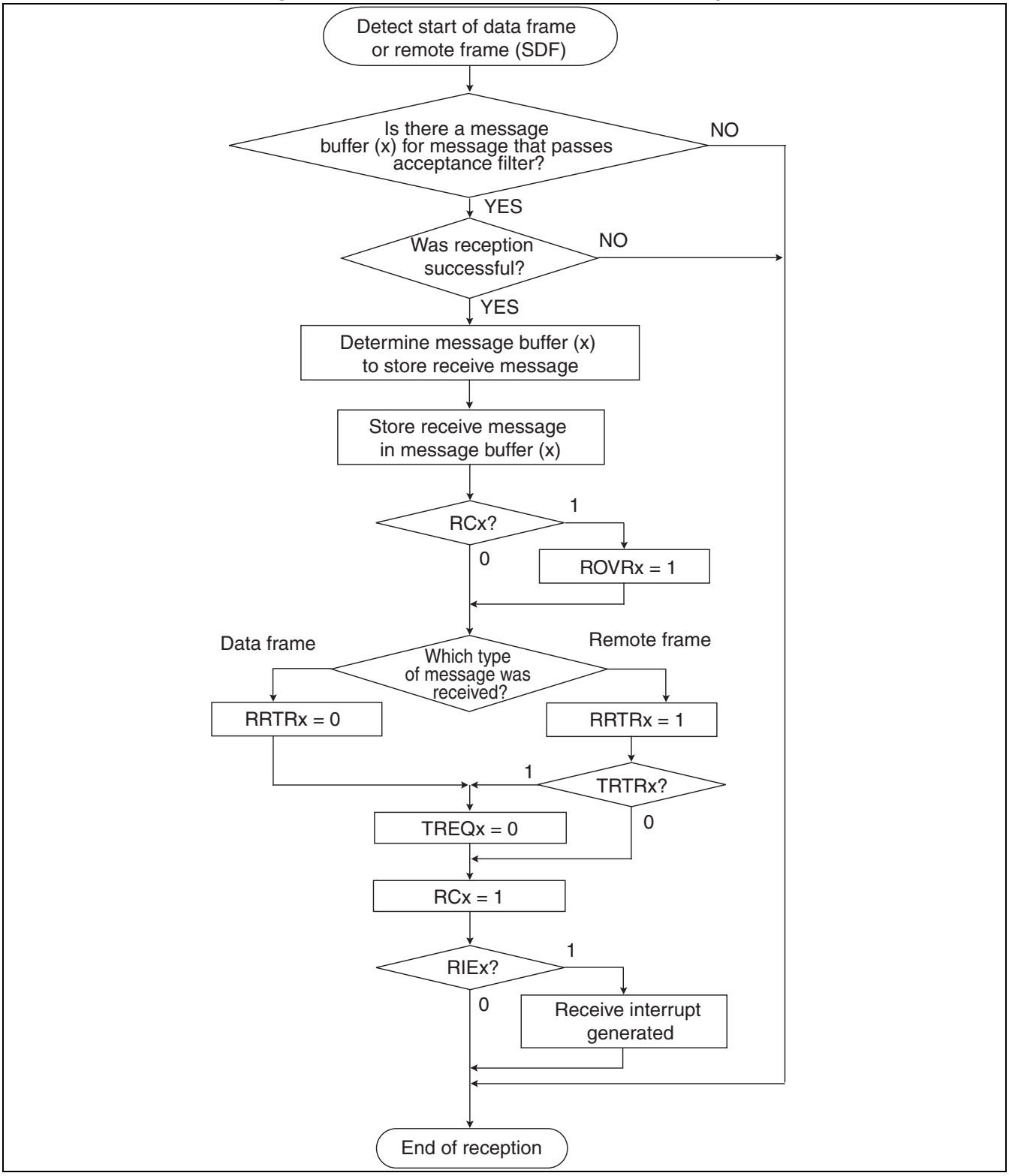
Figure 18.5-2 Flowchart of CAN Reception Setting



■ Flowchart of CAN Controller Reception

Figure 18.5-3 shows a flowchart of the CAN controller reception.

Figure 18.5-3 Flowchart of CAN Controller Reception



18.6 Using CAN Controller

Using the CAN controller requires the following settings:

- **Bit timing setting**
- **Frame format setting**
- **ID setting**
- **Acceptance filter setting**
- **Low-power consumption mode setting**

■ Setting Bit Timing

The bit timing register (BTR) must be set while bus operation is stopped (the bus operation stop bit (HALT) in the control status register (CSR) is "1").

After the setting is completed, release the bus operation stop by writing "0" to HALT.

■ Setting Frame Format

Set a frame format used for message buffer (x). To use the standard frame format, set IDEx in the IDE register (IDER) to "0". To use the extended frame format, set IDEx to "1".

This must be set while the message buffer (x) is invalid (BVALx in the message buffer valid register (BVALR) is "0"). If this is set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.

■ Setting ID

Set the ID of message buffer (x) in ID28 to ID0 of the ID register (IDRx). When the standard frame format is used, it is unnecessary to set the ID of message buffer (x) in ID17 to ID0. The ID of message buffer (x) is used as a send message during transmission, and used as an acceptance code during reception.

This must be set while the message buffer (x) is invalid (BVALx in the message buffer valid register (BVALR) is "0"). If this is set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.

■ Setting Acceptance Filter

The acceptance filter of message buffer (x) is specified via the acceptance code and acceptance mask setting. This must be set while the accepting message buffer (x) is invalid (BVALx in the message buffer valid register (BVALR) is "0"). If this is set while the buffer is valid (BVALx = 1), unnecessary receive messages may be stored.

Set the acceptance mask which was used for each message buffer (x) in the acceptance mask selection register (AMSR). The acceptance mask registers (AMR0 and AMR1) must also be specified if they are used (For details of the setting, see Sections "18.3.17 Acceptance Mask Selection Register (AMSR)" and "18.3.18 Acceptance Mask Registers 0 and 1 (AMR0/AMR1)").

The acceptance mask must be set so that transmission request will not be cleared even when an unnecessary receive message is stored. To send only messages with the same ID, for example, the acceptance mask must be set to "full-bit compare".

■ Setting Low-power Consumption Mode

To set the F^2MC-16LX to low-power consumption mode (stop, watch, etc.), write "1" to the bus operation stop bit (HALT) in the control status register (CSR) and then check that the bus operation is stopped (HALT = 1).

18.7 Procedure of Transmission via Message Buffer (x)

After completing the settings of the bit timing, frame format, ID, and acceptance filter, set BVALx to "1" to validate message buffer (x).

■ Procedure of Transmission Via Message Buffer (x)

● Setting the send data length code

Set the send data length code (in units of bytes) in DLC3 to DLC0 of the DLC register (DLCRx).

For data frame transmission (when TRTRx in the transmission RTR register (TRTRR) is "0"), specify the data length of the send message.

For remote frame transmission (when TRTRx = 1), specify the data length of the request message (in units of bytes).

Setting values other than 0000_B to 1000_B (0 to 8 bytes) is disabled.

● Setting the send data (for data frame transmission only)

For data frame transmission (when TRTRx in the transmission RTR register (TRTRR) is "0"), set the data in the data register (DTRx) as much as the number of send bytes.

The send data must be rewritten by setting the TREQx bit in the transmission request register (TREQR) to "0".

It is unnecessary to set the BVALx bit in the message buffer valid register (BVALR) to "0". Note that setting the BVALx bit to "0" may result in a loss of received remote frames.

● Setting the transmission RTR register

For data frame transmission, set TRTRx in the transmission RTR register (TRTRR) to "0".

For remote frame transmission, set TRTRx to "1".

● Setting the send start conditions (for data frame transmission only)

To start transmission immediately after a request for data frame transmission is specified, set RFWTx in the remote frame receive wait register (RFWTR) to "0" (TREQx in the transmission request register (TREQR) is "1" and TRTRx in the transmission RTR register (TRTRR) is "0").

To delay the start of transmission until a remote frame is received (RRTRx in the remote request receive register (RRTRR) becomes "1") after a request for data frame transmission is specified (TREQx = 1 and TRTRx = 0), set RFWTx to "1".

When RFWTx is set to "1", remote frame transmission is disabled.

● Setting a transmission complete interrupt

To generate a transmission complete interrupt, set TIEx in the transmission interrupt enable register (TIER) to "1".

Otherwise, set TIEx to "0".

● Setting a transmission request

To make a transmission request, set TREQx in the transmission request register (TREQR) to "1".

● Clearing a transmission request

To clear a transmission request to message buffer (x), write "1" to TCANx in the transmission cancel register (TCANR).

Check TREQx. When TREQx = 0, transmission has been cleared or completed. Check TCx in the transmission complete register (TCR). When TCx = 0, transmission has been cleared. When TCx = 1, transmission has been completed.

● Processing for transmission completion

When transmission is successful, TCx in the transmission complete register (TCR) becomes "1".

If transmission complete interrupts are enabled (TIEx in the transmission complete interrupt enable register (TIER) is "1"), an interrupt is generated.

Check that transmission is completed, and then write "0" to TCx to set it to "0". This releases the transmission complete interrupt.

The following transmission requests in wait state are cleared when a message is received or stored:

- A request for data frame transmission based on data frame reception
- A request for remote frame transmission based on data frame reception
- A request for remote frame transmission based on remote frame reception

A request for data frame transmission is not cleared when a remote frame is received or stored. However, the ID and DLC are changed with the ID and DLC of the received remote frame. Note that the ID and DLC of the data frame to be sent will be the values of the received remote frame.

18.8 Procedure of Reception Via Message Buffer (x)

After completing the settings of the bit timing, frame format, ID, and acceptance filter, make the following settings.

■ Procedure of Reception Via Message Buffer (x)

● Setting a receive interrupt

To enable receive interrupts, set RIEx in the receive interrupt enable register (RIER) to "1".

To disable receive interrupts, set RIEx to "0".

● Starting reception

To start reception after the setting, set BVALx in the message buffer valid register (BVALR) to "1" to validate message buffer (x).

● Processing for receive completion

When a message is received successfully after passing through the acceptance filter, the receive message is stored in message buffer (x), and RCx in the receive complete register (RCR) becomes "1". For data frame reception, RRTRx in the remote request receive register (RRTRR) becomes "0". For remote frame reception, RRTRx becomes "1".

When receive interrupts are enabled (RIEx in the receive interrupt enable register (RIER) is "1"), an interrupt is generated.

Check the completion of reception (RCx = 1), and then process the receive message.

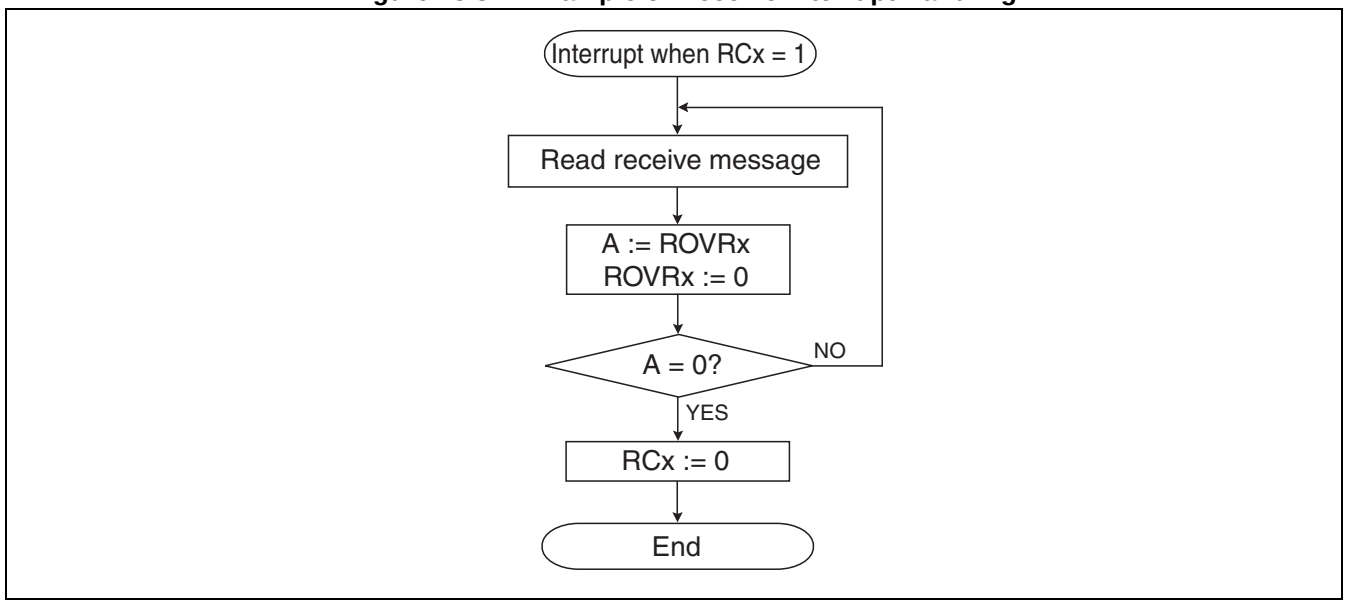
After completing the processing of the receive message, check ROVRx in the receive overrun register (ROVRR).

When ROVRx = 0, the processed receive message is valid. Write "0" to RCx to set it to "0" (the receive complete interrupt is also cleared), and finish reception.

When ROVRx = 1, a receive overrun may have occurred, which means that the processed receive message may have been overwritten with another receive message. In such a case, write "0" to the ROVRx bit to set it to "0", and then process the receive message again.

Figure 18.8-1 shows an example of receive interrupt handling.

Figure 18.8-1 Example of Receive Interrupt Handling



18.9 Specifying the Multi-level Message Buffer Configuration

When the time to process messages is insufficient, such as when reception is performed frequently or an unspecified number of messages are received, combine more than one message buffer into a multi-level message buffer to provide the CPU with some time reserve for processing receive messages.

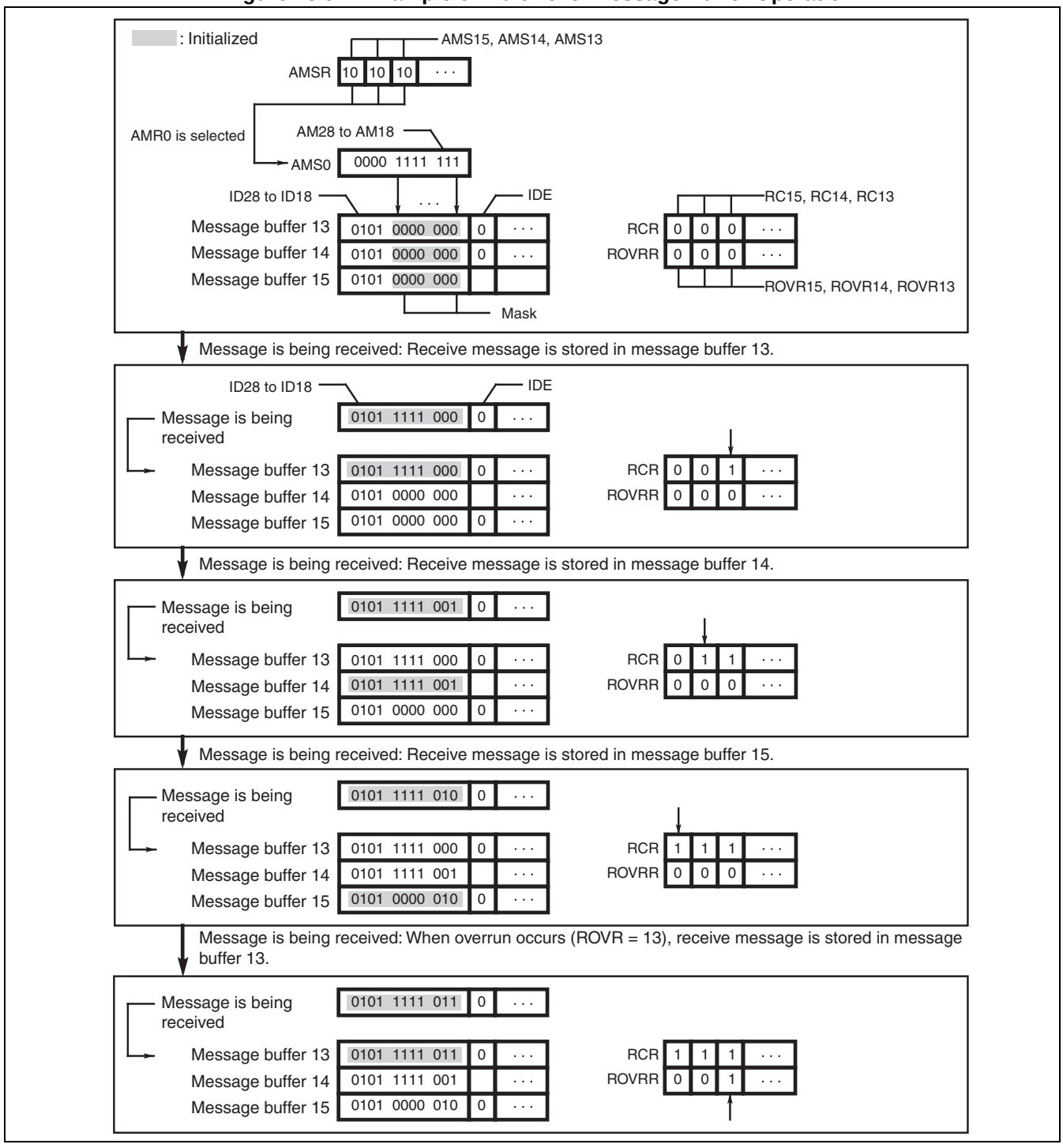
■ Specifying the Multi-level Message Buffer Configuration

To prepare a multi-level message buffer, the same acceptance filter must be set to all of the message buffers being combined.

If the bits in the acceptance mask selection register (AMSR) are set to all-bit compare [(AMSx.1, AMSx.0) = (0, 0)], the message buffers cannot be configured to be a multi-level message buffer. This is because all-bit compare stores receive messages regardless of the value of the RCx bit in the receive complete register (RCR). This means that, even if all-bit compare and the same acceptance code (ID register (IDRx)) are specified for more than one message buffer, a receive message is always stored in the message buffer having the lower number (lower priority). Therefore, you cannot specify all-bit compare and the same acceptance code to more than one message buffer.

Figure 18.9-1 shows an example of multi-level message buffer operation.

Figure 18.9-1 Example of Multi-level Message Buffer Operation



Note:

Four messages are received by message buffers 13, 14, and 15 using the same acceptance filter

setting.

18.10 CAN WAKE UP Function

The RX0 and RX2 pins are shared with INT2 pin, and the RX1 and RX3 pins are shared with INT1 pin respectively. Enabling an interrupt by INT2 and INT1 allows WAKE UP by a CAN receive operation.

■ Pins Used for CAN WAKE UP Function

Since the RX0/RX2 pins are shared with the INT2 pin, and the RX1/RX3 pins are shared with the INT1 pin respectively, enabling an interrupt by INT2 and INT1 allows the use of the WAKE UP function.

Table 18.10-1 shows the relationship among the CAN WAKE UP function, RX pins, and INT pins.

Table 18.10-1 Relationship Among CAN WAKE UP Function, RX Pins, and INT Pins

	RX pin	Interrupt Function
CAN0/CAN2	RX0/RX2	INT2
CAN1/CAN3	RX1/RX3	INT1

■ CAN WAKE UP Function

By receiving CAN data, operation can be returned from sleep mode, time-base timer mode, watch mode, or stop mode.

Note:

To use the CAN WAKE UP function, it is necessary to set external interrupts before the shift to sleep mode, time-base timer mode, watch mode, or stop mode.

18.11 Precautions When Using CAN Controller

Using the CAN controller requires the following cautions.

■ Caution for Disabling Message Buffer by BVAL Bits

When the BVAL bits are used to disable message buffers in order to read/write contents of the message buffer, the CAN controller may not perform send/receive operation properly. This section describes the workaround of this phenomenon.

● Condition

When the following two conditions are satisfied at the same time, the CAN controller may not perform send operation properly.

- The CAN controller is participating in a CAN communication. In other words, the read value of the HALT bit is "0" and the CAN controller is ready for send/receive operation by participating in a CAN bus communication.
- The BVAL bits are set to disable message buffers to read or write the contents of the message buffers.

● Workaround

Operation for suppressing transmission request

To suppress or cancel a transmission request, use the TCAN bit instead of the BVAL bit.

Operation for composing send messages

To set the ID or IDE register to compose a send message, use the BVAL bit to disable message buffers. To do this, read the transmission request bit to confirm that the bit is "0" (TREQ = 0) or check the transmission complete bit to confirm the completion of transmission (TC = 1) before setting the BVAL bit to disable the message buffer (BVAL = 0).

If transmit request was made in advance, be sure to verify that there is no pending transmit request before invaliding the message buffer.

Never write "0" to BVALx bit until you check that transmit operation is not performed.

a) Cancel the transmission request (TCANx=1;), if necessary

b) and wait for the transmission completion (while (TREQx=1;)) by polling bit or transmission complete interrupt.

Only after that the transmission buffer can be disabled (BVALx=0;).

Note:

For case a), if transmission of that buffer has already started, canceling the transmission is ignored and disabling the buffer is delayed until the end of the transmission.

18.12 Sample Program of CAN

This section shows a sample program of CAN.

■ Sample Program for CAN Transmission/Reception

● Processing specifications

Set buffer 5 in CAN0 for data frame transmission, and buffer 0 for the reception.

- Set the frame format to the standard frame format.
- ID setting: Buffer 0: ID = 0, Buffer 5: ID = 5
- Bit rate is 100 kbps (internal operation frequency: f = 16 MHz)
- Acceptance mask (full-bit compare)
- After the entry to the bus (HALT = 0), send data $A0A0_H$.
- Issue a transmission request (TREQx = 1) in the transmission complete interrupt routine and send the same data (Setting TREQx to transmission start clears the transmission complete interrupt flag.)
- In the receive interrupt routine, clear the receive interrupt flag.

[Coding example]

```
            :
            :
            :
//data format set (CAN initialize)
MOVW   IDER0, #0000H          ; Frame format setting (0: set, 1: extended)
MOVW   BTR0, #05CC7H          ; Bit rate setting to 100 kbps
                              ; (internal operation frequency: f = 16 MHz)
MOVW   BVALR0, #21H           ; Message buffer 5, 0 valid
MOVW   IDR51, #0A000H         ; Data frame 5ID setting (ID = 0005)
MOVW   IDR01, #2000H          ; Data frame 0ID setting (ID = 0001)
MOVW   ANSR00, #0000H         ; Acceptance mask selection register
                              ; (full-bit compare)

//send setting
MOVW   DLCR5, #020H           ; Transfer data length setting
                              ; (00H: 0-byte length, 08H: 8-byte length)
MOVW   RFTR0, #0000H          ; Remote frame receive wait register
MOVW   TRTR0, #0000H          ; Remote transmission request
                              ; (0: data transfer, 1: remote frame send)
MOVW   TIRER0, #0020H         ; Transmission interrupt enable register

//send setting
MOVW   RIER5, #0001H          ; Receive interrupt enable register

//bus operation start
MOVW   CSR00, #80H            ; Control status register (HALT = 0)
shift  BBS    CSR00:0, shift  ; HALT = 0  waiting

//send data setting
MOVW   DTR50, #0A0A0H         ; Write A0A0H to data register in message buffer 5
MOVW   TREQR0, #0020H         ; Transmission request register
                              ; (1: transmission start, 0: transmission stop)
```

```
            :
            :
//receive complete interrupt
CAN0RX MOVW, #0000H           ; Receive complete register
       RETI

//transmission complete interrupt
CAN0TX MOVW TREQR0, #0020H    ; Transmission request register
                              ; (1:transmission complete, 0: transmission stop)
       RETI
            :
            :
            :
```

19. LCD Controller/Driver

This chapter describes the functions and operations of the LCD controller/driver.

19.1 Overview of LCD Controller/Driver

19.2 Configuration of LCD Controller/Driver

19.3 LCD Controller/Driver Pins

19.4 Registers of LCD Controller/Driver

19.5 LCD Controller/Driver Display RAM

19.6 Operation of LCD Controller/Driver

19.1 Overview of LCD Controller/Driver

The LCD controller/driver has a built-in display data memory of 16 × 8 bits and controls the LCD display with 4 common outputs and 32 segment outputs. Three types of duty output can be selected to directly drive the LCD (Liquid Crystal Display) panel.

■ Functions of LCD Controller/Driver

The LCD controller/driver has the function that directly displays the contents of the display data memory (Display RAM) using the common and segment outputs.

- It has a built-in LCD drive voltage divide resistor. It also allows connection with an external divide resistor.
- Available up to four common outputs (COM0 to COM3) and 32 segment outputs (SEG00 to SEG31).
- It has a built-in 16-byte display data memory (display RAM).
- Selects a duty cycle of 1/2, 1/3, or 1/4 (limited by bias setting).
- Directly drives the LCD.

Table 19.1-1 shows the available combinations of bias duty.

Table 19.1-1 Combinations of Bias Duties

Bias	1/2 duty	1/3 duty	1/4 duty
1/2 bias	❍	×	×
1/3 bias	×	❍	❍

❍: Recommended mode

× : Use prohibited

Note:

Each SEG pin cannot be used as the segment output if the general-purpose port is selected in the LCRH setting.

19.2 Configuration of LCD Controller/Driver

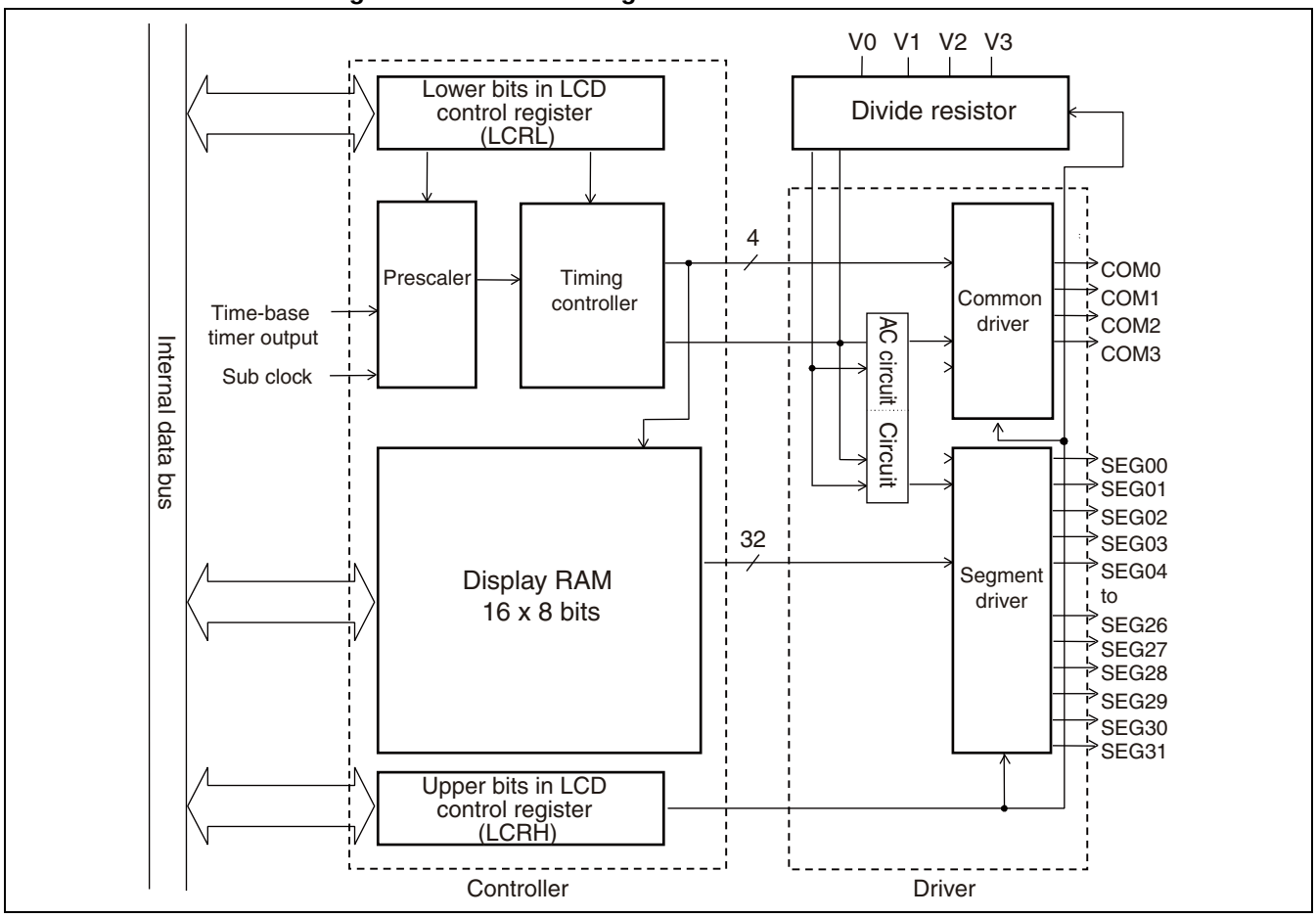
The LCD controller/driver contains the following 8 blocks. It functionally consists of two sections: the controller section where a segment and common signals are generated according to the contents of the display RAM, and the driver section, which drives the LCD.

- **LCD control register (LCRL/LCRH)**
- **Display RAM**
- **Prescaler**
- **Timing controller**
- **AC circuit**
- **Common driver**
- **Segment driver**
- **Divide resistor**

■ Block Diagram of LCD Controller/Driver

Figure 19.2-1 shows the block diagram of the LCD controller/driver.

Figure 19.2-1 Block Diagram of LCD Controller/Driver



- Lower bits in the LCD control register (LCRL)

Performs LCD drive power control, and selects display/display blanking, display mode selection, and LCD clock interval selection.

- Upper bits in the LCD control register (LCRH)

Switches between segment outputs (SEG12 to SEG23) and the general-purpose port.

- LCD output control register 1/2 (LOCR1/LOCR2)

Switches between segment outputs (SEG00 to SEG11, SEG24 to SEG31) and the general-purpose port.

- LCD output control register 3 (LOCR3)

Switches between reference power supply pins (V0 to V2) and the general-purpose port.

- Display RAM

16 × 8-bit RAM to generate a segment output signal. The RAM data is automatically read in synchronization with the selected timing of the common signal and output from the segment output pin.

- Prescaler

Generates one of four frame frequencies depending on its setting.

- Timing controller

Controls the common signal and segment signal based on the setting of the frame frequency and LCRL register.

- AC circuit

Generates the AC waveform used for driving the LCD from the timing controller signal.

- Common driver

A driver for the LCD common pin.

- Segment driver

A driver for the LCD segment pin.

- Divide resistor

Divides the LCD drive current to generate. The divide resistor can be used as external.

■ Power Supply Voltage of LCD Controller/Driver

The LCD driver's power supply voltage is determined using a built-in divide resistor or by connecting a divide resistor to pins V0 to V3.

19.2.1 Internal Divided Resistor of LCD Controller/Driver

The LCD driver's power supply voltage is generated via an external divide resistor connected to pins V0 to V3 or an internal divide resistor.

■ Internal Divided Resistor of LCD Controller/Driver

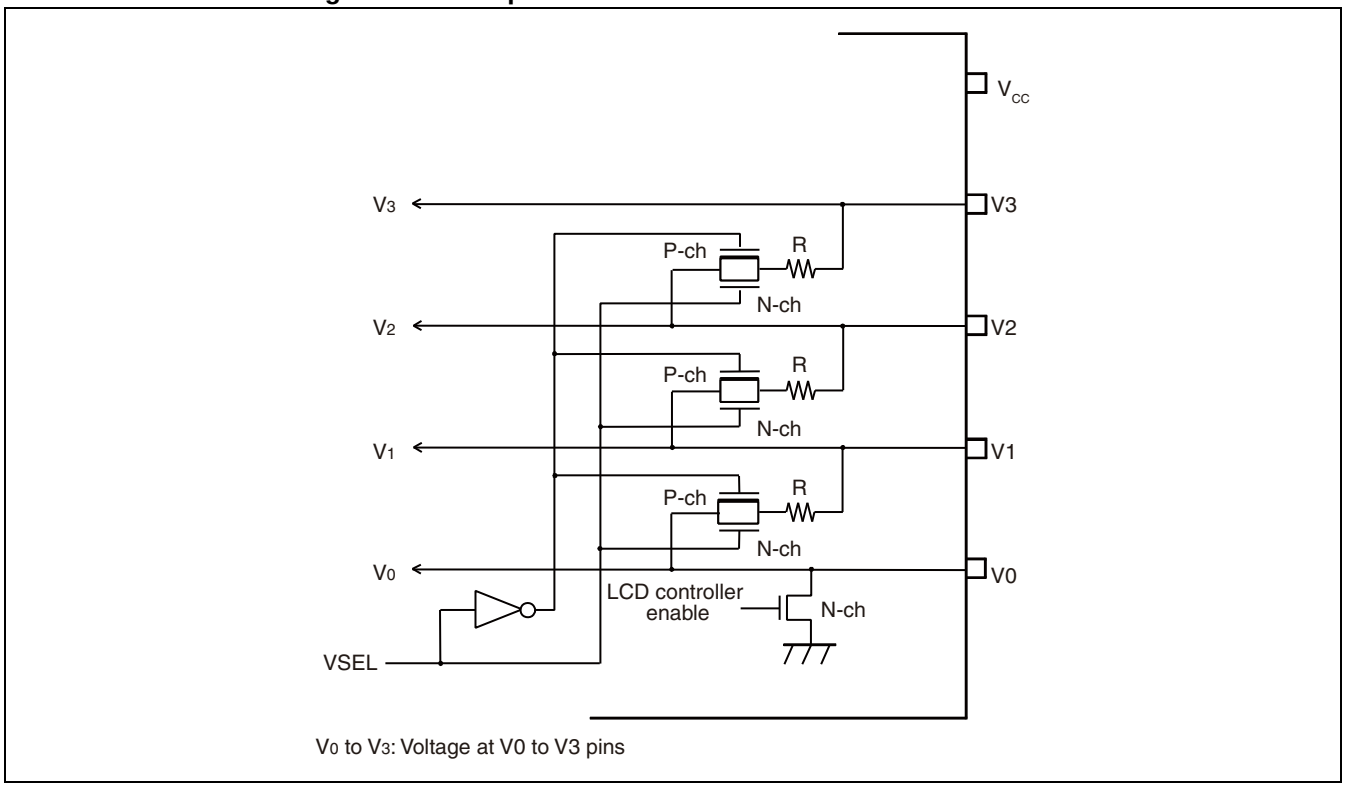
The LCD controller/driver has a built-in internal divide resistor. Alternatively, the LCD drive power supply pins (V0 to V3) can be connected to connect the external divide resistors.

The internal divide resistor and external divide resistor are selected by the LCD control register's drive power supply control bit (LCRL: VSEL). Setting the VSEL bit to "1" allows the internal divide resistor to enter the current-carrying state. Therefore, set it to "1" if the internal divide resistor is to be used instead of the external divide register.

The LCD controller enable is inactive at LCD operation stop (LCRL: MS1, MS0 = 00_B).

Figure 19.2-2 shows an equivalent circuit of the internal divide resistor.

Figure 19.2-2 Equivalent Circuit of Internal Divide Resistor

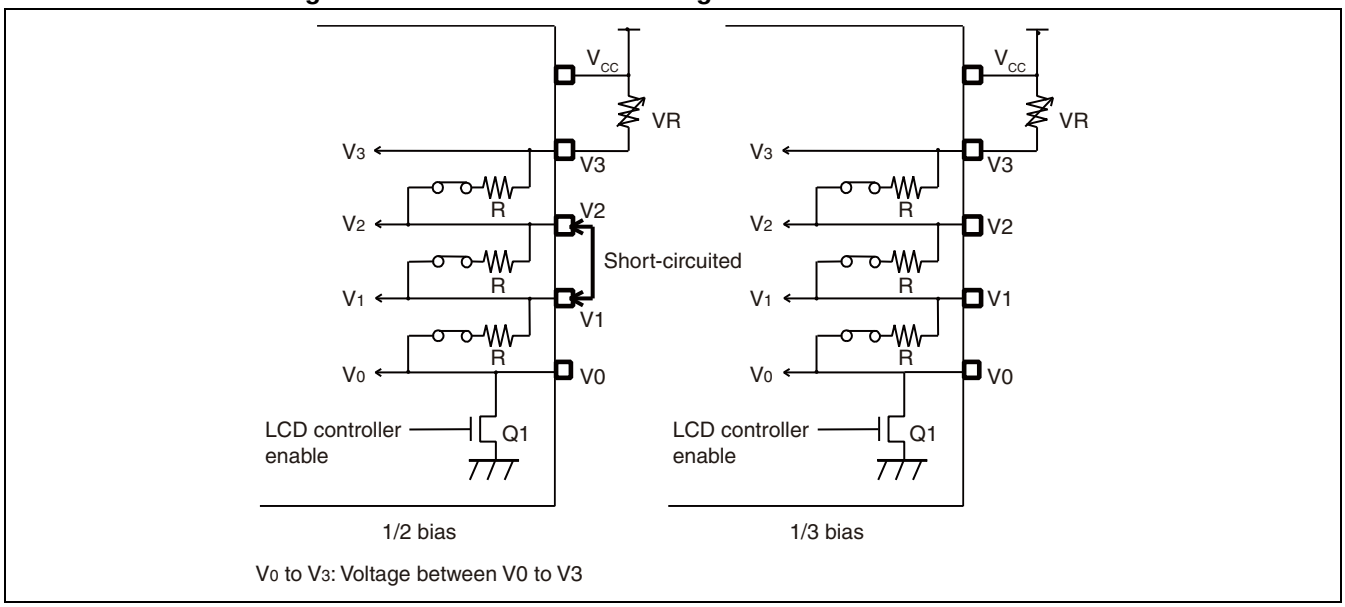


■ Using the Internal Divided Resistor

Even if the internal divide resistor is used, connect an external resistor between V_{CC} and V3. Figure 19.2-3 shows a state when using the internal divide resistor.

For the 1/2 bias setting, short-circuit between V2 and V1 pins.

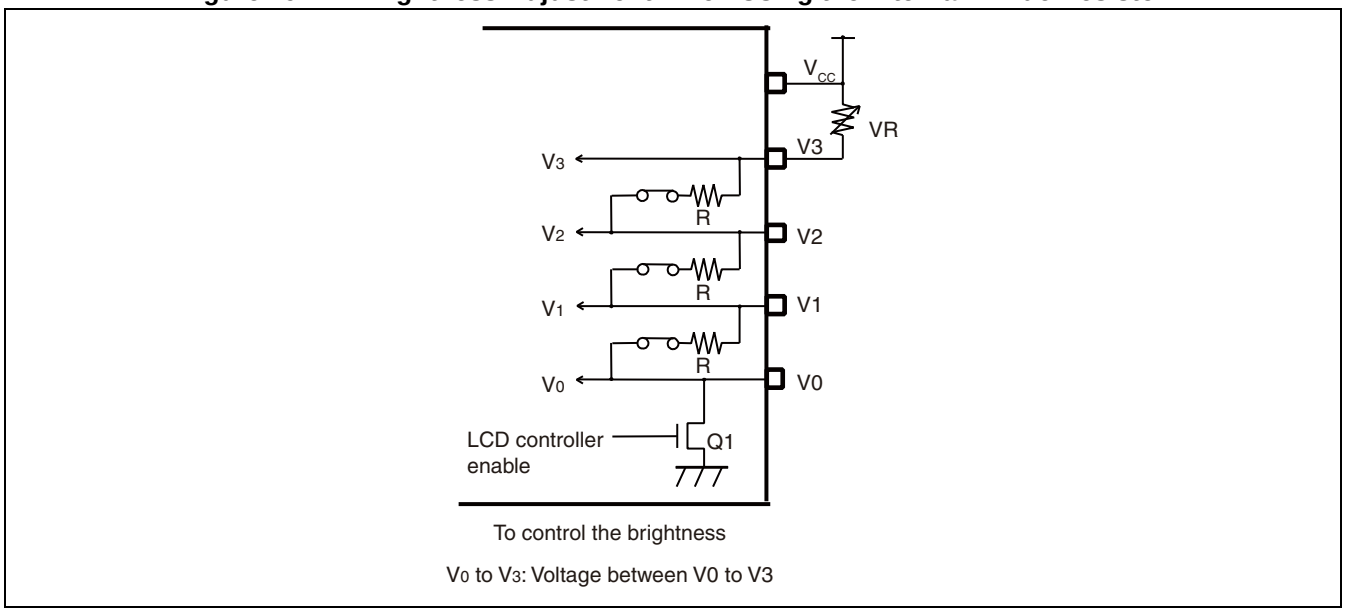
Figure 19.2-3 A State When Using the Internal Divide Resistor



■ Brightness Adjustment When Using the Internal Divided Resistor

Unless the brightness is increased by using the internal divide resistor, connect an external variable resistor (VR) between V_{CC} and V3 as shown in Figure 19.2-4 to adjust the V3 voltage.

Figure 19.2-4 Brightness Adjustment When Using the Internal Divide Resistor



19.2.2 External Divided Resistor of LCD Controller/Driver

An external divide resistor or internal divide resistor is used to generate the LCD drive voltage.
The brightness can be controlled by connecting a variable resistor between the V_{CC} and V3 pins.

■ External Divided Resistor of LCD Controller/Driver

An external divide resistor can be connected between the LCD drive power supply pins (V0 to V3). Figure 19.2-5 and Table 19.2-1 show the connection of the external divide resistor and LCD drive voltage based on the bias method.

Figure 19.2-5 Example Connection of the External Divide Resistor

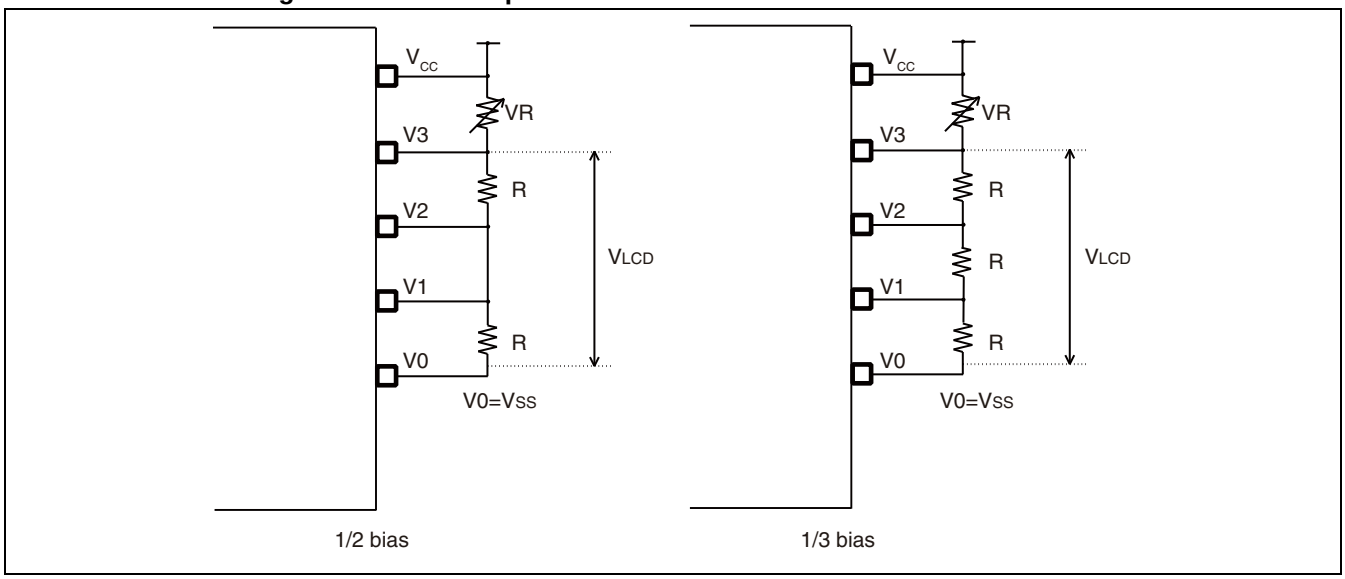


Table 19.2-1 Setting the LCD Drive Voltage

	V3	V2	V1	V0
1/2 bias	V_{LCD}	$1/2V_{LCD}$	$1/2V_{LCD}$	V_{SS}
1/3 bias	V_{LCD}	$2/3V_{LCD}$	$1/3V_{LCD}$	V_{SS}

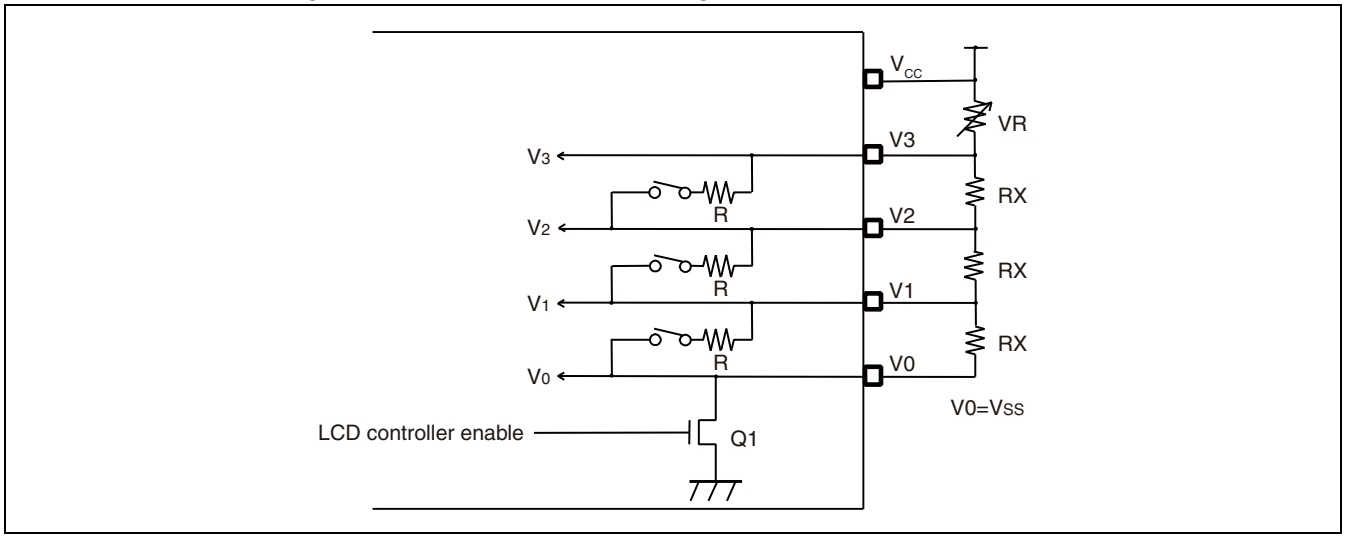
V_0 to V_3 : Voltage between V0 to V3 pins
V_{LCD} : LCD operating voltage

■ Using the External Divide Resistor

If an external divide resistor is used, the current that flows into the resistor when the LCD controller is stopped can be blocked by connecting the V_{SS} side of the divide resistor to the V0 pin only, because the V0 pin is connected to V_{SS} (GND) via an internal transistor.

Figure 19.2-6 shows the state when using an external divide resistor.

Figure 19.2-6 A State When Using an External Divide Resistor



- To connect an external resistor to inhibit the influence from the internal divide resistor, write "0" to the LCD control register's drive voltage control bit (LCRL: VSEL) to disconnect the entire internal divide resistor.
- Writing a value except 00_B to the display mode selection bits (LCRL: MS1, MS0) in the LCD control register while the internal divide resistor is disconnected, the LCDC enable transistor (Q1) is set to "ON", and the current will pass through the external divide resistor.
- On the other hand, writing 00_B to the display mode selection bits (MS1, MS0) turns the LCDC enable transistor (Q1) "OFF", and the current stops passing through the external divide resistor.

Note:

The RX value for the external resistor depends on the LCD used. Select an appropriate value.

19.3 LCD Controller/Driver Pins

The pins of the LCD controller/driver and their block diagram are shown in this section.

■ Pins of the LCD Controller/Driver

The pins of the LCD controller/driver include 4 common output pins (COM0 to COM3), 32 segment output pins (SEG00 to SEG31), and 4 LCD drive power supply pins (V0 to V3).

- COM0 to COM3 pins

COM0 to COM3 pins are used as the LCD common output pin.

- P22/SEG00 to P35/SEG11, P36/SEG12 to P43/SEG17, P44/SEG18 to P47/SEG21, P90/SEG22 to P91/SEG23, and P00/SEG24 to P07/SEG31 pins

All of pins above can function both as the general-purpose I/O port and as the LCD segment output pin. These functions are switched by setting LCRH1 or LOCR2 register.

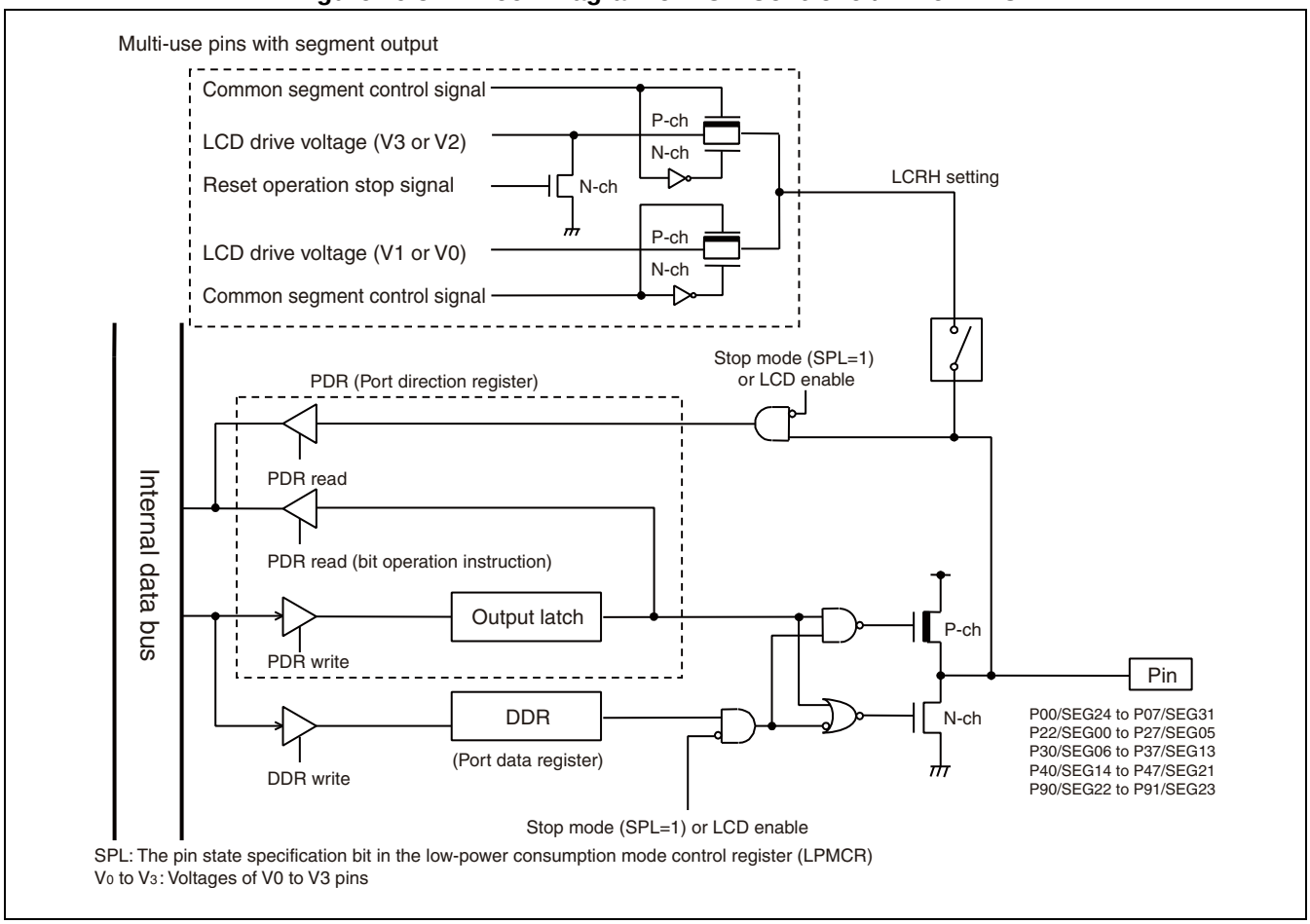
- V0 to V3 pins

The V0 to V3 pins are used as the LCD drive power supply pins (V0 to V3).

■ Block Diagram of LCD Controller/Driver Pins

Figure 19.3-1 shows a block diagram of multiplexed pins with segment output.

Figure 19.3-1 Block Diagram of LCD Controller/Driver Pins



19.4 Registers of LCD Controller/Driver

This section describes the registers of the LCD controller/driver.

■ Bit Configuration of LCD Controller/Driver Register

Figure 19.4-1 shows the bit configuration of the registers of the LCD controller/driver.

Figure 19.4-1 Bit Configuration of LCD Controller/Driver Related Register

LCRL (lower bits in the LCDC control register)

Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
$006C_H$	CSS	LCEN	VSEL	BK	MS1	MS0	FP1	FP0	00010000_B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

LCRH (upper bits in the LCDC control register)

Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
$006D_H$	Reserved	SEG5	SEG4	DTCH	SEG3	SEG2	SEG1	SEG0	00000000_B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W: Readable/Writable

19.4.1 Lower Bits in the LCD Control Register (LCRL)

The lower bits in the LCD control register (LCRL) are used to control the drive power, select display blanking, and select the display mode.

■ Bit Configuration of the Lower Bits in the LCD Control Register (LCRL)

Figure 19.4-2 shows the bit configuration of the lower bits in the LCD control register (LCRL).

Figure 19.4-2 Bit Configuration of the Lower Bits in the LCD Control Register (LCRL)

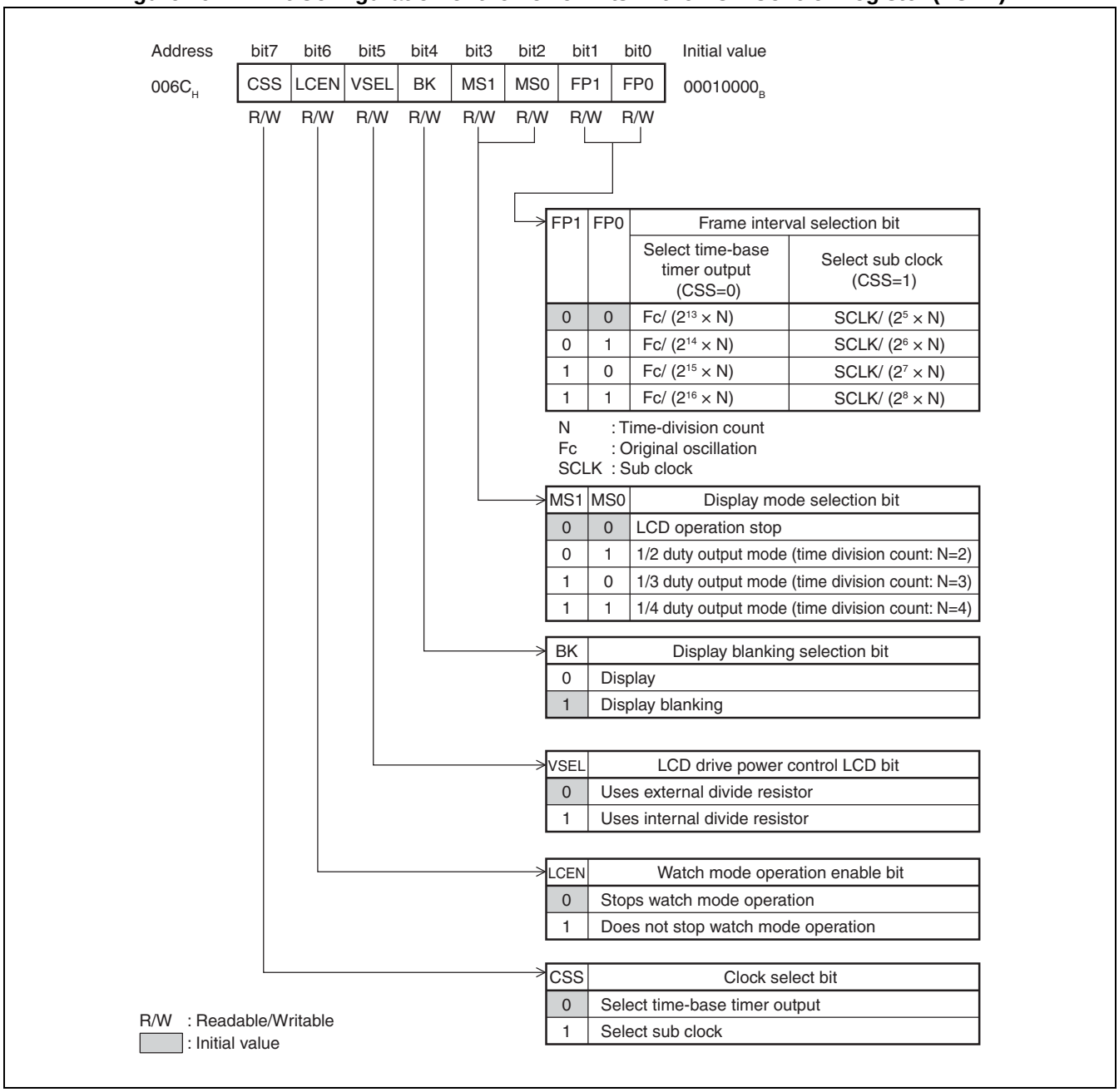


Table 19.4-1 Functions of Each Lower Bit in the LCD Control Register (LCRL) (Sheet 1 of 2)

Bit name		Function
bit7	CSS: Clock selection bit	The selection bit for the frame interval generation clock. If this bit is set to "0", the time-base timer clock output is selected. If this bit is set to "1", the sub clock is selected.
bit6	LCEN: Watch mode operation enable bit	The operation enable bit during the watch mode. In watch mode, if this bit is set to "0", the operation stops. If this bit is set to "1", operation starts.

Table 19.4-1 Functions of Each Lower Bit in the LCD Control Register (LCRL) (Sheet 2 of 2)

Bit name		Function
bit5	VSEL: LCD drive power control bit	Selects whether the electricity is turned on to the internal divide resistor. If this bit is set to "0", the internal divide resistor is cut-off. If this bit is set to "1", the electricity is turned on to the internal divide resistor. To connect an external divide resistor, this bit must be set to "0".
bit4	BK: Display blanking selection bit	Selects LCD display/nondisplay. When in the display section blanking mode (nondisplay, BK = 1), the segment output takes a non-select waveform (that doesn't meet the display conditions).
bit3, bit2	MS1, MS0: Display mode selection bit	Select one of 3 duties for the output waveform. The common pin is specified according to the selected duty output mode. If these bits are set to "0", the LCD controller/driver will stop the display operation. Note: If the selected frame interval generation clock stops due to a transition to the stop mode, the display operation should be stopped in advance.
bit1, bit0	FP1, FP0: Frame interval selection bit	Select one of 4 frame intervals for the LCD display. Note: When setting the register, calculate the optimal frame frequency for the LCD module to be used. The frame frequency is affected by the original oscillator frequency.

19.4.2 Upper Bits in the LCD Control Register (LCRH)

This register is used to switch between segment output (SEG12 to SEG23) and general-purpose port (P36, P37, P40 to P47, P90, P91).

■ Bit Configuration of the Upper Bits in the LCD Control Register (LCRH)

Figure 19.4-3 shows the bit configuration of the upper bits in the LCD control register (LCRH).

Figure 19.4-3 Bit Configuration of the Upper Bits in the LCD Control Register (LCRH)

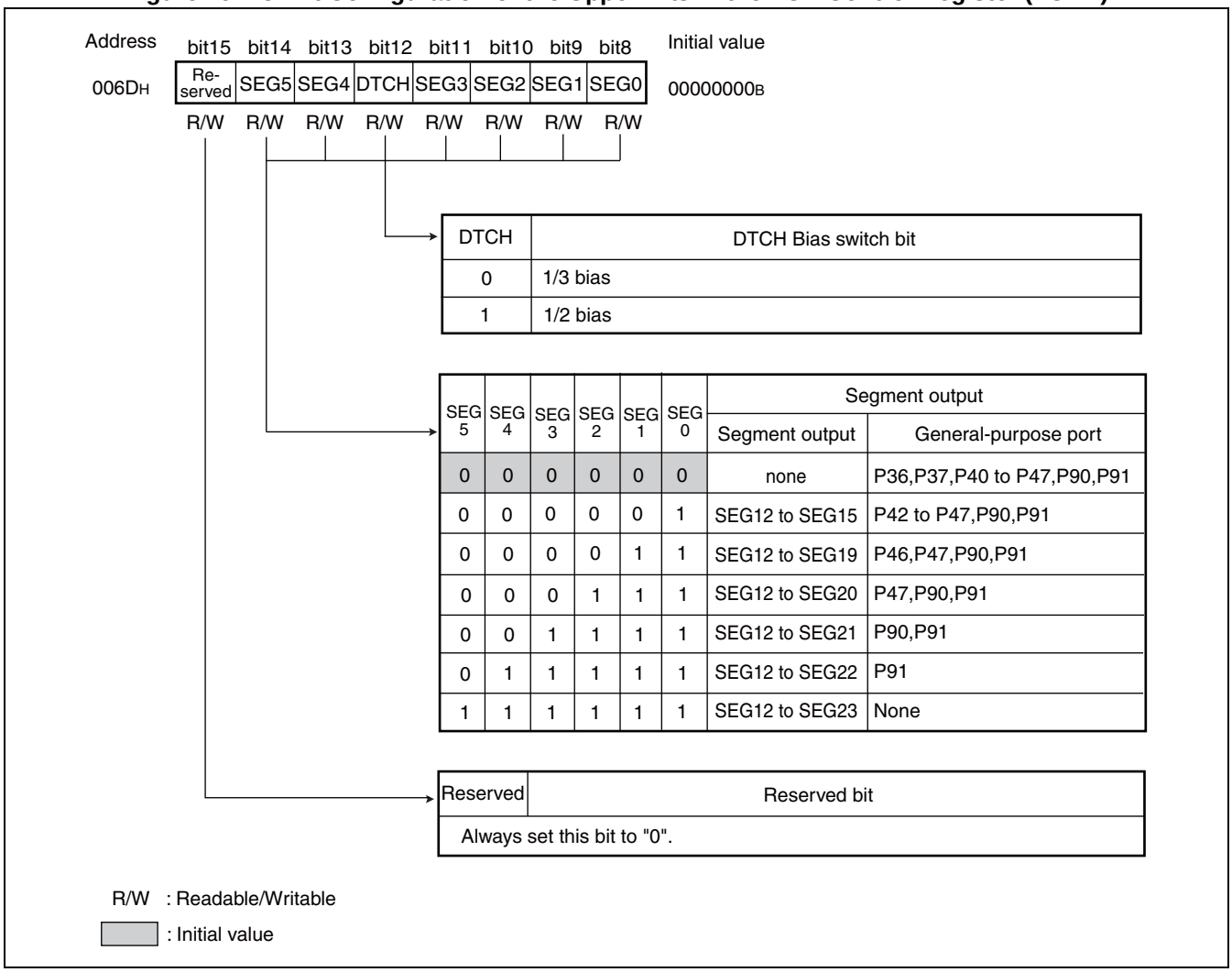


Table 19.4-2 Functions of Each Upper Bit in the LCD Control Register (LCRH)

Bit name		Function
bit15	Reserved: Reserved bit	Be sure to set this bit to "0".
bit14	SEG5: Segment pin switch bit	Switches whether to use the P91/SEG23 pin as the segment output or general-purpose port.
bit13	SEG4: Segment pin switch bit	Switches whether to use the P90/SEG22 pin as the segment output or general-purpose port.
bit12	DTCH: Bias switch bit	Bias switch bit. Please set "0" when you use 1/3 bias. Please set "1" when you use 1/2 bias.
bit11	SEG3: Segment pin switch bit	Switches whether to use the P47/SEG21 pin as the segment output or general-purpose port.
bit10	SEG2: Segment pin switch bit	Switches whether to use the P46/SEG20 pin as the segment output or general-purpose port.
bit9	SEG1: Segment pin switch bit	Switches whether to use the P42/SEG16 to P45/SEG19 pins as the segment output or general-purpose port.
bit8	SEG0: Segment pin switch bit	Switches whether to use the P36/SEG12 to P37/SEG13 and P40/SEG14 to P41/SEG15 pins as the segment output or general-purpose port.

19.4.3 LCD Output Control Register 1/2 (LOCR1/LOCR2)

These registers are used to switch between segment output (SEG00 to SEG11, SEG24 to SEG31) and general-purpose port (P22 to P27, P30 to P35, P00 to P07).

■ Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2)

Figure 19.4-4 shows the bit configuration of the LCD output control register 1/2 (LOCR1/LOCR2).

Figure 19.4-4 Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2)

LOCR1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address 0058_H	SEG10_11	SEG08_09	SEG06_07	SEG04_05	SEG03	SEG02	SEG01	SEG00
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	1	1	1	1	1	1	1	1

LOCR2	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
Address 0059_H	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Table 19.4-3 Setting and Pin Functions of LCD Output Control Register (LOCR1/LOCR2) (Sheet 1 of 2)

SEG31 to SEG24	SEG10_11 to SEG00	Segment output	
		Segment output	General-purpose port
Initial value 00000000_B	Initial value 11111111_B	SEG00 to SEG11	P00 to P07
Initial value 00000000_B	Initial value 11111110_B	SEG01 to SEG11	P00 to P07, P22
Initial value 00000000_B	Initial value 11111100_B	SEG02 to SEG11	P00 to P07, P22 to P23
Initial value 00000000_B	Initial value 11111000_B	SEG03 to SEG11	P00 to P07, P22 to P24
Initial value 00000000_B	Initial value 11110000_B	SEG04 to SEG11	P00 to P07, P22 to P25
Initial value 00000000_B	Initial value 11100000_B	SEG06 to SEG11	P00 to P07, P22 to P27
Initial value 00000000_B	Initial value 11000000_B	SEG08 to SEG11	P00 to P07, P22 to P27, P30 to P31
Initial value 00000000_B	Initial value 10000000_B	SEG10 to SEG11	P00 to P07, P22 to P27, P30 to P33
Initial value 00000000_B	Initial value 00000000_B	None	P00 to P07, P22 to P27, P30 to P35
Initial value 00000001_B	Initial value 00000000_B	SEG24	P01 to P07, P22 to P27, P30 to P35

Table 19.4-3 Setting and Pin Functions of LCD Output Control Register (LOCR1/LOCR2) (Sheet 2 of 2)

SEG31 to SEG24	SEG10_11 to SEG00	Segment output	
		Segment output	General-purpose port
Initial value 00000011_B	Initial value 00000000_B	SEG24, SEG25	P02 to P07, P22 to P27, P30 to P35
Initial value 00000111_B	Initial value 00000000_B	SEG24 to SEG26	P03 to P07, P22 to P27, P30 to P35
Initial value 00001111_B	Initial value 00000000_B	SEG24 to SEG27	P04 to P07, P22 to P27, P30 to P35
Initial value 00011111_B	Initial value 00000000_B	SEG24 to SEG28	P05 to P07, P22 to P27, P30 to P35
Initial value 00111111_B	Initial value 00000000_B	SEG24 to SEG29	P06, P07, P22 to P27, P30 to P35
Initial value 01111111_B	Initial value 00000000_B	SEG24 to SEG30	P07, P22 to P27, P30 to P35
Initial value 11111111_B	Initial value 00000000_B	SEG24 to SEG31	P22 to P27, P30 to P35

Table 19.4-4 Functions of Each Bit in the LCD Output Control Register 2 (LOCR2)

Bit name		Function
bit15	SEG31: Segment pin switch bit	Switches whether to use the P07/SEG31 pin as the segment output or general-purpose port.
bit14	SEG30: Segment pin switch bit	Switches whether to use the P06/SEG30 pin as the segment output or general-purpose port.
bit13	SEG29: Segment pin switch bit	Switches whether to use the P05/SEG29 pin as the segment output or general-purpose port.
bit12	SEG28: Segment pin switch bit	Switches whether to use the P04/SEG28 pin as the segment output or general-purpose port.
bit11	SEG27: Segment pin switch bit	Switches whether to use the P03/SEG27 pin as the segment output or general-purpose port.
bit10	SEG26: Segment pin switch bit	Switches whether to use the P02/SEG26 pin as the segment output or general-purpose port.
bit9	SEG25: Segment pin switch bit	Switches whether to use the P01/SEG25 pin as the segment output or general-purpose port.
bit8	SEG24: Segment pin switch bit	Switches whether to use the P00/SEG24 pin as the segment output or general-purpose port.

Table 19.4-5 Functions of Each Bit in the LCD Output Control Register 1 (LOCR1) (Sheet 1 of 2)

Bit name		Function
bit7	SEG10_11: Segment pin switch bit	Switches whether to use the P34/SEG10 and P35/SEG11 pins as the segment output or general-purpose port.
bit6	SEG08_09: Segment pin switch bit	Switches whether to use the P32/SEG08 and P33/SEG09 pins as the segment output or general-purpose port.
bit5	SEG06_07: Segment pin switch bit	Switches whether to use the P30/SEG06 and P31/SEG07 pins as the segment output or general-purpose port.

Table 19.4-5 Functions of Each Bit in the LCD Output Control Register 1 (LOCR1) (Sheet 2 of 2)

Bit name		Function
bit4	SEG04_05: Segment pin switch bit	Switches whether to use the P26/SEG04 and P27/SEG05 pins as the segment output or general-purpose port.
bit3	SEG03: Segment pin switch bit	Switches whether to use the P25/SEG03 pin as the segment output or general-purpose port.
bit2	SEG02: Segment pin switch bit	Switches whether to use the P24/SEG02 pin as the segment output or general-purpose port.
bit1	SEG01: Segment pin switch bit	Switches whether to use the P23/SEG01 pin as the segment output or general-purpose port.
bit0	SEG00: Segment pin switch bit	Switches whether to use the P22/SEG00 pin as the segment output or general-purpose port.

19.4.4 LCD Output Control Register 3 (LOCR3)

This register is used to switch between reference power supply pins (V0 to V2) and general-purpose port (P94 to P96) of the LCD controller/driver.

■ Bit Configuration of the LCD Output Control Register 3 (LOCR3)

Figure 19.4-5 shows the bit configuration of the LCD output control register 3 (LOCR3).

Figure 19.4-5 Bit Configuration of the LCD Output Control Register 3 (LOCR3)

LOCR3		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address	0088_H	–	–	–	–	–	V2	V1	V0
Read/Write		–	–	–	–	–	R/W	R/W	R/W
Initial value		×	×	×	×	×	1	1	1

Table 19.4-6 Functions of Each Bit in the LCD Output Control Register 3 (LOCR3)

Bit name		Function
bit7	Unused bit	Writing has no effect on operation.
bit6	Unused bit	Writing has no effect on operation.
bit5	Unused bit	Writing has no effect on operation.
bit4	Unused bit	Writing has no effect on operation.
bit3	Unused bit	Writing has no effect on operation.
bit2	V2: LCDC pin power supply switch bit	This bit switches whether to use the P96/V2 pin as the LCDC power supply pin or general-purpose port. 0: P96 1: V2
bit1	V1: LCDC pin power supply switch bit	This bit switches whether to use the P95/V1 pin as the LCDC power supply pin or general-purpose port. 0: P95 1: V1
bit0	V0: LCDC power supply pin switch bit	This bit switches whether to use the P94/V0 pin as the LCDC power supply pin or general-purpose port. 0: P94 1: V0

19.5 LCD Controller/Driver Display RAM

The display RAM is a 16 × 8 bits display data memory to generate a segment output signal.

■ Display RAM and Output Pins

The RAM data is automatically read out in synchronization with the selected timing of the common signal and output from the segment output pin.

The contents of the VRAM are output from the segment output pin at the same time of writing to the display RAM.

If the content of each bit is "1", the segment output signal is converted to the selected voltage (LCD displayed). If the content of each bit is "0", the segment output signal is converted to the nonselect voltage (LCD not displayed) and then output.

The LCD display operates independently of the CPU, therefore reading and writing of the display RAM can be performed with any timing.

Pins among Pin SEG00 to SEG31 that are not specified by the LCRH, LOCR1 and LOCR2 registers as the segment output can be used as general-purpose ports, and the corresponding RAM area can be used as normal RAM (Table 19.5-1).

Table 19.5-1 shows the relationship between the duty value, common output, and display RAM.

Figure 19.5-1 shows the relationship between display RAM, common output pins, and segment output pins.

Figure 19.5-1 Display RAM and Correspondence between Common Output Pins and Segment Output Pins

Address					
	bit3	bit2	bit1	bit0	SEG00
3960_H	bit7	bit6	bit5	bit4	SEG01
	bit11	bit10	bit9	bit8	SEG02
3961_H	bit15	bit14	bit13	bit12	SEG03
	bit3	bit2	bit1	bit0	SEG04
3962_H	bit7	bit6	bit5	bit4	SEG05
	bit11	bit10	bit9	bit8	SEG06
3963_H	bit15	bit14	bit13	bit12	SEG07
	bit3	bit2	bit1	bit0	SEG08
3964_H	bit7	bit6	bit5	bit4	SEG09
	bit11	bit10	bit9	bit8	SEG10
3965_H	bit15	bit14	bit13	bit12	SEG11
	bit3	bit2	bit1	bit0	SEG12
3966_H	bit7	bit6	bit5	bit4	SEG13
	bit11	bit10	bit9	bit8	SEG14
3967_H	bit15	bit14	bit13	bit12	SEG15
	bit3	bit2	bit1	bit0	SEG16
3968_H	bit7	bit6	bit5	bit4	SEG17
	bit11	bit10	bit9	bit8	SEG18
3969_H	bit15	bit14	bit13	bit12	SEG19
	bit3	bit2	bit1	bit0	SEG20
$396A_H$	bit7	bit6	bit5	bit4	SEG21
	bit11	bit10	bit9	bit8	SEG22
$396B_H$	bit15	bit14	bit13	bit12	SEG23
	bit3	bit2	bit1	bit0	SEG24
$396C_H$	bit7	bit6	bit5	bit4	SEG25
	bit11	bit10	bit9	bit8	SEG26
$396D_H$	bit15	bit14	bit13	bit12	SEG27
	bit3	bit2	bit1	bit0	SEG28
$396E_H$	bit7	bit6	bit5	bit4	SEG29
	bit11	bit10	bit9	bit8	SEG30
$396F_H$	bit15	bit14	bit13	bit12	SEG31
	COM3	COM2	COM1	COM0	

Table 19.5-1 Relationship between Display RAM and Segment Output Pins, and Multiplexed Pins

Value of SEG5 to SEG0 bits in the LCRH register	Segment to output	RAM area used for display	Pins used as general-purpose port
00_0000_B	None	-	P36 to P47, P90, P91
00_0001_B	SEG12 to SEG15	3966_H, 3967_H	P42 to P47, P90, P91
00_0011_B	SEG12 to SEG19	3966_H to 3969_H	P46 to P47, P90, P91
00_0111_B	SEG12 to SEG20	3966_H to $396A_H$	P47, P90, P91
00_1111_B	SEG12 to SEG21	3966_H to $396A_H$	P90, P91
01_1111_B	SEG12 to EG22	3966_H to $396B_H$	P91
11_1111_B	SEG12 to SEG23	3966_H to $396B_H$	None

Table 19.5-2 (Sheet 1 of 2)

LOCR2 register SEG31 to SEG24	LOCR1 register SEG10_11 to SEG00	Segment to output	Display RAM area	Pin used as general-purpose port
00000000_B	11111111_B	SEG00 to SEG11	3960_H to 3965_H	P00 to P07
00000000_B	11111110_B	SEG01 to SEG11	3960_H to 3965_H	P00 to P07, P22
00000000_B	11111100_B	SEG02 to SEG11	3961_H to 3965_H	P00 to P07, P22, P23
00000000_B	11111000_B	SEG03 to SEG11	3961_H to 3965_H	P00 to P07, P22 to P24
00000000_B	11110000_B	SEG04 to SEG11	3962_H to 3965_H	P00 to P07, P22 to P25
00000000_B	11100000_B	SEG06 to SEG11	396_H to 3965_H	P00 to P07, P22 to P27
00000000_B	11000000_B	SEG08 to SEG11	3964_H to 3965_H	P00 to P07, P22 to P27, P30, P31
00000000_B	10000000_B	SEG10, SEG11	3965_H	P00 to P07, P22 to P27, P30 to P33
00000000_B	00000000_B	None	-	P00 to P07, P22 to P27, P30 to P35
00000001_B	00000000_B	SEG24	$396C_H$	P01 to P07, P22 to P27, P30 to P35
00000011_B	00000000_B	SEG24, SEG25	$396C_H$	P02 to P07, P22 to P27, P30 to P35
00000111_B	00000000_B	SEG24 to SEG26	$396C_H$, $396D_H$	P03 to P07, P22 to P27, P30 to P35
00001111_B	00000000_B	SEG24 to SEG27	$396C_H$, $396D_H$	P04 to P07, P22 to P27, P30 to P35
00011111_B	00000000_B	SEG24 to SEG28	$396C_H$ to $396E_H$	P05 to P07, P22 to P27, P30 to P35
00111111_B	00000000_B	SEG24 to SEG29	$396C_H$ to $396E_H$	P06, P07, P22 to P27, P30 to P35

Table 19.5-2 (Sheet 2 of 2)

LOCR2 register SEG31 to SEG24	LOCR1 register SEG10_11 to SEG00	Segment to output	Display RAM area	Pin used as general-purpose port
01111111_B	00000000_B	SEG24 to SEG30	$396C_H$ to $396F_H$	P07, P22 to P27, P30 to P35
11111111_B	00000000_B	SEG24 to SEG31	$396C_H$ to $396F_H$	P22 to P27, P30 to P35

Note: RAM area which is not used as display allows to be used as the normal RAM. The byte access is only available.

Table 19.5-3 Relationship between Duty, Common Output, and Bits Used as Display RAM

Duty setting value	Common output	Bits used for each display data							
		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1/2	COM0, COM1 (2)	-	-	❍	❍	-	-	❍	❍
1/3	COM0 to COM2 (3)	-	❍	❍	❍	-	❍	❍	❍
1/4	COM0 to COM3 (4)	❍	❍	❍	❍	❍	❍	❍	❍

❍: Used

- : Not used

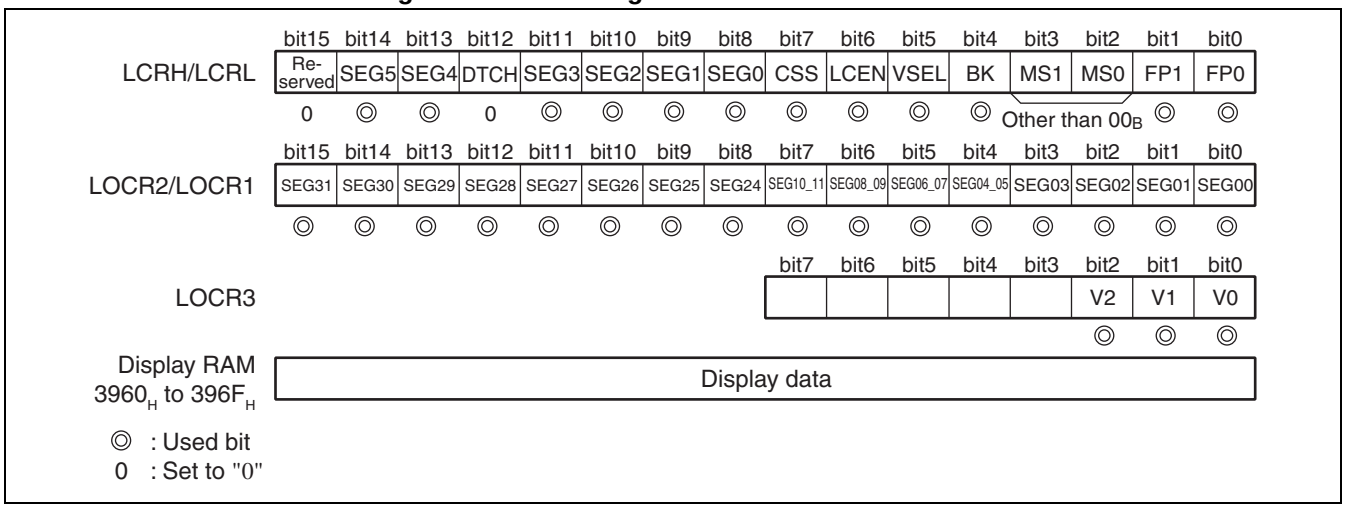
19.6 Operation of LCD Controller/Driver

The LCD controller/driver controls and drives the LCD display.

■ Operation of LCD Controller/Driver

The settings shown in Figure 19.6-1 are required for the LCD display.

Figure 19.6-1 Settings of LCD Controller/Driver



When you setup the LCD controller/driver as above, and the selected frame interval generation clock is oscillating, the LCD panel's drive waveform is output to the common/segment output pins (COM0 to COM3, SEG00 to SEG31) according to the display RAM.

The frame interval generation clock can be switched even during LCD display operation. However, the display may flicker in the event of switching, therefore, stop the display temporarily with blanking (LCRL: BK = 1) before switching the clock.

The display drive output is a two-frame AC waveform selected by the bias and duty settings.

COM2/COM3 pin output in case of a duty setting of 1/2 takes a non-select level waveform. Similarly, the COM3 pin output in case of a duty setting of 1/3 takes a nonselect level waveform.

If the LCD display operation stops (LCRL: MS1, MS0 = 00_B) or is being reset, both common/segment output pins enter "L" level.

Note:

If the frame interval generation clock stops during the LCD display operation, the AC circuit will stop, and then the current is directly applied to the LCD element. In this case, the LCD display operation must be stopped in advance. The conditions under which the original oscillator clock stops depends on the standby mode selection.

■ Drive Waveform of the LCD

If the LCD is driven with the DC, the LCD element, by its nature, undergoes a chemical change causing a deterioration of the element. Therefore, the LCD controller/driver has a built-in AC circuit to drive the LCD with a two-frame AC waveform. There are three types of output waveform:

- 1/2 bias, 1/2 duty output waveform
- 1/3 bias, 1/3 duty output waveform
- 1/3 bias, 1/4 duty output waveform

19.6.1 Output Waveform during the LCD Controller/Driver Operation (1/2 Duty)

The display drive output is a 2-frame AC waveform of the multiplex drive method. Only COM0 and COM1 are used for display while 1/2 duty. COM2 and COM3 are not used.

■ 1/2 Bias, 1/2 Duty Output Waveform

The LCD element is turned ON for which the potential difference between the common output and segment output is greatest.

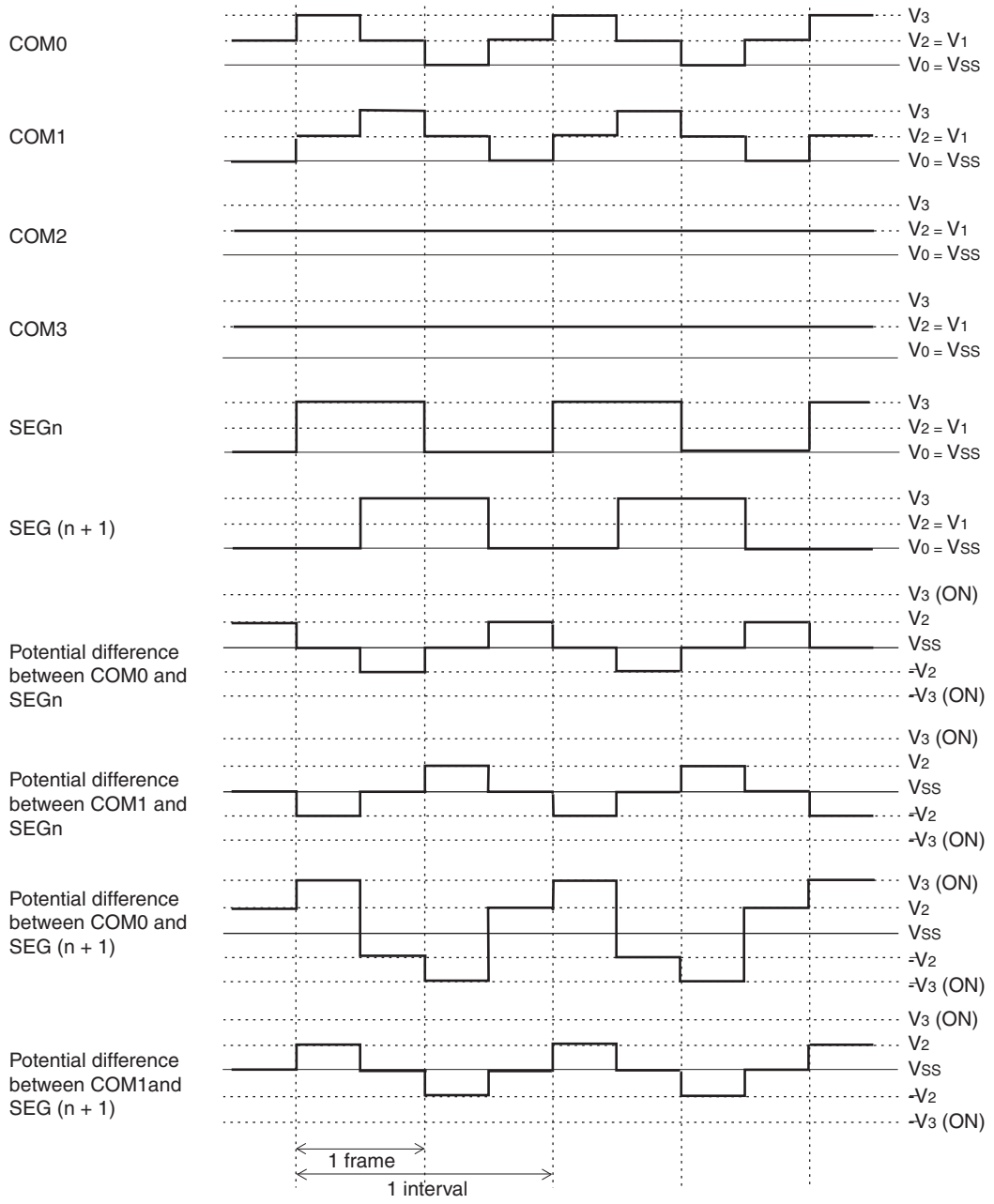
The output waveform when the contents of the display RAM is as shown in Table 19.6-1 is shown in Figure 19.6-2.

Table 19.6-1 Example of the Display RAM Contents

Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	-	-	0	0
SEG (n+1)	-	-	0	1

-: Not used

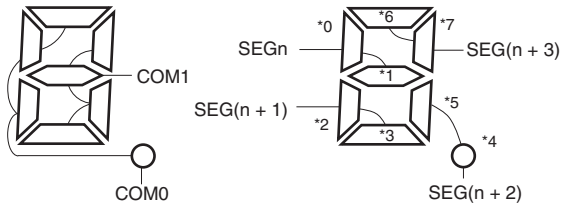
Figure 19.6-2 Example of 1/2 Bias, 1/2 Duty Output Waveform



V0 to V3: Voltages of V0 to V3 pins

■ Example of LCD Panel Connection and Display Data (1/2 Duty Drive Method)

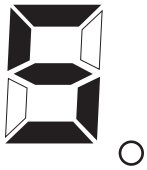
Figure 19.6-3 Example of LCD Panel Display Data



Address	COM3	COM2	COM1	COM0	
m_H	bit3	bit2	bit1*1	bit0*0	SEGn
	bit7	bit6	bit5*3	bit4*2	SEG(n + 1)
$(m + 1)_H$	bit3	bit2	bit1*5	bit0*4	SEG(n + 2)
	bit7	bit6	bit5*7	bit4*6	SEG(n + 3)

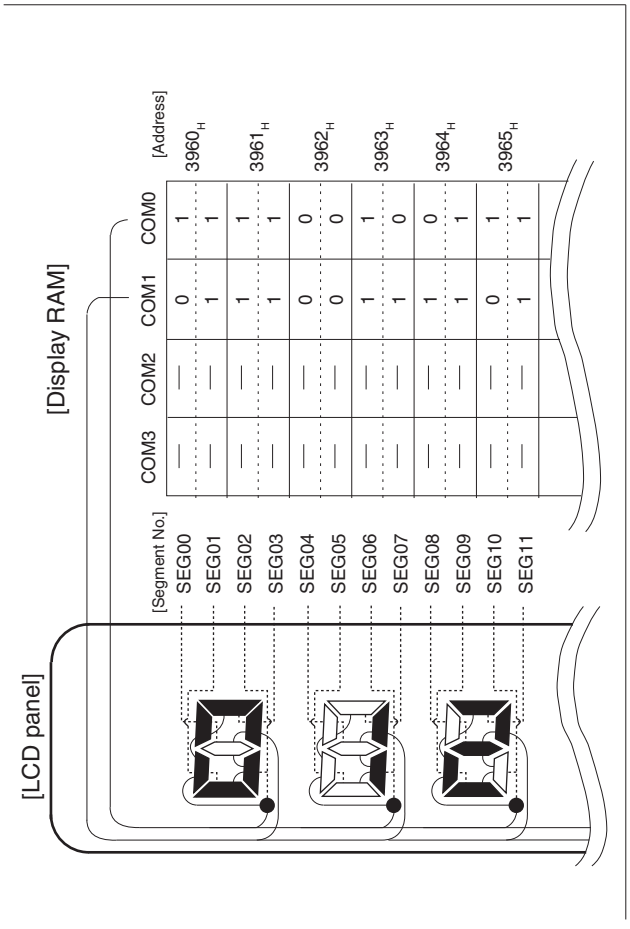
*0 to *7: Indicates the correspondence with the display RAM.
Bit2, bit3, bit6, and bit7 are not used.

e.g.) When displaying 5



Address	COM3	COM2	COM1	COM0	
3960_H	—	—	1	1	SEGn
	—	—	1	0	SEG1
3961_H	—	—	1	0	SEG2
	—	—	0	1	SEG3

0: OFF
1: ON



LCD display	Example data corresponding to number 0 to 9							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	—	—	1	1	—	—	0	1
	—	—	1	1	—	—	1	1
1	—	—	0	0	—	—	0	0
	—	—	1	0	—	—	1	1
2	—	—	1	1	—	—	1	0
	—	—	1	1	—	—	0	1
3	—	—	1	0	—	—	1	0
	—	—	1	1	—	—	1	1
4	—	—	0	0	—	—	1	1
	—	—	1	0	—	—	1	1
5	—	—	1	0	—	—	1	1
	—	—	0	1	—	—	1	1
6	—	—	1	1	—	—	1	1
	—	—	0	1	—	—	1	1
7	—	—	0	0	—	—	0	1
	—	—	1	1	—	—	1	1
8	—	—	1	1	—	—	1	1
	—	—	1	1	—	—	1	1
9	—	—	1	0	—	—	1	1
	—	—	1	1	—	—	1	1

19.6.2 Output Waveform during the LCD Controller/Driver Operation (1/3 Duty)

COM0, COM1, and COM2 are used for display while 1/3 duty. COM3 is not used.

■ 1/3 Bias, 1/3 Duty Output Waveform

The LCD element is turned ON for which the potential difference between the common output and segment output is greatest.

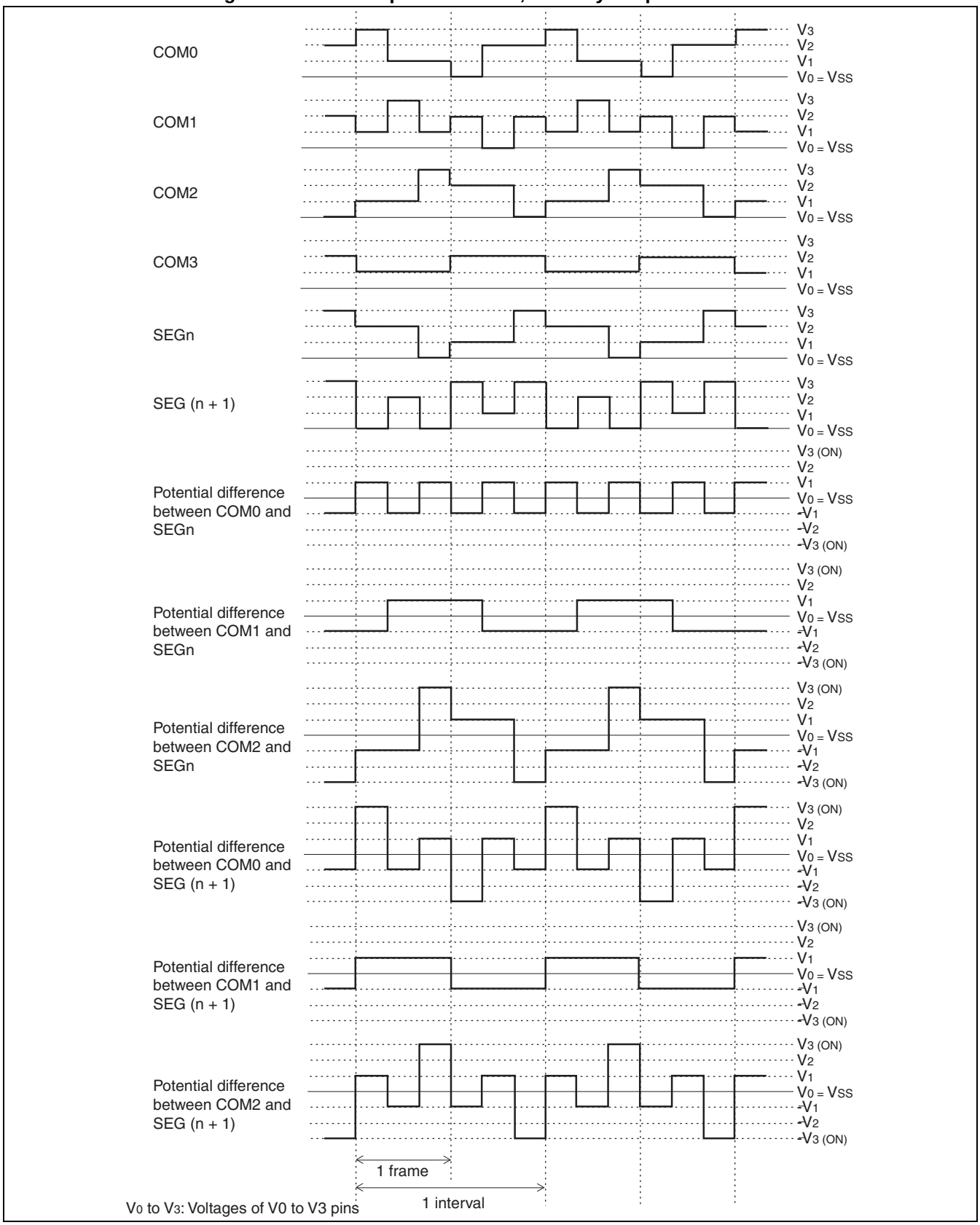
The output waveform corresponding to the display RAM contents shown in Table 19.6-2 is illustrated in Figure 19.6-4.

Table 19.6-2 Example of the Display RAM Contents

Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	-	1	0	0
SEG (n+1)	-	1	0	1

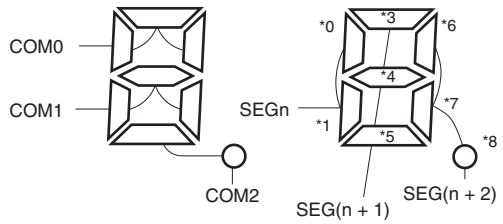
-: Not used

Figure 19.6-4 Example of 1/3 Bias, 1/3 Duty Output Waveform



■ Example of LCD Panel Connection and Display Data (1/3 Duty Drive Method)

Figure 19.6-5 Example of LCD Panel Display Data



Address	COM3	COM2	COM1	COM0	
m H	bit3	bit2*2	bit1*1	bit0*0	SEGn
	bit7	bit6*5	bit5*4	bit4*3	SEG(n + 1)
(m + 1) H	bit3	bit2*8	bit1*7	bit0*6	SEG(n + 2)

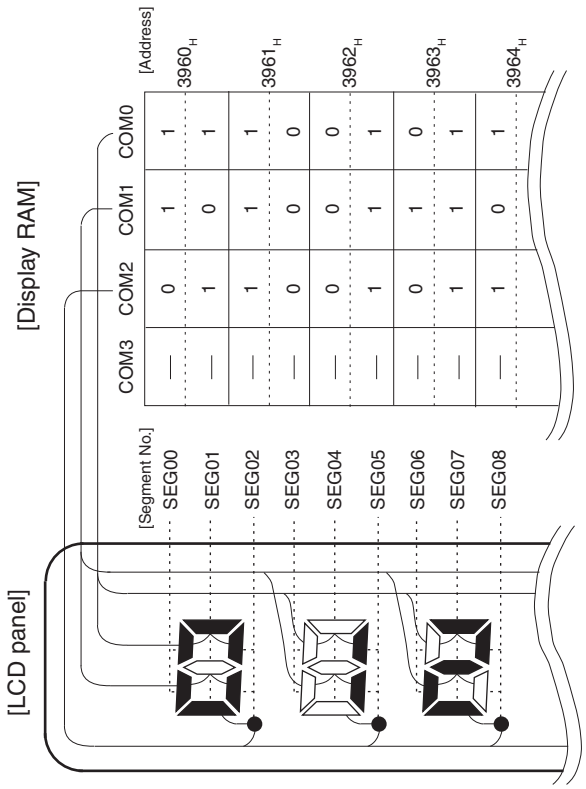
*0 to *8: Indicates the correspondence with the display RAM.
Bit3, bit7, and *2 are not used.

e.g.) When displaying 5



Address	COM3	COM2	COM1	COM0		
3960H	—	0	0	1	SEGn	When starting with bit0
	—	1	1	1	SEG1	
3961H	—	0	1	0	SEG2	
	—	0	0	1	SEG3	When starting with bit4
3962H	—	1	1	1	SEG4	
	—	0	1	0	SEG5	

0: OFF
1: ON



LCD display	Example data corresponding to number 0 to 9							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	—	1	0	1	—	0	1	1
	—	0	1	1	—	1	1	1
	—	1	1	1	—	1	0	1
1	—	0	0	0	—	0	0	0
	—	0	0	0	—	1	1	1
	—	1	1	1	—	0	0	0
2	—	1	1	1	—	0	1	0
	—	0	1	0	—	1	0	1
	—	1	0	1	—	1	1	1
3	—	1	1	1	—	0	0	0
	—	0	0	0	—	1	1	1
	—	1	1	1	—	1	1	1
4	—	0	1	0	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	0	1	0
5	—	1	1	1	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	0	—	1	1	1
6	—	1	1	1	—	0	1	1
	—	0	1	1	—	1	1	0
	—	1	1	0	—	1	1	1
7	—	0	0	1	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	0	0	1
8	—	1	1	1	—	0	1	1
	—	0	1	1	—	1	1	1
	—	1	1	1	—	1	1	1
9	—	1	1	1	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	1	1	1

: Data when starting with bit4
: Data when starting with bit0

As the 1/3 duty uses 3 bytes to display 2 digits, 2 types of data array are provided: starting with the first byte of bit0 and starting with the second byte of bit4.

19.6.3 Output Waveform during the LCD Controller/Driver Operation (1/4 Duty)

COM0, COM1, COM2, and COM3 are all used for display while 1/4 duty.

■ 1/3 Bias, 1/4 Duty Output Waveform

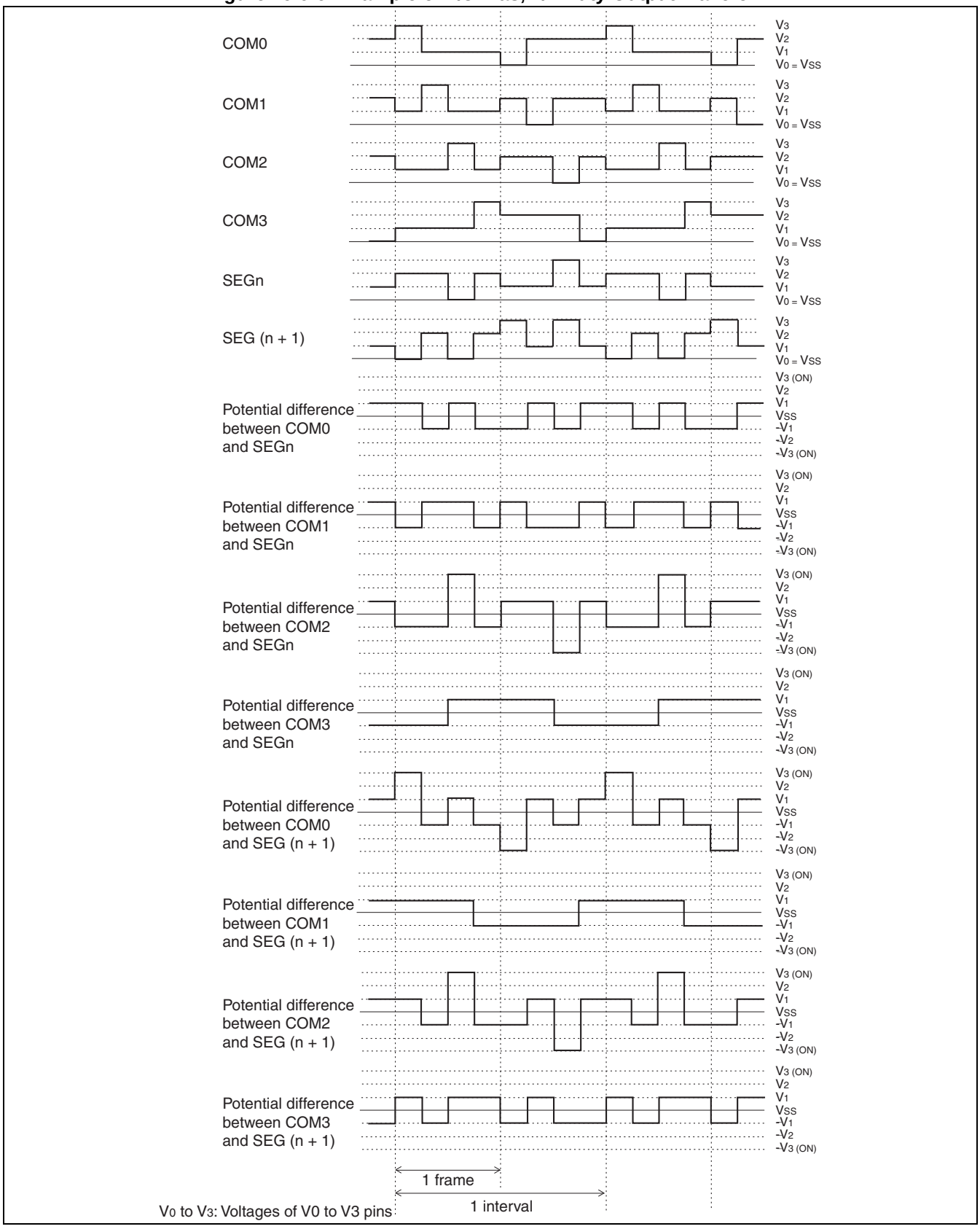
The LCD element is turned ON for which the potential difference between the common output and segment output is greatest.

The output waveform corresponding to the display RAM contents shown in Table 19.6-3 is illustrated in Figure 19.6-6.

Table 19.6-3 Example of the Display RAM Contents

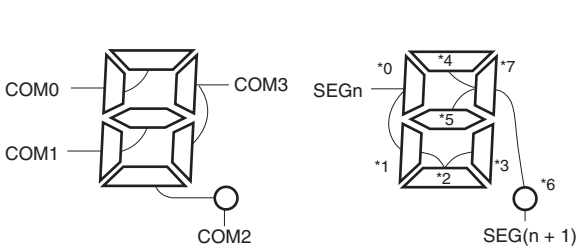
Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	0	1	0	0
SEG (n+1)	0	1	0	1

Figure 19.6-6 Example of 1/3 Bias, 1/4 Duty Output Waveform

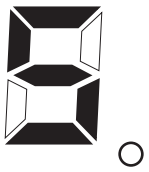


■ Example of LCD Panel Connection and Display Data (1/4 Duty Drive Method)

Figure 19.6-7 Example of LCD Panel Display Data



e.g.) When displaying 5

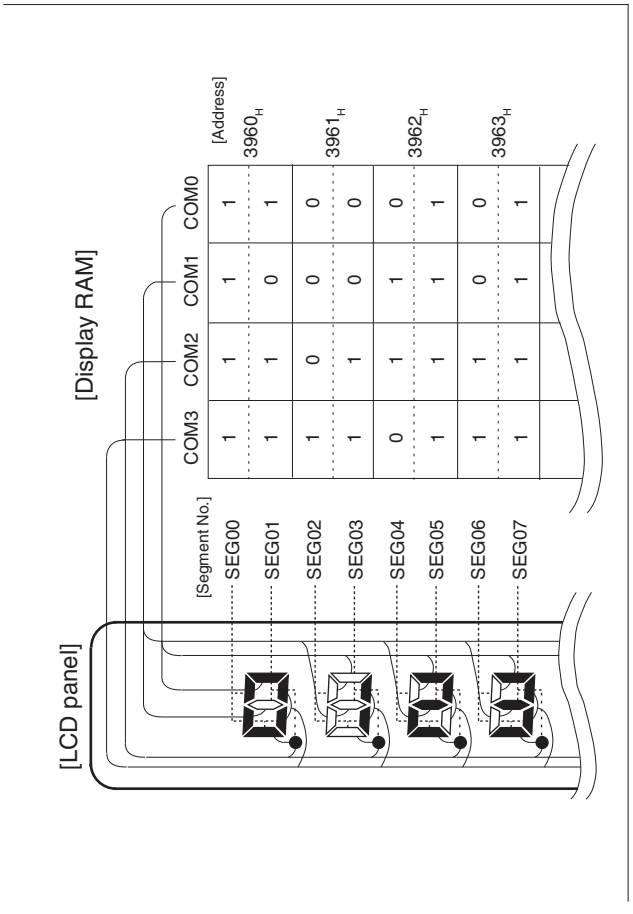


Address	COM3	COM2	COM1	COM0	
m_H	bit3*3	bit2*2	bit1*1	bit0*0	SEGn
	bit7*7	bit6*6	bit5*5	bit4*4	SEG(n + 1)

*0 to *7: Indicates the correspondence with the display RAM.

Address	COM3	COM2	COM1	COM0	
3960_H	1	1	0	1	SEGn
	0	0	1	1	SEG1

0: OFF
1: ON



LCD display	Example data corresponding to number 0 to 9							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0.	1	1	0	1	1	1	1	1
1.	1	1	0	0	1	0	0	0
2.	1	1	1	1	0	1	1	0
3.	1	1	1	1	1	1	0	0
4.	1	1	1	0	1	0	0	1
5.	0	1	1	1	1	1	0	1
6.	0	1	1	1	1	1	1	1
7.	1	1	0	1	1	0	0	1
8.	1	1	1	1	1	1	1	1
9.	1	1	1	1	1	1	0	1

20. Low-Voltage/CPU Operation Detection Reset Circuit

This chapter describes the functions and operations of the low-voltage/CPU operation detection reset circuit.

20.1 Overview of the Low-voltage/CPU Operation Detection Reset

20.2 Configuration of the Low-Voltage/CPU Operation Detection

20.3 Register of the Low-voltage/CPU Operation Detection Reset Circuit

20.4 Operation of the Low-voltage/CPU Operation Detection Reset Circuit

20.5 Notes on Using the Low-voltage/CPU Operation Detection Reset Circuit

20.6 Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit

20.1 Overview of the Low-voltage/CPU Operation Detection Reset

The low-voltage detection reset circuit has the function to monitor the supply voltage and detect the voltage drop. If it detects a low voltage, an internal reset is generated. On the other hand, the CPU operation detection reset circuit is a 20-bit counter that uses the original oscillator as a count clock. If it is not cleared within a specified time after it has started, an internal reset is generated.

■ Low-voltage Detection Reset Circuit

Table 20.1-1 shows the detection voltage of the low-voltage/CPU operation detection reset circuit.

Table 20.1-1 Detection Voltage of the Low-voltage/CPU Operation Detection Reset Circuit

	Detection voltage
Flash Memory/Mask ROM products	4.2V ±0.2V
EVA product	4.0V ±0.3V *

If a low-voltage is detected, the low-voltage detection flag (LVRC: LVRF) is set to "1", and an internal reset is output.

As the operation continues even in STOP mode, a low-voltage detection causes to generate an internal reset and to clear the STOP mode.

In an internal RAM write operation, a low-voltage reset is generated only after the write operation has been completed.

*: For EVA product, the internal reset is not generated even when a low-voltage is detected.

In this case, low-voltage detection flag bit (LVRF) in the low-voltage/CPU operation detection reset control register (LVRC) is set, however, reset source bit in the watchdog timer control register (WDTC) is not set.

■ CPU Operation Detection Reset Circuit

The CPU operation detection reset circuit is a counter to prevent the program from running out of control. This circuit starts automatically after the power is turned on. Once started, the counter of the circuit must be cleared regularly within a specified time. If the program fails to clear the counter within the specified time due to a problem, such as an infinite loop, an internal reset is performed. The CPU operation detection circuit generates an internal reset with a length of 5 machine cycles.

Table 20.1-2 Interval Time of the CPU Operation Detection Reset Circuit

Interval time
2^{20}/Fc (about 262 ms)

Note: The interval time is indicated in the parentheses when the oscillator clock operates at 4 MHz.

In any mode in which the CPU stops, this circuit also stops.

The counter clearing conditions of this circuit are listed below:

- Writing "0" the CL bit in the LVRC register
- Internal reset
- Stop of oscillator clock
- Transition to sleep mode
- Transition to time-base timer mode or watch mode
- Power-on reset

Note: For EVA product, the internal reset is not generated unless the counter of the CPU operation detection reset circuit is cleared for a given length of time.
In this case, the CPU operation detection flag bit (CPUF) in the low-voltage/CPU operation detection reset control register (LVRC) is set, however, reset source bit in the watchdog timer control register (WDTC) is not set.

20.2 Configuration of the Low-Voltage/CPU Operation Detection

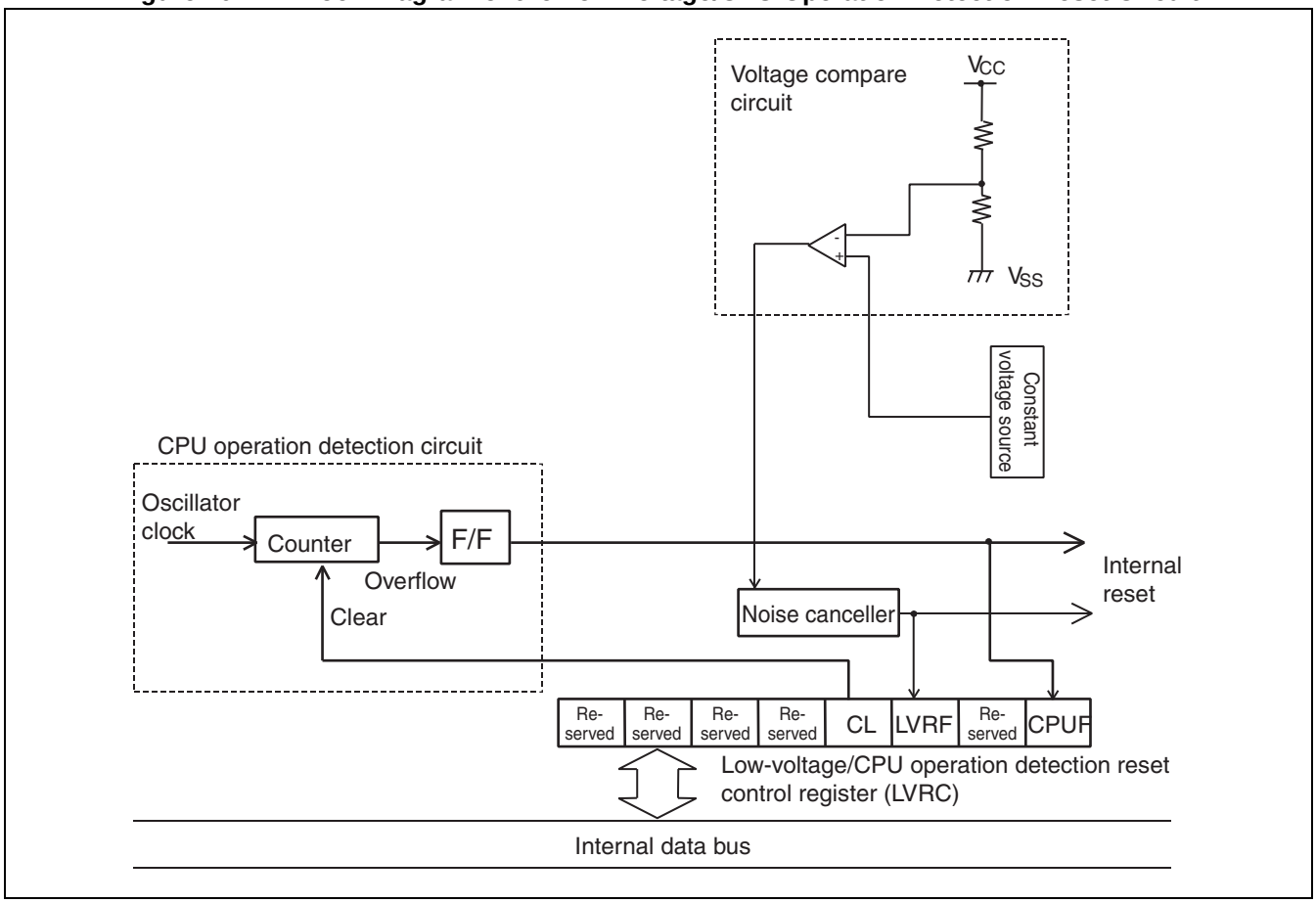
The low-voltage/CPU operation detection reset circuit consists of 3 blocks:

- **CPU operation detection circuit**
- **Voltage compare circuit**
- **Low-voltage/CPU operation detection reset control register (LVRC)**

■ Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit

Figure 20.2-1 shows the block diagram of the low-voltage/CPU operation detection reset circuit.

Figure 20.2-1 Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit



- CPU operation detection circuit

A counter used to prevent the program from running out of control. Once started, the counter of the circuit must be cleared regularly within a specified time.

- Voltage compare circuit

Compares the detection voltage with the power supply voltage, and if it detects a low-voltage, outputs the "H" level.

After power-up, it operates continuously.

- Low-voltage/CPU operation detection reset control register (LVRC)

Contains flags for low-voltage/CPU operation detection reset and is used to clear the counter for the CPU operation detection function.

- Reset sources for the low-voltage/CPU operation detection reset circuit

If the power supply voltage falls below the detection voltage, an internal reset occurs.

If the counter of the CPU operation detection circuit is not cleared within a specified time, an internal reset occurs.

20.3 Register of the Low-voltage/CPU Operation Detection Reset Circuit

The low-voltage/CPU operation detection reset control register (LVRC) clears the counters of the low-voltage/CPU operation detection reset flag and the CPU operation detection circuit. This register accepts only byte access. Do not access the register in words (whether to read from or write to it).

■ Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC)

Figure 20.3-1 shows the bit configuration of the low-voltage/CPU operation detection reset control register (LVRC).

Figure 20.3-1 Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC)

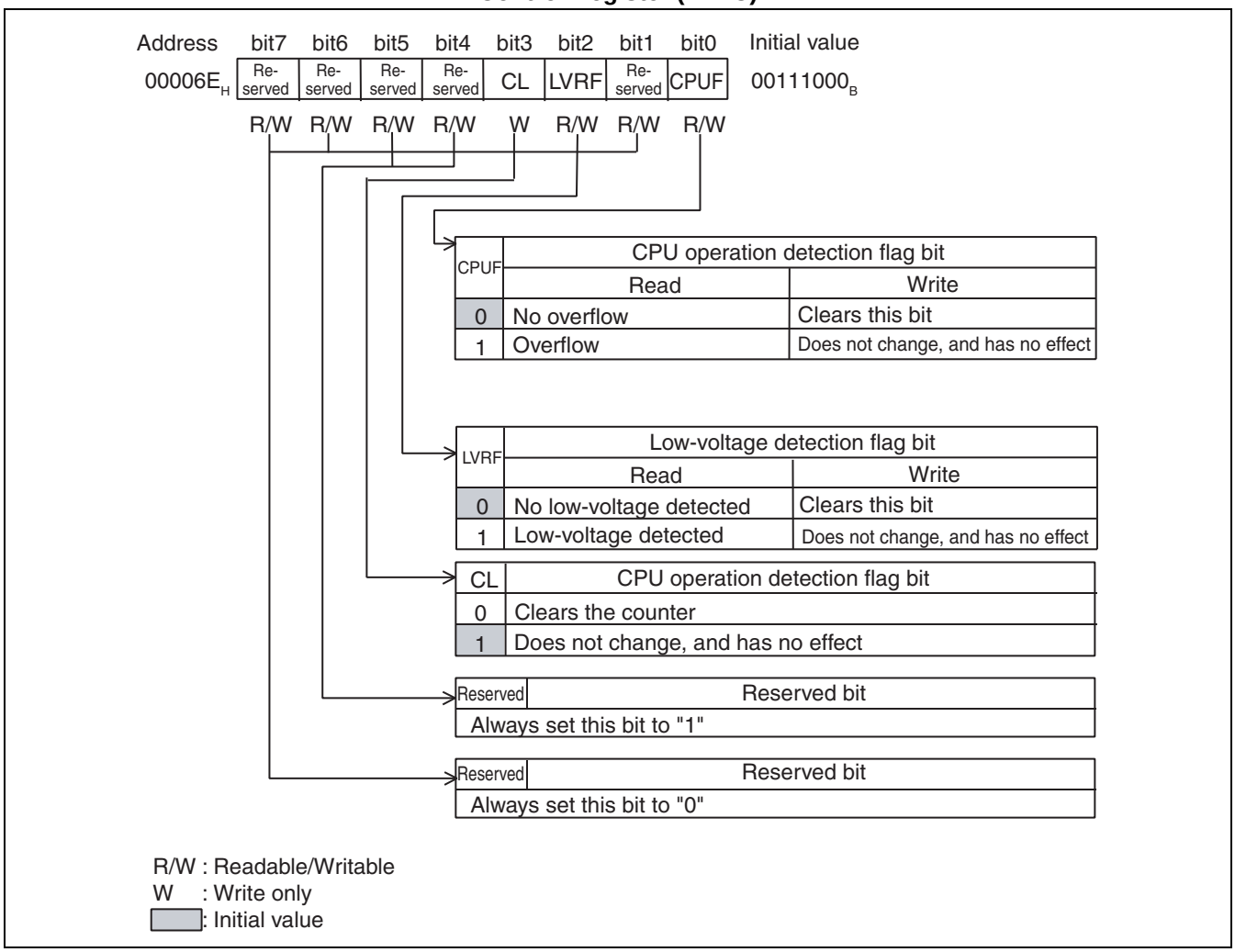


Table 20.3-1 Functions of Each Bit in the Low-voltage/CPU Operation Detection Reset Control Register

Bit name		Function
bit7, bit6	Reserved: Reserved bit	Be sure to write "0" to this bit.
bit5, bit4	Reserved: Reserved bit	Be sure to write "1" to this bit.
bit3	CL: CPU operation detection clear bit	Used to clear the counter of the CPU operation detection circuit. Writing "0" to this bit allows the counter of the CPU operation detection circuit to be cleared.
bit2	LVRF: Low-voltage detection flag bit	If a voltage drop is detected, this bit is set to "1". This bit is cleared by writing "0". Writing "1" has no effect on this bit, and the bit remains unchanged. This bit isn't initialized by the internal reset but it is initialized by the external reset input.
bit1	Reserved: Reserved bit	Be sure to write "0" to this bit.
bit0	CPUF: CPU operation detection flag bit	If CPU operation detection function's counter causes an overflow, this bit is set to "1". This bit is cleared by writing "0". Writing "1" has no effect on this bit, and the bit remains unchanged. This bit is not initialized by the internal reset but it is initialized by the external reset input.

20.4 Operation of the Low-voltage/CPU Operation Detection Reset Circuit

This circuit is used to monitor the power supply voltage. If the power supply voltage is lower than the setting value, this circuit generates an internal reset. The CPU operation detection function generates an internal reset if the counter is not cleared within a certain period. If an internal reset occurs by detecting a low-voltage or CPU runaway, the register content cannot be assured. When a low-voltage reset is cleared and the operation stabilization wait time has elapsed, a reset sequence is executed and then the program will restart from the address specified by the reset vector.

■ Operation of the Low-voltage Detection Reset Circuit

The low-voltage detection reset circuit starts the low-voltage detection operation after a reset is cleared without requiring the operation stabilization wait time.

■ Operation of the CPU Operation Detection Reset Circuit

The CPU operation detection reset circuit starts the CPU operation detection after a reset is cleared without requiring the operation stabilization wait time.

Note:

As the low-voltage reset circuit is always operating, current is consumed even in sleep and stop mode.

20.5 Notes on Using the Low-voltage/CPU Operation Detection Reset Circuit

This section provides notes on using the low-voltage/CPU operation detection reset circuit.

■ Notes on Using the Low-voltage Detection Reset Circuit

● Operation stop disabled in the program

The low-voltage detection reset circuit continuously operates when the operation stabilization wait time has elapsed after power-up. Software does not allow the operation to stop.

● Operation in STOP mode

The low-voltage detection reset function operates even in STOP mode. If a low-voltage condition is detected in STOP mode, a reset occurs and the STOP mode is released.

● Operation of EVA product

For the EVA device, the internal reset is not generated even when a low-voltage is detected.

In this case, low-voltage detection flag bit (LVRF) in the low-voltage/CPU operation detection reset control resister (LVRC) is set, however, reset source bit in the watchdog timer control resister (WDTC) is not set.

■ Notes on Using the CPU Operation Detection Reset Circuit

● Operation stop disabled in the program

The CPU operation detection reset circuit continuously operates after the power-up. Software does not allow the operation to stop.

● Resets by CPU operation detection function suppressed

The CPU operation detection function must clear the counter in constant intervals. Writing "0" to the CL bit in the LVRC register clears the counter, and suppresses a reset generation.

● Stopping and clearing the counter

In the modes where the CPU stops, the CPU operation detection function will clear the counter, causing a stop of operation.

● Operation in sub-oscillation mode

The CPU operation detection function will stop the operation in sub-oscillation mode. For this reason, the watchdog reset function should also be used.

● Operation of EVA product

For the EVA product, the internal reset is not generated even when the counter of the CPU operation detection reset circuit is not cleared for a specified time.

In this case, the CPU operation detection flag bit (CPUF) in the low-voltage/CPU operation detection reset control register

(LVRC) is set, however, reset source bit in the watchdog timer control register (WDTC) is not set.

20.6 Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit

This section provides a sample program for the low-voltage/CPU operation detection reset circuit.

■ Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit

- Specification of processing

Clear the counter of the CPU operation detection function.

[Coding example]

```
LVRC   EQU      006EH            ; Address of the low-voltage/CPU operation
                                  ; detection reset control register
;----------Main program------------------------------------------------------
               CSEG              ; [CODE SEGMENT]
         :
       MOV      LVRC, #00110101B ; Clears the counter of the CPU operation
                                  ; detection function
         :
       END
```

21. Stepping Motor Controller

This chapter describes the functions and operations of the stepping motor controller.

21.1 Overview of the Stepping Motor Controller

21.2 Registers of the Stepping Motor Controller

21.3 Operation of the Stepping Motor Controller

21.4 Notes on Using the Stepping Motor Controller

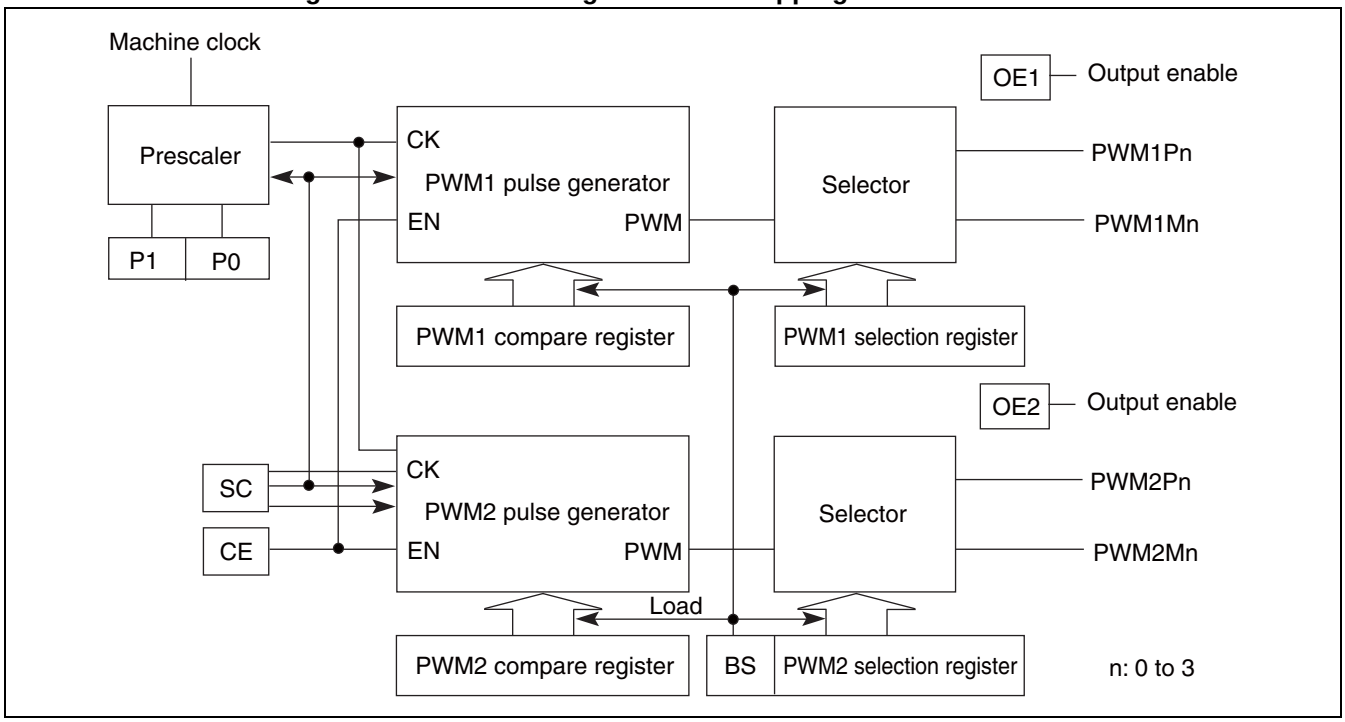
21.1 Overview of the Stepping Motor Controller

The stepping motor controller consists of 2 PWM pulse generators, 4 motor drivers, and the selector logic circuit.
The four motor drivers include a high output performance and can be directly connected at the 4 ends of the 2 motor coils. This stepping motor controller is designed to control the motor rotation by combining the PWM pulse generator and selector logic. The synchronization mechanism assures synchronous operation of the 2 PWMs.

■ Block Diagram of the Stepping Motor Controller

Figure 21.1-1 shows the block diagram of the stepping motor controller.

Figure 21.1-1 Block Diagram of the Stepping Motor Controller



21.2 Registers of the Stepping Motor Controller

The stepping motor controller has the following 5 types of register:

- **PWM control register**
- **PWM1 compare register**
- **PWM2 compare register**
- **PWM1 selection register**
- **PWM2 selection register**

■ Registers of the Stepping Motor Controller

Figure 21.2-1shows the registers of the stepping motor controller.

Figure 21.2-1 Registers of the Stepping Motor Controller

PMW control register (PWC0, PWC1, PWC2, PWC3)

Address: 0080_H, 0082_H, 0084_H, 0086_H

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
OE2	OE1	P1	P0	CE	SC	–	TST
R/W	R/W	R/W	R/W	R/W	R/W	–	R/W
0	0	0	0	0	0	–	0

PMW1 compare register (PWC10, PWC11, PWC12, PWC13)

Address: 3980_H, 3988_H, 3990_H, 3998_H

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
X	X	X	X	X	X	X	X

Address: 3981_H, 3989_H, 3991_H, 3999_H

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
–	–	–	–	–	–	D9	D8
–	–	–	–	–	–	R/W	R/W
–	–	–	–	–	–	X	X

PMW2 compare register (PWC20, PWC21, PWC22, PWC23)

Address: 3982_H, $398A_H$, 3992_H, $399A_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
X	X	X	X	X	X	X	X

Address: 3983_H, $398B_H$, 3993_H, $399B_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
–	–	–	–	–	–	D9	D8
–	–	–	–	–	–	R/W	R/W
–	–	–	–	–	–	X	X

PMW1 selection register (PWS10, PWS11, PWS12, PWS13)

Address: 3984_H, $398C_H$, 3994_H, $399C_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Reserved	Reserved	P2	P1	P0	M2	M1	M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0	0	0	0	0	0	0	0

PMW2 selection register (PWS20, PWS21, PWS22, PWS23)

Address: 3985_H, $398D_H$, 3995_H, $399D_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
–	BS	P2	P1	P0	M2	M1	M0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
–	0	0	0	0	0	0	0

21.2.1 PWM Control Register

The PWM control register is used to control the start/stop operations and interrupts of the stepping motor controller. It also used to specify the external output pins.

■ Bit Configuration of PWM Control Register

Figure 21.2-2 shows the bit configuration of the PWM control register.

Figure 21.2-2 Bit Configuration of PWM Control Register

Address:

0080_H, 0082_H
0084_H, 0086_H

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
OE2	OE1	P1	P0	CE	SC	–	TST
R/W	R/W	R/W	R/W	R/W	R/W	–	R/W
0	0	0	0	0	0	–	0

R/W: Readable/Writable
– : Undefined

[bit7] OE2: Output enable bit

If the OE2 bit is set to "1", external pins are assigned as PWM2P0, PWM2P1, PWM2P2, PWM2P3 and PWM2M0, PWM2M1, PWM2M2, PWM2M3. When this bit is "0", it can be used as a general-purpose I/O.

[bit6] OE1: Output enable bit

If the OE1 bit is set to "1", external pins are assigned as PWM1P0, PWM1P1, PWM1P2, PWM1P3 and PWM1M0, PWM1M1, PWM1M2, PWM1M3. When this bit is "0", it can be used as a general-purpose I/O.

[bit5, bit4] P1, P0: Operation clock selection bit

The P1 and P0 bits are used to specify the clock input signal for the PWM pulse generator.

P1	P0	Clock input
0	0	Machine clock
0	1	1/2 machine clock
1	0	1/4 machine clock
1	1	1/8 machine clock

[bit3] CE: Count enable bit

The CE bit is used to enable the PWM pulse generator to operate. The PWM pulse generator starts operation when the CE bit is set to "1". Note that the PWM2 pulse generator will start one machine clock cycle after the PWM1 pulse generator, which is helpful to decrease the switching noise from the output driver.

By setting the CE bit to "0", the PWM pulse generator is initialized and then stops.

[bit2] SC: 8/10 bit switch bit

If the SC bit is set to "1", PWM operates with 10 bits. If set to "0", PWM operates with 8 bits.

[bit0] TST: Test bit

The TST bit is used for device tests. The TST bit must always be set to "0" for the user application

21.2.2 PWM1 and PWM2 Compare Registers

The contents of the 2 of 8-bit (or 10-bit) compare registers, PWM1 and PWM2, specify the width of the PWM pulse. The stored value of 00_H (000_H) indicates that the PWM duty is 0%, while FF_H ($3FF_H$) indicates that the duty is 99.6% (99.9%).

■ Bit Configuration of the PWM1 and PWM2 Compare Registers

The PWM1 and PWM2 compare registers can be accessed at any time. The changed values are reflected to the pulse width at the end of the current PWM cycle after the BS bit in the PWM2 selection register is set to "1".

This register requires the word access.

Figure 21.2-3 shows the bit configuration of the PWM1 and PWM2 compare registers. Figure 21.2-4 shows the relationship between the PWM pulse width and the setting value.

Figure 21.2-3 Bit Configuration of the PWM1 and PWM2 Compare Registers

PMW1 compare register (PWC10, PWC11, PWC12, PWC13)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
3980_H, 3988_H 3990_H, 3998_H	D7	D6	D5	D4	D3	D2	D1	D0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	X	X	X	X	X	X	X	X

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
3981_H, 3989_H 3991_H, 3999_H	–	–	–	–	–	–	D9	D8
	–	–	–	–	–	–	R/W	R/W
	–	–	–	–	–	–	X	X

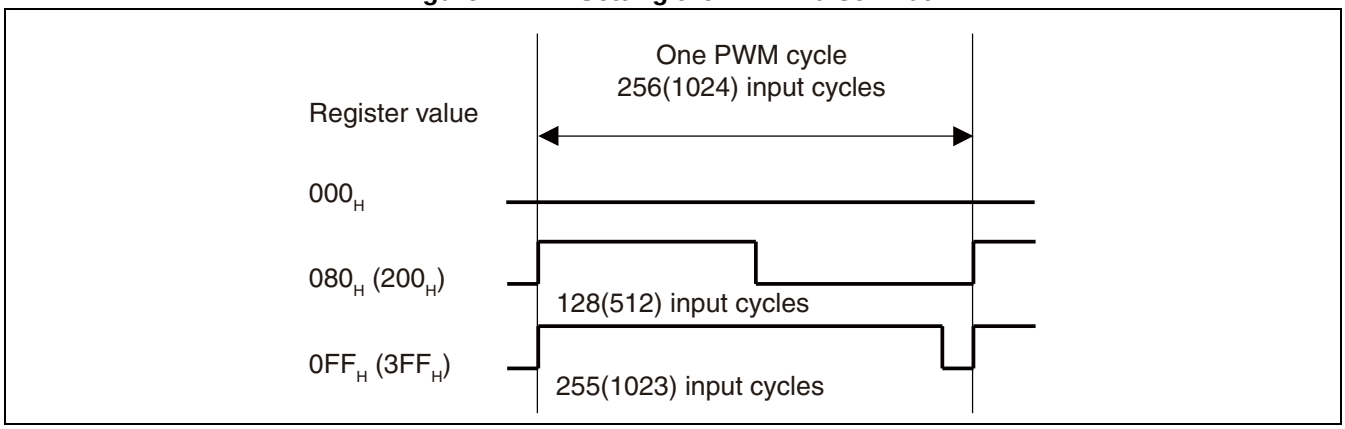
PMW2 compare register (PWC20, PWC21, PWC22, PWC23)

Address:	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
3982_H, $398A_H$ 3992_H, $399A_H$	D7	D6	D5	D4	D3	D2	D1	D0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	X	X	X	X	X	X	X	X

Address:	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
3983_H, $398B_H$ 3993_H, $399B_H$	–	–	–	–	–	–	D9	D8
	–	–	–	–	–	–	R/W	R/W
	–	–	–	–	–	–	X	X

R/W: Readable/Writable
– : Undefined
X : Undefined value

Figure 21.2-4 Setting the PWM Pulse Width



21.2.3 PWM1 and PWM2 Selection Registers

The PWM1/PWM2 selection registers are used to select whether the stepping motor controller's external pin output is either "L", "H", PWM pulse, or high-impedance.

■ Bit Configuration of the PWM1 and PWM2 Select Registers

Figure 21.2-5 shows the bit configuration of the PWM1/PWM2 selection registers.

Figure 21.2-5 Bit Configuration of the PWM1 and PWM2 Select Registers

PMW1 selection register (PWS10, PWS11, PWS12, PWS13)

Address: 3984_H, $398C_H$ / 3994_H, $399C_H$

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Reserved	Reserved	P2	P1	P0	M2	M1	M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
0	0	0	0	0	0	0	0

PMW2 selection register (PWS20, PWS21, PWS22, PWS23)

Address: 3985_H, $398D_H$ / 3995_H, $399D_H$

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
–	BS	P2	P1	P0	M2	M1	M0
–	R/W	R/W	R/W	R/W	R/W	R/W	R/W
–	0	0	0	0	0	0	0

R/W: Readable/Writable
– : Undefined

[bit14] BS: Rewrite bit

The BS bit is provided to keep in synchronization with the settings for PWM output. Any changes applied to the 2 compare registers and 2 selection registers before the BS bit is set will not be reflected to the output signal.

When the BS bit is set to "1", the PWM pulse generator and selector load the register content at the end of the current PWM cycle. The BS bit is automatically reset to "0" at the start of the following PWM cycle. If the BS bit is set to "1" by software at the same time as an automatic reset, the BS bit will be set to "1" (i.e., remains unchanged), and the automatic reset is cancelled.

[bit13 to bit11] P2 to P0: Output selection bits

The P2 to P0 bits are used to select the output signal at PWM2P0.

[bit10 to bit8] M2 to M0: Output selection bits

The M2 to M0 bits are used to select the output signal at PWM2M0.

[bit7, bit6] Reserved bits

Flash Memory/Mask ROM products

・Write: Writing to this bit has no effect on the operation.

・Read: Returns the value written to this bit.

Evaluation product

・Write: Writing to this bit has no effect on the operation.

・Read: Always returns "1".

[bit5 to bit3] P2 to P0: Output selection bits

The P2 to P0 bits are used to select the output signal at PWM1P0.

[bit2 to bit0] M2 to M0: Output selection bits

The M2 to M0 bits are used to select the output signal at PWM1M0. The table below shows the relationship between the output level and selection bits.

P2	P1	P0	PWMnP0	M2	M1	M0	PWMnM0
0	0	0	L	0	0	0	L
0	0	1	H	0	0	1	H
0	1	X	PWM pulse	0	1	X	PWM pulse
1	X	X	High-impedance	1	X	X	High-impedance

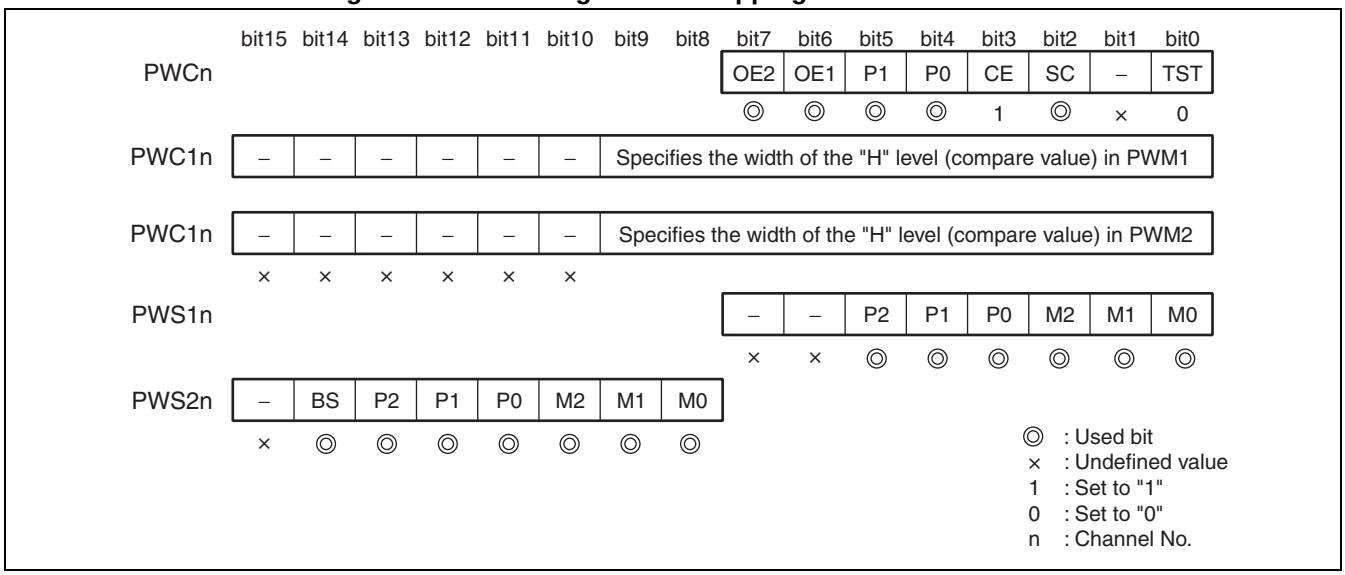
21.3 Operation of the Stepping Motor Controller

This section describes the operation of the stepping motor controller.

■ Settings for Stepping Motor Controller Operation

Figure 21.3-1 shows the setting to operate the stepping motor controller.

Figure 21.3-1 Settings of the Stepping Motor Controller

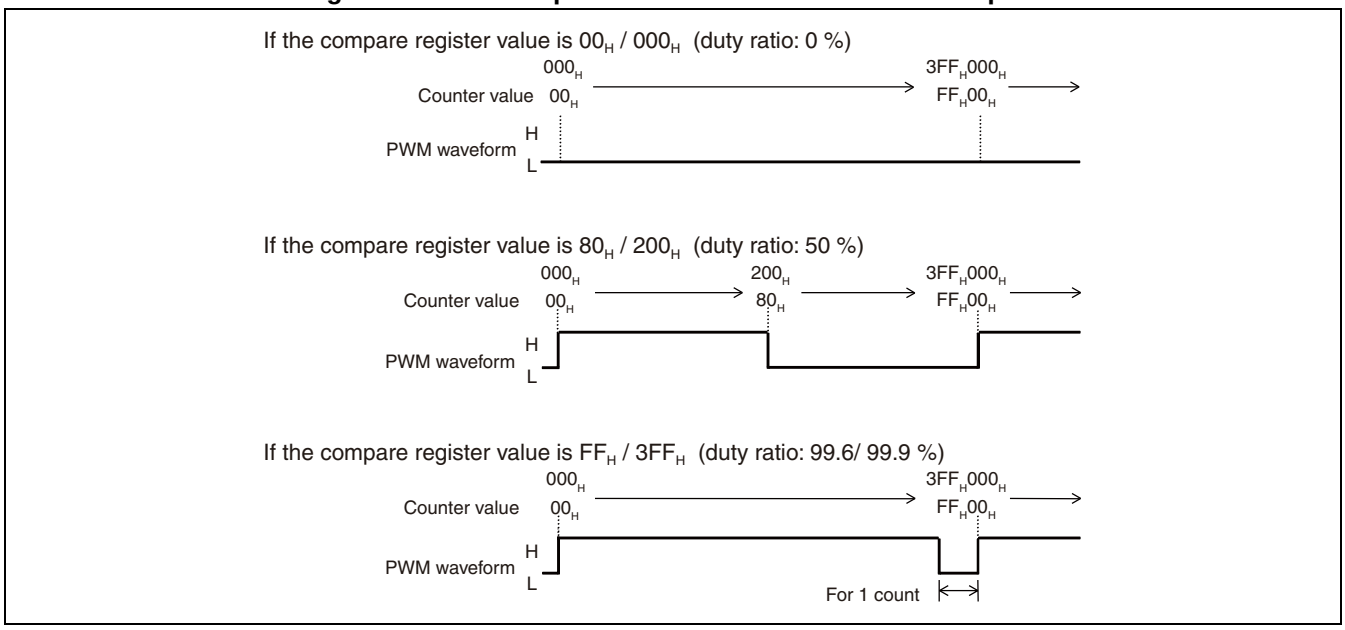


● Operation of the PWM pulse generation circuit

As soon as the counter starts (PWCn: CE = 1), count-up will start with 00_H at the rising edge of the selected count clock. The PWM output waveform stays at the "H" level until the counter value matches the value set in the PWM compare register and then stays at the "L" level until an overflow of the counter value occurs ($FF_H \rightarrow 00_H$).

Figure 21.3-2shows the PWM waveform generated by the PWM generation circuit.

Figure 21.3-2 Example of PWM1/PWM2 Waveform Outputs



- Selection of motor drive signal

The motor drive signals, that are output to the stepping motor controller related pins, can be selected from 4 types of signals for each pin by setting the PWM selection register.

Table 21.3-1 shows the selection of the motor drive signal and the settings of PWM selection register 1 and PWM selection register 2.

When you write "1" to the BS bit in the PWM selection register 2 after the above settings have been made, the new setting value will become effective at the end of the current PWM cycle. This BS bit is automatically cleared at the start of the PWM cycle. If writing to the BS bit and clearing the BS bit both simultaneously occur at the beginning of the PWM cycle, the writing to the BS bit has priority and clearing of the BS bit is cancelled.

Table 21.3-1 Selection of the Motor Drive Signal and Settings of the PWM Selection Registers 1 and 2

P2, P1, P0 bits	PWM1P output PWM2P output	M2, M1, M0 bits	PWM1M output PWM2M output
000_B	L	000_B	L
001_B	H	001_B	H
$01X_B$	PWM pulse	$01X_B$	PWM pulse
$1XX_B$	High impedance	$1XX_B$	High impedance

21.4 Notes on Using the Stepping Motor Controller

This section provides notes on using the stepping motor controller.

■ Notes on Changing the PWM Setting Values

PWM compare register 1 (PWC1n), PWM compare register 2 (PWC2n), PWM selection register 1 (PWS1n), and PWM selection register 2 (PWS2n) can be accessed at any time, however, in order to change the setting of the PWM's "H" width or PWM output, write the setting values to these registers, then set the BS bit in PWM selection register 2 to "1" (or do this simultaneously).

If the BS bit is set to "1", the new setting value will be valid at the end of the current PWM cycle, and the BS bit is automatically cleared.

If you set the BS bit to "1" and reset the BS bit at the end of the PWM cycle at the same time, the writing "1" has priority and resetting the BS bit will be cancelled.

■ Notes on Enabling the PWM Output

Before enabling the PWM output, you must write to the DDR7/DDR8 register of the pin to be used. Otherwise, the pin output is set to "L".

For more information, see Sections "8.10.2 Description of Port 7 Operation" and "8.11.2 Description of Port 8 Operation".

■ Handling of Power Supply (DVcc/DVss) for High-current Output Buffer Pin

・Flash Memory/Mask ROM products (MB90F922/MB90922)

As the high-current output buffer power supply (DVcc/DVss) and the digital power supply (Vcc) are isolated from each other, DVcc can be set to a potential higher than Vcc.

Note that, if the power supply (DVcc/DVss) for high-current output buffer pin is turned on prior to the digital power supply (Vcc), however, port 7 or 8 for stepping motor output may momentarily output an "H" or "L" level signal at the rise of DVcc.

To prevent this, turn on the digital power supply (Vcc) prior to the power supply for high-current output buffer pin.

Apply a voltage to the power supply (DVcc/DVss) for high-current output buffer pin even when the high-current output buffer pin is used as a general-purpose port.

・EVA product (MB90V920)

As the MB90V920 does not have the power supply (DVcc/DVss) for high-current output buffer and digital power supply (Vcc) isolated from each other, set DVcc to a potential equal to or lower than Vcc.

Before turning on the power supply (DVcc/DVss) for high-current output buffer pin, be sure to turn on the digital power supply (Vcc). Also, turn off the digital power supply (Vcc) after turning off the power supply for high-current output buffer pin. (It is acceptable to turn on or off the power supply for high-current output buffer pin and the digital power supply at the same time.)

Apply a voltage to the power supply (DVcc/DVss) for high-current output buffer pin even when the high-current output buffer pin is used as a general-purpose port.

22. Sound Generator

This chapter describes the functions and operations of the sound generator.

22.1 Outline of the Sound Generator
22.2 Registers of the Sound Generator

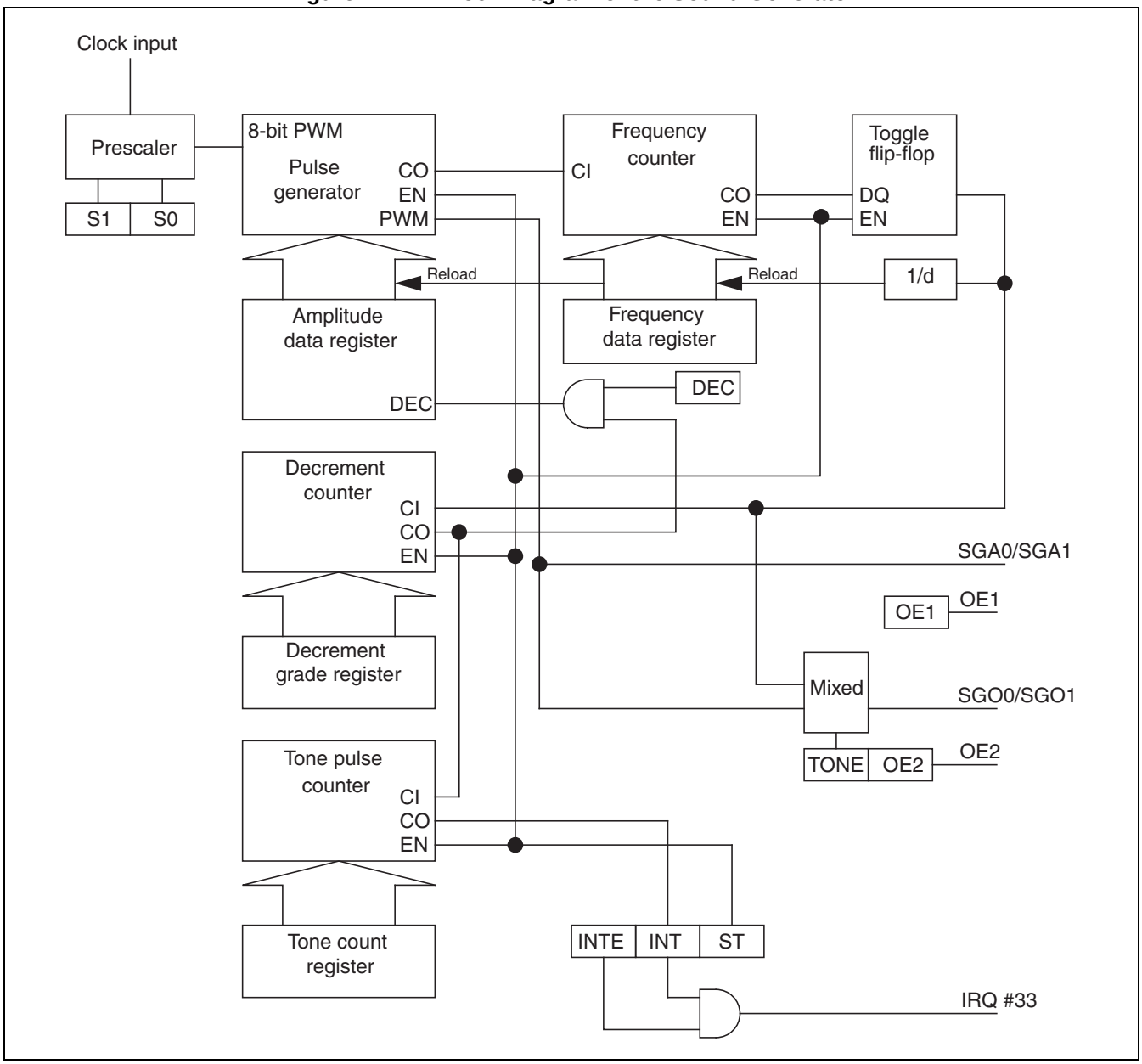
22.1 Outline of the Sound Generator

The sound generator contains the sound control register, frequency data register, amplitude data register, decrement grade register, tone count register, PWM pulse generator, frequency counter, decrement counter, and tone pulse counter.

■ Block Diagram of the Sound Generator

Figure 22.1-1 shows a block diagram of the sound generator.

Figure 22.1-1 Block Diagram of the Sound Generator



22.2 Registers of the Sound Generator

The sound generator has the following 5 types of registers.

- **Sound control registers (SGCRH0/SGCRH1, SGCRL0/SGCRL1)**
- **Frequency data registers (SGFR0/SGFR1)**
- **Amplitude data registers (SGAR0/SGAR1)**
- **Decrement grade registers (SGDR0/SGDR1)**
- **Tone count registers (SGTR0/SGTR1)**

■ Registers of the Sound Generator

Figure 22.2-1 illustrates the registers of the sound generator.

Figure 22.2-1 Registers of the Sound Generator

Upper bits in the sound control register

Bit	15	14	13	12	11	10	9	8	
Address: 00005B$_H$ / Address: 0000D9$_H$	TST	-	-	-	-	Re-served	BUSY	DEC	SGCRH0 / SGCRH1
Read/Write →	R/W	-	-	-	-	R/W	R	R/W	
Initial value →	0	-	-	-	-	1	0	0	

Lower bits in the sound control register

Bit	7	6	5	4	3	2	1	0	
Address: 00005A$_H$ / Address: 0000D8$_H$	S1	S0	TONE	OE2	OE1	INTE	INT	ST	SGCRL0 / SGCRL1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Amplitude Data Register

Bit	15	14	13	12	11	10	9	8	
Address: 00005D$_H$ / Address: 003975$_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGAR0 / SGAR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

Frequency Data Register

Bit	7	6	5	4	3	2	1	0	
Address: 00005C$_H$ / Address: 003974$_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGFR0 / SGFR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Tone Count Register

Bit	15	14	13	12	11	10	9	8	
Address: 00005F$_H$ / Address: 003977$_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGTR0 / SGTR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Decrement Grade Register

Bit	7	6	5	4	3	2	1	0	
Address: 00005E$_H$ / Address: 003976$_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGDR0 / SGDR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

22.2.1 Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1)

The sound control register controls the operating status by controlling the interrupt of the sound generator and setting its external output pins.

■ Bit Configuration of Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1)

Figure 22.2-2 shows the bit configuration of the sound control register.

Figure 22.2-2 Bit Configuration of Sound Control Register

Upper bits in the sound control register

Bit	15	14	13	12	11	10	9	8	
Address: 00005B$_H$ Address: 0000D9$_H$	TST	-	-	-	-	Reserved	BUSY	DEC	SGCRH0 SGCRH1
Read/Write →	R/W	-	-	-	-	R/W	R	R/W	
Initial value →	0	-	-	-	-	1	0	0	

Lower bits in the sound control register

Bit	7	6	5	4	3	2	1	0	
Address: 00005A$_H$ Address: 0000D8$_H$	S1	S0	TONE	OE2	OE1	INTE	INT	ST	SGCRL0 SGCRL1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

[bit15] TST: Test bit

This bit is to test the device. User applications must clear the bit to "0".

[bit10] Reserved bit

Always write "1" to this bit.

Reading the bit returns "1".

[bit9] BUSY: Busy bit

This bit indicates whether the sound generator is operating. The bit is set to "1" as the ST bit is set to "1". It is reset to "0" when the operation is completed at the end of one tone cycle with the ST bit reset to "0". Any write instruction performed on this bit has no effect.

[bit8] DEC: Automatic decrement enable bit

The DEC bit is for automatic degradation of sound in combination with the decrement grade register.

If this bit is set to "1", the value held in the amplitude data register is decremented by one every time the decrement counter counts the number of tone pulses from the toggle flip-flop specified by the decrement grade register.

[bit7, bit6] S1, S0: Operation clock select bits

This bit group specifies the clock input signal for the sound generator.

S1	S0	Clock input
0	0	Machine Clock
0	1	1/2 machine clock
1	0	1/4 machine clock
1	1	1/8 machine clock

[bit5] TONE: Tone output bit

After this bit is set to "1", the SGO signal will be a simple rectangular waveform (tone pulses) from the toggle flip-flop. Otherwise, it will be a combination (AND logic) of the tone and PWM pulses.

[bit4] OE2: Sound output enable bit

If this bit is set to "1", the external pin is assigned to be used for SGO output. In other cases, it can be used as a general-purpose pin.

[bit3] OE1: Amplitude output enable bit

If this bit is set to "1", the external pin is assigned to be used for SGA output. In other cases, it can be used as a general-purpose pin.

The SGA signal is a PWM pulse from the PWM pulse generator, indicating the sound amplitude.

[bit2] INTE: Interrupt enable bit

This bit is used to enable interrupt signals of the sound generator. If the INT bit is set to "1" with this bit "1", the sound generator outputs an interrupt with a signal.

[bit1] INT: Interrupt bit

This bit is set to "1" if the tone pulse count specified by the tone count register and decrement grade register is counted by the tone pulse counter.

Writing "0" to this bit resets it to "0". Writing "1" has no effect, and read-modify-write instructions always return "1".

[bit0] ST: Start bit

This bit is used to start sound generator operation. As long as this bit is "1", the sound generator is operating.

If this bit is reset to "0", the sound generator stops the operation at the end of the current tone cycle. The BUSY bit indicates whether the sound generator has completely stopped the operation.

22.2.2 Frequency Data Register (SGFR0/SGFR1)

The frequency data register stores the reload value for the frequency counter. The value stored indicates the frequency of sound (or tone signal from the toggle flip-flop). The register value is reloaded into the counter by each toggle signal transition.

■ Frequency Data Registers (SGFR0/SGFR1)

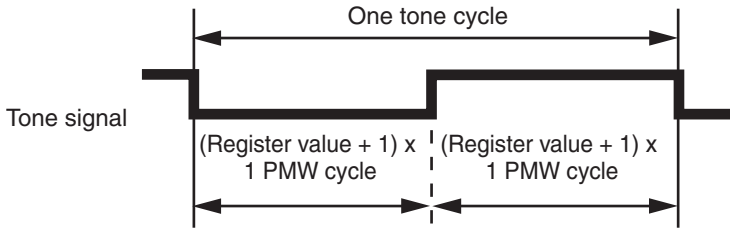
Figure 22.2-3 shows the bit configuration of the frequency data register; Figure 22.2-4 shows the relationship between tone signals and register values.

Figure 22.2-3 Bit Configuration of Frequency Data Register

Frequency Data Register

Bit	7	6	5	4	3	2	1	0	
Address: 00005C$_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGFR0
Address: 003974$_H$									SGFR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

Figure 22.2-4 Relationship between Register Value and Tone Signal



Note:

Changing the register value during operation may cause a deviation of a 50% of duty cycle depending on the change timing.

22.2.3 Amplitude Data Register (SGAR0/SGAR1)

The amplitude data register stores the reload value for the PWM pulse generator. The register value indicates the sound amplitude. It is reloaded to the PWM pulse generator each time a tone cycle ends.

■ Amplitude Data Registers (SGAR0/SGAR1)

Figure 22.2-5 shows the bit configuration of the amplitude data register.

Figure 22.2-5 Bit Configuration of Amplitude Data Register

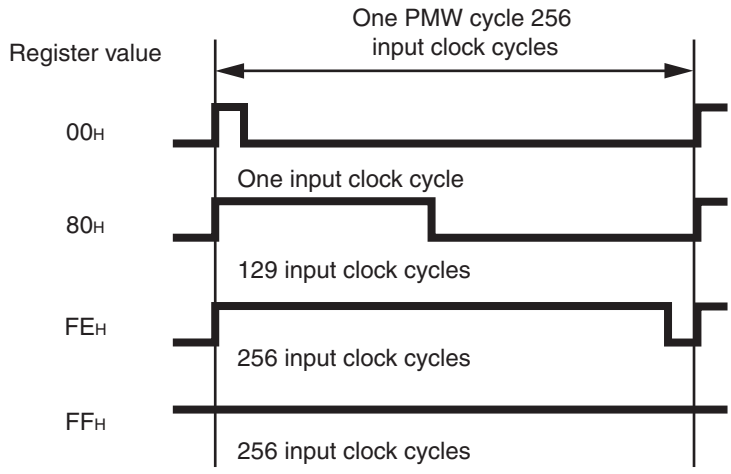
Amplitude Data Register

Bit	15	14	13	12	11	10	9	8	
Address: $00005D_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGAR0
Address: 003975_H									SGAR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	0	0	0	0	0	0	0	0	

If the decrement counter reaches the reload value while the DEC bit is "1", the register value is decremented by 1. If the register value reaches 00_H, no more decrement is made to the register value. However, the sound generator continues its operation until the ST bit is cleared.

Figure 22.2-6 shows the relationship between register values and PWM pulses.

Figure 22.2-6 Relationship between Register Value and PWM Pulse



When the register value is set to FF_H, the PWM signal is always set to "1".

22.2.4 Decrement Grade Register (SGDR0/SGDR1)

The decrement grade register stores the reload value for the decrement counter. The register is to automatically decrement the value held in the amplitude data register.

■ Decrement Grade Registers (SGDR0/SGDR1)

Figure 22.2-7 shows the bit configuration of the decrement grade register.

Figure 22.2-7 Bit Configuration of Decrement Grade Register

Decrement Grade Register

Bit	7	6	5	4	3	2	1	0	
Address: $00005E_H$ Address: 003976_H	D7	D7	D7	D7	D7	D2	D1	D0	SGDR0 SGDR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

If the decrement counter counts tone pulses up to the reload value with the DEC bit "1", the amplitude data register is decremented by 1 at the end of the tone cycle.

This operation enables automatic sound degradation while reducing the number of CPU interventions.

Note that the tone pulse count specified by the register is "register value +1". With the decrement grade register set to 00_H, the decrement operation is performed by every tone cycle.

22.2.5 Tone Count Register (SGTR0/SGTR1)

The tone count register stores the reload value for the tone pulse counter. The tone pulse counter counts the number of tone pulses (or the number of decrement operations), and sets the INT bit when it reaches to the reload value. The register aims to decrease the interrupts.

■ Tone Count Registers (SGTR0/SGTR1)

Figure 22.2-8 shows the bit configuration of the tone count register.

Figure 22.2-8 Bit Configuration of Tone Count Register

Tone Count Register

Bit	15	14	13	12	11	10	9	8	
Address: $00005F_H$	D7	D7	D7	D7	D7	D2	D1	D0	SGTR0
Address: 003977_H									SGTR1
Read/Write →	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value →	X	X	X	X	X	X	X	X	

The count input of the tone pulse counter is connected to the carry-out signal from the decrement counter. If the tone count register is set to 00_H, the tone pulse counter sets the INT bit every time a carry-out occurs at the decrement counter. The tone pulse count stored is therefore expressed as follows:

((decrement grade register) + 1) × ((tone count register) + 1)

In other words, if both registers are set to 00_H, the INT bit is set by every tone cycle.

23. Address Match Detection Function

This chapter describes the functions and operations of the address match detection function.

23.1 Outline of the Address Match Detection Function

23.2 Sample Application of the Address Match Detection Function

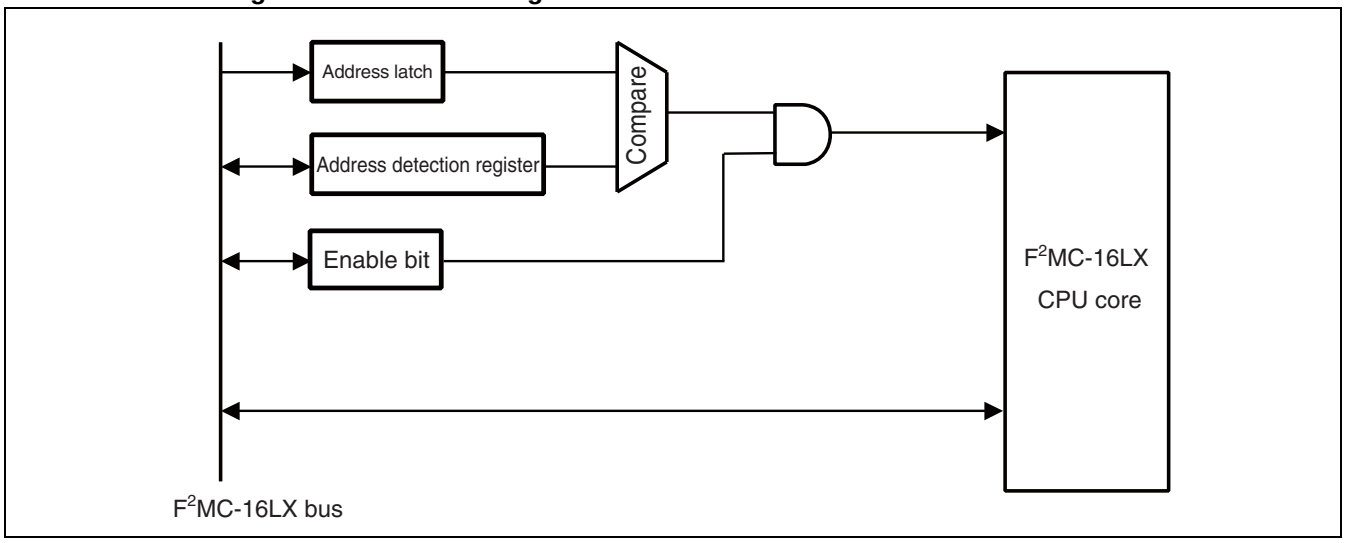
23.1 Outline of the Address Match Detection Function

Once the address matches the setting value in the address detection register, the INT9 instruction is executed. Once the INT9 interrupt service routine is processed, the address match can be detected.
There are two address detection registers, each of which has a compare enable bit. If the address detection register matches the program counter with the compare enable bit "1", the CPU forces the execution of an INT9 instruction.

■ Block Diagram of the Address Match Detection Function

Figure 23.1-1 shows a block diagram of the address match detection function.

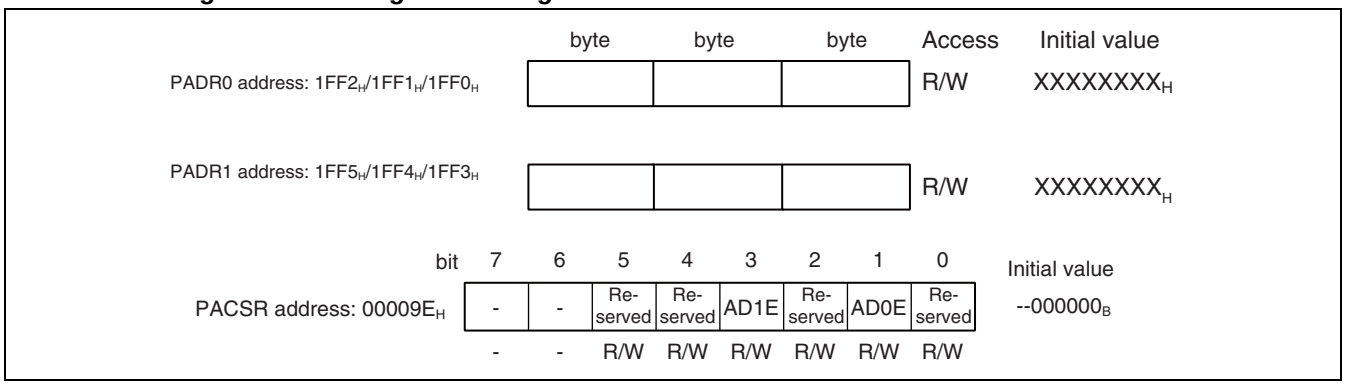
Figure 23.1-1 Block Diagram of the Address Match Detection Function



■ Register Configuration of the Address Match Detection Function

Figure 23.1-2 shows the register configuration of the address match detection function.

Figure 23.1-2 Register Configuration of the Address Match Detection Function



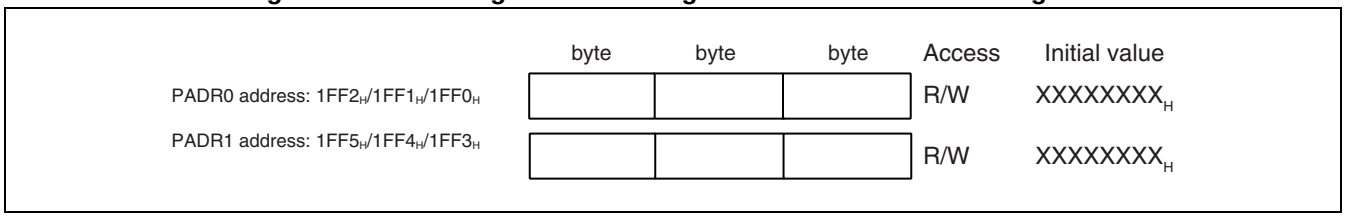
■ Program Address Detection Registers (PADR0/PADR1)

The program address detection register holds the address to be compared with the program counter value. When the address of the instruction executed by the program matches the set value, with the PACSR interrupt enable bit "1", this module requests the CPU to execute the INT9 instruction.

If the corresponding interrupt enable bit is "0", no actions are taken.

Figure 23.1-3 shows the register configuration of the program address detection register.

Figure 23.1-3 Configuration of Program Address Detection Register



PADR0 and PADR1 correspond to the PACSR's interrupt enable bits as follows:

Program address detection register	Interrupt enable bit
PADR0	PACSR: AD0E
PADR1	PACSR: AD1E

■ Program Address Detection Control Register (PACSR)

The program address detection control register (PACSR) controls the address detection function and indicates its state.

Figure 23.1-4 shows the bit configuration of the program address detection control register (PACSR).

Figure 23.1-4 Bit Configuration of Program Address Detection Control Register (PACSR)

bit	7	6	5	4	3	2	1	0
PACSR address: 00009EH	-	-	Re-served	Re-served	AD1E	Re-served	AD0E	Re-served
Read/write ⇒	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial value ⇒	-	-	0	0	0	0	0	0

[bit7, bit6] Undefined bits

Reading the bit returns an indeterminate value; writing it has no effect on the operation.

[bit5, bit4] Reserved bits

Always write "0".

[bit3] AD1E (Compare Enable 1)

This bit enables the operation of PADR1.

If the PADR1 register value matches the address with this bit "1", the INT9 instruction is issued to the CPU.

[bit2] Reserved bit

Always write "0".

[bit1] AD0E (Compare Enable 0)

This bit enables the operation of PADR0.

If the PADR0 register value matches the address with this bit "1", the INT9 instruction is issued to the CPU.

[bit0] Reserved bit

Always write "0".

■ Operation of the Address Match Detection Function

If the program counter has the same address as the program address detection register, the INT9 instruction is executed. If the INT9 interrupt service routine is processed, the address match detection function can be achieved.

There are two address detection registers, each of which has a compare enable bit. If the address detection register matches the program counter, with the compare enable bit "1", the CPU forces the execution of the INT9 instruction.

Note:

If the address detection register and program counter values match, the contents of the internal data bus are replaced to 01_H and the INT9 instruction is executed. Before updating the content of the address detection register, set the compare enable bit to "0". Updating it with the compare enable bit "1" may cause an error.

The address match detection function is effective only for addresses in built-in ROM. Even if an address in the external memory area is specified. The INT9 instruction will not be executed.

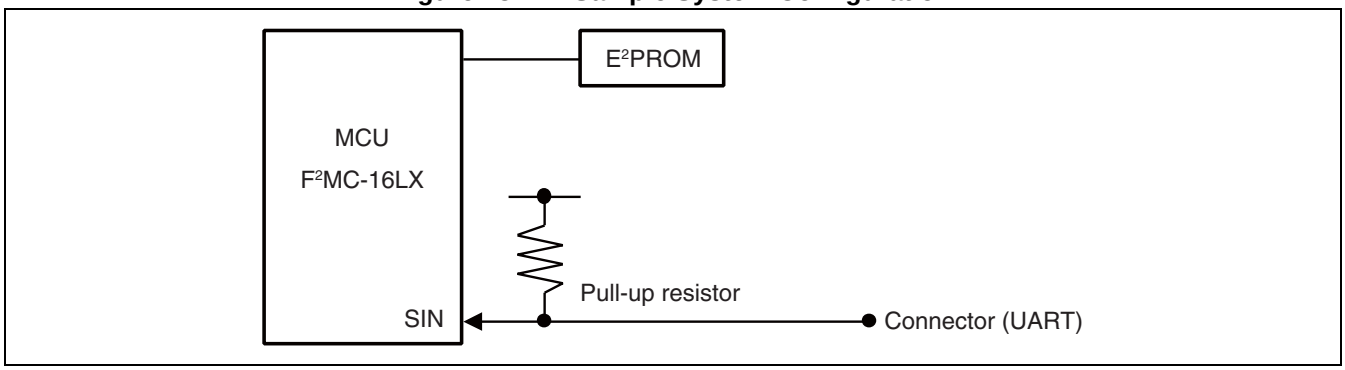
23.2 Sample Application of the Address Match Detection Function

The address match detection can be functioned by providing the E^{2}PROM and storing correction-related information and patch programs in it. The CPU specifies the address necessary to be corrected based on the information stored in the E^{2}PROM and transmits to the patch program to RAM. Address match detection allows the execution of the INT9 instruction to pass control to the patch program.

■ System Configuration

Figure 23.2-1 shows a sample system configuration.

Figure 23.2-1 Sample System Configuration



■ E^{2}PROM Memory Map

Table 23.2-1 shows the E^{2}PROM memory map.

Table 23.2-1 E^{2}PROM Memory Map

Address	Explanation
0000_H	Corrected program No. 0 byte count (0 indicates no ROM correction.)
0001_H	Program address No. 0 bit7 to bit0
0002_H	Program address No. 0 bit15 to bit8
0003_H	Program address No. 0 bit24 to bit16
0004_H	Corrected program No. 1 byte count (0 indicates no ROM correction.)
0005_H	Program address No. 1 bit7 to bit0
0006_H	Program address No. 1 bit15 to bit8
0007_H	Program address No. 1 bit24 to bit16
0010_H or greater	Corrected program No. 0/1 main body

Note:

E^2PROM in the initial state must be all "0".

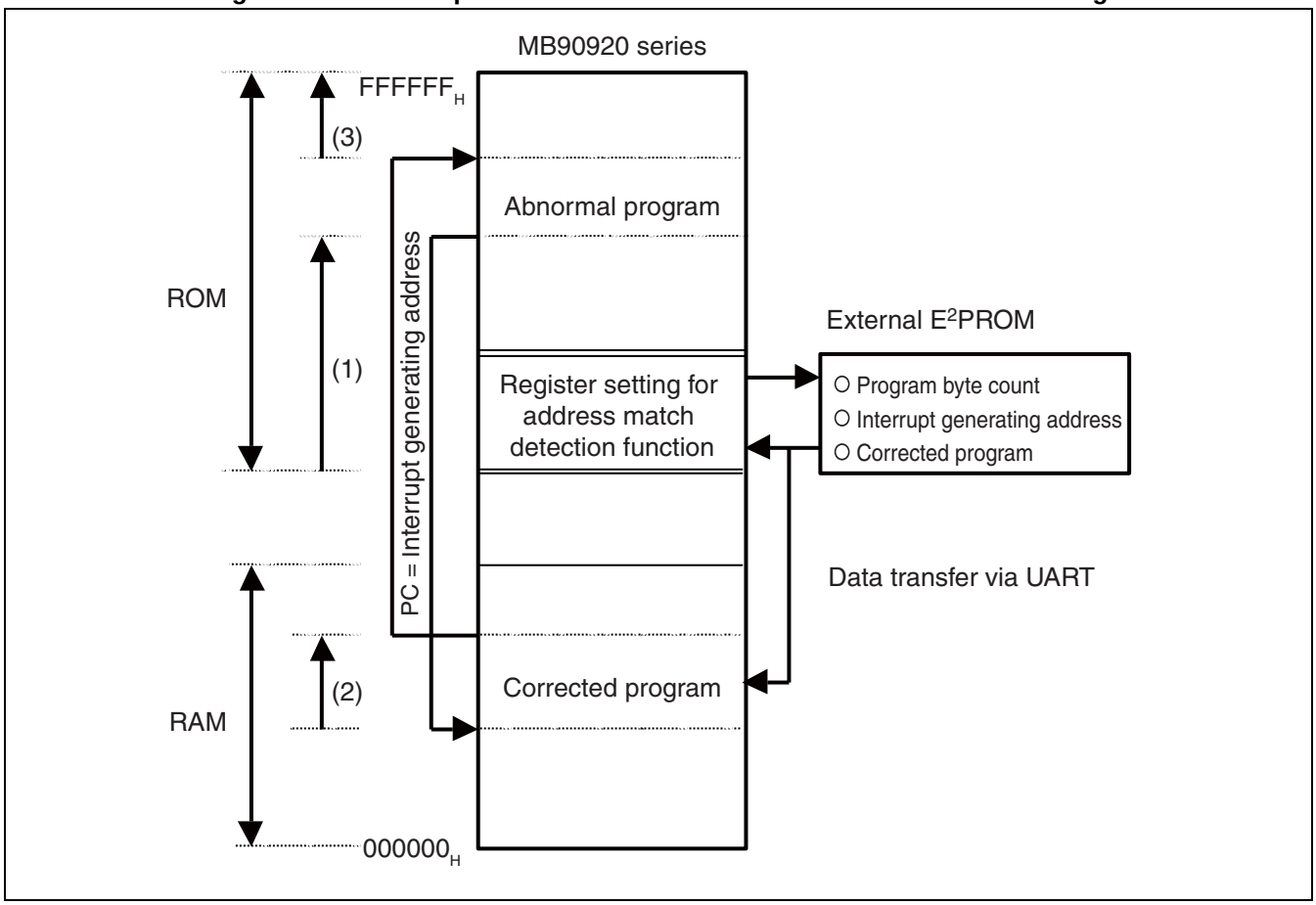
23.2.1 Example of Program Error Correction

The main part of the patch program and its program address are transferred to the MCU via the connector (UART). The MCU writes the information to E^2PROM.

■ If a Program Error Occurs

Figure 23.2-3 shows an example of address match detection function processing in which a program error occurs.

Figure 23.2-2 Example of Address Match Detection Function Processing



23.2.2 Example of Correction Processing

The MCU reads the E^{2}PROM value after a reset. If the byte count of the patch program is not "0", the MCU reads the main part of the patch program and writes it to RAM. It then sets PADR0 or PADR1 to the program address to enable the operation. The start address of the program written to RAM is stored in RAM at the address specified in each address detection register.
In this case, the INT9 service routine searches for the user defined address to jump to the corrected program.

■ Flowchart of the Address Match Detection Function Processing

Figure 23.2-3 shows a flowchart of address match detection function processing.

Figure 23.2-3 Flowchart of Address Match Detection Function Processing

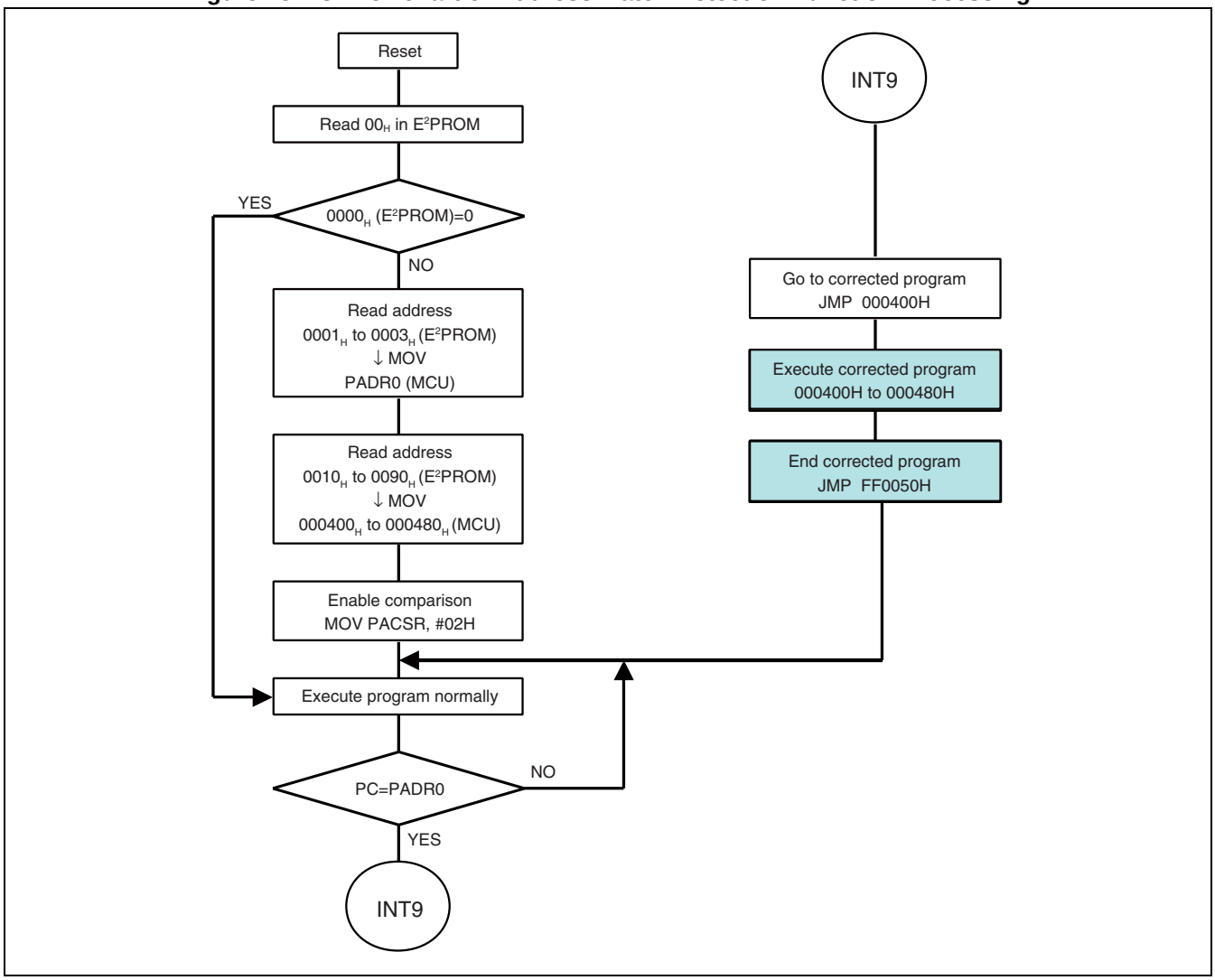
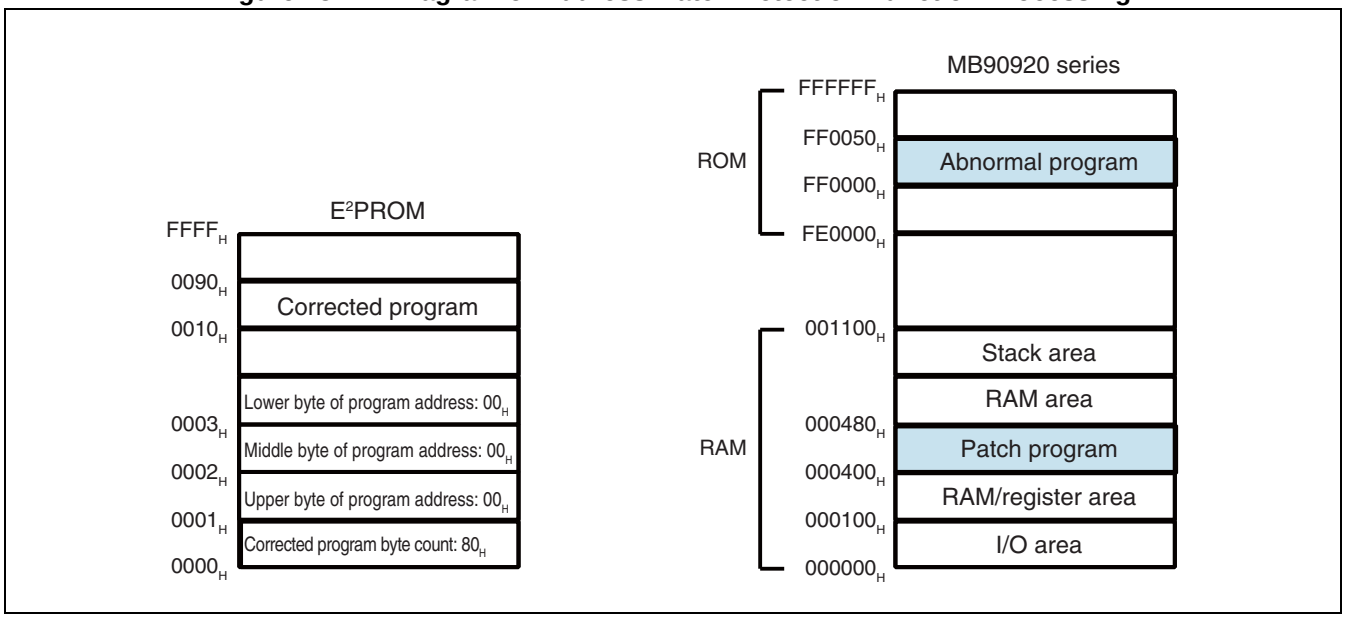


Figure 23.2-4 Diagram of Address Match Detection Function Processing



■ INT9 Interrupt

The interrupt routine checks the PC value saved to the stack to identify the address detected as having caused an interrupt and branches control to the corresponding program. Information stacked upon the occurrence of the interrupt is discarded.

24. ROM Mirror Function Select Module

This chapter describes the ROM mirror function select module.

24.1 Outline of the ROM Mirror Function Select Module
24.2 ROM Mirror Function Select Register (ROMM)

24.1 Outline of the ROM Mirror Function Select Module

The ROM mirror function select module can be used to select the viewing of bank FF via bank 00 by setting its register.

■ Register of the ROM Mirror Function Select Module

Figure 24.1-1 shows the bit configuration of register of the ROM mirror function select module.

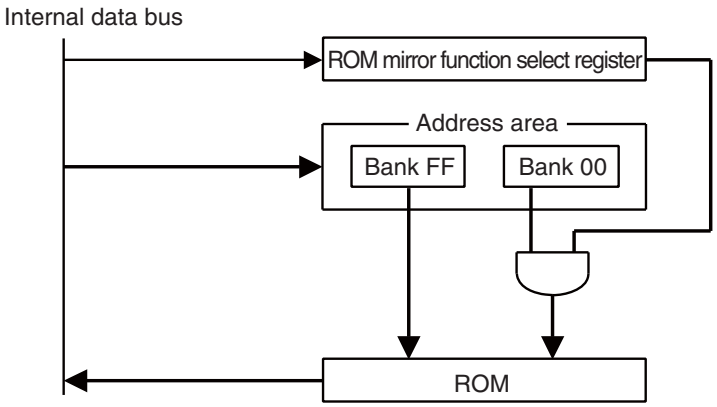
Figure 24.1-1 Bit Configuration of Register of ROM Mirror Function Select Module

Bit	15	14	13	12	11	10	9	8	
Address: 00006F$_H$	–	–	–	–	–	–	–	MI	ROMM
Read/Write →	–	–	–	–	–	–	–	W	
Initial value →	–	–	–	–	–	–	–	1	

■ Block Diagram of the ROM Mirror Function Select Module

Figure 24.1-2 is a block diagram of the ROM mirror function select module.

Figure 24.1-2 Block Diagram of ROM Mirror Select Function



24.2 ROM Mirror Function Select Register (ROMM)

Do not access the ROM mirror function select register (ROMM) with address 008000_H to $00FFFF_H$ being used.

This register accepts only byte access. Do not access the register in words (whether to read from or write to it).

■ ROM Mirror Function Select Register (ROMM)

Figure 24.2-1 shows the bit configuration of the ROM mirror function select register (ROMM).

Figure 24.2-1 Bit Configuration of ROM Mirror Function Select Register (ROMM)

Bit	15	14	13	12	11	10	9	8	
Address: $00006F_H$	–	–	–	–	–	–	–	MI	ROMM
Read/Write →	–	–	–	–	–	–	–	W	
Initial value →	–	–	–	–	–	–	–	1	

[bit8] MI

Writing "1" to this bit allows ROM data in bank FF to be read from bank 00. Writing "0" to the bit disables this function via bank 00.

This bit is a write-only bit.

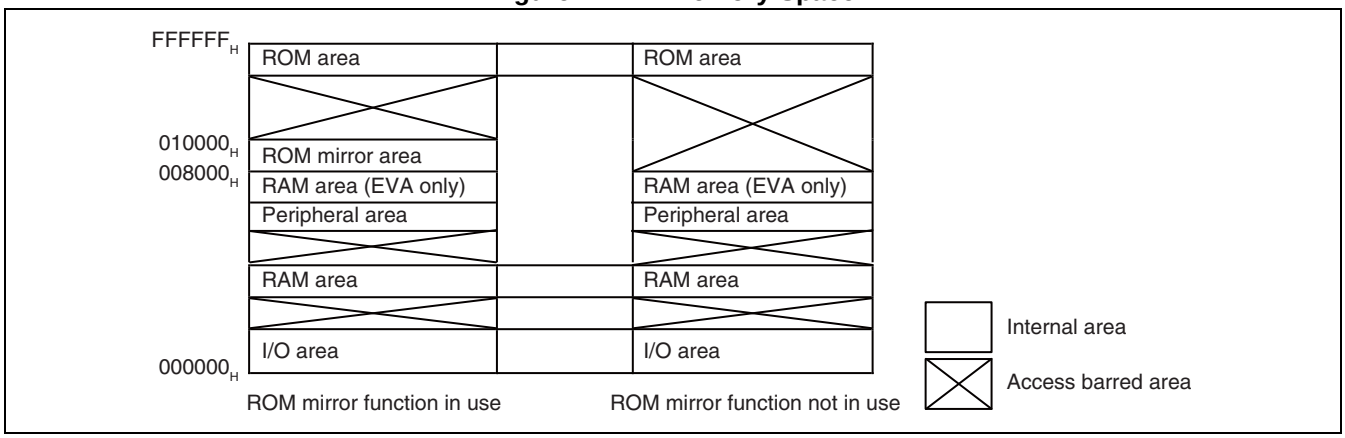
Note:

When the ROM mirror function is enabled, addresses 008000_H to $00FFFF_H$ in bank 00 mirror addresses $FF8000_H$ to $FFFFFF_H$. ROM addresses of $FF7FFF_H$ and below are not mirrored in bank 00 even if the ROM mirror function is enabled.

■ Memory Space

Figure 24.2-2 shows memory space.

Figure 24.2-2 Memory Space



25. 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".

25.1 Overview of Flash Memory

25.2 Sector Configuration of Flash Memory

25.3 Flash Memory Control Status Register (FMCS)

25.4 Flash Memory Write Control Registers (FWR0/FWR1)

25.5 Starting the Flash Memory Automatic Algorithm

25.6 Confirming the Automatic Algorithm Execution State

25.7 Writing Data to and Erasing Data from Flash Memory

25.8 Flash Security Function

25.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems

25.10 Notes on Using Flash Memory

Code: CM44-00107-1E

Page: 676, 678, 682, 682, 683, 704, 704, 708

25.1 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 : MB90F922NC, MB90F922NCS
- Capacity : 256K bytes / 128K words
- Sector configuration : 64 K × 2 + 48 K × 2 + 8 K × 4

● 3M-bit flash memory

- Compatible models : MB90F923NC, MB90F923NCS
- Capacity : 384K bytes / 192K words
- Sector configuration : 64 K × 5 + 48 K + 8 K × 2

● 4M-bit flash memory

- Compatible models : MB90F924NC, MB90F924NCS
- 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.

25.2 Sector Configuration of Flash Memory

This section shows the sector configuration of the flash memory.

■ Sector Configuration

Figure 25.2-1 to Figure 25.2-3 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 25.2-1 Sector Configuration of 2M-bit flash memory

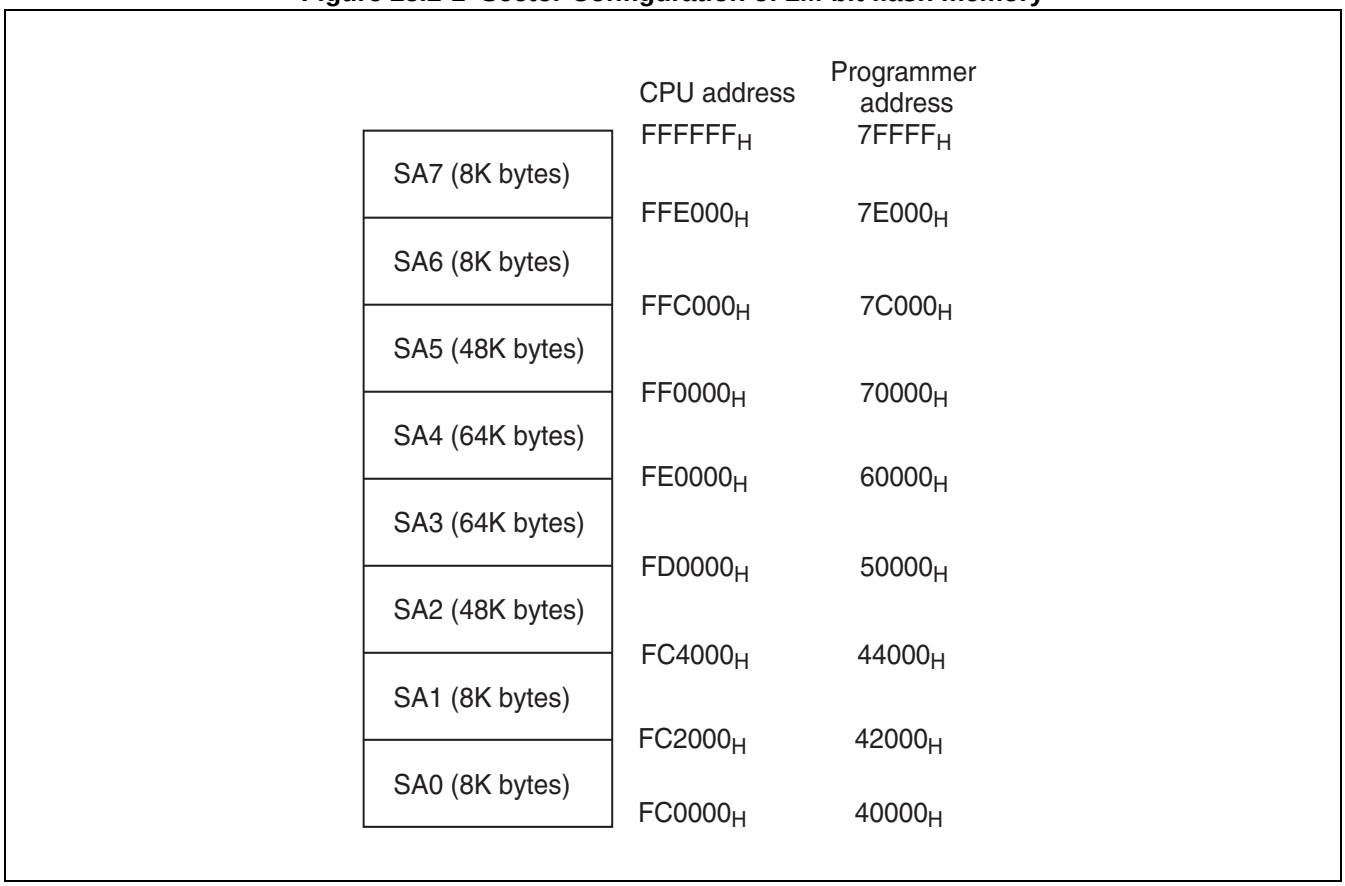


Figure 25.2-2 Sector Configuration of 3M-bit flash memory

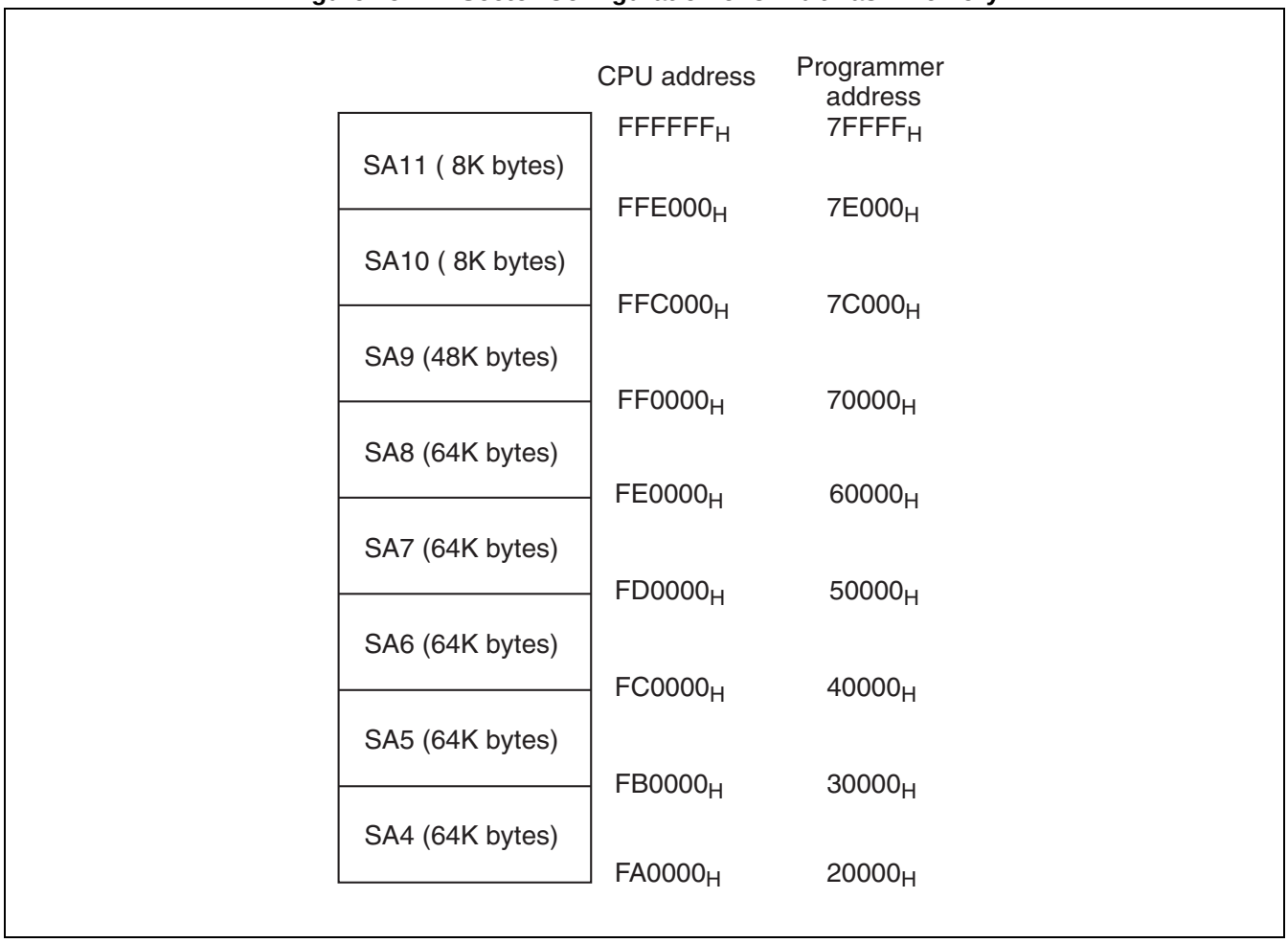
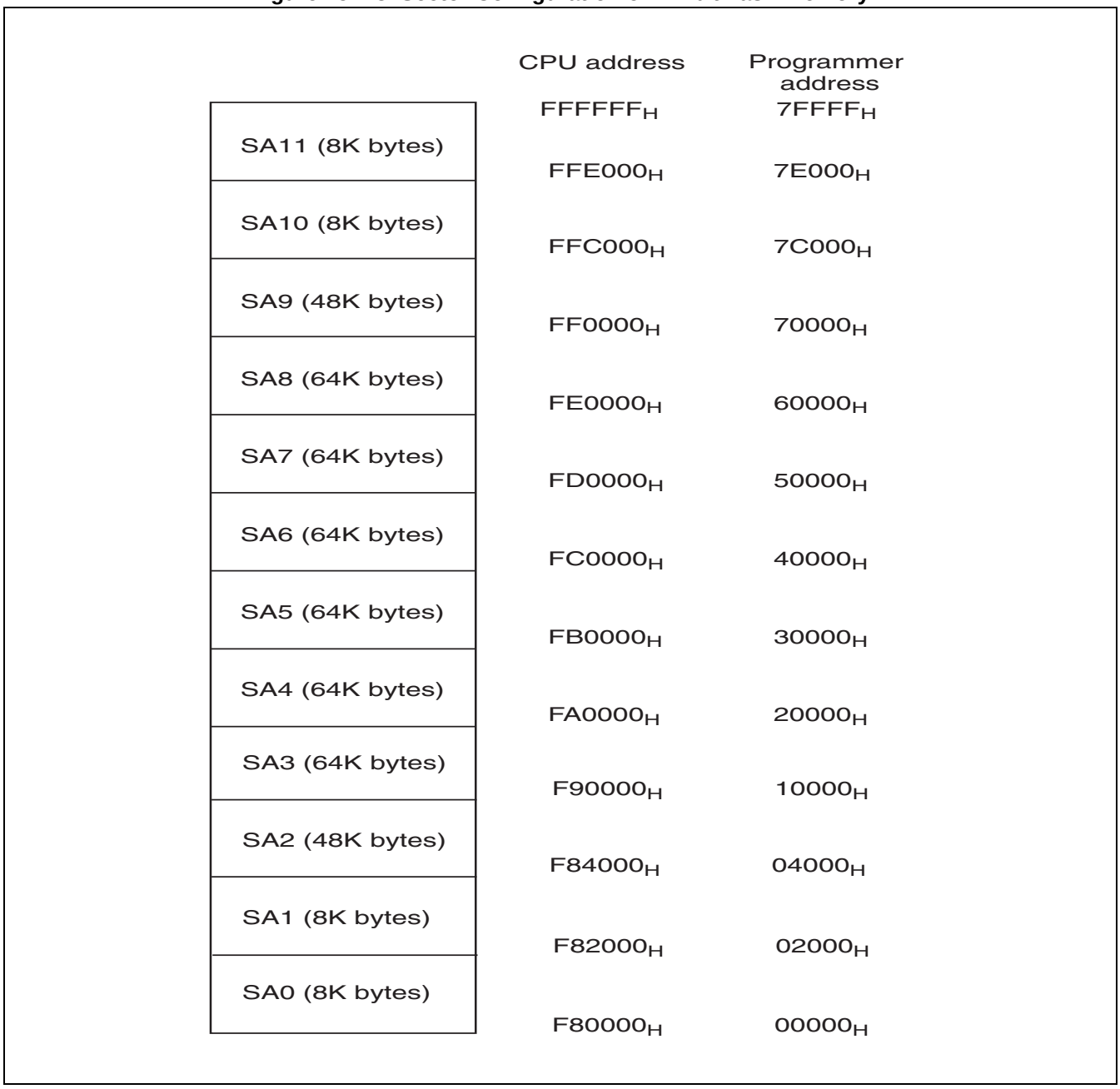


Figure 25.2-3 Sector Configuration of 4M-bit flash memory



- Programmer address

The programmer address in the Figure 25.2-1 to Figure 25.2-3 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.

25.3 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".

25.4 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.

■ 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 25.4-1 Flash Memory Write Control Registers (FWR0/FWR1)

FWR0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address: 0039A6$_H$	SA7E	SA6E	SA5E	SA4E	SA3E	SA2E	SA1E	SA0E
Read/Write→	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value→	0	0	0	0	0	0	0	0

FWR1	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
Address: 0039A7$_H$	Reserved	Reserved	Reserved	Reserved	SA11E	SA10E	SA9E	SA8E
Read/Write→	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value→	0	0	0	0	0	0	0	0

R/W : Readable/writable
0 : Write disabled [Initial value]

Table 25.4-1 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 25.4-2 Write-protection setup bits and corresponding flash sectors

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 25.4-3 Write-protection setup bits and corresponding flash sectors

Product with 3M-bit flash memory

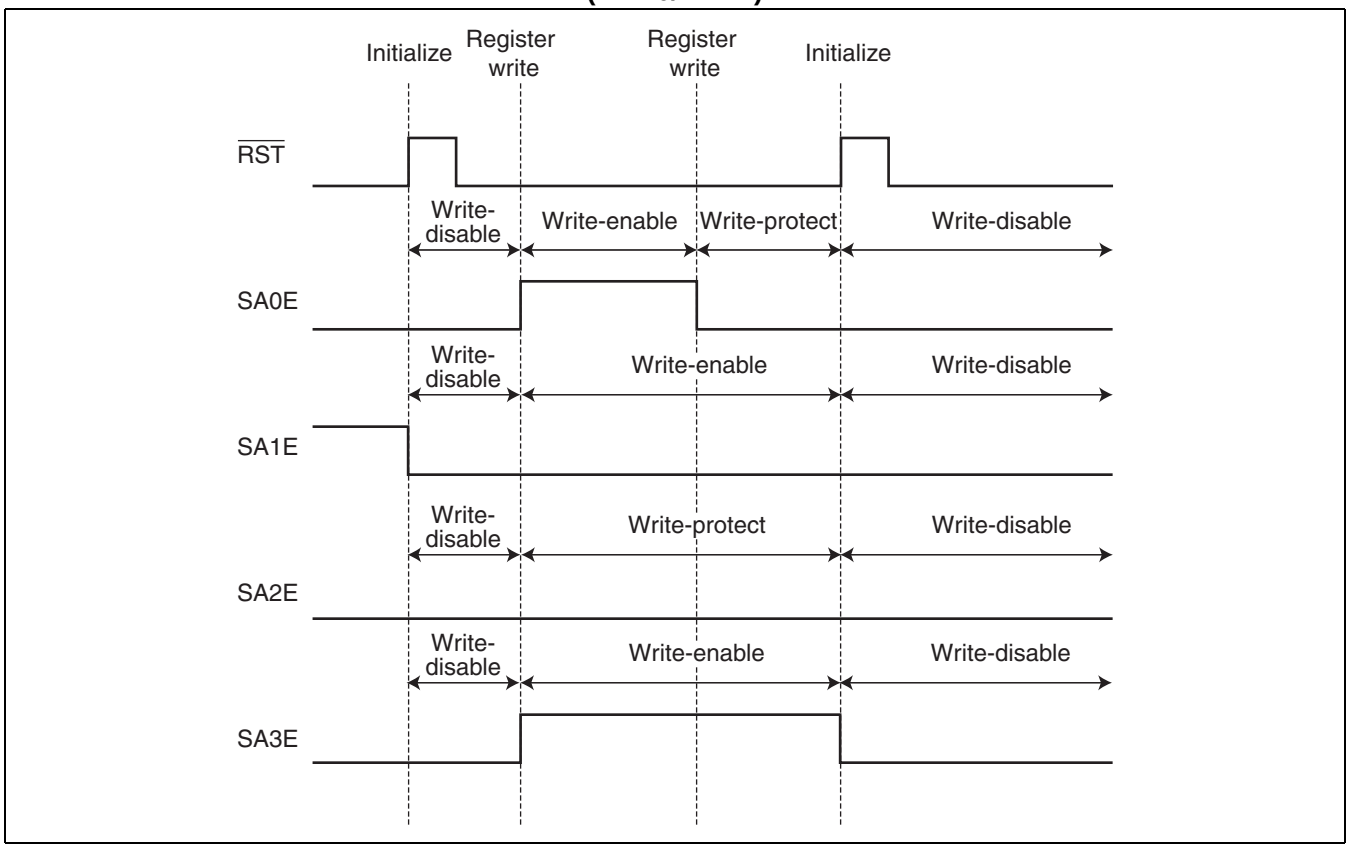
Bit	Bit name	Corresponding flash memory sector
15	Reserved	-
14	Reserved	-
13	Reserved	-
12	Reserved	-
11	SA11E	SA11
10	SA10E	SA10
9	SA9E	SA9
8	SA8E	SA8
7	SA7E	SA7
6	SA6E	SA6
5	SA5E	SA5
4	SA4E	SA4
3	Reserved	No corresponding sector available: Write "0" to these bits.
2	Reserved	
1	Reserved	
0	Reserved	Corresponding to the security bit: Write "1", only when writing to the security bit.

Table 25.4-4 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 25.4-2 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.

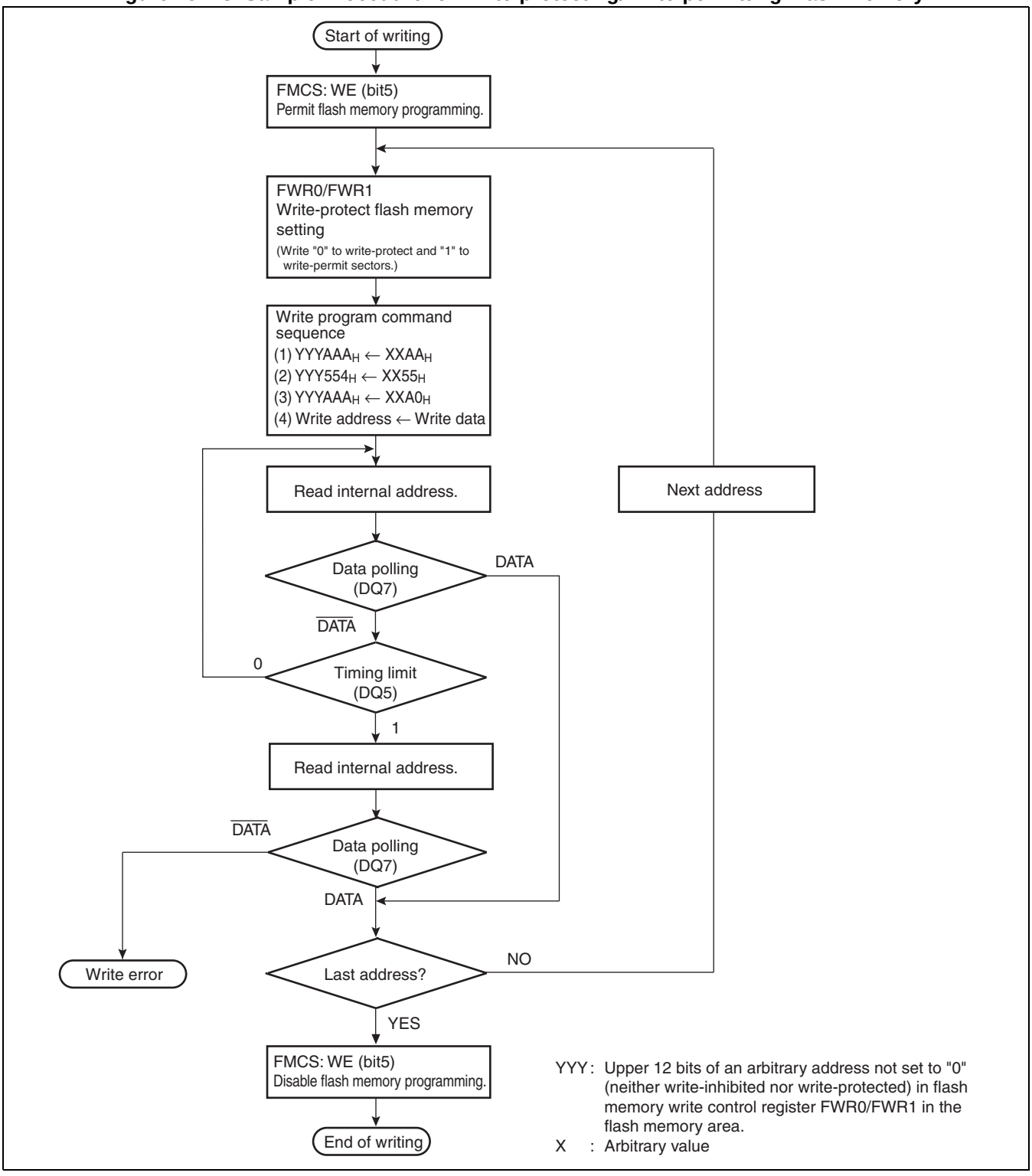
Write-protect:

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 25.4-3 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).

25.5 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 25.5-1 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 25.5-1 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 "25.2 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.

25.6 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 25.6-1 lists the bit assignments of the hardware sequence flags.

Table 25.6-1 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 25.5-1).

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 25.6-2 lists the functions of the hardware sequence flags.

Table 25.6-2 Hardware Sequence Flag Functions List

Status			DQ7	DQ6	DQ5	DQ3
Status change for normal operation	Data Write → Write completed (write address specified)		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

25.6.1 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 25.6-3 and Table 25.6-4.

Table 25.6-3 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 25.6-4 Data Polling Flag State Transition (State Change for Abnormal Operation)

Operating Status	Data write	Chip/sector Erase
DQ7	$\overline{DQ7}$	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 "25.9 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.

25.6.2 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 25.6-5 and Table 25.6-6.

Table 25.6-5 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 25.6-6 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.

25.6.3 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 25.6-7 and Table 25.6-8.

Table 25.6-7 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 25.6-8 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.

25.6.4 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 25.6-9 and Table 25.6-10.

Table 25.6-9 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 25.6-10 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.

25.7 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 25.5-1) 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

25.7.1 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 25.5-1) 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.

25.7.2 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 25.5-1) 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

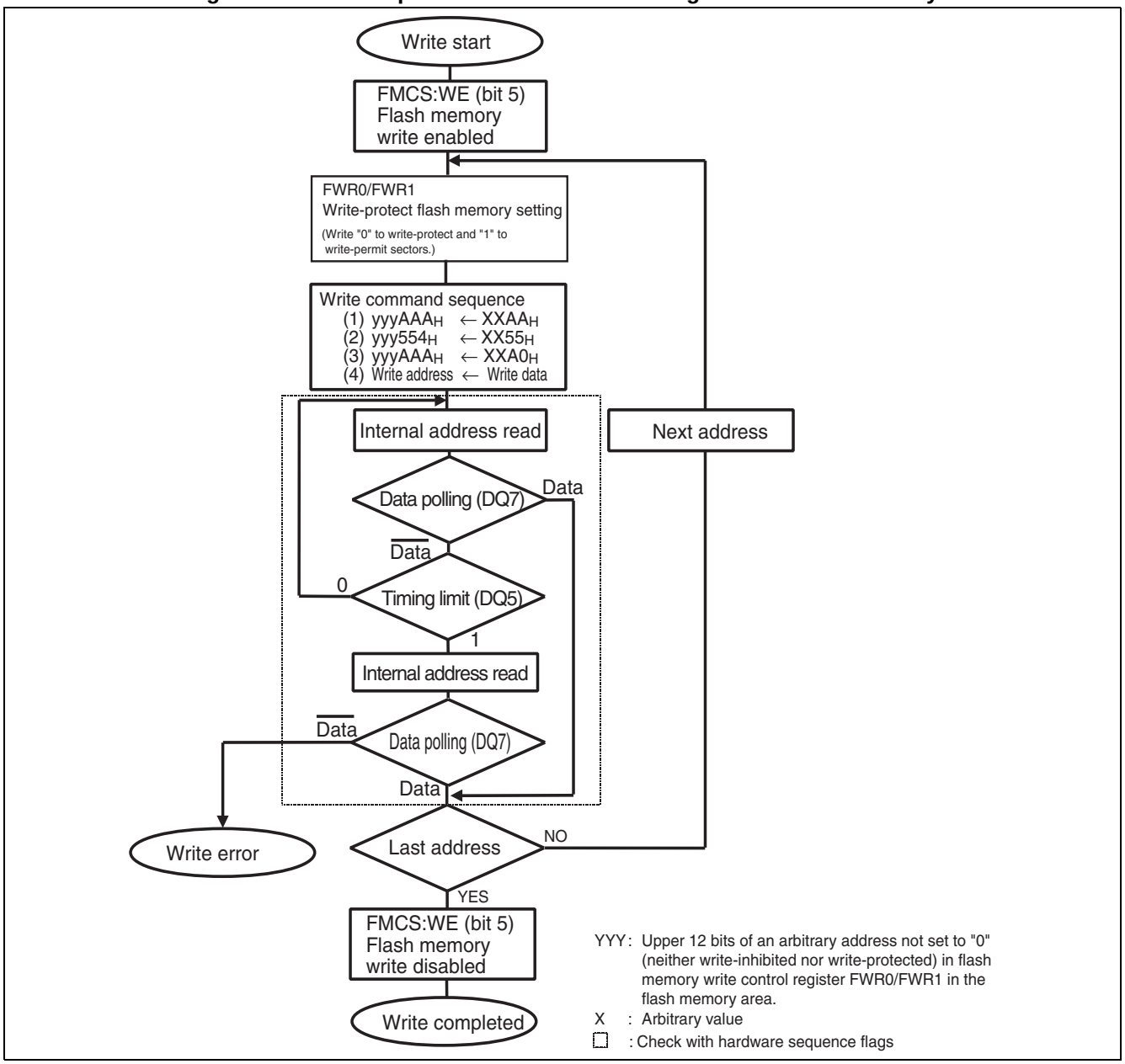
Figure 25.7-1 shows an example of the data-writing procedure. The hardware sequence flags (see Section "25.6 Confirming the Automatic Algorithm Execution State") can be used to determine the state of the automatic algorithm in the flash memory. In this procedure, the data polling flag (DQ7) is used to 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 25.7-1 Example of Procedure for Writing Data to Flash Memory



25.7.3 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 25.5-1) 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.

25.7.4 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 25.5-1) 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

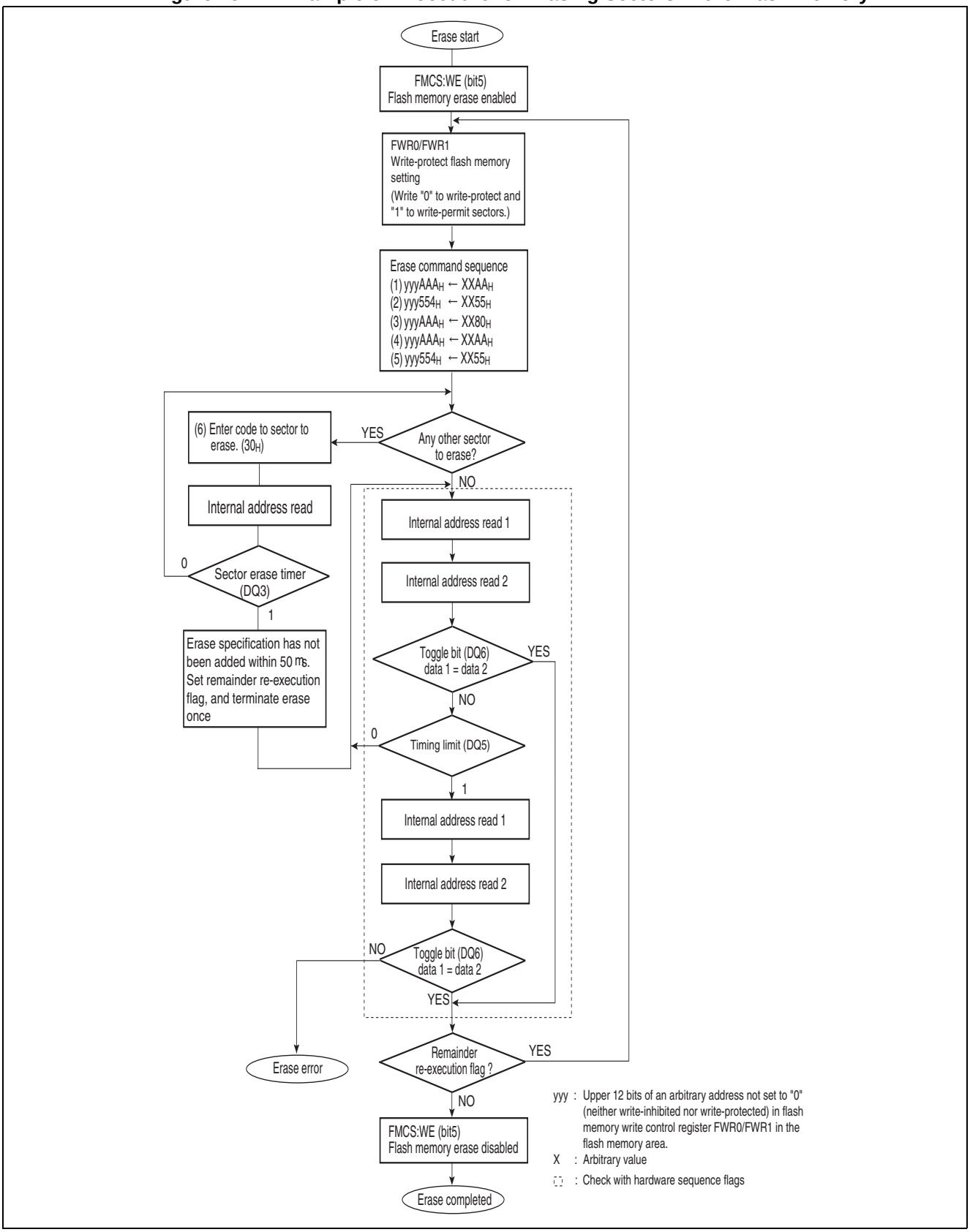
Figure 25.7-2 shows an example of the procedure for erasing sectors in the flash memory. Here, the toggle bit flag (DQ6) is used to confirm that erasing has terminated.

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 Figure 25.7-2).

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 25.7-2 Example of Procedure for Erasing Sectors in the Flash Memory



25.7.5 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 25.5-1) 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.

25.7.6 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 25.5-1) 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.

25.8 Flash Security Function

The flash security function enables to protect the contents in the flash memory.

■ 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 25.8-1 shows the address of security bit.

Table 25.8-1 Address and Protection Code of Flash Security Bit

	Flash memory size	Address of security bit	Protection code
MB90F922	2M-bit	$FC0001_H$	01_H
MB90F923/F924	3M/4M-bit	$F80001_H$	01_H

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.

25.9 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.

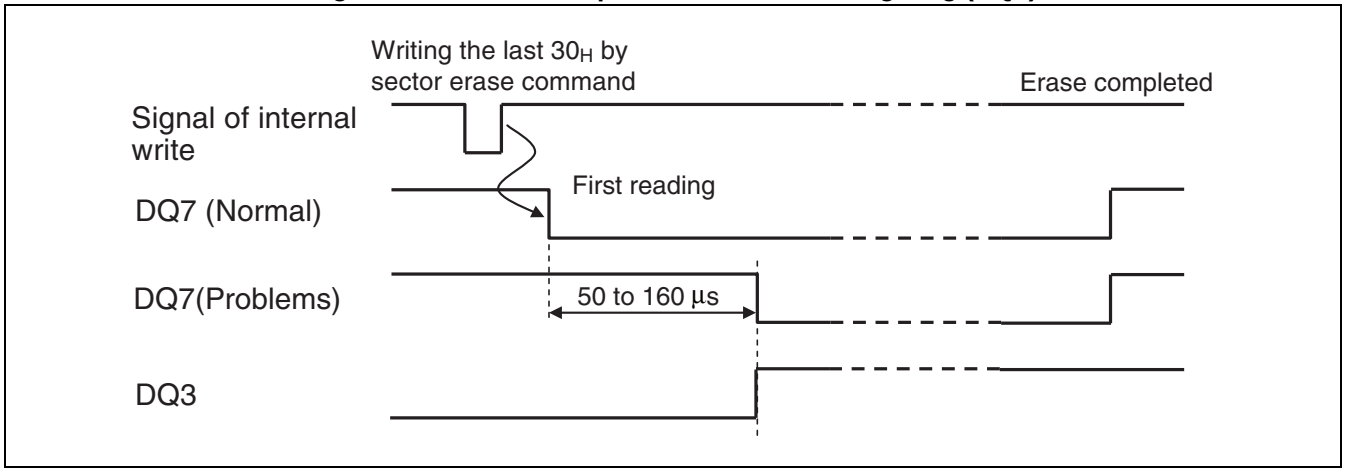
■ Description of Problems due to Restrictions

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 25.9-1, 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 25.9-1 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)

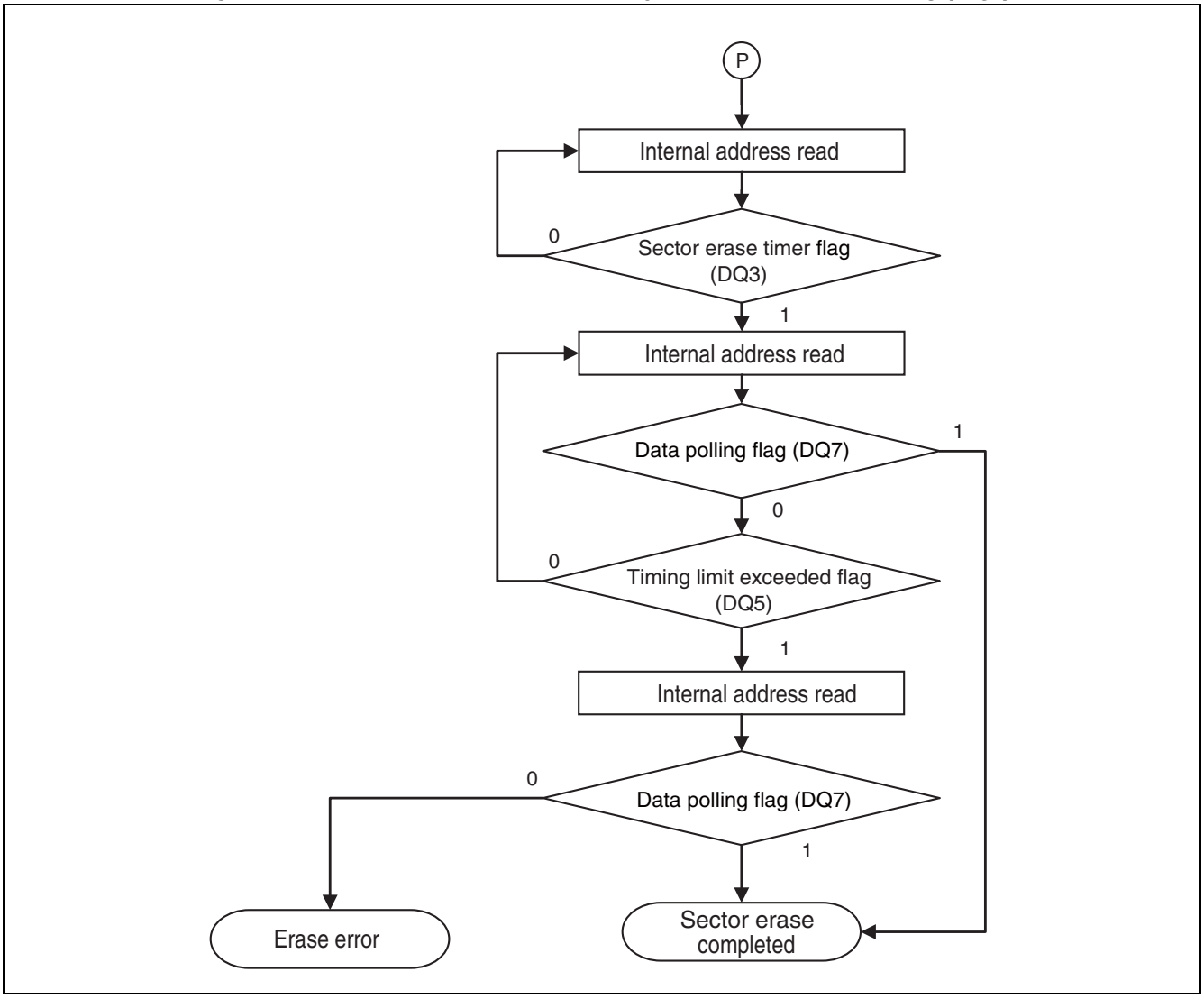
Determine the state of the automatic algorithm using DQ6, as shown in Figure 25.7-2 in "25.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase)"

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 25.9-2 shows the judgment method using DQ3 after the sector erase command is issued.

Figure 25.9-2 How to Avoid Problems by Sector Erase Timer Flag (DQ3)

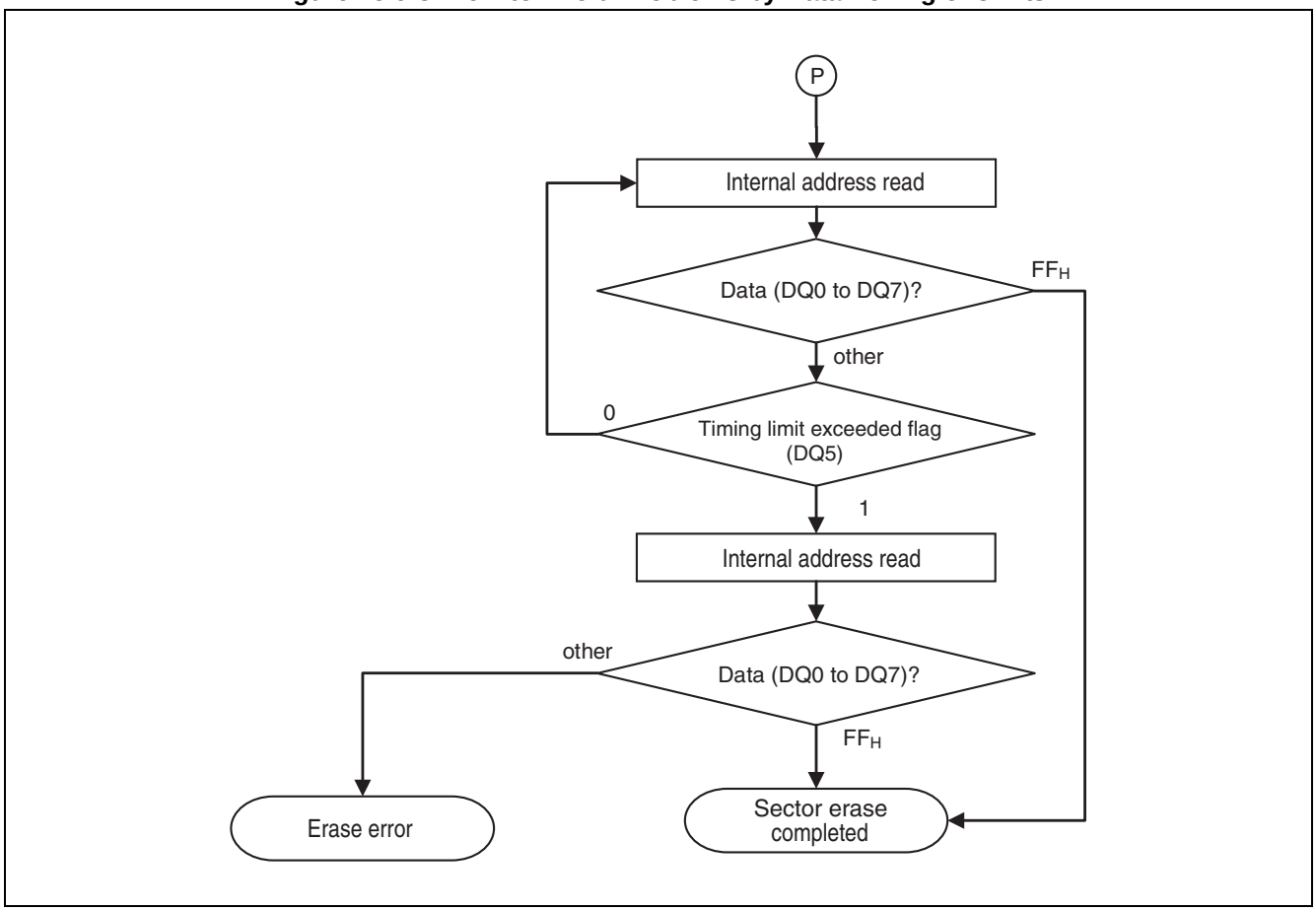


- Data polling using the 8 bits of hardware sequence flags

Make a judgment by data polling using the 8 bits of hardware sequence flags, rather than using only the polling of DQ7.

Figure 25.9-3 shows the judgment method using the data polling of the 8 bits after the sector erase command is issued.

Figure 25.9-3 How to Avoid Problems by Data Polling of 8 Bits



25.10 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 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.

26. Examples of Serial Programming Connection for Flash Memory Products

This chapter gives the examples of connection for serial programming using the AF220/AF210/AF120/AF110 Flash Microcontroller Programmer manufactured by Yokogawa Digital Computer Corporation.

26.1 Basic Configuration for Serial Programming Connection

26.2 Example of Connection in Single-Chip Mode (Using Power from User System)

26.3 Example of Connection with Flash Microcontroller Programmer (Using Power from the User System)

26.1 Basic Configuration for Serial Programming Connection

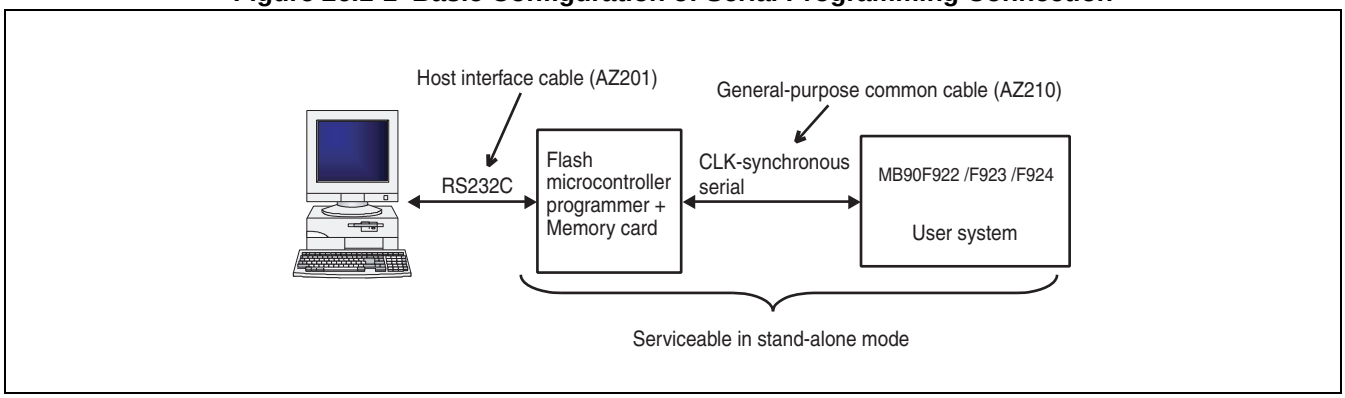
The Flash Memory products support serial onboard programming (Cypress standard) of flash ROM. This section provides the relevant specifications.

■ Basic Configuration for Serial Programming Connection

Cypress-standard serial onboard programming uses the AF220/AF210/AF120/AF110 flash microcontroller programmer manufactured by Yokogawa Digital Computer Corporation. It is possible to write it by selecting either of the program that operates by the single chip mode or the internal ROM external ROM bus mode.

Figure 26.1-1 shows the basic configuration of the connection for serial programming.

Figure 26.1-1 Basic Configuration of Serial Programming Connection



Note:

Contact Yokogawa Digital Computer Corporation for information on the functions and operating procedures of the AF220/AF210/AF120/AF110 flash microcontroller programmers and on their general-purpose common connection cable (AZ210) and connector.

■ Pins Used for Cypress Standard Serial On-board Writing

Table 26.1-1 shows the functions of the related pins used for Cypress Standard serial on-board writing.

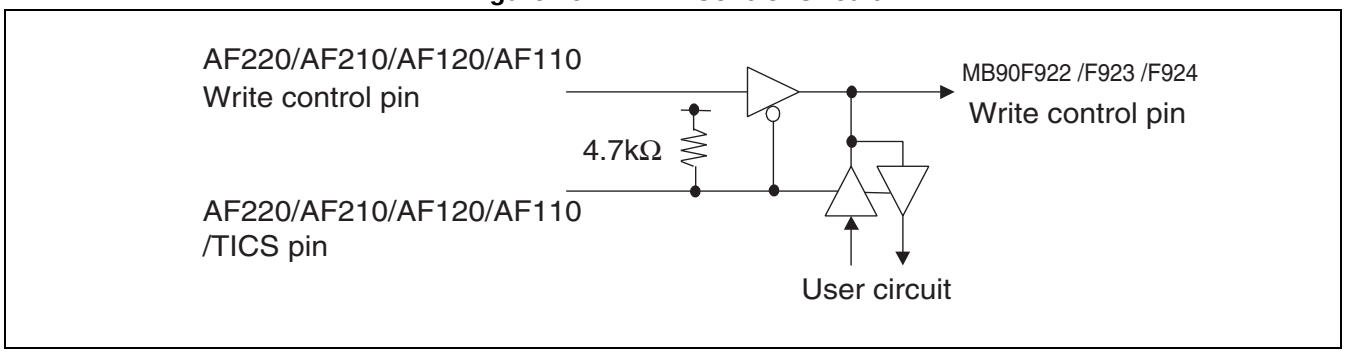
Table 26.1-1 Pins Used for Cypress-standard Serial Onboard Programming

Pin	Function	Description
MD2, MD1, MD0	Mode Pin	The serial programming mode is set according to the setting of MD2=1, MD1=1, and MD0=0.
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 32 MHz.
P00, P01	Programming program start pin	Input "L" level to P00, and "H" level to P01.
$\overline{RST}$	Reset pin	-
SIN1	Serial data input pin	Use UART1 in CLK synchronous mode.
SOT1	Serial data output pin	
SCK1	Serial clock input pin	
V_{CC}	Source voltage supply pin	If the programming voltage (V_{CC}=5.0V±10%) is supplied from the user system, connection with the flash microcontroller programmer is not required.
V_{SS}	GND pin	Must be shared with GND of the flash microcontroller programmer.

To use the P00, SIN1, SOT1, and SCK1 pins within the user system as well, the control circuit shown in the Figure 26.1-2 is required.

Using the flash microcontroller programmer's /TICS signal, the user circuit can be disconnected in serial programming mode. See the connection example.

Figure 26.1-2 Pin Control Circuit



■ Oscillator Clock Frequency and Serial Clock Input Frequency

The serial clock frequency that can be input can be derived by the formula shown below. Based on the oscillator clock frequency used, modify the serial clock input frequency in accordance with the oscillator clock frequency as required.

Calculate the serial clock frequency that can be input as follows:

Table 26.1-2 shows a calculation example.

Serial clock frequency = 0.125 × oscillator clock frequency

Table 26.1-2 Example of Serial Clock Frequency Calculation

Oscillator 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	1 MHz	850 kHz	500 kHz
16 MHz	2 MHz	1.25 MHz	500 kHz

■ System Configuration of Flash Microcontroller Programmer

Table 26.1-3 shows the system configuration of the flash microcontroller programmer.

Table 26.1-3 System Configuration of the Flash Microcontroller Programmer

Type		Function	
Main body	AF220/AC4P	Model with built-in Ethernet interface	/100V to 220V power adapter
	AF210/AC4P	Standard model	/100V to 220V power adapter
	AF120/AC4P	Model with built-in single key Ethernet interface	/100V to 220V power adapter
	AF110/AC4P	Single key model	/100V to 220V power adapter
AZ221		Programmer dedicated RS232C cable for PC/AT	
AZ210		Standard target probe (a) length: 1m	
FF201		Cypress F^2MC-16LX flash microcontroller control module	
AZ290		Remote controller	
/P2		2MB PC Card (Option) FLASH memory capacity up to 128 K bytes supported	
/P4		4MB PC Card (Option) FLASH memory capacity of up to 512 K bytes supported	

Inquiries: Yokogawa Digital Computer Corporation
Telephone number: (81)-42-333-6224

■ Examples of Serial Programming Connections

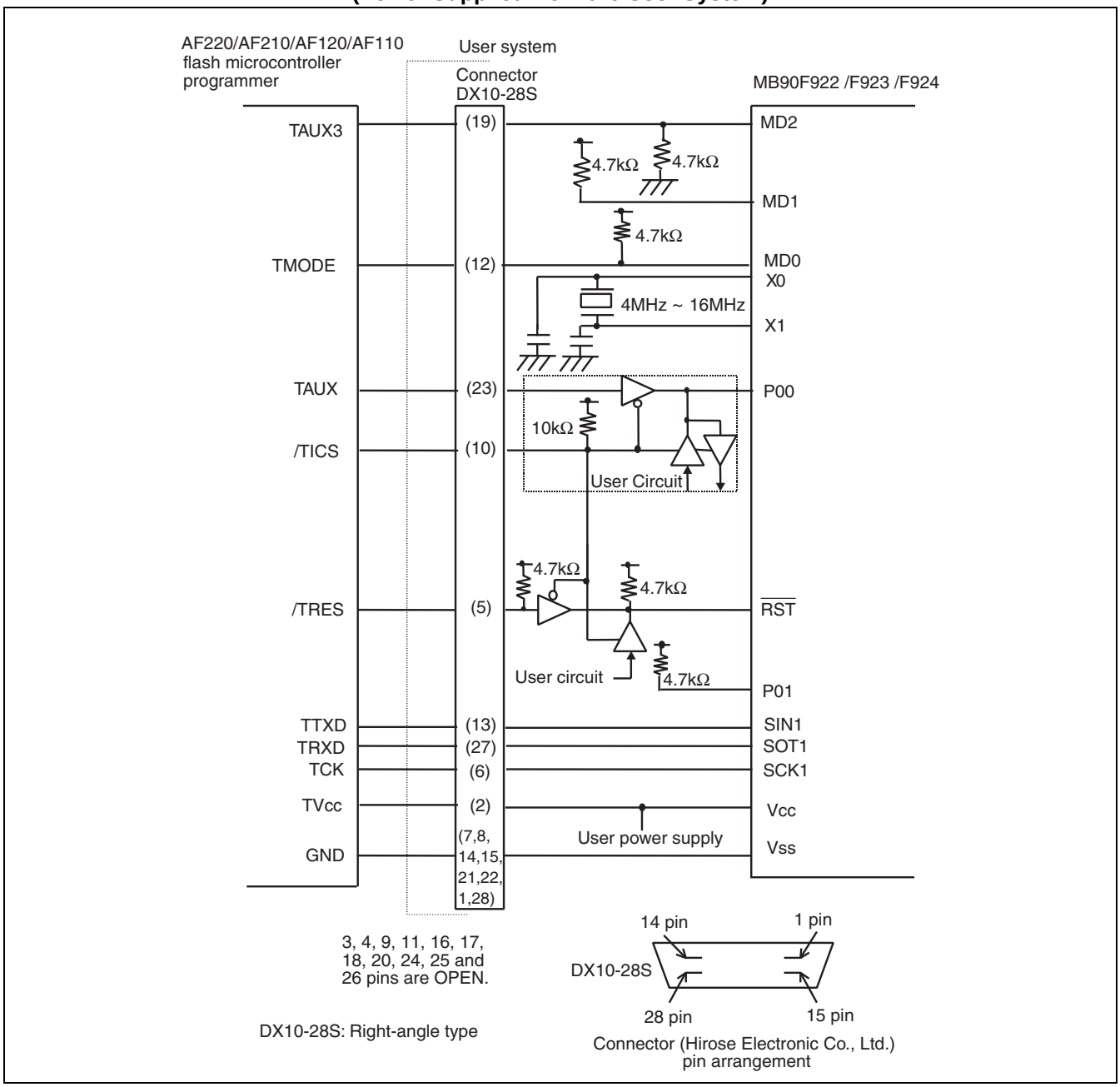
Examples for the following four types of connections are shown below.

- Example of connection in single-chip mode (using power from the user system)
- Example minimum connection with flash microcontroller programmer (using power from the user system)

26.2 Example of Connection in Single-Chip Mode (Using Power from User System)

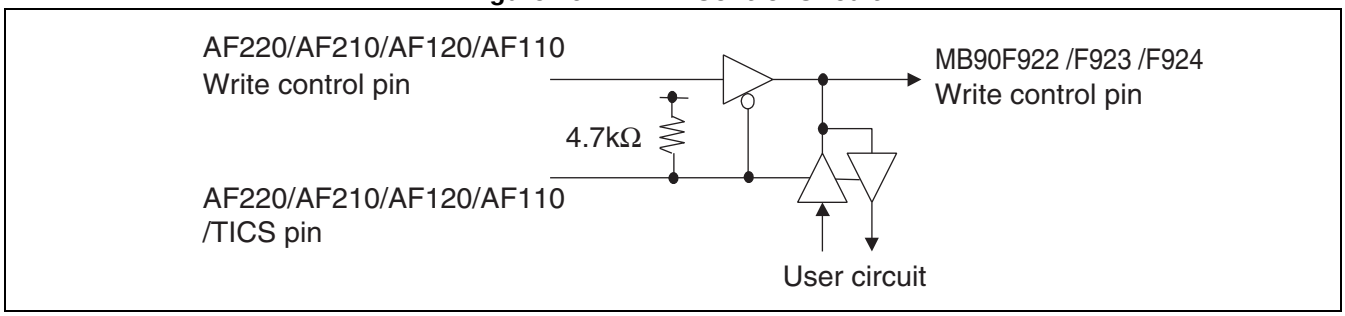
In the user system, mode pins MD2 and MD0, which are set to single-chip mode, are supplied with the inputs MD2=1 and MD0=0 by TAUX3 and TMODE of AF220/AF210/ AF120/AF110, and the system is set to serial programming mode (serial programming mode: MD2, MD1, MD0= 110).

Figure 26.2-1 Example of Serial Programming Connection in Single Chip Mode (Power Supplied from the User System)



Using the P00, SIN1, SOT1, and SCK1 pins in the user system requires a control circuit as shown in Figure 26.2-2 (the user circuit is disconnected in serial programming mode by outputting "L" in the flash microcontroller programmer's "/TICS" signal).

Figure 26.2-2 Pin Control Circuit

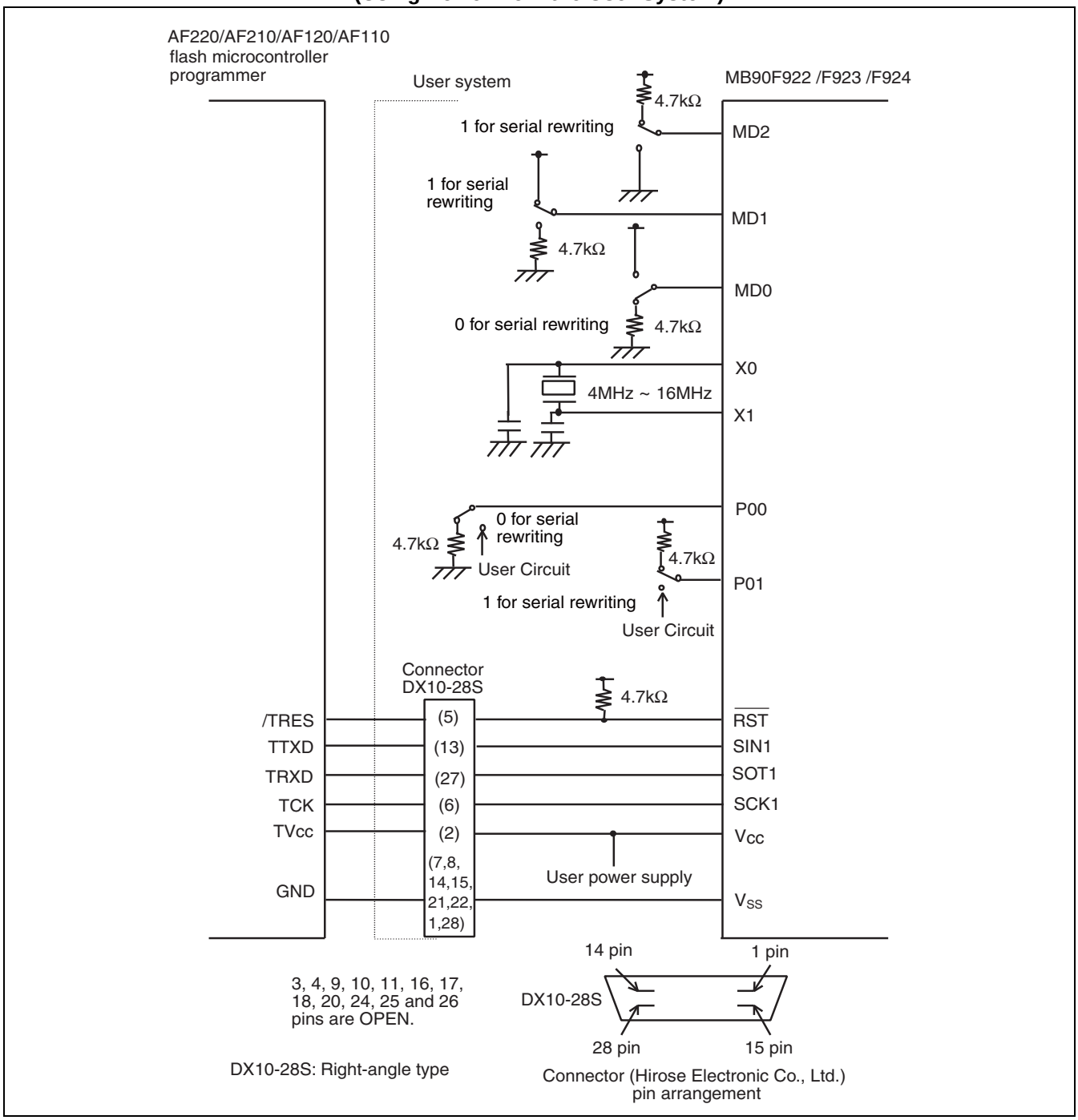


Connect to AF220/AF210/AF120/AF110 when the power of the user system is turned off.

26.3 Example of Connection with Flash Microcontroller Programmer (Using Power from the User System)

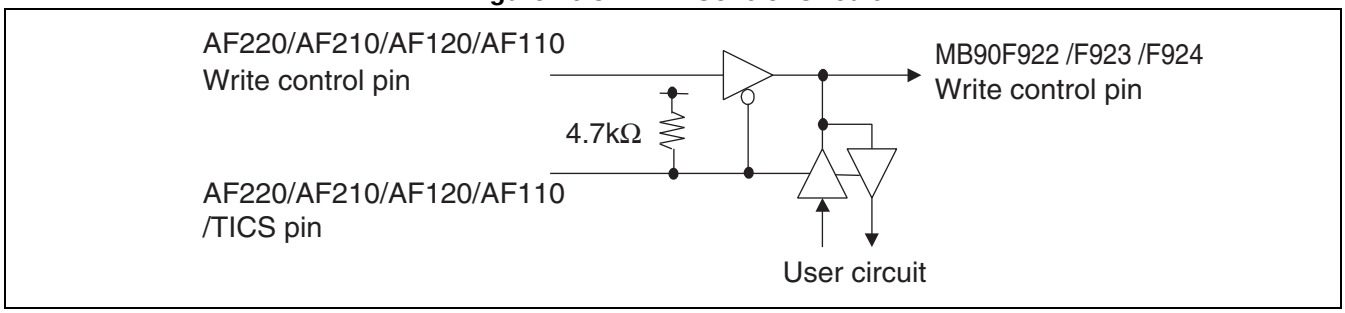
If, in serial programming mode, pins (MD2, MD0 and P00) are set as shown below, MD2, MD0, and P00 do not need to be connected with the flash microcontroller programmer.

Figure 26.3-1 Example of Minimum Connection with Flash Microcontroller Programmer (Using Power from the User System)



Using the P00, SIN1, SOT1, and SCK1 pins in the user system requires a control circuit as shown in Figure 26.3-2 (the user circuit is disconnected in serial programming mode by outputting "L" in the flash microcontroller programmer's "/TICS" signal).

Figure 26.3-2 Pin Control Circuit



Connect to AF220/AF210/AF120/AF110 when the power of the user system is turned off.

27. ROM Security Function

This chapter explains the ROM security function.

27.1 Overview of ROM Security Function

27.1 Overview of ROM Security Function

The ROM security function protects the content of ROM.

■ Overview of ROM Security Function

The ROM security function is to avoid disclosing the ROM data to third parties by restricting the access to ROM.

Please contact sales representatives for details of this function.

28. Appendix

The appendix provides the I/O map and describes the available instructions.

APPENDIX A I/O Maps
APPENDIX B Instructions
APPENDIX C Main Changes

APPENDIX A I/O Maps

The registers of individual peripheral resources are assigned the following addresses. Table A-1, Table A-2, and Table A-3 list the addresses assigned for the registers of individual peripheral resources.

Table A-1 I/O Map (Sheet 1 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
00_H	Port 0 data register	PDR0	R/W	Port 0	XXXXXXXX_B
01_H	Port 1 data register	PDR1	R/W	Port 1	XXXXXXXX_B
02_H	Port 2 data register	PDR2	R/W	Port 2	XXXXXXXX_B
03_H	Port 3 data register	PDR3	R/W	Port 3	XXXXXXXX_B
04_H	Port 4 data register	PDR4	R/W	Port 4	XXXXXXXX_B
05_H	Port 5 data register	PDR5	R/W	Port 5	XXXXXXXX_B
06_H	Port 6 data register	PDR6	R/W	Port 6	XXXXXXXX_B
07_H	Port 7 data register	PDR7	R/W	Port 7	XXXXXXXX_B
08_H	Port 8 data register	PDR8	R/W	Port 8	XXXXXXXX_B
09_H	Port 9 data register	PDR9	R/W	Port 9	XXXXXXXX_B
$0A_H$ to $0B_H$	(Not available)				
$0C_H$	Port C data register	PDRC	R/W	Port C	XXXXXXXX_B
$0D_H$	Port D data register	PDRD	R/W	Port D	XXXXXXXX_B
$0E_H$	Port E data register	PDRE	R/W	Port E	XXXXXXXX_B
$0F_H$	(Not available)				
10_H	Port 0 direction register	DDR0	R/W	Port 0	00000000_B
11_H	Port 1 direction register	DDR1	R/W	Port 1	$XX000000_B$
12_H	Port 2 direction register	DDR2	R/W	Port 2	$000000XX_B$
13_H	Port 3 direction register	DDR3	R/W	Port 3	00000000_B
14_H	Port 4 direction register	DDR4	R/W	Port 4	00000000_B
15_H	Port 5 direction register	DDR5	R/W	Port 5	00000000_B
16_H	Port 6 direction register	DDR6	R/W	Port 6	00000000_B
17_H	Port 7 direction register	DDR7	R/W	Port 7	00000000_B
18_H	Port 8 direction register	DDR8	R/W	Port 8	00000000_B

Table A-1 I/O Map (Sheet 2 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
19_H	Port 9 direction register	DDR9	R/W	Port 9	$X0000000_B$
$1A_H$	Analog input enable	ADER6	R/W	Port 6, A/D	11111111_B
$1B_H$	(Not available)				
$1C_H$	Port C direction register	DDRC	R/W	Port C	00000000_B
$1D_H$	Port D direction register	DDRD	R/W	Port D	$X0000000_B$
$1E_H$	Port E direction register	DDRE	R/W	Port E	$XXXXX000_B$
$1F_H$	(Not available)				
20_H	A/D control status register lower	ADCS0	R/W	A/D converter	$000XXXX0_B$
21_H	A/D control status register upper	ADCS1	R/W		$0000000X_B$
22_H	A/D data register lower	ADCR0	R		00000000_B
23_H	A/D data register upper	ADCR1	R		$XXXXXX00_B$
24_H	Compare clear register	CPCLR	R/W	16bit free-run timer	XXXXXXXX_B
25_H			R/W		XXXXXXXX_B
26_H	Timer data register	TCDT	R/W		00000000_B
27_H			R/W		00000000_B
28_H	Timer control status register lower	TCCSL	R/W		00000000_B
29_H	Timer control status register upper	TCCSH	R/W		$01\text{-}00000_B$
$2A_H$	PPG0 control status register lower	PCNTL0	R/W	16bit PPG0	00000000_B
$2B_H$	PPG0 control status register upper	PCNTH0	R/W		00000001_B
$2C_H$	PPG1 control status register lower	PCNTL1	R/W	16bit PPG1	00000000_B
$2D_H$	PPG1 control status register upper	PCNTH1	R/W		00000001_B
$2E_H$	PPG2 control status register lower	PCNTL2	R/W	16bit PPG2	00000000_B
$2F_H$	PPG2 control status register upper	PCNTH2	R/W		00000001_B
30_H	External interrupt enable	ENIR	R/W	External interrupt	00000000_B
31_H	External interrupt request	EIRR	R/W		00000000_B
32_H	External interrupt level lower	ELVRL	R/W		00000000_B
33_H	External interrupt level upper	ELVRH	R/W		00000000_B

Table A-1 I/O Map (Sheet 3 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
34_H	Serial mode register 0	SMR0	R/W, W	UART (LIN/SCI) 0	00000000_B
35_H	Serial control register 0	SCR0	R/W, W		00000000_B
36_H	Reception/Transmission data register 1	RDR0/ TDR0	R/W		00000000_B
37_H	Serial status register 0	SSR0	R/W, R		00001000_B
38_H	Extended Communication Control register 0	ECCR0	R/W, R, W		$000000XX_B$
39_H	Extended status Control Register 0	ESCR0	R/W		00000100_B
$3A_H$	Baud Rate Generator Register 00	BGR00	R/W		00000000_B
$3B_H$	Baud Rate Generator Register 01	BGR01	R/W, R		00000000_B
$3C_H$ to $3E_H$	(Not available)				
$3F_H$	(Not available)				
40_H	Message buffer valid area	BVALR	(R/W)	CAN0	00000000_B
41_H					00000000_B
42_H	Transmission request register	TREQR	(R/W)		00000000_B
43_H					00000000_B
44_H	Transmission cancel register	TCANR	(W)		00000000_B
45_H					00000000_B
46_H	Transmission completed register	TCR	(R/W)		00000000_B
47_H					00000000_B
48_H	Receiving completed register	RCR	(R/W)		00000000_B
49_H					00000000_B
$4A_H$	Remote request receiving register	RRTRR	(R/W)	CAN0	00000000_B
$4B_H$					00000000_B
$4C_H$	Receiving overrun register	ROVRR	(R/W)		00000000_B
$4D_H$					00000000_B
$4E_H$	Receiving interrupt enable register	RIER	(R/W)		00000000B
$4F_H$					00000000B

Table A-1 I/O Map (Sheet 4 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
50_H	Timer control status register 0 lower	TMCSR0L	R/W	16bit reload timer 0	00000000_B
51_H	Timer control status register 0 upper	TMCSR0H	R/W		$XXX10000_B$
52_H	Timer register 0/Reload register 0	TMR0/ TMRLR0	R/W		XXXXXXXX_B
53_H					XXXXXXXX_B
54_H	Timer control status register 1 lower	TMCSR1L	R/W	16bit reload timer 0	00000000_B
55_H	Timer control status register 1 upper	TMCSR1H	R/W		$XXX10000_B$
56_H	Timer register 1/Reload register 1	TMR1/ TMRLR1	R/W		XXXXXXXX_B
57_H					XXXXXXXX_B
58_H	LCD output control register 1	LOCR1	R/W	LCDC	11111111_B
59_H	LCD output control register 2	LOCR2	R/W		00000000_B
$5A_H$	Sound control register 0 lower	SGCRL0	R/W	Sound generator 0	00000000_B
$5B_H$	Sound control register 0 upper	SGCRH0	R/W		$0XXXX100_B$
$5C_H$	Frequency data register 0	SGFR0	R/W		XXXXXXXX_B
$5D_H$	Amplitude data register 0	SGAR0	R/W		00000000_B
$5E_H$	Decrement grade register 0	SGDR0	R/W	Sound generator 0	XXXXXXXX_B
$5F_H$	Tone count register 0	SGTR0	R/W		XXXXXXXX_B
60_H	Input capture register 0	IPCP0	R	Input capture 0/1	XXXXXXXX_B
61_H					XXXXXXXX_B
62_H	Input capture register 1	IPCP1	R		XXXXXXXX_B
63_H					XXXXXXXX_B
64_H	Input capture register 2	IPCP2	R	Input capture 2/3	XXXXXXXX_B
65_H					XXXXXXXX_B
66_H	Input capture register 3	IPCP3	R		XXXXXXXX_B
67_H					XXXXXXXX_B
68_H	Input capture control status 0/1	ICS01	R/W	Input capture 0/1	00000000_B
69_H	Input Capture Edge Register 01	ICE01	R/W		$XXX0X0XX_B$
$6A_H$	Input capture control status 2/3	ICS23	R/W	Input capture 2/3	00000000_B
$6B_H$	Input Capture Edge Register 23	ICE23	R/W		XXXXXXXX_B
$6C_H$	LCD control register lower	LCRL	R/W	LCD controller/ Driver	00010000_B
$6D_H$	LCD control register upper	LCRH	R/W		00000000_B
$6E_H$	Low voltage/CPU operation detection reset control register	LVRC	R/W	Low voltage/CPU operation detection reset	00111000_B

Table A-1 I/O Map (Sheet 5 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$6F_H$	ROM mirror	ROMM	W	ROM mirror	XXXXXXX1B
70_H	Message buffer valid area	BVALR	(R/W)	CAN1	00000000_B
71_H					00000000_B
72_H	Transmission request register	TREQR	(R/W)		00000000_B
73_H					00000000_B
74_H	Transmission cancel register	TCANR	(W)		00000000_B
75_H					00000000_B
76_H	Transmission completed register	TCR	(R/W)		00000000_B
77_H					00000000_B
78_H	Receiving completed register	RCR	(R/W)		00000000_B
79_H					00000000_B
$7A_H$	Remote request receiving register	RRTRR	(R/W)		00000000_B
$7B_H$					00000000_B
$7C_H$	Receiving overrun register	ROVRR	(R/W)		00000000_B
$7D_H$					00000000_B
$7E_H$	Receiving interrupt enable register	RIER	(R/W)		00000000_B
$7F_H$					00000000_B
80_H	PWM control register 0	PWC0	R/W	Stepping motor controller 0	$000000X0_B$
81_H	(Not available)				
82_H	PWM control register 1	PWC1	R/W	Stepping motor controller 1	$000000X0_B$
83_H	(Not available)				
84_H	PWM control register 2	PWC2	R/W	Stepping motor controller 2	$000000X0_B$
85_H	(Not available)				
86_H	PWM control register 3	PWC3	R/W	Stepping motor controller 3	$000000X0_B$
87_H	(Not available)				
88_H	LCD output control register 1	LOCR3	R/W	LCDC	$XXXXX111_B$
89_H	(Not available)				
$8A_H$	A/D setting register 0	ADSR0	R/W	A/D	00000000_B
$8B_H$	A/D setting register 1	ADSR1	R/W		00000000_B

Table A-1 I/O Map (Sheet 6 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$8C_H$	Port input Level Select 0	PIL0	R/W	Port Input Level Select	00000000_B
$8D_H$	Port input Level Select 1	PIL1	R/W		$XXXX0000_B$
$8E_H$	Port input Level Select 2	PIL2	R/W		$XXXX0000_B$
$8F_H$ to $9D_H$	(Not available)				
$9E_H$	Program address detection control register	PACSR	R/W	Address match detection function	$XXXX0X0X_B$
$9F_H$	Delay interrupt/release register	DIRR	R/W	Delayed interrupt	$XXXXXXX0_B$
$A0_H$	Low-power mode control register	LPMCR	R/W	Low-power consumption control circuit	00011000_B
$A1_H$	Clock select register	CKSCR	R/W, R		11111100_B
$A2_H$ to $A7_H$	(Not available)				
$A8_H$	Watchdog timer control register	WDTC	R, W	Watchdog timer	$XXXXX111_B$
$A9_H$	Time base timer control register	TBTC	R/W, W	Time base timer	$1XX00100_B$
AA_H	Watchdog timer control register	WTC	R/W, W, R	Watch timer (sub clock)	$1X000000_B$
AB_H to AD_H	(Not available)				
AE_H	FLASH control register	FMCS	R/W	Flash Memory interface	$000X0000_B$
AF_H	(Not available)				

Table A-1 I/O Map (Sheet 7 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$B0_H$	Interrupt control register 00	ICR00	R/W	Interrupt controller	00000111_B
$B1_H$	Interrupt control register 01	ICR01	R/W		00000111_B
$B2_H$	Interrupt control register 02	ICR02	R/W		00000111_B
$B3_H$	Interrupt control register 03	ICR03	R/W		00000111_B
$B4_H$	Interrupt control register 04	ICR04	R/W		00000111_B
$B5_H$	Interrupt control register 05	ICR05	R/W		00000111_B
$B6_H$	Interrupt control register 06	ICR06	R/W		00000111_B
$B7_H$	Interrupt control register 07	ICR07	R/W		00000111_B
$B8_H$	Interrupt control register 08	ICR08	R/W		00000111_B
$B9_H$	Interrupt control register 09	ICR09	R/W		00000111_B
BA_H	Interrupt control register 10	ICR10	R/W		00000111_B
BB_H	Interrupt control register 11	ICR11	R/W		00000111_B
BC_H	Interrupt control register 12	ICR12	R/W		00000111_B
BD_H	Interrupt control register 13	ICR13	R/W		00000111_B
BE_H	Interrupt control register 14	ICR14	R/W		00000111_B
BF_H	Interrupt control register 15	ICR15	R/W		00000111_B
$C0_H$ to $C3_H$	(Not available)				
$C4_H$	Serial mode register 1	SMR1	R/W, W	UART (LIN/SCI) 1	00000000_B
$C5_H$	Serial control register 1	SCR1	R/W, W		00000000_B
$C6_H$	Reception/Transmission data register 1	RDR1/TDR1	R/W		00000000_B
$C7_H$	Serial status register 1	SSR1	R/W, R		00001000_B
$C8_H$	Extended Communication Control Register 1	ECCR1	R/W, R		$000000XX_B$
$C9_H$	Extended Status Control Register 1	ESCR1	R/W		00000100_B
CA_H	Baud Rate Generator Register 10	BGR10	R/W		00000000_B
CB_H	Baud Rate Generator Register 10	BGR11	R/W, R		00000000_B
CC_H	Watch timer control register lower	WTCRL	R/W	Real-time watch Timer	$000XXXX0_B$
CD_H	Watch timer control register middle	WTCRM	R/W		00000000_B
CE_H	Watch timer control register upper	WTCRH	R/W		$XXXXXX00_B$
CF_H	Sub clock Control register	PSCCR	W	Sub clock	$XXXX0000_B$

Table A-1 I/O Map (Sheet 8 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$D0_H$	Input capture control status 4/5	ICS45	R/W	Input capture 4/5	00000000_B
$D1_H$	Input Capture Edge Register 45	ICE45	R/W, R		XXXXXXXX_B
$D2_H$	Input capture control status 6/7	ICS67	R/W	Input capture 6/7	00000000_B
$D3_H$	Input Capture Edge Register 67	ICE67	R/W, R		$XXX0X0XX_B$
$D4_H$	Timer control status register 2 lower	TMCSR2L	R/W	16bit reload timer 2	00000000_B
$D5_H$	Timer control status register 2 upper	TMCSR2H	R/W		$XXX10000_B$
$D6_H$	Timer control status register 3 lower	TMCSR3L	R/W	16bit reload timer 3	00000000_B
$D7_H$	Timer control status register 3 upper	TMCSR3H	R/W		$XXX10000_B$
$D8_H$	Sound control register 1 lower	SGCRL1	R/W	Sound generator 1	00000000_B
$D9_H$	Sound control register 1 upper	SGCRH1	R/W		$0XXXX100_B$
DA_H	PPG3 control status register lower	PCNTL3	R/W	16bit PPG3	00000000_B
DB_H	PPG3 control status register upper	PCNTH3	R/W		00000001_B
DC_H	PPG4 control status register lower	PCNTL4	R/W	16bit PPG4	00000000_B
DD_H	PPG4 control status register upper	PCNTH4	R/W		00000001_B
DE_H	PPG5 control status register lower	PCNTL5	R/W	16bit PPG5	00000000_B
DF_H	PPG5 control status register upper	PCNTH5	R/W		00000001_B
$E0_H$	Serial mode register 2	SMR2	R/W, W	UART (LIN/SCI) 2	00000000_B
$E1_H$	Serial control register 2	SCR2	R/W, W		00000000_B
$E2_H$	Reception/Transmission data register 2	RDR2/TDR2	R/W		00000000_B
$E3_H$	Serial status register 2	SSR2	R/W, R		00001000_B
$E4_H$	Extended Communication Control register 2	ECCR2	R/W, R		$000000XX_B$
$E5_H$	Extended status Control Register 2	ESCR2	R/W		00000100_B
$E6_H$	Baud Rate Generator Register 20	BGR20	R/W		00000000_B
$E7_H$	Baud Rate Generator Register 21	BGR21	R/W, R		00000000_B

Table A-1 I/O Map (Sheet 9 of 9)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$E8_H$	Serial mode register 3	SMR3	R/W, W	UART (LIN/SCI) 3	00000000_B
$E9_H$	Serial control register 3	SCR3	R/W, W		00000000_B
EA_H	Reception/Transmission data register 3	RDR3/TDR3	R/W		00000000_B
EB_H	Serial status register 3	SSR3	R/W, R		00001000_B
EC_H	Extended Communication Control register 3	ECCR3	R/W, R		$000000XX_B$
ED_H	Extended status Control Register 3	ESCR3	R/W	UART (LIN/SCI) 3	00000100_B
EE_H	Baud Rate Generator Register 30	BGR30	R/W		00000000_B
EF_H	Baud Rate Generator Register 31	BGR31	R/W, R		00000000_B
$1FF0_H$	Program address detection register 0	PADR0	R/W	Address match detection function	XXXXXXXX_B
$1FF1_H$	Program address detection register 1	PADR0	R/W		XXXXXXXX_B
$1FF2_H$	Program address detection register 2	PADR0	R/W		XXXXXXXX_B
$1FF3_H$	Program address detection register 3	PADR1	R/W		XXXXXXXX_B
$1FF4_H$	Program address detection register 4	PADR1	R/W		XXXXXXXX_B
$1FF5_H$	ROM correction address 5	PADR1	R/W		XXXXXXXX_B

Table A-2 I/O Map (Sheet 1 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3700_H	General-purpose RAM	-	(R/W)	CAN2	XXXXXXXX_B
3701_H					XXXXXXXX_B
3702_H					XXXXXXXX_B
3703_H					XXXXXXXX_B
3704_H					XXXXXXXX_B
3705_H					XXXXXXXX_B
3706_H					XXXXXXXX_B
3707_H					XXXXXXXX_B
3708_H					XXXXXXXX_B
3709_H					XXXXXXXX_B
$370A_H$					XXXXXXXX_B
$370B_H$					XXXXXXXX_B
$370C_H$					XXXXXXXX_B
$370D_H$					XXXXXXXX_B
$370E_H$					XXXXXXXX_B
$370F_H$					XXXXXXXX_B
3710_H					XXXXXXXX_B
3711_H					XXXXXXXX_B
3712_H					XXXXXXXX_B
3713_H					XXXXXXXX_B
3714_H					XXXXXXXX_B
3715_H					XXXXXXXX_B
3716_H					XXXXXXXX_B
3717_H					XXXXXXXX_B
3718_H					XXXXXXXX_B
3719_H					XXXXXXXX_B
$371A_H$					XXXXXXXX_B
$371B_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 2 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$371C_H$	General-purpose RAM	-	(R/W)	CAN2	XXXXXXXX_B
$371D_H$					XXXXXXXX_B
$371E_H$					XXXXXXXX_B
$371F_H$					XXXXXXXX_B
3720_H	ID register 0	IDR0	(R/W)		XXXXXXXX_B
3721_H					XXXXXXXX_B
3722_H					XXXXXXXX_B
3723_H					XXXXXXXX_B
3724_H	ID register 1	IDR1	(R/W)		XXXXXXXX_B
3725_H					XXXXXXXX_B
3726_H					XXXXXXXX_B
3727_H					XXXXXXXX_B
3728_H	ID register 2	IDR2	(R/W)		XXXXXXXX_B
3729_H					XXXXXXXX_B
$372A_H$					XXXXXXXX_B
$372B_H$					XXXXXXXX_B
$372C_H$	ID register 3	IDR3	(R/W)		XXXXXXXX_B
$372D_H$					XXXXXXXX_B
$372E_H$					XXXXXXXX_B
$372F_H$					XXXXXXXX_B
3730_H	ID register 4	IDR4	(R/W)		XXXXXXXX_B
3731_H					XXXXXXXX_B
3732_H					XXXXXXXX_B
3733_H					XXXXXXXX_B
3734_H	ID register 5	IDR5	(R/W)		XXXXXXXX_B
3735_H					XXXXXXXX_B
3736_H					XXXXXXXX_B
3737_H					XXXXXXXX_B

Table A-2 I/O Map (Sheet 3 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3738_H	ID register 6	IDR6	(R/W)	CAN2	XXXXXXXX_B
3739_H					XXXXXXXX_B
$373A_H$					XXXXXXXX_B
$373B_H$					XXXXXXXX_B
$373C_H$	ID register 7	IDR7	(R/W)		XXXXXXXX_B
$373D_H$					XXXXXXXX_B
$373E_H$					XXXXXXXX_B
$373F_H$					XXXXXXXX_B
3740_H	ID register 8	IDR8	(R/W)		XXXXXXXX_B
3741_H					XXXXXXXX_B
3742_H					XXXXXXXX_B
3743_H					XXXXXXXX_B
3744_H	ID register 9	IDR9	(R/W)		XXXXXXXX_B
3745_H					XXXXXXXX_B
3746_H					XXXXXXXX_B
3747_H					XXXXXXXX_B
3748_H	ID register 10	IDR10	(R/W)		XXXXXXXX_B
3749_H					XXXXXXXX_B
$374A_H$					XXXXXXXX_B
$374B_H$					XXXXXXXX_B
$374C_H$	ID register 11	IDR11	(R/W)		XXXXXXXX_B
$374D_H$					XXXXXXXX_B
$374E_H$					XXXXXXXX_B
$374F_H$					XXXXXXXX_B
3750_H	ID register 12	IDR12	(R/W)		XXXXXXXX_B
3751_H					XXXXXXXX_B
3752_H					XXXXXXXX_B
3753_H					XXXXXXXX_B

Table A-2 I/O Map (Sheet 4 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3754_H	ID register 13	IDR13	(R/W)	CAN2	XXXXXXXX_B
3755_H					XXXXXXXX_B
3756_H					XXXXXXXX_B
3757_H					XXXXXXXX_B
3758_H	ID register 14	IDR14	(R/W)		XXXXXXXX_B
3759_H					XXXXXXXX_B
$375A_H$					XXXXXXXX_B
$375B_H$					XXXXXXXX_B
$375C_H$	ID register 15	IDR15	(R/W)		XXXXXXXX_B
$375D_H$					XXXXXXXX_B
$375E_H$					XXXXXXXX_B
$375F_H$					XXXXXXXX_B
3760_H	DLC register 0	DLCR0	(R/W)		XXXXXXXX_B
3761_H					XXXXXXXX_B
3762_H	DLC register 1	DLCR1	(R/W)		XXXXXXXX_B
3763_H					XXXXXXXX_B
3764_H	DLC register 2	DLCR2	(R/W)		XXXXXXXX_B
3765_H					XXXXXXXX_B
3766_H	DLC register 3	DLCR3	(R/W)		XXXXXXXXB
3767_H					XXXXXXXX_B
3768_H	DLC register 4	DLCR4	(R/W)		XXXXXXXX_B
3769_H					XXXXXXXX_B
$376A_H$	DLC register 5	DLCR5	(R/W)		XXXXXXXX_B
$376B_H$					XXXXXXXX_B
$376C_H$	DLC register 6	DLCR6	(R/W)		XXXXXXXX_B
$376D_H$					XXXXXXXX_B
$376E_H$	DLC register 7	DLCR7	(R/W)		XXXXXXXX_B
$376F_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 5 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3770_H	DLC register 8	DLCR8	(R/W)	CAN2	XXXXXXXX_B
3771_H					XXXXXXXX_B
3772_H	DLC register 9	DLCR9	(R/W)		XXXXXXXX_B
3773_H					XXXXXXXX_B
3774_H	DLC register 10	DLCR10	(R/W)		XXXXXXXX_B
3775_H					XXXXXXXX_B
3776_H	DLC register 11	DLCR11	(R/W)		XXXXXXXX_B
3777_H					XXXXXXXX_B
3778_H	DLC register 12	DLCR12	(R/W)		XXXXXXXX_B
3779_H					XXXXXXXX_B
$377A_H$	DLC register 13	DLCR13	(R/W)		XXXXXXXX_B
$377B_H$					XXXXXXXX_B
$377C_H$	DLC register 14	DLCR14	(R/W)		XXXXXXXX_B
$377D_H$					XXXXXXXX_B
$377E_H$	DLC register 15	DLCR15	(R/W)		XXXXXXXX_B
$377F_H$					XXXXXXXX_B
3780_H	Data register 0 (8 bytes)	DTR0	(R/W)		XXXXXXXX_B
3781_H					XXXXXXXX_B
3782_H					XXXXXXXX_B
3783_H					XXXXXXXX_B
3784_H					XXXXXXXX_B
3785_H					XXXXXXXX_B
3786_H					XXXXXXXX_B
3787_H					XXXXXXXX_B

Table A-2 I/O Map (Sheet 6 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3788_H	Data register 1 (8 bytes)	DTR1	(R/W)	CAN2	XXXXXXXX_B
3789_H					XXXXXXXX_B
$378A_H$					XXXXXXXX_B
$378B_H$					XXXXXXXX_B
$378C_H$					XXXXXXXX_B
$378D_H$					XXXXXXXX_B
$378E_H$					XXXXXXXX_B
$378F_H$					XXXXXXXX_B
3790_H	Data register 2 (8 bytes)	DTR2	(R/W)		XXXXXXXX_B
3791_H					XXXXXXXX_B
3792_H					XXXXXXXX_B
3793_H					XXXXXXXX_B
3794_H					XXXXXXXX_B
3795_H					XXXXXXXX_B
3796_H					XXXXXXXX_B
3797_H					XXXXXXXX_B
3798_H	Data register 3 (8 bytes)	DTR3	(R/W)		XXXXXXXX_B
3799_H					XXXXXXXX_B
$379A_H$					XXXXXXXX_B
$379B_H$					XXXXXXXX_B
$379C_H$					XXXXXXXX_B
$379D_H$					XXXXXXXX_B
$379E_H$					XXXXXXXX_B
$379F_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 7 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$37A0_H$	Data register 4 (8 bytes)	DTR4	(R/W)	CAN2	XXXXXXXX_B
$37A1_H$					XXXXXXXX_B
$37A2_H$					XXXXXXXX_B
$37A3_H$					XXXXXXXX_B
$37A4_H$					XXXXXXXX_B
$37A5_H$					XXXXXXXX_B
$37A6_H$					XXXXXXXX_B
$37A7_H$					XXXXXXXX_B
$37A8_H$	Data register 5 (8 bytes)	DTR5	(R/W)		XXXXXXXX_B
$37A9_H$					XXXXXXXX_B
$37AA_H$					XXXXXXXX_B
$37AB_H$					XXXXXXXX_B
$37AC_H$					XXXXXXXX_B
$37AD_H$					XXXXXXXX_B
$37AE_H$					XXXXXXXX_B
$37AF_H$					XXXXXXXX_B
$37B0_H$	Data register 6 (8 bytes)	DTR6	(R/W)		XXXXXXXX_B
$37B1_H$					XXXXXXXX_B
$37B2_H$					XXXXXXXX_B
$37B3_H$					XXXXXXXX_B
$37B4_H$					XXXXXXXX_B
$37B5_H$					XXXXXXXX_B
$37B6_H$					XXXXXXXX_B
$37B7_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 8 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$37B8_H$	Data register 7 (8 bytes)	DTR7	(R/W)	CAN2	XXXXXXXX_B
$37B9_H$					XXXXXXXX_B
$37BA_H$					XXXXXXXX_B
$37BB_H$					XXXXXXXX_B
$37BC_H$					XXXXXXXX_B
$37BD_H$					XXXXXXXX_B
$37BE_H$					XXXXXXXX_B
$37BF_H$					XXXXXXXX_B
$37C0_H$	Data register 8 (8 bytes)	DTR8	(R/W)		XXXXXXXX_B
$37C1_H$					XXXXXXXX_B
$37C2_H$					XXXXXXXX_B
$37C3_H$					XXXXXXXX_B
$37C4_H$					XXXXXXXX_B
$37C5_H$					XXXXXXXX_B
$37C6_H$					XXXXXXXX_B
$37C7_H$					XXXXXXXX_B
$37C8_H$	Data register 9 (8 bytes)	DTR9	(R/W)		XXXXXXXX_B
$37C9_H$					XXXXXXXX_B
$37CA_H$					XXXXXXXX_B
$37CB_H$					XXXXXXXX_B
$37CC_H$					XXXXXXXX_B
$37CD_H$					XXXXXXXX_B
$37CE_H$					XXXXXXXX_B
$37CF_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 9 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$37D0_H$	Data register 10 (8 bytes)	DTR10	(R/W)	CAN2	XXXXXXXX_B
$37D1_H$					XXXXXXXX_B
$37D2_H$					XXXXXXXX_B
$37D3_H$					XXXXXXXX_B
$37D4_H$					XXXXXXXX_B
$37D5_H$					XXXXXXXX_B
$37D6_H$					XXXXXXXX_B
$37D7_H$					XXXXXXXX_B
$37D8_H$	Data register 11 (8 bytes)	DTR11	(R/W)		XXXXXXXX_B
$37D9_H$					XXXXXXXX_B
$37DA_H$					XXXXXXXX_B
$37DB_H$					XXXXXXXX_B
$37DC_H$					XXXXXXXX_B
$37DD_H$					XXXXXXXX_B
$37DE_H$					XXXXXXXX_B
$37DF_H$					XXXXXXXX_B
$37E0_H$	Data register 12 (8 bytes)	DTR12	(R/W)		XXXXXXXX_B
$37E1_H$					XXXXXXXX_B
$37E2_H$					XXXXXXXX_B
$37E3_H$					XXXXXXXX_B
$37E4_H$					XXXXXXXX_B
$37E5_H$					XXXXXXXX_B
$37E6_H$					XXXXXXXX_B
$37E7_H$					XXXXXXXX_B

Table A-2 I/O Map (Sheet 10 of 10)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$37E8_H$	Data register 13 (8 bytes)	DTR13	(R/W)	CAN2	XXXXXXXX_B
$37E9_H$					XXXXXXXX_B
$37EA_H$					XXXXXXXX_B
$37EB_H$					XXXXXXXX_B
$37EC_H$					XXXXXXXX_B
$37ED_H$					XXXXXXXX_B
$37EE_H$					XXXXXXXX_B
$37EF_H$					XXXXXXXX_B
$37F0_H$	Data register 14 (8 bytes)	DTR14	(R/W)		XXXXXXXX_B
$37F1_H$					XXXXXXXX_B
$37F2_H$					XXXXXXXX_B
$37F3_H$					XXXXXXXX_B
$37F4_H$					XXXXXXXX_B
$37F5_H$					XXXXXXXX_B
$37F6_H$					XXXXXXXX_B
$37F7_H$					XXXXXXXX_B
$37F8_H$	Data register 15 (8 bytes)	DTR15	(R/W)		XXXXXXXX_B
$37F9_H$					XXXXXXXX_B
$37FA_H$					XXXXXXXX_B
$37FB_H$					XXXXXXXX_B
$37FC_H$					XXXXXXXX_B
$37FD_H$					XXXXXXXX_B
$37FE_H$					XXXXXXXX_B
$37FF_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 1 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3800_H	General-purpose RAM	-	(R/W)	CAN3	XXXXXXXX_B
3801_H					XXXXXXXX_B
3802_H					XXXXXXXX_B
3803_H					XXXXXXXX_B
3804_H					XXXXXXXX_B
3805_H					XXXXXXXX_B
3806_H					XXXXXXXX_B
3807_H					XXXXXXXX_B
3808_H					XXXXXXXX_B
3809_H					XXXXXXXX_B
$380A_H$					XXXXXXXX_B
$380B_H$					XXXXXXXX_B
$380C_H$					XXXXXXXX_B
$380D_H$					XXXXXXXX_B
$380E_H$					XXXXXXXX_B
$380F_H$					XXXXXXXX_B
3810_H					XXXXXXXX_B
3811_H					XXXXXXXX_B
3812_H					XXXXXXXX_B
3813_H					XXXXXXXX_B
3814_H					XXXXXXXX_B
3815_H					XXXXXXXX_B
3816_H					XXXXXXXX_B
3817_H					XXXXXXXX_B
3818_H					XXXXXXXX_B
3819_H					XXXXXXXX_B
$381A_H$					XXXXXXXX_B
$381B_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 2 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$381C_H$	General-purpose RAM	-	(R/W)	CAN3	XXXXXXXX_B
$381D_H$					XXXXXXXX_B
$381E_H$					XXXXXXXX_B
$381F_H$					XXXXXXXX_B
3820_H	ID register 0	IDR0	(R/W)		XXXXXXXX_B
3821_H					XXXXXXXX_B
3822_H					XXXXXXXX_B
3823_H					XXXXXXXX_B
3824_H	ID register 1	IDR1	(R/W)		XXXXXXXX_B
3825_H					XXXXXXXX_B
3826_H					XXXXXXXX_B
3827_H					XXXXXXXX_B
3828_H	ID register 2	IDR2	(R/W)		XXXXXXXX_B
3829_H					XXXXXXXX_B
$382A_H$					XXXXXXXX_B
$382B_H$					XXXXXXXX_B
$382C_H$	ID register 3	IDR3	(R/W)		XXXXXXXX_B
$382D_H$					XXXXXXXX_B
$382E_H$					XXXXXXXX_B
$382F_H$					XXXXXXXX_B
3830_H	ID register 4	IDR4	(R/W)		XXXXXXXX_B
3831_H					XXXXXXXX_B
3832_H					XXXXXXXX_B
3833_H					XXXXXXXX_B
3834_H	ID register 5	IDR5	(R/W)		XXXXXXXX_B
3835_H					XXXXXXXX_B
3836_H					XXXXXXXX_B
3837_H					XXXXXXXX_B

Table A-3 I/O Map (Sheet 3 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3838_H	ID register 6	IDR6	(R/W)	CAN3	XXXXXXXX_B
3839_H					XXXXXXXX_B
$383A_H$					XXXXXXXX_B
$383B_H$					XXXXXXXX_B
$383C_H$	ID register 7	IDR7	(R/W)		XXXXXXXX_B
$383D_H$					XXXXXXXX_B
$383E_H$					XXXXXXXX_B
$383F_H$					XXXXXXXX_B
3840_H	ID register 8	IDR8	(R/W)		XXXXXXXX_B
3841_H					XXXXXXXX_B
3842_H					XXXXXXXX_B
3843_H					XXXXXXXX_B
3844_H	ID register 9	IDR9	(R/W)		XXXXXXXX_B
3845_H					XXXXXXXX_B
3846_H					XXXXXXXX_B
3847_H					XXXXXXXX_B
3848_H	ID register 10	IDR10	(R/W)		XXXXXXXX_B
3849_H					XXXXXXXX_B
$384A_H$					XXXXXXXX_B
$384B_H$					XXXXXXXX_B
$384C_H$	ID register 11	IDR11	(R/W)		XXXXXXXX_B
$384D_H$					XXXXXXXX_B
$384E_H$					XXXXXXXX_B
$384F_H$					XXXXXXXX_B
3850_H	ID register 12	IDR12	(R/W)		XXXXXXXX_B
3851_H					XXXXXXXX_B
3852_H					XXXXXXXX_B
3853_H					XXXXXXXX_B

Table A-3 I/O Map (Sheet 4 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3854_H	ID register 13	IDR13	(R/W)	CAN3	XXXXXXXX_B
3855_H					XXXXXXXX_B
3856_H					XXXXXXXX_B
3857_H					XXXXXXXX_B
3858_H	ID register 14	IDR14	(R/W)		XXXXXXXX_B
3859_H					XXXXXXXX_B
$385A_H$					XXXXXXXX_B
$385B_H$					XXXXXXXX_B
$385C_H$	ID register 15	IDR15	(R/W)		XXXXXXXX_B
$385D_H$					XXXXXXXX_B
$385E_H$					XXXXXXXX_B
$385F_H$					XXXXXXXX_B
3860_H	DLC register 0	DLCR0	(R/W)		XXXXXXXX_B
3861_H					XXXXXXXX_B
3862_H	DLC register 1	DLCR1	(R/W)		XXXXXXXX_B
3863_H					XXXXXXXX_B
3864_H	DLC register 2	DLCR2	(R/W)		XXXXXXXX_B
3865_H					XXXXXXXX_B
3866_H	DLC register 3	DLCR3	(R/W)		XXXXXXXX_B
3867_H					XXXXXXXX_B
3868_H	DLC register 4	DLCR4	(R/W)		XXXXXXXX_B
3869_H					XXXXXXXX_B
$386A_H$	DLC register 5	DLCR5	(R/W)		XXXXXXXX_B
$386B_H$					XXXXXXXX_B
$386C_H$	DLC register 6	DLCR6	(R/W)		XXXXXXXX_B
$386D_H$					XXXXXXXX_B
$386E_H$	DLC register 7	DLCR7	(R/W)		XXXXXXXX_B
$386F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 5 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3870_H	DLC register 8	DLCR8	(R/W)	CAN3	XXXXXXXX_B
3871_H					XXXXXXXX_B
3872_H	DLC register 9	DLCR9	(R/W)		XXXXXXXX_B
3873_H					XXXXXXXX_B
3874_H	DLC register 10	DLCR10	(R/W)		XXXXXXXX_B
3875_H					XXXXXXXX_B
3876_H	DLC register 11	DLCR11	(R/W)		XXXXXXXX_B
3877_H					XXXXXXXX_B
3878_H	DLC register 12	DLCR12	(R/W)		XXXXXXXX_B
3879_H					XXXXXXXX_B
$387A_H$	DLC register 13	DLCR13	(R/W)		XXXXXXXX_B
$387B_H$					XXXXXXXX_B
$387C_H$	DLC register 14	DLCR14	(R/W)		XXXXXXXX_B
$387D_H$					XXXXXXXX_B
$387E_H$	DLC register 15	DLCR15	(R/W)		XXXXXXXX_B
$387F_H$					XXXXXXXX_B
3880_H	Data register 0 (8 bytes)	DTR0	(R/W)		XXXXXXXX_B
3881_H					XXXXXXXX_B
3882_H					XXXXXXXX_B
3883_H					XXXXXXXX_B
3884_H					XXXXXXXX_B
3885_H					XXXXXXXX_B
3886_H					XXXXXXXX_B
3887_H					XXXXXXXX_B

Table A-3 I/O Map (Sheet 6 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3888_H	Data register 1 (8 bytes)	DTR1	(R/W)	CAN3	XXXXXXXX_B
3889_H					XXXXXXXX_B
$388A_H$					XXXXXXXX_B
$388B_H$					XXXXXXXX_B
$388C_H$					XXXXXXXX_B
$388D_H$					XXXXXXXX_B
$388E_H$					XXXXXXXX_B
$388F_H$					XXXXXXXX_B
3890_H	Data register 2 (8 bytes)	DTR2	(R/W)		XXXXXXXX_B
3891_H					XXXXXXXX_B
3892_H					XXXXXXXX_B
3893_H					XXXXXXXX_B
3894_H					XXXXXXXX_B
3895_H					XXXXXXXX_B
3896_H					XXXXXXXX_B
3897_H					XXXXXXXX_B
3898_H	Data register 3 (8 bytes)	DTR3	(R/W)		XXXXXXXX_B
3899_H					XXXXXXXX_B
$389A_H$					XXXXXXXX_B
$389B_H$					XXXXXXXX_B
$389C_H$					XXXXXXXX_B
$389D_H$					XXXXXXXX_B
$389E_H$					XXXXXXXX_B
$389F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 7 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$38A0_H$	Data register 4 (8 bytes)	DTR4	(R/W)	CAN3	XXXXXXXX_B
$38A1_H$					XXXXXXXX_B
$38A2_H$					XXXXXXXX_B
$38A3_H$					XXXXXXXX_B
$38A4_H$					XXXXXXXX_B
$38A5_H$					XXXXXXXX_B
$38A6_H$					XXXXXXXX_B
$38A7_H$					XXXXXXXX_B
$38A8_H$	Data register 5 (8 bytes)	DTR5	(R/W)		XXXXXXXX_B
$38A9_H$					XXXXXXXX_B
$38AA_H$					XXXXXXXX_B
$38AB_H$					XXXXXXXX_B
$38AC_H$					XXXXXXXX_B
$38AD_H$					XXXXXXXX_B
$38AE_H$					XXXXXXXX_B
$38AF_H$					XXXXXXXX_B
$38B0_H$	Data register 6 (8 bytes)	DTR6	(R/W)		XXXXXXXX_B
$38B1_H$					XXXXXXXX_B
$38B2_H$					XXXXXXXX_B
$38B3_H$					XXXXXXXX_B
$38B4_H$					XXXXXXXX_B
$38B5_H$					XXXXXXXX_B
$38B6_H$					XXXXXXXX_B
$38B7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 8 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$38B8_H$	Data register 7 (8 bytes)	DTR7	(R/W)	CAN3	XXXXXXXX_B
$38B9_H$					XXXXXXXX_B
$38BA_H$					XXXXXXXX_B
$38BB_H$					XXXXXXXX_B
$38BC_H$					XXXXXXXX_B
$38BD_H$					XXXXXXXX_B
$38BE_H$					XXXXXXXX_B
$38BF_H$					XXXXXXXX_B
$38C0_H$	Data register 8 (8 bytes)	DTR8	(R/W)		XXXXXXXX_B
$38C1_H$					XXXXXXXX_B
$38C2_H$					XXXXXXXX_B
$38C3_H$					XXXXXXXX_B
$38C4_H$					XXXXXXXX_B
$38C5_H$					XXXXXXXX_B
$38C6_H$					XXXXXXXX_B
$38C7_H$					XXXXXXXX_B
$38C8_H$	Data register 9 (8 bytes)	DTR9	(R/W)		XXXXXXXX_B
$38C9_H$					XXXXXXXX_B
$38CA_H$					XXXXXXXX_B
$38CB_H$					XXXXXXXX_B
$38CC_H$					XXXXXXXX_B
$38CD_H$					XXXXXXXX_B
$38CE_H$					XXXXXXXX_B
$38CF_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 9 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$38D0_H$	Data register 10 (8 bytes)	DTR10	(R/W)	CAN3	XXXXXXXX_B
$38D1_H$					XXXXXXXX_B
$38D2_H$					XXXXXXXX_B
$38D3_H$					XXXXXXXX_B
$38D4_H$					XXXXXXXX_B
$38D5_H$					XXXXXXXX_B
$38D6_H$					XXXXXXXX_B
$38D7_H$					XXXXXXXX_B
$38D8_H$	Data register 11 (8 bytes)	DTR11	(R/W)		XXXXXXXX_B
$38D9_H$					XXXXXXXX_B
$38DA_H$					XXXXXXXX_B
$38DB_H$					XXXXXXXX_B
$38DC_H$					XXXXXXXX_B
$38DD_H$					XXXXXXXX_B
$38DE_H$					XXXXXXXX_B
$38DF_H$					XXXXXXXX_B
$38E0_H$	Data register 12 (8 bytes)	DTR12	(R/W)		XXXXXXXX_B
$38E1_H$					XXXXXXXX_B
$38E2_H$					XXXXXXXX_B
$38E3_H$					XXXXXXXX_B
$38E4_H$					XXXXXXXX_B
$38E5_H$					XXXXXXXX_B
$38E6_H$					XXXXXXXX_B
$38E7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 10 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$38E8_H$	Data register 13 (8 bytes)	DTR13	(R/W)	CAN3	XXXXXXXX_B
$38E9_H$					XXXXXXXX_B
$38EA_H$					XXXXXXXX_B
$38EB_H$					XXXXXXXX_B
$38EC_H$					XXXXXXXX_B
$38ED_H$					XXXXXXXX_B
$38EE_H$					XXXXXXXX_B
$38EF_H$					XXXXXXXX_B
$38F0_H$	Data register 14 (8 bytes)	DTR14	(R/W)		XXXXXXXX_B
$38F1_H$					XXXXXXXX_B
$38F2_H$					XXXXXXXX_B
$38F3_H$					XXXXXXXX_B
$38F4_H$					XXXXXXXX_B
$38F5_H$					XXXXXXXX_B
$38F6_H$					XXXXXXXX_B
$38F7_H$					XXXXXXXX_B
$38F8_H$	Data register 15 (8 bytes)	DTR15	(R/W)		XXXXXXXX_B
$38F9_H$					XXXXXXXX_B
$38FA_H$					XXXXXXXX_B
$38FB_H$					XXXXXXXX_B
$38FC_H$					XXXXXXXX_B
$38FD_H$					XXXXXXXX_B
$38FE_H$					XXXXXXXX_B
$38FF_H$					XXXXXXXX_B
3900_H to $391F_H$	(Not available)				
3920_H	PPG0 down counter register	PDCR0	R	16bit PPG0	11111111_B
3921_H					11111111_B
3922_H	PPG0 cycle setting register	PCSR0	W		11111111_B
3923_H					11111111_B
3924_H	PPG0 duty setting register	PDUT0	W		00000000_B
3925_H					00000000_B
3926_H	PPG0 output frequency-divide setting register	PPGDIV0	R/W, R		11111100_B
3927_H	(Not available)				

Table A-3 I/O Map (Sheet 11 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3928_H	PPG1 down counter register	PDCR1	R	16bit PPG1	11111111_B
3929_H					11111111_B
$392A_H$	PPG1 cycle setting register	PCSR1	W		11111111_B
$392B_H$					11111111_B
$392C_H$	PPG1 duty setting register	PDUT1	W		00000000_B
$392D_H$					00000000_B
$392E_H$	PPG1 output frequency-divide setting register	PPGDIV1	R/W, R		11111100_B
$392F_H$	(Not available)				
3930_H	PPG2 down counter register	PDCR2	R	16bit PPG2	11111111_B
3931_H					11111111_B
3932_H	PPG2 cycle setting register	PCSR2	W		11111111_B
3933_H					11111111_B
3934_H	PPG2 duty setting register	PDUT2	W		00000000_B
3935_H					00000000_B
3936_H	PPG2 output frequency-divide setting register	PPGDIV2	R/W, R		11111100_B
3937_H to $393F_H$	(Not available)				
3940_H	Input capture register 4	IPCP4	R	Input capture 4/5	XXXXXXXX_B
3941_H					XXXXXXXX_B
3942_H	Input capture register 5	IPCP5	R		XXXXXXXX_B
3943_H					XXXXXXXX_B
3944_H	Input capture register 6	IPCP6	R	Input capture 6/7	XXXXXXXX_B
3945_H					XXXXXXXX_B
3946_H	Input capture register 7	IPCP7	R	Input capture 6/7	XXXXXXXX_B
3947_H					XXXXXXXX_B
3948_H to $394F_H$	(Not available)				
3950_H	Timer register 2/Reload register 2	TMR2/ TMRLR2	R/W	16bit reload timer 2	XXXXXXXX_B
3951_H					XXXXXXXX_B
3952_H	Timer register 3/Reload register 3	TMR3/ TMRLR3	R/W	16bit reload timer 3	XXXXXXXX_B
3953_H					XXXXXXXX_B
3954_H to 3957_H	(Not available)				

Table A-3 I/O Map (Sheet 12 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3958_H	Sub second data register	WTBR	R/W	Real time watch timer	XXXXXXXX_B
3959_H					XXXXXXXX_B
$395A_H$					XXXXXXXX_B
$395B_H$	Second data register	WTSR	R/W		$XX000000_B$
$395C_H$	Minute data register	WTMR	R/W		$XX000000_B$
$395D_H$	Hour data register	WTHR	R/W		$XXX00000_B$
$395E_H$	Day data register	WTDR	R/W		$00X00001_B$
$395F_H$	(Not available)				
3960_H	LCD display RAM	VRAM	R/W	LCD controller/ driver	XXXXXXXX_B
3961_H					XXXXXXXX_B
3962_H					XXXXXXXX_B
3963_H					XXXXXXXX_B
3964_H					XXXXXXXX_B
3965_H	LCD display RAM	VRAM	R/W	LCD controller/ driver	XXXXXXXX_B
3966_H					XXXXXXXX_B
3967_H					XXXXXXXX_B
3968_H					XXXXXXXX_B
3969_H					XXXXXXXX_B
$396A_H$					XXXXXXXX_B
$396B_H$					XXXXXXXX_B
$396C_H$					XXXXXXXX_B
$396D_H$					XXXXXXXX_B
$396E_H$					XXXXXXXX_B
$396F_H$					XXXXXXXX_B
3970_H to 3973_H	(Not available)				
3974_H	Frequency data register 1	SGFR1	R/W	Sound generator 1	XXXXXXXX_B
3975_H	Amplitude data register 1	SGAR1	R/W		00000000_B
3976_H	Decrement grade register 1	SGDR1	R/W		XXXXXXXX_B
3977_H	Tone count register 1	SGTR1	R/W		XXXXXXXX_B
3978_H to $397F_H$	(Not available)				

Table A-3 I/O Map (Sheet 13 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
3980_H	PWM1 compare register 0	PWC10	R/W	Stepping motor controller 0	XXXXXXXX_B
3981_H					XXXXXXXX_B
3982_H	PMW2 compare register 0	PWC20	R/W		XXXXXXXX_B
3983_H					XXXXXXXX_B
3984_H	PWM1 select register 0	PWS10	R/W		$X0000000_B$
3985_H	PWM2 select register 0	PWS20	R/W		$X0000000_B$
3986_H to 3987_H	(Not available)				
3988_H	PWM1 compare register 1	PWC11	R/W	Stepping motor controller 1	XXXXXXXX_B
3989_H					XXXXXXXX_B
$398A_H$	PMW2 compare register 1	PWC21	R/W		XXXXXXXX_B
$398B_H$					XXXXXXXX_B
$398C_H$	PWM1 select register 1	PWS11	R/W		00000000_B
$398D_H$	PWM2 select register 1	PWS21	R/W		$X0000000_B$
$398E_H$ to $398F_H$	(Not available)				
3990_H	PWM1 compare register 2	PWC12	R/W	Stepping motor controller 2	XXXXXXXX_B
3991_H					XXXXXXXX_B
3992_H	PMW2 compare register 2	PWC22	R/W		XXXXXXXX_B
3993_H					XXXXXXXX_B
3994_H	PWM1 select register 2	PWS12	R/W		00000000_B
3995_H	PWM2 select register 2	PWS22	R/W		$X0000000_B$
3996_H to 3997_H	(Not available)				
3998_H	PWM1 compare register 3	PWC13	R/W	Stepping motor controller 3	XXXXXXXX_B
3999_H					XXXXXXXX_B
$399A_H$	PMW2 compare register 3	PWC23	R/W		XXXXXXXX_B
$399B_H$					XXXXXXXX_B
$399C_H$	PWM1 select register 3	PWS13	R/W		$XX000000_B$
$399D_H$	PWM2 select register 3	PWS23	R/W		$X0000000_B$
$399E_H$ to $39A5_H$	(Not available)				
$39A6_H$	Flash writing control register 0	FWR0	R/W	Flash Memory I/F	00000000_B
$39A7_H$	Flash writing control register 1	FWR1			00000000_B

Table A-3 I/O Map (Sheet 14 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$39A8_H$ to $39B8_H$	(Not available)				
$39C0_H$	Message buffer valid area	BVALR	(R/W)	CAN2	00000000_B
$39C1_H$					00000000_B
$39C2_H$	Transmission request register	TREQR	(R/W)		00000000_B
$39C3_H$					00000000_B
$39C4_H$	Transmission cancel register	TCANR	(W)	CAN2	00000000_B
$39C5_H$					00000000_B
$39C6_H$	Transmission completed register	TCR	(R/W)		00000000_B
$39C7_H$					00000000_B
$39C8_H$	Receiving completed register	RCR	(R/W)		00000000_B
$39C9_H$					00000000_B
$39CA_H$	Remote request receiving register	RRTRR	(R/W)		00000000_B
$39CB_H$					00000000_B
$39CC_H$	Receiving overrun register	ROVRR	(R/W)		00000000_B
$39CD_H$					00000000_B
$39CE_H$	Receiving interrupt enable register	RIER	(R/W)		00000000_B
$39CF_H$					00000000_B
$39D0_H$	Message buffer valid area	BVALR	(R/W)	CAN3	00000000_B
$39D1_H$					00000000_B
$39D2_H$	Transmission request register	TREQR	(R/W)		00000000_B
$39D3_H$					00000000_B
$39D4_H$	Transmission cancel register	TCANR	(W)		00000000_B
$39D5_H$					00000000_B
$39D6_H$	Transmission completed register	TCR	(R/W)		00000000_B
$39D7_H$					00000000_B
$39D8_H$	Receiving completed register	RCR	(R/W)		00000000_B
$39D9_H$					00000000_B
$39DA_H$	Remote request receiving register	RRTRR	(R/W)		00000000_B
$39DB_H$					00000000_B
$39DC_H$	Receiving overrun register	ROVRR	(R/W)		00000000_B
$39DD_H$					00000000_B
$39DE_H$	Receiving interrupt enable register	RIER	(R/W)		00000000_B
$39DF_H$					00000000_B

Table A-3 I/O Map (Sheet 15 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$39E0_H$	PPG3 down counter register	PDCR3	R	16bit PPG3	11111111_B
$39E1_H$					11111111_B
$39E2_H$	PPG3 cycle setting register	PCSR3	W		11111111_B
$39E3_H$					11111111_B
$39E4_H$	PPG3 duty setting register	PDUT3	W		00000000_B
$39E5_H$					00000000_B
$39E6_H$	PPG3 output frequency-divide setting register	PPGDIV3	R/W, R		11111100_B
$39E7_H$	(Not available)				
$39E8_H$	PPG4 down counter register	PDCR4	R	16bit PPG4	11111111_B
$39E9_H$					11111111_B
$39EA_H$	PPG4 cycle setting register	PCSR4	W		11111111_B
$39EB_H$					11111111_B
$39EC_H$	PPG4 duty setting register	PDUT4	W		00000000_B
$39ED_H$					00000000_B
$39EE_H$	PPG4 output frequency-divide setting register	PPGDIV4	R/W, R		11111100_B
$39EF_H$	(Not available)				
$39F0_H$	PPG5 down counter register	PDCR5	R	16bit PPG5	11111111_B
$39F1_H$					11111111_B
$39F2_H$	PPG5 cycle setting register	PCSR5	W		11111111_B
$39F3_H$					11111111_B
$39F4_H$	PPG5 duty setting register	PDUT5	W		00000000_B
$39F5_H$					00000000_B
$39E6_H$	PPG5 output frequency-divide setting register	PPGDIV5	R/W, R		11111100_B
$39F7_H$ to $39FF_H$	(Not available)				
$3A00_H$	General-purpose RAM	-	(R/W)	CAN0	XXXXXXXX_B
$3A01_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 16 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A02_H$	General-purpose RAM	-	(R/W)	CAN0	XXXXXXXX_B
$3A03_H$					XXXXXXXX_B
$3A04_H$					XXXXXXXX_B
$3A05_H$					XXXXXXXX_B
$3A06_H$					XXXXXXXX_B
$3A07_H$					XXXXXXXX_B
$3A08_H$					XXXXXXXX_B
$3A09_H$					XXXXXXXX_B
$3A0A_H$					XXXXXXXX_B
$3A0B_H$					XXXXXXXX_B
$3A0C_H$					XXXXXXXX_B
$3A0D_H$					XXXXXXXX_B
$3A0E_H$					XXXXXXXX_B
$3A0F_H$					XXXXXXXX_B
$3A10_H$					XXXXXXXX_B
$3A11_H$					XXXXXXXX_B
$3A12_H$					XXXXXXXX_B
$3A13_H$					XXXXXXXX_B
$3A14_H$					XXXXXXXX_B
$3A15_H$					XXXXXXXX_B
$3A16_H$					XXXXXXXX_B
$3A17_H$					XXXXXXXX_B
$3A18_H$					XXXXXXXX_B
$3A19_H$					XXXXXXXX_B
$3A1A_H$					XXXXXXXX_B
$3A1B_H$					XXXXXXXX_B
$3A1C_H$					XXXXXXXX_B
$3A1D_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 17 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A1E_H$	General-purpose RAM	-	(R/W)	CAN0	XXXXXXXX_B
$3A1F_H$					XXXXXXXX_B
$3A20_H$	ID register 0	IDR0	(R/W)		XXXXXXXX_B
$3A21_H$					XXXXXXXX_B
$3A22_H$					XXXXXXXX_B
$3A23_H$					XXXXXXXX_B
$3A24_H$	ID register 1	IDR1	(R/W)		XXXXXXXX_B
$3A25_H$					XXXXXXXX_B
$3A26_H$					XXXXXXXX_B
$3A27_H$					XXXXXXXX_B
$3A28_H$	ID register 2	IDR2	(R/W)		XXXXXXXX_B
$3A29_H$					XXXXXXXX_B
$3A2A_H$					XXXXXXXX_B
$3A2B_H$					XXXXXXXX_B
$3A2C_H$	ID register 3	IDR3	(R/W)		XXXXXXXX_B
$3A2D_H$					XXXXXXXX_B
$3A2E_H$					XXXXXXXX_B
$3A2F_H$					XXXXXXXX_B
$3A30_H$	ID register 4	IDR4	(R/W)		XXXXXXXX_B
$3A31_H$					XXXXXXXX_B
$3A32_H$					XXXXXXXX_B
$3A33_H$					XXXXXXXX_B
$3A34_H$	ID register 5	IDR5	(R/W)		XXXXXXXX_B
$3A35_H$					XXXXXXXX_B
$3A36_H$					XXXXXXXX_B
$3A37_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 18 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A38_H$	ID register 6	IDR6	(R/W)	CAN0	XXXXXXXX_B
$3A39_H$					XXXXXXXX_B
$3A3A_H$					XXXXXXXX_B
$3A3B_H$					XXXXXXXX_B
$3A3C_H$	ID register 7	IDR7	(R/W)		XXXXXXXX_B
$3A3D_H$					XXXXXXXX_B
$3A3E_H$					XXXXXXXX_B
$3A3F_H$					XXXXXXXX_B
$3A40_H$	ID register 8	IDR8	(R/W)		XXXXXXXX_B
$3A41_H$					XXXXXXXX_B
$3A42_H$					XXXXXXXX_B
$3A43_H$					XXXXXXXX_B
$3A44_H$	ID register 9	IDR9	(R/W)		XXXXXXXX_B
$3A45_H$					XXXXXXXX_B
$3A46_H$					XXXXXXXX_B
$3A47_H$					XXXXXXXX_B
$3A48_H$	ID register 10	IDR10	(R/W)		XXXXXXXX_B
$3A49_H$					XXXXXXXX_B
$3A4A_H$					XXXXXXXX_B
$3A4B_H$					XXXXXXXX_B
$3A4C_H$	ID register 11	IDR11	(R/W)		XXXXXXXX_B
$3A4D_H$					XXXXXXXX_B
$3A4E_H$					XXXXXXXX_B
$3A4F_H$					XXXXXXXX_B
$3A50_H$	ID register 12	IDR12	(R/W)		XXXXXXXX_B
$3A51_H$					XXXXXXXX_B
$3A52_H$					XXXXXXXX_B
$3A53_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 19 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A54_H$	ID register 13	IDR13	(R/W)	CAN0	XXXXXXXX_B
$3A55_H$					XXXXXXXX_B
$3A56_H$					XXXXXXXX_B
$3A57_H$					XXXXXXXX_B
$3A58_H$	ID register 14	IDR14	(R/W)		XXXXXXXX_B
$3A59_H$					XXXXXXXX_B
$3A5A_H$					XXXXXXXX_B
$3A5B_H$					XXXXXXXX_B
$3A5C_H$	ID register 15	IDR15	(R/W)		XXXXXXXX_B
$3A5D_H$					XXXXXXXX_B
$3A5E_H$					XXXXXXXX_B
$3A5F_H$					XXXXXXXX_B
$3A60_H$	DLC register 0	DLCR0	(R/W)		XXXXXXXX_B
$3A61_H$					XXXXXXXX_B
$3A62_H$	DLC register 1	DLCR1	(R/W)		XXXXXXXX_B
$3A63_H$					XXXXXXXX_B
$3A64_H$	DLC register 2	DLCR2	(R/W)		XXXXXXXX_B
$3A65_H$					XXXXXXXX_B
$3A66_H$	DLC register 3	DLCR3	(R/W)		XXXXXXXX_B
$3A67_H$					XXXXXXXX_B
$3A68_H$	DLC register 4	DLCR4	(R/W)		XXXXXXXX_B
$3A69_H$					XXXXXXXX_B
$3A6A_H$	DLC register 5	DLCR5	(R/W)		XXXXXXXX_B
$3A6B_H$					XXXXXXXX_B
$3A6C_H$	DLC register 6	DLCR6	(R/W)		XXXXXXXX_B
$3A6D_H$					XXXXXXXX_B
$3A6E_H$	DLC register 7	DLCR7	(R/W)		XXXXXXXX_B
$3A6F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 20 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A70_H$	DLC register 8	DLCR8	(R/W)	CAN0	XXXXXXXX_B
$3A71_H$					XXXXXXXX_B
$3A72_H$	DLC register 9	DLCR9	(R/W)		XXXXXXXX_B
$3A73_H$					XXXXXXXX_B
$3A74_H$	DLC register 10	DLCR10	(R/W)		XXXXXXXX_B
$3A75_H$					XXXXXXXX_B
$3A76_H$	DLC register 11	DLCR11	(R/W)		XXXXXXXX_B
$3A77_H$					XXXXXXXX_B
$3A78_H$	DLC register 12	DLCR12	(R/W)		XXXXXXXX_B
$3A79_H$					XXXXXXXX_B
$3A7A_H$	DLC register 13	DLCR13	(R/W)		XXXXXXXX_B
$3A7B_H$					XXXXXXXX_B
$3A7C_H$	DLC register 14	DLCR14	(R/W)		XXXXXXXX_B
$3A7D_H$					XXXXXXXX_B
$3A7E_H$	DLC register 15	DLCR15	(R/W)		XXXXXXXX_B
$3A7F_H$					XXXXXXXX_B
$3A80_H$	Data register 0 (8 bytes)	DTR0	(R/W)		XXXXXXXX_B
$3A81_H$					XXXXXXXX_B
$3A82_H$					XXXXXXXX_B
$3A83_H$					XXXXXXXX_B
$3A84_H$					XXXXXXXX_B
$3A85_H$					XXXXXXXX_B
$3A86_H$					XXXXXXXX_B
$3A87_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 21 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3A88_H$	Data register 1 (8 bytes)	DTR1	(R/W)	CAN0	XXXXXXXX_B
$3A89_H$					XXXXXXXX_B
$3A8A_H$					XXXXXXXX_B
$3A8B_H$					XXXXXXXX_B
$3A8C_H$					XXXXXXXX_B
$3A8D_H$					XXXXXXXX_B
$3A8E_H$					XXXXXXXX_B
$3A8F_H$					XXXXXXXX_B
$3A90_H$	Data register 2 (8 bytes)	DTR2	(R/W)		XXXXXXXX_B
$3A91_H$					XXXXXXXX_B
$3A92_H$					XXXXXXXX_B
$3A93_H$					XXXXXXXX_B
$3A94_H$					XXXXXXXX_B
$3A95_H$					XXXXXXXX_B
$3A96_H$					XXXXXXXX_B
$3A97_H$					XXXXXXXX_B
$3A98_H$	Data register 3 (8 bytes)	DTR3	(R/W)		XXXXXXXX_B
$3A99_H$					XXXXXXXX_B
$3A9A_H$					XXXXXXXX_B
$3A9B_H$					XXXXXXXX_B
$3A9C_H$					XXXXXXXX_B
$3A9D_H$					XXXXXXXX_B
$3A9E_H$					XXXXXXXX_B
$3A9F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 22 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3AA0_H$	Data register 4 (8 bytes)	DTR4	(R/W)	CAN0	XXXXXXXX_B
$3AA1_H$					XXXXXXXX_B
$3AA2_H$					XXXXXXXX_B
$3AA3_H$					XXXXXXXX_B
$3AA4_H$					XXXXXXXX_B
$3AA5_H$					XXXXXXXX_B
$3AA6_H$					XXXXXXXX_B
$3AA7_H$					XXXXXXXX_B
$3AA8_H$	Data register 5 (8 bytes)	DTR5	(R/W)		XXXXXXXX_B
$3AA9_H$					XXXXXXXX_B
$3AAA_H$					XXXXXXXX_B
$3AAB_H$					XXXXXXXX_B
$3AAC_H$					XXXXXXXX_B
$3AAD_H$					XXXXXXXX_B
$3AAE_H$					XXXXXXXX_B
$3AAF_H$					XXXXXXXX_B
$3AB0_H$	Data register 6 (8 bytes)	DTR6	(R/W)		XXXXXXXX_B
$3AB1_H$					XXXXXXXX_B
$3AB2_H$					XXXXXXXX_B
$3AB3_H$					XXXXXXXX_B
$3AB4_H$					XXXXXXXX_B
$3AB5_H$					XXXXXXXX_B
$3AB6_H$					XXXXXXXX_B
$3AB7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 23 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3AB8_H$	Data register 7 (8 bytes)	DTR7	(R/W)	CAN0	XXXXXXXX_B
$3AB9_H$					XXXXXXXX_B
$3ABA_H$					XXXXXXXX_B
$3ABB_H$					XXXXXXXX_B
$3ABC_H$					XXXXXXXX_B
$3ABD_H$					XXXXXXXX_B
$3ABE_H$					XXXXXXXX_B
$3ABF_H$					XXXXXXXX_B
$3AC0_H$	Data register 8 (8 bytes)	DTR8	(R/W)		XXXXXXXX_B
$3AC1_H$					XXXXXXXX_B
$3AC2_H$					XXXXXXXX_B
$3AC3_H$					XXXXXXXX_B
$3AC4_H$					XXXXXXXX_B
$3AC5_H$					XXXXXXXX_B
$3AC6_H$					XXXXXXXX_B
$3AC7_H$					XXXXXXXX_B
$3AC8_H$	Data register 9 (8 bytes)	DTR9	(R/W)		XXXXXXXX_B
$3AC9_H$					XXXXXXXX_B
$3ACA_H$					XXXXXXXX_B
$3ACB_H$					XXXXXXXX_B
$3ACC_H$					XXXXXXXX_B
$3ACD_H$					XXXXXXXX_B
$3ACE_H$					XXXXXXXX_B
$3ACF_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 24 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3AD0_H$	Data register 10 (8 bytes)	DTR10	(R/W)	CAN0	XXXXXXXX_B
$3AD1_H$					XXXXXXXX_B
$3AD2_H$					XXXXXXXX_B
$3AD3_H$					XXXXXXXX_B
$3AD4_H$					XXXXXXXX_B
$3AD5_H$					XXXXXXXX_B
$3AD6_H$					XXXXXXXX_B
$3AD7_H$					XXXXXXXX_B
$3AD8_H$	Data register 11 (8 bytes)	DTR11	(R/W)		XXXXXXXX_B
$3AD9_H$					XXXXXXXX_B
$3ADA_H$					XXXXXXXX_B
$3ADB_H$					XXXXXXXX_B
$3ADC_H$					XXXXXXXX_B
$3ADD_H$					XXXXXXXX_B
$3ADE_H$					XXXXXXXX_B
$3ADF_H$					XXXXXXXX_B
$3AE0_H$	Data register 12 (8 bytes)	DTR12	(R/W)		XXXXXXXX_B
$3AE1_H$					XXXXXXXX_B
$3AE2_H$					XXXXXXXX_B
$3AE3_H$					XXXXXXXX_B
$3AE4_H$					XXXXXXXX_B
$3AE5_H$					XXXXXXXX_B
$3AE6_H$					XXXXXXXX_B
$3AE7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 25 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3AE8_H$	Data register 13 (8 bytes)	DTR13	(R/W)	CAN0	XXXXXXXX_B
$3AE9_H$					XXXXXXXX_B
$3AEA_H$					XXXXXXXX_B
$3AEB_H$					XXXXXXXX_B
$3AEC_H$					XXXXXXXX_B
$3AED_H$					XXXXXXXX_B
$3AEE_H$					XXXXXXXX_B
$3AEF_H$					XXXXXXXX_B
$3AF0_H$	Data register 14 (8 bytes)	DTR14	(R/W)		XXXXXXXX_B
$3AF1_H$					XXXXXXXX_B
$3AF2_H$					XXXXXXXX_B
$3AF3_H$					XXXXXXXX_B
$3AF4_H$					XXXXXXXX_B
$3AF5_H$					XXXXXXXX_B
$3AF6_H$					XXXXXXXX_B
$3AF7_H$					XXXXXXXX_B
$3AF8_H$	Data register 15 (8 bytes)	DTR15	(R/W)		XXXXXXXX_B
$3AF9_H$					XXXXXXXX_B
$3AFA_H$					XXXXXXXX_B
$3AFB_H$					XXXXXXXX_B
$3AFC_H$					XXXXXXXX_B
$3AFD_H$					XXXXXXXX_B
$3AFE_H$					XXXXXXXX_B
$3AFF_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 26 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B00_H$	General-purpose RAM	-	(R/W)	CAN1	XXXXXXXX_B
$3B01_H$					XXXXXXXX_B
$3B02_H$					XXXXXXXX_B
$3B03_H$					XXXXXXXX_B
$3B04_H$					XXXXXXXX_B
$3B05_H$					XXXXXXXX_B
$3B06_H$					XXXXXXXX_B
$3B07_H$					XXXXXXXX_B
$3B08_H$					XXXXXXXX_B
$3B09_H$					XXXXXXXX_B
$3B0A_H$					XXXXXXXX_B
$3B0B_H$					XXXXXXXX_B
$3B0C_H$					XXXXXXXX_B
$3B0D_H$					XXXXXXXX_B
$3B0E_H$					XXXXXXXX_B
$3B0F_H$					XXXXXXXX_B
$3B10_H$					XXXXXXXX_B
$3B11_H$					XXXXXXXX_B
$3B12_H$					XXXXXXXX_B
$3B13_H$					XXXXXXXX_B
$3B14_H$					XXXXXXXX_B
$3B15_H$					XXXXXXXX_B
$3B16_H$					XXXXXXXX_B
$3B17_H$					XXXXXXXX_B
$3B18_H$					XXXXXXXX_B
$3B19_H$					XXXXXXXX_B
$3B1A_H$					XXXXXXXX_B
$3B1B_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 27 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B1C_H$	General-purpose RAM	-	(R/W)	CAN1	XXXXXXXX_B
$3B1D_H$					XXXXXXXX_B
$3B1E_H$					XXXXXXXX_B
$3B1F_H$					XXXXXXXX_B
$3B20_H$	ID register 0	IDR0	(R/W)		XXXXXXXX_B
$3B21_H$					XXXXXXXX_B
$3B22_H$					XXXXXXXX_B
$3B23_H$					XXXXXXXX_B
$3B24_H$	ID register 1	IDR1	(R/W)		XXXXXXXX_B
$3B25_H$					XXXXXXXX_B
$3B26_H$					XXXXXXXX_B
$3B27_H$					XXXXXXXX_B
$3B28_H$	ID register 2	IDR2	(R/W)		XXXXXXXX_B
$3B29_H$					XXXXXXXX_B
$3B2A_H$					XXXXXXXX_B
$3B2B_H$					XXXXXXXX_B
$3B2C_H$	ID register 3	IDR3	(R/W)		XXXXXXXX_B
$3B2D_H$					XXXXXXXX_B
$3B2E_H$					XXXXXXXX_B
$3B2F_H$					XXXXXXXX_B
$3B30_H$	ID register 4	IDR4	(R/W)		XXXXXXXX_B
$3B31_H$					XXXXXXXX_B
$3B32_H$					XXXXXXXX_B
$3B33_H$					XXXXXXXX_B
$3B34_H$	ID register 5	IDR5	(R/W)		XXXXXXXX_B
$3B35_H$					XXXXXXXX_B
$3B36_H$					XXXXXXXX_B
$3B37_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 28 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B38_H$	ID register 6	IDR6	(R/W)	CAN1	XXXXXXXX_B
$3B39_H$					XXXXXXXX_B
$3B3A_H$					XXXXXXXX_B
$3B3B_H$					XXXXXXXX_B
$3B3C_H$	ID register 7	IDR7	(R/W)		XXXXXXXX_B
$3B3D_H$					XXXXXXXX_B
$3B3E_H$					XXXXXXXX_B
$3B3F_H$					XXXXXXXX_B
$3B40_H$	ID register 8	IDR8	(R/W)		XXXXXXXX_B
$3B41_H$					XXXXXXXX_B
$3B42_H$					XXXXXXXX_B
$3B43_H$					XXXXXXXX_B
$3B44_H$	ID register 9	IDR9	(R/W)		XXXXXXXX_B
$3B45_H$					XXXXXXXX_B
$3B46_H$					XXXXXXXX_B
$3B47_H$					XXXXXXXX_B
$3B48_H$	ID register 10	IDR10	(R/W)		XXXXXXXX_B
$3B49_H$					XXXXXXXX_B
$3B4A_H$					XXXXXXXX_B
$3B4B_H$					XXXXXXXX_B
$3B4C_H$	ID register 11	IDR11	(R/W)		XXXXXXXX_B
$3B4D_H$					XXXXXXXX_B
$3B4E_H$					XXXXXXXX_B
$3B4F_H$					XXXXXXXX_B
$3B50_H$	ID register 12	IDR12	(R/W)		XXXXXXXX_B
$3B51_H$					XXXXXXXX_B
$3B52_H$					XXXXXXXX_B
$3B53_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 29 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B54_H$	ID register 13	IDR13	(R/W)	CAN1	XXXXXXXX_B
$3B55_H$					XXXXXXXX_B
$3B56_H$					XXXXXXXX_B
$3B57_H$					XXXXXXXX_B
$3B58_H$	ID register 14	IDR14	(R/W)		XXXXXXXX_B
$3B59_H$					XXXXXXXX_B
$3B5A_H$					XXXXXXXX_B
$3B5B_H$					XXXXXXXX_B
$3B5C_H$	ID register 15	IDR15	(R/W)		XXXXXXXX_B
$3B5D_H$					XXXXXXXX_B
$3B5E_H$					XXXXXXXX_B
$3B5F_H$					XXXXXXXX_B
$3B60_H$	DLC register 0	DLCR0	(R/W)		XXXXXXXX_B
$3B61_H$					XXXXXXXX_B
$3B62_H$	DLC register 1	DLCR1	(R/W)		XXXXXXXX_B
$3B63_H$					XXXXXXXX_B
$3B64_H$	DLC register 2	DLCR2	(R/W)		XXXXXXXX_B
$3B65_H$					XXXXXXXX_B
$3B66_H$	DLC register 3	DLCR3	(R/W)		XXXXXXXX_B
$3B67_H$					XXXXXXXX_B
$3B68_H$	DLC register 4	DLCR4	(R/W)		XXXXXXXX_B
$3B69_H$					XXXXXXXX_B
$3B6A_H$	DLC register 5	DLCR5	(R/W)		XXXXXXXX_B
$3B6B_H$					XXXXXXXX_B
$3B6C_H$	DLC register 6	DLCR6	(R/W)		XXXXXXXX_B
$3B6D_H$					XXXXXXXX_B
$3B6E_H$	DLC register 7	DLCR7	(R/W)		XXXXXXXX_B
$3B6F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 30 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B70_H$	DLC register 8	DLCR8	(R/W)	CAN1	XXXXXXXX_B
$3B71_H$					XXXXXXXX_B
$3B72_H$	DLC register 9	DLCR9	(R/W)		XXXXXXXX_B
$3B73_H$					XXXXXXXX_B
$3B74_H$	DLC register 10	DLCR10	(R/W)		XXXXXXXX_B
$3B75_H$					XXXXXXXX_B
$3B76_H$	DLC register 11	DLCR11	(R/W)		XXXXXXXX_B
$3B77_H$					XXXXXXXX_B
$3B78_H$	DLC register 12	DLCR12	(R/W)		XXXXXXXX_B
$3B79_H$					XXXXXXXX_B
$3B7A_H$	DLC register 13	DLCR13	(R/W)		XXXXXXXX_B
$3B7B_H$					XXXXXXXX_B
$3B7C_H$	DLC register 14	DLCR14	(R/W)		XXXXXXXX_B
$3B7D_H$					XXXXXXXX_B
$3B7E_H$	DLC register 15	DLCR15	(R/W)		XXXXXXXX_B
$3B7F_H$					XXXXXXXX_B
$3B80_H$	Data register 0 (8 bytes)	DTR0	(R/W)		XXXXXXXX_B
$3B81_H$					XXXXXXXX_B
$3B82_H$					XXXXXXXX_B
$3B83_H$					XXXXXXXX_B
$3B84_H$					XXXXXXXX_B
$3B85_H$					XXXXXXXX_B
$3B86_H$					XXXXXXXX_B
$3B87_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 31 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3B88_H$	Data register 1 (8 bytes)	DTR1	(R/W)	CAN1	XXXXXXXX_B
$3B89_H$					XXXXXXXX_B
$3B8A_H$					XXXXXXXX_B
$3B8B_H$					XXXXXXXX_B
$3B8C_H$					XXXXXXXX_B
$3B8D_H$					XXXXXXXX_B
$3B8E_H$					XXXXXXXX_B
$3B8F_H$					XXXXXXXX_B
$3B90_H$	Data register 2 (8 bytes)	DTR2	(R/W)		XXXXXXXX_B
$3B91_H$					XXXXXXXX_B
$3B92_H$					XXXXXXXX_B
$3B93_H$					XXXXXXXX_B
$3B94_H$					XXXXXXXX_B
$3B95_H$					XXXXXXXX_B
$3B96_H$					XXXXXXXX_B
$3B97_H$					XXXXXXXX_B
$3B98_H$	Data register 3 (8 bytes)	DTR3	(R/W)		XXXXXXXX_B
$3B99_H$					XXXXXXXX_B
$3B9A_H$					XXXXXXXX_B
$3B9B_H$					XXXXXXXX_B
$3B9C_H$					XXXXXXXX_B
$3B9D_H$					XXXXXXXX_B
$3B9E_H$					XXXXXXXX_B
$3B9F_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 32 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3BA0_H$	Data register 4 (8 bytes)	DTR4	(R/W)	CAN1	XXXXXXXX_B
$3BA1_H$					XXXXXXXX_B
$3BA2_H$					XXXXXXXX_B
$3BA3_H$					XXXXXXXX_B
$3BA4_H$					XXXXXXXX_B
$3BA5_H$					XXXXXXXX_B
$3BA6_H$					XXXXXXXX_B
$3BA7_H$					XXXXXXXX_B
$3BA8_H$	Data register 5 (8 bytes)	DTR5	(R/W)		XXXXXXXX_B
$3BA9_H$					XXXXXXXX_B
$3BAA_H$					XXXXXXXX_B
$3BAB_H$					XXXXXXXX_B
$3BAC_H$					XXXXXXXX_B
$3BAD_H$					XXXXXXXX_B
$3BAE_H$					XXXXXXXX_B
$3BAF_H$					XXXXXXXX_B
$3BB0_H$	Data register 6 (8 bytes)	DTR6	(R/W)		XXXXXXXX_B
$3BB1_H$					XXXXXXXX_B
$3BB2_H$					XXXXXXXX_B
$3BB3_H$					XXXXXXXX_B
$3BB4_H$					XXXXXXXX_B
$3BB5_H$					XXXXXXXX_B
$3BB6_H$					XXXXXXXX_B
$3BB7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 33 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3BB8_H$	Data register 7 (8 bytes)	DTR7	(R/W)	CAN1	XXXXXXXX_B
$3BB9_H$					XXXXXXXX_B
$3BBA_H$					XXXXXXXX_B
$3BBB_H$					XXXXXXXX_B
$3BBC_H$					XXXXXXXX_B
$3BBD_H$					XXXXXXXX_B
$3BBE_H$					XXXXXXXX_B
$3BBF_H$					XXXXXXXX_B
$3BC0_H$	Data register 8 (8 bytes)	DTR8	(R/W)		XXXXXXXX_B
$3BC1_H$					XXXXXXXX_B
$3BC2_H$					XXXXXXXX_B
$3BC3_H$					XXXXXXXX_B
$3BC4_H$					XXXXXXXX_B
$3BC5_H$					XXXXXXXX_B
$3BC6_H$					XXXXXXXX_B
$3BC7_H$					XXXXXXXX_B
$3BC8_H$	Data register 9 (8 bytes)	DTR9	(R/W)		XXXXXXXX_B
$3BC9_H$					XXXXXXXX_B
$3BCA_H$					XXXXXXXX_B
$3BCB_H$					XXXXXXXX_B
$3BCC_H$					XXXXXXXX_B
$3BCD_H$					XXXXXXXX_B
$3BCE_H$					XXXXXXXX_B
$3BCF_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 34 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3BD0_H$	Data register 10 (8 bytes)	DTR10	(R/W)	CAN1	XXXXXXXX_B
$3BD1_H$					XXXXXXXX_B
$3BD2_H$					XXXXXXXX_B
$3BD3_H$					XXXXXXXX_B
$3BD4_H$					XXXXXXXX_B
$3BD5_H$					XXXXXXXX_B
$3BD6_H$					XXXXXXXX_B
$3BD7_H$					XXXXXXXX_B
$3BD8_H$	Data register 11 (8 bytes)	DTR11	(R/W)		XXXXXXXX_B
$3BD9_H$					XXXXXXXX_B
$3BDA_H$					XXXXXXXX_B
$3BDB_H$					XXXXXXXX_B
$3BDC_H$					XXXXXXXX_B
$3BDD_H$					XXXXXXXX_B
$3BDE_H$					XXXXXXXX_B
$3BDF_H$					XXXXXXXX_B
$3BE0_H$	Data register 12 (8 bytes)	DTR12	(R/W)		XXXXXXXX_B
$3BE1_H$					XXXXXXXX_B
$3BE2_H$					XXXXXXXX_B
$3BE3_H$					XXXXXXXX_B
$3BE4_H$					XXXXXXXX_B
$3BE5_H$					XXXXXXXX_B
$3BE6_H$					XXXXXXXX_B
$3BE7_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 35 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3BE8_H$	Data register 13 (8 bytes)	DTR13	(R/W)	CAN1	XXXXXXXX_B
$3BE9_H$					XXXXXXXX_B
$3BEA_H$					XXXXXXXX_B
$3BEB_H$					XXXXXXXX_B
$3BEC_H$					XXXXXXXX_B
$3BED_H$					XXXXXXXX_B
$3BEE_H$					XXXXXXXX_B
$3BEF_H$					XXXXXXXX_B
$3BF0_H$	Data register 14 (8 bytes)	DTR14	(R/W)		XXXXXXXX_B
$3BF1_H$					XXXXXXXX_B
$3BF2_H$					XXXXXXXX_B
$3BF3_H$					XXXXXXXX_B
$3BF4_H$					XXXXXXXX_B
$3BF5_H$					XXXXXXXX_B
$3BF6_H$					XXXXXXXX_B
$3BF7_H$					XXXXXXXX_B
$3BF8_H$	Data register 15 (8 bytes)	DTR15	(R/W)		XXXXXXXX_B
$3BF9_H$					XXXXXXXX_B
$3BFA_H$					XXXXXXXX_B
$3BFB_H$					XXXXXXXX_B
$3BFC_H$					XXXXXXXX_B
$3BFD_H$					XXXXXXXX_B
$3BFE_H$					XXXXXXXX_B
$3BFF_H$					XXXXXXXX_B
$3C00_H$	Control status register	CSR	(R/W,R)	CAN0	$0XXXX0X1_B$
$3C01_H$					$00XXX000_B$
$3C02_H$	Last event indicator register	LEIR	(R/W)		$000X0000_B$
$3C03_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 36 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3C04_H$	RX/TX error counter	RTEC	(R)	CAN0	00000000_B
$3C05_H$					00000000_B
$3C06_H$	Bit timing register	BTR	(R/W)		11111111_B
$3C07_H$					$X1111111_B$
$3C08_H$	IDE register	IDER	(R/W)		XXXXXXXX_B
$3C09_H$					XXXXXXXX_B
$3C0A_H$	Transmission RTR register	TRTRR	(R/W)		00000000_B
$3C0B_H$					00000000_B
$3C0C_H$	Remote frame receiving wait register	RFWTR	(R/W)		XXXXXXXX_B
$3C0D_H$					XXXXXXXX_B
$3C0E_H$	Transmission interrupt enable register	TIER	(R/W)		00000000_B
$3C0F_H$					00000000_B
$3C10_H$	Acceptance mask select register	AMSR	(R/W)		XXXXXXXX_B
$3C11_H$					XXXXXXXX_B
$3C12_H$					XXXXXXXX_B
$3C13_H$					XXXXXXXX_B
$3C14_H$	Acceptance mask register 0	AMR0	(R/W)		XXXXXXXX_B
$3C15_H$					XXXXXXXX_B
$3C16_H$					XXXXXXXX_B
$3C17_H$					XXXXXXXX_B
$3C18_H$	Acceptance mask register 1	AMR1	(R/W)		XXXXXXXX_B
$3C19_H$					XXXXXXXX_B
$3C1A_H$					XXXXXXXX_B
$3C1B_H$					XXXXXXXX_B
$3C1C_H$ to $3CFF_H$	(Not available)				
$3D00_H$	Control status register	CSR	(R/W,R)	CAN1	$0XXXX0X1_B$
$3D01_H$					$00XXX000_B$

Table A-3 I/O Map (Sheet 37 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3D02_H$	Last event indicator register	LEIR	(R/W)		$000X0000_B$
$3D03_H$					XXXXXXXX_B
$3D04_H$	RX/TX error counter	RTEC	(R)		00000000_B
$3D05_H$					00000000_B
$3D06_H$	Bit timing register	BTR	(R/W)		11111111_B
$3D07_H$					$X1111111_B$
$3D08_H$	IDE register	IDER	(R/W)		XXXXXXXX_B
$3D09_H$					XXXXXXXX_B
$3D0A_H$	Transmission RTR register	TRTRR	(R/W)	CAN1	00000000_B
$3D0B_H$					00000000_B
$3D0C_H$	Remote frame receiving wait register	RFWTR	(R/W)		XXXXXXXX_B
$3D0D_H$					XXXXXXXX_B
$3D0E_H$	Transmission interrupt enable register	TIER	(R/W)		00000000_B
$3D0F_H$					00000000_B
$3D10_H$	Acceptance mask select register	AMSR	(R/W)		XXXXXXXX_B
$3D11_H$					XXXXXXXX_B
$3D12_H$					XXXXXXXX_B
$3D13_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 38 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3D14_H$	Acceptance mask register 0	AMR0	(R/W)	CAN1	XXXXXXXX_B
$3D15_H$					XXXXXXXX_B
$3D16_H$					XXXXXXXX_B
$3D17_H$					XXXXXXXX_B
$3D18_H$	Acceptance mask register 1	AMR1	(R/W)		XXXXXXXX_B
$3D19_H$					XXXXXXXX_B
$3D1A_H$					XXXXXXXX_B
$3D1B_H$					XXXXXXXX_B
$3D1C_H$ to $3DFF_H$	(Not available)				
$3E00_H$	Control status register	CSR	(R/W,R)	CAN2	$0XXXX0X1_B$
$3E01_H$					$00XXX000_B$
$3E02_H$	Last event indicator register	LEIR	(R/W)		$000X0000_B$
$3E03_H$					XXXXXXXX_B
$3E04_H$	RX/TX error counter	RTEC	(R)		00000000_B
$3E05_H$					00000000_B
$3E06_H$	Bit timing register	BTR	(R/W)		11111111_B
$3E07_H$					$X1111111_B$
$3E08_H$	IDE register	IDER	(R/W)		XXXXXXXX_B
$3E09_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 39 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3E0A_H$	Transmission RTR register	TRTRR	(R/W)	CAN2	00000000_B
$3E0B_H$					00000000_B
$3E0C_H$	Remote frame receiving wait register	RFWTR	(R/W)		XXXXXXXX_B
$3E0D_H$					XXXXXXXX_B
$3E0E_H$	Transmission interrupt enable register	TIER	(R/W)		00000000_B
$3E0F_H$					00000000_B
$3E10_H$	Acceptance mask select register	AMSR	(R/W)		XXXXXXXX_B
$3E11_H$					XXXXXXXX_B
$3E12_H$					XXXXXXXX_B
$3E13_H$					XXXXXXXX_B
$3E14_H$	Acceptance mask register 0	AMR0	(R/W)		XXXXXXXX_B
$3E15_H$					XXXXXXXX_B
$3E16_H$					XXXXXXXX_B
$3E17_H$					XXXXXXXX_B
$3E18_H$	Acceptance mask register 1	AMR1	(R/W)		XXXXXXXX_B
$3E19_H$					XXXXXXXX_B
$3E1A_H$					XXXXXXXX_B
$3E1B_H$					XXXXXXXX_B
$3E1C_H$ to $3E1F_H$	(Disabled)				

Table A-3 I/O Map (Sheet 40 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3F00_H$	Control status register	CSR	(R/W,R)	CAN3	$0XXXX0X1_B$
$3F01_H$					$00XXX000_B$
$3F02_H$	Last event indicator register	LEIR	(R/W)		$000X0000_B$
$3F03_H$					XXXXXXXX_B
$3F04_H$	RX/TX error counter	RTEC	(R)		00000000_B
$3F05_H$					00000000_B
$3F06_H$	Bit timing register	BTR	(R/W)		11111111_B
$3F07_H$					$X1111111_B$
$3F08_H$	IDE register	IDER	(R/W)		XXXXXXXX_B
$3F09_H$					XXXXXXXX_B
$3F0A_H$	Transmission RTR register	TRTRR	(R/W)		00000000_B
$3F0B_H$					00000000_B
$3F0C_H$	Remote frame receiving wait register	RFWTR	(R/W)		XXXXXXXX_B
$3F0D_H$					XXXXXXXX_B
$3F0E_H$	Transmission interrupt enable register	TIER	(R/W)		00000000_B
$3F0F_H$					00000000_B
$3F10_H$	Acceptance mask select register	AMSR	(R/W)		XXXXXXXX_B
$3F11_H$					XXXXXXXX_B
$3F12_H$					XXXXXXXX_B
$3F13_H$					XXXXXXXX_B

Table A-3 I/O Map (Sheet 41 of 41)

Address	Register Name	Abbreviation	Access	Peripheral Functions	Initial Value
$3F14_H$	Acceptance mask register 0	AMR0	(R/W)	CAN3	XXXXXXXX_B
$3F15_H$					XXXXXXXX_B
$3F16_H$					XXXXXXXX_B
$3F17_H$					XXXXXXXX_B
$3F18_H$	Acceptance mask register 1	AMR1	(R/W)		XXXXXXXX_B
$3F19_H$					XXXXXXXX_B
$3F1A_H$					XXXXXXXX_B
$3F1B_H$					XXXXXXXX_B
$3F1C_H$ to $3FFF_H$	(Not available)				

APPENDIX B Instructions

APPENDIX B describes the instructions used by the F^{2}MC-16LX.

B.1 Instruction Types

B.2 Addressing

B.3 Direct Addressing

B.4 Indirect Addressing

B.5 Execution Cycle Count

B.6 Effective address field

B.7 How to Read the Instruction List

B.8 F^{2}MC-16LX Instruction List

B.9 Instruction Map

B.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

B.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 B.2-1 lists the address formats specified by the effective address field.

Table B.2-1 Effective Address Field

Code	Representation			Address format	Default bank
00	R0	RW0	RL0	Register direct: Individual parts correspond to the byte, word, and long word types in order from the left.	None
01	R1	RW1	(RL0)		
02	R2	RW2	RL1		
03	R3	RW3	(RL1)		
04	R4	RW4	RL2		
05	R5	RW5	(RL2)		
06	R6	RW6	RL3		
07	R7	RW7	(RL3)		
08	@RW0			Register indirect	DTB
09	@RW1				DTB
0A	@RW2				ADB
0B	@RW3				SPB
0C	@RW0+			Register indirect with post increment	DTB
0D	@RW1+				DTB
0E	@RW2+				ADB
0F	@RW3+				SPB
10	@RW0+disp8			Register indirect with 8-bit displacement	DTB
11	@RW1+disp8				DTB
12	@RW2+disp8				ADB
13	@RW3+disp8				SPB
14	@RW4+disp8			Register indirect with 8-bit displacement	DTB
15	@RW5+disp8				DTB
16	@RW6+disp8				ADB
17	@RW7+disp8				SPB
18	@RW0+disp16			Register indirect with 16-bit displacement	DTB
19	@RW1+disp16				DTB
1A	@RW2+disp16				ADB
1B	@RW3+disp16				SPB
1C	@RW0+RW7			Register indirect with index	DTB
1D	@RW1+RW7			Register indirect with index	DTB
1E	@PC+disp16			PC indirect with 16-bit displacement	PCB
1F	addr16			Direct address	DTB

B.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 B.3-1 Example of Immediate Addressing (#imm)

MOVW A, #01212H (This instruction stores the operand value in A.)

Before execution A | 2233 | 4455 |

After execution A | 4455 | 1212 | (Some instructions transfer AL to AH.)

- Register direct addressing

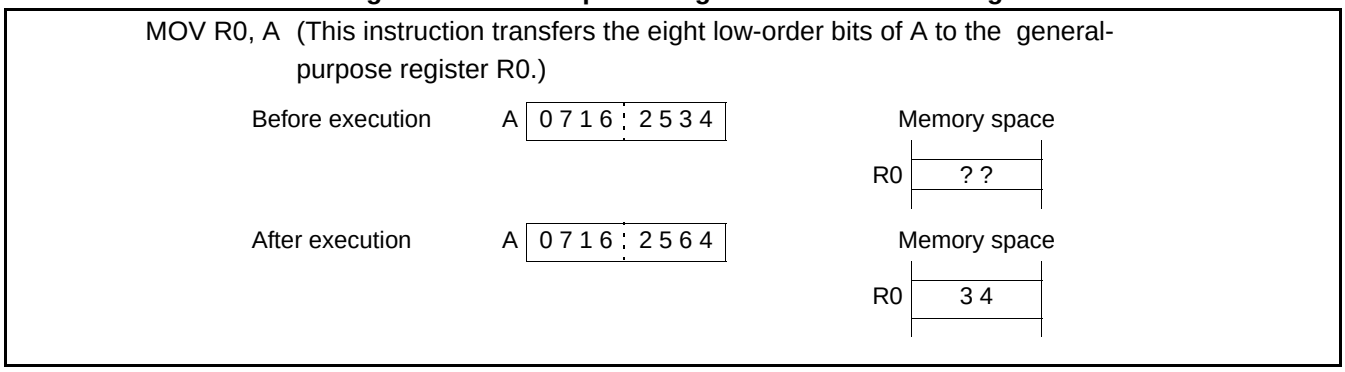
Specify a register explicitly as an operand. Table B.3-1 lists the registers that can be specified. Figure B.3-2 shows an example of register direct addressing.

Table B.3-1 Direct Addressing Registers

General-purpose register	Byte	R0, R1, R2, R3, R4, R5, R6, R7
	Word	RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7
	Long word	RL0, RL1, RL2, RL3
Special-purpose register	Accumulator	A, AL
	Pointer	SP *
	Bank	PCB, DTB, USB, SSB, ADB
	Page	DPR
	Control	PS, CCR, RP, ILM

*: One of the user stack pointer (USP) and system stack pointer (SSP) is selected and used depending on the value of the S flag bit in the condition code register (CCR). For branch instructions, the program counter (PC) is not specified in an instruction operand but is specified implicitly.

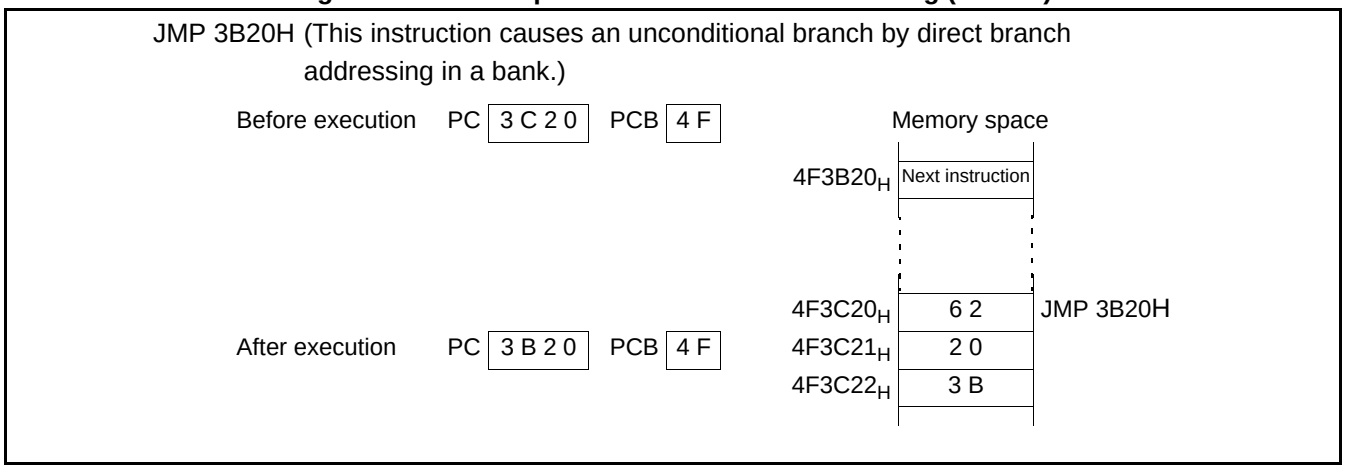
Figure B.3-2 Example of Register Direct Addressing



- Direct branch addressing (addr16)

Specify an offset explicitly for the branch destination address. The size of the offset is 16 bits, which indicates the branch destination in the logical address space. Direct branch addressing is used for an unconditional branch, subroutine call, or software interrupt instruction. Bit23 to bit16 of the address are specified by the program counter bank register (PCB).

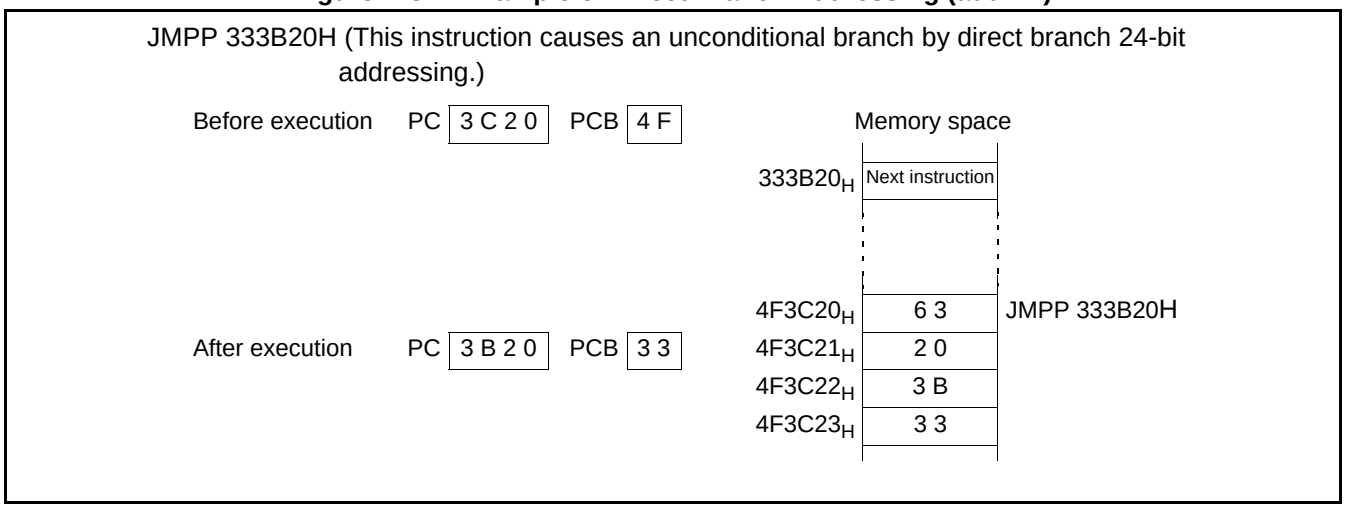
Figure B.3-3 Example of Direct Branch Addressing (addr16)



- Physical direct branch addressing (addr24)

Specify an offset explicitly for the branch destination address. The size of the offset is 24 bits. Physical direct branch addressing is used for unconditional branch, subroutine call, or software interrupt instruction.

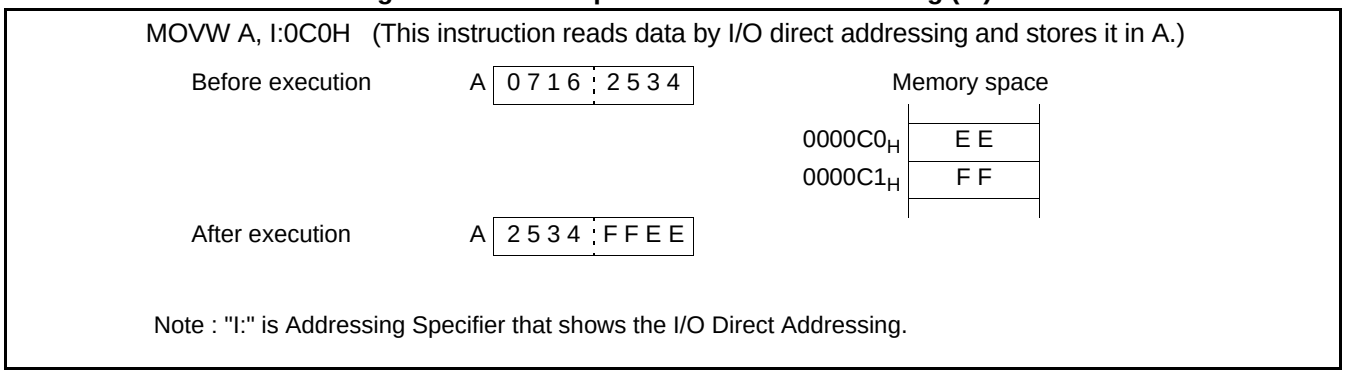
Figure B.3-4 Example of Direct Branch Addressing (addr24)



- I/O direct addressing (io)

Specify an 8-bit offset explicitly for the memory address in an operand. The I/O address space in the physical address space from 000000_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 B.3-5 Example of I/O Direct Addressing (io)



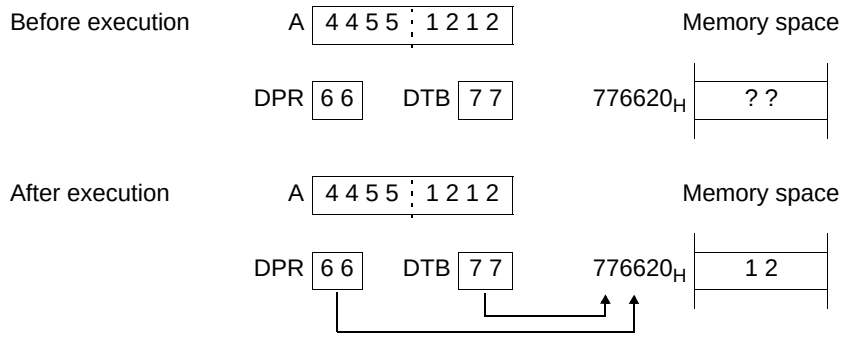
Note : "I:" is Addressing Specifier that shows the I/O Direct Addressing.

- 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 B.3-6 Example of Abbreviated Direct Addressing (dir)

MOV S:20H, A (This instruction writes the contents of the eight low-order bits of A in abbreviated direct addressing mode.)



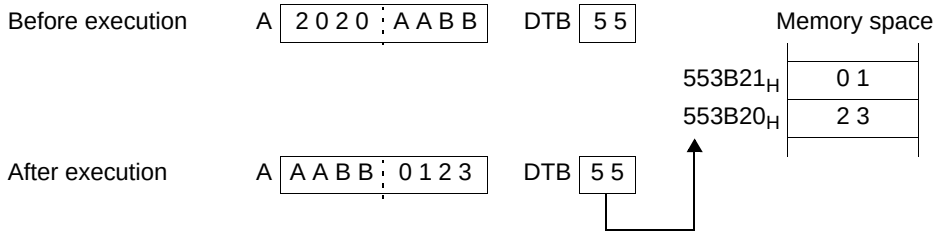
Note : "S:" is Addressing Specifier that shows the Abbreviated Direct Addressing.

- 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 B.3-7 Example of Direct Addressing (addr16)

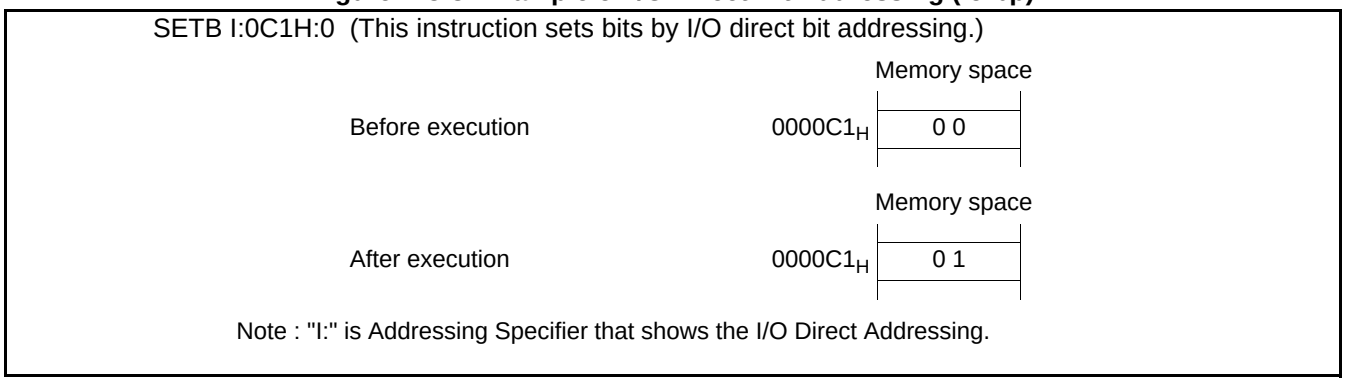
MOVW A, 3B20H (This instruction reads data by direct addressing and stores it in A.)



- 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).

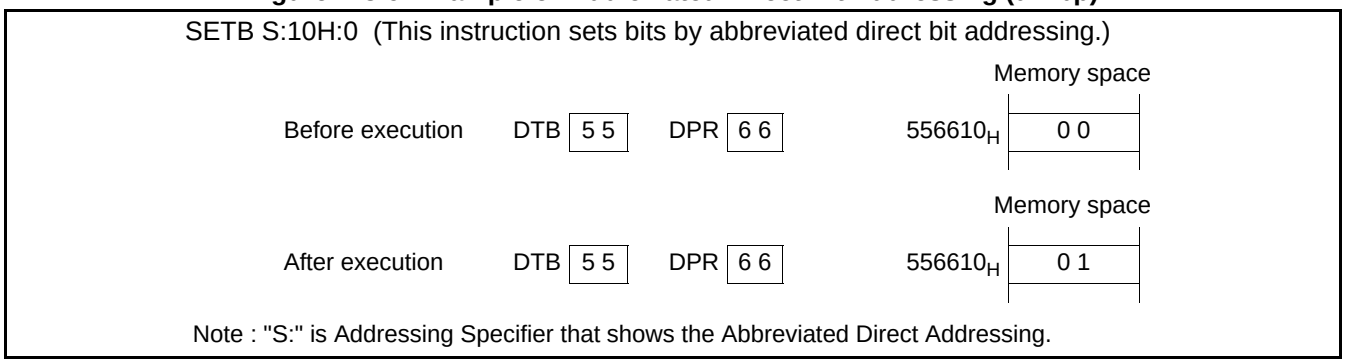
Figure B.3-8 Example of I/O Direct Bit Addressing (io:bp)



- Abbreviated direct bit addressing (dir:bp)

Specify the eight low-order bits of a memory address explicitly in an operand. Address bits 8 to 15 are specified by the direct page register (DPR). Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number indicates the least significant bit (LSB).

Figure B.3-9 Example of Abbreviated Direct Bit Addressing (dir:bp)

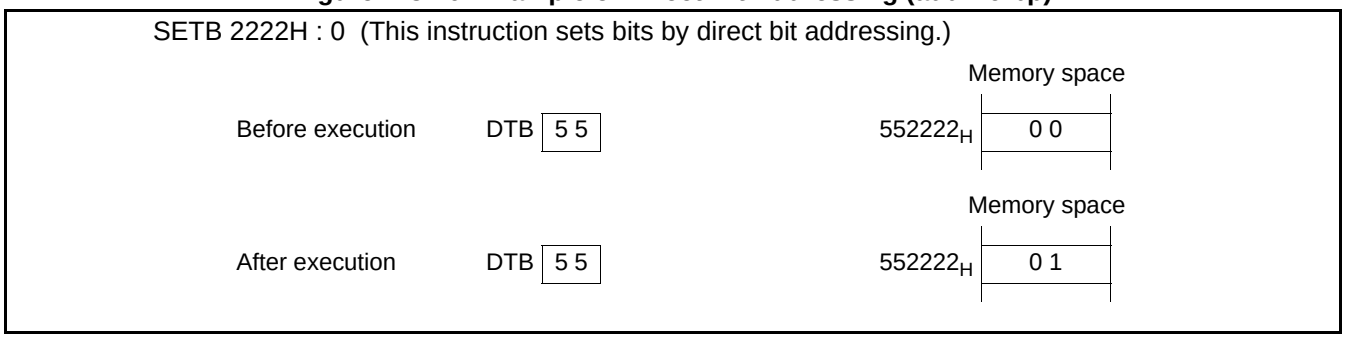


- Direct bit addressing (addr16:bp)

Specify arbitrary bits in 64 kilobytes explicitly. Address bits 16 to 23 are specified by the data bank register (DTB). Bit positions are indicated by ":bp", where the larger number indicates the most significant bit (MSB) and the lower number

indicates the least significant bit (LSB).

Figure B.3-10 Example of Direct Bit Addressing (addr16:bp)



- Vector Addressing (#vct)

Specify vector data in an operand to indicate the branch destination address. There are two sizes for vector numbers: 4 bits and 8 bits. Vector addressing is used for a subroutine call or software interrupt instruction.

Figure B.3-11 Example of Vector Addressing (#vct)

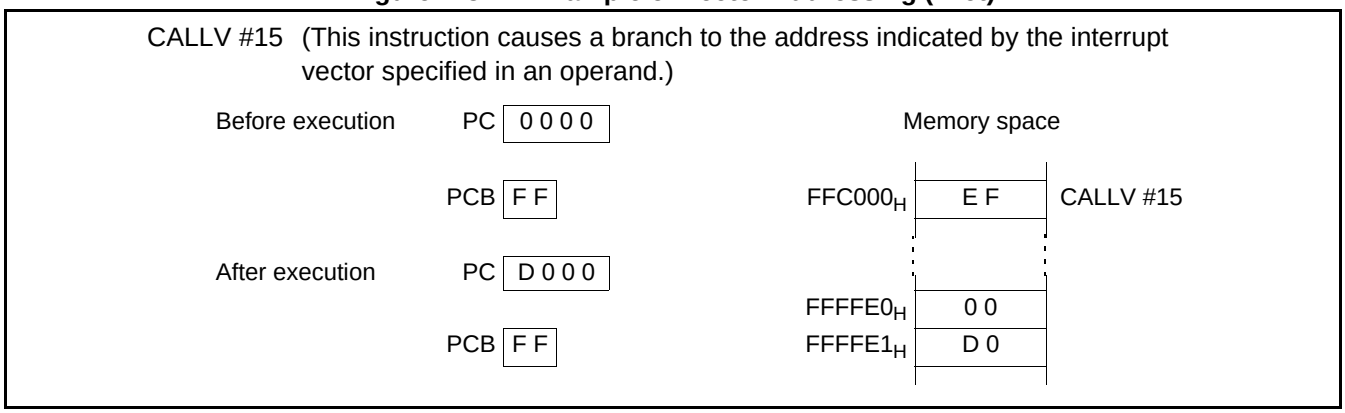


Table B.3-2 CALLV Vector List (Sheet 1 of 2)

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	$XXFFEA_H$	$XXFFEB_H$

Table B.3-2 CALLV Vector List (Sheet 2 of 2)

Instruction	Vector address L	Vector address 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 B.3-2).

B.4 Indirect Addressing

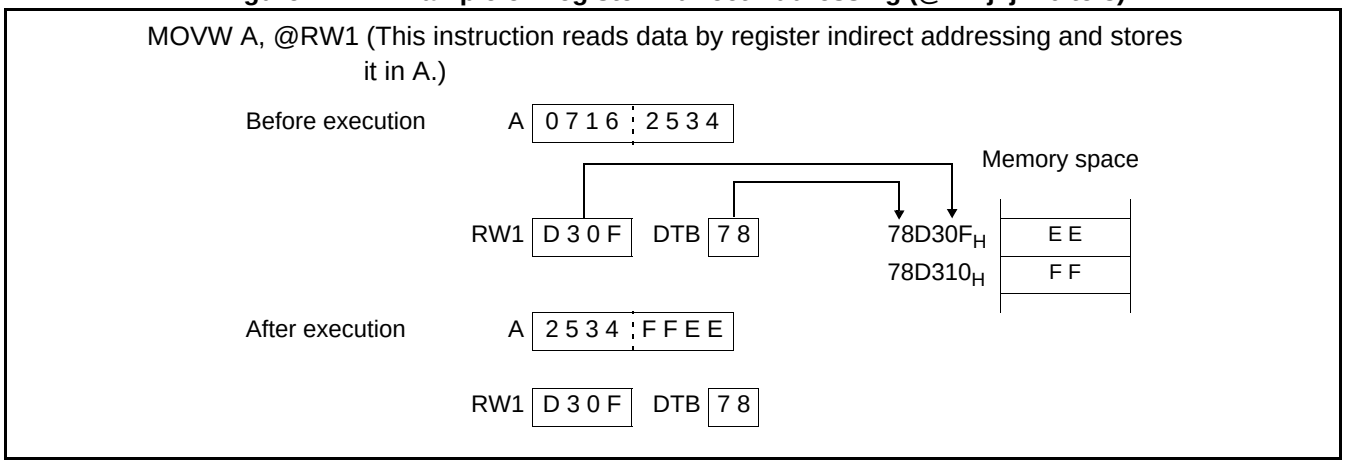
In indirect addressing mode, an address is specified indirectly by the address data of an operand.

■ 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 B.4-1 Example of Register Indirect Addressing (@RWj j = 0 to 3)

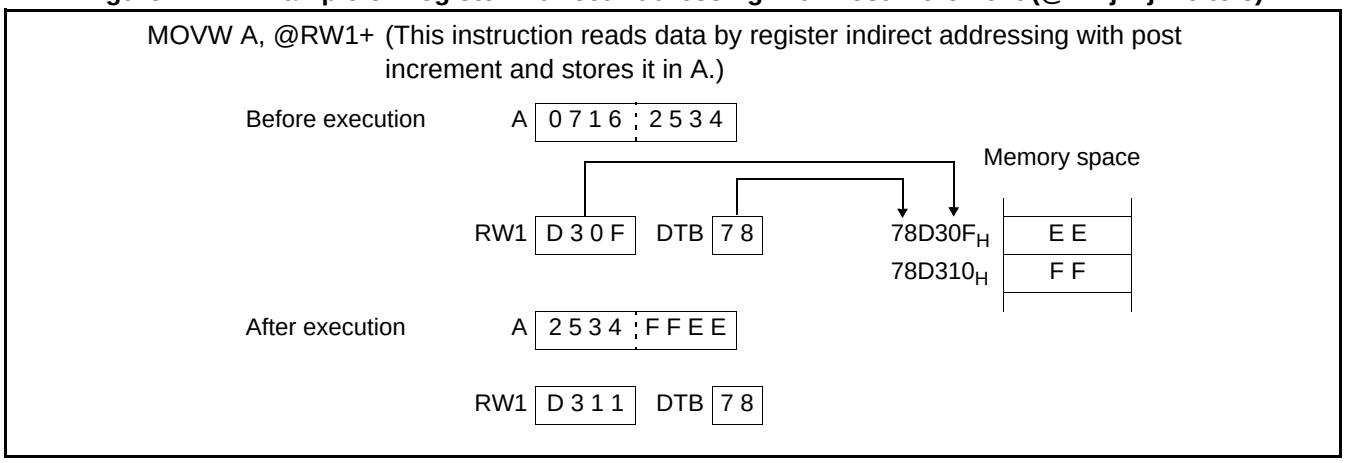


- Register indirect addressing with post increment (@RWj+ j = 0 to 3)

Memory is accessed using the contents of general-purpose register RWj as an address. After operand operation, RWj is incremented by the operand size (1 for a byte, 2 for a word, or 4 for a long word). Address bits 16 to 23 are indicated by the data bank register (DTB) when RW0 or RW1 is used, system stack bank register (SSB) or user stack bank register (USB) when RW3 is used, or additional data bank register (ADB) when RW2 is used.

If the post increment results in the address of the register that specifies the increment, the incremented value is referenced after that. In this case, if the next instruction is a write instruction, priority is given to writing by an instruction and, therefore, the register that would be incremented becomes write data.

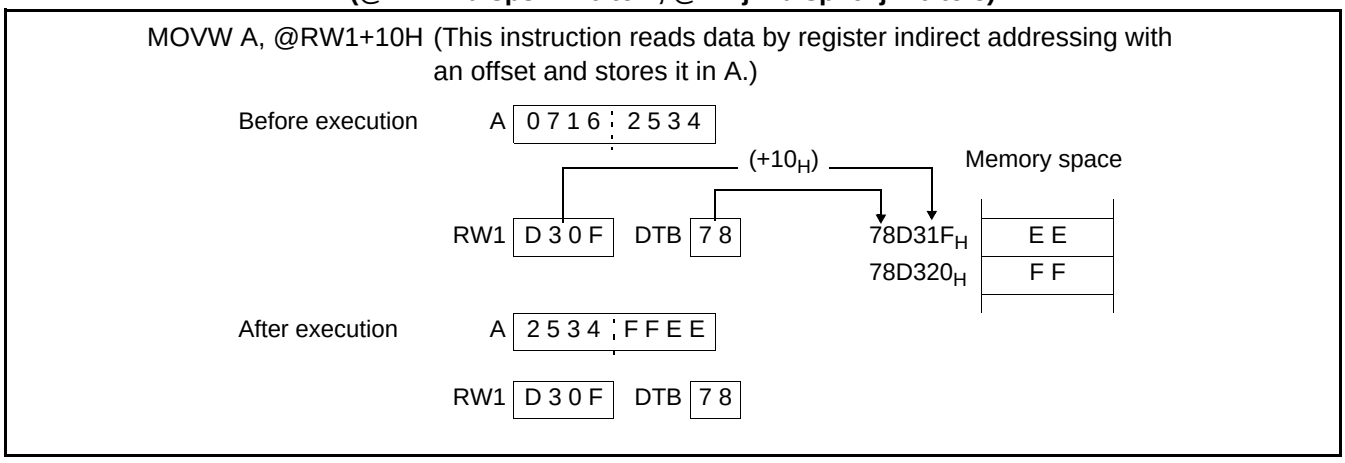
Figure B.4-2 Example of Register Indirect Addressing with Post Increment (@RWj+ j = 0 to 3)



- Register indirect addressing with offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)

Memory is accessed using the address obtained by adding an offset to the contents of general-purpose register RWj. Two types of offset, byte and word offsets, are used. They are added as signed numeric values. Address bits 16 to 23 are indicated by the data bank register (DTB) when RW0, RW1, RW4, or RW5 is used, system stack bank register (SSB) or user stack bank register (USB) when RW3 or RW7 is used, or additional data bank register (ADB) when RW2 or RW6 is used.

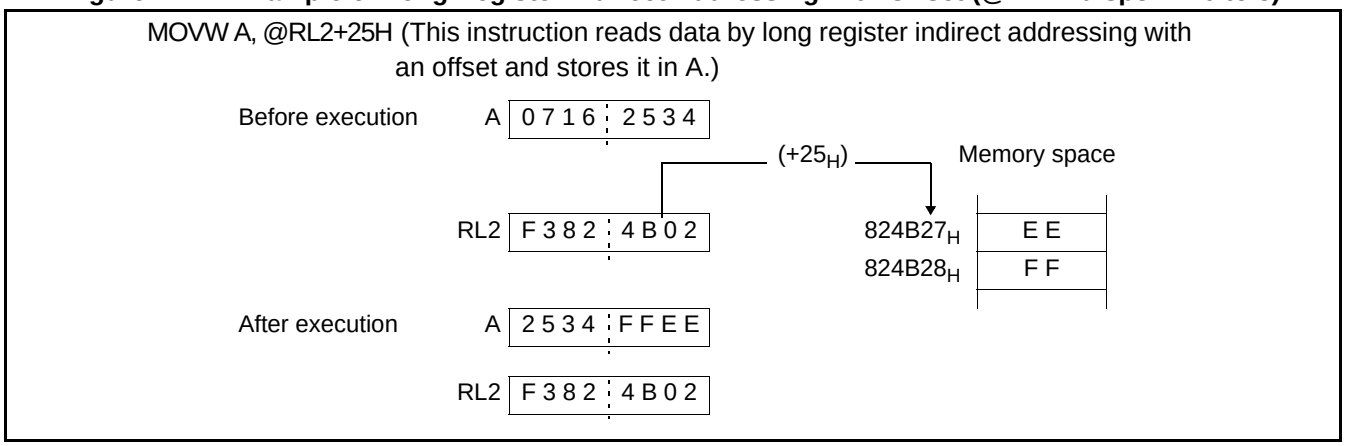
Figure B.4-3 Example of Register Indirect Addressing with Offset (@RWi + disp8 i = 0 to 7, @RWj + disp16 j = 0 to 3)



- Long register indirect addressing with offset (@RLi + disp8 i = 0 to 3)

Memory is accessed using the address that is the 24 low-order bits obtained by adding an offset to the contents of general-purpose register RLi. The offset is 8-bits long and is added as a signed numeric value.

Figure B.4-4 Example of Long Register Indirect Addressing with Offset (@RLi + disp8 i = 0 to 3)



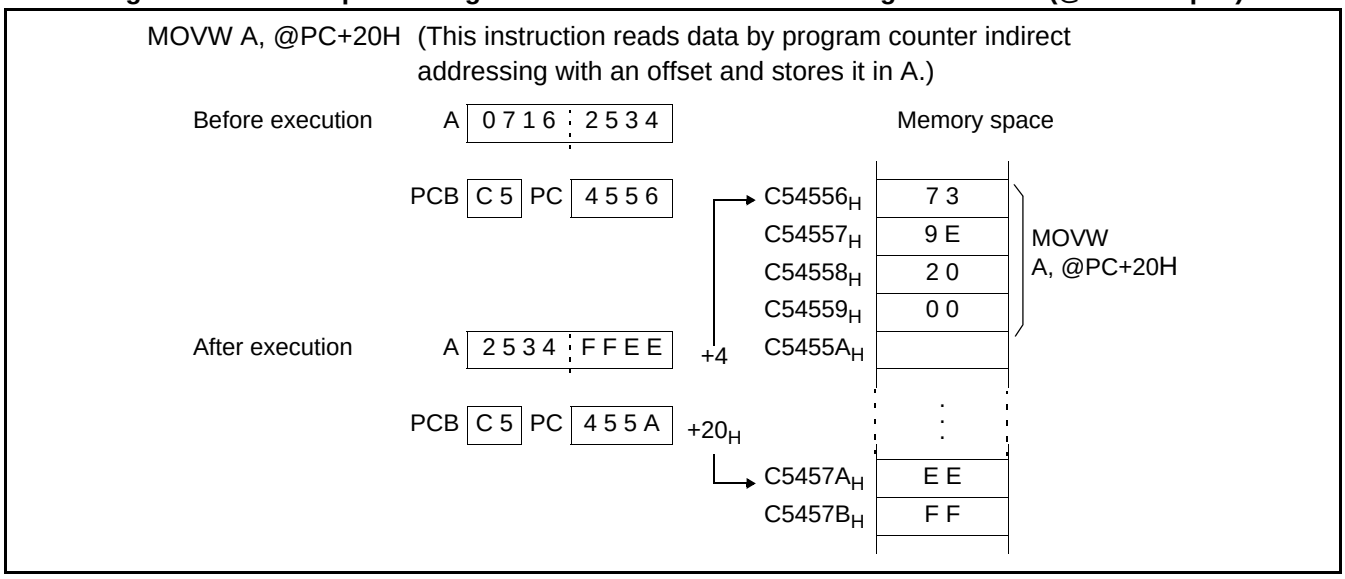
- Program counter indirect addressing with offset (@PC + disp16)

Memory is accessed using the address indicated by (instruction address + 4 + disp16). The offset is one word long. Address bits 16 to 23 are specified by the program counter bank register (PCB). Note that the operand address of each of the following instructions is not deemed to be (next instruction address + disp16):

- DBNZ eam, rel
- DWBNZ eam, rel
- CBNE eam, #imm8, rel
- CWBNE eam, #imm16, rel
- MOV eam, #imm8

- MOVW eam, #imm16

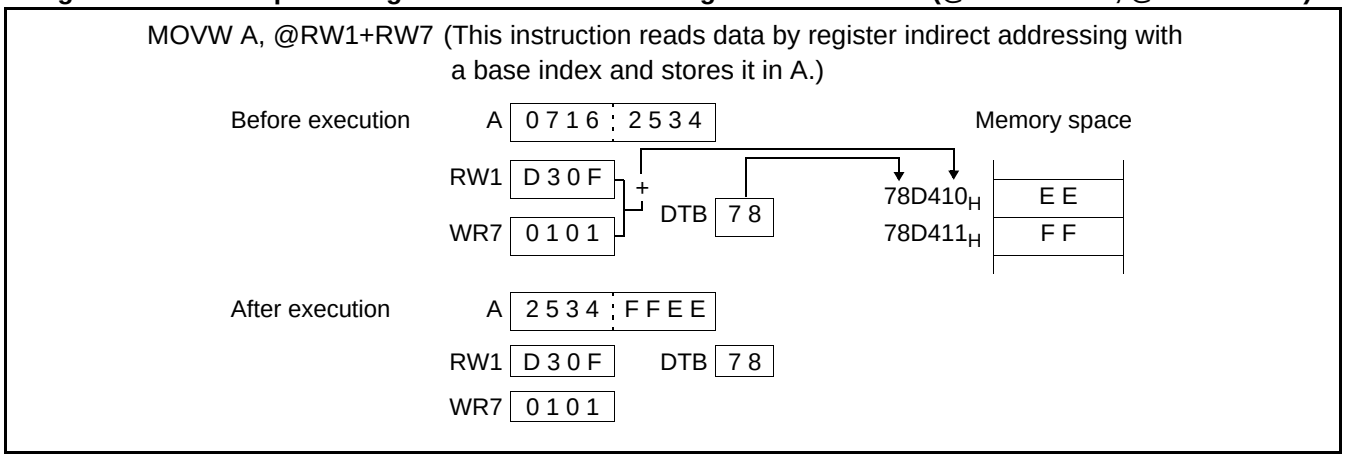
Figure B.4-5 Example of Program Counter Indirect Addressing with Offset (@PC + disp16)



- Register indirect addressing with base index (@RW0 + RW7, @RW1 + RW7)

Memory is accessed using the address determined by adding RW0 or RW1 to the contents of general-purpose register RW7. Address bits 16 to 23 are indicated by the data bank register (DTB).

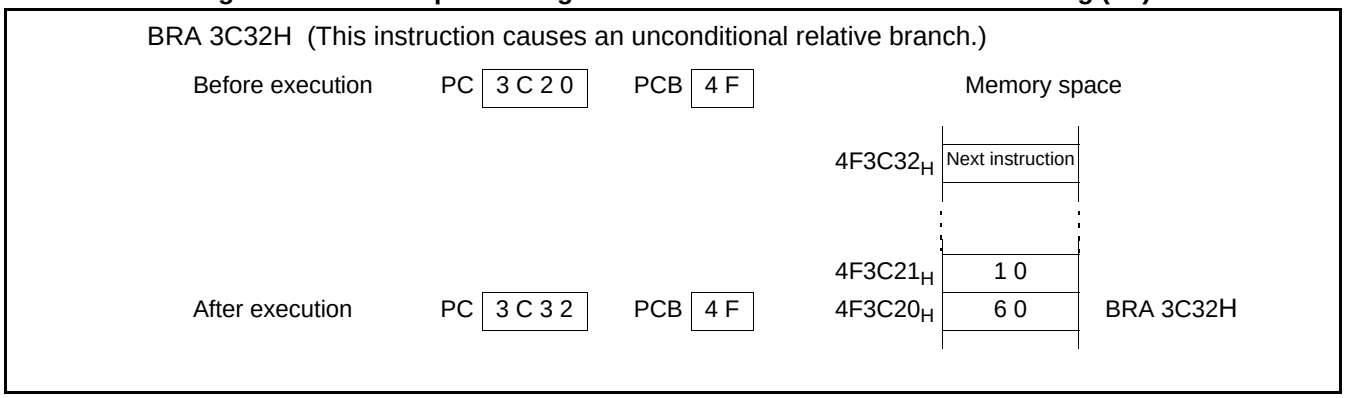
Figure B.4-6 Example of Register Indirect Addressing with Base Index (@RW0 + RW7, @RW1 + RW7)



- Program counter relative branch addressing (rel)

The address of the branch destination is a value determined by adding an 8-bit offset to the program counter (PC) value. If the result of addition exceeds 16 bits, bank register incrementing or decrementing is not performed and the excess part is ignored, and therefore the address is contained within a 64-kilobyte bank. This addressing is used for both conditional and unconditional branch instructions. Address bits 16 to 23 are indicated by the program counter bank register (PCB).

Figure B.4-7 Example of Program Counter Relative Branch Addressing (rel)



- Register list (rlst)

Specify a register to be pushed onto or popped from a stack.

Figure B.4-8 Configuration of the Register List

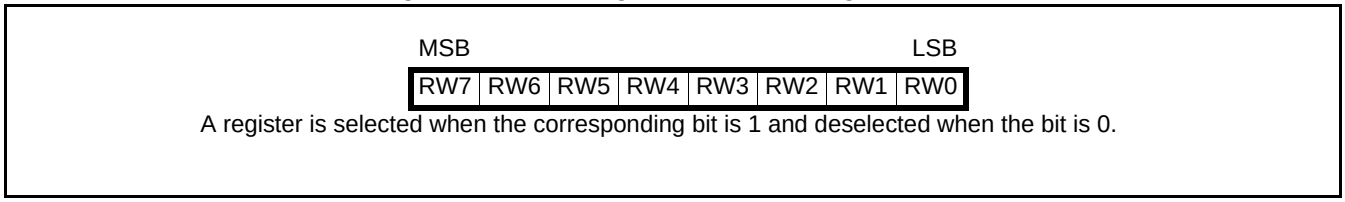
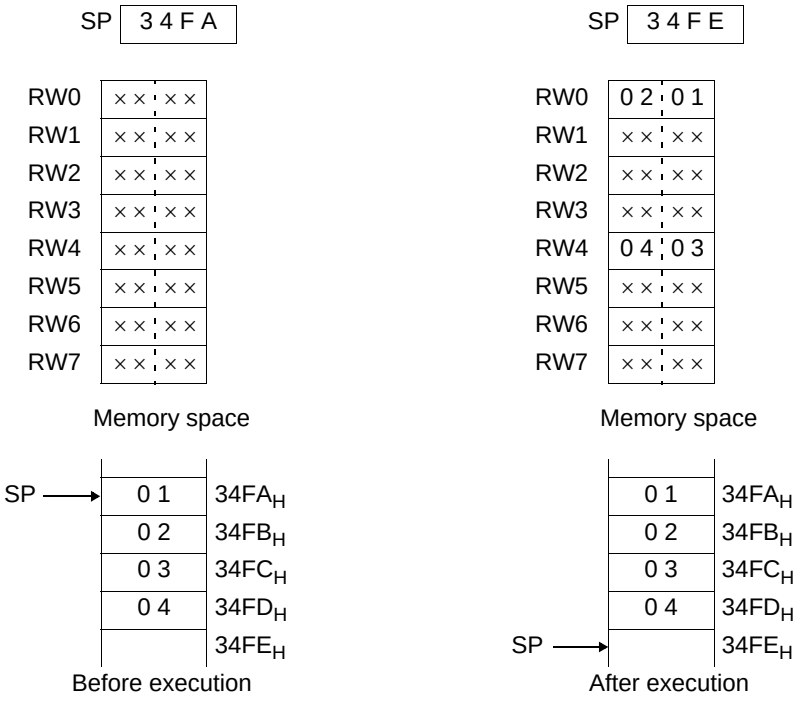


Figure B.4-9 Example of Register List (rlst)

POPW RW0, RW4 (This instruction transfers memory data indicated by the SP to multiple word registers indicated by the register list.)

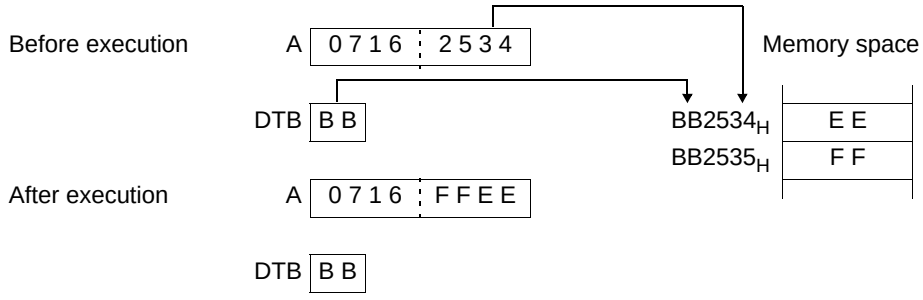


- Accumulator indirect addressing (@A)

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 B.4-10 Example of Accumulator Indirect Addressing (@A)

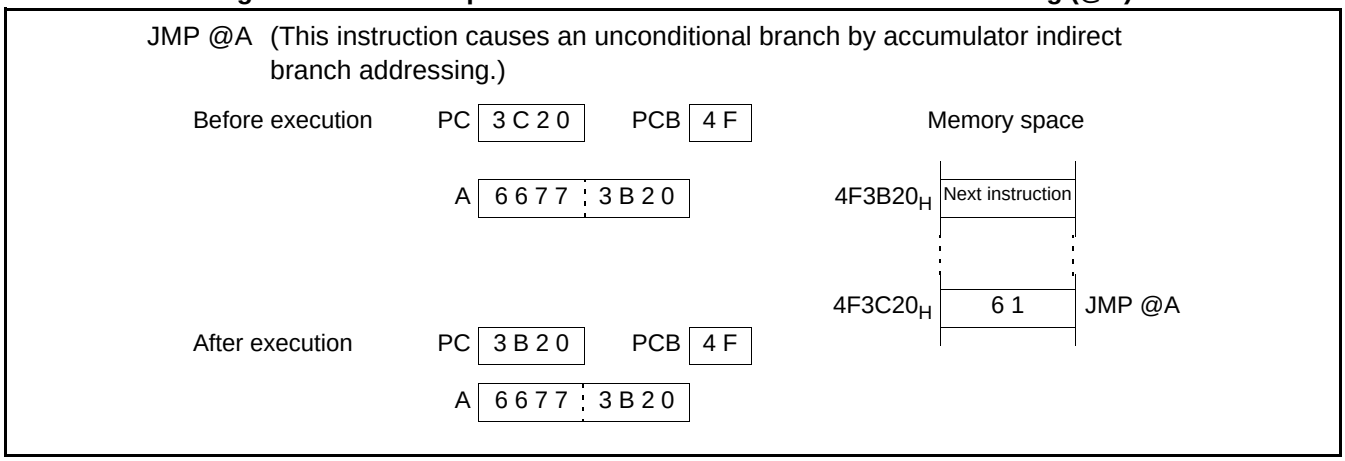
MOVW A, @A (This instruction reads data by accumulator indirect addressing and stores it in A.)



- 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.

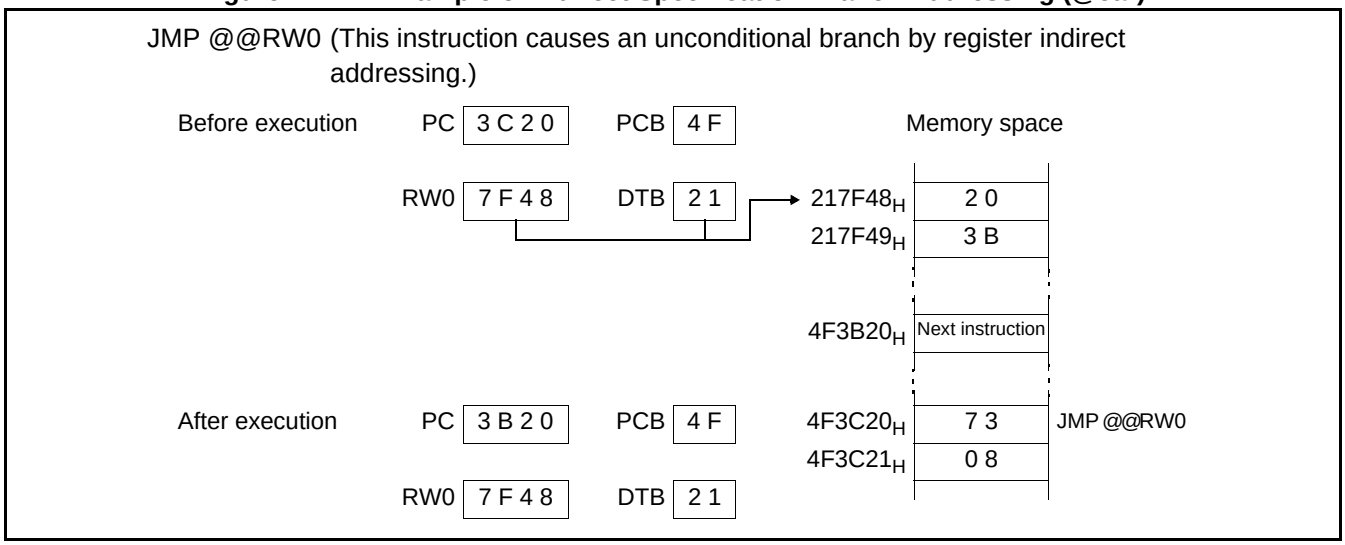
Figure B.4-11 Example of Accumulator Indirect Branch Addressing (@A)



- Indirect specification branch addressing (@ear)

The address of the branch destination is the word data at the address indicated by ear.

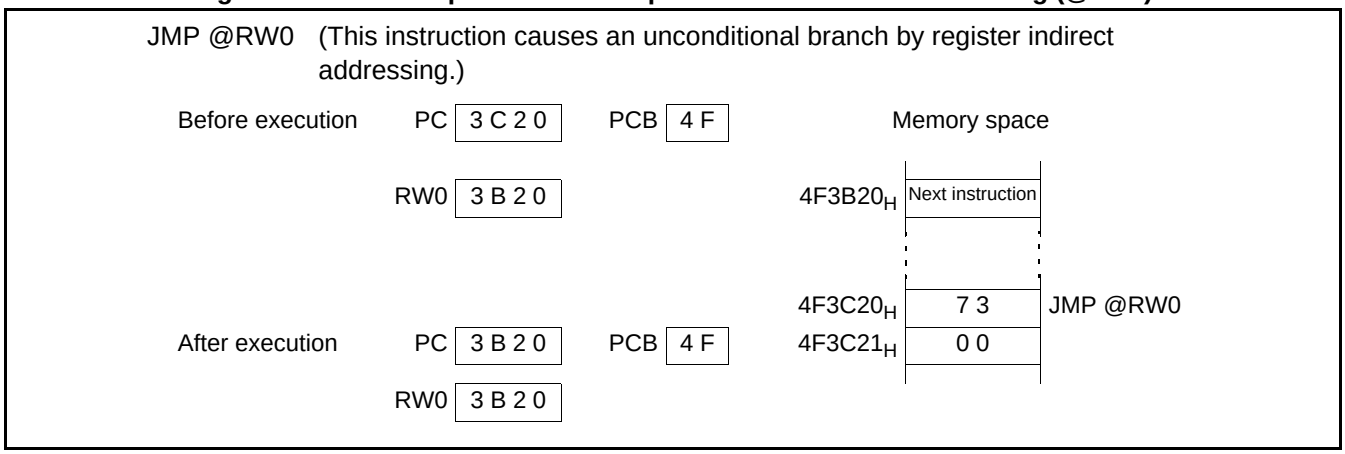
Figure B.4-12 Example of Indirect Specification Branch Addressing (@ear)



- Indirect specification branch addressing (@eam)

The address of the branch destination is the word data at the address indicated by eam.

Figure B.4-13 Example of Indirect Specification Branch Addressing (@eam)



B.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 B.5-1 lists execution cycle counts and Table B.5-2 and Table B.5-3 summarize correction value data.

Table B.5-1 Execution Cycle Counts in Each Addressing Mode

Code	Operand	(a) *	Register access count in each addressing mode
		Execution cycle count in each addressing mode	
00 | 07	Ri Rwi RLi	See the instruction list.	See the instruction list.
08 | 0B	@RWj	2	1
0C | 0F	@RWj+	4	2
10 | 17	@RWi+disp8	2	1
18 | 1B	@RWi+disp16	2	1
1C 1D 1E 1F	@RW0+RW7 @RW1+RW7 @PC+disp16 addr16	4 4 2 1	2 2 0 0

*: (a) is used for ~ (cycle count) and B (correction value) in "B.8 F^{2}MC-16LX Instruction List".

Table B.5-2 Cycle Count Correction Values for Counting Execution Cycles

Operand	(b) byte *		(c) word *		(d) long *	
	Cycle count	Access count	Cycle count	Access count	Cycle count	Access count
Internal register	+0	1	+0	1	+0	2
Internal memory Even address	+0	1	+0	1	+0	2
Internal memory Odd address	+0	1	+2	2	+4	4
External data bus 16-bit even address	+1	1	+1	1	+2	2
External data bus 16-bit odd address	+1	1	+4	2	+8	4
External data bus 8-bits	+1	1	+4	2	+8	4

*: (b), (c), and (d) are used for ~ (cycle count) and B (correction value) in "B.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 B.5-3 Cycle Count Correction Values for Counting Instruction Fetch Cycles

Instruction	Byte boundary	Word boundary
Internal memory	-	+2
External data bus 16-bits	-	+3
External data bus 8-bits	+3	-

Notes:

- When an external data bus is used, the cycle counts during which an instruction is made to wait by ready input or automatic ready must also be added.
- Actually, instruction execution is not delayed by every instruction fetch. Therefore, use the correction values to calculate the worst case.

B.6 Effective address field

Table B.6-1 shows the effective address field.

■ Effective Address Field

Table B.6-1 Effective Address Field

Code	Representation			Address format	Byte count of extended address part *
00	R0	RW0	RL0	Register direct: Individual parts correspond to the byte, word, and long word types in order from the left.	-
01	R1	RW1	(RL0)		
02	R2	RW2	RL1		
03	R3	RW3	(RL1)		
04	R4	RW4	RL2		
05	R5	RW5	(RL2)		
06	R6	RW6	RL3		
07	R7	RW7	(RL3)		
08	@RW0			Register indirect	0
09	@RW1				
0A	@RW2				
0B	@RW3				
0C	@RW0+			Register indirect with post increment	0
0D	@RW1+				
0E	@RW2+				
0F	@RW3+				
10	@RW0+disp8			Register indirect with 8-bit displacement	1
11	@RW1+disp8				
12	@RW2+disp8				
13	@RW3+disp8				
14	@RW4+disp8				
15	@RW5+disp8				
16	@RW6+disp8				
17	@RW7+disp8				
18	@RW0+disp16			Register indirect with 16-bit displacement	2
19	@RW1+disp16				
1A	@RW2+disp16				
1B	@RW3+disp16				
1C	@RW0+RW7			Register indirect with index	0
1D	@RW1+RW7			Register indirect with index	0
1E	@PC+disp16			PC indirect with 16-bit displacement	2
1F	addr16			Direct address	2

*1: Each byte count of the extended address part applies to + in the # (byte count) column in "B.8 F^{2}MC-16LX Instruction List".

B.7 How to Read the Instruction List

Table B.7-1 describes the items used in "B.8 F²MC-16LX Instruction List", and Table B.7-2 describes the symbols used in the same list.

■ Description of Instruction Presentation Items and Symbols

Table B.7-1 Description of Items in the Instruction List (Sheet 1 of 2)

Item	Description
Mnemonic	Uppercase, symbol: Represented as is in the assembler. Lowercase: Rewritten in the assembler. Number of following lowercase: Indicates bit length in the instruction.
#	Indicates the number of bytes.
~	Indicates the number of cycles. See Table B.2-1 for the alphabetical letters in items.
RG	Indicates the number of times a register access is performed during instruction execution. The number is used to calculate the correction value for CPU intermittent operation.
B	Indicates the correction value used to calculate the actual number of cycles during instruction execution. The actual number of cycles during instruction execution can be determined by adding the value in the ~ column to this value.
Operation	Indicates the instruction operation.
LH	Indicates the special operation for bit15 to bit08 of the accumulator. Z: Transfers 0. X: Transfers after sign extension. -: No transfer
AH	Indicates the special operation for the 16 high-order bits of the accumulator. *: Transfers from AL to AH. -: No transfer Z: Transfers 00 to AH. X: Transfers 00_H or FF_H to AH after AL sign extension.

Table B.7-1 Description of Items in the Instruction List (Sheet 2 of 2)

Item	Description
I	Each indicates the state of each flag: I (interrupt enable), S (stack), T (sticky bit), N (negative), Z (zero), V (overflow), C (carry). *: Changes upon instruction execution. -: No change S: Set upon instruction execution. R: Reset upon instruction execution.
S	
T	
N	
Z	
V	
C	
RMW	Indicates whether the instruction is a Read Modify Write instruction (reading data from memory by the I instruction and writing the result to memory). *: Read Modify Write instruction -: Not Read Modify Write instruction **Note:** Cannot be used for an address that has different meanings between read and write operations.

Table B.7-2 Explanation on Symbols in the Instruction List (Sheet 1 of 2)

Symbol	Explanation
A	The bit length used varies depending on the 32-bit accumulator instruction. Byte: Low-order 8 bits of byte AL Word: 16 bits of word AL Long word: 32 bits of AL and AH
AH	16 high-order bits of A
AL	16 low-order bits of A
SP	Stack pointer (USP or SSP)
PC	Program counter
PCB	program counter bank register
DTB	Data bank register
ADB	Additional data bank register
SSB	System stack bank register
USB	User stack bank register
SPB	Current stack bank register (SSB or USB)
DPR	Direct page register
brg1	DTB, ADB, SSB, USB, DPR, PCB, SPB

Table B.7-2 Explanation on Symbols in the Instruction List (Sheet 2 of 2)

Symbol	Explanation
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

B.8 F²MC-16LX Instruction List

Table B.8-1 to Table B.8-18 list the instructions used by the F²MC-16LX.

■ F²MC-16LX Instruction List

Table B.8-1 41 Transfer Instructions (Byte)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOV	A,dir	2	3	0	(b)	byte (A) ← (dir)	Z	*	-	-	-	*	*	-	-	-
MOV	A,addr16	3	4	0	(b)	byte (A) ← (addr16)	Z	*	-	-	-	*	*	-	-	-
MOV	A,Ri	1	2	1	0	byte (A) ← (Ri)	Z	*	-	-	-	*	*	-	-	-
MOV	A,ear	2	2	1	0	byte (A) ← (ear)	Z	*	-	-	-	*	*	-	-	-
MOV	A,eam	2+	3 + (a)	0	(b)	byte (A) ← (eam)	Z	*	-	-	-	*	*	-	-	-
MOV	A,io	2	3	0	(b)	byte (A) ← (io)	Z	*	-	-	-	*	*	-	-	-
MOV	A,#imm8	2	2	0	0	byte (A) ← imm8	Z	*	-	-	-	*	*	-	-	-
MOV	A,@A	2	3	0	(b)	byte (A) ← ((A))	Z	-	-	-	-	*	*	-	-	-
MOV	A,@RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	Z	*	-	-	-	*	*	-	-	-
MOVN	A,#imm4	1	1	0	0	byte (A) ← imm4	Z	*	-	-	-	R	*	-	-	-
MOVX	A,dir	2	3	0	(b)	byte (A) ← (dir)	X	*	-	-	-	*	*	-	-	-
MOVX	A,addr16	3	4	0	(b)	byte (A) ← (addr16)	X	*	-	-	-	*	*	-	-	-
MOVX	A,Ri	2	2	1	0	byte (A) ← (Ri)	X	*	-	-	-	*	*	-	-	-
MOVX	A,ear	2	2	1	0	byte (A) ← (ear)	X	*	-	-	-	*	*	-	-	-
MOVX	A,eam	2+	3 + (a)	0	(b)	byte (A) ← (eam)	X	*	-	-	-	*	*	-	-	-
MOVX	A,io	2	3	0	(b)	byte (A) ← (io)	X	*	-	-	-	*	*	-	-	-
MOVX	A,#imm8	2	2	0	0	byte (A) ← imm8	X	*	-	-	-	*	*	-	-	-
MOVX	A,@A	2	3	0	(b)	byte (A) ← ((A))	X	-	-	-	-	*	*	-	-	-
MOVX	A,@RWi+disp8	2	5	1	(b)	byte (A) ← ((RWi)+disp8)	X	*	-	-	-	*	*	-	-	-
MOVX	A,@RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	X	*	-	-	-	*	*	-	-	-
MOV	dir,A	2	3	0	(b)	byte (dir) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	addr16,A	3	4	0	(b)	byte (addr16) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	Ri,A	1	2	1	0	byte (Ri) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	ear,A	2	2	1	0	byte (ear) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	eam,A	2+	3 + (a)	0	(b)	byte (eam) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	io,A	2	3	0	(b)	byte (io) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	@RLi+disp8,A	3	10	2	(b)	byte ((RLi)+disp8) ← (A)	-	-	-	-	-	*	*	-	-	-
MOV	Ri,ear	2	3	2	0	byte (Ri) ← (ear)	-	-	-	-	-	*	*	-	-	-
MOV	Ri,eam	2+	4 + (a)	1	(b)	byte (Ri) ← (eam)	-	-	-	-	-	*	*	-	-	-
MOV	ear,Ri	2	4	2	0	byte (ear) ← (Ri)	-	-	-	-	-	*	*	-	-	-
MOV	eam,Ri	2+	5 + (a)	1	(b)	byte (eam) ← (Ri)	-	-	-	-	-	*	*	-	-	-
MOV	Ri,#imm8	2	2	1	0	byte (Ri) ← imm8	-	-	-	-	-	*	*	-	-	-
MOV	io,#imm8	3	5	0	(b)	byte (io) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV	dir,#imm8	3	5	0	(b)	byte (dir) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV	ear,#imm8	3	2	1	0	byte (ear) ← imm8	-	-	-	-	-	*	*	-	-	-
MOV	eam,#imm8	3+	4 + (a)	0	(b)	byte (eam) ← imm8	-	-	-	-	-	-	-	-	-	-
MOV	@AL,AH	2	3	0	(b)	byte ((A)) ← (AH)	-	-	-	-	-	*	*	-	-	-
XCH	A,ear	2	4	2	0	byte (A) ↔ (ear)	Z	-	-	-	-	-	-	-	-	-
XCH	A,eam	2+	5 + (a)	0	2 × (b)	byte (A) ↔ (eam)	Z	-	-	-	-	-	-	-	-	-
XCH	Ri,ear	2	7	4	0	byte (Ri) ↔ (ear)	-	-	-	-	-	-	-	-	-	-
XCH	Ri,eam	2+	9 + (a)	2	2 × (b)	byte (Ri) ↔ (eam)	-	-	-	-	-	-	-	-	-	-

Note:

See Table B.5-1 and Table B.5-2 for information on (a) and (b) in the table.

Table B.8-2 38 Transfer Instructions (Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVW	A,dir	2	3	0	(c)	word (A) ← (dir)	-	*	-	-	-	*	*	-	-	-
MOVW	A,addr16	3	4	0	(c)	word (A) ← (addr16)	-	*	-	-	-	*	*	-	-	-
MOVW	A,SP	1	1	0	0	word (A) ← (SP)	-	*	-	-	-	*	*	-	-	-
MOVW	A,RWi	1	2	1	0	word (A) ← (RWi)	-	*	-	-	-	*	*	-	-	-
MOVW	A,ear	2	2	1	0	word (A) ← (ear)	-	*	-	-	-	*	*	-	-	-
MOVW	A,eam	2+	3 + (a)	0	(c)	word (A) ← (eam)	-	*	-	-	-	*	*	-	-	-
MOVW	A,io	2	3	0	(c)	word (A) ← (io)	-	*	-	-	-	*	*	-	-	-
MOVW	A,@A	2	3	0	(c)	word (A) ← ((A))	-	-	-	-	-	*	*	-	-	-
MOVW	A,#imm16	3	2	0	0	word (A) ← imm16	-	*	-	-	-	*	*	-	-	-
MOVW	A,@RWi+disp8	2	5	1	(c)	word (A) ← ((RWi)+disp8)	-	*	-	-	-	*	*	-	-	-
MOVW	A,@RLi+disp8	3	10	2	(c)	word (A) ← ((RLi)+disp8)	-	*	-	-	-	*	*	-	-	-
MOVW	dir,A	2	3	0	(c)	word (dir) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW	addr16,A	3	4	0	(c)	word (addr16) ← (A)	-	-	-	-	-	*	*	-	-	-
MOVW	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 B.5-1 and Table B.5-2 for information on (a), (c), and (d) in the table.

Table B.8-3 42 Addition/Subtraction Instructions (Byte, Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ADD	A,#imm8	2	2	0	0	byte (A) ← (A) + imm8	Z	-	-	-	-	*	*	*	*	-
ADD	A,dir	2	5	0	(b)	byte (A) ← (A) + (dir)	Z	-	-	-	-	*	*	*	*	-
ADD	A,ear	2	3	1	0	byte (A) ← (A) + (ear)	Z	-	-	-	-	*	*	*	*	-
ADD	A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) + (eam)	Z	-	-	-	-	*	*	*	*	-
ADD	ear,A	2	3	2	0	byte (ear) ← (ear) + (A)	-	-	-	-	-	*	*	*	*	-
ADD	eam,A	2+	5 + (a)	0	2 × (b)	byte (eam) ← (eam) + (A)	Z	-	-	-	-	*	*	*	*	*
ADDC	A	1	2	0	0	byte (A) ← (AH) + (AL) + (C)	Z	-	-	-	-	*	*	*	*	-
ADDC	A,ear	2	3	1	0	byte (A) ← (A) + (ear)+ (C)	Z	-	-	-	-	*	*	*	*	-
ADDC	A,eam	2+	4 + (a)	0	(b)	byte (A) ← (A) + (eam)+ (C)	Z	-	-	-	-	*	*	*	*	-
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 B.5-1 and Table B.5-2 for information on (a) to (d) in the table.

Table B.8-4 12 Increment/decrement Instructions (Byte, Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
INC	ear	2	3	2	0	byte (ear) ← (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 B.5-1 and Table B.5-2 for information on (a) to (d) in the table.

Table B.8-5 11 Compare Instructions (Byte, Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CMP	A	1	1	0	0	byte (AH) - (AL)	-	-	-	-	-	*	*	*	*	-
CMP	A,ear	2	2	1	0	byte (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMP	A,eam	2+	3+(a)	0	(b)	byte (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMP	A,#imm8	2	2	0	0	byte (A) - imm8	-	-	-	-	-	*	*	*	*	-
CMPW	A	1	1	0	0	word (AH) - (AL)	-	-	-	-	-	*	*	*	*	-
CMPW	A,ear	2	2	1	0	word (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMPW	A,eam	2+	3+(a)	0	(c)	word (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMPW	A,#imm16	3	2	0	0	word (A) - imm16	-	-	-	-	-	*	*	*	*	-
CMPL	A,ear	2	6	2	0	long (A) - (ear)	-	-	-	-	-	*	*	*	*	-
CMPL	A,eam	2+	7+(a)	0	(d)	long (A) - (eam)	-	-	-	-	-	*	*	*	*	-
CMPL	A,#imm32	5	3	0	0	long (A) - imm32	-	-	-	-	-	*	*	*	*	-

Note:

See Table B.5-1 and Table B.5-2 for information on (a) to (d) in the table.

Table B.8-6 11 Unsigned Multiplication/Division Instructions (Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIVU	A	1	*1	0	0	word (AH) / byte (AL) quotient → byte (AL) remainder → byte (AH)	-	-	-	-	-	-	-	*	*	-
DIVU	A,ear	2	*2	1	0	word (A) / byte (ear) quotient → byte (A) remainder → byte (ear)	-	-	-	-	-	-	-	*	*	-
DIVU	A,eam	2+	*3	0	*6	word (A) / byte (eam) quotient → byte (A) remainder → byte (eam)	-	-	-	-	-	-	-	*	*	-
DIVUW	A,ear	2	*4	1	0	long (A) / word (ear) quotient → word (A) remainder → word (ear)	-	-	-	-	-	-	-	*	*	-
DIVUW	A,eam	2+	*5	0	*7	long (A) / word (eam) quotient → word (A) remainder → word (eam)	-	-	-	-	-	-	-	*	*	-
MULU	A	1	*8	0	0	byte (AH) * byte (AL) → word (A)	-	-	-	-	-	-	-	-	-	-
MULU	A,ear	2	*9	1	0	byte (A) * byte (ear) → word (A)	-	-	-	-	-	-	-	-	-	-
MULU	A,eam	2+	*10	0	(b)	byte (A) * byte (eam) → word (A)	-	-	-	-	-	-	-	-	-	-
MULUW	A	1	*11	0	0	word (AH) * word (AL) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULUW	A,ear	2	*12	1	0	word (A) * word (ear) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULUW	A,eam	2+	*13	0	(c)	word (A) * word (eam) → Long (A)	-	-	-	-	-	-	-	-	-	-

*1: 3: Division by 0 7: Overflow 15: Normal
*2: 4: Division by 0 8: Overflow 16: Normal
*3: 6+(a): Division by 0 9+(a): Overflow 19+(a): Normal
*4: 4: Division by 0 7: Overflow 22: Normal
*5: 6+(a): Division by 0 8+(a): Overflow 26+(a): Normal
*6: (b): Division by 0 or overflow 2 × (b): Normal
*7: (c): Division by 0 or overflow 2 × (c): Normal
*8: 3: Byte (AH) is 0. 7: Byte (AH) is not 0.
*9: 4: Byte (ear) is 0. 8: Byte (ear) is not 0.
*10: 5+(a): Byte (eam) is 0, 9+(a): Byte (eam) is not 0.
*11: 3: Word (AH) is 0. 11: Word (AH) is not 0.
*12: 4: Word (ear) is 0. 12: Word (ear) is not 0.
*13: 5+(a): Word (eam) is 0. 13+(a): Word (eam) is not 0.

Note:

See Table B.5-1 and Table B.5-2 for information on (a) to (c) in the table.

Table B.8-7 11 Signed Multiplication/Division Instructions (Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIV	A	2	*1	0	0	word (AH) / byte (AL) quotient → byte (AL) remainder → byte (AH)	Z	-	-	-	-	-	-	*	*	-
DIV	A,ear	2	*2	1	0	word (A) / byte (ear) quotient → byte (A) remainder → byte (ear)	Z	-	-	-	-	-	-	*	*	-
DIV	A,eam	2+	*3	0	*6	word (A) / byte (eam) quotient → byte (A) remainder → byte (eam)	Z	-	-	-	-	-	-	*	*	-
DIVW	A,ear	2	*4	1	0	long (A) / word (ear) quotient → word (A) remainder → word (ear)	-	-	-	-	-	-	-	*	*	-
DIVW	A,eam	2+	*5	0	*7	long (A) / word (eam) quotient → word (A) remainder → word (eam)	-	-	-	-	-	-	-	*	*	-
MUL	A	2	*8	0	0	byte (AH) * byte (AL) → word (A)	-	-	-	-	-	-	-	-	-	-
MUL	A,ear	2	*9	1	0	byte (A) * byte (ear) → word (A)	-	-	-	-	-	-	-	-	-	-
MUL	A,eam	2+	*10	0	(b)	byte (A) * byte (eam) → word (A)	-	-	-	-	-	-	-	-	-	-
MULW	A	2	*11	0	0	word (AH) * word (AL) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULW	A,ear	2	*12	1	0	word (A) * word (ear) → Long (A)	-	-	-	-	-	-	-	-	-	-
MULW	A,eam	2+	*13	0	(c)	word (A) * word (eam) → Long (A)	-	-	-	-	-	-	-	-	-	-

*1: 3: Division by 0, 8 or 18: Overflow, 18: Normal
*2: 4: Division by 0, 11 or 22: Overflow, 23: Normal
*3: 5+(a): Division by 0, 12+(a) or 23+(a): Overflow, 24+(a): Normal
*4: When dividend is positive; 4: Division by 0, 12 or 30: Overflow, 31: Normal
When dividend is negative; 4: Division by 0, 12 or 31: Overflow, 32: Normal
*5: When dividend is positive; 5+(a): Division by 0, 12+(a) or 31+(a): Overflow, 32+(a): Normal
When dividend is negative; 5+(a): Division by 0, 12+(a) or 32+(a): Overflow, 33+(a): Normal
*6: (b): Division by 0 or overflow, 2 × (b): Normal
*7: (c): Division by 0 or overflow, 2 × (c): Normal
*8: 3: Byte (AH) is 0, 12: result is positive, 13: result is negative
*9: 4: Byte (ear) is 0, 13: result is positive, 14: result is negative
*10: 5+(a): Byte (eam) is 0, 14+(a): result is positive, 15+(a): result is negative
*11: 3: Word (AH) is 0, 16: result is positive, 19: result is negative
*12: 4: Word (ear) is 0, 17: result is positive, 20: result is negative
*13: 5+(a): Word (eam) is 0, 18+(a): result is positive, 21+(a): result is negative

---

Notes:

- The execution cycle count found when an overflow occurs in a DIV or DIVW instruction may be a pre-operation count or a post-operation count depending on the detection timing.
- When an overflow occurs with DIV or DIVW instruction, the contents of the AL are destroyed.
- See Table B.5-1 and Table B.5-2 for information on (a) to (c) in the table.

Table B.8-8 39 Logic 1 Instructions (Byte, Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
AND	A,#imm8	2	2	0	0	byte (A) ← (A) and imm8	-	-	-	-	-	*	*	R	-	-
AND	A,ear	2	3	1	0	byte (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
AND	A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
AND	ear,A	2	3	2	0	byte (ear) ← (ear) and (A)	-	-	-	-	-	*	*	R	-	-
AND	eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) and (A)	-	-	-	-	-	*	*	R	-	*
OR	A,#imm8	2	2	0	0	byte (A) ← (A) or imm8	-	-	-	-	-	*	*	R	-	-
OR	A,ear	2	3	1	0	byte (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
OR	A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
OR	ear,A	2	3	2	0	byte (ear) ← (ear) or (A)	-	-	-	-	-	*	*	R	-	-
OR	eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) or (A)	-	-	-	-	-	*	*	R	-	*
XOR	A,#imm8	2	2	0	0	byte (A) ← (A) xor imm8	-	-	-	-	-	*	*	R	-	-
XOR	A,ear	2	3	1	0	byte (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XOR	A,eam	2+	4+(a)	0	(b)	byte (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-
XOR	ear,A	2	3	2	0	byte (ear) ← (ear) xor (A)	-	-	-	-	-	*	*	R	-	-
XOR	eam,A	2+	5+(a)	0	2 × (b)	byte (eam) ← (eam) xor (A)	-	-	-	-	-	*	*	R	-	*
NOT	A	1	2	0	0	byte (A) ← not (A)	-	-	-	-	-	*	*	R	-	-
NOT	ear	2	3	2	0	byte (ear) ← not (ear)	-	-	-	-	-	*	*	R	-	-
NOT	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← not (eam)	-	-	-	-	-	*	*	R	-	*
ANDW	A	1	2	0	0	word (A) ← (AH) and (A)	-	-	-	-	-	*	*	R	-	-
ANDW	A,#imm16	3	2	0	0	word (A) ← (A) and imm16	-	-	-	-	-	*	*	R	-	-
ANDW	A,ear	2	3	1	0	word (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
ANDW	A,eam	2+	4+(a)	0	(c)	word (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
ANDW	ear,A	2	3	2	0	word (ear) ← (ear) and (A)	-	-	-	-	-	*	*	R	-	-
ANDW	eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) and (A)	-	-	-	-	-	*	*	R	-	*
ORW	A	1	2	0	0	word (A) ← (AH) or (A)	-	-	-	-	-	*	*	R	-	-
ORW	A,#imm16	3	2	0	0	word (A) ← (A) or imm16	-	-	-	-	-	*	*	R	-	-
ORW	A,ear	2	3	1	0	word (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
ORW	A,eam	2+	4+(a)	0	(c)	word (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
ORW	ear,A	2	3	2	0	word (ear) ← (ear) or (A)	-	-	-	-	-	*	*	R	-	-
ORW	eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) or (A)	-	-	-	-	-	*	*	R	-	*
XORW	A	1	2	0	0	word (A) ← (AH) xor (A)	-	-	-	-	-	*	*	R	-	-
XORW	A,#imm16	3	2	0	0	word (A) ← (A) xor imm16	-	-	-	-	-	*	*	R	-	-
XORW	A,ear	2	3	1	0	word (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XORW	A,eam	2+	4+(a)	0	(c)	word (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-
XORW	ear,A	2	3	2	0	word (ear) ← (ear) xor (A)	-	-	-	-	-	*	*	R	-	-
XORW	eam,A	2+	5+(a)	0	2 × (c)	word (eam) ← (eam) xor (A)	-	-	-	-	-	*	*	R	-	*
NOTW	A	1	2	0	0	word (A) ← not (A)	-	-	-	-	-	*	*	R	-	-
NOTW	ear	2	3	2	0	word (ear) ← not (ear)	-	-	-	-	-	*	*	R	-	-
NOTW	eam	2+	5+(a)	0	2 × (c)	word (eam) ← not (eam)	-	-	-	-	-	*	*	R	-	*

Note:

See Table B.5-1 and Table B.5-2 for information on (a) to (c) in the table.

Table B.8-9 6 Logic 2 Instructions (Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ANDL	A,ear	2	6	2	0	long (A) ← (A) and (ear)	-	-	-	-	-	*	*	R	-	-
ANDL	A,eam	2+	7+(a)	0	(d)	long (A) ← (A) and (eam)	-	-	-	-	-	*	*	R	-	-
ORL	A,ear	2	6	2	0	long (A) ← (A) or (ear)	-	-	-	-	-	*	*	R	-	-
ORL	A,eam	2+	7+(a)	0	(d)	long (A) ← (A) or (eam)	-	-	-	-	-	*	*	R	-	-
XORL	A,ear	2	6	2	0	long (A) ← (A) xor (ear)	-	-	-	-	-	*	*	R	-	-
XORL	A,eam	2+	7+(a)	0	(d)	long (A) ← (A) xor (eam)	-	-	-	-	-	*	*	R	-	-

Note:

See Table B.5-1 and Table B.5-2 for information on (a) and (d) in the table.

Table B.8-10 6 Sign Inversion Instructions (Byte, Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NEG	A	1	2	0	0	byte (A) ← 0 - (A)	X	-	-	-	-	*	*	*	*	-
NEG	ear	2	3	2	0	byte (ear) ← 0 - (ear)	-	-	-	-	-	*	*	*	*	-
NEG	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← 0 - (eam)	-	-	-	-	-	*	*	*	*	*
NEGW	A	1	2	0	0	word (A) ← 0 - (A)	-	-	-	-	-	*	*	*	*	-
NEGW	ear	2	3	2	0	word (ear) ← 0 - (ear)	-	-	-	-	-	*	*	*	*	-
NEGW	eam	2+	5+(a)	0	2 × (c)	word (eam) ← 0 - (eam)	-	-	-	-	-	*	*	*	*	*

Note:

See Table B.5-1 and Table B.5-2 for information on (a) to (c) in the table.

Table B.8-11 1 Normalization Instruction (Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NRML	A,R0	2	*1	1	0	long (A) ← Shift left to the position where '1' is set for the first time. byte (R0) ← Shift count at that time	-	-	-	-	-	-	*	-	-	-

*1: 4 when all accumulators have a value of 0; otherwise, 6+(R0)

Table B.8-12 18 Shift Instructions (Byte, Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
RORC	A	2	2	0	0	byte (A) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
ROLC	A	2	2	0	0	byte (A) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
RORC	ear	2	3	2	0	byte (ear) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	-
RORC	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← Right rotation with carry	-	-	-	-	-	*	*	-	*	*
ROLC	ear	2	3	2	0	byte (ear) ← Left rotation with carry	-	-	-	-	-	*	*	-	*	-
ROLC	eam	2+	5+(a)	0	2 × (b)	byte (eam) ← Left rotation with carry	-	-	-	-	-	*	*	-	*	*
ASR	A,R0	2	*1	1	0	byte (A) ← Arithmetic right shift (A, 1 bit)	-	-	-	-	*	*	*	-	*	-
LSR	A,R0	2	*1	1	0	byte (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSL	A,R0	2	*1	1	0	byte (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-
ASRW	A	1	2	0	0	word (A) ← Arithmetic right shift (A, 1 bit)	-	-	-	-	*	*	*	-	*	-
LSRW	A/SHRW A	1	2	0	0	word (A) ← Logical right shift (A, 1 bit)	-	-	-	-	*	R	*	-	*	-
LSLW	A/SHLW A	1	2	0	0	word (A) ← Logical left shift (A, 1 bit)	-	-	-	-	-	*	*	-	*	-
ASRW	A,R0	2	*1	1	0	word (A) ← Arithmetic right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSRW	A,R0	2	*1	1	0	word (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSLW	A,R0	2	*1	1	0	word (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-
ASRL	A,R0	2	*2	1	0	long (A) ← Arithmetic right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSRL	A,R0	2	*2	1	0	long (A) ← Logical right barrel shift (A, R0)	-	-	-	-	*	*	*	-	*	-
LSLL	A,R0	2	*2	1	0	long (A) ← Logical left barrel shift (A, R0)	-	-	-	-	-	*	*	-	*	-

*1: 6 when R0 is 0; otherwise, 5 + (R0)
*2: 6 when R0 is 0; otherwise, 6 + (R0)

Note:

See Table B.5-1 and Table B.5-2 for information on (a) and (b) in the table.

Table B.8-13 31 Branch 1 Instructions

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
BZ/BEQ	rel	2	*1	0	0	Branch on (Z) = 1	-	-	-	-	-	-	-	-	-	-
BNZ/ 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 B.5-1 and Table B.5-2 for information on (a) to (d) in the table.

Table B.8-14 19 Branch 2 Instructions

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CBNE	A,#imm8,rel	3	*1	0	0	Branch on byte (A) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CWBNE	A,#imm16,rel	4	*1	0	0	Branch on word (A) not equal to imm16	-	-	-	-	-	*	*	*	*	-
CBNE	ear,#imm8,rel	4	*2	1	0	Branch on byte (ear) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CBNE	eam,#imm8,rel *9	4+	*3	0	(b)	Branch on byte (eam) not equal to imm8	-	-	-	-	-	*	*	*	*	-
CWBNE	ear,#imm16,rel	5	*4	1	0	Branch on word (ear) not equal to imm16	-	-	-	-	-	*	*	*	*	-
CWBNE	eam,#imm16,rel*9	5+	*3	0	(c)	Branch on word (eam) not equal to imm16	-	-	-	-	-	*	*	*	*	-
DBNZ	ear,rel	3	*5	2	0	byte (ear) ← (ear) - 1, Branch on (ear) not equal to 0	-	-	-	-	-	*	*	*	-	-
DBNZ	eam,rel	3+	*6	2	2 × (b)	byte (eam) ← (eam) - 1, Branch on (eam) not equal to 0	-	-	-	-	-	*	*	*	-	*
DWBNZ	ear,rel	3	*5	2	0	word (ear) ← (ear) - 1, Branch on (ear) not equal to 0	-	-	-	-	-	*	*	*	-	-
DWBNZ	eam,rel	3+	*6	2	2 × (c)	word (eam) ← (eam) - 1, Branch on (eam) not equal to 0	-	-	-	-	-	*	*	*	-	*
INT	#vct8	2	20	0	8 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INT	addr16	3	16	0	6 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INTP	addr24	4	17	0	6 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
INT9		1	20	0	8 × (c)	Software interrupt	-	-	R	S	-	-	-	-	-	-
RETI		1	*8	0	*7	Return from interrupt	-	-	*	*	*	*	*	*	*	-
LINK	#imm8	2	6	0	(c)	Saves the old frame pointer in the stack upon entering the function, then sets the new frame pointer and reserves the local pointer area.	-	-	-	-	-	-	-	-	-	-
UNLINK		1	5	0	(c)	Recovers the old frame pointer from the stack upon exiting the function.	-	-	-	-	-	-	-	-	-	-
RET	*10	1	4	0	(c)	Return from subroutine	-	-	-	-	-	-	-	-	-	-
RETP	*11	1	6	0	(d)	Return from subroutine	-	-	-	-	-	-	-	-	-	-

*1: 5 when a branch is made; otherwise, 4
*2: 13 when a branch is made; otherwise, 12
*3: 7+(a) when a branch is made; otherwise, 6+(a)
*4: 8 when a branch is made; otherwise, 7
*5: 7 when a branch is made; otherwise, 6
*6: 8+(a) when a branch is made; otherwise, 7+(a)
*7: 3 × (b) + 2 × (c) when jumping to the next interruption request; 6 × (c) when returning from the current interruption
*8: 15 when jumping to the next interruption request; 17 when returning from the current interruption
*9: Do not use RWj+ addressing mode with a CBNE or CWBNE instruction.
*10: Return from stack (word)
*11: Return from stack (long word)

Note:

See Table B.5-1 and Table B.5-2 for information on (a) to (d) in the table.

Table B.8-15 28 Other Control Instructions (Byte, Word, Long Word)

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
PUSHW	A	1	4	0	(c)	word (SP) ← (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 B.5-1 and Table B.5-2 for information on (a) and (c) in the table.

Table B.8-16 21 Bit Operand Instructions

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVB	A,dir:bp	3	5	0	(b)	byte (A) ← (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 B.5-1 and Table B.5-2 for information on (b) in the table.

Table B.8-17 6 Accumulator Operation Instructions (Byte, Word)

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
SWAP	1	3	0	0	byte (A)0-7 ↔ (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 B.8-18 10 String Instructions

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVS / MOVSI	2	*2	*5	*3	byte transfer @AH+ ← @AL+, counter = RW0	-	-	-	-	-	-	-	-	-	-
MOVSD	2	*2	*5	*3	byte transfer @AH- ← @AL-, counter = RW0	-	-	-	-	-	-	-	-	-	-
SCEQ / SCEQI	2	*1	*8	*4	byte search @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
SCEQD	2	*1	*8	*4	byte search @AH- ← AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
FILS / FILSI	2	6m+6	*8	*3	byte fill @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	-	-	-
MOVSW / MOVSWI	2	*2	*5	*6	word transfer @AH+ ← @AL+, counter = RW0	-	-	-	-	-	-	-	-	-	-
MOVSWD	2	*2	*5	*6	word transfer @AH- ← @AL-, counter = RW0	-	-	-	-	-	-	-	-	-	-
SCWEQ / SCWEQI	2	*1	*8	*7	word search @AH+ - AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
SCWEQD	2	*1	*8	*7	word search @AH- - AL, counter = RW0	-	-	-	-	-	*	*	*	*	-
FILSW / FILSWI	2	6m+6	*8	*6	word fill @AH+ ← AL, counter = RW0	-	-	-	-	-	*	*	-	-	-

*1: 5 when RW0 is 0, 4 + 7 × (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

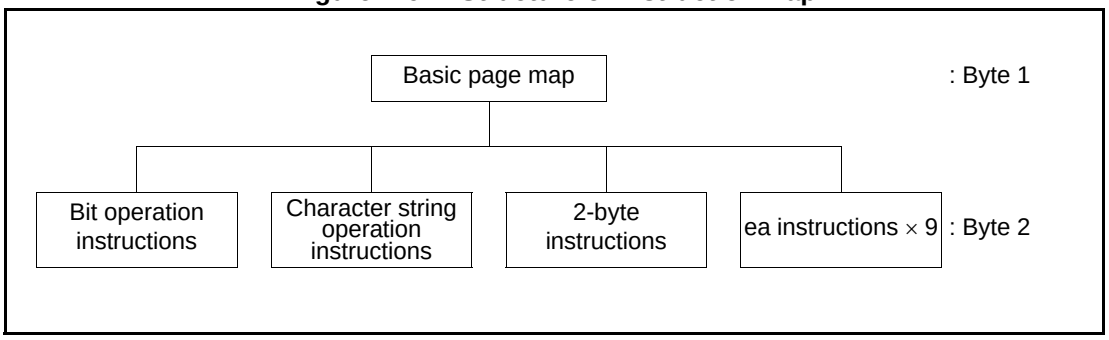
See Table B.5-1 and Table B.5-2 for information on (b) and (c) in the table.

B.9 Instruction Map

Each F^{2}MC-16LX instruction code consists of 1 or 2 bytes. Therefore, the instruction map consists of multiple pages. Table B.9-2 to Table B.9-21 summarize the F^{2}MC-16LX instruction map.

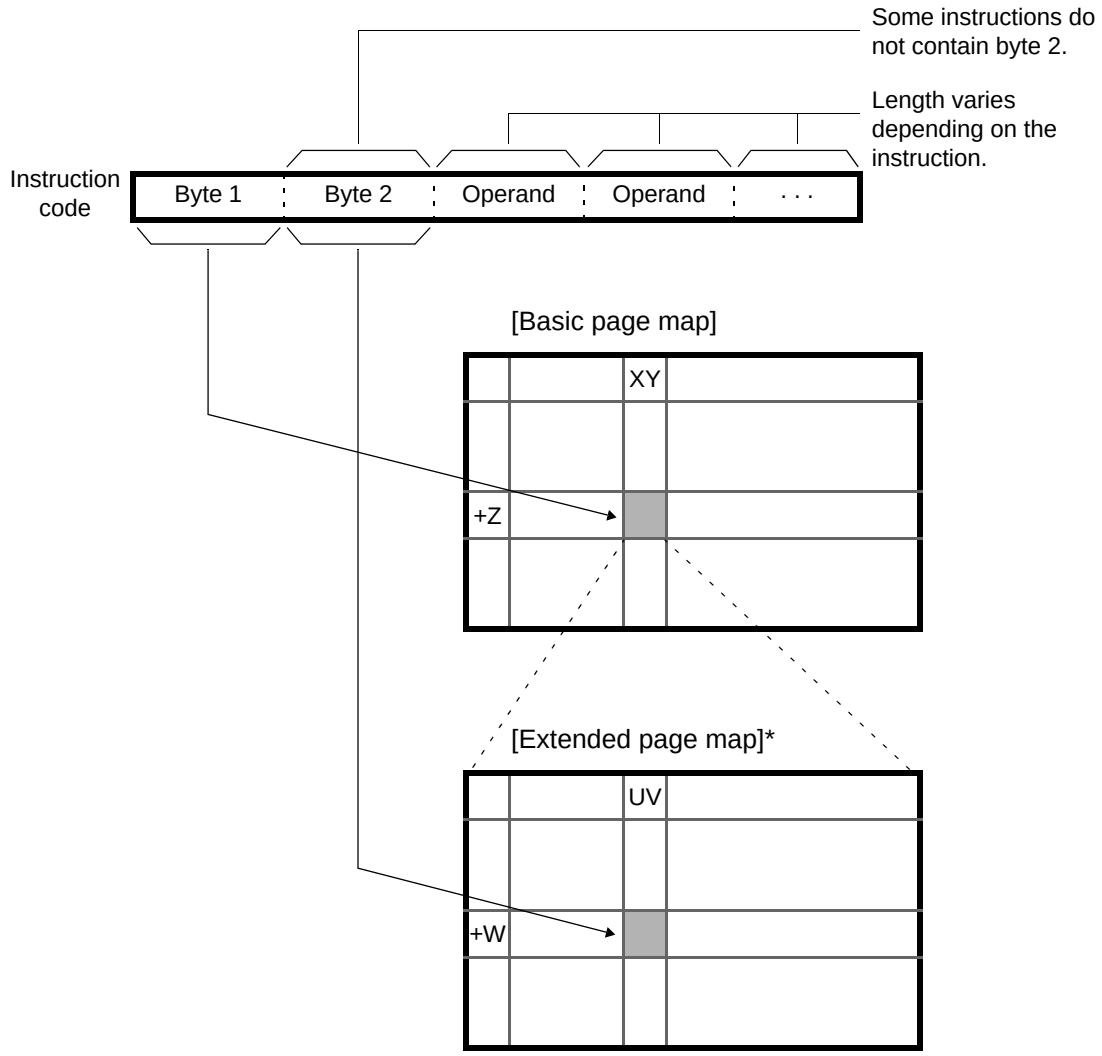
■ Structure of Instruction Map

Figure B.9-1 Structure of Instruction Map



An instruction such as the NOP instruction that ends in one byte is completed within the basic page. An instruction such as the MOVS instruction that requires two bytes recognizes the existence of byte 2 when it references byte 1, and can check the following one byte by referencing the map for byte 2. Figure B.9-2 shows the correspondence between an actual instruction code and instruction map.

Figure B.9-2 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 B.9-1.

Table B.9-1 Example of an Instruction Code (Sheet 1 of 2)

Instruction	Byte 1 (from basic page map)	Byte 2 (from extended page map)
NOP	00 +0=00	-
AND A, #8	30 +4=34	-

Table B.9-1 Example of an Instruction Code (Sheet 2 of 2)

Instruction	Byte 1 (from basic page map)	Byte 2 (from extended page map)
MOV A, ADB	60 +F=6F	00 +0=00
CBNE @RW2+d8, #8, rel	70 +0=70	F0 +2=F2

Table B.9-2 Basic Page Map

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0	NOP	CMR	ADD A, dir	ADD A, #8	MOV A, dir	MOV A, io	BRA rel	ea instruction 1	MOV A, Ri	MOV Ri, A	MOV Ri, #8	MOVX A, Ri	MOVX A, @RWi+d8	MOVN A, #4	CALL #vct4	BZ/BEQ rel
+1	INT9	NCC	SUB A, dir	SUB A, #8	MOV dir, A	MOV io, A	JMP @A	ea instruction 2								BNZ/BNE rel
+2	ADDDC A	SUBDC A	ADDC A	SUBC A	MOV A, #8	MOV A, addr16	JMP addr16	ea instruction 3								BC/BLO rel
+3	NEG A	JCTX @A	CMP A	CMP A, #8	MOVX A, #8	MOV addr16, A	JMPP addr24	ea instruction 4								BNC/BHS rel
+4	PCB	EXT	AND CCR, #8	AND A, #8	MOV dir, #8	MOV io, #8	CALL addr16	ea instruction 5								BN rel
+5	DTB	ZEXT	OR CCR, #8	OR A, #8	MOVX A, dir	MOVX A, io	CALLP addr24	ea instruction 6								BP rel
+6	ADB	SWAP	DIVU A	XOR A, #8	MOVW A, SP	MOVW io, #16	RETP	ea instruction 7								BV rel
+7	SPB	ADDSP #8	MULU A	NOT A	MOVW SP, A	MOVX A, addr16	RET	ea instruction 8								BNV rel
+8	LINK #imm8	ADDL A, #32	ADDW A	ADDW A, #16	MOVW A, dir	MOVW A, io	INT #vct8	ea instruction 9	MOVW A, RWi	MOVW RWi, A	MOVW RWi, #16	MOVW A, @RWi+d8	MOVW @RWi+d8, A			BT rel
+9	UNLINK	SUBL A, #32	SUBW A	SUBW A, #16	MOVW dir, A	MOVW io, A	INT addr16	MOVEA RWi, ea								BNT rel
+A	MOV RP, #8	MOV ILM, #8	CBNE A, #8, rel	CWBNE A, #16, rel	MOVW A, #16	MOVW A,addr16	INTP addr24	MOV Ri, ea								BLT rel
+B	NEGW A	CMPL A, #32	CMPW A	CMPW A, #16	MOVL A, #32	MOVW addr16, A	RETI	MOVW RWi, ea								BGE rel
+C	LSLW A	EXTW	ANDW A	ANDW A, #16	PUSHW A	POPW A	Bit operation instruction	MOV ea, Ri								BLE rel
+D		ZEXTW	ORW A	ORW A, #16	PUSHW AH	POPW AH	Character string operation instruction	MOVW ea, RWi								BGT rel
+E	ASRW A	SWAPW	XORW A	XORW A, #16	PUSHW PS	POPW PS	2-byte instruction	XCH Ri, ea								BLS rel
+F	LSRW A	ADDSP #16	MULUW A	NOTW A	PUSHW rlst	POPW rlst		XCHW RWi, ea								BHI rel

Table B.9-3 Bit Operation Instruction Map (First Byte = $6C_H$)

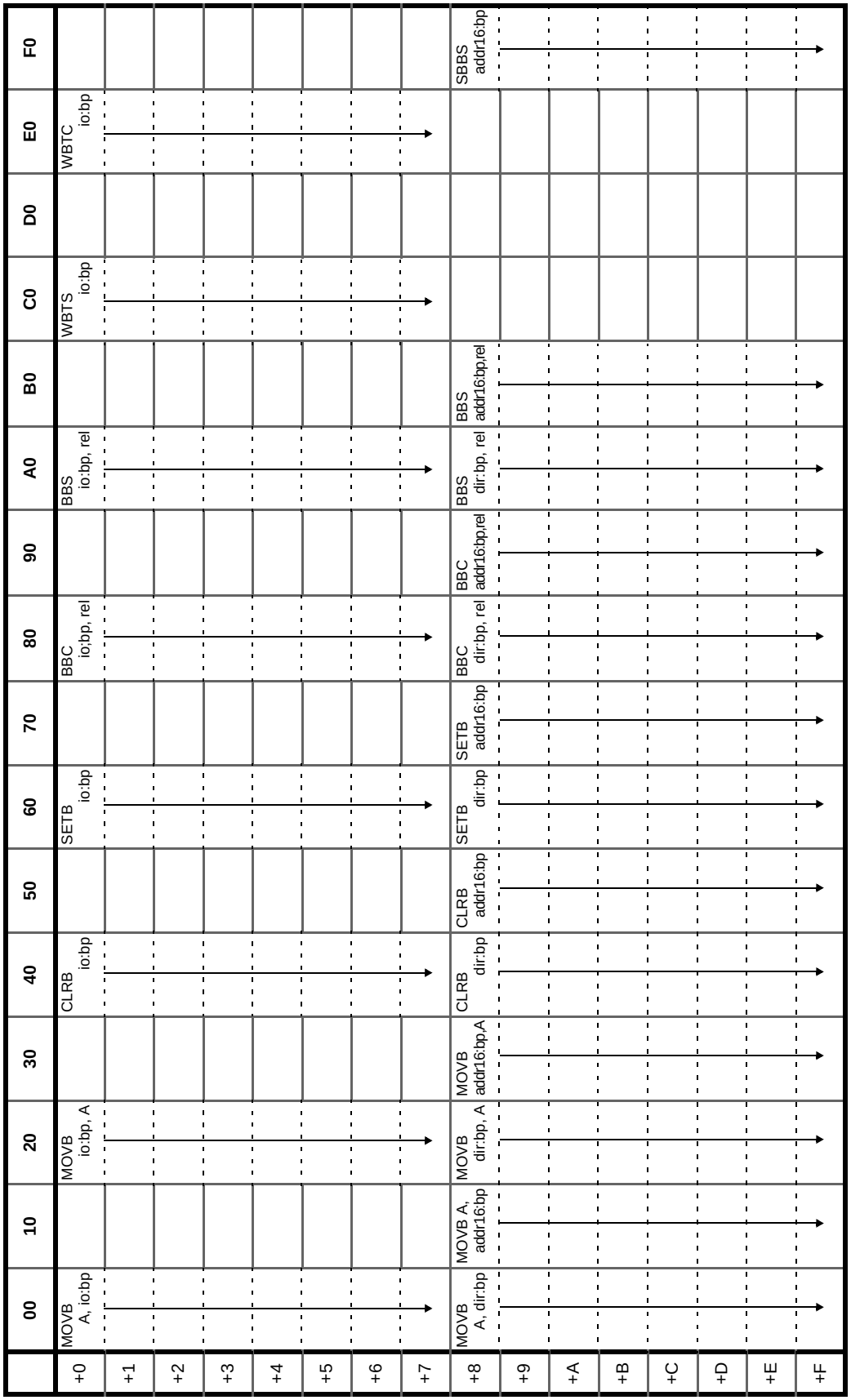


Table B.9-4 Character String Operation Instruction Map (First Byte = $6E_H$)

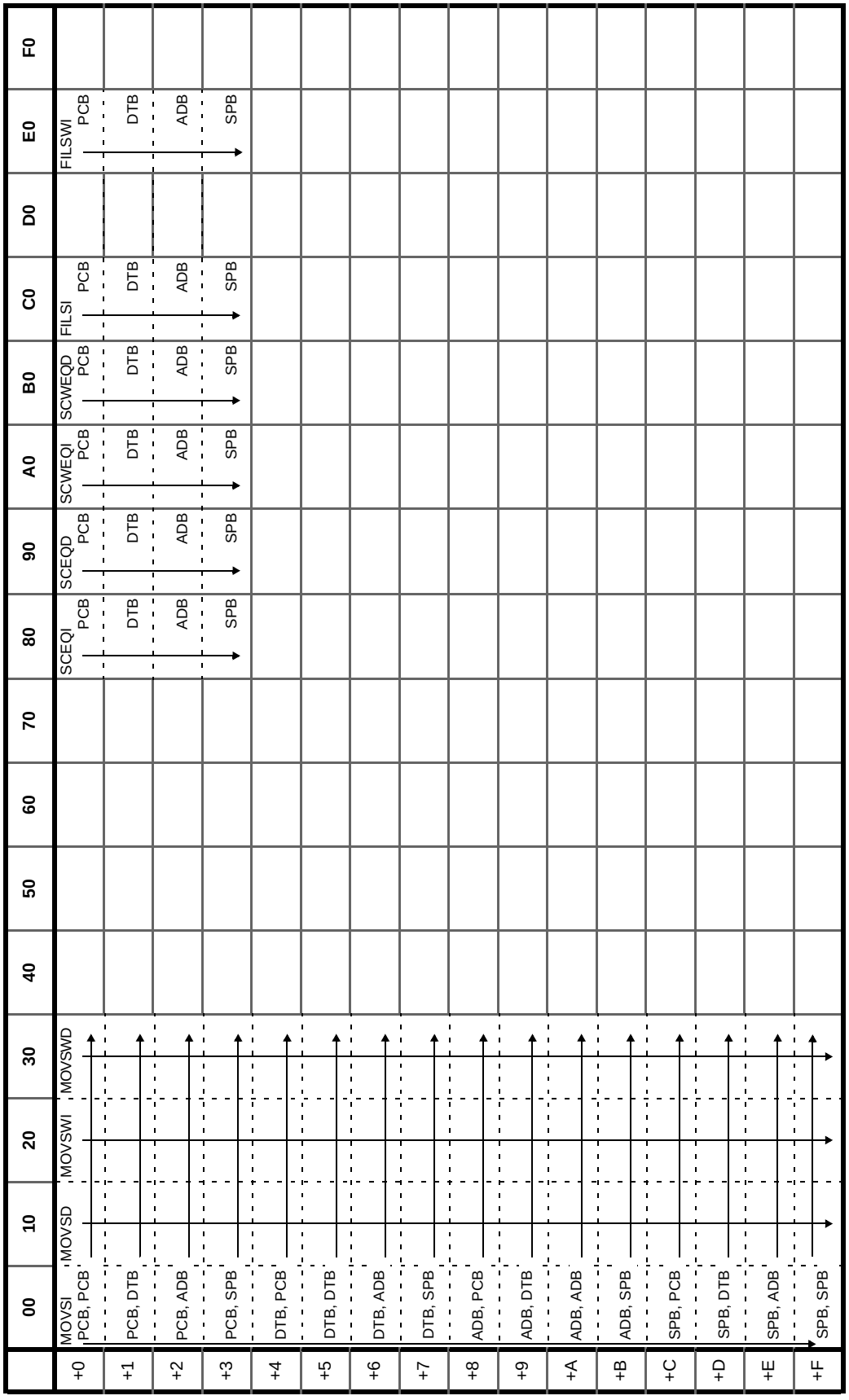


Table B.9-5 2-byte Instruction Map (First Byte = $6F_H$)

	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 B.9-6 ea Instruction 1 (First Byte = 70_H)

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
					CWBNE ↓, RW0, #16, rel	CWBNE ↓, @RW0+d8 #16, rel									CBNE ↓, R0, #8, rel	CBNE ↓, @RW0+d8, #8, rel
+0	ADDL A, RL0	ADDL A, @RW0+d8	SUBL A, RL0	SUBL A, @RW0+d8			CMPL A, RL0	CMPL A, @RW0+d8	ANDL A, RL0	ANDL A, @RW0+d8	ORL A, RL0	ORL A, @RW0+d8	XORL A, RL0	XORL A, @RW0+d8		
+1	ADDL A, RL0	ADDL A, @RW1+d8	SUBL A, RL0	SUBL A, @RW1+d8	RW1, #16, rel	@RW1+d8 #16, rel	CMPL A, RL0	CMPL A, @RW1+d8	ANDL A, RL0	ANDL A, @RW1+d8	ORL A, RL0	ORL A, @RW1+d8	XORL A, RL0	XORL A, @RW1+d8	R1, #8, rel	@RW1+d8, #8, rel
+2	ADDL A, RL1	ADDL A, @RW2+d8	SUBL A, RL1	SUBL A, @RW2+d8	RW2, #16, rel	@RW2+d8 #16, rel	CMPL A, RL1	CMPL A, @RW2+d8	ANDL A, RL1	ANDL A, @RW2+d8	ORL A, RL1	ORL A, @RW2+d8	XORL A, RL1	XORL A, @RW2+d8	R2, #8, rel	@RW2+d8, #8, rel
+3	ADDL A, RL1	ADDL A, @RW3+d8	SUBL A, RL1	SUBL A, @RW3+d8	RW3, #16, rel	@RW3+d8 #16, rel	CMPL A, RL1	CMPL A, @RW3+d8	ANDL A, RL1	ANDL A, @RW3+d8	ORL A, RL1	ORL A, @RW3+d8	XORL A, RL1	XORL A, @RW3+d8	R3, #8, rel	@RW3+d8, #8, rel
+4	ADDL A, RL2	ADDL A, @RW4+d8	SUBL A, RL2	SUBL A, @RW4+d8	RW4, #16, rel	@RW4+d8 #16, rel	CMPL A, RL2	CMPL A, @RW4+d8	ANDL A, RL2	ANDL A, @RW4+d8	ORL A, RL2	ORL A, @RW4+d8	XORL A, RL2	XORL A, @RW4+d8	R4, #8, rel	@RW4+d8, #8, rel
+5	ADDL A, RL2	ADDL A, @RW5+d8	SUBL A, RL2	SUBL A, @RW5+d8	RW5, #16, rel	@RW5+d8 #16, rel	CMPL A, RL2	CMPL A, @RW5+d8	ANDL A, RL2	ANDL A, @RW5+d8	ORL A, RL2	ORL A, @RW5+d8	XORL A, RL2	XORL A, @RW5+d8	R5, #8, rel	@RW5+d8, #8, rel
+6	ADDL A, RL3	ADDL A, @RW6+d8	SUBL A, RL3	SUBL A, @RW6+d8	RW6, #16, rel	@RW6+d8 #16, rel	CMPL A, RL3	CMPL A, @RW6+d8	ANDL A, RL3	ANDL A, @RW6+d8	ORL A, RL3	ORL A, @RW6+d8	XORL A, RL3	XORL A, @RW6+d8	R6, #8, rel	@RW6+d8, #8, rel
+7	ADDL A, RL3	ADDL A, @RW7+d8	SUBL A, RL3	SUBL A, @RW7+d8	RW7, #16, rel	@RW7+d8 #16, rel	CMPL A, RL3	CMPL A, @RW7+d8	ANDL A, RL3	ANDL A, @RW7+d8	ORL A, RL3	ORL A, @RW7+d8	XORL A, RL3	XORL A, @RW7+d8	R7, #8, rel	@RW7+d8, #8, rel
+8	ADDL A,@RW0	ADDL A, @RW0+d16	SUBL A,@RW0	SUBL A, @RW0+d16	@RW0, #16, rel	@RW0+d16 #16, rel	CMPL A,@RW0	CMPL A, @RW0+d16	ANDL A,@RW0	ANDL A, @RW0+d16	ORL A,@RW0	ORL A, @RW0+d16	XORL A,@RW0	XORL A, @RW0+d16	@RW0, #8, rel	@RW0+d16, #8, rel
+9	ADDL A,@RW1	ADDL A, @RW1+d16	SUBL A,@RW1	SUBL A, @RW1+d16	@RW1, #16, rel	@RW1+d16 #16, rel	CMPL A,@RW1	CMPL A, @RW1+d16	ANDL A,@RW1	ANDL A, @RW1+d16	ORL A,@RW1	ORL A, @RW1+d16	XORL A,@RW1	XORL A, @RW1+d16	@RW1, #8, rel	@RW1+d16, #8, rel
+A	ADDL A,@RW2	ADDL A, @RW2+d16	SUBL A,@RW2	SUBL A, @RW2+d16	@RW2, #16, rel	@RW2+d16 #16, rel	CMPL A,@RW2	CMPL A, @RW2+d16	ANDL A,@RW2	ANDL A, @RW2+d16	ORL A,@RW2	ORL A, @RW2+d16	XORL A,@RW2	XORL A, @RW2+d16	@RW2, #8, rel	@RW2+d16, #8, rel
+B	ADDL A,@RW3	ADDL A, @RW3+d16	SUBL A,@RW3	SUBL A, @RW3+d16	@RW3, #16, rel	@RW3+d16 #16, rel	CMPL A,@RW3	CMPL A, @RW3+d16	ANDL A,@RW3	ANDL A, @RW3+d16	ORL A,@RW3	ORL A, @RW3+d16	XORL A,@RW3	XORL A, @RW3+d16	@RW3, #8, rel	@RW3+d16, #8, rel
+C	ADDL A,@RW0+	ADDL A, @RW0+RW7	SUBL A,@RW0+	SUBL A, @RW0+RW7	Use prohibited	@RW0+RW7 #16, rel	CMPL A,@RW0+	CMPL A, @RW0+RW7	ANDL A,@RW0+	ANDL A, @RW0+RW7	ORL A,@RW0+	ORL A, @RW0+RW7	XORL A,@RW0+	XORL A, @RW0+RW7	Use prohibited	@RW0+RW7, #8, rel
+D	ADDL A,@RW1+	ADDL A, @RW1+RW7	SUBL A,@RW1+	SUBL A, @RW1+RW7	Use prohibited	@RW1+RW7 #16, rel	CMPL A,@RW1+	CMPL A, @RW1+RW7	ANDL A,@RW1+	ANDL A, @RW1+RW7	ORL A,@RW1+	ORL A, @RW1+RW7	XORL A,@RW1+	XORL A, @RW1+RW7	Use prohibited	@RW1+RW7, #8, rel
+E	ADDL A,@RW2+	ADDL A, @PC+d16	SUBL A,@RW2+	SUBL A, @PC+d16	Use prohibited	@PC+d16, #16, rel	CMPL A,@RW2+	CMPL A, @PC+d16	ANDL A,@RW2+	ANDL A, @PC+d16	ORL A,@RW2+	ORL A, @PC+d16	XORL A,@RW2+	XORL A, @PC+d16	Use prohibited	@PC+d16, #8, rel
+F	ADDL A,@RW3+	ADDL A, addr16	SUBL A,@RW3+	SUBL A, addr16	Use prohibited	addr16, #16, rel	CMPL A,@RW3+	CMPL A, addr16	ANDL A,@RW3+	ANDL A, addr16	ORL A,@RW3+	ORL A, addr16	XORL A,@RW3+	XORL A, addr16	Use prohibited	addr16, #8, rel

Table B.9-7 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 B.9-8 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 @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 @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 @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 @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 @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 @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 @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 @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 B.9-9 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 A, @RW0+d8	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 A, @RW1+d8	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 A, @RW2+d8	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 A, @RW3+d8	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 A, @RW4+d8	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 A, @RW5+d8	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 A, @RW6+d8	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 A, @RW7+d8	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 A,@RW0+d16	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 A,@RW1+d16	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 A,@RW2+d16	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 A,@RW3+d16	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 A,@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 A,@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 A,@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 A, addr16, A	MOVW @RW3+, #16	MOVW addr16, #16	XCHW A,@RW3+	XCHW A, addr16

Table B.9-10 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 B.9-11 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 B.9-12 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 B.9-13 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 B.9-14 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, A, R0	MULU A, @RW0+d8	MULUW A, 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, A, RW0	DIVUW A, @RW0+d8	DIV A, R0	DIV A, @RW0+d8	DIVW A, RW0	DIVW A, @RW0+d8
+1	MULU A, A, R1	MULU A, @RW1+d8	MULUW A, 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, A, RW1	DIVUW A, @RW1+d8	DIV A, R1	DIV A, @RW1+d8	DIVW A, RW1	DIVW A, @RW1+d8
+2	MULU A, A, R2	MULU A, @RW2+d8	MULUW A, 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, A, RW2	DIVUW A, @RW2+d8	DIV A, R2	DIV A, @RW2+d8	DIVW A, RW2	DIVW A, @RW2+d8
+3	MULU A, A, R3	MULU A, @RW3+d8	MULUW A, 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, A, RW3	DIVUW A, @RW3+d8	DIV A, R3	DIV A, @RW3+d8	DIVW A, RW3	DIVW A, @RW3+d8
+4	MULU A, A, R4	MULU A, @RW4+d8	MULUW A, 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, A, RW4	DIVUW A, @RW4+d8	DIV A, R4	DIV A, @RW4+d8	DIVW A, RW4	DIVW A, @RW4+d8
+5	MULU A, A, R5	MULU A, @RW5+d8	MULUW A, 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, A, RW5	DIVUW A, @RW5+d8	DIV A, R5	DIV A, @RW5+d8	DIVW A, RW5	DIVW A, @RW5+d8
+6	MULU A, A, R6	MULU A, @RW6+d8	MULUW A, 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, A, RW6	DIVUW A, @RW6+d8	DIV A, R6	DIV A, @RW6+d8	DIVW A, RW6	DIVW A, @RW6+d8
+7	MULU A, A, R7	MULU A, @RW7+d8	MULUW A, 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, A, RW7	DIVUW A, @RW7+d8	DIV A, R7	DIV A, @RW7+d8	DIVW A, RW7	DIVW A, @RW7+d8
+8	MULU A, A,@RW0	MULU A, @RW0+d16	MULUW A, A,@RW0	MULUW A, @RW0+d16	MUL A, @RW0	MUL A, @RW0+d16	MULW A, A,@RW0	MULW A, @RW0+d16	DIVU A, A,@RW0	DIVU A, @RW0+d16	DIVUW A, A,@RW0	DIVUW A, @RW0+d16	DIV A,@RW0	DIV A, @RW0+d16	DIVW A, A,@RW0	DIVW A, @RW0+d16
+9	MULU A, A,@RW1	MULU A, @RW1+d16	MULUW A, A,@RW1	MULUW A, @RW1+d16	MUL A, @RW1	MUL A, @RW1+d16	MULW A, A,@RW1	MULW A, @RW1+d16	DIVU A, A,@RW1	DIVU A, @RW1+d16	DIVUW A, A,@RW1	DIVUW A, @RW1+d16	DIV A,@RW1	DIV A, @RW1+d16	DIVW A, A,@RW1	DIVW A, @RW1+d16
+A	MULU A, A,@RW2	MULU A, @RW2+d16	MULUW A, A,@RW2	MULUW A, @RW2+d16	MUL A, @RW2	MUL A, @RW2+d16	MULW A, A,@RW2	MULW A, @RW2+d16	DIVU A, A,@RW2	DIVU A, @RW2+d16	DIVUW A, A,@RW2	DIVUW A, @RW2+d16	DIV A,@RW2	DIV A, @RW2+d16	DIVW A, A,@RW2	DIVW A, @RW2+d16
+B	MULU A, A,@RW3	MULU A, @RW3+d16	MULUW A, A,@RW3	MULUW A, @RW3+d16	MUL A, @RW3	MUL A, @RW3+d16	MULW A, A,@RW3	MULW A, @RW3+d16	DIVU A, A,@RW3	DIVU A, @RW3+d16	DIVUW A, A,@RW3	DIVUW A, @RW3+d16	DIV A,@RW3	DIV A, @RW3+d16	DIVW A, A,@RW3	DIVW A, @RW3+d16
+C	MULU A, A,@RW0+	MULU A, @RW0+RW7	MULUW A, A,@RW0+	MULUW A, @RW0+RW7	MUL A, A,@RW0+	MUL A, @RW0+RW7	MULW A, A,@RW0+	MULW A, @RW0+RW7	DIVU A, A,@RW0+	DIVU A, @RW0+RW7	DIVUW A, A,@RW0+	DIVUW A, @RW0+RW7	DIV A,@RW0+	DIV A, @RW0+RW7	DIVW A, A,@RW0+	DIVW A, @RW0+RW7
+D	MULU A, A,@RW1+	MULU A, @RW1+RW7	MULUW A, A,@RW1+	MULUW A, @RW1+RW7	MUL A, A,@RW1+	MUL A, @RW1+RW7	MULW A, A,@RW1+	MULW A, @RW1+RW7	DIVU A, A,@RW1+	DIVU A, @RW1+RW7	DIVUW A, A,@RW1+	DIVUW A, @RW1+RW7	DIV A,@RW1+	DIV A, @RW1+RW7	DIVW A, A,@RW1+	DIVW A, @RW1+RW7
+E	MULU A, A,@RW2+	MULU A, @PC+d16	MULUW A, A,@RW2+	MULUW A, @PC+d16	MUL A, A,@RW2+	MUL A, @PC+d16	MULW A, A,@RW2+	MULW A, @PC+d16	DIVU A, A,@RW2+	DIVU A, @PC+d16	DIVUW A, A,@RW2+	DIVUW A, @PC+d16	DIV A,@RW2+	DIV A, @PC+d16	DIVW A, A,@RW2+	DIVW A, @PC+d16
+F	MULU A, A,@RW3+	MULU A, addr16	MULUW A, A,@RW3+	MULUW A, addr16	MUL A, A,@RW3+	MUL A, addr16	MULW A, A,@RW3+	MULW A, addr16	DIVU A, A,@RW3+	DIVU A, addr16	DIVUW A, A,@RW3+	DIVUW A, addr16	DIV A,@RW3+	DIV A, addr16	DIVW A, A,@RW3+	DIVW A, addr16

Table B.9-15 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 B.9-16 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 B.9-17 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 RW5, @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 B.9-18 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 B.9-19 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+, 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 B.9-20 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 B.9-21 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

APPENDIX C Main Changes

Spansion Document Code: CM44-10142-5E

Page	Changes (For details, refer to main body.)	
2	CHAPTER 1 OVERVIEW 1.1 Overview of MB90920 Series ■ Overview of MB90920 Series	Changed Table 1.1-1. MB90F921 changed the planning product.
32	CHAPTER 2 CPU 2.3 Memory Map ■ Memory Map	Changed Figure 2.3-1. Correct the Parts No. MB90F923 / MB90923(under development) → MB90F923 / MB90923 MB90F924 / MB90924(under development) → MB90F924 / MB90924
55	2.8 General-purpose Register	Changed the summary sentence. (long word register RL0 to RL7) → (long word register RL0 to RL3)
91	CHAPTER 3 INTERRUPT 3.6 Interrupt by Extended Intelligent I/O Service (EI^{2}OS) ■ Buffer Address Pointer (BAP)	Changed the comments. (BAPH, BAPL) → (BAPM, BAPL)
122	CHAPTER 5 CLOCK 5.1 Clock ■ Clock Supply Map	Changed Figure 5.1-1. Added the note comment of X0A and X1A.
135	5.5 Clock Mode ■ Machine Clock	Changed Figure 5.1-1. MCS : Machine clock selection bit → PLL clock selection bit MCM : Machine clock display bit → PLL clock operation flag bit SCS : Machine clock display bit (sub) → Sub clock selection bit SCM :Machine clock selection bit (sub) → Sub clock operation flag bit CS1, CS0 : Machine clock → Multiplication rate selection bits
283	CHAPTER 10 INPUT CAPTURE 10.2 List of Input Capture Registers ■ List of Registers for 16-bit Free-run Timer Section	Changed Figure 10.2-1. Changed bit14 - → R/W - → 1
293	■ Timer Control Status Register (TCCSH, TCCSL)	Changed Figure 10.2-8. Changed bit14 - → R/W - → 1

Page	Changes (For details, refer to main body.)	
363	CHAPTER 14 DELAY INTERRUPT GENERATION MODULE 14.2 Operation of Delay Interrupt Generation Module ■ Operation of Delay Interrupt Generation Module	Changed Figure 14.2-1. ICR yy → IL ICR xx → ILM
501	CHAPTER 18 CAN CONTROLLER 18.3 Classification of CAN Controller Registers ■ Bit Configuration of Bit Timing Register (BTR)	Changed the comments. TSI → TS1 TS1.0 → TS2.0
627	CHAPTER 25 FLASH MEMORY	Corrected the term. write/erase → data write/erase sector erase wait → sector erase time-out
628	CHAPTER 25 FLASH MEMORY 25.1 Overview of Flash Memory	Deleted the following sentences from the summary sentence. (The selector operation, such as "enable sector protection", cannot be used.)
632	CHAPTER 25 FLASH MEMORY 25.3 Flash Memory Control Status Register (FMCS)	Corrected the attribute of read/writing bit3 and bit1. (W → R/W)
		Corrected the definition of the RDYINT bit. Corrected the explanation.
633		Corrected the definition of the RDY bit. (This bit enables to write/erase to the flash memory. → This bit is a status bit that indicates the status of data-writing/erasing operation of the flash memory)
		Deleted the item of "■ Automatic Algorithm Termination Timing".
634	CHAPTER 25 FLASH MEMORY 25.4 Flash Memory Write Control Registers (FWR0/FWR1)	Corrected function of Reserved bits in Table 25.4-1.
635		Corrected the definition of bit3 to bit0 for product with 3M-bit flash memory of Table 25.4-3.
640	CHAPTER 25 FLASH MEMORY 25.5 Starting the Flash Memory Automatic Algorithm	Corrected the command sequence of Table 25.5-1. (Read/reset → Reset Programming program → Data write Deleted the RA and RD from the line of reset.)
641	CHAPTER 25 FLASH MEMORY 25.6 Confirming the Automatic Algorithm Execution State	Corrected the line of the sector deletion in Table 25.6-2.

Page	Changes (For details, refer to main body.)	
642	CHAPTER 25 FLASH MEMORY 25.6.1 Data Polling Flag (DQ7)	Corrected the transition of "Chip/sector erase → Completed" in the Table 25.6-3. (DQ7 : "0 → 1" → "0 → 1(DATA:7)")
		Corrected the transition of "Sector erase time-out → Sector erase started" in the Table 25.6-3. (DQ7 : "0" → "1 → 0")
		Changed the item. (■ Chip/Sector Erase → ■ Sector Erase) Corrected the explanation. Added the restriction matter.
		Added the item of "■ Chip Erase".
644	CHAPTER 25 FLASH MEMORY 25.6.2 Toggle Bit Flag (DQ6)	Corrected the transition of "Chip/sector erase → Completed" in the Table 25.6-5. (DQ6 : "Toggle → Stop" → "Toggle → DATA:6")
645	CHAPTER 25 FLASH MEMORY 25.6.3 Timing Limit Exceeded Flag (DQ5)	Corrected the transition of "Chip/sector erase → Completed" in the Table 25.6-7. (DQ5 : "0 → 1" → "0 → DATA:5")
646	CHAPTER 25 FLASH MEMORY 25.6.4 Sector Erase Timer Flag (DQ3)	Corrected the transition of "Chip/sector erase → Completed" in the Table 25.6-9. (DQ3 : "1" → "1 → DATA:3")
650	CHAPTER 25 FLASH MEMORY 25.7.2 Write Data to Flash Memory	Added the setting FWR0/FWR1 in Figure 25.7-1.
652	CHAPTER 25 FLASH MEMORY 25.7.4 Erasing Arbitrary Data of Flash Memory (Sector Erase)	Corrected the time of the sector erase time-out. (80 μs → at least 50 μs)
653		Corrected flowchart of Figure 25.7-2.
656	CHAPTER 25 FLASH MEMORY 25.8 Flash Security Function	Added the protection code in Table 25.8-1.
657 to 659	CHAPTER 25 FLASH MEMORY 25.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems	Added the "25.9 Restrictions on Data Polling Flag (DQ7) and How to Avoid Problems.
660	CHAPTER 25 FLASH MEMORY 25.10 Notes on Using Flash Memory	Added the "25.10 Notes on Using Flash Memory".

Page	Changes (For details, refer to main body.)	
746	B.3 Direct Addressing ● I/O direct addressing (io)	Changed Figure B.3-5. (MOVW A, i : 0C0H → MOVW A, I:0C0H)
		Added the note to Figure B.3-5.
747	B.3 Direct Addressing ● Abbreviated direct addressing (dir)	Added the note to Figure B.3-6.
748	B.3 Direct Addressing ● I/O direct bit addressing (io:bp)	Changed Figure B.3-8. (SETB i : 0C1H : 0 → SETB I:0C1H:0)
		Added the note to Figure B.3-8.
	B.3 Direct Addressing ● Abbreviated direct bit addressing (dir:bp)	Added the note to Figure B.3-9.
754	B.4 Indirect Addressing ● Program counter relative branch addressing (rel)	Changed Figure B.4-7. (BRA 10H → BRA 3C32H)
755	B.4 Indirect Addressing ● Register list (rlst)	Changed Figure B.4-9. (POPW, RW0, RW4 → POPW RW0, RW4)
780	B.9 Instruction Map ■ Structure of Instruction Map	Changed column: instruction in Table B.9-1. (@RW2+d8, #8, rel → CBNE @RW2+d8, #8, rel)
781	B.9 Instruction Map	Changed the operand at row: +0, column: E0 in Table B.9-2. (#4 → #vct4)
		Changed the mnemonic at row: +0, column: D0 in Table B.9-2. (MOV → MOVN)
		Changed the mnemonic at row: +0, column: B0 in Table B.9-2. (MOV → MOVX)
		Changed the mnemonic at row: +8, column: B0 in Table B.9-2. (MOV → MOVW)
783		Changed the mnemonic at row: +0, column: E0 in Table B.9-4. (FILSI → FILSWI)
784		Changed Table B.9-5. (・Moved "MUL A" and "MULW A" instruction from column:60 to column:70. ・Changed mnemonic and moved the Instruction from column:60, row:+A to column:70, row:+A. (DIVU → DIV))

Page	Changes (For details, refer to main body.)	
785	B.9 Instruction Map	Changed the operand at row: +E and +F, column: F0 in Table B.9-6. (,#8, rel → #8, rel)
788		Changed the operand at row: +8 to +E, column: 50 in Table B.9-9. (@@ → @)
		Changed the operand at row: +0 to +7, column: 20 in Table B.9-9. (RWi → @RWi)
789		Changed the operand at column: E0 and F0 in Table B.9-10. (,r → ,rel)
790		Changed the operand at column: 70 in Table B.9-11. (NEG A, → NEG)
791		Changed the operand at column: E0 and F0 in Table B.9-12. (,r → ,rel)

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

Index

Numerics

1/2 Bias
1/2 Bias, 1/2 Duty Output Waveform 618
1/2 Duty
1/2 Bias, 1/2 Duty Output Waveform 618
Example of LCD Panel Connection and Display Data (1/2 Duty Drive Method) 620
1/3 Bias
1/3 Bias, 1/3 Duty Output Waveform 621
1/3 Bias, 1/4 Duty Output Waveform 624
1/3 Duty
1/3 Bias, 1/3 Duty Output Waveform 621
Example of LCD Panel Connection and Display Data (1/3 Duty Drive Method) 623
1/4 Duty
1/3 Bias, 1/4 Duty Output Waveform 624
Example of LCD Panel Connection and Display Data (1/4 Duty Drive Method) 626
16-bit Free-run Timer
Clear Timing for 16-bit Free-run Timer 322
Count Timing of 16-bit Free-run Timer 322
List of Registers for 16-bit Free-run Timer Section 304
Operations of 16-bit Free-run Timer 321
16-bit Input Capture
Input Timing for a 16-bit Input Capture 320
Operation of 16-bit Input Capture 319
16-bit Reload Registers
16-bit Reload Registers (TMRLR0 to TMRLR3) 338
16-bit Reload Timer
16-bit Reload Timer Settings 340
Block Diagram of 16-bit Reload Timer 326
Block Diagram of Pins for 16-bit Reload Timer 329
EI^{2}OS Function of 16-bit Reload Timer 339
Interrupts and EI^{2}OS of 16-bit Reload Timer 325, 339
Interrupts of 16-bit Reload Timer 339
Notes on Using 16-bit Reload Timer 348
Operation Mode of 16-bit Reload Timer 324
Pins of 16-bit Reload Timer 328
Register List of 16-bit Reload Timer 330
16-bit Timer Registers
16-bit Timer Registers (TMR0 to TMR3) 337
24-bit Operand
Specification with 24-bit Operand 45
32-bit Register
Indirect Specification with 32-bit Register 45
8-/10-bit A/D Converter
Block Diagram of 8-/10-bit A/D Converter 417
Conversion Modes of 8-/10-bit A/D Converter 415
EI^{2}OS of 8-/10-bit A/D Converter 435
Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter 445
Functions of 8-/10-bit A/D Converter 414
Interrupts and EI^{2}OS of 8-/10-bit A/D Converter 435
List of Registers and Initial Values of 8-/10-bit A/D Converter 421
Pins of 8-/10-bit A/D Converter 420
Precautions When Using 8-/10-bit A/D Converter 449

A

A
Accumulator (A) 54
A/D Control Status Register
Lower Bits in the A/D Control Status Register (ADCS0) 427
Upper Bits in the A/D Control Status Register (ADCS1) 423
A/D Conversion
Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter 445
A/D Converter
Block Diagram of 8-/10-bit A/D Converter 417
Conversion Modes of 8-/10-bit A/D Converter 415
EI^{2}OS of 8-/10-bit A/D Converter 435
Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter 445
Functions of 8-/10-bit A/D Converter 414
Interrupts and EI^{2}OS of 8-/10-bit A/D Converter 435
Interrupts of A/D Converter 435
List of Registers and Initial Values of 8-/10-bit A/D Converter 421
Pins of 8-/10-bit A/D Converter 420
Power-on Sequence for A/D Converter Power Supply and Analog Input 34
Precautions When Using 8-/10-bit A/D Converter 449
Processing of A/D Converter Power Supply Pins 33
A/D Data Registers
A/D Data Registers (ADCR0/ADCR1) 429
A/D Setting Registers
A/D Setting Registers (ADSR0/ADSR1) 430
Acceptance Filter
Setting Acceptance Filter 578
Acceptance Filtering
Acceptance Filtering 574
Acceptance Mask Registers
Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1) 561
Acceptance Mask Selection Register
Bit Configuration of Acceptance Mask Selection Register (AMSR) 559
Access Space
Bank Register and Access Space 46
Accumulator
Accumulator (A) 54
ADB
Bank Registers (PCB, DTB, USB, SSB, ADB) ... 65
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
ADCR
A/D Data Registers (ADCR0/ADCR1) 429
ADCS
Continuous Conversion Mode (ADCS: MD1, MD0=10_B) 436
Lower Bits in the A/D Control Status Register (ADCS0) 427
Single Conversion Mode (ADCS: MD1, MD0=00_B or 01_B) 436
Stop Conversion Mode (ADCS: MD1, MD0=11_B) 436
Upper Bits in the A/D Control Status Register (ADCS1) 423
Address Match Detection
Block Diagram of the Address Match Detection Function 662
Flowchart of the Address Match Detection Function Processing 668
Operation of the Address Match Detection Function 664
Register Configuration of the Address Match Detection Function 662
Addressing
Addressing 786
Bank Addressing and Default Space 47
Direct Addressing 788
Indirect Addressing 795
Linear Addressing and Bank Addressing 44
ADER
Analog Input Enable Register (ADER6) 434
ADSR
A/D Setting Registers (ADSR0/ADSR1) 430
Amplitude Data Registers
Amplitude Data Registers (SGAR0/SGAR1) 658
AMR
Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1) 561
AMSR
Bit Configuration of Acceptance Mask Selection Register (AMSR) 559
Analog Input
Power-on Sequence for A/D Converter Power Supply and Analog Input 34
Analog Input Enable Register
Analog Input Enable Register (ADER6) 434
Asynchronous LIN Mode
Operation in Asynchronous LIN Mode 506

Asynchronous Mode
Operation in Asynchronous Mode 498

B

Bank Addressing
Bank Addressing and Default Space 47
Linear Addressing and Bank Addressing 44
Bank Register
Bank Register and Access Space 46
Bank Registers (PCB, DTB, USB, SSB, ADB) 65
Bank Select Prefix
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
BAP
Buffer Address Pointer (BAP) 107
Basic Configuration
Basic Configuration of MB90F922 Serial Programming Connection 710
Baud Rate
Calculating the Baud Rate 491
Reload Value and Baud Rate for Each Clock Speed 492
UART Baud Rate Selection 489
Baud Rate Generator Registers
Bit Configuration of Baud Rate Generator Registers (BGRn0/BGRn1) 481
BGRn
Bit Configuration of Baud Rate Generator Registers (BGRn0/BGRn1) 481
Bias
1/2 Bias, 1/2 Duty Output Waveform 618
1/3 Bias, 1/3 Duty Output Waveform 621
1/3 Bias, 1/4 Duty Output Waveform 624
Bidirectional Communication
Bidirectional Communication Function 510
Bit Configuration
Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1) 561
Bit Configuration of Acceptance Mask Selection Register (AMSR) 559
Bit Configuration of Baud Rate Generator Registers (BGRn0/BGRn1) 481
Bit Configuration of Bit Timing Register (BTR) 544
Bit Configuration of Control Status Register (CSR) 536
Bit Configuration of Data Register x (x=0 to 15) (DTRx) 569
Bit Configuration of DLC Register x (x=0 to 15) (DLCRx) 568
Bit Configuration of Extended Communication Control Register (ECCR) 478
Bit Configuration of Extended Status Control Register (ESCR) 475
Bit Configuration of ID Register x (x=0 to 15) (IDRx) 564
Bit Configuration of IDE Register (IDER) 548
Bit Configuration of Last Event Indication Register (LEIR) 541
Bit Configuration of LCD Controller/Driver Register 602
Bit Configuration of Message Buffer Valid Register (BVALR) 547
Bit Configuration of Processor Status (PS) 59
Bit Configuration of PWM Control Register 643
Bit Configuration of Receive and Transmit Error Counters (RTEC) 543
Bit Configuration of Receive Complete Register (RCR) 555
Bit Configuration of Receive Interrupt Enable Register (RIER) 558
Bit Configuration of Receive Overrun Register (ROVRR) 557
Bit Configuration of Remote Frame Receive Wait Register (RFWTR) 551
Bit Configuration of Remote Request Receive Register (RRTRR) 556
Bit Configuration of Second/Minute/Hour/Day Data Registers 381
Bit Configuration of Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1) 655
Bit Configuration of Sub-second Data Register 380
Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2) 608
Bit Configuration of the LCD Output Control Register 3 (LOCR3) 611
Bit Configuration of the Lower Bits in the LCD Control Register (LCRL) 603
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
Bit Configuration of the PWM1 and PWM2 Compare Registers 644
Bit Configuration of the PWM1 and PWM2 Select Registers 646
Bit Configuration of the Time-base Timer Control Register (TBTC) 283
Bit Configuration of the Upper Bits in the LCD Control Register (LCRH) 606
Bit Configuration of the Watchdog Timer Control Register (WDTC) 281
Bit Configuration of Timer Control Register 378

Bit Configuration of Transmission Cancel Register (TCANR) 552
Bit Configuration of Transmission Complete Register (TCR) 553
Bit Configuration of Transmission Interrupt Enable Register (TIER) 554
Bit Configuration of Transmission Request Register (TREQR) 549
Bit Configuration of Transmission RTR Register (TRTRR) 550
Bit Timing Register
Bit Configuration of Bit Timing Register (BTR) 544
Sample Setting of the Bit Timing Register 546
Block Diagram
Block Diagram 17, 355
Block Diagram of 16-bit Reload Timer 326
Block Diagram of 8-/10-bit A/D Converter 417
Block Diagram of CAN Controller 526
Block Diagram of Delay Interrupt Generation Module 386
Block Diagram of DTP/External Interrupt Circuit 392
Block Diagram of DTP/External Interrupt Circuit Pins 394
Block Diagram of External Reset Pin 127
Block Diagram of Input Capture 301
Block Diagram of LCD Controller/Driver 593
Block Diagram of LCD Controller/Driver Pins 601
Block Diagram of LIN-UART 456
Block Diagram of LIN-UART Pins 461
Block Diagram of Low-power Consumption Control Circuit 164
Block Diagram of Pins for 16-bit Reload Timer 329
Block Diagram of Real-time Watch Timer 374
Block Diagram of the Address Match Detection Function 662
Block Diagram of the Clock Generation Block 141
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
Block Diagram of the ROM Mirror Function Select Module 672
Block Diagram of the Sound Generator 652
Block Diagram of the Stepping Motor Controller 640
Block Diagram of Watchdog Timer/Time-base Timer/ Watch Timer 279
Pin Block Diagram for Port 0 204
Pin Block Diagram for Port 1 209
Pin Block Diagram for Port 2 214
Pin Block Diagram for Port 3 219
Pin Block Diagram for Port 4 224
Pin Block Diagram for Port 5 230
Pin Block Diagram for Port 6 234
Pin Block Diagram for Port 7 240
Pin Block Diagram for Port 8 246
Pin Block Diagram for Port 9 252
Pin Block Diagram for Port C 258
Pin Block Diagram for Port D 264
Pin Block Diagram for Port E 269
Brightness Adjustment
Brightness Adjustment When Using the Internal Divided Resistor 597
BTR
Bit Configuration of Bit Timing Register (BTR) 544
Buffer Address Pointer
Buffer Address Pointer (BAP) 107
Bus Mode
Bus Mode 192
Bus Mode Setting Bits 194
Bus Operation Stop
Bus Operation Stop Bit (HALT=1) 539
State Between Bus Operation Stops (HALT=1) 539
BVAL
Caution for Disabling Message Buffer by BVAL Bits 587
BVALR
Bit Configuration of Message Buffer Valid Register (BVALR) 547

C

Calculating
Calculating the Execution Cycle Count 804
CAN Controller
Block Diagram of CAN Controller 526
Clearing a CAN Controller Transmission Request 571
Completing CAN Controller Transmission 571
Features of CAN Controller 524
Flowchart of CAN Controller Reception 577
Flowchart of CAN Controller Transmission 573
Starting CAN Controller Transmission 571
CAN Reception
Flowchart of CAN Reception Setting 576
CAN Transmission
Flowchart of CAN Transmission Setting 572

CAN Transmission/Reception
Sample Program for CAN Transmission/Reception 588
CAN WAKE UP
CAN WAKE UP Function 586
Pins Used for CAN WAKE UP Function 586
CCR
Condition Code Register (PS:CCR) 60
Chip
Chip Erase 690
Data Write and Chip/Sector Erase 692, 693
Erasing All Data of Flash Memory (Chip Erase) 699
Notes on Chip Erase 699
CKSCR
Clock Selection Register (CKSCR) 145
Clock
Clock 138
Clock Mode 151, 161
Clock Supply Map 140
Connection of Oscillator and External Clock 156
Event Count Mode (External Clock Mode) 325
External Clock 492
Internal Clock Mode 324
Machine Clock 152
Operation in Internal Clock Mode (Reload Mode) 342
Operation of Internal Clock Mode (One Shot Mode) 344
Oscillator Clock Frequency and Serial Clock Input Frequency 712
Precautions for PLL Clock Mode Operation 35
Reload Value and Baud Rate for Each Clock Speed 492
Sample Program for Internal Clock Mode 349
Selection of PLL Clock Multiplication Rate 152
Switching the Clock Mode 188
Transition of Clock Mode 151
Use of External Clock 33
Clock Generation Block
Block Diagram of the Clock Generation Block 141
List of the Registers in the Clock Generation Block and Its Initial Values 144
Clock Mode
Clock Mode 151, 161
Switching the Clock Mode 188
Transition of Clock Mode 151
Clock Selection Register
Clock Selection Register (CKSCR) 145
CMR
Common Register Bank Prefix (CMR) 69
Command Sequence Table
Command Sequence Table 688
Common Register Bank Prefix
Common Register Bank Prefix (CMR) 69
Communication
LIN Master/Slave Type Communication Function 515
Master/Slave Type Communication Function 512
Compare Clear Register
Compare Clear Register (CPCLR) 314
Compare Time
Setup for Compare Time (Bits CT2 to CT0) 433
Condition Code Register
Condition Code Register (PS:CCR) 60
Continuous Conversion Mode
Continuous Conversion Mode (ADCS: MD1, MD0=10_B) 436
Operations and Applications of Continuous Conversion Mode 441
Setup for Continuous Conversion Mode 440
Control Status Register
Bit Configuration of Control Status Register (CSR) 536
Controller/Driver
Block Diagram of LCD Controller/Driver 593
Conversion Mode
Continuous Conversion Mode (ADCS: MD1, MD0=10_B) 436
Conversion Modes of 8-/10-bit A/D Converter 415
Operations and Applications of Continuous Conversion Mode 441
Operations and Applications of Single Conversion Mode 439
Operations and Applications of Stop Conversion Mode 442
Setup for Continuous Conversion Mode 440
Setup for Single Conversion Mode 438
Setup for Stop Conversion Mode 442
Single Conversion Mode (ADCS: MD1, MD0=00_B or 01_B) 436
Stop Conversion Mode (ADCS: MD1, MD0=11_B) 436
Conversion Operation
Conversion Operation by EI^2OS Function 444
CPCLR
Compare Clear Register (CPCLR) 314

CPU
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
CPU Features 38
CPU Intermittent Operation Mode 162, 170
CPU Operation Detection Reset Circuit 629
CPU Operation Modes and Current Consumption 160
Inter-CPU Connection Method 496
Notes on Using the CPU Operation Detection Reset Circuit 635
Operation of the CPU Operation Detection Reset Circuit 634
Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637
CPU Intermittent Operation Mode
CPU Intermittent Operation Mode 162, 170
CPU Operation Detection Reset Circuit
CPU Operation Detection Reset Circuit 629
Notes on Using the CPU Operation Detection Reset Circuit 635
Operation of the CPU Operation Detection Reset Circuit 634
CSR
Bit Configuration of Control Status Register (CSR) 536
CT
Setup for Compare Time (Bits CT2 to CT0) 433
Current Consumption
CPU Operation Modes and Current Consumption 160
Cypress Standard
Pins Used for Cypress Standard Serial On-board Writing 711

D

Data Counter
Data Counter (DCT) 105
Data Polling Flag
Data Polling Flag (DQ7) State Transition 690
Data Protection
Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter 445
Data Register
Bit Configuration of Data Register x (x=0 to 15) (DTRx) 569
Data Write
Data Write 690
Data Write and Chip/Sector Erase 692, 693
Data-Writing
Data-Writing Procedure to the Flash Memory 698
Data-Writing/Erasing Flash Memory 695
DCT
Data Counter (DCT) 105
Decrement Grade Registers
Decrement Grade Registers (SGDR0/SGDR1) 659
Dedicated Register
Configuration of Dedicated Register 51
Dedicated Registers and General-purpose Registers 50
Default Space
Bank Addressing and Default Space 47
Delay Interrupt Generation Module
Block Diagram of Delay Interrupt Generation Module 386
Interrupts of Delay Interrupt Generation Module and EI^{2}OS 386
List of Registers of Delay Interrupt Generation Module 386
Notes on Using Delay Interrupt Generation Module 387
Operation of Delay Interrupt Generation Module 387
Description
Description of Instruction Presentation Items and Symbols 807
Descriptor
Configuration of Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) 103
Direct Addressing
Direct Addressing 788
Direct Page Register
Direct Page Register (DPR) 64
Display
Display RAM and Output Pins 612
DLC Register
Bit Configuration of DLC Register x (x=0 to 15) (DLCRx) 568
DLCRx
Bit Configuration of DLC Register x (x=0 to 15) (DLCRx) 568
DPR
Direct Page Register (DPR) 64

DQ3
Transition of State of Sector Erase Timer Flag (DQ3) 694
DQ5
Transition of State of Timing Limit Exceeded Flag (DQ5) 693
DQ6
Toggle Bit Flag (DQ6) State Transition 692
DQ7
Data Polling Flag (DQ7) State Transition 690
Drive Waveform
Drive Waveform of the LCD 617
Driver
Output Driver Driving Power Supply for Port 7 244
Output Driver Driving Power Supply for Port 8 250
DTB
Bank Registers (PCB, DTB, USB, SSB, ADB) 65
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
DTP
Operation of DTP Function 405
Sample Program of DTP Function 410
DTP/External Interrupt
Block Diagram of DTP/External Interrupt Circuit 392
Block Diagram of DTP/External Interrupt Circuit Pins 394
DTP/External Interrupt Function 390
DTP/External Interrupt Operation 402
Interrupts of DTP/External Interrupt Circuit and EI^{2}OS 391
Notes on Using the DTP/External Interrupt Circuit 407
Pins of DTP/External Interrupt Circuit 394
Registers of DTP/External Interrupt Circuit 396
Settings of the DTP/External Interrupt Circuit ...401
DTRx
Bit Configuration of Data Register x (x=0 to 15) (DTRx) 569
Duty
1/2 Bias, 1/2 Duty Output Waveform 618
1/3 Bias, 1/3 Duty Output Waveform 621
1/3 Bias, 1/4 Duty Output Waveform 624
Example of LCD Panel Connection and Display Data (1/2 Duty Drive Method) 620
Example of LCD Panel Connection and Display Data (1/3 Duty Drive Method) 623
Example of LCD Panel Connection and Display Data (1/4 Duty Drive Method) 626
DV_{CC}, DV_{SS}
Handling of Power Supply for High-current Output Buffer Pins (DV_{CC}, DV_{SS}) 34

E

E^{2}PROM
E^{2}PROM Memory Map 665
ECCR
Bit Configuration of Extended Communication Control Register (ECCR) 478
Effective Address Field
Effective Address Field 787, 806
EI^{2}OS
Configuration of Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) 103
Conversion Operation by EI^{2}OS Function 444
EI^{2}OS Function of 16-bit Reload Timer 339
EI^{2}OS Functions of PPG Timer 367
EI^{2}OS of 8-/10-bit A/D Converter 435
Extended Intelligent I/O Service (EI^{2}OS) 101
Extended Intelligent I/O Service (EI^{2}OS) Status Register (ISCS) 106
Input Capture Interrupts and EI^{2}OS 318
Interrupt of PPG Timer and EI^{2}OS 366
Interrupts and EI^{2}OS of 16-bit Reload Timer 325, 339
Interrupts and EI^{2}OS of 8-/10-bit A/D Converter 435
Interrupts and EI^{2}OS of Time-base Timer 291
Interrupts of Delay Interrupt Generation Module and EI^{2}OS 386
Interrupts of DTP/External Interrupt Circuit and EI^{2}OS 391
Interrupts of Real-time Watch Timer and EI^{2}OS 383
LIN-UART EI^{2}OS Functions 484
LIN-UART Interrupt and EI^{2}OS 454
LIN-UART Interrupts and EI^{2}OS 484
Operation of Extended Intelligent I/O Service (EI^{2}OS) 102
Procedure for Using the Extended Intelligent I/O Service (EI^{2}OS) 110
Processing Procedure for Extended Intelligent I/O Service (EI^{2}OS) 108
Processing Time for Extended Intelligent I/O Service (EI^{2}OS) (Time Consumed per Transfer) 112
Specification of Processing for Sample Program of Extended Intelligent I/O Service (EI^{2}OS) 119
Watch Timer Interrupts and EI^{2}OS 293

EIRR
External Interrupt Source Register (EIRR) 397
ELVRH
External Interrupt Level Setting Register (ELVRH/ELVRL) 399
ELVRL
External Interrupt Level Setting Register (ELVRH/ELVRL) 399
Erase
Chip Erase 690
Erasing All Data of Flash Memory (Chip Erase) 699
How to Write/Erase Data Flash Memory 676
Notes on Chip Erase 699
Restarting Sector Erase of Flash Memory 703
Sector Erase 690, 694
Sector Erase Suspended 690, 692, 694
Suspending Sector Erase of Flash Memory 702
Erasing
Data-Writing/Erasing Flash Memory 695
Erasing Optional Data (Erasing Sectors) in Flash Memory 700
Erasing Sectors in the Flash Memory 700
ESCR
Bit Configuration of Extended Status Control Register (ESCR) 475
Event Count Mode
Event Count Mode 346
Event Count Mode (External Clock Mode) 325
Sample Program for Event Count Mode 350
Exception
Exception Handling Interrupt by Execution of Undefined Instruction 114
Execution Cycle Count
Calculating the Execution Cycle Count 804
Execution Cycle Count 803
Extended Communication Control Register
Bit Configuration of Extended Communication Control Register (ECCR) 478
Extended Intelligent I/O Service
Configuration of Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) 103
Extended Intelligent I/O Service (EI^{2}OS) 101
Extended Intelligent I/O Service (EI^{2}OS) Status Register (ISCS) 106
Operation of Extended Intelligent I/O Service (EI^{2}OS) 102
Procedure for Using the Extended Intelligent I/O Service (EI^{2}OS) 110
Processing Procedure for Extended Intelligent I/O Service (EI^{2}OS) 108
Processing Time for Extended Intelligent I/O Service (EI^{2}OS) (Time Consumed per Transfer) 112
Specification of Processing for Sample Program of Extended Intelligent I/O Service (EI^{2}OS) 119
Extended Status Control Register
Bit Configuration of Extended Status Control Register (ESCR) 475
External Clock
Connection of Oscillator and External Clock 156
External Clock 492
Use of External Clock 33
External Clock Mode
Event Count Mode (External Clock Mode) 325
External Divide Resistor
Using the External Divide Resistor 599
External Divided Resistor
External Divided Resistor of LCD Controller/Driver 598
External Interrupt
External Interrupt Function 404
Sample Program of External Interrupt Function 409
External Interrupt Level Setting Register
External Interrupt Level Setting Register (ELVRH/ELVRL) 399
External Interrupt Source Register
External Interrupt Source Register (EIRR) 397
External Reset Pin
Block Diagram of External Reset Pin 127

F

F2MC-16LX Instruction List
F2MC-16LX Instruction List 810
Filter
Setting Acceptance Filter 578
Filtering
Acceptance Filtering 574
Flag
Data Polling Flag (DQ7) State Transition 690
Toggle Bit Flag (DQ6) State Transition 692
Transition of State of Sector Erase Timer Flag (DQ3) 694
Transition of State of Timing Limit Exceeded Flag (DQ5) 693
Flag Change Suppress Prefix
Flag Change Suppress Prefix (NCC) 70
Flash Memory
Capacities and Models of Flash Memory 676

Data-Writing Procedure to the Flash Memory 698
Data-Writing/Erasing Flash Memory 695
Erasing All Data of Flash Memory (Chip Erase) 699
Erasing Optional Data (Erasing Sectors) in Flash Memory 700
Erasing Sectors in the Flash Memory 700
Features of Flash Memory 676
How to Write/Erase Data Flash Memory 676
Restarting Sector Erase of Flash Memory 703
Setting Flash Memory to the Read/Reset State 696
Suspending Sector Erase of Flash Memory 702
Write Data to the Flash Memory 697
Flash Memory Control Status Register
Flash Memory Control Status Register (FMCS) 680
Flash Memory Write Control Registers
Flash Memory Write Control Registers (FWR0/FWR1) 682
Setup Flowchart for Flash Memory Write Control Registers (FWR0/FWR1) 685
Flash Microcontroller Programmer
System Configuration of Flash Microcontroller Programmer 713
Flash Security
Flash Security Function 35
FMCS
Flash Memory Control Status Register (FMCS) 680
Note on Setting the FMCS:WE Bit 687
Frame
Processing for Receiving Data Frame and Remote Frame 575
Setting Frame Format 578
Free-run Timer
Clear Timing for 16-bit Free-run Timer 322
Count Timing of 16-bit Free-run Timer 322
List of Registers for 16-bit Free-run Timer Section 304
Operations of 16-bit Free-run Timer 321
Frequency Data Registers
Frequency Data Registers (SGFR0/SGFR1) 657
FWR
Flash Memory Write Control Registers (FWR0/FWR1) 682
Setup Flowchart for Flash Memory Write Control Registers (FWR0/FWR1) 685

G

General Control Registers
General Control Registers 527
General-purpose Register
Configuration of General-purpose Register 66
Dedicated Registers and General-purpose Registers 50
General-purpose Register Area and Register Bank Pointer 61

H

HALT
Bus Operation Stop Bit (HALT=1) 539
State Between Bus Operation Stops (HALT=1) 539
Hardware Interrupt
Functions of Hardware Interrupt 87
Hardware Interrupt Operation 91
Hardware Interrupt Processing Time 97
Operation Flow of Hardware Interrupt 92
Procedure for Using Hardware Interrupt 94
Return from Hardware Interrupt 90
Starting Hardware Interrupt 90
Structure of Hardware Interrupt 88
Suppressing Hardware Interrupt 88
Hardware Sequence Flags
Hardware Sequence Flags 689
High-current Output Buffer
Handling of Power Supply (DVcc/DVss) for High-current Output Buffer Pin 650
Handling of Power Supply for High-current Output Buffer Pins (DV_{CC}, DV_{SS}) 34

I

I/O Area
I/O Area 41
I/O Circuit Types
I/O Circuit Types 27
I/O Port
I/O Port Functions 198
List of I/O Port Registers 201
Sample Program for I/O Ports 276
I/O Register Address Pointer
I/O Register Address Pointer (IOA) 105
ICE
Input Capture Edge Register (ICE) 310
ICR
Configuration of Interrupt Control Register (ICR) 84
Interrupt Control Registers (ICR00 to ICR15) 82

ICS
Input Capture Control Status Register (ICS01/23/45/67) 307
ID
Setting ID 578
ID Register
Bit Configuration of ID Register x (x=0 to 15) (IDRx) 564
Sample Setting of the ID Register 565
IDE Register
Bit Configuration of IDE Register (IDER) 548
IDER
Bit Configuration of IDE Register (IDER) 548
IDRx
Bit Configuration of ID Register x (x=0 to 15) (IDRx) 564
ILM
Interrupt Level Mask Register (PS:ILM) 62
Indirect Addressing
Indirect Addressing 795
Indirect Specification
Indirect Specification with 32-bit Register 45
Initialization
Initialization 35
Input Capture
Block Diagram of Input Capture 301
Input Capture Interrupts and EI^{2}OS 318
Input Timing for a 16-bit Input Capture 320
List of Registers for Input Capture Section 305
Operation of 16-bit Input Capture 319
Input Capture Control Status Register
Input Capture Control Status Register (ICS01/23/45/67) 307
Input Capture Edge Register
Input Capture Edge Register (ICE) 310
Input Capture Register
Input Capture Register (IPCP0 to IPCP7) 307
Input Level Select Register
Input Level Select Register 0 (PIL0) 273
Input Level Selection Register
Input Level Selection Register 1 (PIL1) 273
Input Level Selection Register 2 (PIL2) 274
Instruction
Description of Instruction Presentation Items and Symbols 807
Exception Handling Interrupt by Execution of Undefined Instruction 114
F2MC-16LX Instruction List 810
Instruction Types 785
Structure of Instruction Map 824
Instruction Presentation Items and Symbols
Description of Instruction Presentation Items and Symbols 807
INT9
INT9 Interrupt 669
Inter-CPU Connection Method
Inter-CPU Connection Method 496
Internal Clock Mode
Internal Clock Mode 324
Operation in Internal Clock Mode (Reload Mode) 342
Operation of Internal Clock Mode (One Shot Mode) 344
Sample Program for Internal Clock Mode 349
Internal Divided Resistor
Brightness Adjustment When Using the Internal Divided Resistor 597
Internal Divided Resistor of LCD Controller/Driver 596
Using the Internal Divided Resistor 597
Interrupt
Example Program for Interrupt Handling 117
Exception Handling Interrupt by Execution of Undefined Instruction 114
External Interrupt Function 404
Functions of Hardware Interrupt 87
Hardware Interrupt Operation 91
Hardware Interrupt Processing Time 97
Input Capture Interrupts and EI^{2}OS 318
INT9 Interrupt 669
Interrupt of PPG Timer 366
Interrupt of PPG Timer and EI^{2}OS 366
Interrupt of Real-time Watch Timer 383
Interrupt Operation 75
Interrupt Sources and Interrupt Vectors/Interrupt Control Registers 77
Interrupt Vector 76
Interrupts and EI^{2}OS of 16-bit Reload Timer 325, 339
Interrupts and EI^{2}OS of 8-/10-bit A/D Converter 435
Interrupts and EI^{2}OS of Time-base Timer 291
Interrupts of 16-bit Reload Timer 339
Interrupts of A/D Converter 435
Interrupts of Delay Interrupt Generation Module and EI^{2}OS 386
Interrupts of DTP/External Interrupt Circuit and EI^{2}OS 391
Interrupts of LIN-UART 482
Interrupts of Real-time Watch Timer and EI^{2}OS 383
Interrupts of the Time-base Timer 290

Interval Interrupt Function 290
Interval Interrupt Function of Watch Timer 293
LIN-UART Interrupt and EI^{2}OS 454
LIN-UART Interrupts and EI^{2}OS 484
Operation Flow of Hardware Interrupt 92
Precaution for Software Interrupt 100
Procedure for Using Hardware Interrupt 94
Releasing the Standby Mode by Interrupt 187
Return from Hardware Interrupt 90
Return from Software Interrupt 99
Software Interrupt Operation 100
Stack Operation after Return from Interrupt Handling 115
Stack Operation when Interrupt Handling Starts 115
Starting Hardware Interrupt 90
Starting Software Interrupt 99
Structure of Hardware Interrupt 88
Suppressing Hardware Interrupt 88
Timing of Reception Interrupt Generation and Flag Set 485
Timing of Transmission Interrupt Generation and Flag Set 487
Transition to Standby Mode and Interrupts 187
Transmission Interrupt Request Generation Timing 488
Types of Interrupts and Corresponding Functions 74
Watch Timer Interrupts and EI^{2}OS 293

Interrupt Control Register
Configuration of Interrupt Control Register (ICR) 84
Functions of Interrupt Control Registers 79, 85
Interrupt Control Registers (ICR00 to ICR15) 82
Interrupt Sources and Interrupt Vectors/Interrupt Control Registers 77
List of Interrupt Control Registers 79

Interrupt Level Mask Register
Interrupt Level Mask Register (PS:ILM) 62

Interrupt Vector
Interrupt Sources and Interrupt Vectors/Interrupt Control Registers 77
Interrupt Vector 76

Interval Interrupt
Interval Interrupt Function 290
Interval Interrupt Function of Watch Timer 293

IOA
I/O Register Address Pointer (IOA) 105

IPCP
Input Capture Register (IPCP0 to IPCP7) 307

ISCS
Extended Intelligent I/O Service (EI^{2}OS) Status Register (ISCS) 106

ISD
Configuration of Extended Intelligent I/O Service (EI^{2}OS) Descriptor (ISD) 103

L

Last Event Indication Register
Bit Configuration of Last Event Indication Register (LEIR) 541

Latch-up
Strict Observance of the Maximum Voltage Rating (for Latch-up Prevention) 32

LCD
Block Diagram of LCD Controller/Driver 593
Drive Waveform of the LCD 617
Example of LCD Panel Connection and Display Data (1/2 Duty Drive Method) 620
Example of LCD Panel Connection and Display Data (1/3 Duty Drive Method) 623
Example of LCD Panel Connection and Display Data (1/4 Duty Drive Method) 626

LCD Control Register
Bit Configuration of the Lower Bits in the LCD Control Register (LCRL) 603
Bit Configuration of the Upper Bits in the LCD Control Register (LCRH) 606

LCD Controller/Driver
Block Diagram of LCD Controller/Driver Pins 601
External Divided Resistor of LCD Controller/Driver 598
Functions of LCD Controller/Driver 592
Internal Divided Resistor of LCD Controller/Driver 596
Operation of LCD Controller/Driver 616
Pins of the LCD Controller/Driver 600
Power Supply Voltage of LCD Controller/Driver 595

LCD Controller/Driver Register
Bit Configuration of LCD Controller/Driver Register 602

LCD Output Control Register
Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2) 608
Bit Configuration of the LCD Output Control Register 3 (LOCR3) 611

LCRH
Bit Configuration of the Upper Bits in the LCD Control Register (LCRH) 606

LCRL
Bit Configuration of the Lower Bits in the LCD Control Register (LCRL) 603
LEIR
Bit Configuration of Last Event Indication Register (LEIR) 541
LIN
LIN Master/Slave Type Communication Function 515
Operation in Asynchronous LIN Mode 506
LIN Master
LIN Master Device 517
LIN Master/Slave Type Communication Function 515
LIN Slave
LIN Slave Device 518
Linear Addressing
Linear Addressing and Bank Addressing 44
LIN-UART
Block Diagram of LIN-UART 456
Block Diagram of LIN-UART Pins 461
Functions of LIN-UART 452
Interrupts of LIN-UART 482
LIN-UART Consists of the Following Blocks. 455
LIN-UART EI^{2}OS Functions 484
LIN-UART Interrupt and EI^{2}OS 454
LIN-UART Interrupts and EI^{2}OS 484
LIN-UART Pin Direct Access 509
LIN-UART Serial Mode Register (SMR) 466
List of LIN-UART Registers 462
Notes on Using LIN-UART 520
Operation of LIN-UART 496
Pins of LIN-UART 460
LIN-UART Serial Mode Register
LIN-UART Serial Mode Register (SMR) 466
LOCR
Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2) 608
Bit Configuration of the LCD Output Control Register 3 (LOCR3) 611
Low-power Consumption Control Circuit
Block Diagram of Low-power Consumption Control Circuit 164
Low-power Consumption Mode
Operation States in the Low-power Consumption Mode 184
Setting Low-power Consumption Mode 578
Low-power Consumption Mode Control Register
Access to the Low-power Consumption Mode Control Register 169
Low-power Consumption Mode Control Register (LPMCR) 167
Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode 189
Low-voltage
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
Low-voltage Detection Reset Circuit 628
Notes on Using the Low-voltage Detection Reset Circuit 635
Operation of the Low-voltage Detection Reset Circuit 634
Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637
Low-voltage Detection Reset Circuit
Low-voltage Detection Reset Circuit 628
Notes on Using the Low-voltage Detection Reset Circuit 635
Operation of the Low-voltage Detection Reset Circuit 634
Low-voltage/CPU Operation Detection Reset Circuit
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637
Low-voltage/CPU Operation Detection Reset Control Register
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
LPMCR
Low-power Consumption Mode Control Register (LPMCR) 167
Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode 189
LVRC
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632

M

Machine Clock
Machine Clock 152
Master
LIN Master Device 517

LIN Master/Slave Type Communication Function 515
Master/Slave Type Communication Function 512
Maximum Voltage
Strict Observance of the Maximum Voltage Rating (for Latch-up Prevention) 32
MB90920 Series
Overview of MB90920 Series 14
MB90F922
Basic Configuration of MB90F922 Serial Programming Connection 710
MD
Continuous Conversion Mode (ADCS: MD1, MD0=10_B) 436
Mode Pins (MD2 to MD0) 193
Single Conversion Mode (ADCS: MD1, MD0=00_B or 01_B) 436
Stop Conversion Mode (ADCS: MD1, MD0=11_B) 436
Memory Map
E^2PROM Memory Map 665
Memory Map 42
Memory Space
Memory Space 40, 673
Message Buffer
Caution for Disabling Message Buffer by BVAL Bits 587
Message Buffers 530, 563
Procedure of Reception Via Message Buffer (x) 581
Procedure of Transmission Via Message Buffer (x) 579
Specifying the Multi-level Message Buffer Configuration 583
Message Buffer Control Registers
Message Buffer Control Registers 528
Message Buffer Valid Register
Bit Configuration of Message Buffer Valid Register (BVALR) 547
Mode
At Releasing the Stop Mode 188
Bus Mode 192
Bus Mode Setting Bits 194
Clock Mode 151, 161
Continuous Conversion Mode (ADCS: MD1, MD0=10_B) 436
CPU Intermittent Operation Mode 162, 170
Event Count Mode 346
Event Count Mode (External Clock Mode) 325
Internal Clock Mode 324
Mode Data 194
Mode Fetch 128
Mode Pins 128
Mode Setting 192
Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode 189
Notes on Transition to Standby Mode 187
Operation in Asynchronous LIN Mode 506
Operation in Asynchronous Mode 498
Operation in Internal Clock Mode (Reload Mode) 342
Operation in Synchronous Mode (Operation Mode 2) 502
Operation Mode 192
Operation Mode of 16-bit Reload Timer 324
Operation of Internal Clock Mode (One Shot Mode) 344
Operation States in the Low-power Consumption Mode 184
Operation States in the Standby Mode 171
Operations and Applications of Continuous Conversion Mode 441
Operations and Applications of Single Conversion Mode 439
Operations and Applications of Stop Conversion Mode 442
Pin States in Single Chip Mode 185
Precautions for PLL Clock Mode Operation 35
Precautions When Sub Clock Mode is not Used 34
Relationship between Mode Pins and Mode Data 195
Releasing Sleep Mode 174
Releasing Stop Mode 181
Releasing the Standby Mode by Interrupt 187
Releasing Time-base Timer Mode 177
Releasing Watch Mode 179
RUN Mode 192
Sample Program for Event Count Mode 350
Sample Program for Internal Clock Mode 349
Setting Low-power Consumption Mode 578
Setup for Continuous Conversion Mode 440
Setup for Single Conversion Mode 438
Setup for Stop Conversion Mode 442
Single Conversion Mode (ADCS: MD1, MD0=00_B or 01_B) 436
Standby Mode 163
State of Pin After Reading the Mode Data 134
Stop Conversion Mode (ADCS: MD1, MD0=11_B) 436
Switching the Clock Mode 188
Transition of Clock Mode 151
Transition to Sleep Mode 173

Transition to Standby Mode and Interrupts 187
Transition to Stop Mode 180
Transition to Time-base Timer Mode 176
Transition to Watch Mode 178
Mode Data
Mode Data 194
Relationship between Mode Pins and Mode Data 195
State of Pin After Reading the Mode Data 134
Mode Pins
Mode Pins (MD2 to MD0) 193
Relationship between Mode Pins and Mode Data 195
Multi-byte
Allocating Multi-byte Data in RAM 48
Allocating Multi-byte Data on the Stack 49
Allocating Multi-byte Operand 48
Multi-level Message Buffer
Specifying the Multi-level Message Buffer Configuration 583
Multiple Interrupts
Example of Multiple Interrupts 96
Multiple Interrupts Operation 95
Multiple-Byte
Accessing Multiple-Byte Data 49
Multiplication Rate
Selection of PLL Clock Multiplication Rate 152

N

NCC
Flag Change Suppress Prefix (NCC) 70

O

One Shot Mode
Operation of Internal Clock Mode (One Shot Mode) 344
Operation Mode
Operation in Synchronous Mode (Operation Mode 2) 502
Operation Mode 192
Operation Mode of 16-bit Reload Timer 324
Oscillation Stabilization Wait
Oscillation Stabilization Wait Reset State 126
Oscillation Stabilization Wait Time
Operation During Oscillation Stabilization Wait Time 155
Oscillation Stabilization Wait Time 188
Reset Sources and Oscillation Stabilization Wait Time 125
Timer Function for the Oscillation Stabilization Wait Time 291
Oscillator
Connection of Oscillator and External Clock 156
Oscillator Clock Frequency and Serial Clock Input Frequency 712
Others
Others 704
Output Driver
Output Driver Driving Power Supply for Port 7 244
Output Driver Driving Power Supply for Port 8 250
Overview
Overview 704

P

Package Dimension
Package Dimension 18
PACSR
Program Address Detection Control Register (PACSR) 663
PADR
Program Address Detection Registers (PADR0/PADR1) 663
Panel Connection
Example of LCD Panel Connection and Display Data (1/2 Duty Drive Method) 620
Example of LCD Panel Connection and Display Data (1/3 Duty Drive Method) 623
Example of LCD Panel Connection and Display Data (1/4 Duty Drive Method) 626
PC
Program Counter (PC) 63
PCB
Bank Registers (PCB, DTB, USB, SSB, ADB) 65
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
PCNT
PPG Control Status Register (PCNT) 359
PCSR
PPG Cycle Setting Register (PCSR) 363
PDCR
PPG Down Counter Register (PDCR) 363
PDUT
PPG Duty Set Register (PDUT) 364
PIL
Input Level Select Register 0 (PIL0) 273
Input Level Selection Register 1 (PIL1) 273
Input Level Selection Register 2 (PIL2) 274

Pin Assignment
Pin Assignment 19
Pin Functions
Description of Pin Functions 20
PLL Clock
Selection of PLL Clock Multiplication Rate 152
PLL Clock Mode
Precautions for PLL Clock Mode Operation 35
PLL/Sub Clock Control Register
PLL/Sub Clock Control Register (PSCCR) 149
Port 0
Functions of Port 0 Registers 205
Operation of Port 0 206
Pin Block Diagram for Port 0 204
Port 0 Configuration 203
Port 0 Pins 203
Registers for Port 0 204
Port 1
Functions of Port 1 Registers 210
Operation of Port 1 211
Pin Block Diagram for Port 1 209
Port 1 Configuration 208
Port 1 Pins 208
Registers for Port 1 209
Port 2
Functions of Port 2 Registers 215
Operations of Port 2 216
Pin Block Diagram for Port 2 214
Port 2 Configuration 213
Port 2 Pins 213
Registers for Port 2 214
Port 3
Functions of Port 3 Registers 220
Operation of Port 3 221
Pin Block Diagram for Port 3 219
Port 3 Configuration 218
Port 3 Pins 218
Registers for Port 3 219
Port 4
Functions of Port 4 Registers 225
Operation of Port 4 226
Pin Block Diagram for Port 4 224
Port 4 Configuration 223
Port 4 Pins 223
Registers for Port 4 224
Port 5
Functions of Port 5 Registers 231
Operation of Port 5 232
Pin Block Diagram for Port 5 230
Port 5 Configuration 228
Port 5 Pins 229
Registers for Port 5 230
Port 6
Functions of Port 6 Registers 236
Operation of Port 6 237
Pin Block Diagram for Port 6 234
Port 6 Configuration 234
Port 6 Pins 234
Registers for Port 6 235
Port 7
Functions of Port 7 Registers 241
Operation of Port 7 243
Output Driver Driving Power Supply for Port 7 244
Pin Block Diagram for Port 7 240
Port 7 Configuration 239
Port 7 Pins 239
Registers for Port 7 240
Port 8
Functions of Port 8 Registers 247
Operation of Port 8 249
Output Driver Driving Power Supply for Port 8 250
Pin Block Diagram for Port 8 246
Port 8 Configuration 245
Port 8 Pins 245
Registers for Port 8 246
Port 9
Functions of Port 9 Registers 253
Operation of Port 9 254
Pin Block Diagram for Port 9 252
Port 9 Configuration 251
Port 9 Pins 251
Registers for Port 9 252
Port C
Functions of Port C Registers 259
Operation of Port C 260
Pin Block Diagram for Port C 258
Port C Configuration 256
Port C Pins 257
Registers for Port C 258
Port D
Functions of Port D Registers 265
Operation of Port D 266
Pin Block Diagram for Port D 264
Pins of Port D 263
Port D Configuration 263
Registers for Port D 264
Port E
Functions of Port E Registers 270
Operation of Port E 271
Pin Block Diagram for Port E 269
Port E Configuration 268

Port E Pins 268
Registers for Port E 269
Power Supply
Handling of Power Supply (DVcc/DVss) for High-current Output Buffer Pin 650
Handling of Power Supply for High-current Output Buffer Pins (DV_{CC}, DV_{SS}) 34
Output Driver Driving Power Supply for Port 7 244
Output Driver Driving Power Supply for Port 8 250
Power-on Sequence for A/D Converter Power Supply and Analog Input 34
Processing of A/D Converter Power Supply Pins 33
Processing of Power Supply Pins 33
Power-on
Power-on Sequence for A/D Converter Power Supply and Analog Input 34
Voltage Start Up Time at Power-on 32
PPG Control Status Register
PPG Control Status Register (PCNT) 359
PPG Cycle Setting Register
PPG Cycle Setting Register (PCSR) 363
PPG Down Counter Register
PPG Down Counter Register (PDCR) 363
PPG Duty Set Register
PPG Duty Set Register (PDUT) 364
PPG Timer
EI^2OS Functions of PPG Timer 367
Interrupt of PPG Timer 366
Interrupt of PPG Timer and EI^2OS 366
List of PPG Timer Registers 357
PPG0 Output Division Set Register
PPG0 Output Division Set Register (PPGDIV) 364
PPGDIV
PPG0 Output Division Set Register (PPGDIV) 364
Prefix
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
Common Register Bank Prefix (CMR) 69
Flag Change Suppress Prefix (NCC) 70
Prefix Codes 68
Restrictions on Prefix Codes 71
Processor Status
Bit Configuration of Processor Status (PS) 59
Program Address Detection Control Register
Program Address Detection Control Register (PACSR) 663
Program Address Detection Registers
Program Address Detection Registers (PADR0/PADR1) 663
Program Counter
Program Counter (PC) 63
Protection
Explanation of A/D Conversion Data Protection Function in 8-/10-bit A/D Converter ... 445
PS
Bit Configuration of Processor Status (PS) 59
Condition Code Register (PS:CCR) 60
Interrupt Level Mask Register (PS:ILM) 62
Register Bank Pointer (PS:RP) 61
PSCCR
PLL/Sub Clock Control Register (PSCCR) 149
Pull-up/Pull-down Resistor
Pull-up/Pull-down Resistor 34
PWM
Example of PWM Output for All "L" or All "H" 372
Notes on Changing the PWM Setting Values 650
Notes on Enabling the PWM Output 650
PWM Control Register
Bit Configuration of PWM Control Register 643
PWM1 and PWM2 Compare Registers
Bit Configuration of the PWM1 and PWM2 Compare Registers 644
PWM1 and PWM2 Select Registers
Bit Configuration of the PWM1 and PWM2 Select Registers 646

R

RAM
Allocating Multi-byte Data in RAM 48
Display RAM and Output Pins 612
RAM Area 41
RCR
Bit Configuration of Receive Complete Register (RCR) 555
RDR
Reception Data Register and Transmission Data Register (RDR/TDR) 473
Read/Reset State
Setting Flash Memory to the Read/Reset State 696
Real-time Watch Timer
Block Diagram of Real-time Watch Timer 374
Interrupt of Real-time Watch Timer 383
Interrupts of Real-time Watch Timer and EI^2OS 383

List of Registers of Real-time Watch Timer 376
Receive and Transmit Error Counters
Bit Configuration of Receive and Transmit Error Counters (RTEC) 543
Receive Complete
Receive Complete 576
Receive Complete Register
Bit Configuration of Receive Complete Register (RCR) 555
Receive Interrupt Enable Register
Bit Configuration of Receive Interrupt Enable Register (RIER) 558
Receive Message
Storing the Receive Message 574
Receive Overrun
Receive Overrun 575
Receive Overrun Register
Bit Configuration of Receive Overrun Register (ROVRR) 557
Receiving Data Frame
Processing for Receiving Data Frame and Remote Frame 575
Reception
Enabling Transmission/Reception 497
Flowchart of CAN Controller Reception 577
Flowchart of CAN Reception Setting 576
Procedure of Reception Via Message Buffer (x) 581
Reception Data Register
Reception Data Register and Transmission Data Register (RDR/TDR) 473
Reception Interrupt
Timing of Reception Interrupt Generation and Flag Set 485
Register
A/D Data Registers (ADCR0/ADCR1) 429
A/D Setting Registers (ADSR0/ADSR1) 430
Access to the Low-power Consumption Mode Control Register 169
Analog Input Enable Register (ADER6) 434
Bank Register and Access Space 46
Bit Configuration of Acceptance Mask Registers 0 and 1 (AMR0/AMR1) 561
Bit Configuration of Acceptance Mask Selection Register (AMSR) 559
Bit Configuration of Bit Timing Register (BTR) 544
Bit Configuration of Control Status Register (CSR) 536
Bit Configuration of Data Register x (x=0 to 15) (DTRx) 569
Bit Configuration of DLC Register x (x=0 to 15) (DLCRx) 568
Bit Configuration of Extended Communication Control Register (ECCR) 478
Bit Configuration of Extended Status Control Register (ESCR) 475
Bit Configuration of ID Register x (x=0 to 15) (IDRx) 564
Bit Configuration of IDE Register (IDER) 548
Bit Configuration of Last Event Indication Register (LEIR) 541
Bit Configuration of LCD Controller/Driver Register 602
Bit Configuration of Message Buffer Valid Register (BVALR) 547
Bit Configuration of PWM Control Register 643
Bit Configuration of Receive Complete Register (RCR) 555
Bit Configuration of Receive Interrupt Enable Register (RIER) 558
Bit Configuration of Receive Overrun Register (ROVRR) 557
Bit Configuration of Remote Frame Receive Wait Register (RFWTR) 551
Bit Configuration of Remote Request Receive Register (RRTRR) 556
Bit Configuration of Sub-second Data Register 380
Bit Configuration of the LCD Output Control Register 1/2 (LOCR1/LOCR2) 608
Bit Configuration of the LCD Output Control Register 3 (LOCR3) 611
Bit Configuration of the Lower Bits in the LCD Control Register (LCRL) 603
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
Bit Configuration of the Time-base Timer Control Register (TBTC) 283
Bit Configuration of the Upper Bits in the LCD Control Register (LCRH) 606
Bit Configuration of the Watchdog Timer Control Register (WDTC) 281
Bit Configuration of Timer Control Register 378
Bit Configuration of Transmission Cancel Register (TCANR) 552
Bit Configuration of Transmission Complete Register (TCR) 553
Bit Configuration of Transmission Interrupt Enable Register (TIER) 554
Bit Configuration of Transmission Request Register (TREQR) 549

Bit Configuration of Transmission RTR Register (TRTRR) 550
Clock Selection Register (CKSCR) 145
Compare Clear Register (CPCLR) 314
Condition Code Register (PS:CCR) 60
Configuration of Interrupt Control Register (ICR) 84
Direct Page Register (DPR) 64
Extended Intelligent I/O Service (EI^2OS) Status Register (ISCS) 106
External Interrupt Level Setting Register (ELVRH/ELVRL) 399
External Interrupt Source Register (EIRR) 397
Flash Memory Control Status Register (FMCS) 680
Functions of Port 5 Registers 231
Input Capture Control Status Register (ICS01/23/45/67) 307
Input Capture Edge Register (ICE) 310
Input Capture Register (IPCP0 to IPCP7) 307
Input Level Select Register 0 (PIL0) 273
Input Level Selection Register 1 (PIL1) 273
Input Level Selection Register 2 (PIL2) 274
Interrupt Level Mask Register (PS:ILM) 62
LIN-UART Serial Mode Register (SMR) 466
Lower Bits in the A/D Control Status Register (ADCS0) 427
Low-power Consumption Mode Control Register (LPMCR) 167
PLL/Sub Clock Control Register (PSCCR) 149
PPG Control Status Register (PCNT) 359
PPG Cycle Setting Register (PCSR) 363
PPG Down Counter Register (PDCR) 363
PPG Duty Set Register (PDUT) 364
Program Address Detection Control Register (PACSR) 663
Reception Data Register and Transmission Data Register (RDR/TDR) 473
ROM Mirror Function Select Register (ROMM) 673
Sample Setting of the Bit Timing Register 546
Sample Setting of the ID Register 565
Serial Control Register (SCR) 463
Serial Status Register (SSR) 470
Timer Control Status Register (TCCSH, TCCSL) 315
Timer Data Register (TCDT) 314
Transmission Data Register (TDR) 474
Upper Bits in the A/D Control Status Register (ADCS1) 423
Watch Timer Control Register (WTC) 285

Register Bank
General-purpose Register Area and Register Bank Pointer 61
Register Bank 67

Register Bank Pointer
General-purpose Register Area and Register Bank Pointer 61
Register Bank Pointer (PS:RP) 61

Reload Counter
Function of Reload Counter 494
Operation of the Reload Counter 493

Reload Mode
Operation in Internal Clock Mode (Reload Mode) 342

Reload Timer
16-bit Reload Timer Settings 340
Block Diagram of 16-bit Reload Timer 326
Block Diagram of Pins for 16-bit Reload Timer 329
EI^2OS Function of 16-bit Reload Timer 339
Interrupts and EI^2OS of 16-bit Reload Timer 325, 339
Interrupts of 16-bit Reload Timer 339
Notes on Using 16-bit Reload Timer 348
Operation Mode of 16-bit Reload Timer 324
Pins of 16-bit Reload Timer 328
Register List of 16-bit Reload Timer 330

Reload Value
Reload Value and Baud Rate for Each Clock Speed 492

Remote Frame
Processing for Receiving Data Frame and Remote Frame 575

Remote Frame Receive Wait Register
Bit Configuration of Remote Frame Receive Wait Register (RFWTR) 551

Remote Request Receive Register
Bit Configuration of Remote Request Receive Register (RRTRR) 556

Reset
Bit Configuration of the Low-voltage/CPU Operation Detection Reset Control Register (LVRC) 632
Block Diagram of External Reset Pin 127
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
Checking Reset Sources 288
Correspondence Between Reset Source Bit and Reset Source 131
CPU Operation Detection Reset Circuit 629
Low-voltage Detection Reset Circuit 628

Notes on the Reset Source Bit 133
Notes on Using the CPU Operation Detection Reset Circuit 635
Notes on Using the Low-voltage Detection Reset Circuit 635
Operation of the CPU Operation Detection Reset Circuit 634
Operation of the Low-voltage Detection Reset Circuit 634
Oscillation Stabilization Wait Reset State 126
Outline of Reset Operation 128
Prohibiting Watchdog Timer Reset 288
Reset Output Function 135
Reset Source Bit 130
Reset Sources 122
Reset Sources and Oscillation Stabilization Wait Time 125
Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637
Setting Flash Memory to the Read/Reset State 696
State of Pins During Reset 134
State of Reset Source Bits 132

Reset Circuit
Block Diagram of the Low-voltage/CPU Operation Detection Reset Circuit 630
CPU Operation Detection Reset Circuit 629
Low-voltage Detection Reset Circuit 628
Notes on Using the CPU Operation Detection Reset Circuit 635
Notes on Using the Low-voltage Detection Reset Circuit 635
Operation of the CPU Operation Detection Reset Circuit 634
Operation of the Low-voltage Detection Reset Circuit 634
Sample Program for the Low-voltage/CPU Operation Detection Reset Circuit 637

Reset Source Bit
Correspondence Between Reset Source Bit and Reset Source 131
Notes on the Reset Source Bit 133
Reset Source Bit 130
State of Reset Source Bits 132

Restart
When Disabling Restart 369, 371
When Enabling Restart 369, 371

RFWTR
Bit Configuration of Remote Frame Receive Wait Register (RFWTR) 551

RIER
Bit Configuration of Receive Interrupt Enable Register (RIER) 558

ROM
Overview of ROM Security Function 722
ROM Area 41

ROM Mirror Function Select Module
Block Diagram of the ROM Mirror Function Select Module 672
Register of the ROM Mirror Function Select Module 672

ROM Mirror Function Select Register
ROM Mirror Function Select Register (ROMM) 673

ROM Security
Overview of ROM Security Function 722

ROMM
ROM Mirror Function Select Register (ROMM) 673

ROVRR
Bit Configuration of Receive Overrun Register (ROVRR) 557

RP
Register Bank Pointer (PS:RP) 61

RRTRR
Bit Configuration of Remote Request Receive Register (RRTRR) 556

RTEC
Bit Configuration of Receive and Transmit Error Counters (RTEC) 543

RUN Mode
RUN Mode 192

S

Sampling Time
Setup for Sampling Time (Bits ST2 to ST0) 433

SCR
Serial Control Register (SCR) 463

Second/Minute/Hour/Day Data Registers
Bit Configuration of Second/Minute/Hour/Day Data Registers 381

Sector Configuration
Sector Configuration 677

Sector Erase
Data Write and Chip/Sector Erase 692, 693
Restarting Sector Erase of Flash Memory 703
Sector Erase 690, 694
Sector Erase Suspended 690, 692, 694

Sector Erase Timer Flag
Transition of State of Sector Erase Timer Flag (DQ3) 694
Security
Flash Security Function 35
How to Apply Security 704
How to Release the Security 704
Operation with the Security Enabled 704
Overview of ROM Security Function 722
Serial Clock
Oscillator Clock Frequency and Serial Clock Input Frequency 712
Serial Control Register
Serial Control Register (SCR) 463
Serial On-board Writing
Pins Used for Cypress Standard Serial On-board Writing 711
Serial Programming Connection
Basic Configuration of MB90F922 Serial Programming Connection 710
Examples of Serial Programming Connections 713
Serial Status Register
Serial Status Register (SSR) 470
SGAR
Amplitude Data Registers (SGAR0/SGAR1) 658
SGCRH
Bit Configuration of Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1) 655
SGCRL
Bit Configuration of Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1) 655
SGDR
Decrement Grade Registers (SGDR0/SGDR1) 659
SGFR
Frequency Data Registers (SGFR0/SGFR1) 657
SGTR
Tone Count Registers (SGTR0/SGTR1) 660
Single Chip Mode
Pin States in Single Chip Mode 185
Single Conversion Mode
Operations and Applications of Single Conversion Mode 439
Setup for Single Conversion Mode 438
Single Conversion Mode (ADCS: MD1, MD0=00_B or 01_B) 436
Slave
LIN Master/Slave Type Communication Function 515
LIN Slave Device 518
Master/Slave Type Communication Function 512
Sleep Mode
Releasing Sleep Mode 174
Transition to Sleep Mode 173
SMR
LIN-UART Serial Mode Register (SMR) 466
Software Interrupt
Precaution for Software Interrupt 100
Return from Software Interrupt 99
Software Interrupt Operation 100
Starting Software Interrupt 99
Sound Control Register
Bit Configuration of Sound Control Register (SGCRH0/SGCRH1, SGCRL0/SGCRL1) 655
Sound Generator
Block Diagram of the Sound Generator 652
Registers of the Sound Generator 653
Source
Checking Reset Sources 288
Correspondence Between Reset Source Bit and Reset Source 131
Interrupt Sources and Interrupt Vectors/Interrupt Control Registers 77
Notes on the Reset Source Bit 133
Reset Source Bit 130
Reset Sources 122
Reset Sources and Oscillation Stabilization Wait Time 125
State of Reset Source Bits 132
SPB
Bank Select Prefix (PCB, DTB, ADB, SPB) 68
SSB
Bank Registers (PCB, DTB, USB, SSB, ADB) ... 65
SSP
System Stack Pointer (SSP) 58
SSR
Serial Status Register (SSR) 470
ST
Setup for Sampling Time (Bits ST2 to ST0) 433
Stabilization
Stabilization of Supplied Voltage 32
Stack
Allocating Multi-byte Data on the Stack 49
Stack Area 116
Stack Operation after Return from Interrupt Handling 115

Stack Operation when Interrupt Handling Starts 115
Stack Selection 57
Standby Mode
Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode 189
Notes on Transition to Standby Mode 187
Operation States in the Standby Mode 171
Releasing the Standby Mode by Interrupt 187
Standby Mode 163
Transition to Standby Mode and Interrupts 187
Start Up Time
Voltage Start Up Time at Power-on 32
State Transition Diagram
State Transition Diagram 183
Status Register
Extended Intelligent I/O Service (EI^2OS) Status Register (ISCS) 106
Stepping Motor Controller
Block Diagram of the Stepping Motor Controller 640
Registers of the Stepping Motor Controller 641
Settings for Stepping Motor Controller Operation 648
Stop Conversion Mode
Operations and Applications of Stop Conversion Mode 442
Setup for Stop Conversion Mode 442
Stop Conversion Mode (ADCS: MD1, MD0=11_B) 436
Stop Mode
At Releasing the Stop Mode 188
Releasing Stop Mode 181
Transition to Stop Mode 180
Structure
Structure of Instruction Map 824
Sub Clock Mode
Precautions When Sub Clock Mode is not Used 34
Sub-second Data Register
Bit Configuration of Sub-second Data Register 380
Supplied Voltage
Stabilization of Supplied Voltage 32
Synchronous Method
Synchronous Method 497
Synchronous Mode
Operation in Synchronous Mode (Operation Mode 2) 502
System Configuration
System Configuration of Flash Microcontroller Programmer 713
System Stack Pointer
System Stack Pointer (SSP) 58

T

TBTC
Bit Configuration of the Time-base Timer Control Register (TBTC) 283
TCANR
Bit Configuration of Transmission Cancel Register (TCANR) 552
TCCSH, TCCSL
Timer Control Status Register (TCCSH, TCCSL) 315
TCDT
Timer Data Register (TCDT) 314
TCR
Bit Configuration of Transmission Complete Register (TCR) 553
TDR
Reception Data Register and Transmission Data Register (RDR/TDR) 473
Transmission Data Register (TDR) 474
TIER
Bit Configuration of Transmission Interrupt Enable Register (TIER) 554
Time-base Timer
Functions of Time-base Timer 278
Interrupts and EI^2OS of Time-base Timer 291
Interrupts of the Time-base Timer 290
Notes on Using the Time-base Timer 294
Operation of Time-base Timer 290, 295
Program Example for Time-base Timer 298
Time-base Timer Control Register
Bit Configuration of the Time-base Timer Control Register (TBTC) 283
Time-base Timer Mode
Releasing Time-base Timer Mode 177
Transition to Time-base Timer Mode 176
Timer Control Register
Bit Configuration of Timer Control Register 378
Timer Control Status Register
Timer Control Status Register (TCCSH, TCCSL) 315
Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L) 335
Timer Control Status Registers, Upper (TMCSR0H to TMCSR3H) 332

Timer Data Register
Timer Data Register (TCDT) 314
Timing Limit Exceeded Flag
Transition of State of Timing Limit Exceeded Flag (DQ5) 693
TMCSR
Timer Control Status Registers, Lower (TMCSR0L to TMCSR3L) 335
Timer Control Status Registers, Upper (TMCSR0H to TMCSR3H) 332
TMR
16-bit Timer Registers (TMR0 to TMR3) 337
TMRLR
16-bit Reload Registers (TMRLR0 to TMRLR3) 338
Toggle Bit Flag
Toggle Bit Flag (DQ6) State Transition 692
Tone Count Registers
Tone Count Registers (SGTR0/SGTR1) 660
Transition
Notes on Accessing to the Low-power Consumption Mode Control Register (LPMCR) for the Transition to the Standby Mode 189
Notes on Transition to Standby Mode 187
Transition of Clock Mode 151
Transition to Sleep Mode 173
Transition to Standby Mode and Interrupts 187
Transition to Stop Mode 180
Transition to Time-base Timer Mode 176
Transition to Watch Mode 178
Transmission
Clearing a CAN Controller Transmission Request 571
Completing CAN Controller Transmission 571
Enabling Transmission/Reception 497
Flowchart of CAN Controller Transmission 573
Flowchart of CAN Transmission Setting 572
Procedure of Transmission Via Message Buffer (x) 579
Starting CAN Controller Transmission 571
Transmission Cancel Register
Bit Configuration of Transmission Cancel Register (TCANR) 552
Transmission Complete Register
Bit Configuration of Transmission Complete Register (TCR) 553
Transmission Data Register
Reception Data Register and Transmission Data Register (RDR/TDR) 473
Transmission Data Register (TDR) 474
Transmission Interrupt
Timing of Transmission Interrupt Generation and Flag Set 487
Transmission Interrupt Request Generation Timing 488
Transmission Interrupt Enable Register
Bit Configuration of Transmission Interrupt Enable Register (TIER) 554
Transmission Request Register
Bit Configuration of Transmission Request Register (TREQR) 549
Transmission RTR Register
Bit Configuration of Transmission RTR Register (TRTRR) 550
TREQR
Bit Configuration of Transmission Request Register (TREQR) 549
TRTRR
Bit Configuration of Transmission RTR Register (TRTRR) 550

U

UART
UART Baud Rate Selection 489
Undefined Instruction
Exception Handling Interrupt by Execution of Undefined Instruction 114
Unused Pins
Processing of Unused Pins 32
USB
Bank Registers (PCB, DTB, USB, SSB, ADB) ... 65
User Stack Pointer
User Stack Pointer (USP) 58
USP
User Stack Pointer (USP) 58

V

Voltage
Voltage Start Up Time at Power-on 32

W

Watch Mode
Releasing Watch Mode 179
Transition to Watch Mode 178
Watch Timer
Interval Interrupt Function of Watch Timer 293
Operation of Watch Timer 293
Watch Timer Function 278
Watch Timer Interrupts and EI^{2}OS 293

Watch Timer Control Register
Watch Timer Control Register (WTC) 285
Watchdog Timer
Block Diagram of Watchdog Timer/Time-base Timer/ Watch Timer 279
Clearing the Watchdog Timer 288
Functions of the Watchdog Timer 278
Interval Time of the Watchdog Timer 288
List of Registers of Watchdog Timer/Time-base Timer/ Watch Timer 280
Method of Starting the Watchdog Timer 288
Notes on Using the Watchdog Timer 294
Operation of Watchdog Timer/Time-base Timer/Watch Timer 287
Program Example for Watchdog Timer 297
Prohibiting Watchdog Timer Reset 288
Watchdog Timer Control Register
Bit Configuration of the Watchdog Timer Control Register (WDTC) 281
Waveform
1/2 Bias, 1/2 Duty Output Waveform 618
1/3 Bias, 1/3 Duty Output Waveform 621
1/3 Bias, 1/4 Duty Output Waveform 624
Drive Waveform of the LCD 617
WDTC
Bit Configuration of the Watchdog Timer Control Register (WDTC) 281
WE
Note on Setting the FMCS:WE Bit 687
Write
How to Write/Erase Data Flash Memory 676
Write Data to the Flash Memory 697
WTC
Watch Timer Control Register (WTC) 285

Revision History

Revision History

Document Title: 16-Bit Microcontroller F^{2}MC-16LX Family MB90920 Series Hardware Manual Document Number: 002-06975				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
**	-	03/03/2010	TROS	Migrated Spansion CM44-10142-5E to Cypress and assigned document number 002-06975. No change to document contents or format.
*A	5772096	06/14/2017	SSAS	Migrated to Cypress template